statinfer

Machine Learning with AzureML

statinfer.com

Contents

statinfer

Part 1/12 - AzureML for Data Science

Contents

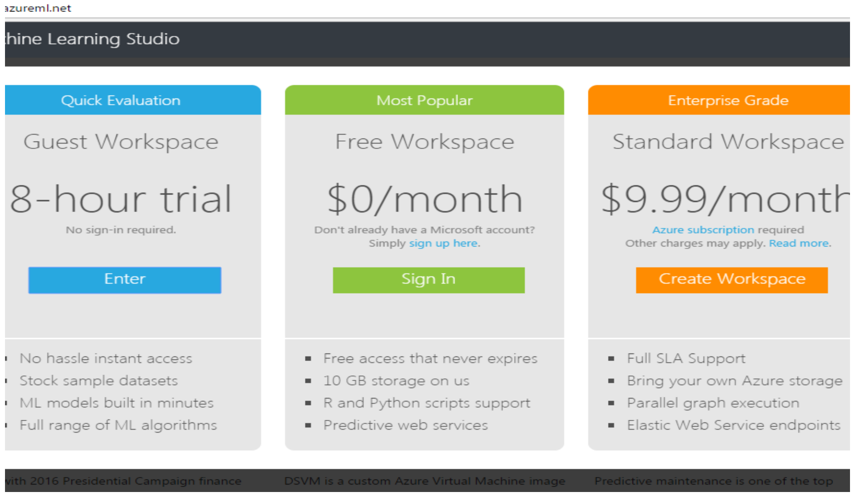
Introduction

- Azure Machine Learning Studio is an interactive workspace to create a predictive analysis model in cloud
- Creating experiments becomes easy Azure Machine Learning Studio
- We can drag and drop the datasets, modules, etc.. Into the canvas
- Creating proper connection between elements inside the canvas and running it completes the experiment
- Experiments can be iterated until expected result is obtained
- Once finished the modal can be published as a web service so that It can be accessed by others

Creating a Login

- Enter the URL https://studio.azureml.net
- Click on 'sign up'
- A window with three types of workspace will appear(Guest, Free, Standard) →
- In 'Free Workspace' Sign in with Microsoft account or if you don't have Microsoft account click 'Sign up here'
- Fill up the credentials and get a Microsoft account and sign in with that account
- The Azure ML studio will be like →

Fig1: Types of Workspaces



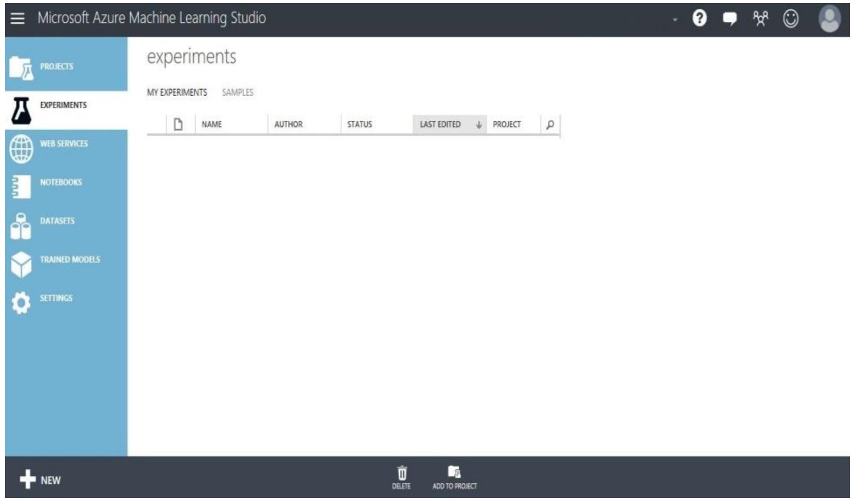


Fig2: Azure ML studio

Getting started with Azure

- Click the upper-left menu
- Menu contains

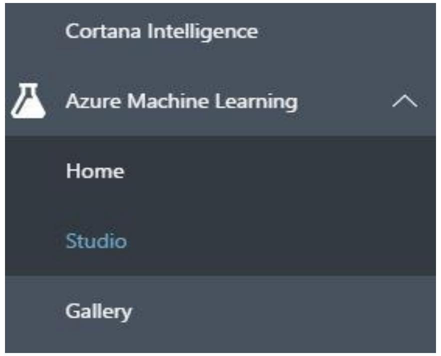


Fig3: Upper-left menu

- **Cortana Intelligence**
  - This takes you to Cortana Intelligence Suite which is a fully managed ‘big data and advanced analytics suite’ to transform data into intelligent action
- **Azure Machine Learning**
  - Home
    - This page contains documentation, videos, webinars, etc..

Getting started with Azure cont..

- **Studio:**
  - PROJECTS – Combining related items such as experiments, notebook, datasets as a group becomes a project
  - EXPERIMENTS – It has the experiments that we create and run or saved as drafts
  - WEB SERVICES – this contains the web services that we have deployed from our experiments
  - NOTEBOOKS - Jupyter notebooks that we have created
  - DATASETS - Datasets that we have uploaded into Studio
  - TRAINED MODELS - Models that are trained for prediction gets saved into this
  - SETTINGS - used to configure your account and resources
- **Gallery**
  - Contains solutions created using components of the Cortana Intelligence Suite,by various data scientist and developers

Data Importing in Azure

- Click on the Datasets icon in the left pane, The Datasets page appears
- Click the New button which is at bottom left corner
- The new window appears, select ‘From Local File’
- ‘Upload a new Dataset’ window appears
- In ‘select a file to upload’ field click on ‘choose file’ and locate the file in the system and click add.
- Here we import Sales_by_country_v1.csv file
- The description field is optional, where we can specify a short description about the Dataset
- Once finished click on the button and the Dataset gets uploaded(this may take some time based on the size of the file)

Data Importing in Azure cont..

Fig4:Datasets page

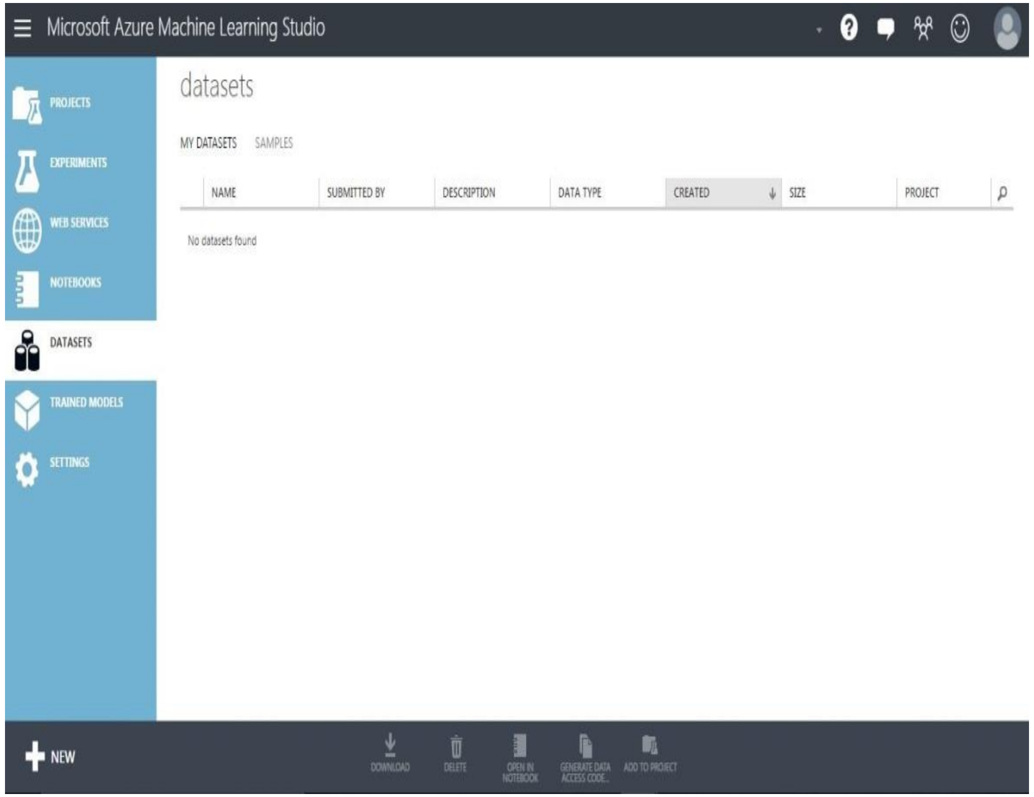
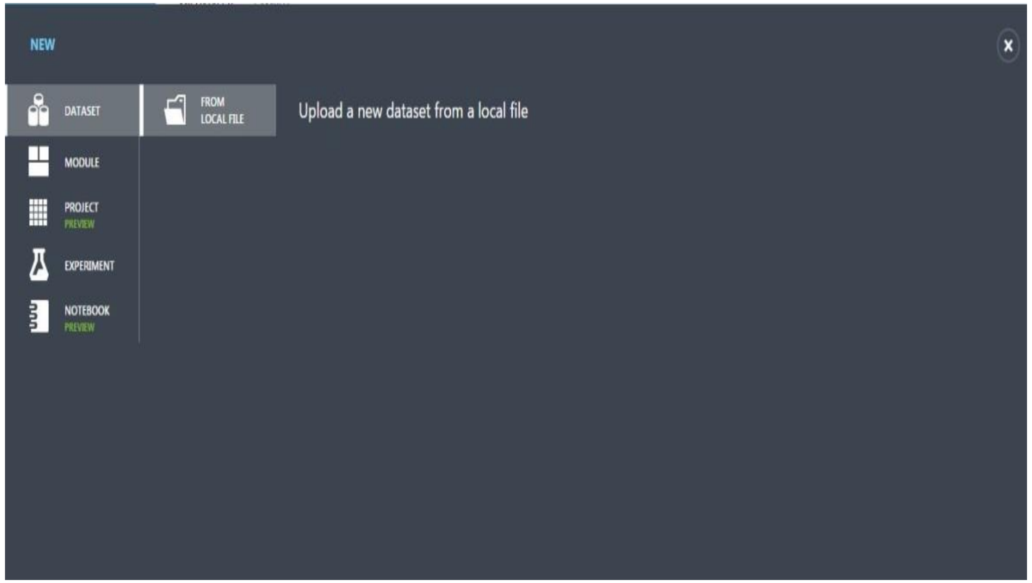


Fig5:New window



Data Importing in Azure cont..

Fig6: Upload dataset window

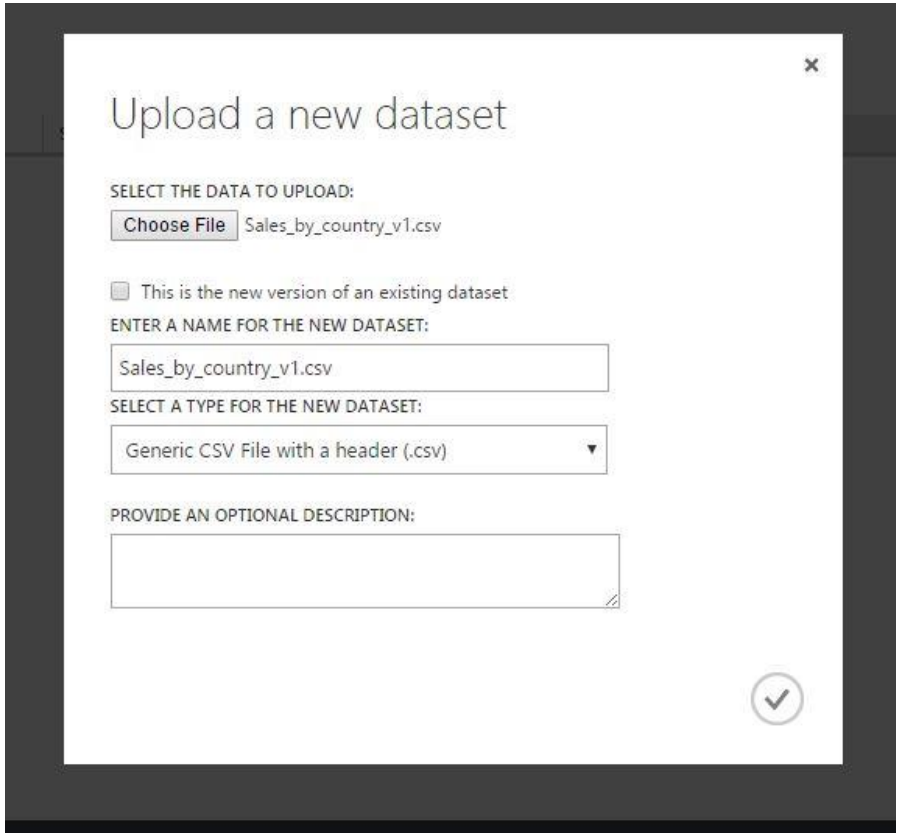
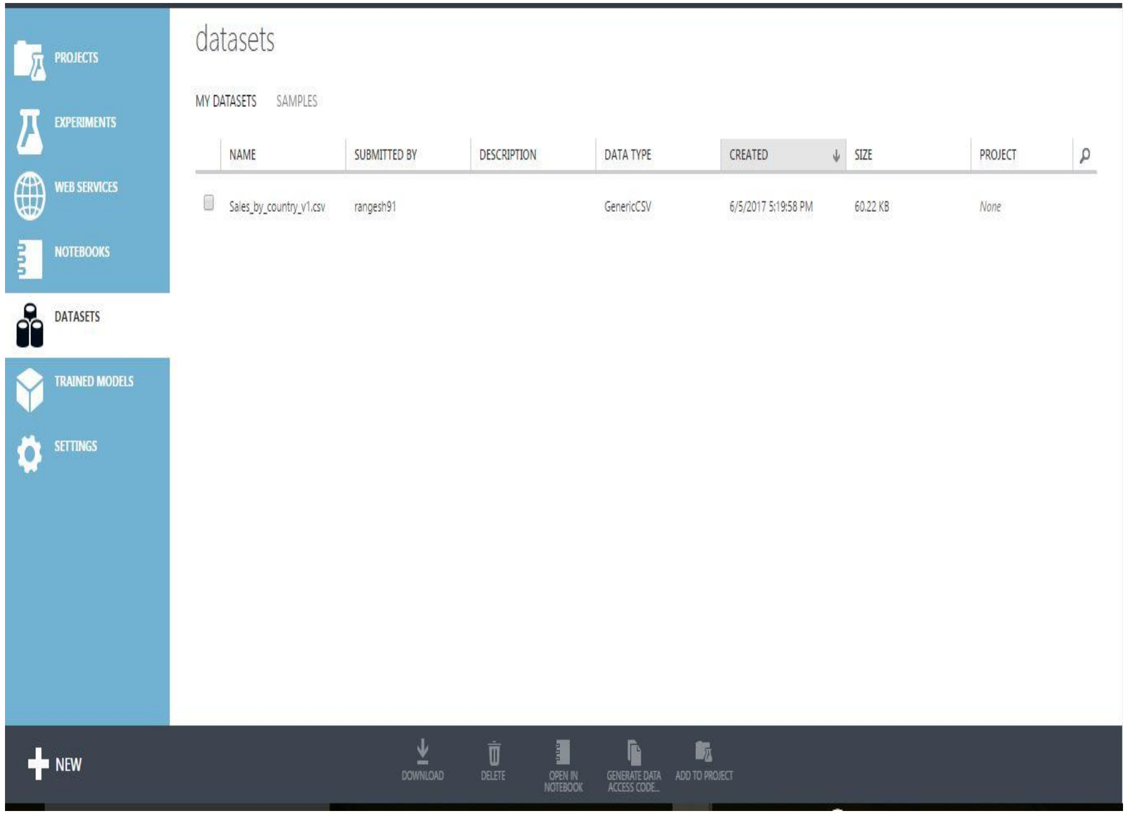


Fig7: Dataset added to Datasets page



Creating and Saving Experiment

- Creating an experiment to find sum in numerical columns:
  - Click on the Experiments in the left pane
  - In experiment window click the 

 button
  - Select blank experiment in the new window
  - New blank experiment is created, change the name that appears on the top of the canvas
  - Select Saved datasets from the left pane of canvas, it contains "My Datasets and Sample"
  - Select My Datasets ant it lists the datasets we have imported into studio
  - Select Sales_by_country_v1.csv , drag and drop into the canvas
  - Click on the output circle of Sales_by_country_v1.csv in canvas and select Visulize
  - A new window showing the basic statistics and visualizations will appear

Creating and Saving Experiment

- Search ‘Compute Elementary Statistics’ in left pane of canvas, and drag it to the canvas
- Connect the output circle of dataset to the input of Compute Elementary Statistics
- Click on Compute Elementary Statistics, in properties window select sum in method
- Click on launch column selector in properties window
- In select column window, Include → Column Name → unitSold (name of the column)
- Click on 
- Click on save 

 in the bottom pane to save the experiment
- Click on Run 

- Once finished running, right click on the output circle of Compute Elementary Statistics and click on visualize
- We can see the sum on unitsSold column

Creating and Saving Experiment

Fig8: New Experiment

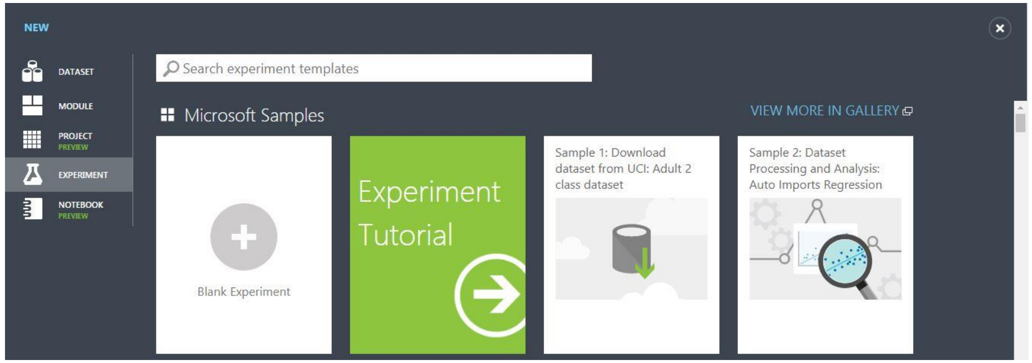
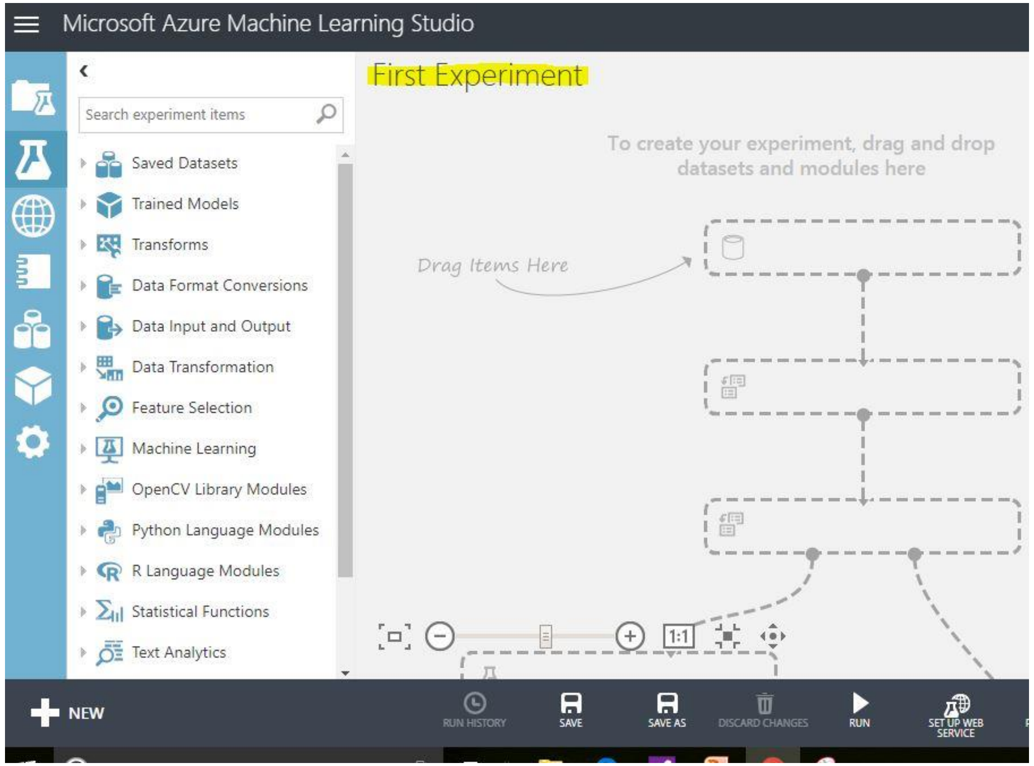


Fig9: Change the name of the Experiment



Creating and Saving Experiment

Fig10: Selecting dataset

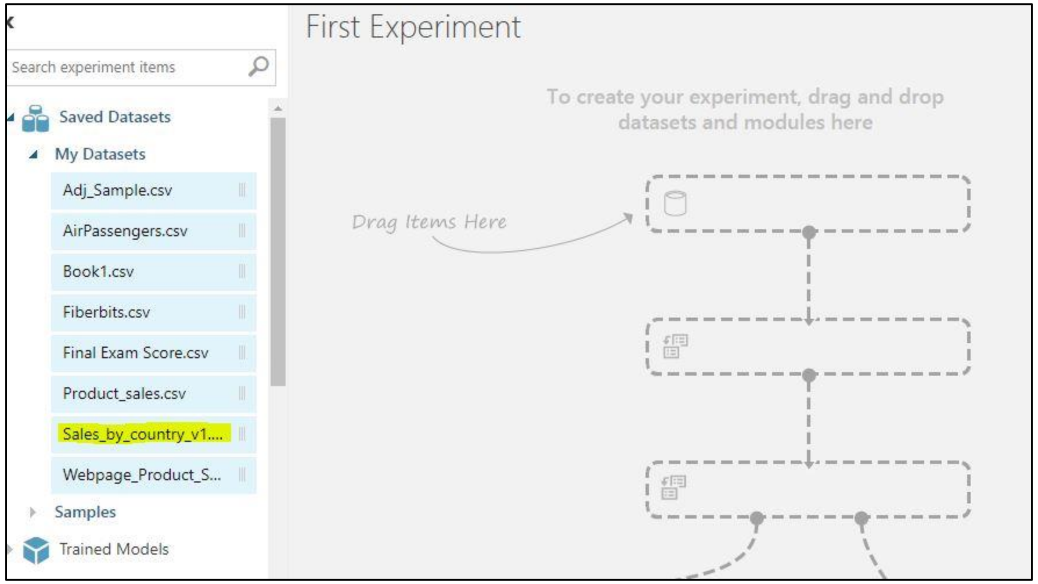
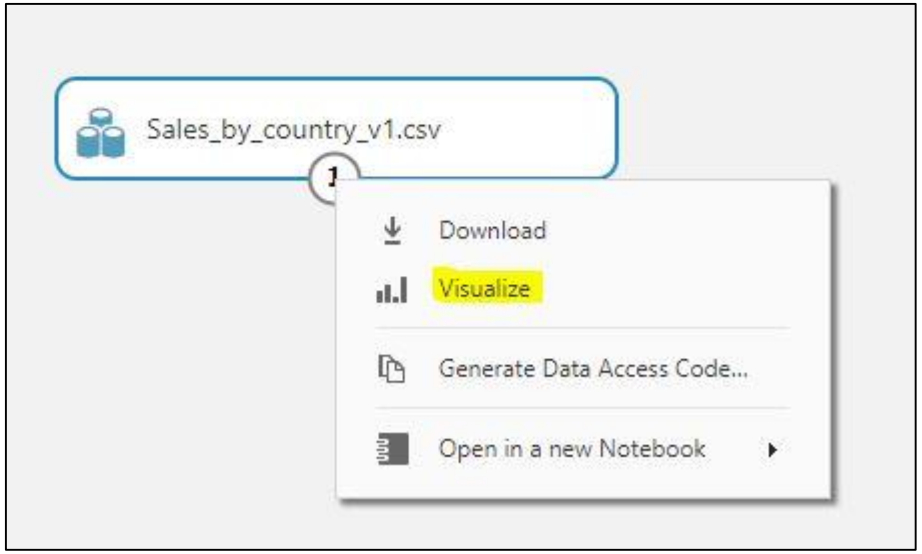
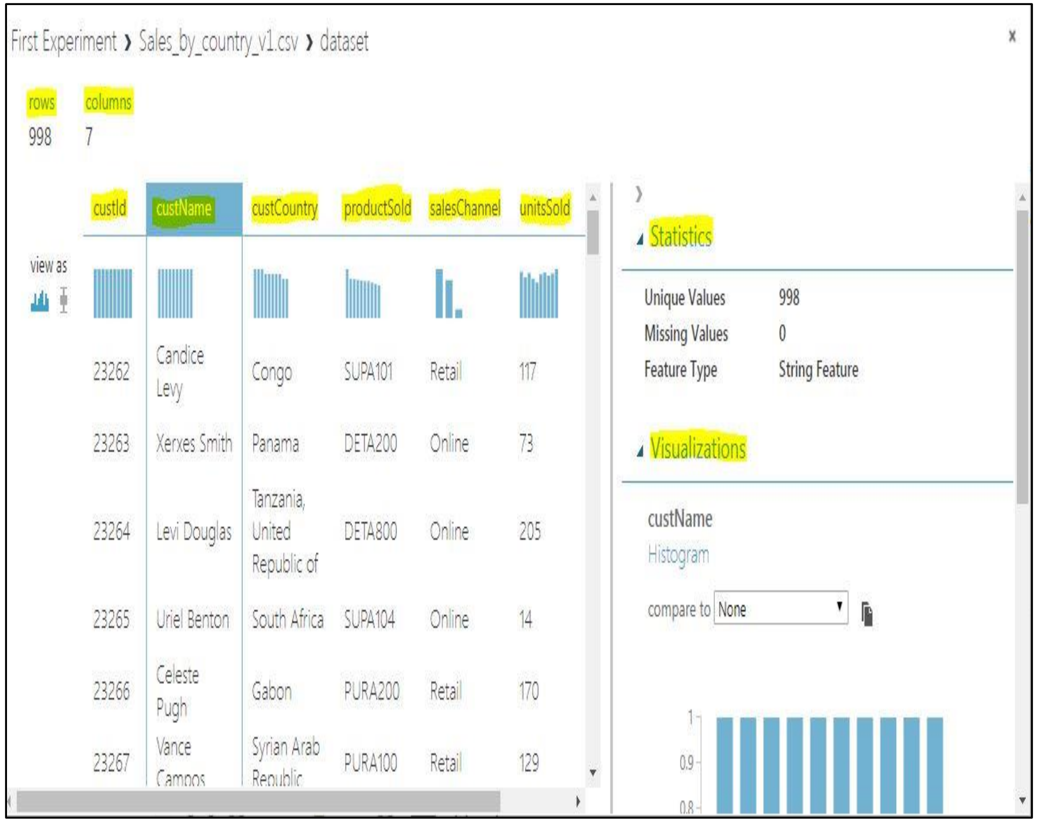


Fig11: To visualize dataset

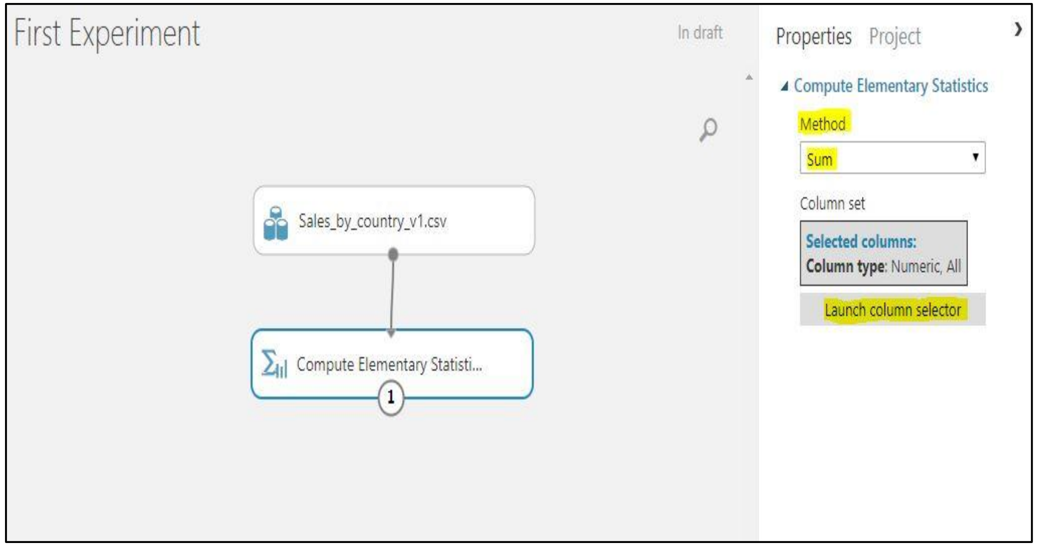


Creating and Saving Experiment

Fig12: Visulizing the dataset



Fib13: Compute Elementary Statistics



Creating and Saving Experiment

Fig14: Selecting Columns

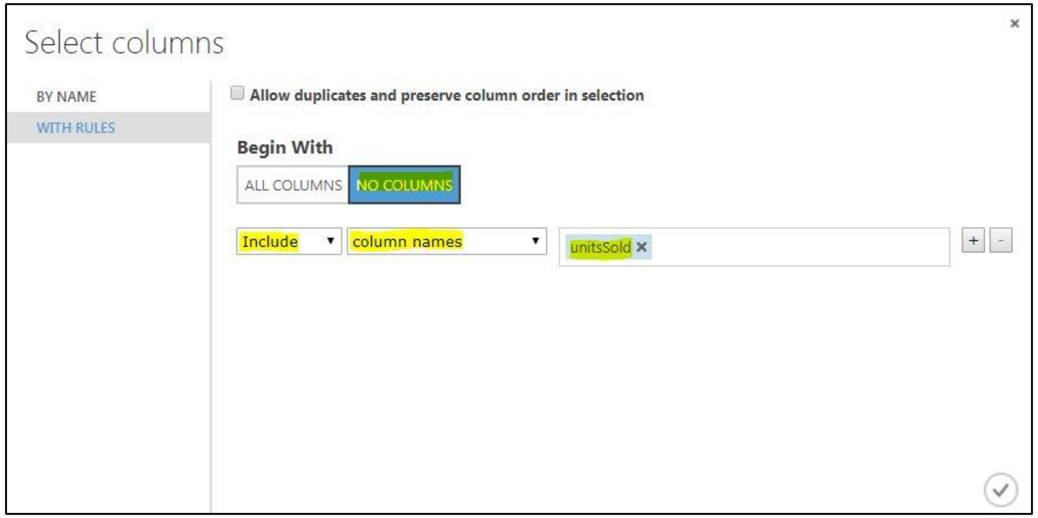
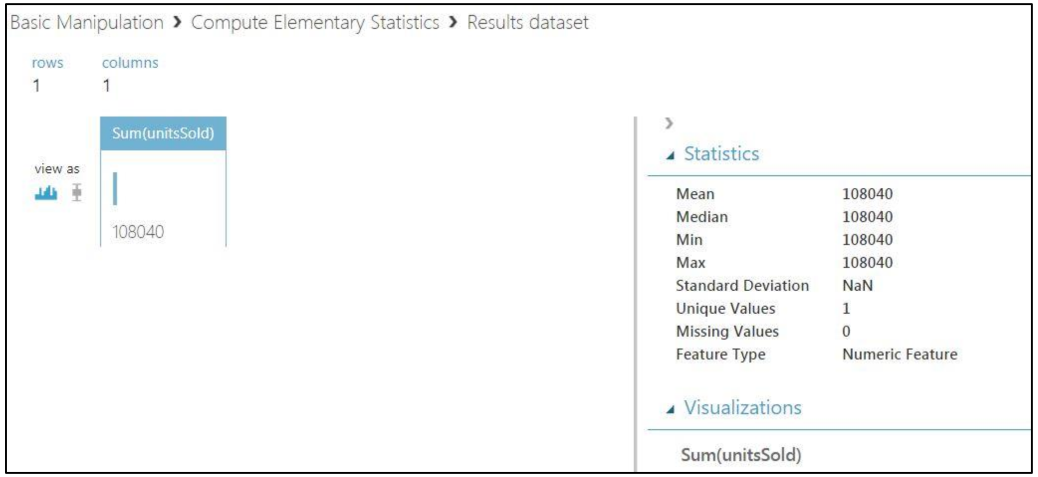


Fig15: Visualizing the sum



Loading Experiment

- Click on experiments in the left pane
- Click on the experiment name with which we saved
- This loads the experiment

Fig16: Selecting experiment

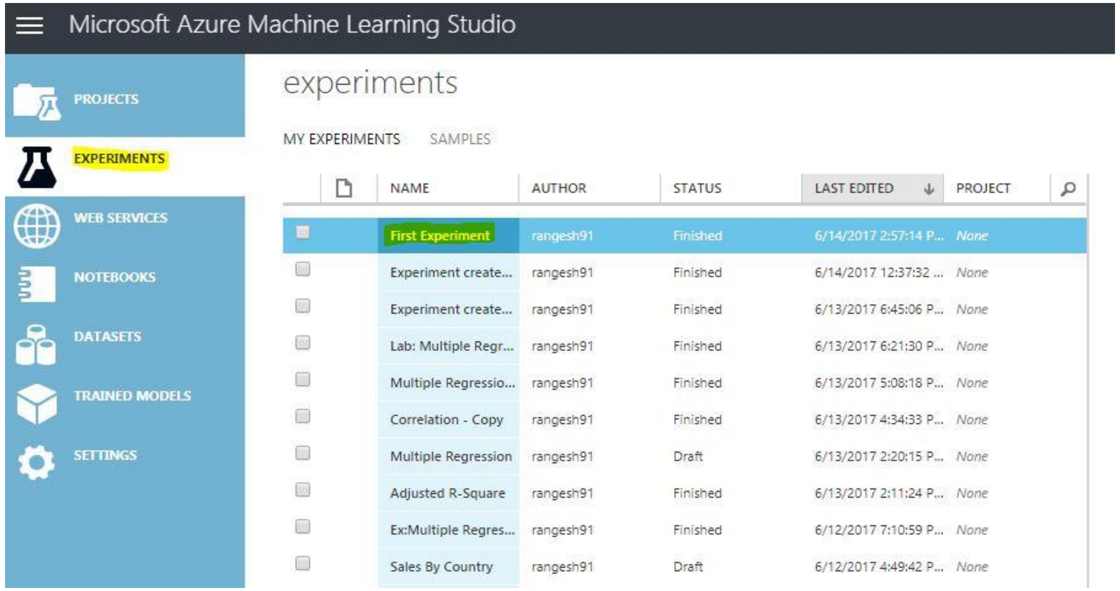
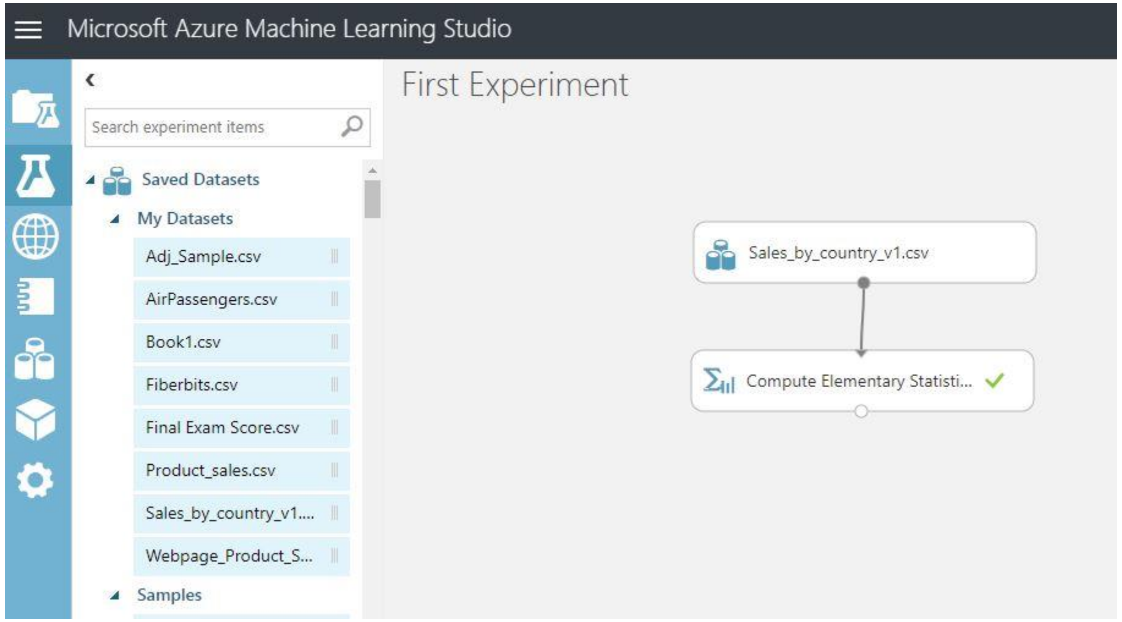


Fig17: Loaded experiment



Thank you

statinfer

We offer training on

www.statinfer.com

- Data Analytics
- Data Visualization
- Predictive Modelling
- Data Science
- Machine Learning
- Deep Learning
- R
- Python
- TensorFlow

statinfer

Part 2/12 - Data Manipulations on Azure

statinfer.com

Contents

- Sub setting the data
  - Sub setting Columns
  - Sub setting Rows with R-script
- Splitting data
- Calculated fields
- Sorting with R-script
- Removing duplicate values
- Joining datasets

Sub setting the data

- Sub setting is that we are taking out a part of data from the dataset to have a closer look in to it
- Sub setting can be done in columns, rows or both
- As of now for columns it is available azure where as for rows we use R-script
- Import the dataset: ~/World Bank Data/GDP.csv

Steps - Sub setting the data (Columns)

- Click on Experiments, drag and drop the GDP.csv dataset
- Search for ‘select columns from the dataset’ tile, drop it into the canvas.
- Connect dataset to ‘select columns from the dataset’ tile
- Click ‘select columns from the dataset’ tile, and look for ‘Launch column selector’ in the prorerties window(right side)
- Select columns dialog box appears, left pane of that has two options, ‘By Name’ and ‘With Rules’
- By Name – will list the set of column names in the dataset
- With Rules – in this we can select columns by names, indices and type(ie. Numeric, String, Integer)

Steps - Sub setting the data (Columns)

- Select ‘By Names’ select the columns and click the tick button
- Click on Run
- Once finished running, select the output circle of ‘select columns from the dataset’ tile and click visualize
- We can see the data with the selected columns

Steps - Sub setting the data (Columns)

Fig1: Launch Column Selector

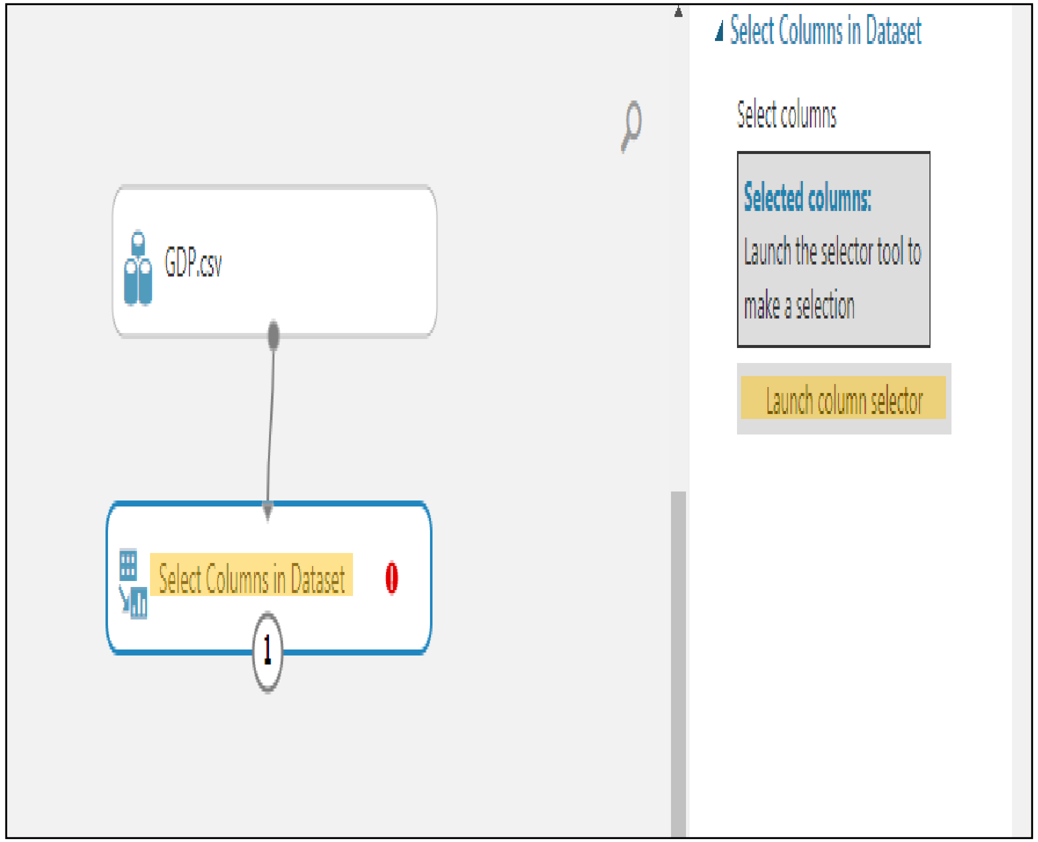
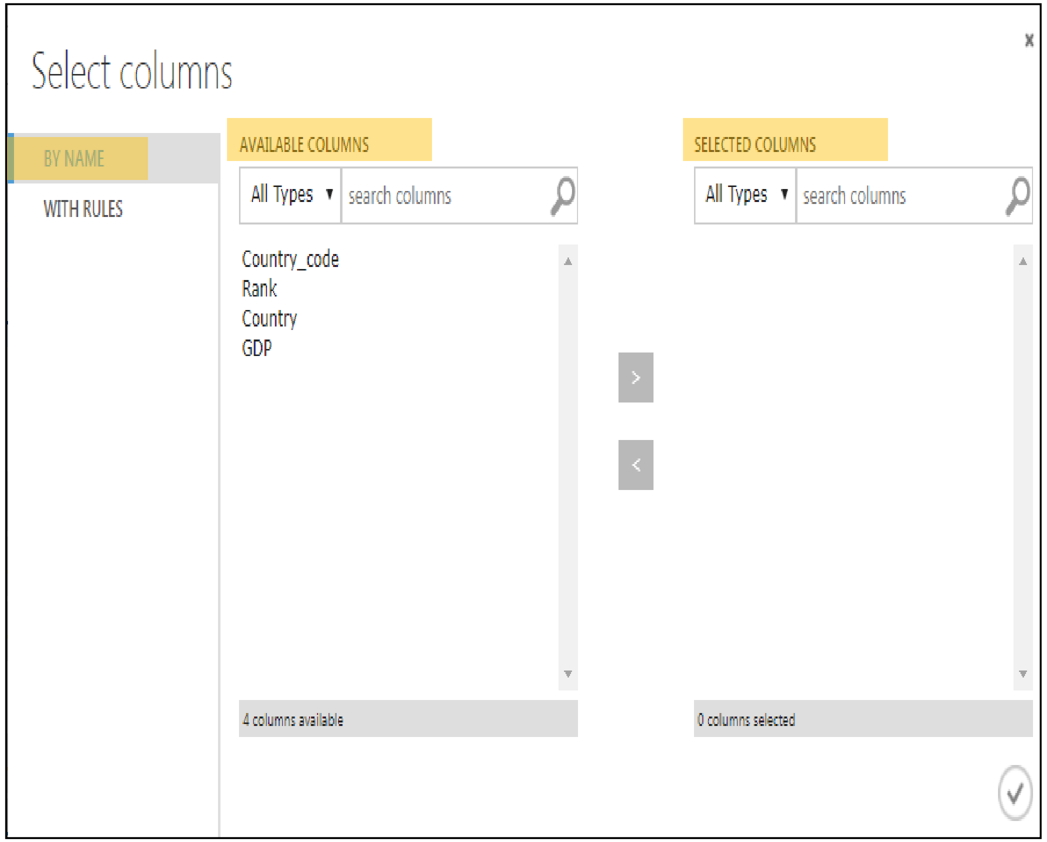


Fig2: Select Columns



Steps - Sub setting the data (Columns)

Fig3: Select Visualize

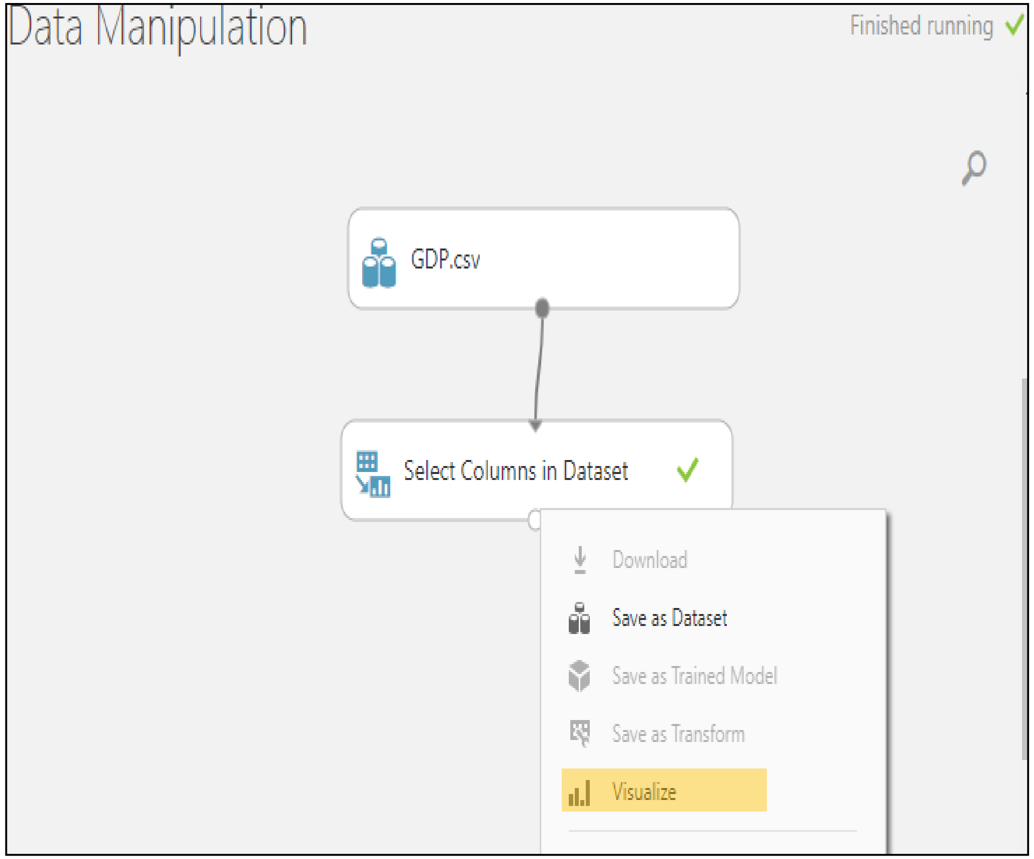
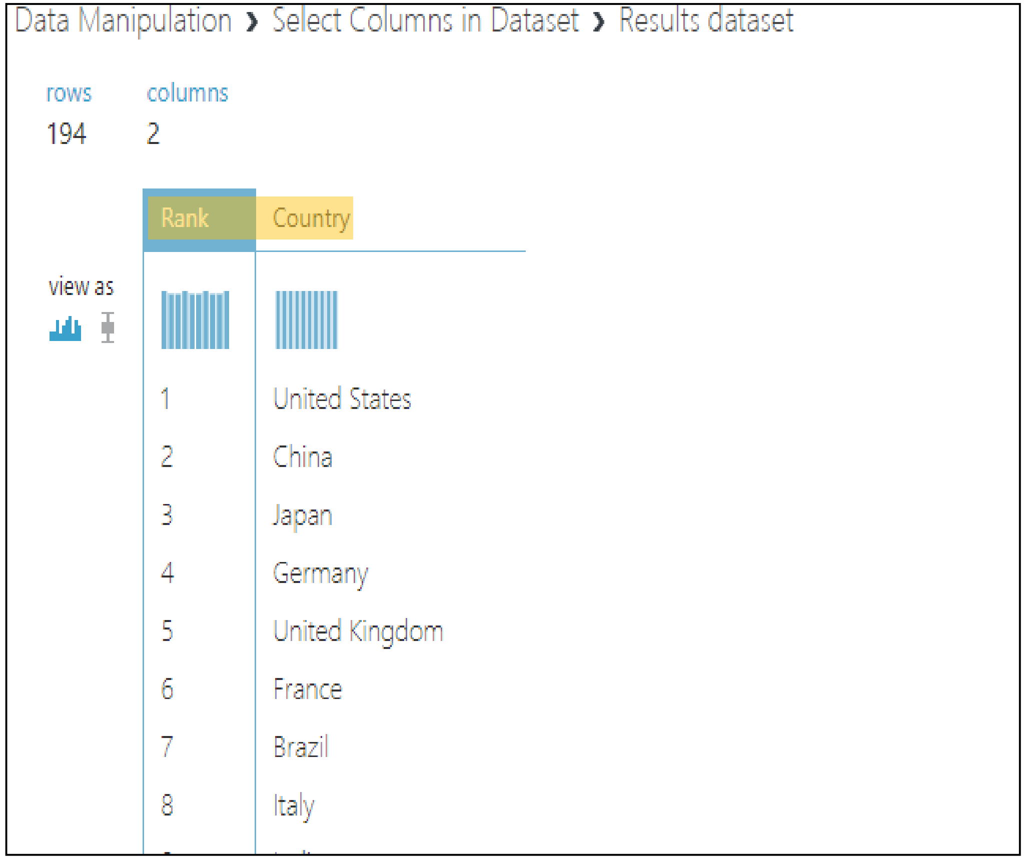


Fig4: Data with Selected Columns



Steps - Sub setting the data (Rows)

- For creating subset of rows we use R-script
- we shall take the output of selected columns as input for this
- Drag and drop dataset and select columns
- Search for ‘Execute R Script’ tile, drag and drop into the canvas
- Connect the output of selected columns to the first input port of ‘Execute R Script’
- Click on ‘Execute R Script’, in the properties window we can write the code for it(Code: fig-6)
- After writing the code, click on Run
- Once finished running click on the first output port to visualize the output

Steps - Sub setting the data (Rows)

Fig5: Add Execute R Script

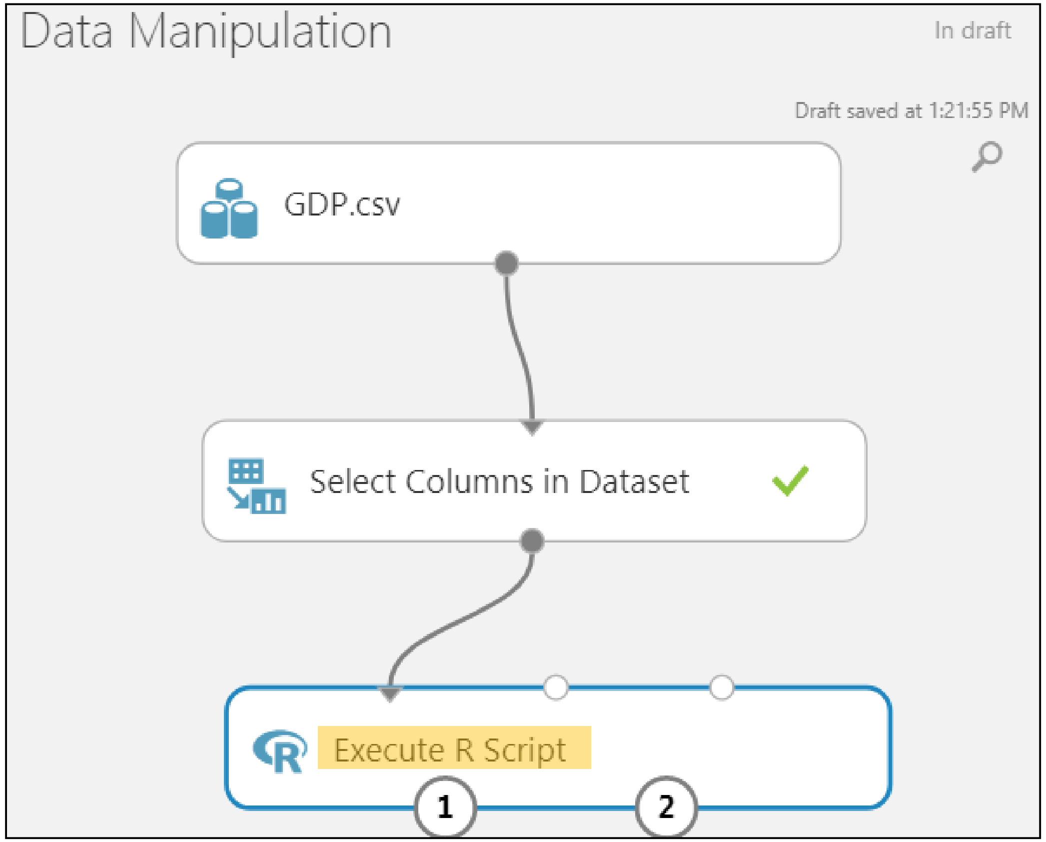


Fig6: R-Script

R Script

```r
dataset1 <- maml.mapInputPort(1) # class: data.frame

ss <- dataset1[1:10,]

maml.mapOutputPort("ss");
```

Steps - Sub setting the data (Rows)

Fig7: Visualize

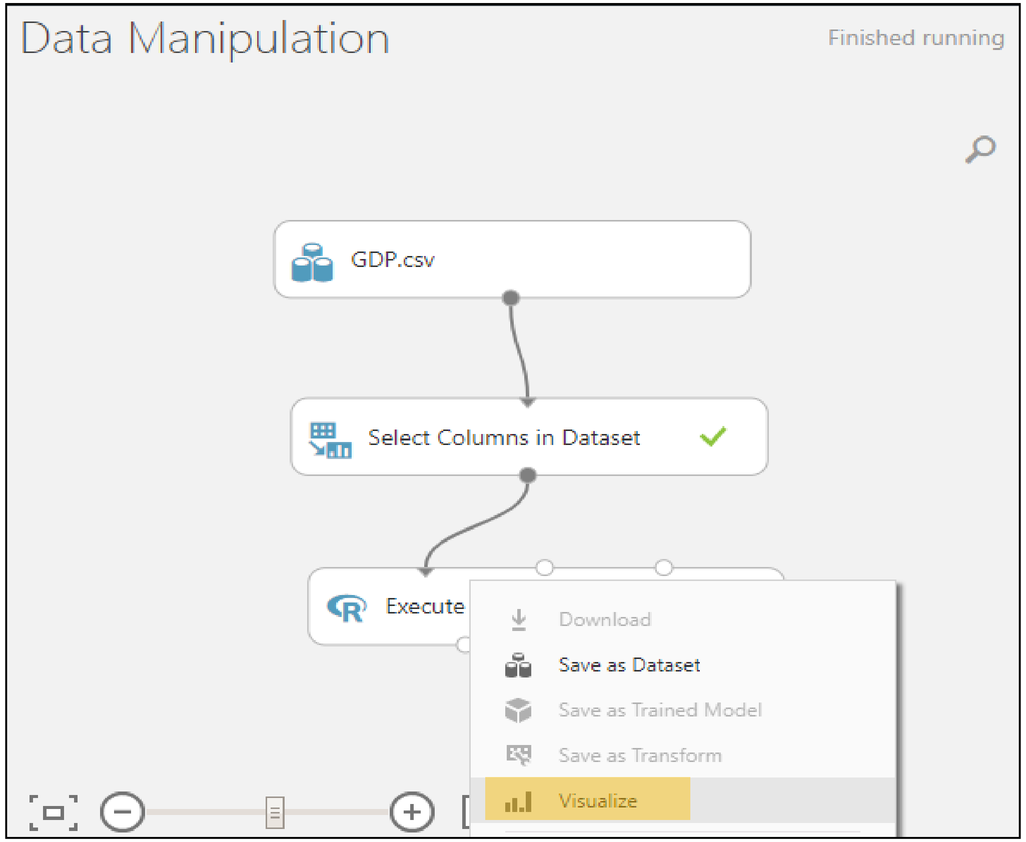
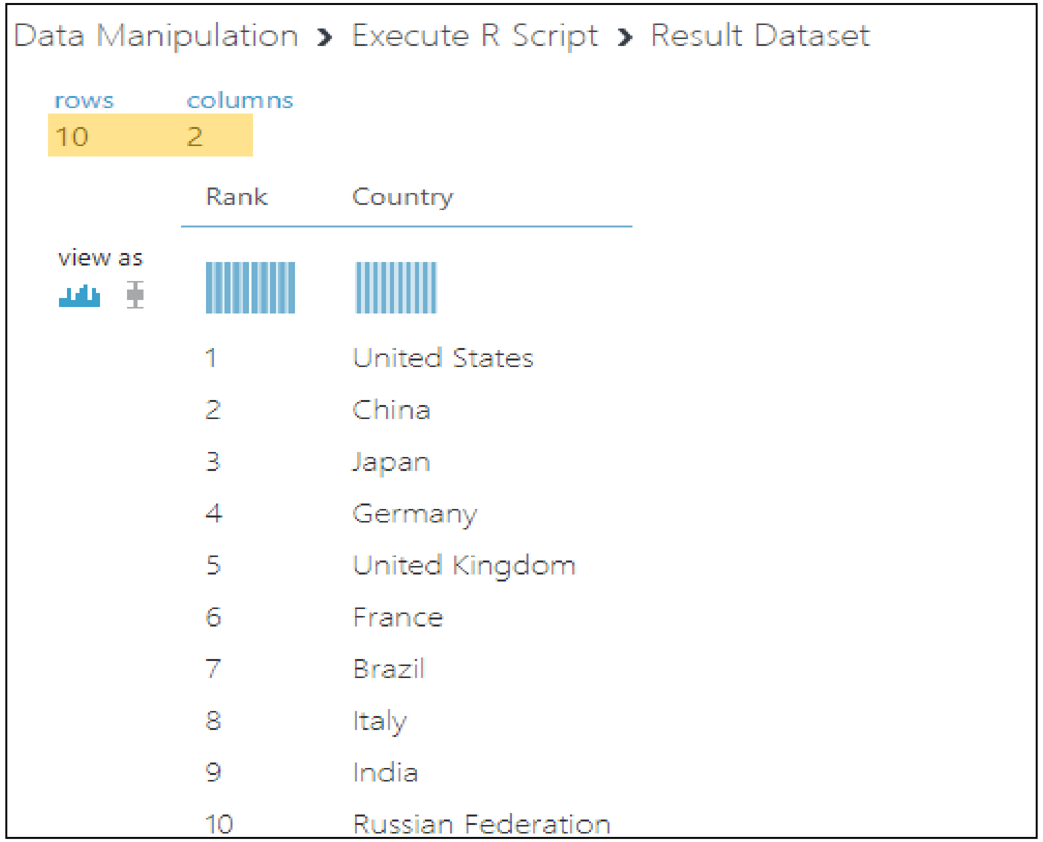


Fig8:Data with selected rows



Splitting data

- Splitting data, splits the data into two sets based on the given condition
- Splittng data has four modes:
  - Split Rows - splits the data into two sets based on the fraction value
  - Recommender split – splits data for training and testing
  - Regular expression - splits based on known value or part of value
  - Relative expression – splits based on the values in Numerical column
- Import the dataset: ~/World Bank Data/GDP.csv
- Here we split data based on GDP value(numeric)
- We use Relative expression to do this

Steps – Splitting data

- Drag and drop the dataset into the canvas
- Search for ‘Split Data’ tile, drag and drop into the canvas
- Select Relative expression in the splitting mode(properties)
- In the expression box type the following expression:
  - \"GDP" > 1923400 (\”column name” operator value)
- Click on run
- Once finished running visualize the output through both the ports

Steps – Splitting data

Fig9: Add Split Data

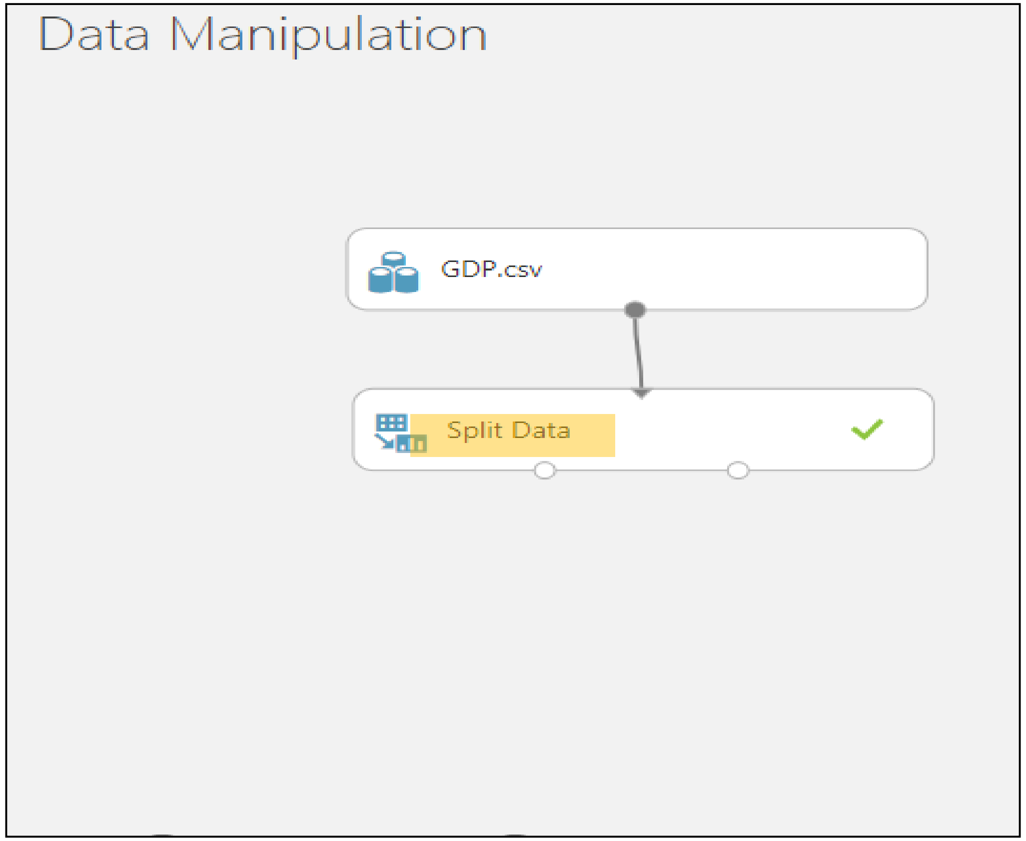
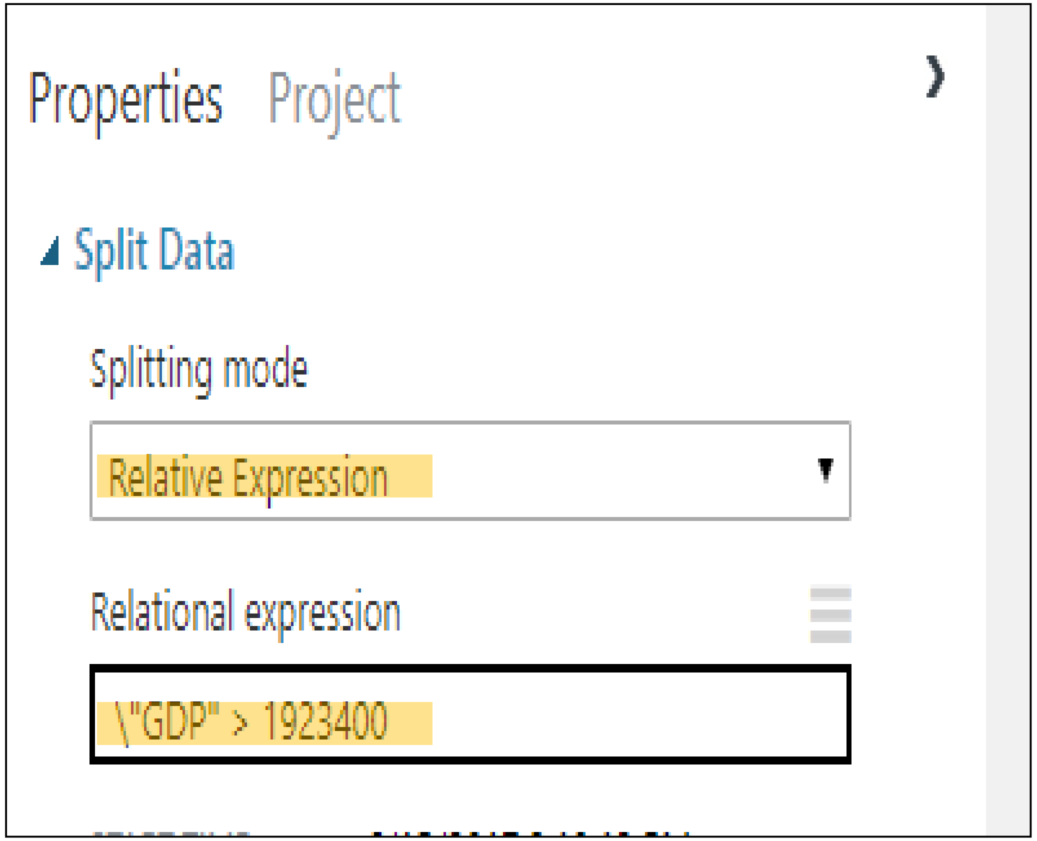


Fig10: Relative Expression



statinfer

Steps – Splitting data

Fig11: GDP > 1923400

Data Manipulation › Split Data › Results dataset1

rows	columns
9	4

Country_code	Rank	Country	GDP
USA	1	United States	17419000
CHN	2	China	10354832
JPN	3	Japan	4601461
DEU	4	Germany	3868291
GBR	5	United Kingdom	2988893
FRA	6	France	2829192
BRA	7	Brazil	2346076
ITA	8	Italy	2141161
IND	9	India	2048517

Fig12: GDP < 1923400

Data Manipulation › Split Data › Results dataset2

rows	columns
185	4

Country_code	Rank	Country	GDP
RUS	10	Russian Federation	1860598
CAN	11	Canada	1785387
AUS	12	Australia	1454675
KOR	13	Korea Rep	1410383
ESP	14	Spain	1381342
MEX	15	Mexico	1294690
IDN	16	Indonesia	888538
NLD	17	Netherlands	879319
TUR	18	Turkey	798429

Calculated fields

- We can perform mathematical operations between numerical fields in a dataset
- This can be done between two columns (column1 + column2) or between column and a constant
- The resultant column is the Calculated field
- To do this we use 'Apply Math Operation' tile
- We are going to find area of the car in the AutoDataset.csv
- Import Automobile Data Set/AutoDataset.csv

Steps - Calculated fields

- Drag and drop AutoDataset.csv into the canvas
- Search for ‘Apply Math Operation’, drag and drop connect it to dataset
- Select ‘operation’ in the category, ‘multiply’ in basic operation, ‘column set’ operation argument type
- Select ‘length’ in operation argument and ‘height’ in column set
- Give output mode as Append
- Click run, visializing this we can see a column named ‘Multiply(length_height)’
- Add another ‘Apply Math Operation’ connect the first output to this
- Select ‘operation’ in the category, ‘multiply’ in basic operation, ‘column set’ operation argument type

Steps - Calculated fields

- Select 'width' in operation argument and 'Multiply(length_height)' in column set
- Give output mode as inplace
- Click on run, visualizing this we can see that the Multiply(length_height) is updated with the new values
- To change the name of Multiply(length_height):
  - Search for Edit Metadata, drag and drop it into the canvas
  - Select Multiply(length_height) column in properties
  - Select Datatype → unchanged, Categorical → unchanged, Fields → unchanged
  - Give new column name as 'Area'
  - Click on run
- Visualize the output of Edit Metadata

Steps - Calculated fields

Fig13: Apply Math Operation

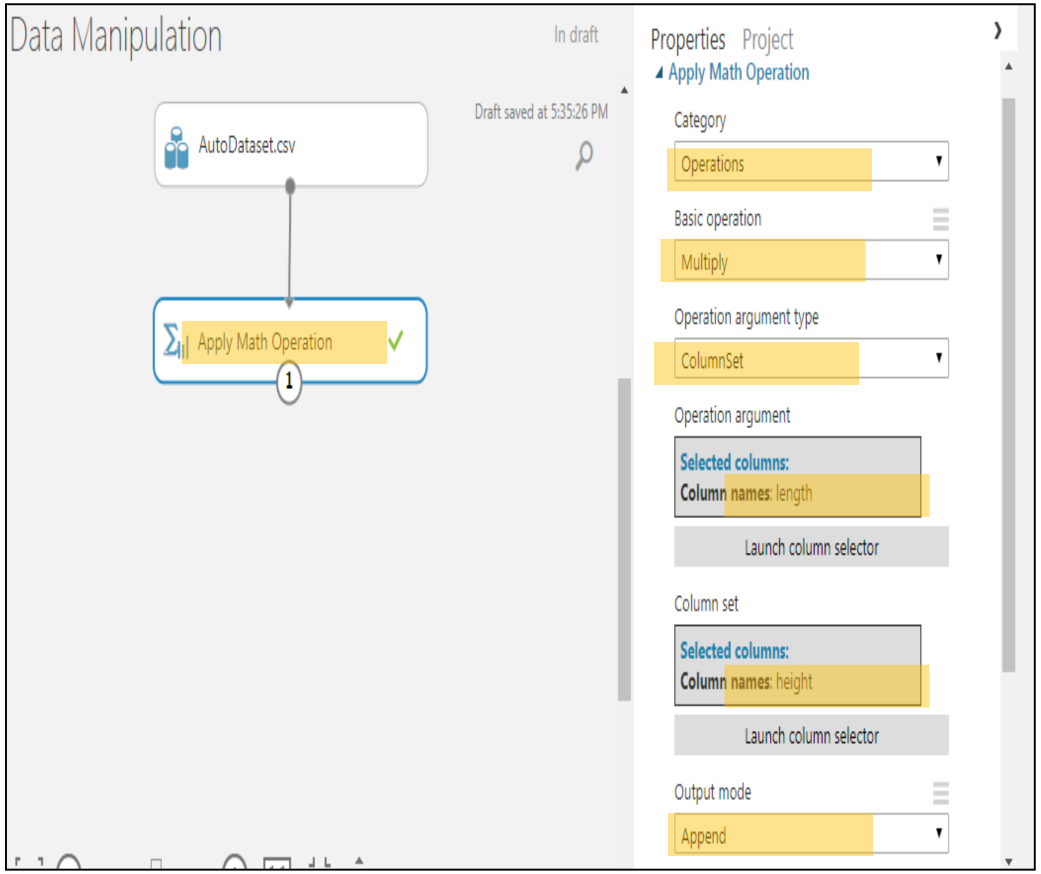
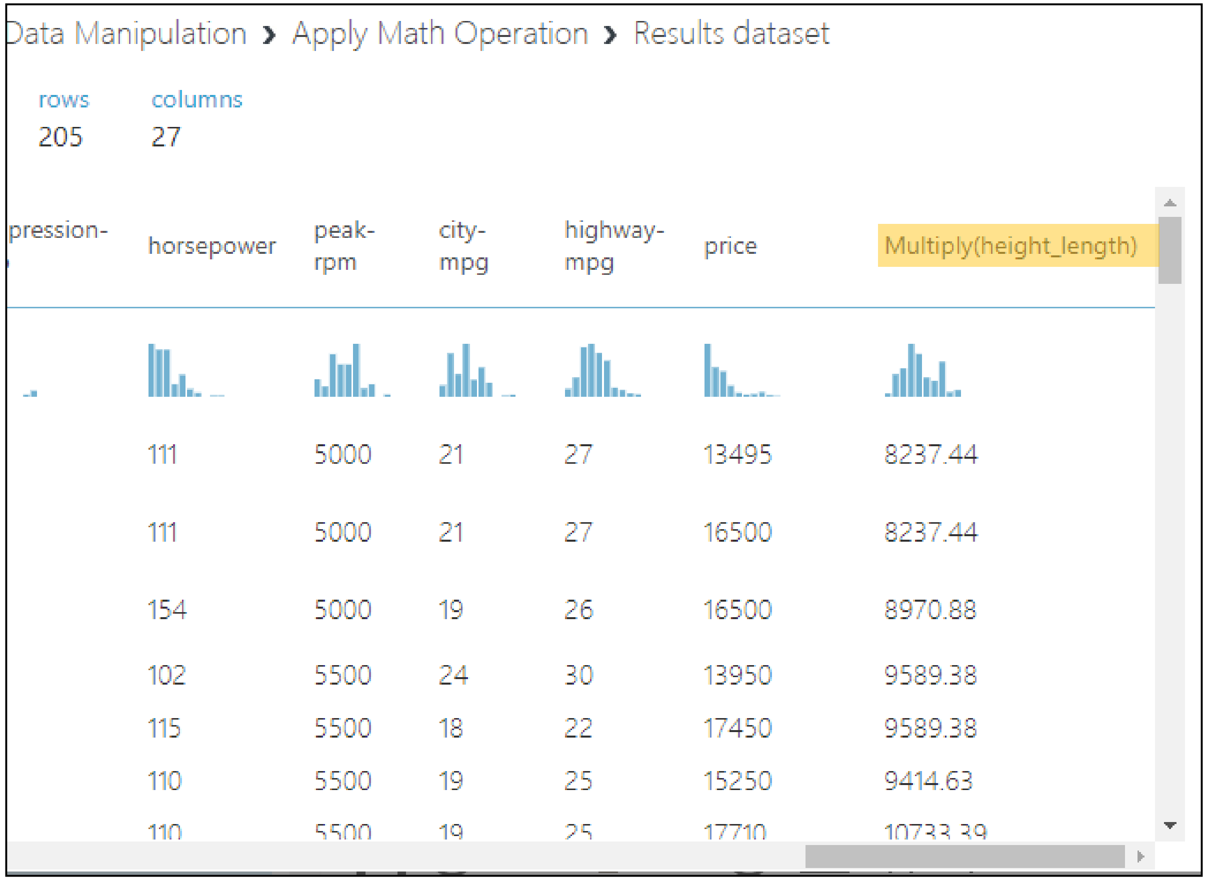


Fig14: Visualization



Steps - Calculated fields

Fig15: Apply Math Operation

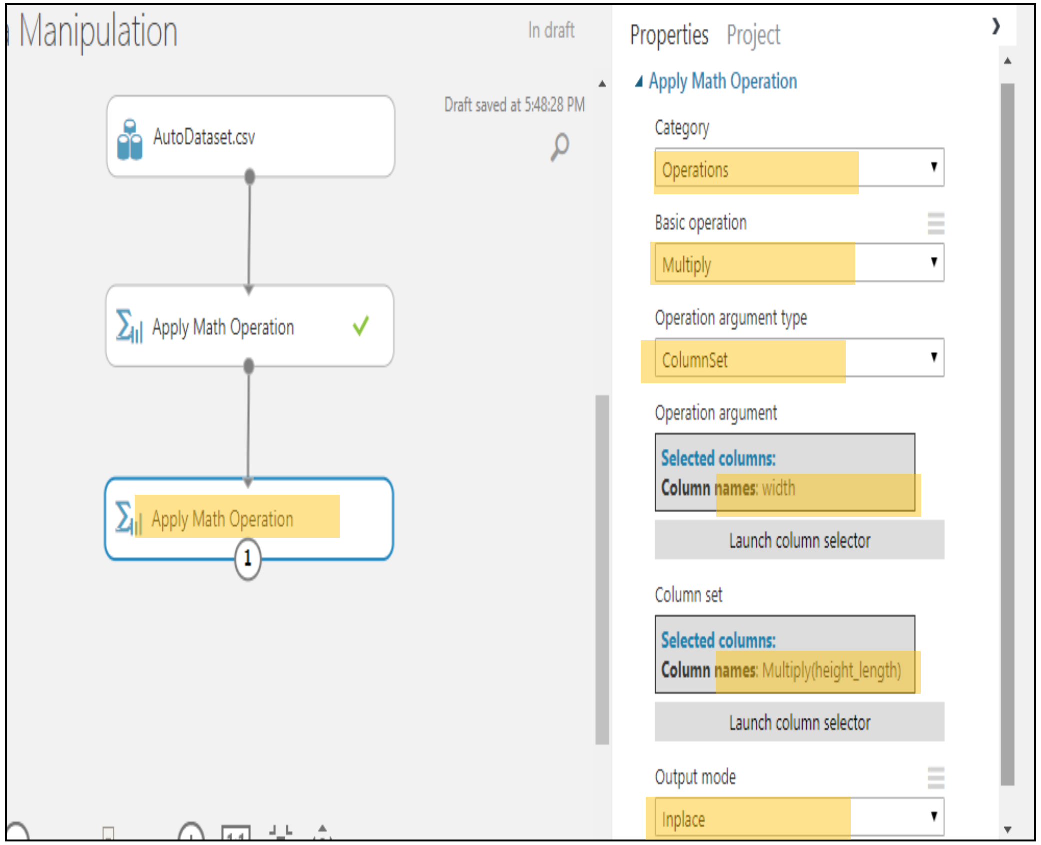
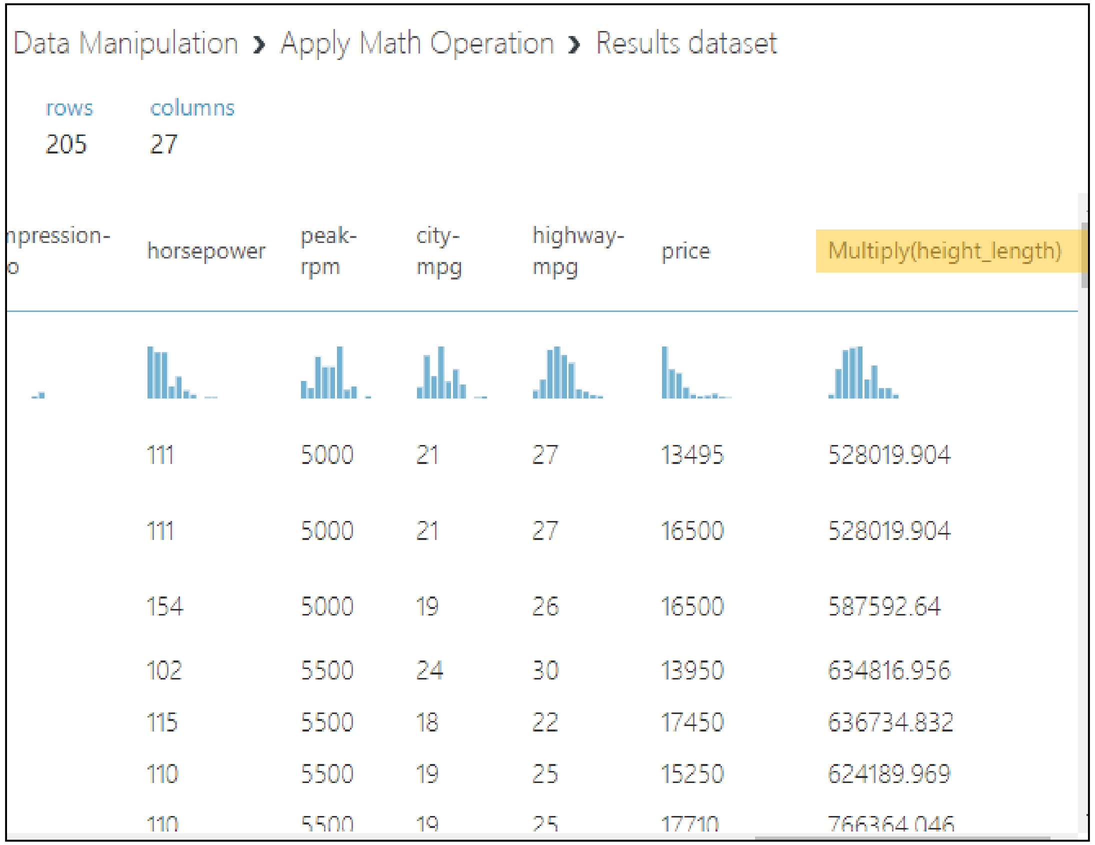


fig:16: Visualization



Steps - Calculated fields

Fig17: Edit Metadata

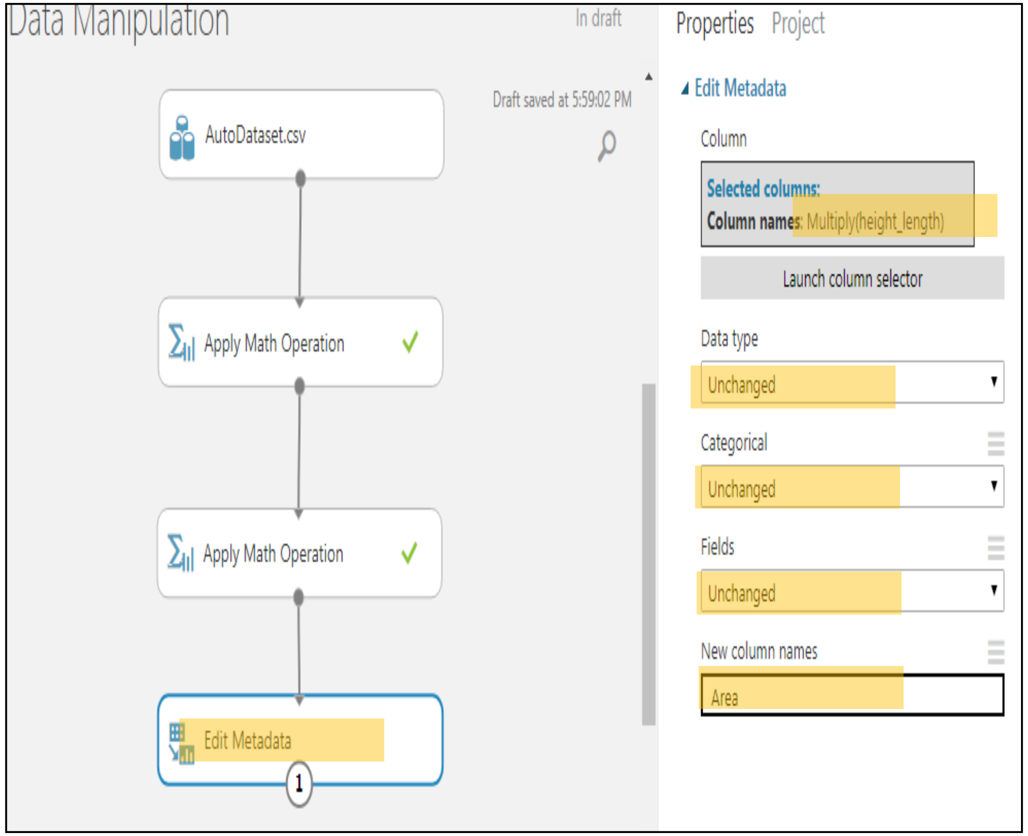
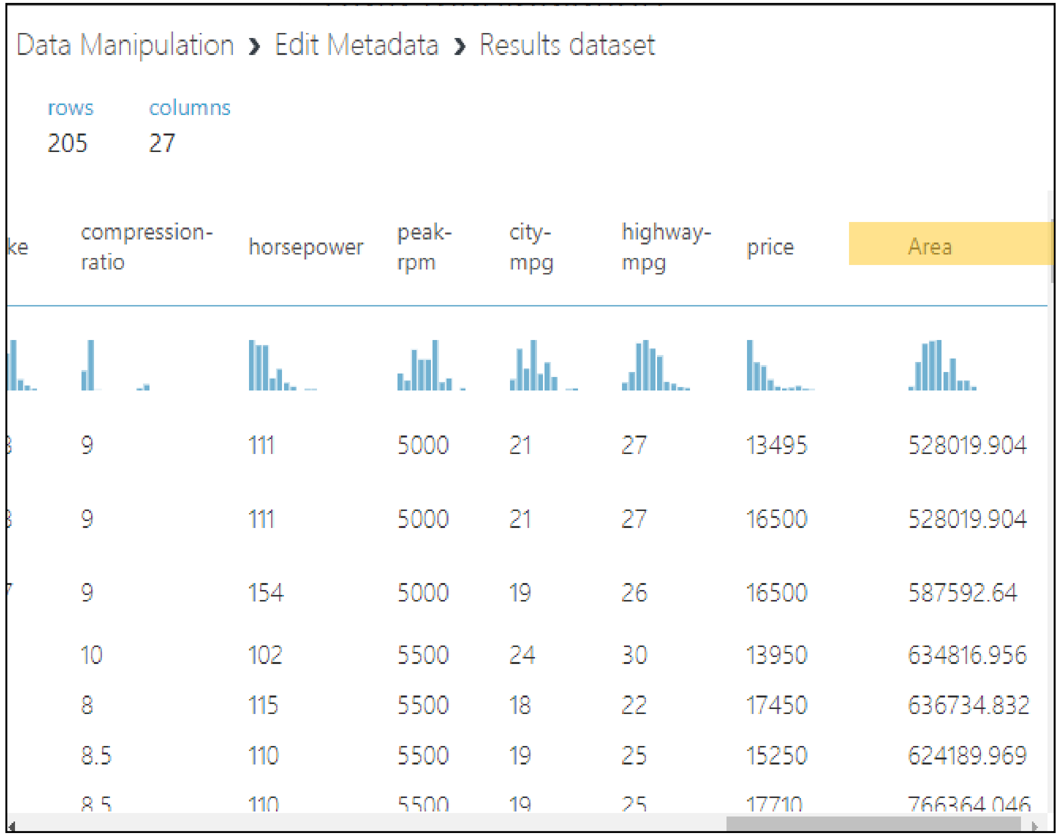


Fig18:Visualization



Sorting

- Sorting orders the data either ascending or descending based on the value in the column
- As of now we don't have direct sorting in azure
- We use R-Script code for sorting the data
- Import the dataset: ~/World Bank Data/GDP.csv

Steps - Sorting

- Drag and drop GDP.csv into the canvas
- Drag and drop Execute R Script into the canvas
- Connect the dataset to Execute R Script
- Write the code to sort in properties
- Click on run
- Once finished running, Visualize the data

Steps - Sorting

Fig19: Execute R Script

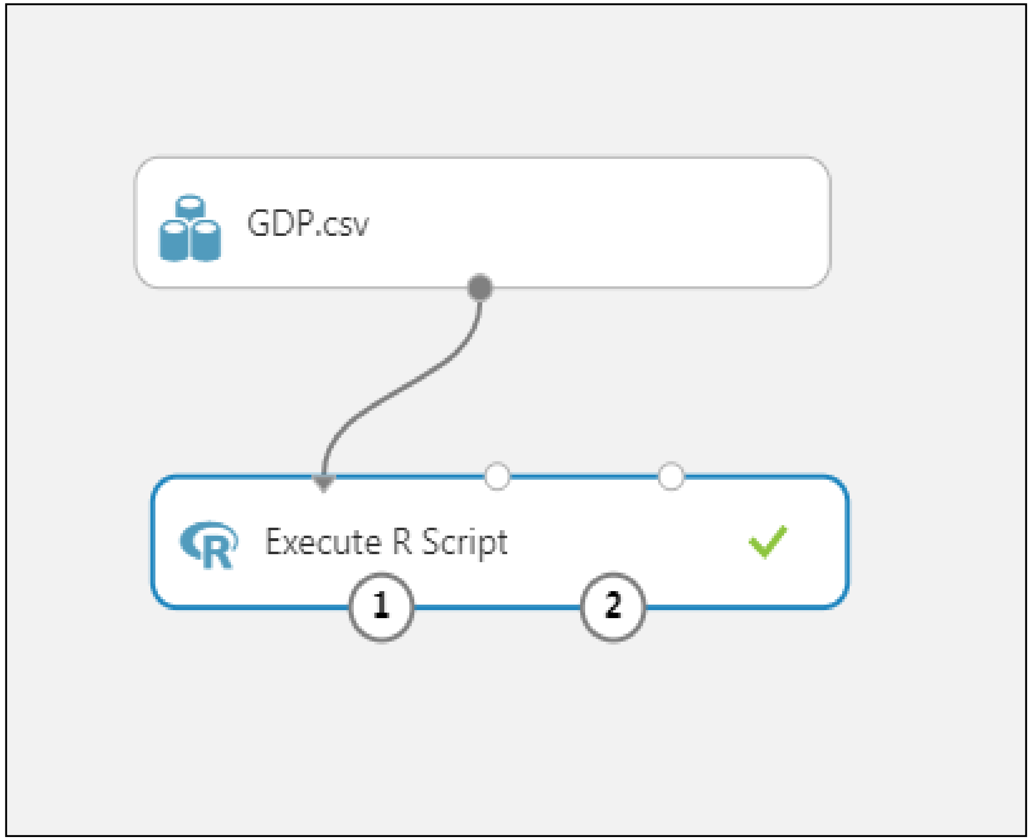


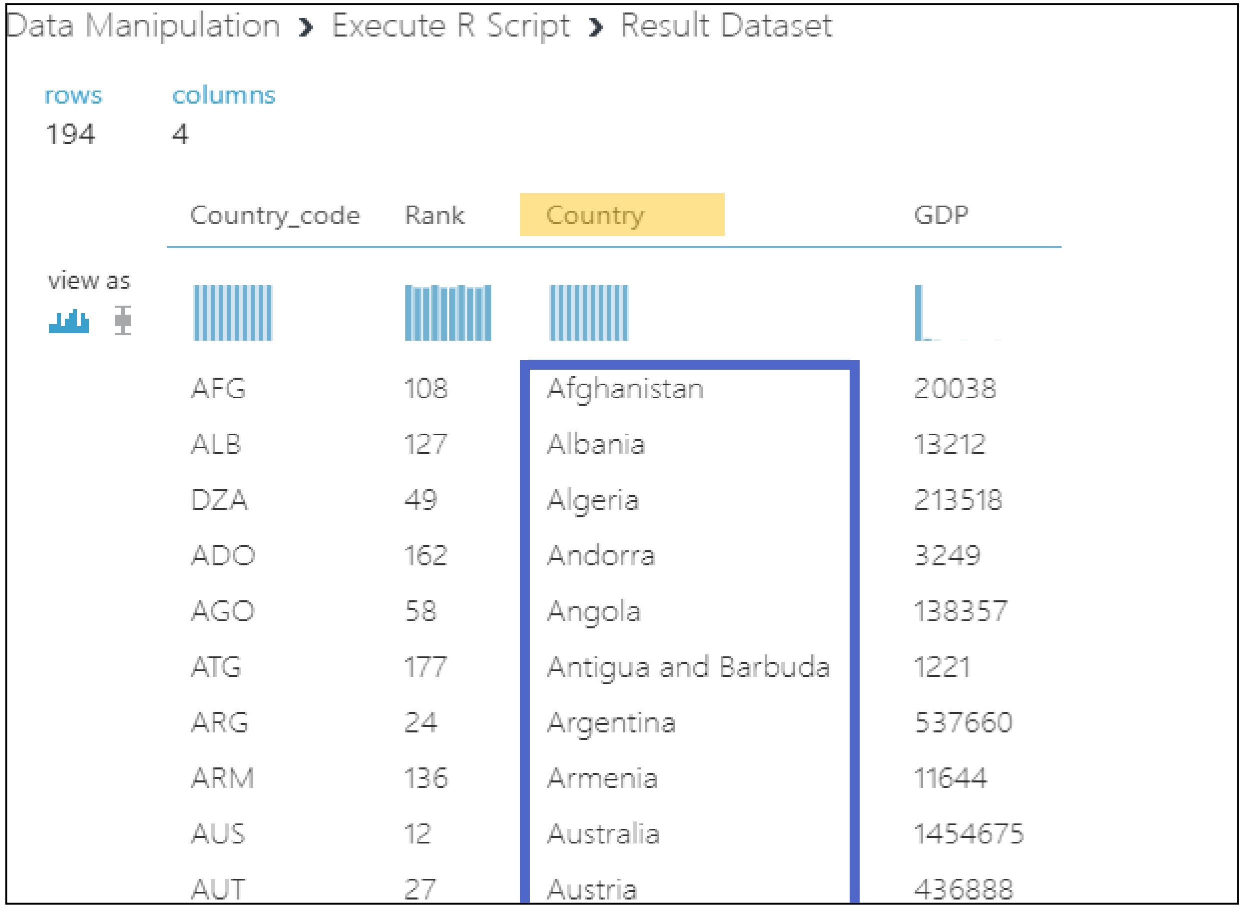
Fig20: R-Script Sorting

R Script

```r
dataset1 <- maml.mapInputPort(1) # class: data.frame

ss <- dataset1[order(dataset1$Country),]

maml.mapOutputPort("ss");
```

Steps - Sorting

Fig21: Visualization(sorted data)



Data Manipulation › Execute R Script › Result Dataset

rows 194 columns 4

view as

Country_code	Rank	Country	GDP
AFG	108	Afghanistan	20038
ALB	127	Albania	13212
DZA	49	Algeria	213518
ADO	162	Andorra	3249
AGO	58	Angola	138357
ATG	177	Antigua and Barbuda	1221
ARG	24	Argentina	537660
ARM	136	Armenia	11644
AUS	12	Australia	1454675
AUT	27	Austria	436888

Removing duplicate values

- Duplicate values in the dataset may cause inconsistency in the data processing
- Removing it from the data set solves the problem
- To remove the duplicate values in the data ‘Remove Duplicate Rows’ tile is used
- Import: Telecom Data Analysis/Bill.csv

Steps - Removing duplicate values

- Drag and drop bill.csv into the canvas
- Search for Remove Duplicate Rows, drag and drop into the canvas
- In launch column selector, select all the columns
- Ensure that Retain first duplicate row is checked

Steps - Removing duplicate values

Fig22: Remove Duplicate Rows

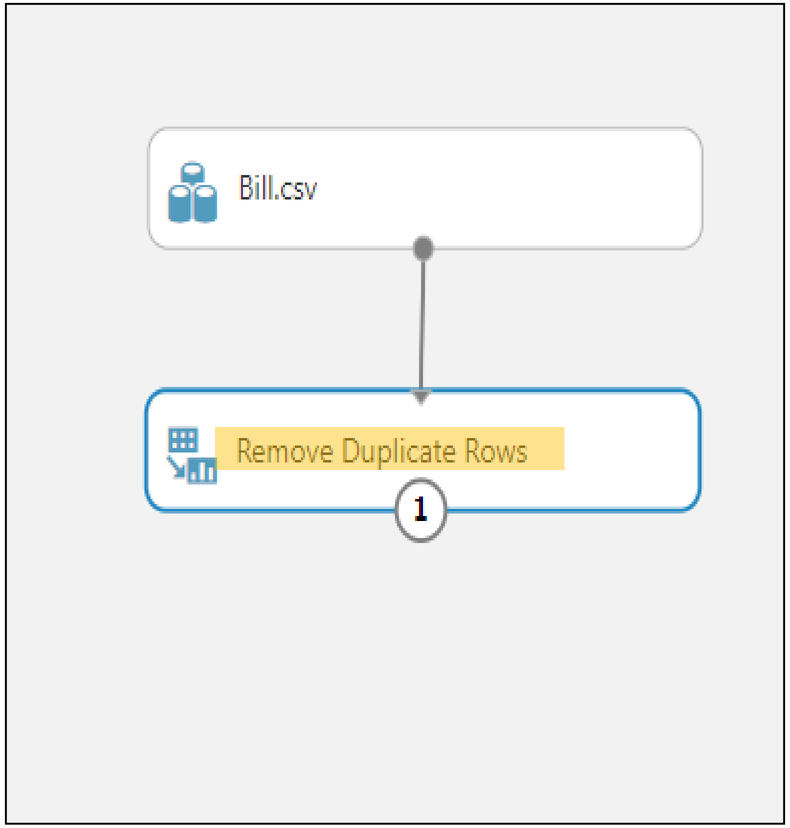
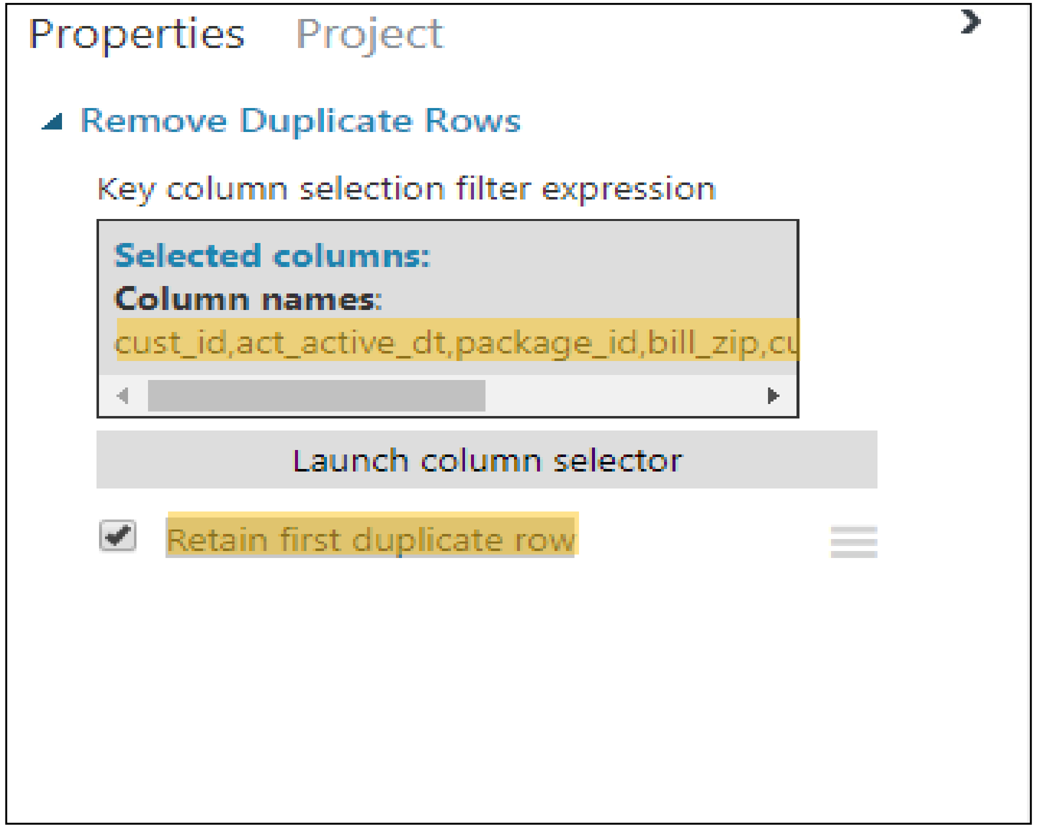
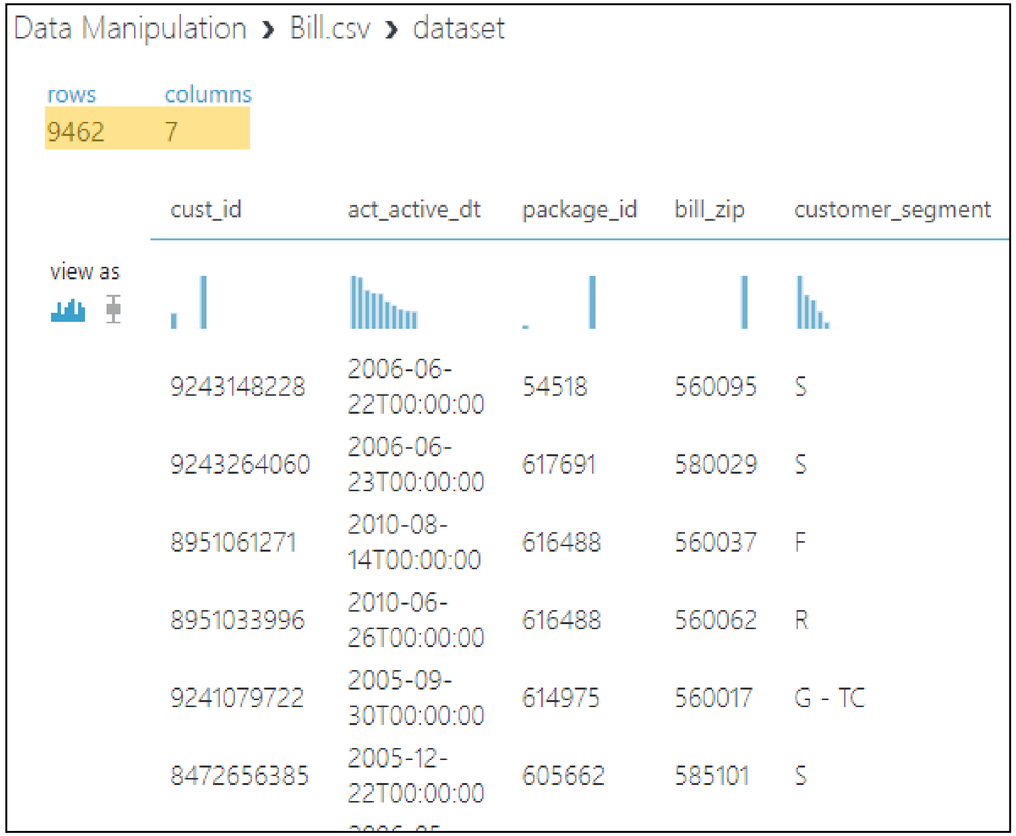


Fig23: Select All Columns



Steps - Removing duplicate values

Fig24:Before Removing Duplicate Values

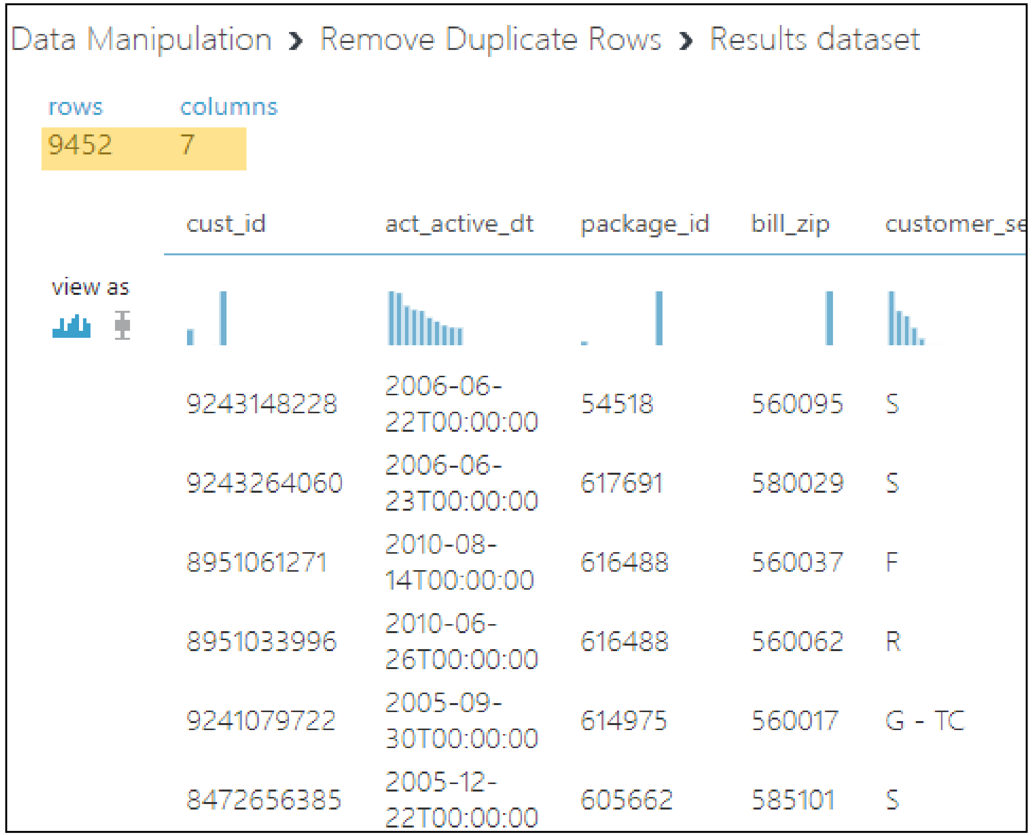


Data Manipulation › Bill.csv › dataset

rows	columns
9462	7

cust_id	act_active_dt	package_id	bill_zip	customer_segment
9243148228	2006-06-22T00:00:00	54518	560095	S
9243264060	2006-06-23T00:00:00	617691	580029	S
8951061271	2010-08-14T00:00:00	616488	560037	F
8951033996	2010-06-26T00:00:00	616488	560062	R
9241079722	2005-09-30T00:00:00	614975	560017	G - TC
8472656385	2005-12-22T00:00:00	605662	585101	S

Fig25: After Removing Duplicate Values



Data Manipulation › Remove Duplicate Rows › Results dataset

rows	columns
9452	7

cust_id	act_active_dt	package_id	bill_zip	customer_se
9243148228	2006-06-22T00:00:00	54518	560095	S
9243264060	2006-06-23T00:00:00	617691	580029	S
8951061271	2010-08-14T00:00:00	616488	560037	F
8951033996	2010-06-26T00:00:00	616488	560062	R
9241079722	2005-09-30T00:00:00	614975	560017	G - TC
8472656385	2005-12-22T00:00:00	605662	585101	S

Joining datasets

- Joining two datasets is done based on the primary key
- Joining has four types:
  - Inner Join
  - Full Outer Join
  - Left Outer Join
  - Left semi Join
- Import: TV Commercial Slots Analysis/orders.csv
- Import: TV Commercial Slots Analysis/slots.csv

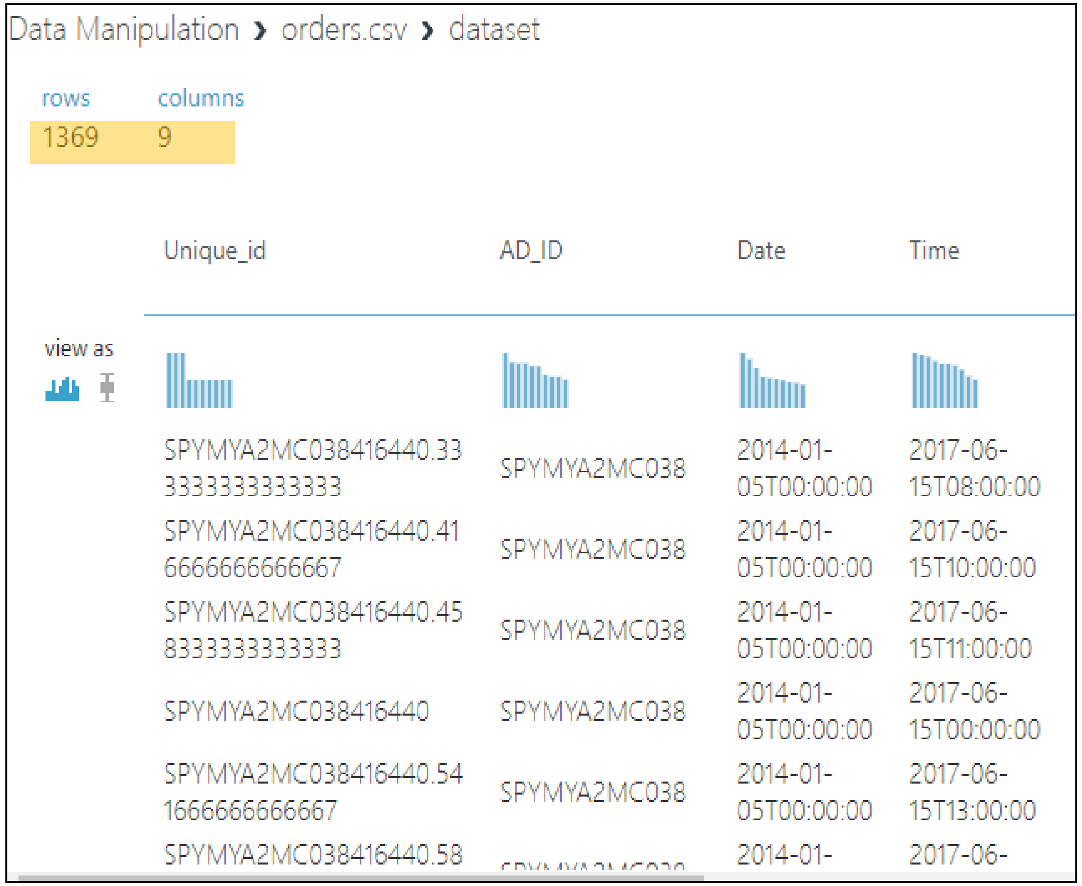
Steps - Joining datasets

- Drag and drop both the datasets into the canvas
- Search for ‘Join Data’ tile, drag and drop into the canvas
- Connect the first dataset to the first input port of ‘Join Data’ and second dataset to the second input port of ‘Join Data’
- Select the Join Key for both the datasets
- Select the type of join and click on run
- Once finished running, visualize the data

Steps - Joining datasets

statinfer

Fig26: Orders.csv



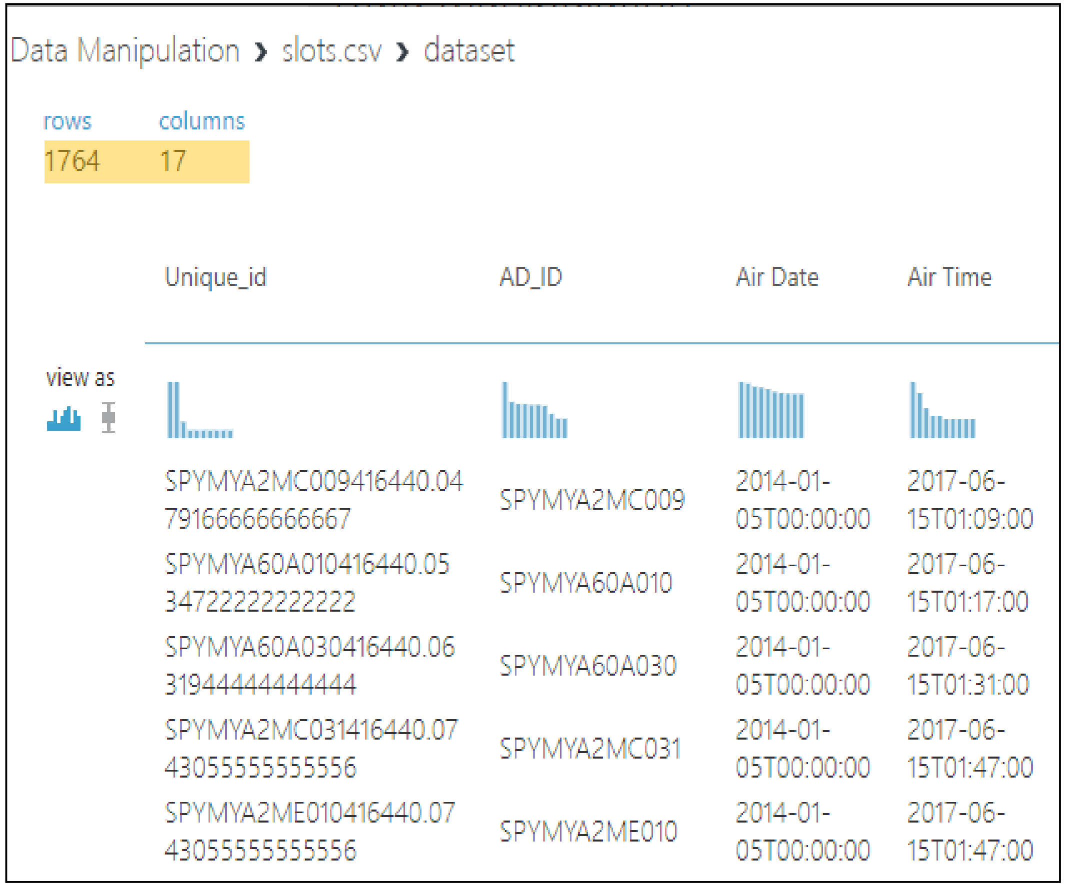
Data Manipulation › orders.csv › dataset

rows	columns
1369	9

view as

Unique_id	AD_ID	Date	Time
SPYMYA2MC038416440.33333333333333	SPYMYA2MC038	2014-01-05T00:00:00	2017-06-15T08:00:00
SPYMYA2MC038416440.41666666666667	SPYMYA2MC038	2014-01-05T00:00:00	2017-06-15T10:00:00
SPYMYA2MC038416440.45833333333333	SPYMYA2MC038	2014-01-05T00:00:00	2017-06-15T11:00:00
SPYMYA2MC038416440	SPYMYA2MC038	2014-01-05T00:00:00	2017-06-15T00:00:00
SPYMYA2MC038416440.54166666666667	SPYMYA2MC038	2014-01-05T00:00:00	2017-06-15T13:00:00
SPYMYA2MC038416440.58		2014-01-	2017-06-

Fig27: Slots.csv



Data Manipulation › slots.csv › dataset

rows	columns
1764	17

view as

Unique_id	AD_ID	Air Date	Air Time
SPYMYA2MC009416440.04791666666667	SPYMYA2MC009	2014-01-05T00:00:00	2017-06-15T01:09:00
SPYMYA60A010416440.05347222222222	SPYMYA60A010	2014-01-05T00:00:00	2017-06-15T01:17:00
SPYMYA60A030416440.06319444444444	SPYMYA60A030	2014-01-05T00:00:00	2017-06-15T01:31:00
SPYMYA2MC031416440.07430555555556	SPYMYA2MC031	2014-01-05T00:00:00	2017-06-15T01:47:00
SPYMYA2ME010416440.07430555555556	SPYMYA2ME010	2014-01-05T00:00:00	2017-06-15T01:47:00

Steps - Joining datasets

Fig28: Join Data

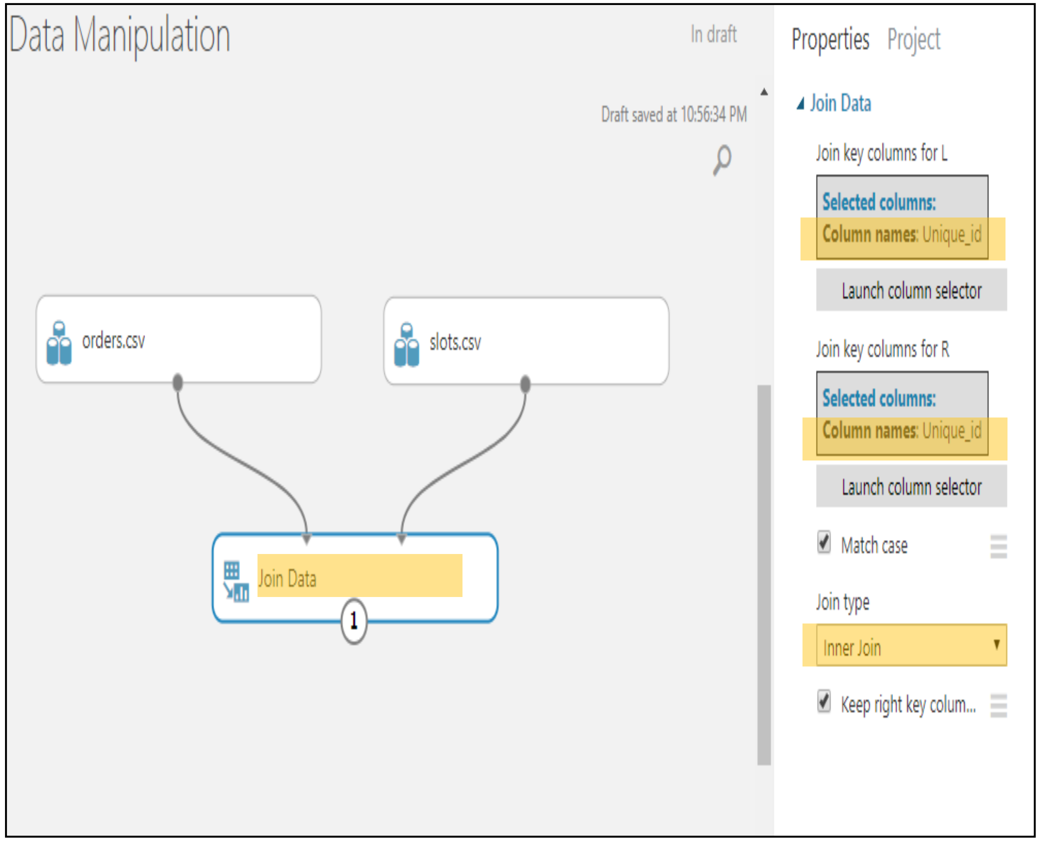
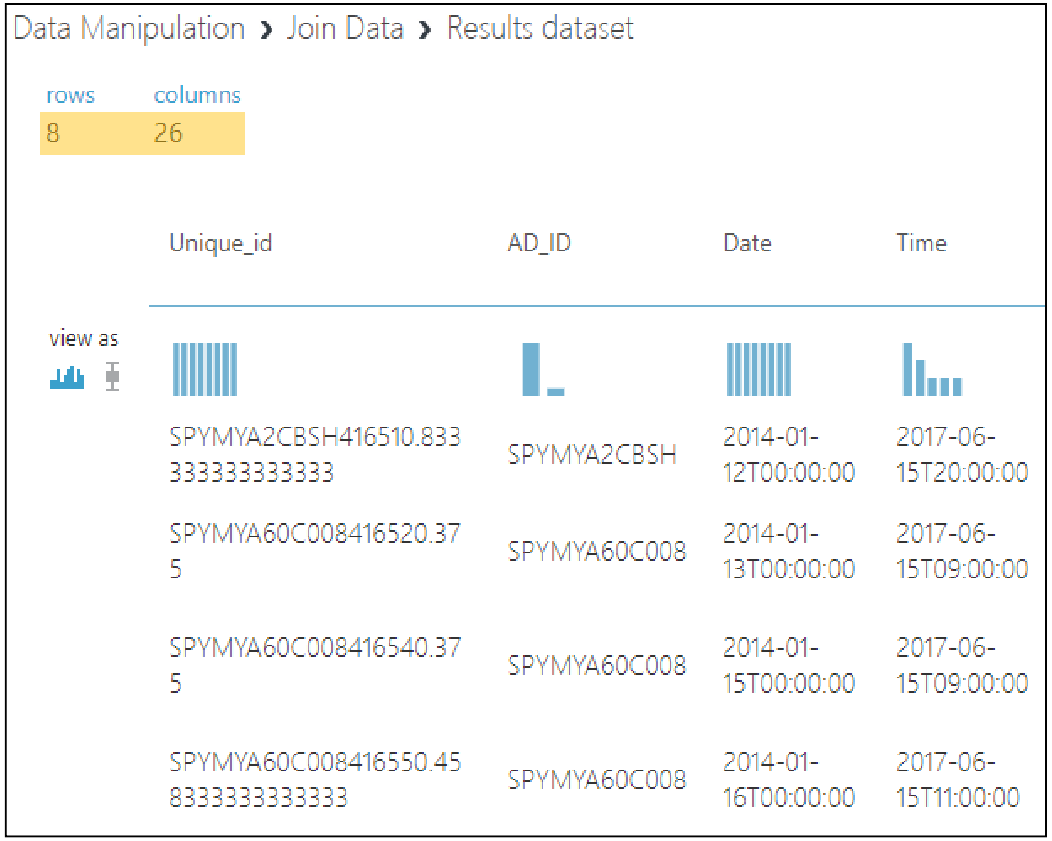


Fig29: Joined dataset



Thank you

Part 3/12 - Basic Statistics on Azure

statinfer.com

Contents

- Partition and Sampling
- Descriptive Statistics
  - Central Tendencies
  - Dispersion
- Quartiles and Percentiles
- Boxplots **and Outlier Detection**
- Creating Graphs

Partition and Sampling

- Partition and sampling allows to partition the dataset into samples
- In Azure, Partition and sample has four modes:
  - Assign to Folds - This assigns a number to each sample
  - Pick Fold - picks a sample based on the number in Assign to Folds
  - Sampling - This gives random sample based on the fraction
  - Head - This gives the top n values of the dataset
- Import: Online Retail Sales Data/Online Retail.csv

Steps - Partition and Sampling

- Sampling:
  - Drag and drop Online Retail.csv into the canvas
  - Search for ‘Partition and Sample’ module, drag and drop in to the canvas
  - Connect it to the dataset
  - Click on ‘Partition and Sample’, in properties select mode as sampling
  - Rate of sampling is the fraction for the sample(here we use 0.1 i.e. 10%)
  - Random seed accepts an positive integer, every time when we run with the same number we get the same sample, if 0 means random sample
  - Stratified split for sampling is true means sampling occurs based on the column specified
- Partition:
  - Drag and drop Online Retail.csv into the canvas
  - Search for ‘Partition and Sample’ module, drag and drop in to the canvas
  - Connect it to the dataset

Steps - Partition and Sampling

- Click on ‘Partition and Sample’, in properties select mode as Assign to Folds
- Select a number for Random seed
- Specify the partition method, evenly or customised
- Number of Folds is the number of distinct samples we want(here we give 3)
- Give stratified split as false
- Drag and drop three more ‘Partition and Sample’ module
- Connect the output of the previous one to these three
- Select mode as Pick Folds for all the three
- In specify the fold give 1 for the first, 2 for second and 3 for third
- Click on run

Steps - Partition and Sampling

Fig1: Online Retail.csv

Basic Statistics > Online Retail.csv > dataset

rows	columns
541909	8

view as

InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice
536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01T08:26:00	2.55
536365	71053	WHITE METAL LANTERN	6	2010-12-01T08:26:00	3.39
536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01T08:26:00	2.75

Steps - Partition and Sampling

Fig2: Partition and Sample

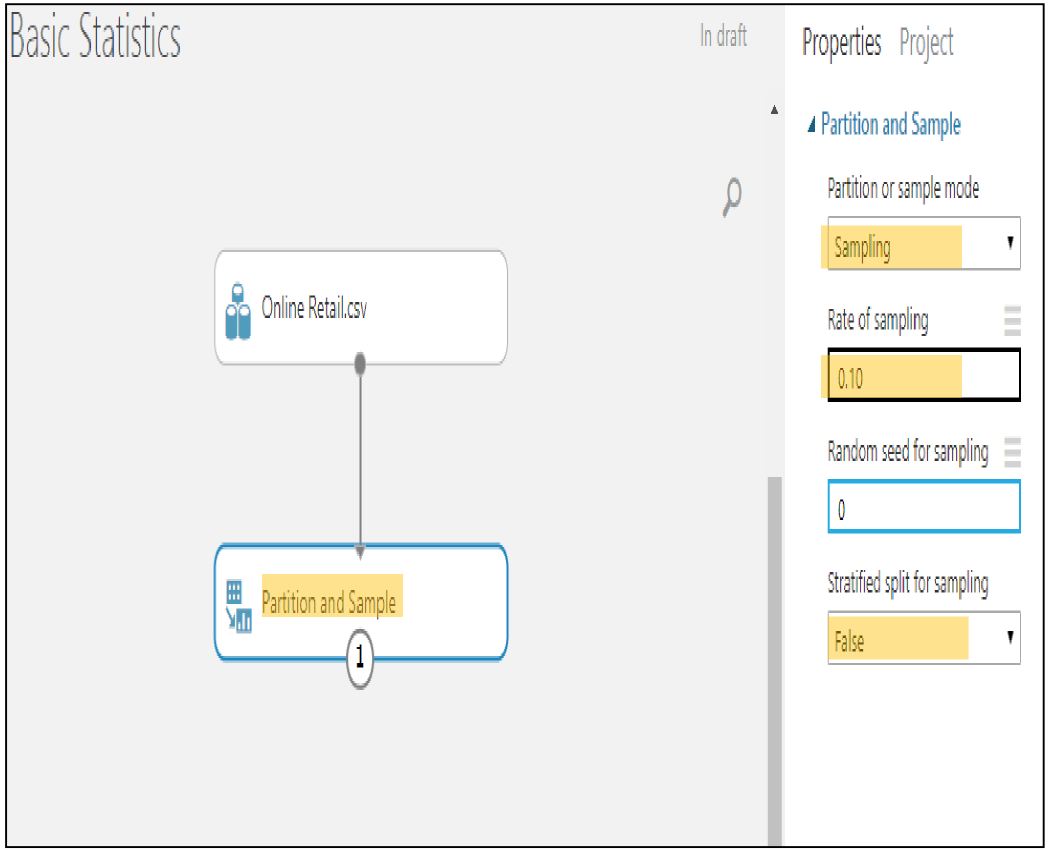
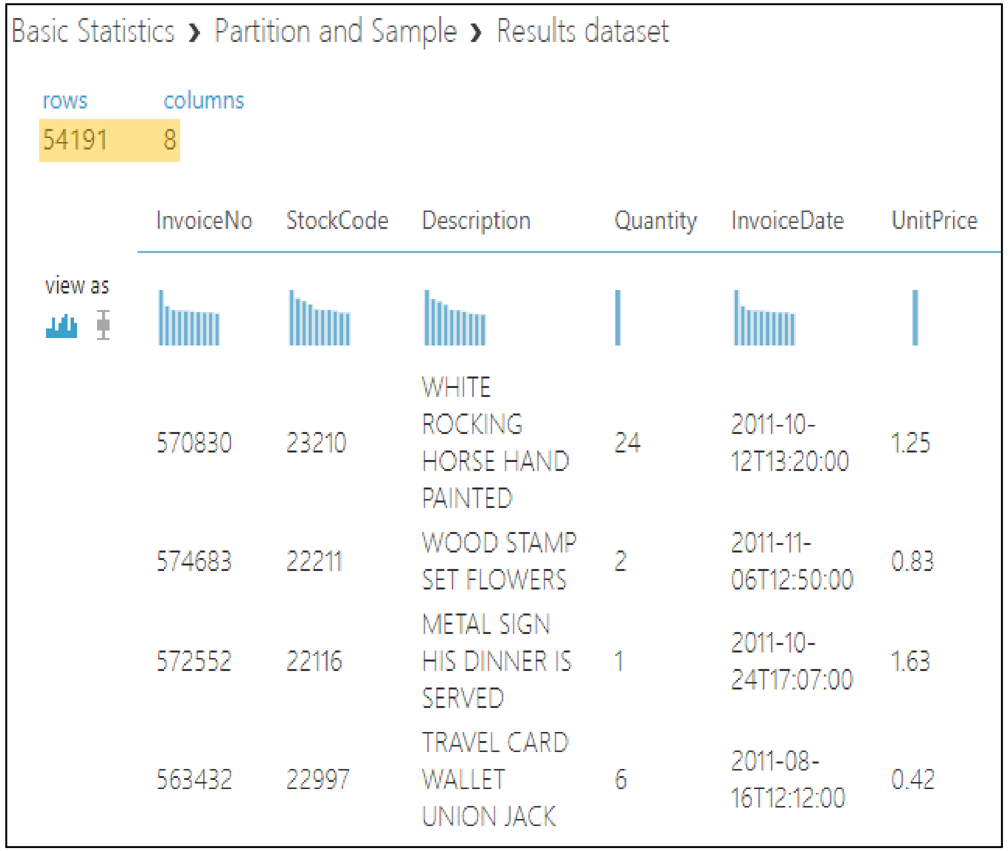


Fig3: Sample Data(10%)



Steps - Partition and Sampling

Fig4: Assign to Folds

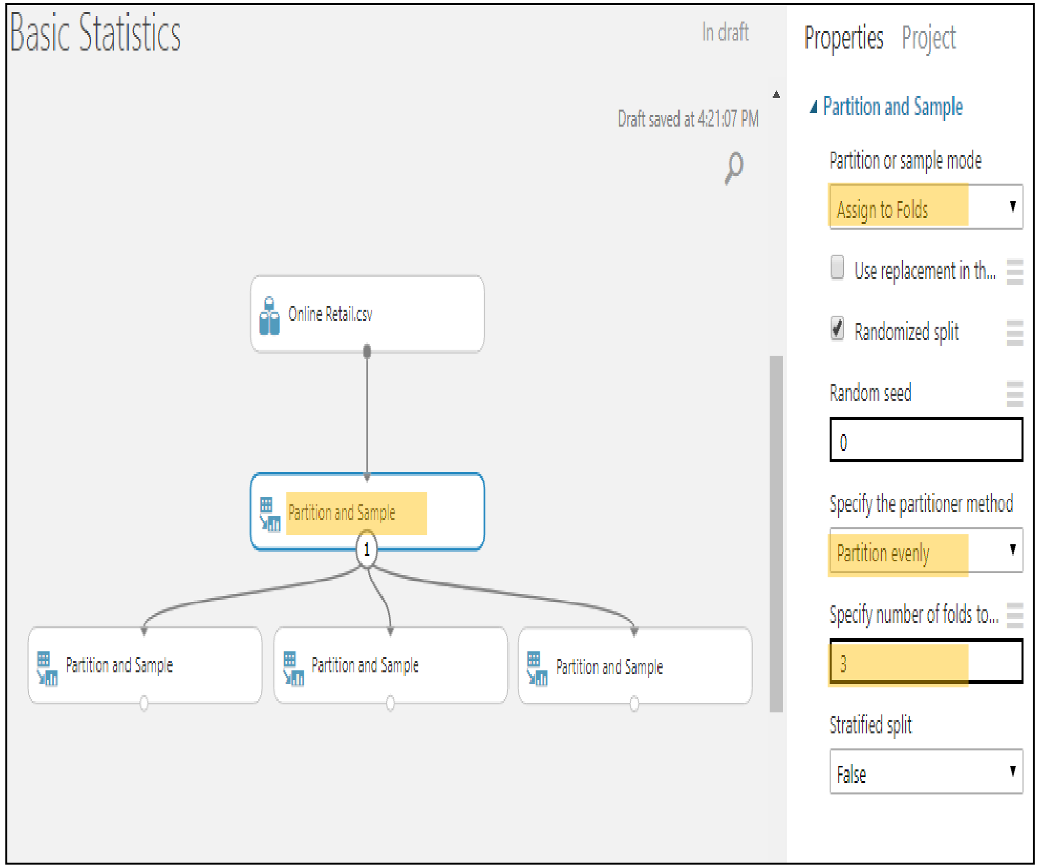
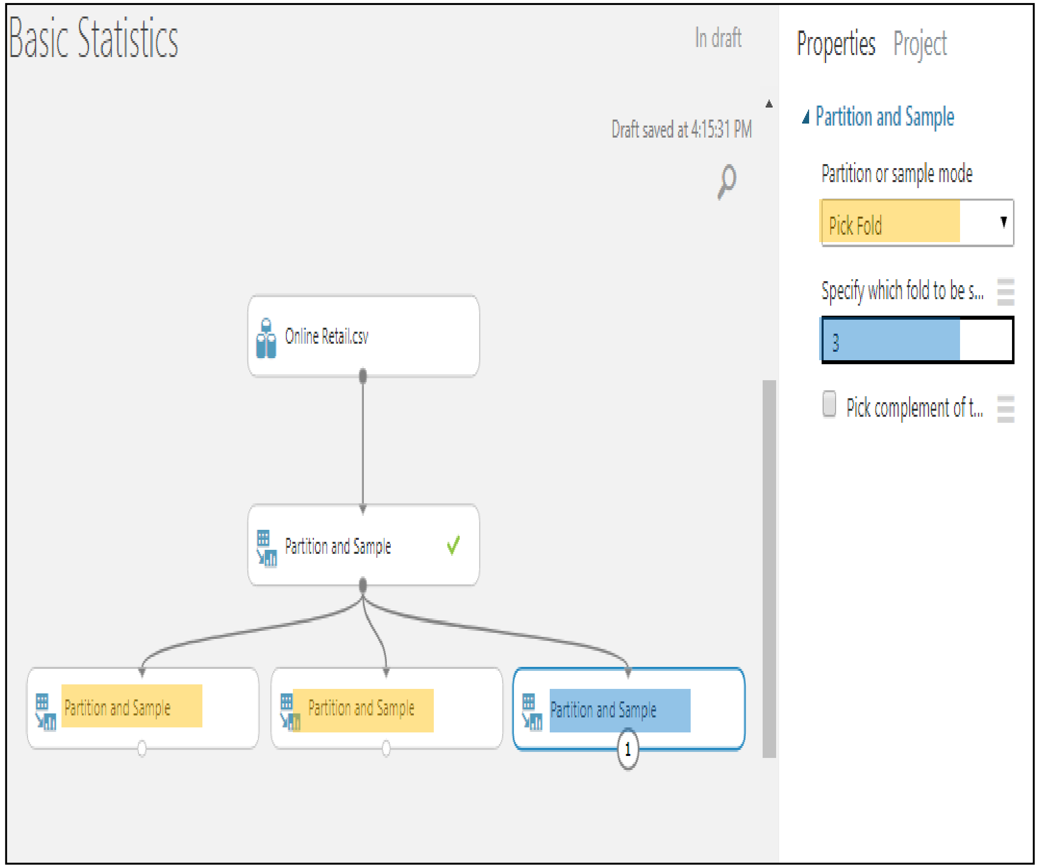


Fig5: Pick Folds



statinfer

Steps - Partition and Sampling

Fig6: Sample 1

Basic Statistics › Partition and Sample › Results dataset

rows	columns
180636	8

view as

InvoiceNo	StockCode	Description	Quantity
541215	20685	DOORMAT RED RETROSPOT	4
538177	84970S	HANGING HEART ZINC T-LIGHT HOLDER	1
554093	20979	36 PENCILS TUBE RED RETROSPOT	16
560640	23285	PINK VINTAGE SPOT BEAKER	8
574088	22627	MINT KITCHEN SCALES	2
		REGENCY	

Fig7: Sample 2

Basic Statistics › Partition and Sample › Results dataset

rows	columns
180636	8

view as

InvoiceNo	StockCode	Description	Quantity
573496	21934	SKULL SHOULDER BAG	1
565067	22616	PACK OF 12 LONDON TISSUES	12
579529	20750	RED RETROSPOT MINI CASES	2
554283	84509B	SET OF 4 FAIRY CAKE PLACEMATS	4
		CHILDREN'S	

Fig8: Sample 3

Basic Statistics › Partition and Sample › Results dataset

rows	columns
180637	8

view as

InvoiceNo	StockCode	Description	Quantity
563382	21621	VINTAGE UNION JACK BUNTING	2
577078	20983	12 PENCILS TALL TUBE RED RETROSPOT	1
562286	22630	DOLLY GIRL LUNCH BOX	24
577598	22196	SMALL HEART MEASURING SPOONS	12
548893	22994	TRAVEL CARD WALLET RETROSPOT	1

Descriptive Statistics

- Descriptive statistics allows us to know about the variable and their distribution
- Central tendencies
  - Mean
  - Median
- Dispersion
  - Variance
  - Standard deviation
- There are two ways to find these values:
  - Method 1: Visualize the Dataset
  - Method 2: Compute Elementary Statistics
- Import: Census Income Data/Income_data.csv

Steps - Descriptive Statistics

- Method 1:Visualize the Dataset
  - Drag and drop the dataset into the canvas
  - Click the output port to visualize the data
  - Click on the column name for which the statistics to be calculated
  - On the right side you can find the values
- Method 2: Compute Elementary Statistics
  - Drag and drop the dataset into the canvas
  - Search for Compute Elementary Statistics, drag and drop it into the canvas
  - Click on the Compute Elementary Statistics, in properties select the method
  - Select the columns and click run
  - Once finished running, visualize the data

Steps - Descriptive Statistics

Fig9: Visualize the Dataset

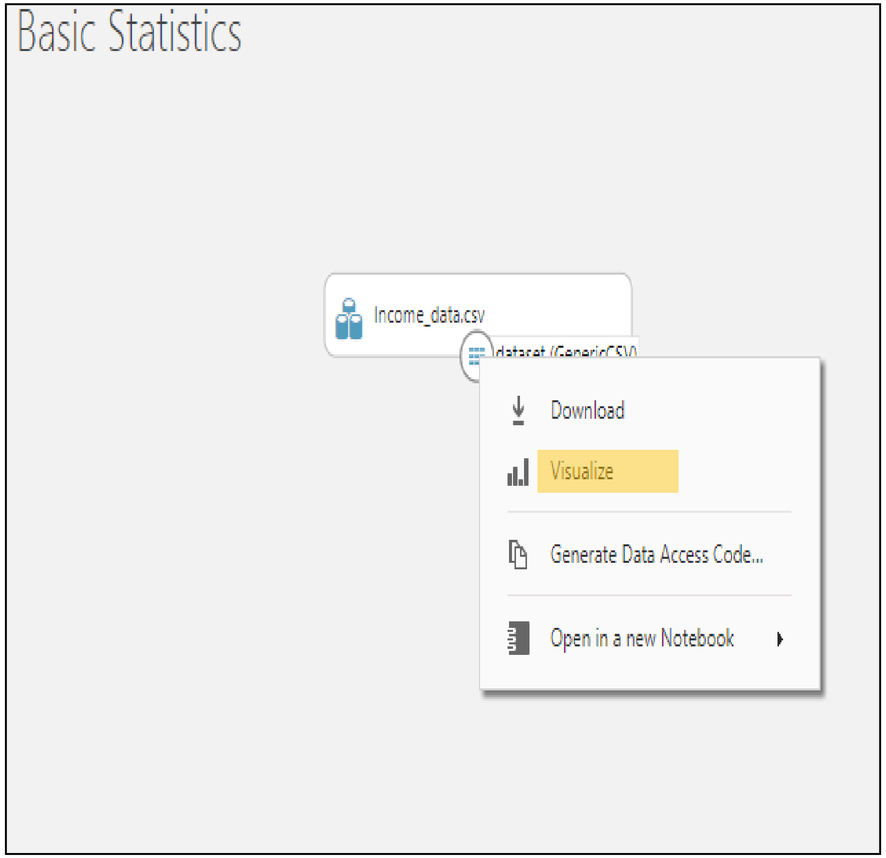
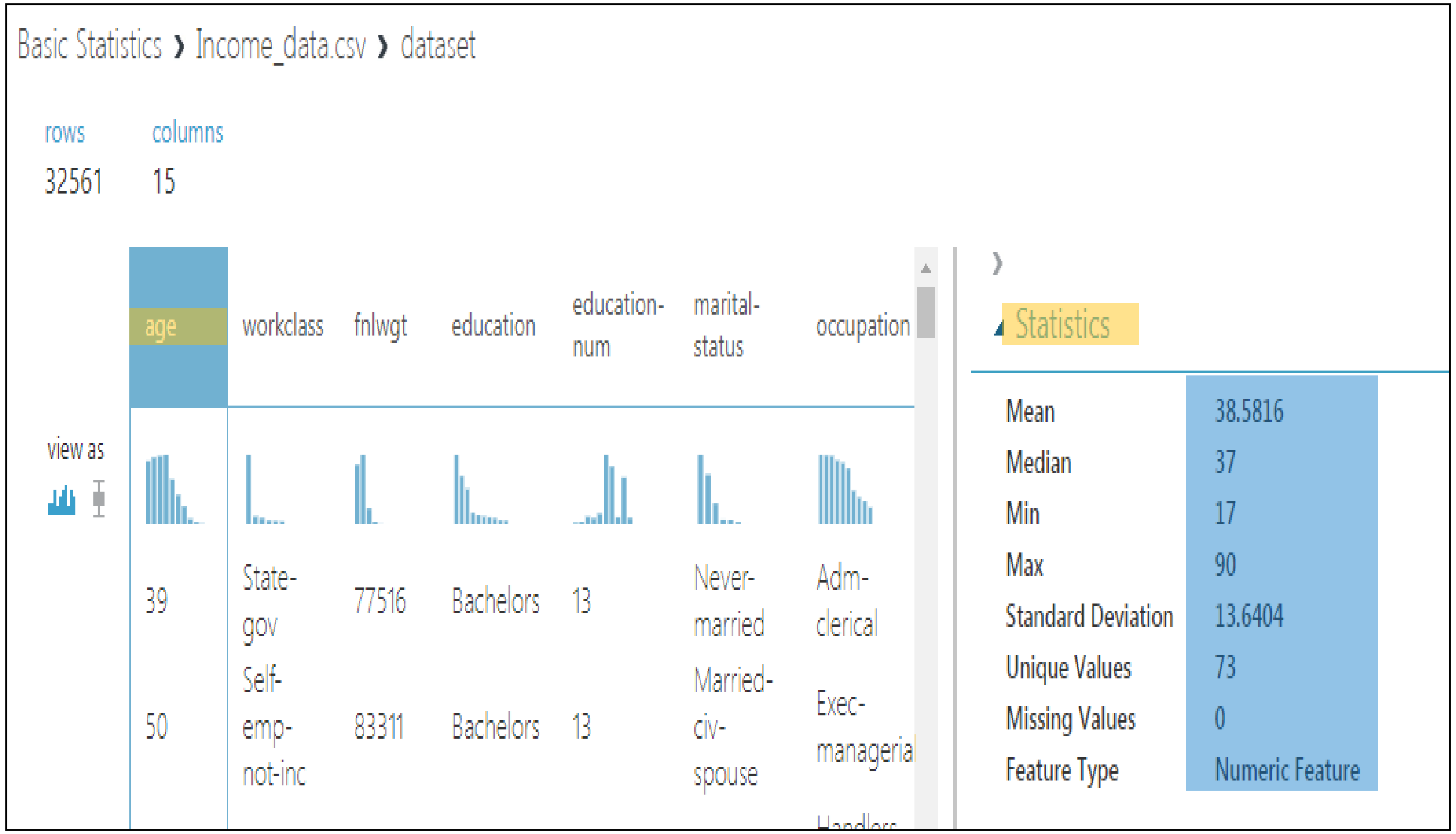


Fig10: Basic Statistics



Steps - Descriptive Statistics

Fig11: Compute Elementary Statistics

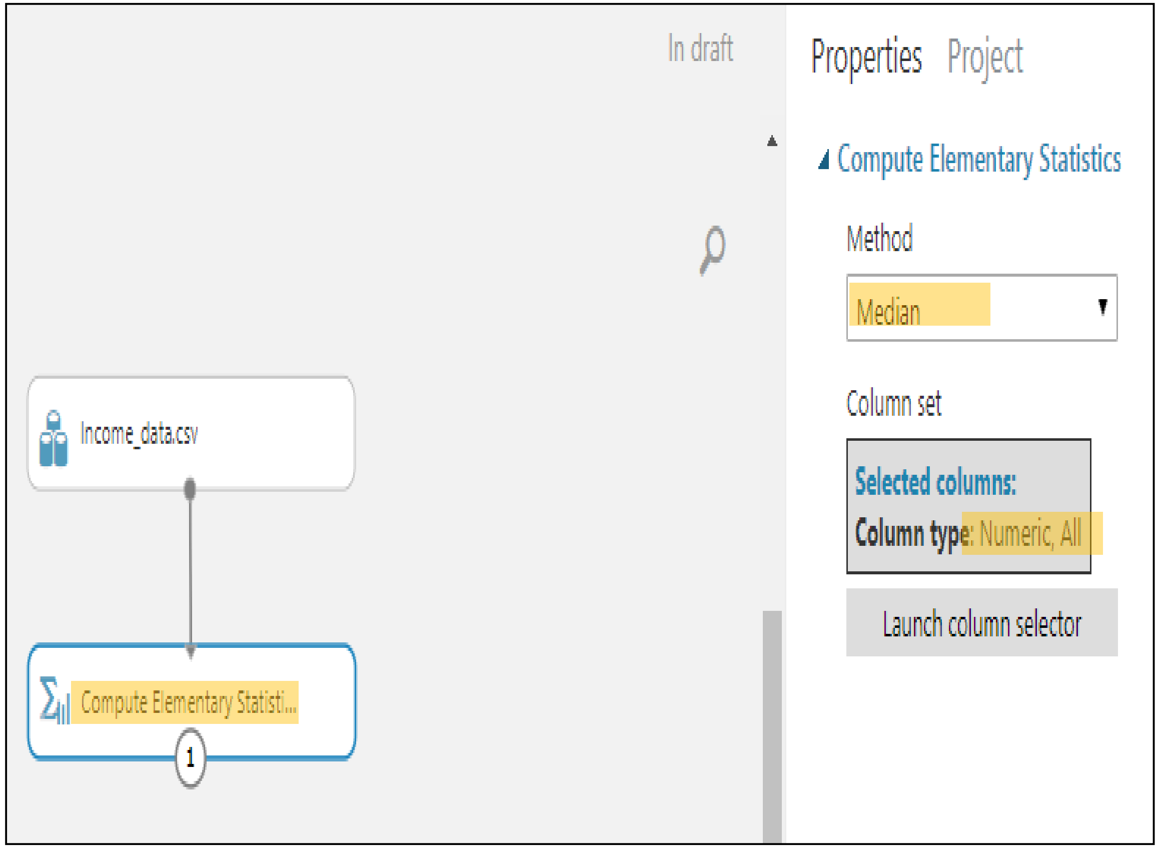
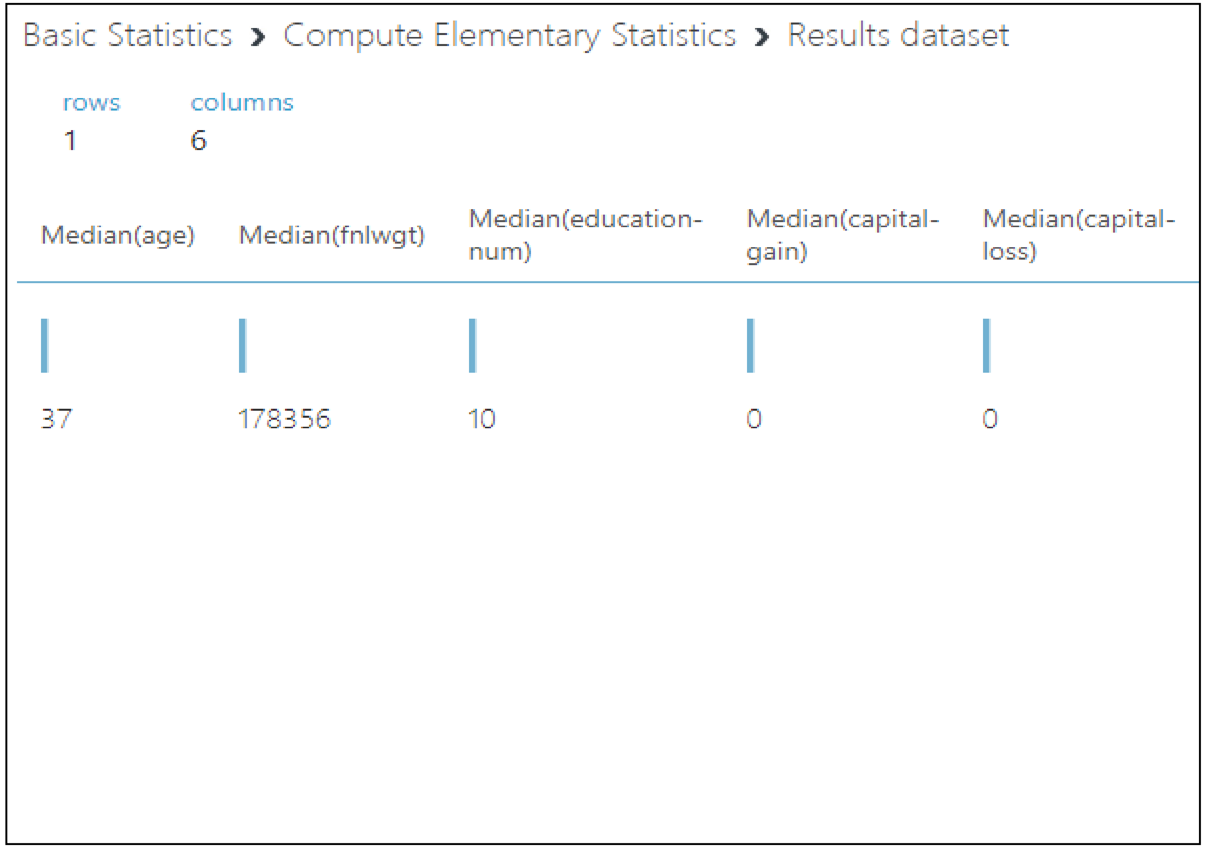


Fig12: Median(For Numeric Columns)



Percentiles and Quartiles

- **Percentiles**
  - A student attended an exam along with 1000 others.
  - He got 68% marks? How good or bad he performed in the exam?
  - What will be his rank overall?
  - What will be his rank if there were 100 students overall?
  - For example, with 68 marks, he stood at 90th position. There are 910 students who got less than 68, only 89 students got more marks than him
  - He is standing at 91 percentile.
  - Instead of stating 68 marks, 91% gives a good idea on his performance
  - Percentiles make the data easy to read
  - p^{th} percentile: p percent of observations below it, (100 - p)% above it.
  - Marks are 40 but percentile is 80%, what does this mean?

Percentiles and Quartiles

- 80% of CAT exam percentile means
- 20% are above & 80% are below
- Percentiles help us in getting an idea on outliers.
- For example the highest income value is 400,000 but 95th percentile is 20,000 only. That means 95% of the values are less than 20,000. So the values near 400,000 are clearly outliers

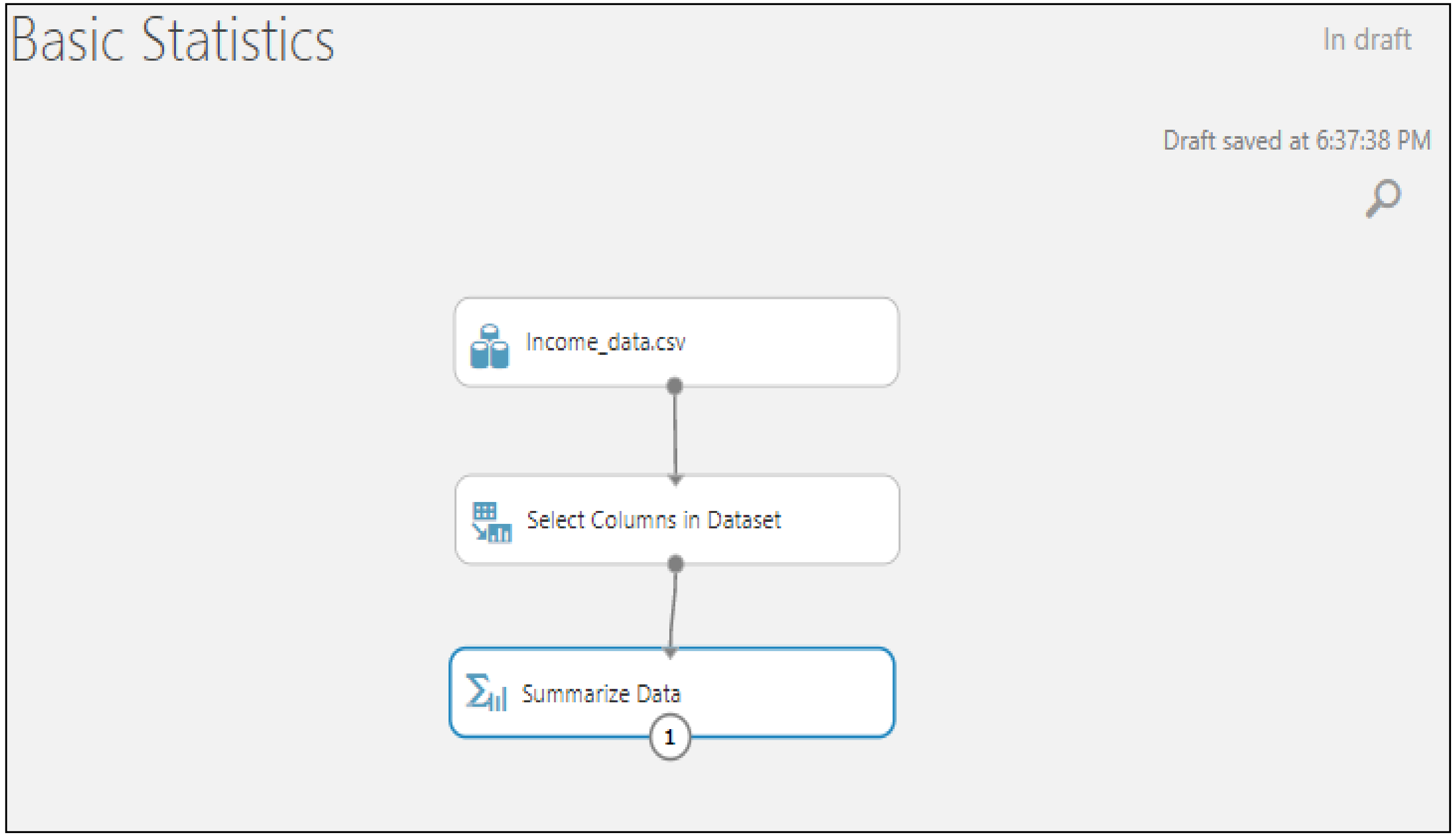
- **Quartiles**
  - Percentiles divide the whole population into 100 groups where as quartiles divide the population into 4 groups
  - p = 25: First Quartile or Lower quartile (LQ)
  - p = 50: second quartile or Median
  - p = 75: Third Quartile or Upper quartile (UQ)

Steps - Percentiles and Quartiles

- Drag and drop the dataset into the canvas
- Drag and drop select columns from dataset into the canvas
- Connect it to the dataset and select the columns
- Search for 'Summarize Data' module, drag and drop into the canvas
- Connect it to the select columns from the dataset
- Click on run
- Once Finished Running, visualize the data
- This gives the basic descriptive statistics report for the columns in a dataset (including Percentiles and Quartiles)

Steps - Percentiles and Quartiles

Fig13: Summarize Data



Steps - Percentiles and Quartiles

Fig14: Visualization (Percentiles)

P0.5	P1	P5	P95	P99	P99.5
0	0	0	5013	15024	34095

Fig15: Visualization (Quartiles)

Basic Statistics › Summarize Data › Results dataset

rows	columns
1	23

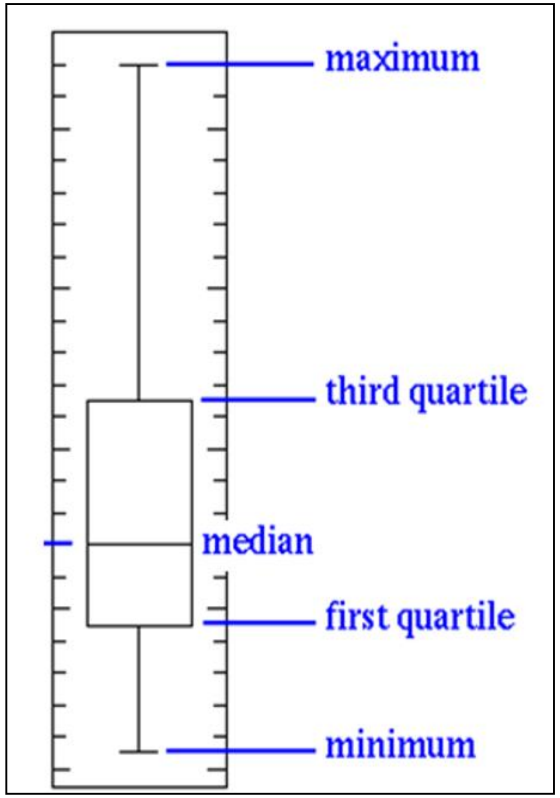
Min	Max	Mean	Mean Deviation	1st Quartile	Median	3rd Quartile	Mode	Rang
0	99999	1077.648844	1977.373437	0	0	0	0	999

Box Plots and Outlier Detection

- Box plots have box from LQ to UQ, with median marked.
- They portray a five-number graphical summary of the data Minimum, LQ, Median, UQ, Maximum
- Helps us to get an idea on the data distribution
- Helps us to identify the outliers easily
- 25% of the population is below first quartile,
- 75% of the population is below third quartile
- If the box is pushed to one side and some values are far away from the box then it’s a clear indication of outliers

Box Plots and Outlier Detection

Fig16: Box Plot



Steps - Box Plots and Outlier Detection

- Drag and drop the dataset into the canvas
- Drag and drop the split data into the canvas, join it to the dataset
- Click on the split data, in properties select mode as Regular Expression
- Give Regular Expression as \"native-country" United-States
- Click on the first output circle of split data to visualize the data
- Click on any column for which Box plot should be plotted(here Capital gain)
- Click on the box plot icon which is in the left side of the table
- In visualization we can see the Box plot

Steps - Box Plots and Outlier Detection

Fig17: Regular expression Split(United-States)

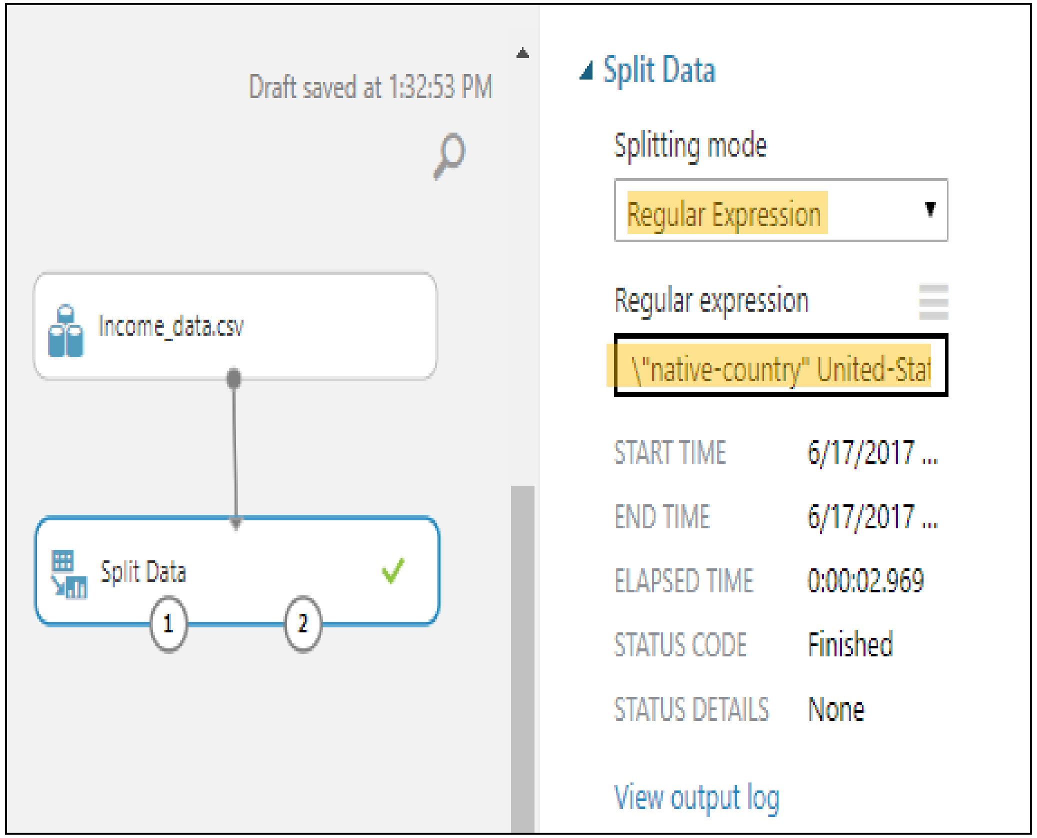
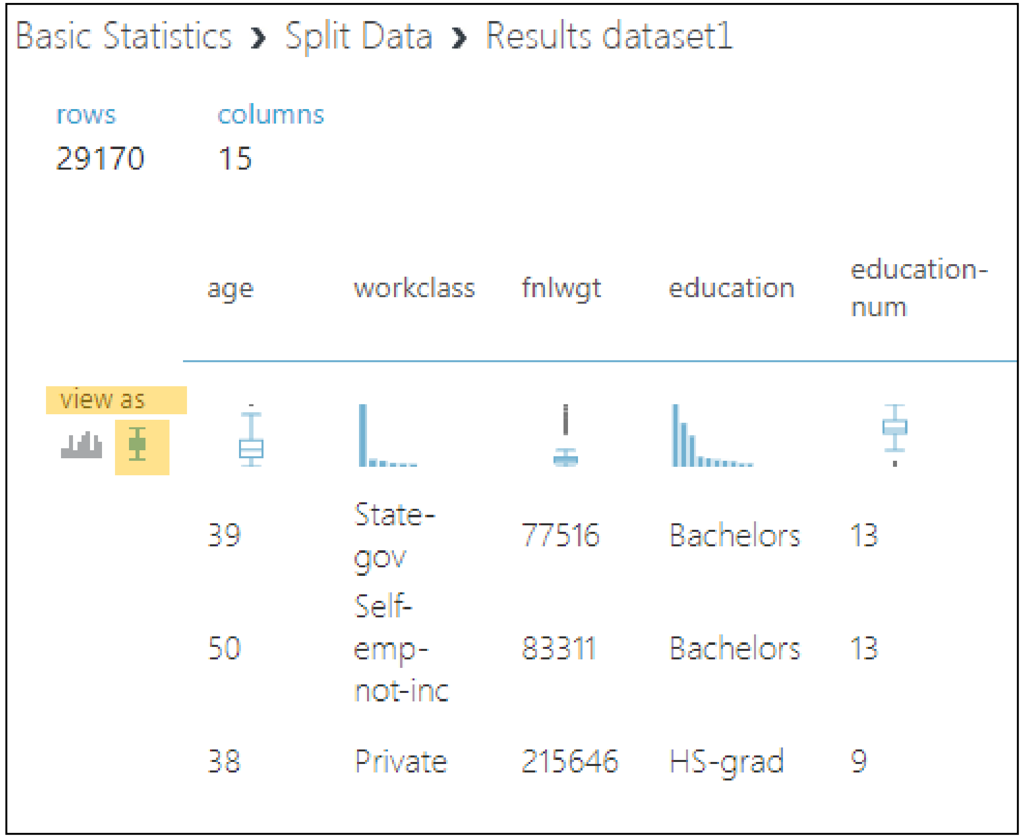


Fig18: View as(boxplot)



Steps - Box Plots and Outlier Detection

Fig19: Box Plot(United-States)

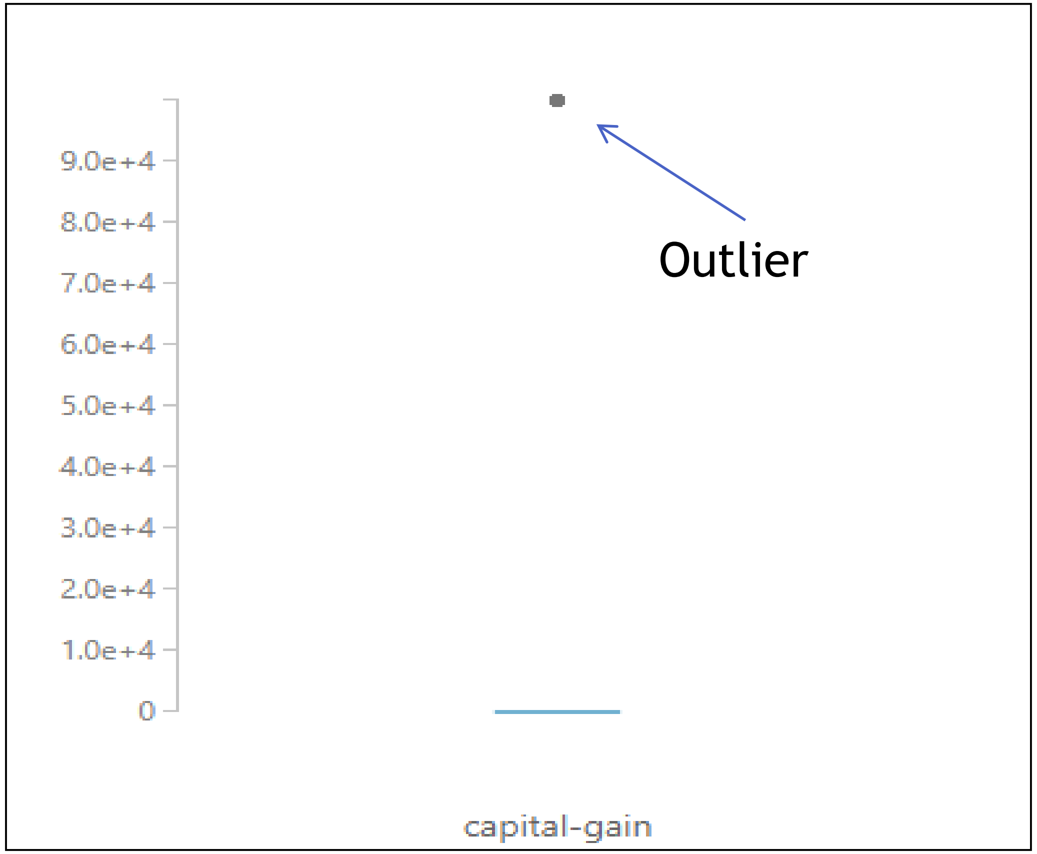
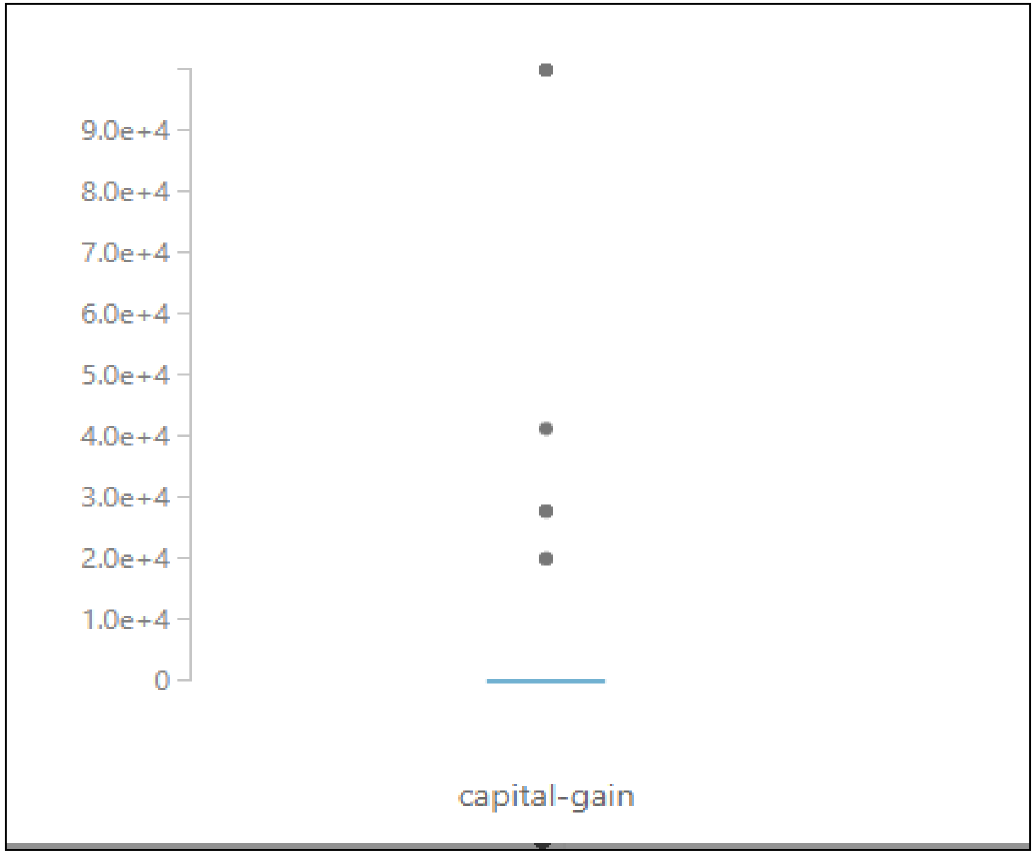


Fig20: Box Plot(Rest of Country)

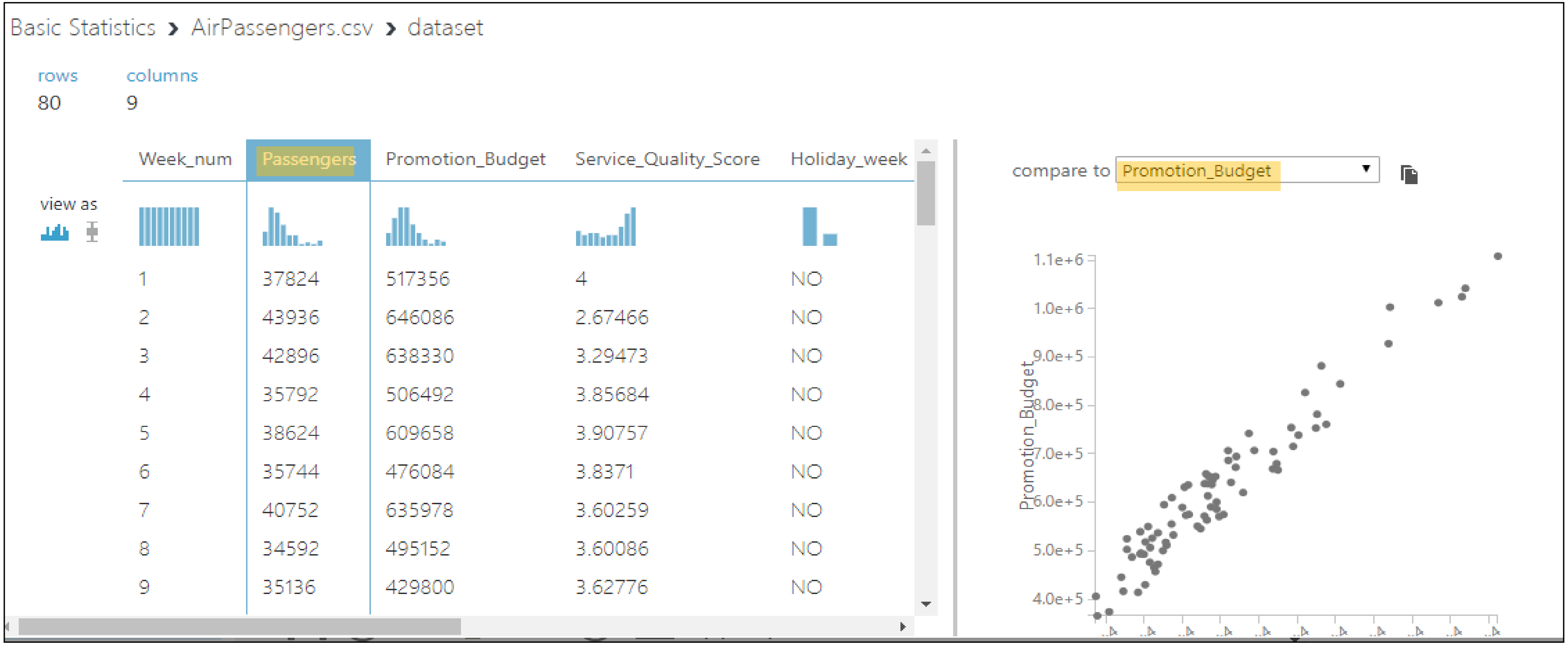


Creating Graphs

- Scatter Plot:
  - Scatter plot needs to be plot between two variables
  - It give us the relation between the two chosen variables
- Steps - **Creating Graphs**
  - Drag and drop the dataset into the canvas
  - Click the output circle to visualize the data
  - Select a column and see on visualization
  - In compare to dropdown box, select the column which to be compared
  - Scatter plot appears below

Steps - Creating Graphs

Fig21: Scatter Plot (Passengers vs Promotion_Budget)



Thank you

statinfer

We offer training on

www.statinfer.com

- Data Analytics
- Data Visualization
- Predictive Modelling
- Data Science
- Machine Learning
- Deep Learning
- R
- Python
- TensorFlow

statinfer

Part 4/12 - Data Cleaning and Preparing Data for Analysis

Venkat Reddy Konasani

Contents

- Raw Data - issues
- Data Exploration
- Data Validation
- Data Sensitization techniques

Raw Data - issues

The raw data is dirty

- Wrong formats- expenses is read as date
- Might have missing values – Income missing for some records
- Might have outliers – Number of loans is 25000
- Erroneous values – Age is less than 0
- Default values – Account tenure is 999999
- Inconsistent – Age is 25, year of birth is 1970

statinfer

Preparing data for analysis

- We can't directly start the analysis and model building with raw data.
- Before getting on to core analysis and strategy building it is very important to
  - Explore the data
  - Validate the data
  - And finally clean the data and prepare it for analysis

Case Study- Data Exploration

Give me some credit data

- We will try to understand the data exploration, validation and data cleaning using a case study on loans data
- Give me some credit data. It is loans data. Historical data are provided on 150,000 borrowers.
- The final objective is to build a model that borrowers can use to help make the best financial decisions.
- We generally get the data and data dictionary from the data team.

statinfer

Data Dictionary

No	Variable Name	Short Description	Description	Varibale Type
1	SeriousDlqin2yrs	Target Variable (loan defaulter)	Person experienced 90 days past due delinquency or worse	Y/N
2	RevolvingUtilizationOfUnsecuredLines	Credit Utilization	Total balance on credit cards and personal lines of credit except real estate and no installment debt like car loans divided by the sum of credit limits	percentage
3	age	Age	Age of borrower in years	integer
4	NumberOfTime30-59DaysPastDueNotWorse	One month late frequency	Number of times borrower has been 30-59 days past due but no worse in the last 2 years.	integer
5	DebtRatio	Debt to income ratio	Monthly debt payments, alimony,living costs divided by monthy gross income	percentage
6	MonthlyIncome	Income	Monthly income	real
7	NumberOfOpenCreditLinesAndLoans	Number of loans	Number of Open loans (installment like car loan or mortgage) and Lines of credit (e.g. credit cards)	Integer
8	NumberOfTimes90DaysLate	Three months late frequency	Number of times borrower has been 90 days or more past due.	integer
9	NumberRealEstateLoansOrLines	House loans	Number of mortgage and real estate loans including home equity lines of credit	integer
10	NumberOfTime60-89DaysPastDueNotWorse	Two months late frequency	Number of times borrower has been 60-89 days past due but no worse in the last 2 years.	integer
11	NumberOfDependents	Dependents	Number of dependents in family excluding themselves (spouse, children etc.)	integer

Steps in Data Exploration and Cleaning

- Step-1: Basic details of the data
- Step-2: Categorical variables exploration
- Step-3: Continuous variables exploration
- Step-4: Missing Values and Outlier Treatment

Step-1: Basic details of the data

Check the Metadata

- Metadata is data about data
- What are total number of observations and variables
- Check each field name, field type, Length of field
- Are there some variables which are unexpected say q9 r10?
- Are the data types and length across variables correct
- For known variables is the data type as expected (For example if age is in date format something is suspicious)

Print first few records

- Do we have any unique identifier? Is the unique identifier getting repeated in different records?
- Do the text variables have meaningful data?
- Are there some coded values in the data
- Do all the variables appear to have data? Are there any missing values

Lab: Basic contents of the data

- Import “Give me some Credit\cs-training.csv”
- What are number of rows and columns
- Are there any suspicious variables?
- Are all the variable names correct?
- Display the variable formats
- Print the first 10 observations
- Do we have any unique identifier?
- Do the text and numeric variables have meaningful data?
- Are there some coded values in the data?
- Do all the variables appear to have data

Steps - Basic contents of the data

- Drag and drop the dataset into the canvas
- Click on the output circle to visualize the data
- Check for number of Rows and Columns
- Check for suspicious variable if any, other than that in Data Dictionary
- Check the names of the variable with the variable names in the Data Dictionary
- Check for the variable formats in the statistics menu
- Is there any unique identifier, note it down if any
- Check whether the text and numeric columns have meaningful data
- Are there any coded values, note down if any
- Check whether all the variables have data

Steps - Basic contents of the data

Fig1: Visualize (cs-training.csv)

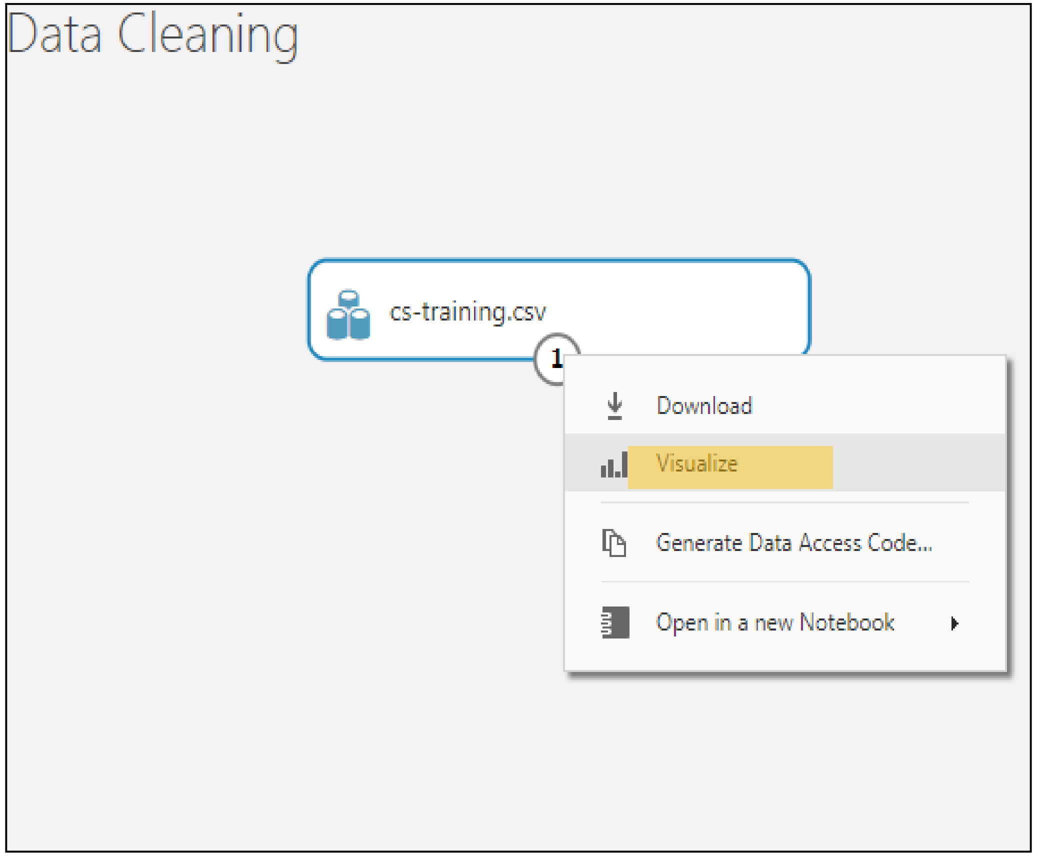
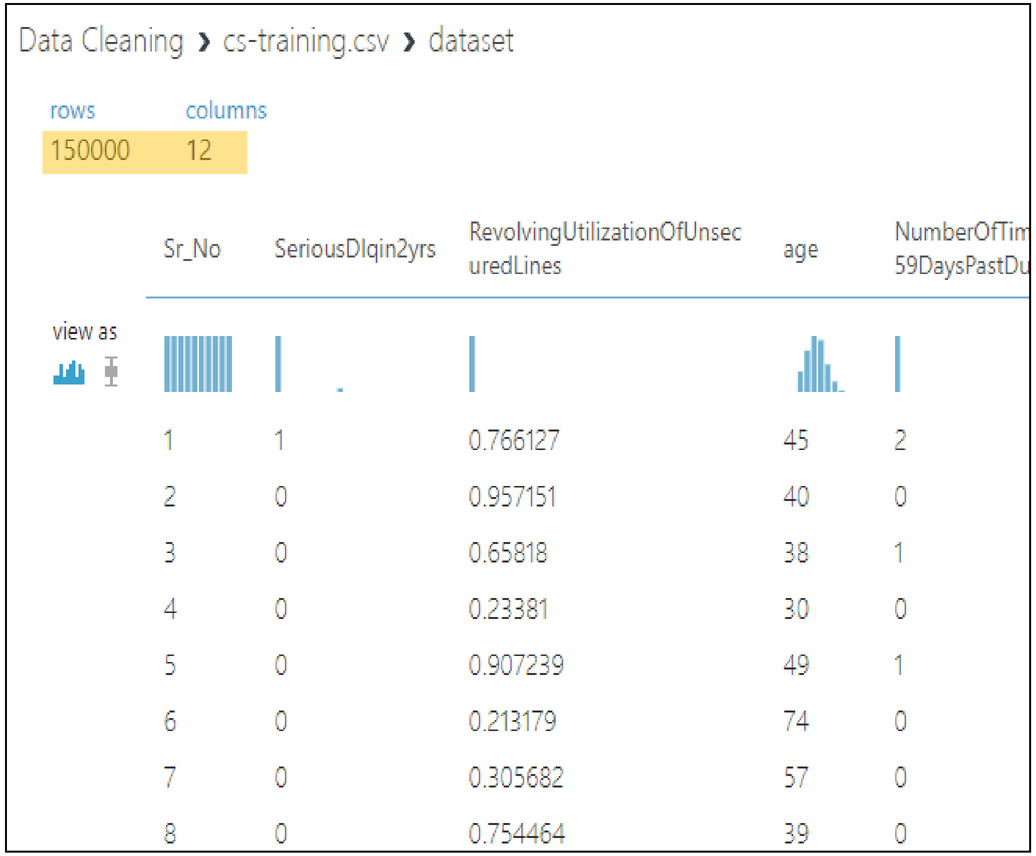
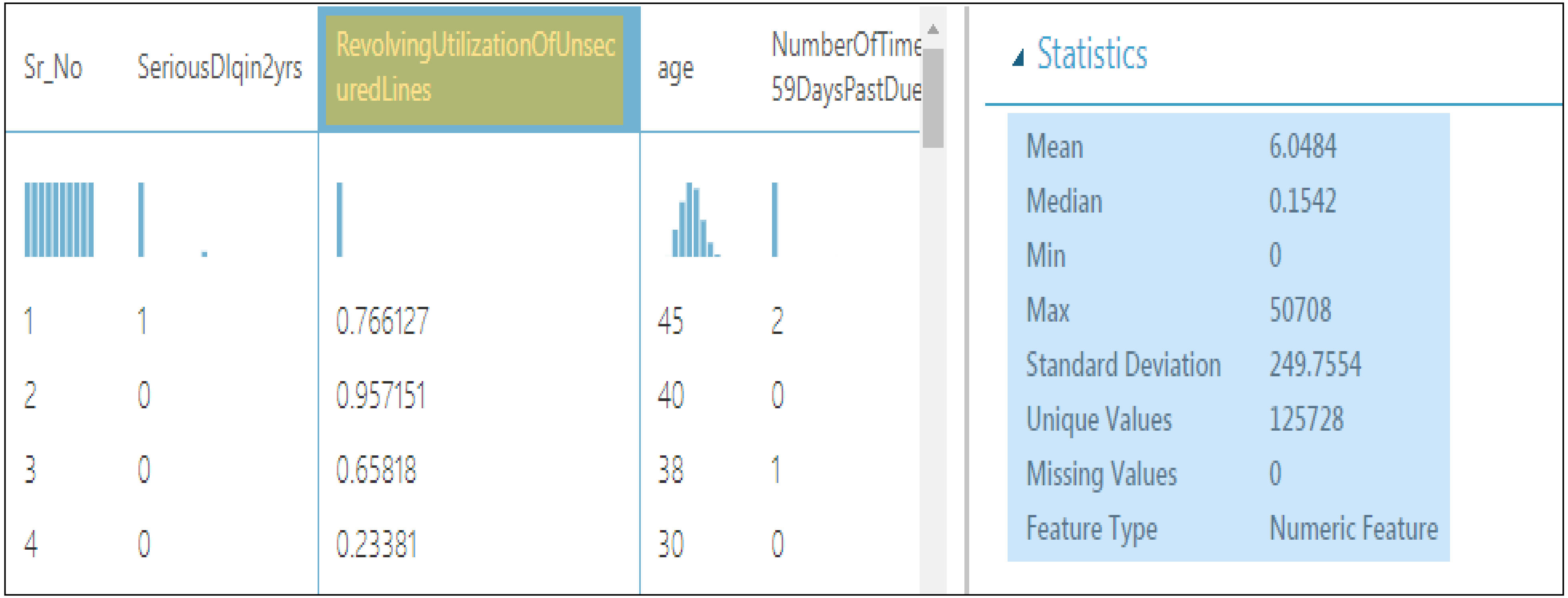


Fig2: Visualization



Steps - Basic contents of the data

Fig3: Checking Format



Note - Basic contents of the data

- New variable Sr_No but not suspicious, may be unique identifier
- The variable MonthlyIncome must be of numeric type but it is of string type (need to be treated and changed)
- We also see some missing value in the data

Step-2: Categorical variables exploration

The Frequency Table and Summary

- Calculate frequency counts cross-tabulation frequencies for Especially for categorical, discrete & class fields
- Frequencies (Histogram)
  - help us understanding the variable by looking at the values it’s taking and data count at each value.
  - They also helps us in analyzing the relationships between variables by looking at the cross tab frequencies or by looking at association

Check Points

1. Are values as expected?
2. Variable understanding : Distinct values of a particular variable, missing percentages
3. Are there any extreme values or outliers?
4. Any possibility of creating a new variable having small number of distinct category by clubbing certain categories with others.

Lab: Frequencies (Histogram)

- What are the categorical and discrete variables? What are the continues variables.
- Find the frequencies of all class variables in the data
- Are there any variables with missing values?
- Are there any default values?
- Can you identify the variables with outliers?
- Are there any variables with other issues?

Steps - Frequencies (Histogram)

- By comparing the variables with the data dictionary find out which variables are categorical or discrete and which are continuous
- Visualize the data
- Check the histogram for each and every variable
- Find whether there are any missing values, default values
- Click on box plot and check for Outliers

Steps - Frequencies (Histogram)

Fig4: SeriousDlqin2yrs

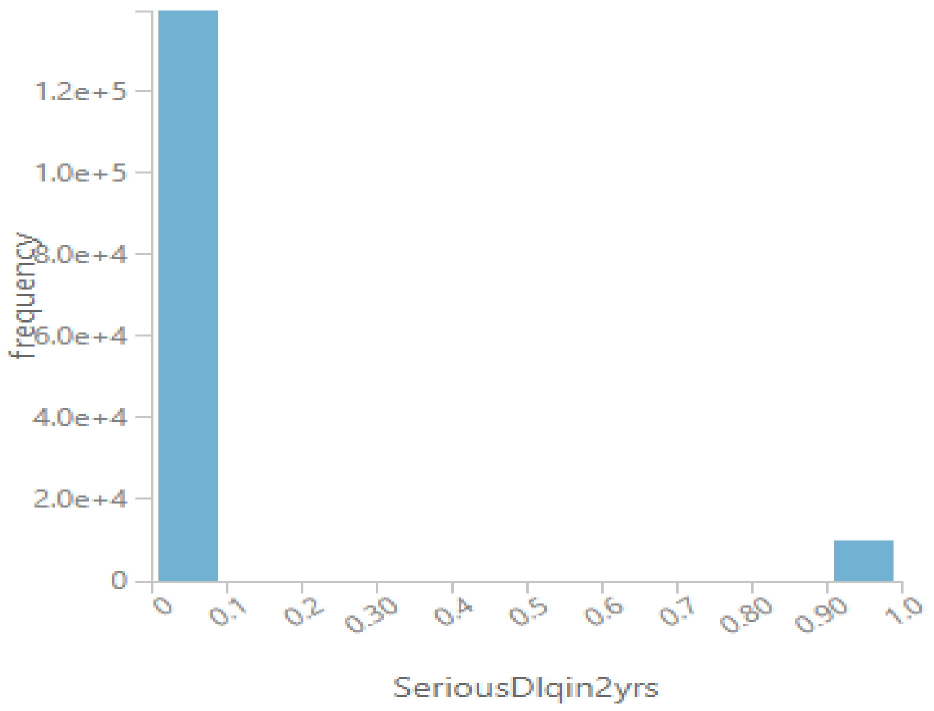
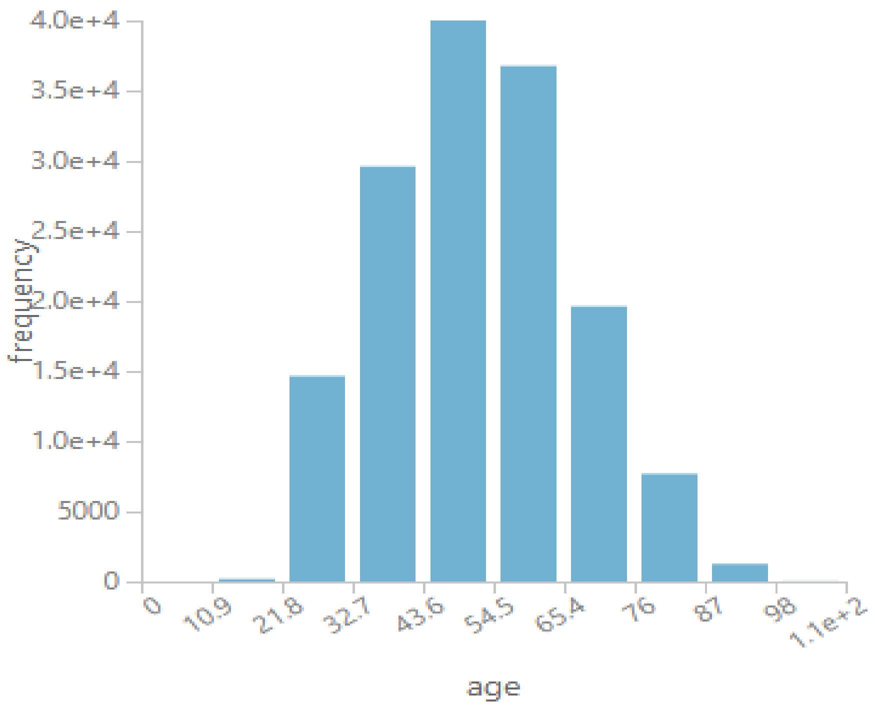


Fig5: Age



Steps - Frequencies (Histogram)

Fig6:NumberOfTime30-59DaysPastDueNotWorse

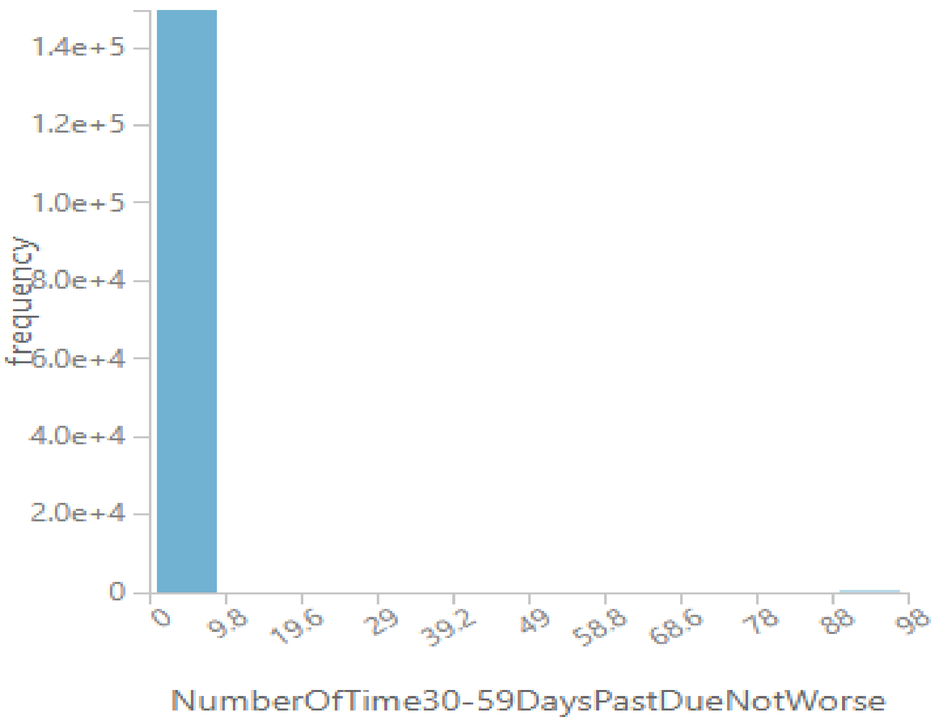
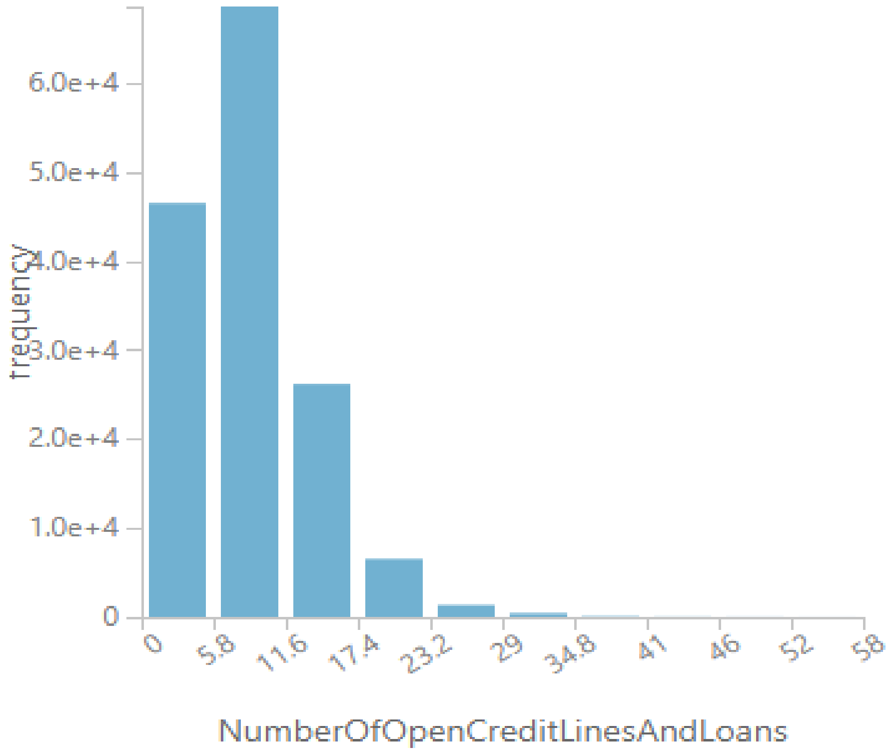


Fig7: NumberOfOpenCreditLinesAndLoans



Steps - Frequencies (Histogram)

Fig8: NumberOfTimes90DaysLate

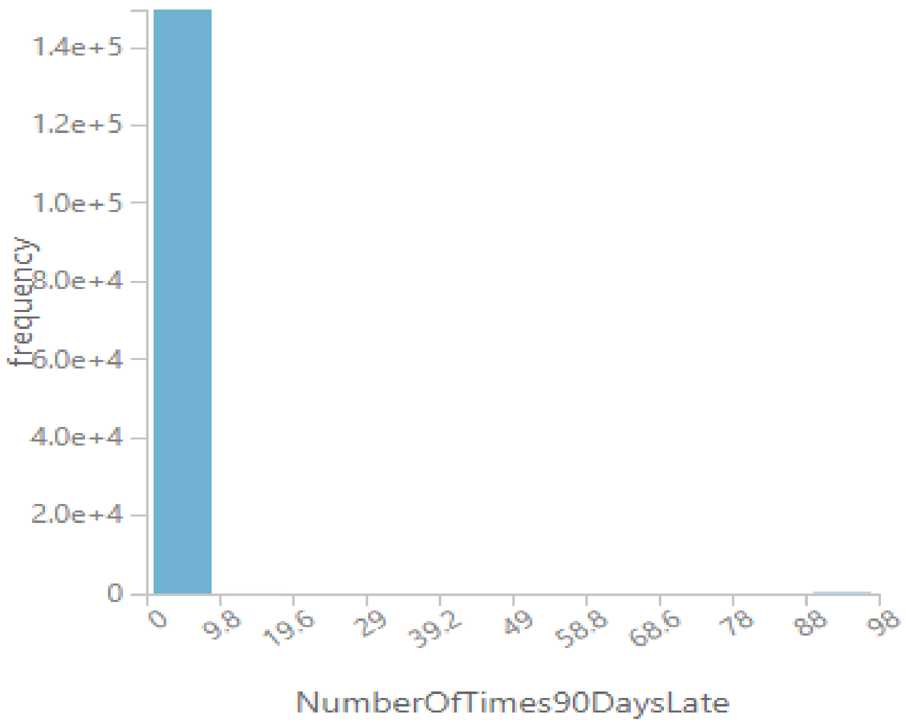
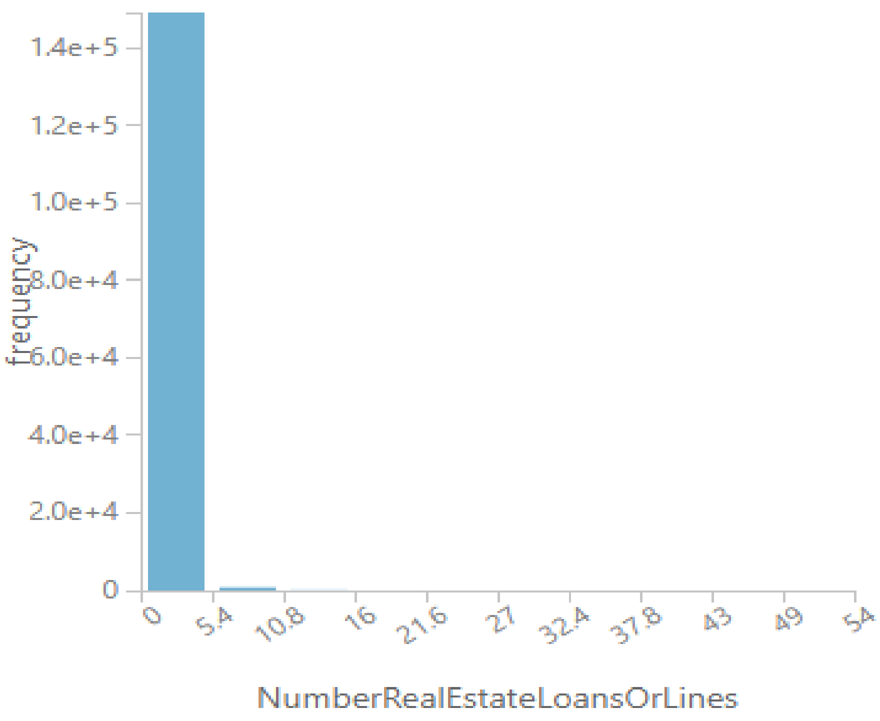


Fig9: NumberRealEstateLoansOrLines



Steps - Frequencies (Histogram)

Fig10: NumberOfTime60-89DaysPastDueNotWorse

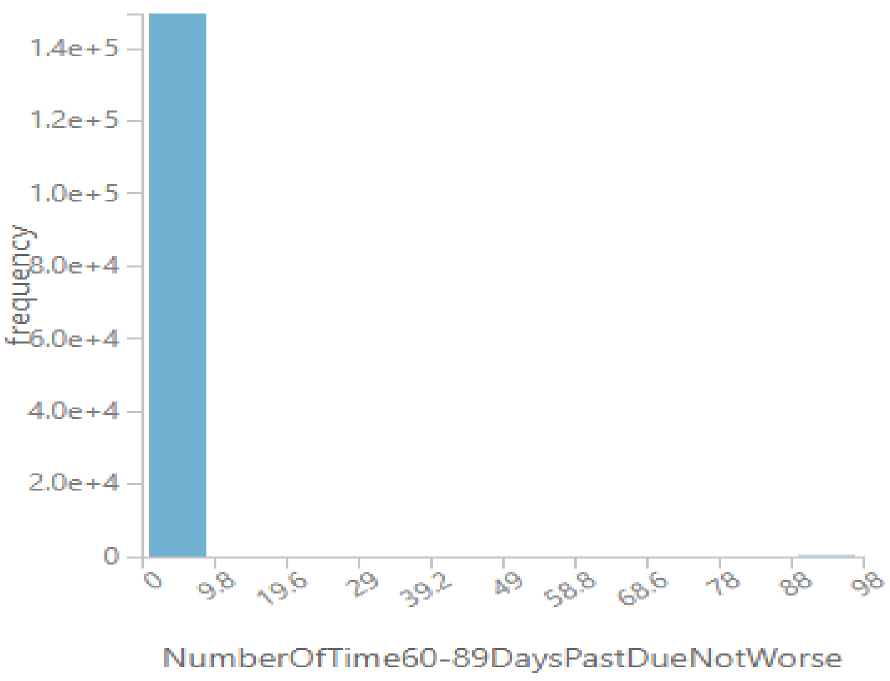
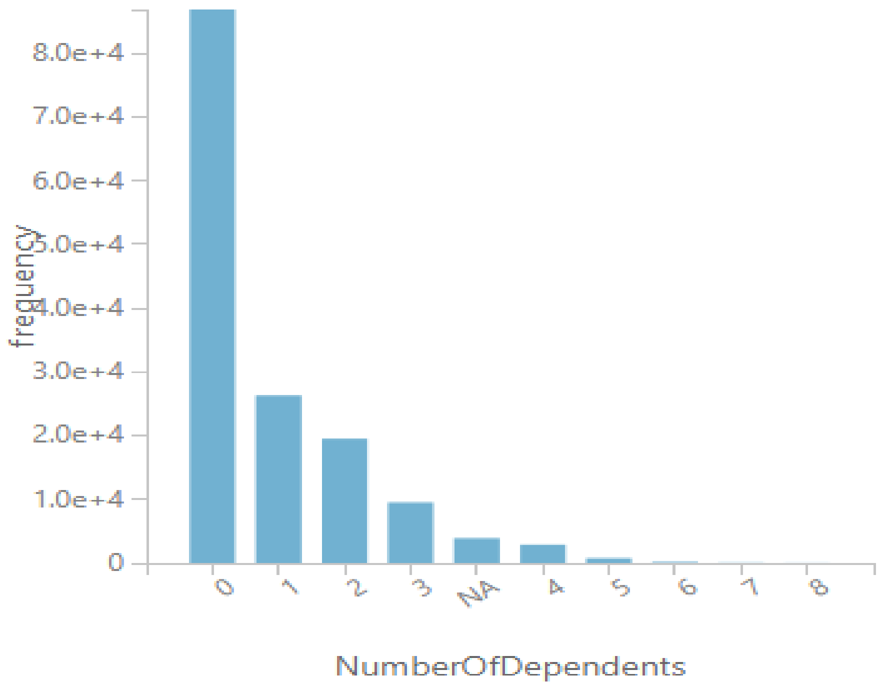


Fig11: NumberOfDependents



Notes - Frequencies (Histogram)

- There are outliers in some variables
- There are some missing values
- Other issues: the variables
  - NumberOfTime30-59DaysPastDueNotWorse
  - NumberOfOpenCreditLinesAndLoans
  - NumberOfTimes90DaysLate
  - NumberRealEstateLoansOrLines and
  - NumberOfTime60-89DaysPastDueNotWorse should be of categorical type, but by seeing the histogram of these variables we can see that they are continuous

Treating MonthlyIncome

- We know that the MonthlyIncome variable must be of numeric type, but it is shown as string (because of NA values)
- To overcome this we change NA to 0
- Now change MonthlyIncome to integer
- Steps:
  - Drag and drop the dataset
  - Search for Convert to dataset, drag and drop into the canvas
  - Connect it to the dataset
  - Click on Convert to dataset, in properties select Action → ReplaceValues, Replce→Custom, Custom value → NA, New value → 0
  - Drag and drop Edit Metadata, connect Convert to dataset to this
  - In properties select column → MonthlyIncome and data type a → Integer
  - Click on run, when we visualize the output of Edit Metadata we can see MonthlyIncome type as Numeric and the NA values changed to 0

Treating Other Issues

- Drag and drop another Edit Metadata
- Connect it to the previous Edit Metadata
- In properties, select the variables which should be changed to Categorical
- Data type → unchanged, Categorical → MakeCategorical, Fields → unchanged and leave New column names blank
- Click on run
- Once finished running, visualize the output of Edit Metadata and check the Histogram of the changed varibles

Treating Variables

Fig12: Treating Variables

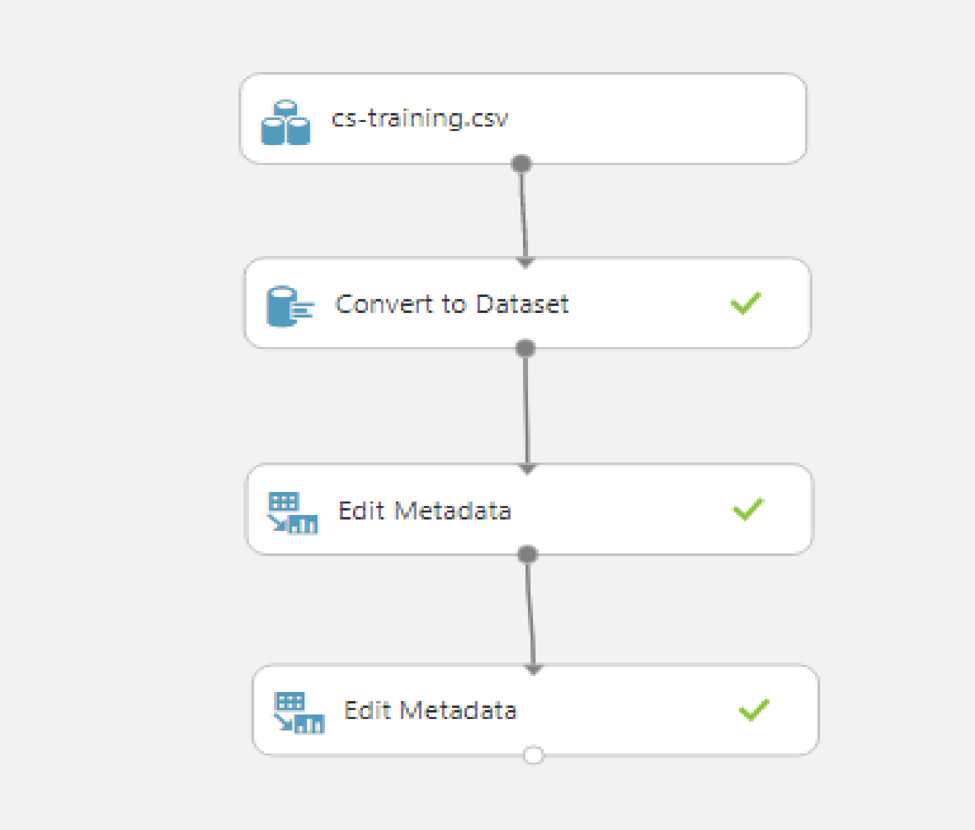
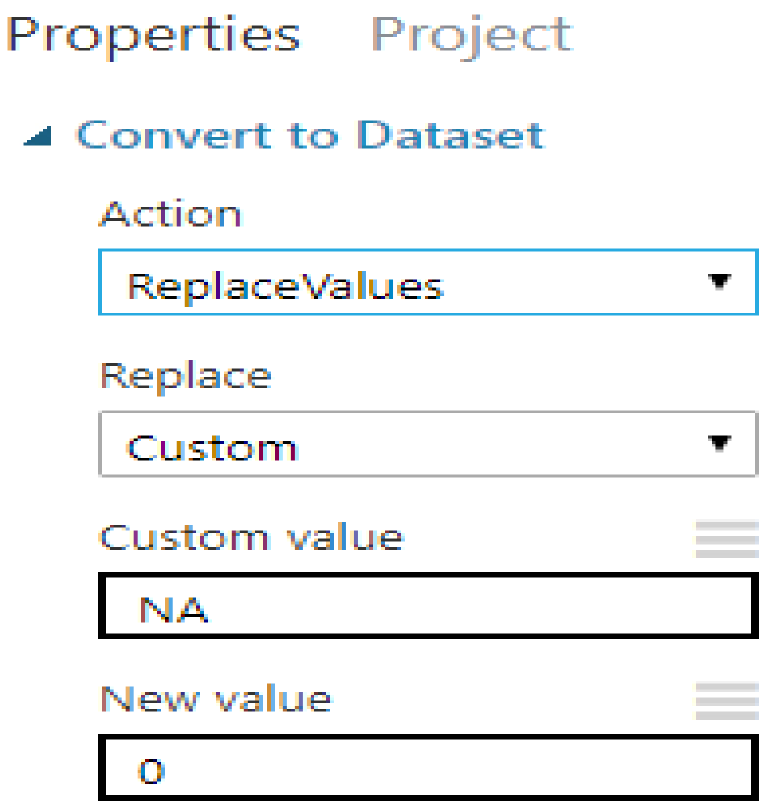


Fig13: Convert to Dataset



Treating Variables

Fig14: MonthlyIncome(string to integer)

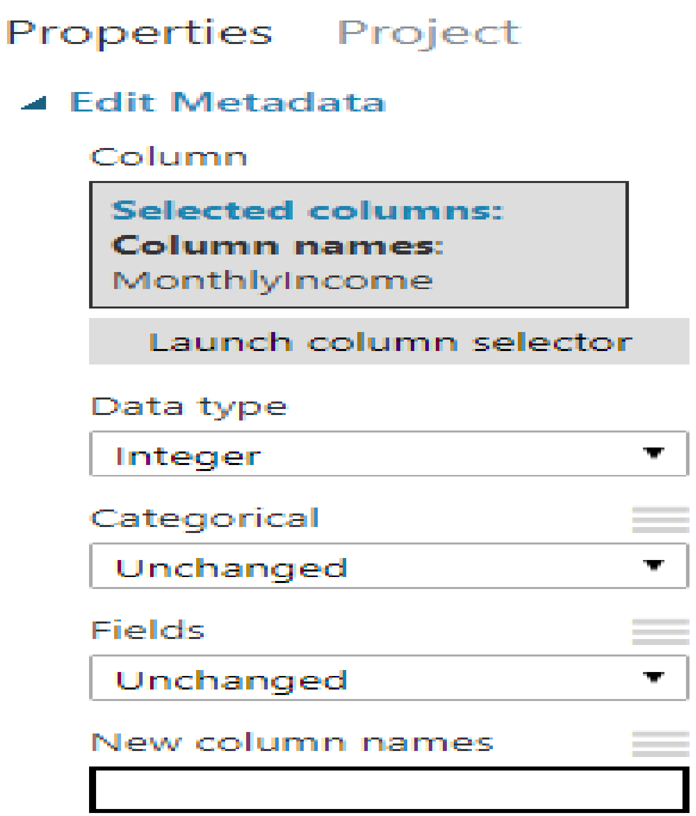
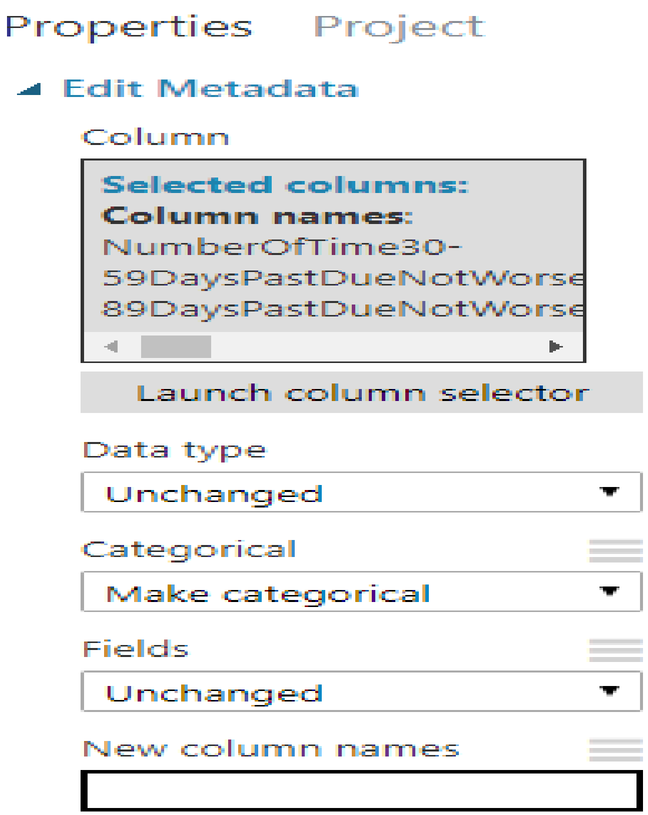


Fig15: Treating Other Issues(Categorical Variables)



statinfer

Step-3: Continuous variables exploration

Summary of Continuous variables

- Min, Max, Median, Mean, sd, Var
- Quartiles
- Box plots and identification of outliers
- Percentiles– P1, p5,p10,q1(p25),q3(p75), p90,p99

Check Points

- Are variable distribution as expected.
- What is the central tendency of the variable? Mean, Median and Mode across each variable
- Is the concentration of variables as expected ? What are quartiles?
- Indicates variables which are unary I.e stddev=0 ; the variables which are useless for the current objective.
- Are there any outliers / extreme values for the variable?
- Are outlier values as expected or they have abnormally high values - for ex for Age if max and p99 values are 10000. Then should investigate if it’s the default value or there is some error in data
- What is the % of missing value associated with the variable?

LAB: Continuous variables summary

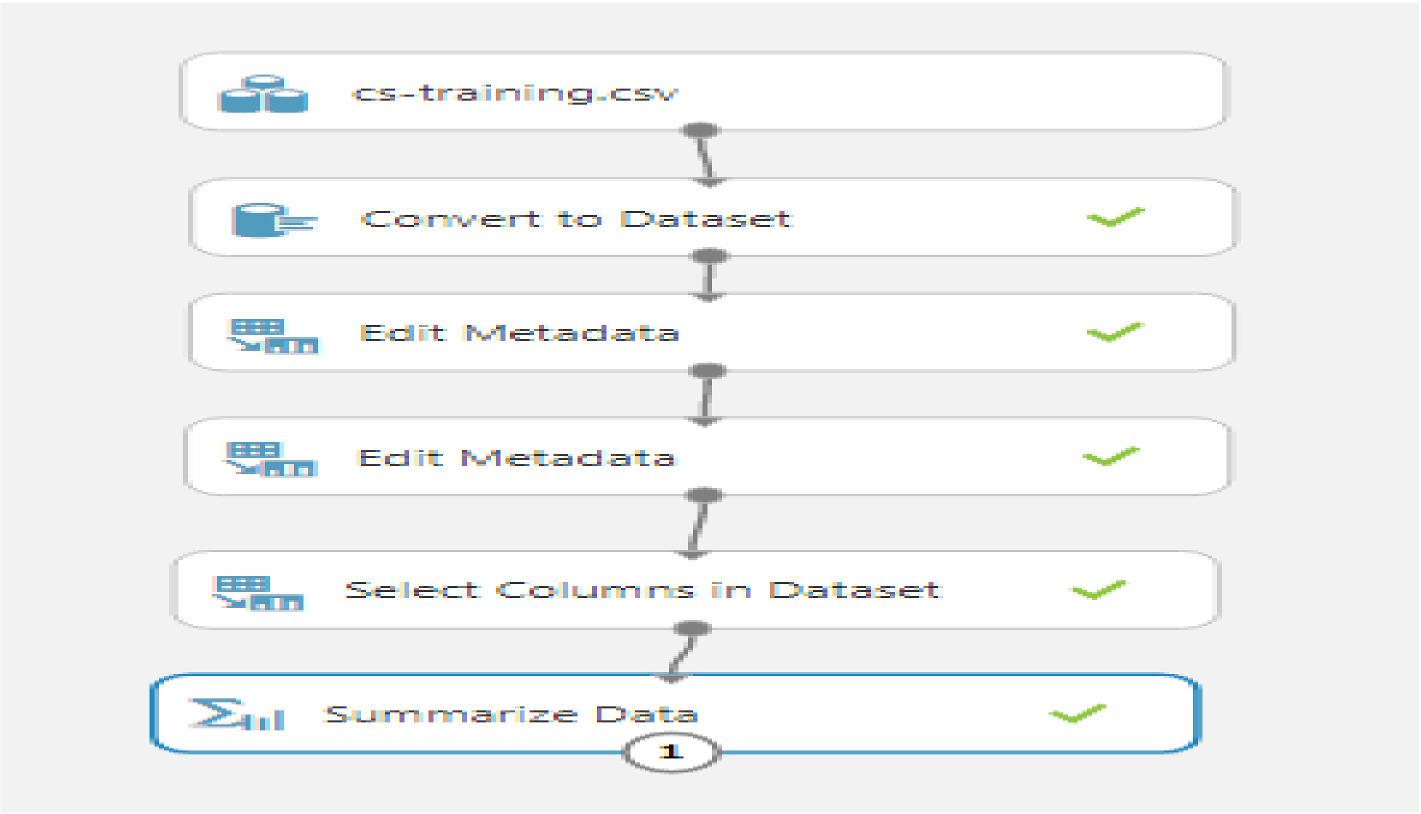
- List down the continuous variables
- Find summary statistics for each variable. Min, Max, Median, Mean, sd, Var
- Find Quartiles for each of the variables
- Create Box plots and identify outliers
- Find the percentage of missing values
- Find Percentiles and find percentage of outliers, if any P1, p5,p10,q1(p25),q3(p75), p90,p99

Steps - Continuous variables summary

- Select the columns which are continuous using Select columns from data and connect it to the previous Edit Metadata
- Drag and drop Summarize Data into canvas and connect it to the select column from data
- Click on Run
- Once finished running click on the output circle of Summarize Data
- This gives Min, Max, Median, Mean, sd, Var, Quartiles and Percentiles
- Visualize the Edit Metadata, click on box plot and check for the outliers

Steps - Continuous variables summary

Fig16:Adding Summarize Data



Steps - Continuous variables summary

Fig17: Visualising the Statistics

Data Cleaning › Summarize Data › Results dataset

rows	columns
2	23

view as

Feature	Count	Unique Value Count	Missing Value Count	Min	Max	Max	Mean	Mean Deviation	1st Quartile	Median	3rd Quartile	Mode	Range
RevolvingUtilizationOfUnsecuredLines	150000	125728	0	0	50708	50708	6.048438	11.43306	0.029867	0.154181	0.559046	0	50708
MonthlyIncome	150000	13594	0	0	300875(	3008750	5348.13892	3811.641839	1550	4357.5	7400	0	3008750

P0.5	P1	P5	P95	P99	P99.5
0	0	0	1	1.092956	1.366269
0	0	0	13500	23000	31250

Steps - Continuous variables summary(outliers)

Fig18: Visualize

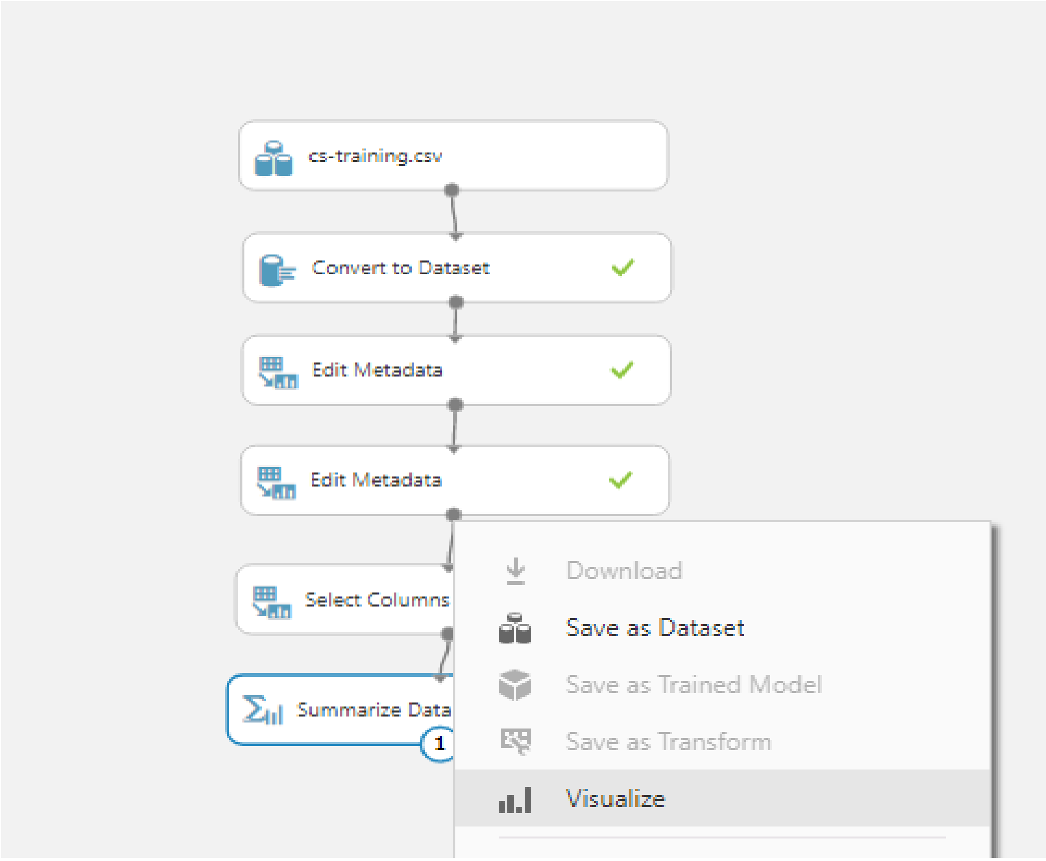
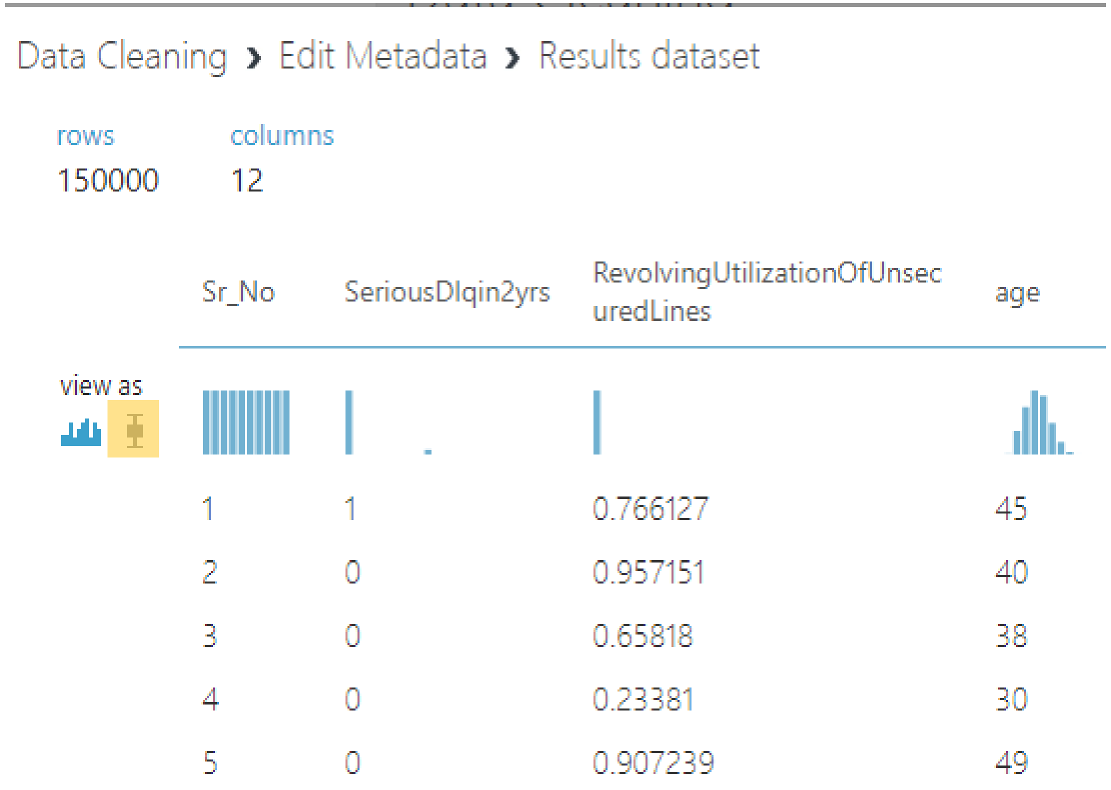


Fig19: Selecting Box Plot



Steps - Continuous variables summary(outliers)

Fig20: **BoxPlot**(RevolvingUtilizationOfUnsecuredLines)

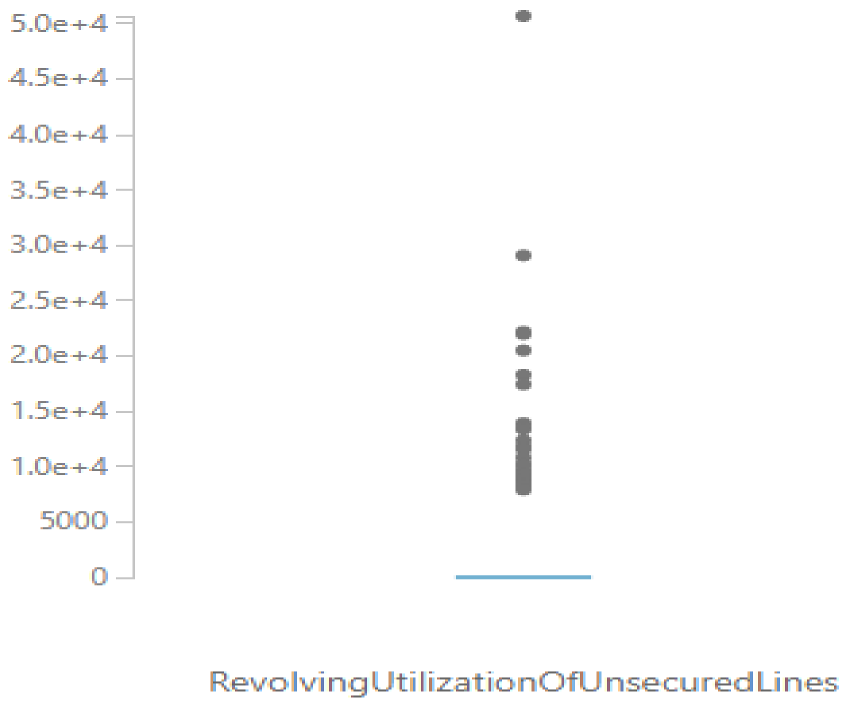


Fig21: **BoxPlot**(MonthlyIncome)

Data Cleaning

Data Cleaning

- Some variables contain outliers
- Some variables have default values
- Some variables have missing values

- RevolvingUtilizationOfUnsecuredL
- NumberOfTime30_59DaysPastDueNotW
- Monthly income has missing values

- Shall we delete them and go ahead with our analysis?

Missing values & Outliers

- Data is not always available E.g., many tuples have no recorded value for several attributes, such as customer income in sales data
- Missing data may be due to
  - Equipment malfunction
  - Inconsistent with other recorded data and thus deleted
  - Data not entered due to misunderstanding
  - Certain data may not be considered important at the time of entry
  - Not register history or changes of the data
- Missing data may need to be inferred.
- Missing data - values, attributes, entire records, entire sections
- Missing values and defaults are indistinguishable

Imputation

Missing Value Imputation1

- **Standalone imputation**
  - Mean, median, other point estimates
  - Convenient, easy to implement
  - **Assume:** Distribution of the missing values is the same as the non-missing values.
  - Does not take into account inter-relationships

  - **Eg:** The average of available values is 11.4. Can we replace the missing value in this table by **11.4** ?

X1
11.0
11.1
11.9
10.9
10.8
.
11.5
11.6
11.6
11.4
11
12
11.8
11.4
11.9

Missing Value Imputation2

statinfer

- Use attribute relationships
- Better imputation
- Two techniques
  - Propensity score (nonparametric). Useful for discrete variables
  - Regression (parametric)

- There are two missing values in x2. What are the most appropriate replacements

X1	X2
-4	-12
2	6
-6	-18
8	24
-1	
-4	-12
-5	-15
4	12
-4	-12
-5	-15
-2	
4	12
10	30
-10	-30
-3	-9

Missing Value Imputation3

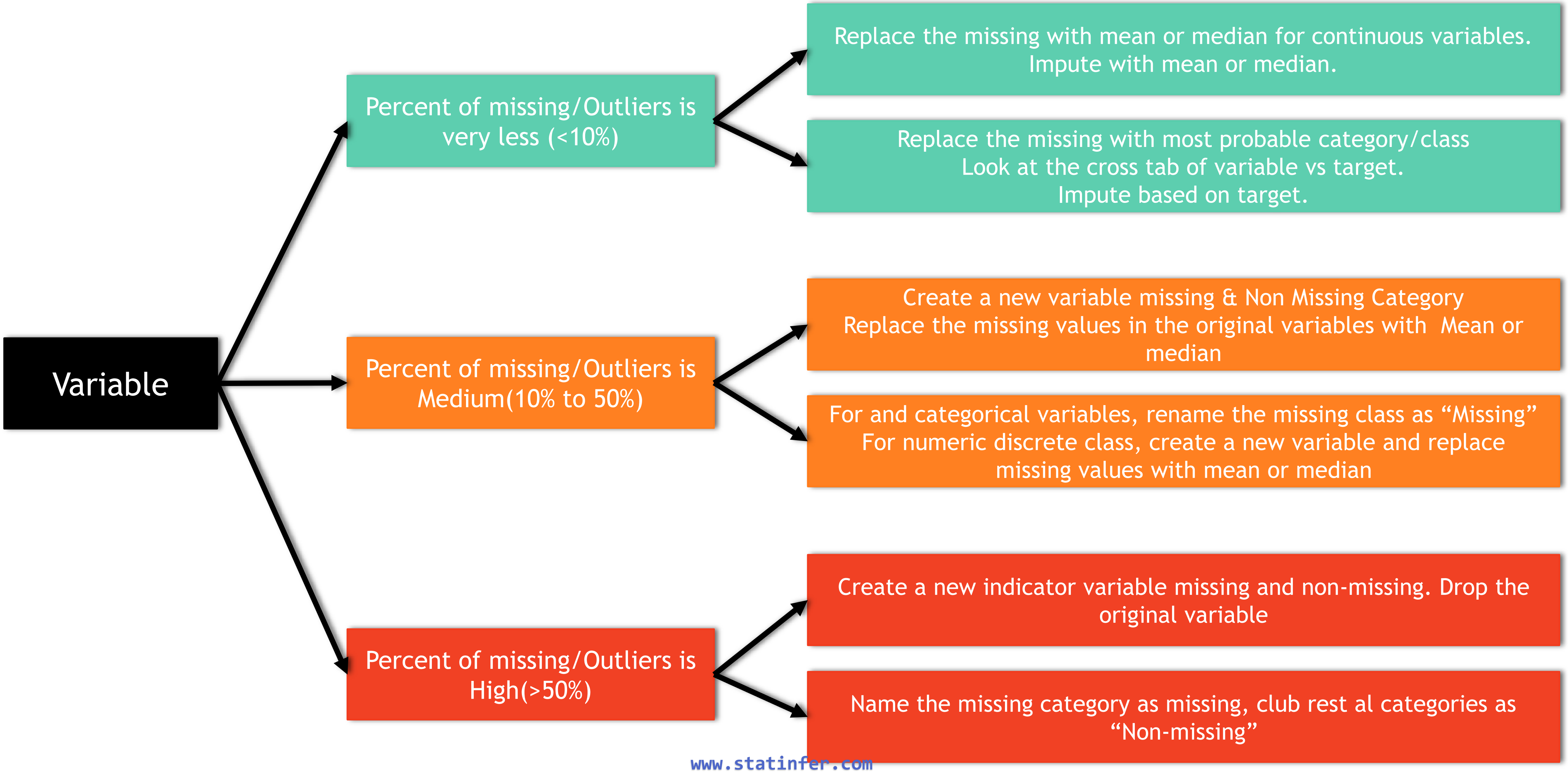
- There are two missing values in x2. Find the most appropriate replacements

X1	X2
4	1
5	1
4	1
3	1
3	1
4	1
5	1
3	
31	0
39	0
32	0
37	0
32	0
32	0
32	

Missing Value Imputation4

- What if more than 50% are missing?
- It doesn't make sense to carry out the analysis on 20% or 30% of the whole data and give inferences on overall data
- The best imputation is ignore the actual values and take available or not available info

Step-4: Missing Values and Outlier Treatment

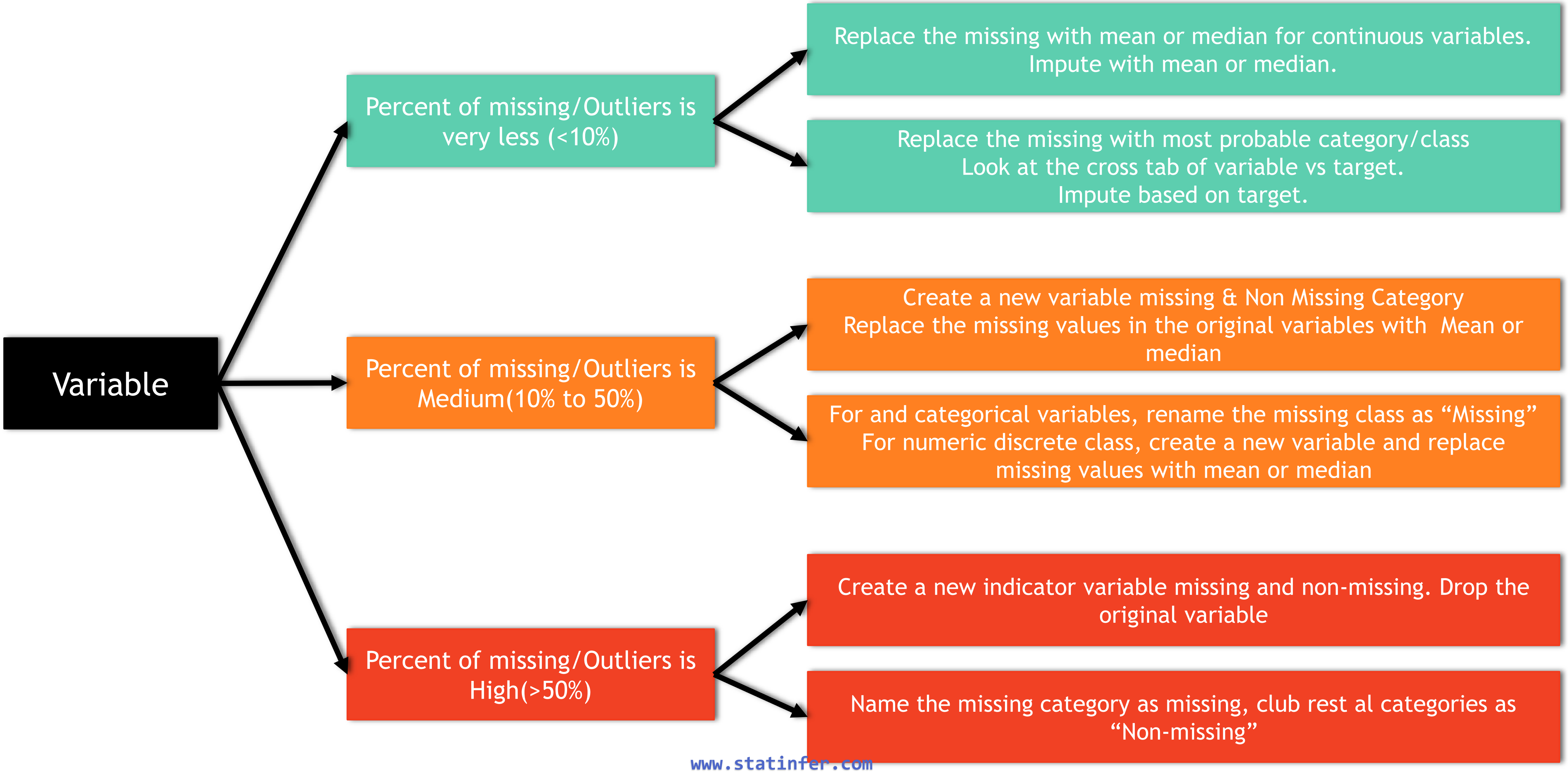

Variable
Percent of missing/Outliers is very less (<10%)
Replace the missing with mean or median for continuous variables. Impute with mean or median.
Replace the missing with most probable category/class
Look at the cross tab of variable vs target.
Impute based on target.
Percent of missing/Outliers is Medium(10% to 50%)
Create a new variable missing & Non Missing Category
Replace the missing values in the original variables with Mean or median
For and categorical variables, rename the missing class as “Missing”
For numeric discrete class, create a new variable and replace missing values with mean or median
Percent of missing/Outliers is High(>50%)
Create a new indicator variable missing and non-missing. Drop the original variable
Name the missing category as missing, club rest al categories as “Non-missing”

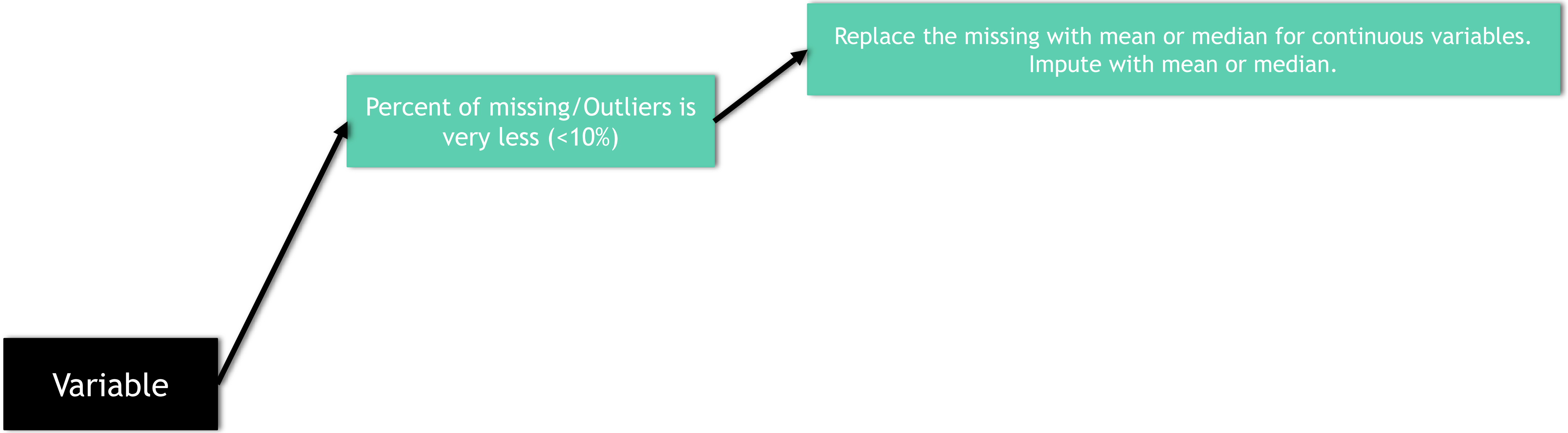
Data Cleaning Scenario-1

RevolvingUtilizationOfUnsecuredLines

- RevolvingUtilizationOfUnsecuredLines has outliers.
- What type of variable is this? What are the possible values?

- Its' mean is 6.05 which is greater than 1. So variable has some faulty values. Its maximum value is 50710 which is way too high.
- Lets look at percentiles to know from where it is exceeding 1.


Variable
Percent of missing/Outliers is very less (<10%)
Replace the missing with mean or median for continuous variables. Impute with mean or median.
Replace the missing with most probable category/class Look at the cross tab of variable vs target. Impute based on target.
Percent of missing/Outliers is Medium(10% to 50%)
Create a new variable missing & Non Missing Category Replace the missing values in the original variables with Mean or median
For and categorical variables, rename the missing class as "Missing" For numeric discrete class, create a new variable and replace missing values with mean or median
Percent of missing/Outliers is High(>50%)
Create a new indicator variable missing and non-missing. Drop the original variable
Name the missing category as missing, club rest al categories as "Non-missing"


Replace the missing with mean or median for continuous variables.
Impute with mean or median.
Percent of missing/Outliers is very less (<10%)
Variable

Data Cleaning

- RevolvingUtilizationOfUnsecuredLines has outliers.
- Since outliers percentage is less than 10% We will replace outliers with mean of reaming data.
- Outliers are with value greater than 1.

LAB: Data Cleaning Scenario-1

- What percent are missing values in RevolvingUtilizationOfUnsecuredLines?
- Get the detailed percentile distribution
- Clean the variable, and create a new variable by removing all the issues

Steps - Data Cleaning Scenario-1

- Drag and drop the dataset into the canvas
- Search for Clip Values, drag and drop into the canvas
- In properties, select Set of thresholds → ClipPeaks, Upper threshold → Constant, Constant value for upper threshold → 1, Upper substitute value → Median
- Select the columns(RevolvingUtilizationOfUnsecuredLines), uncheck the Overwrite flag, check the Add indicator columns
- Click on run
- Once finished running, click to visualize the data
- A new column is added in which the outliers are replaced with median(RevolvingUtilizationOfUnsecuredLines_clipped_value)
- By checking the Box Plot for both the variables we identify that the outliers are replaced

Steps - Data Cleaning Scenario-1

Fig22: Clip Values

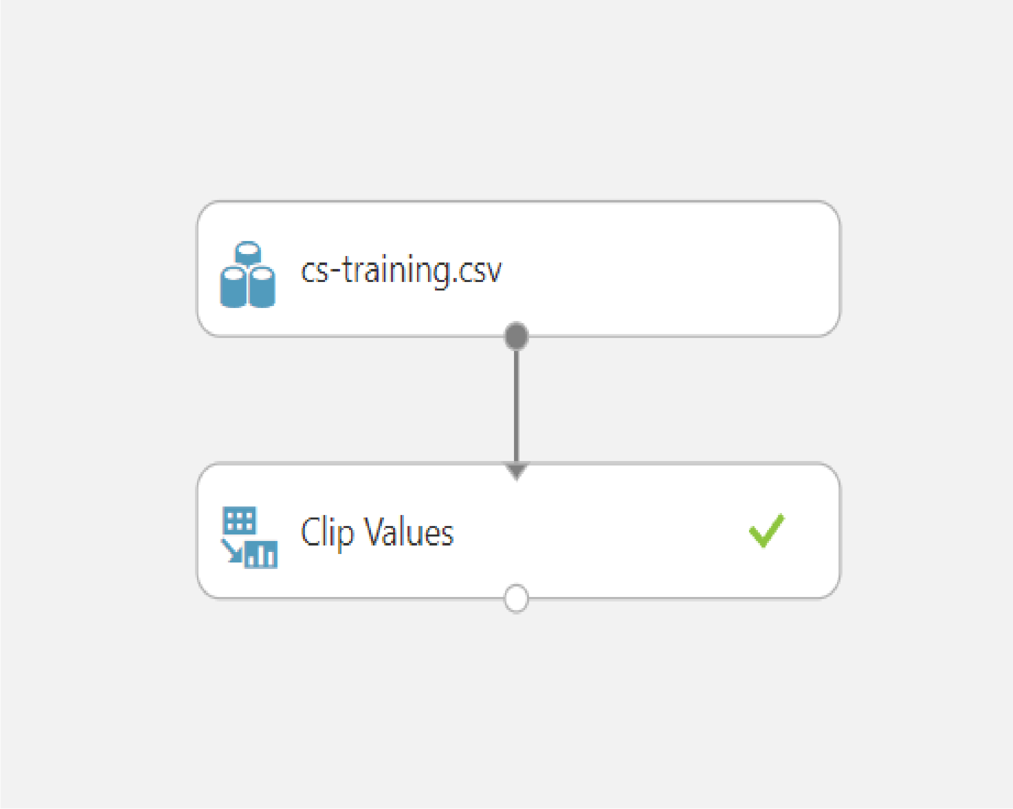
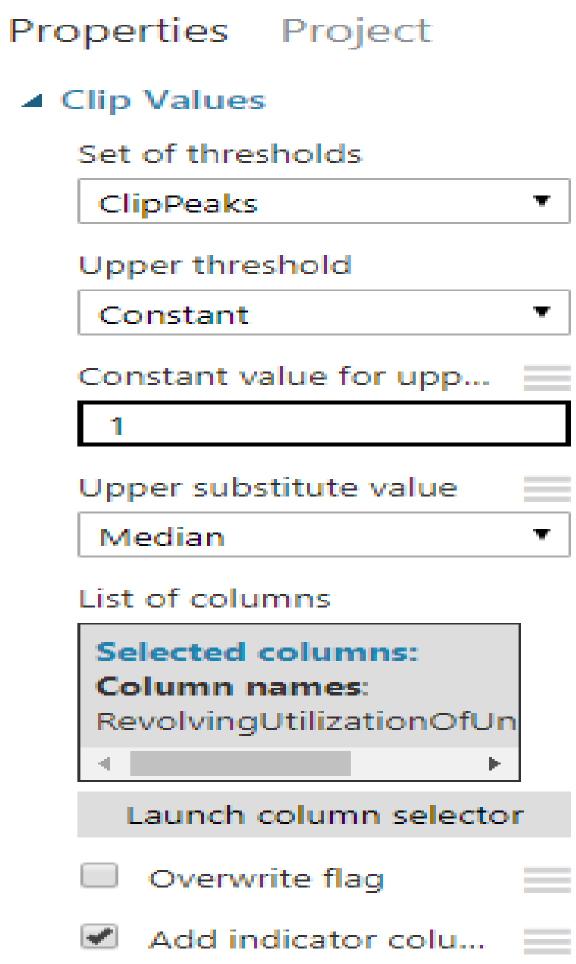


Fig23: Clip Value (Properties)



Steps - Data Cleaning Scenario-1

Fig24: Box Plot (before clipping)

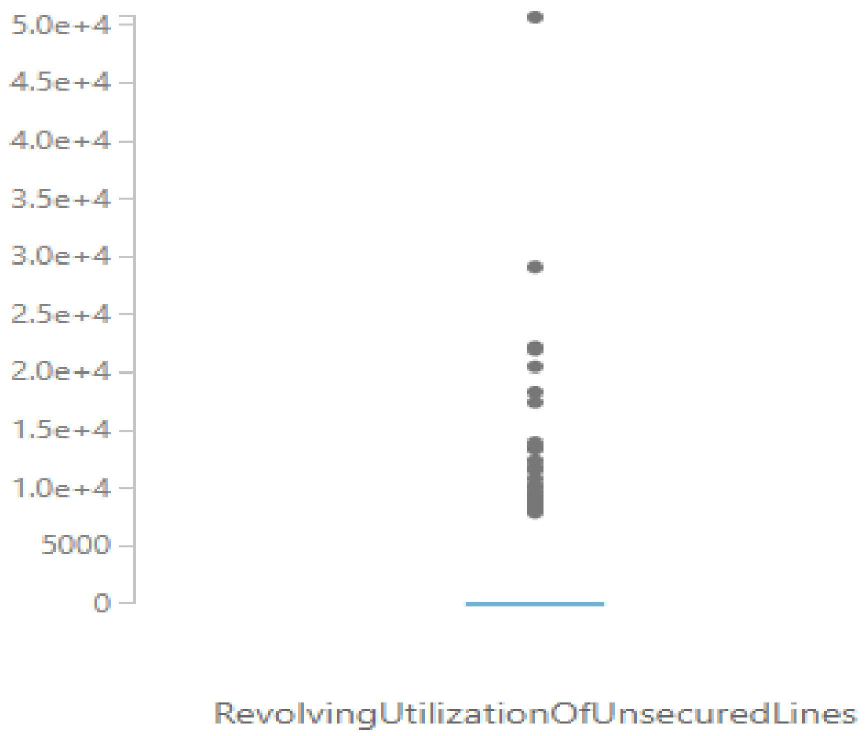
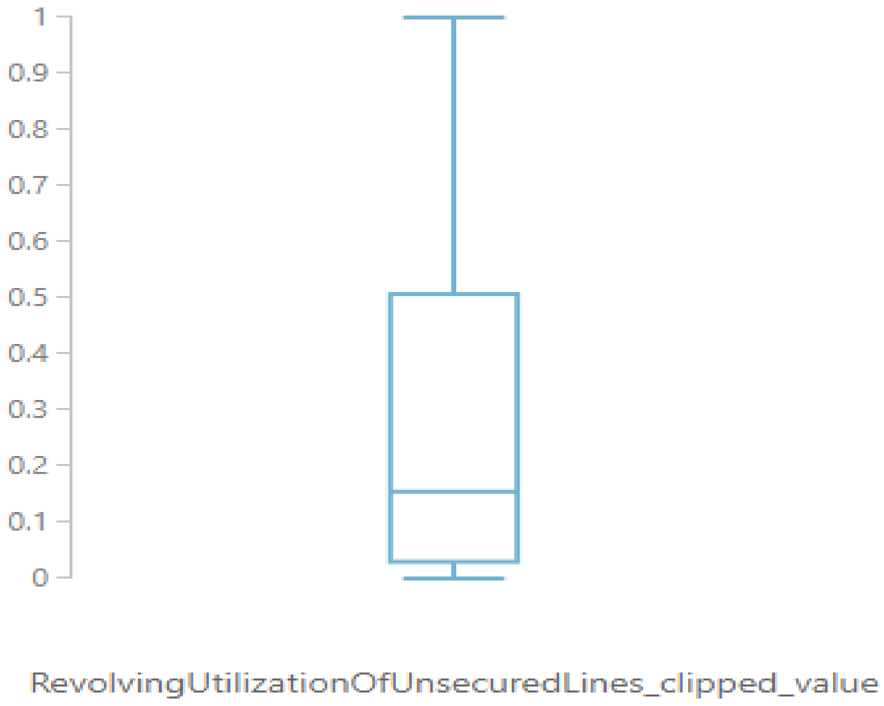


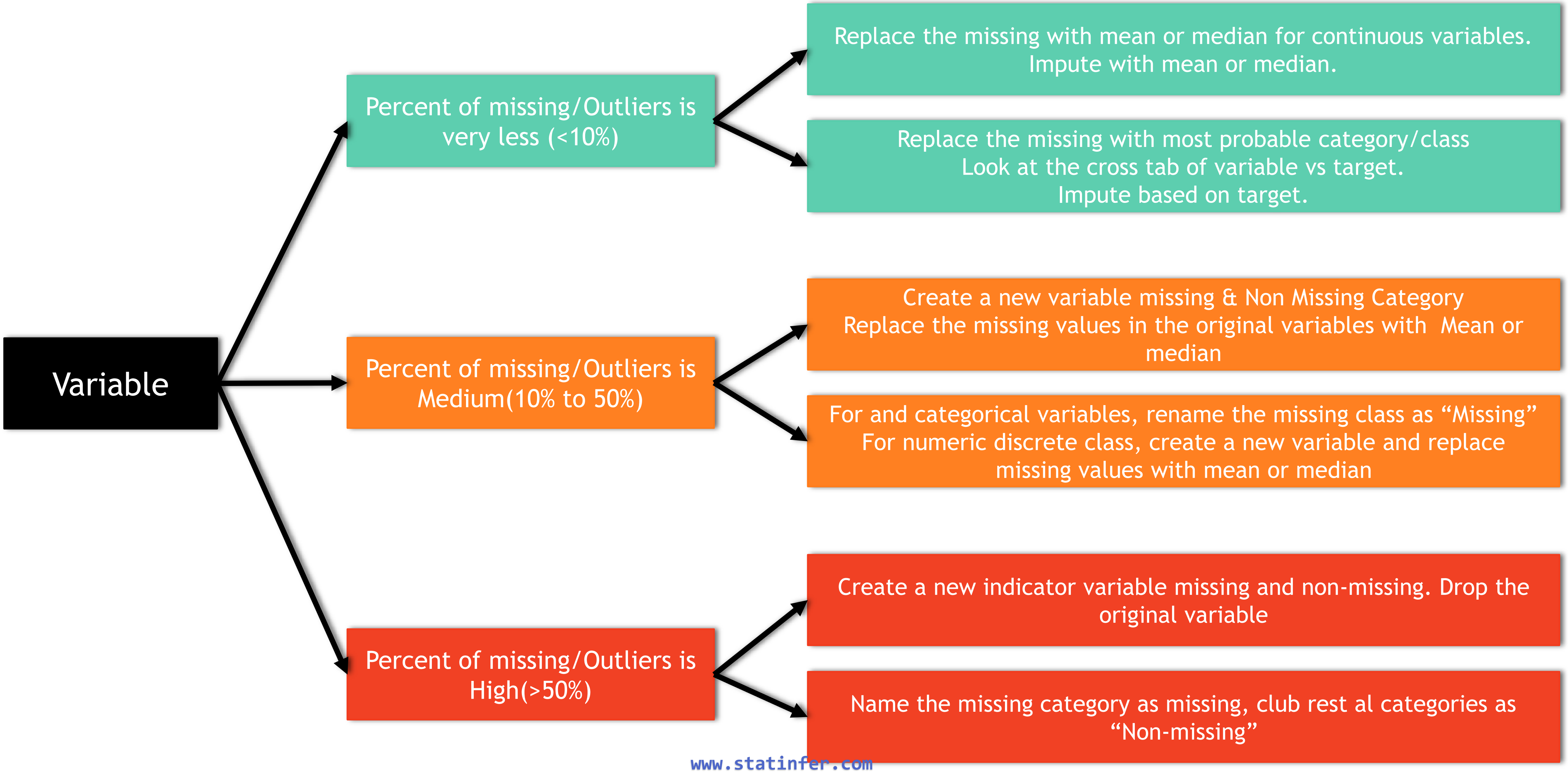
Fig25: Box Plot(after clipping)



Data Cleaning Scenario-2

NumberOfTime30_59DaysPastDueNotW

- Find bad rate in each category of this variable
- Replace 96 with ______? Replace 98 with______?


Variable
Percent of missing/Outliers is very less (<10%)
Replace the missing with mean or median for continuous variables. Impute with mean or median.
Replace the missing with most probable category/class
Look at the cross tab of variable vs target.
Impute based on target.
Percent of missing/Outliers is Medium(10% to 50%)
Create a new variable missing & Non Missing Category
Replace the missing values in the original variables with Mean or median
For and categorical variables, rename the missing class as "Missing"
For numeric discrete class, create a new variable and replace missing values with mean or median
Percent of missing/Outliers is High(>50%)
Create a new indicator variable missing and non-missing. Drop the original variable
Name the missing category as missing, club rest al categories as "Non-missing"

Variable

Percent of missing/Outliers is very less (<10%)

Replace the missing with most probable category/class
Look at the cross tab of variable vs target.
Impute based on target.

LAB: Data Cleaning Scenario-2

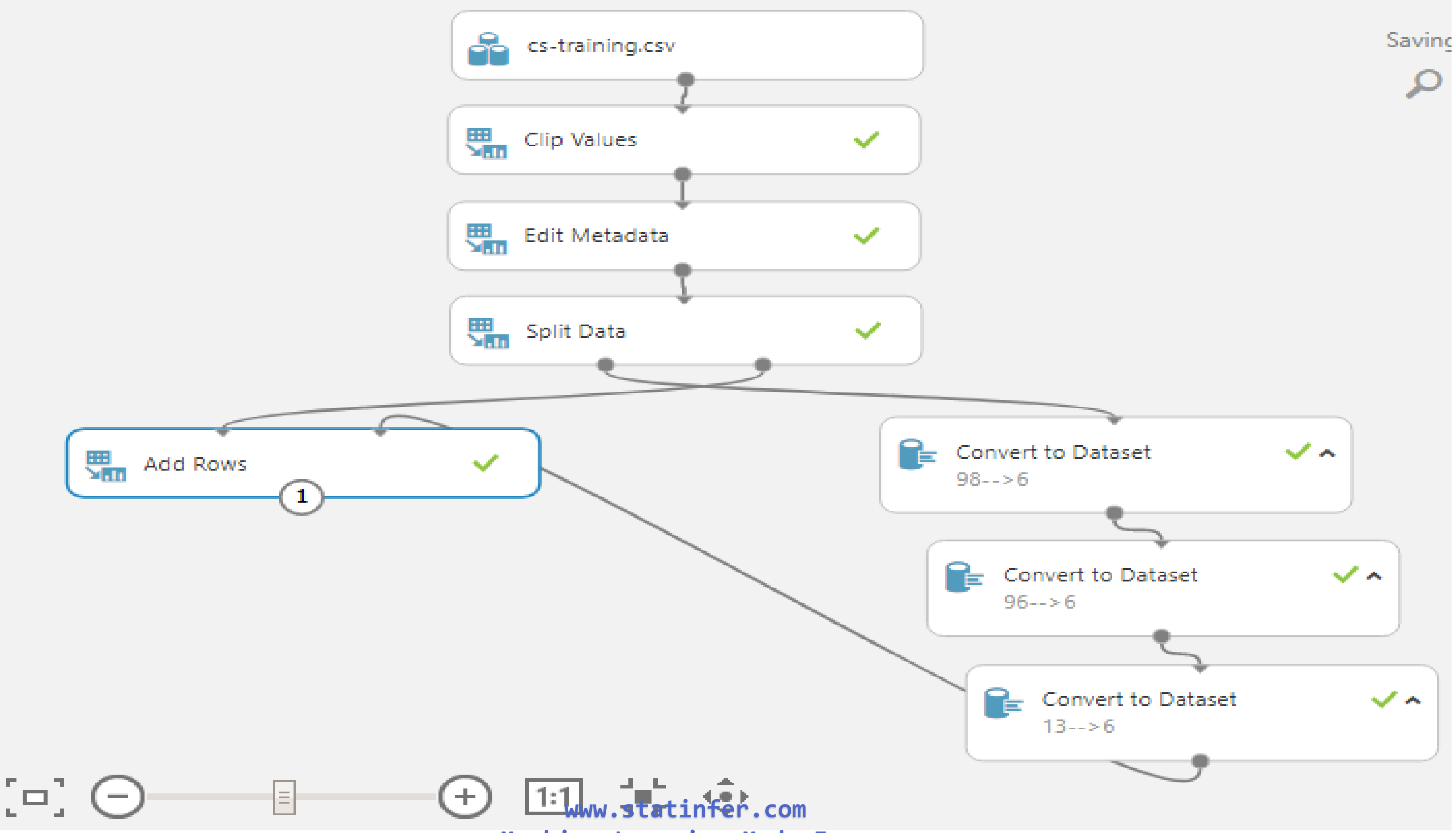
- What is the issue with NumberOfTime30_59DaysPastDueNotW
- Draw a frequency table
- What percent of the values are erroneous?
- Clean the variable- Look at the cross tab of variable vs target. Impute based on target .
- Create frequency table for cleaned variable

Steps - Data Cleaning Scenario-2

- For this to be done first the variable should be changed to categorical type
- Drag and drop Edit Metadata, connect it to the Previous Clip Value
- Select the column(NumberOfTime30-59DaysPastDueNotWorse) and in category select MakeCategorical
- Drag and drop Split Data, select Relative Expression and give the expression as \"NumberOfTime30-59DaysPastDueNotWorse" > 12
- Click on run
- Once finished running, drag and drop Convert to Dataset into the canvas
- In properties, select Action→ReplaceValues, Replace→Custom, Custom value→98, New value→6
- Repeat previous two steps for the values 93 and 13
- Drag and drop Join Data, connect the second output of Split Data to the first input of Add Rows and the output of last Convert to Dataset to the second input of Add Rows
- Click on run
- Once finished running, the output of Add Rows contains no outlier values in NumberOfTime30-59DaysPastDueNotWorse column

Steps - Data Cleaning Scenario-2

Fig26: Splitting-Treating- Joining



Steps - Data Cleaning Scenario-2

Fig27: Properties of Convert to Dataset

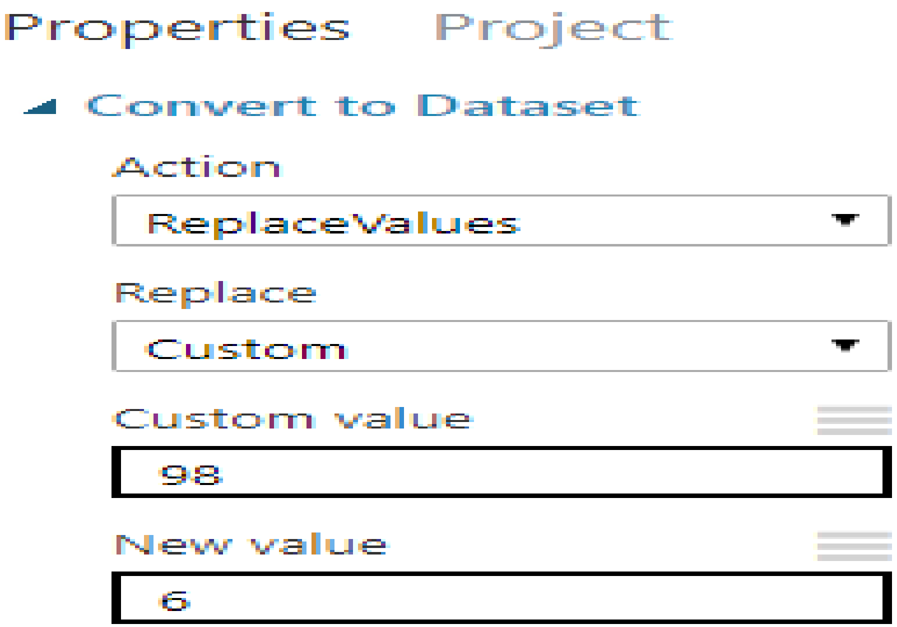
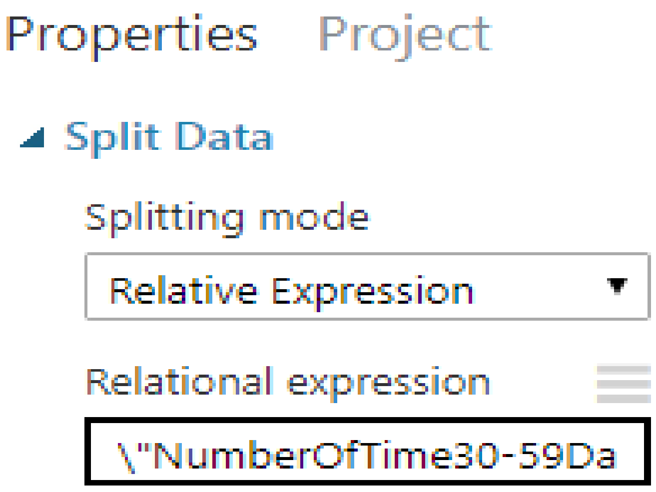


Fig28: Properties of Split Data



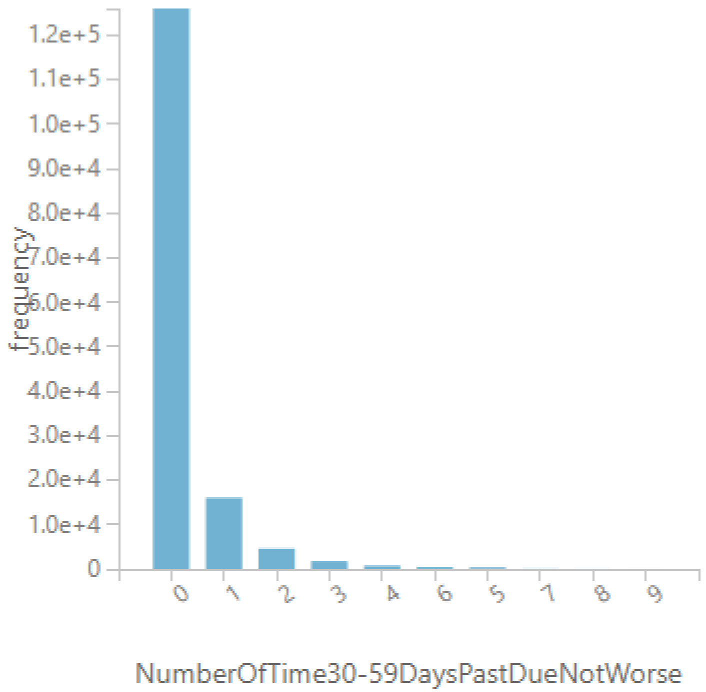
Steps - Data Cleaning Scenario-2

Fig29: Visualize and Histogram

Data Cleaning › Add Rows › Results dataset

rows	columns
150000	14

RevolvingUtilizationOfUnsecuredLines_clipped_value	RevolvingUtilizationOfUnsecuredLines_clipped	age	NumberOfTime30-59DaysPastDueNotWorse
0.766127	false	45	2
0.957151	false	40	0
0.65818	false	38	1
0.23381	false	30	0
0.907239	false	49	1
0.213179	false	74	0
0.305682	false	57	0
0.754464	false	39	0
0.116951	false	27	0



Data Cleaning Scenario-3

Monthly Income

- Monthly Income has nearly 20% missing values
- Missing value percentage is significant
- Simply replacing with mean or median is not sufficient
- We can create an indicator variable to keep track of missing and non-missing values

- Variable
  - Percent of missing/Outliers is very less (<10%)
    - Replace the missing with mean or median for continuous variables. Impute with mean or median.
    - Replace the missing with most probable category/class
      Look at the cross tab of variable vs target.
      Impute based on target.
  - Percent of missing/Outliers is Medium(10% to 50%)
    - Create a new variable missing & Non Missing Category
      Replace the missing values in the original variables with Mean or median
    - For and categorical variables, rename the missing class as “Missing”
      For numeric discrete class, create a new variable and replace missing values with mean or median
  - Percent of missing/Outliers is High(>50%)
    - Create a new indicator variable missing and non-missing. Drop the original variable
    - Name the missing category as missing, club rest al categories as “Non-missing”

Variable

Percent of missing/Outliers is Medium(10% to 50%)

Create a new variable missing & Non Missing Category
Replace the missing values in the original variables with Mean or median

LAB: Monthly Income

- Find the missing value percentage in monthly income
- Create an indicator variable for missing and non-missing
- Replace the missing values with median

Steps - Monthly Income

- Drag and drop Convert to Dataset, connect it to the Previous Add Rows
- In properties, select Action→SetMissingValues, Custom missing value→NA
- Click on run
- Drag and drop Clean Missing Data, connect it to Convert to Dataset
- In Properties, select the column to be treated
- Minimum missing value ratio → 0, Maximum missing value ratio →1
  Cleaning mode → Replace With Median,
  Cols with all missing values →Remove, check Generate missing value indicator column
- Click on run
- Once finished running, visualize the output of Clean Missing Data

Steps - Monthly Income

Fig30: Clean Missing Data

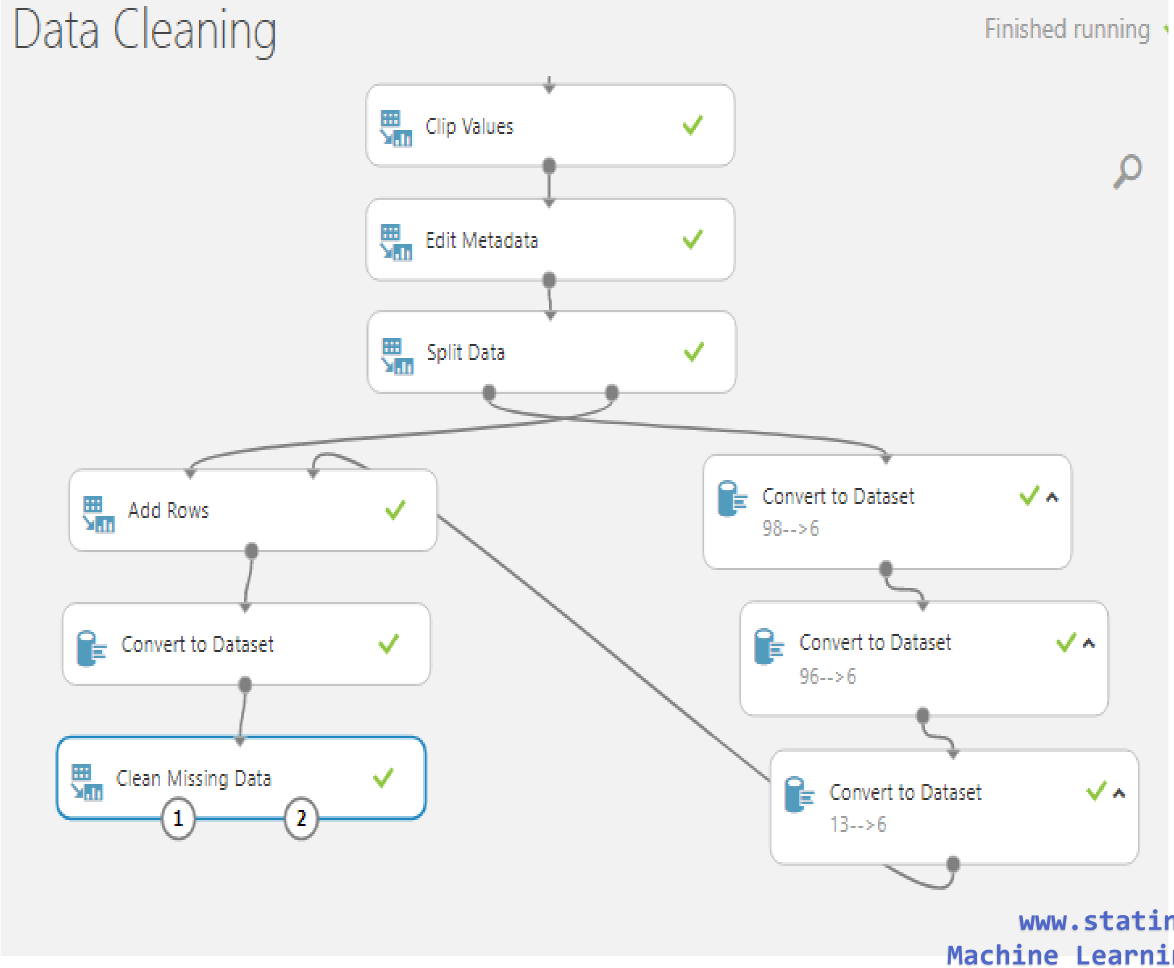
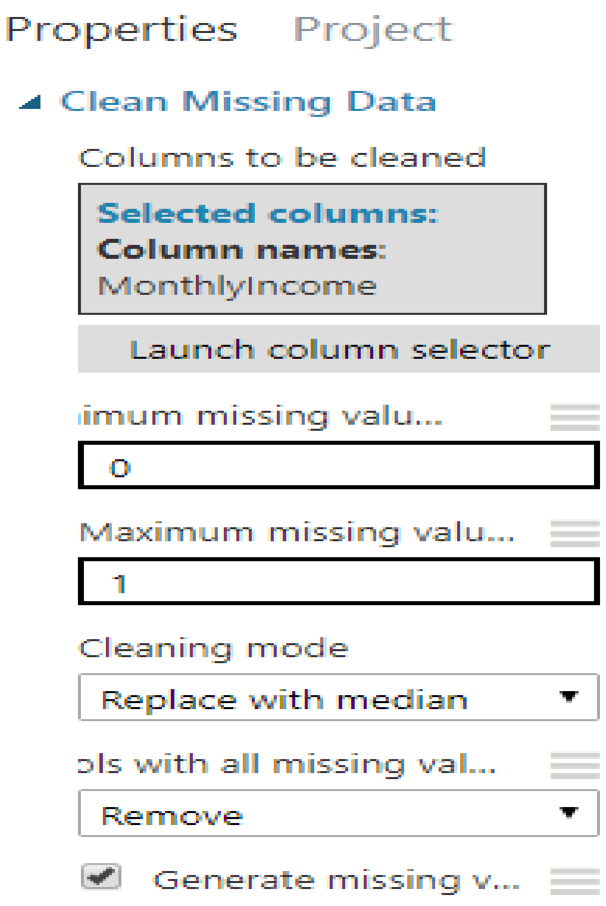


Fig31: Properties of Clean Missing Data



Steps - Monthly Income

Fig32: Monthly Income(with Missing values)

Data Cleaning › Convert to Dataset › Results dataset

rows	columns
150000	14

age	NumberOfTime30-59DaysPastDueNotWorse	DebtRatio	MonthlyIncome	NumberOfOpenCreditLinesAndLoans
45	2	0.802982	9120	13
40	0	0.121876	2600	4
38	1	0.085113	3042	2
30	0	0.03605	3300	5
49	1	0.024926	63588	7
74	0	0.375607	3500	3
57	0	5710		8
39	0	0.20994	3500	8

Statistics

Mean	6670.2212
Median	5400
Min	0
Max	3008750
Standard Deviation	14384.6742
Unique Values	13594
Missing Values	29731
Feature Type	Numeric Feature

Visualizations

MonthlyIncome
Histogram

Steps - Monthly Income

Fig33: Monthly Income(without Missing values)

Data Cleaning › Clean Missing Data › Cleaned dataset

rows	columns
150000	15

age	NumberOfTime30-59DaysPastDueNotWorse	DebtRatio	MonthlyIncome	NumberOfOpenCreditLinesAndLoans
45	2	0.802982	9120	13
40	0	0.121876	2600	4
38	1	0.085113	3042	2
30	0	0.03605	3300	5
49	1	0.024926	63588	7
74	0	0.375607	3500	3
57	0	5710	5400	8
39	0	0.20994	3500	8

Statistics

Mean	6418.4549
Median	5400
Min	0
Max	3008750
Standard Deviation	12890.3955
Unique Values	13594
Missing Values	0
Feature Type	Numeric Feature

Visualizations

MonthlyIncome
Histogram

Data Cleaning Other Variables

Remaining Variables Imputation

- Debt Ratio: Imputation
- NumberOfOpenCreditLinesAndLoans : No issues in this variable
- NumberOfTimes90DaysLate: Imputation similar to NumberOfTime30_59DaysPastDueNotW
- NumberRealEstateLoansOrLines: : No issues in this variable
- NumberOfTime60_89DaysPastDueNotW: Imputation similar to NumberOfTime30_59DaysPastDueNotW
- NumberOfDependents: Impute based on target variable

Conclusion

statinfer

Conclusion

- Data cleaning is as important as data analysis
- Sometimes 80% of the overall project time is spent on data cleaning
- Data cleaning needs patience, we need to clean for each individual variable
- Apart from suggested methods, there are many heuristic ways of cleaning the data

statinfer

Part 5/12 - Regression Analysis with Azure

Venkat Reddy

Contents

- Correlation
- Regression
- Simple Regression
- R-Squared
- Multiple Regression
- Adj R-Squared
- P-value
- Multicollinearity
- Interaction terms

Correlation

What is need of correlation?

- Is there any association between hours of study and grades?
- Is there any association between number of temples in a city & murder rate?
- What happens to sweater sales with increase in temperature? What is the strength of association between them?
- What happens to ice-cream sales v.s temperature? What is the strength of association between them?
- How to quantify the association?
- Which of the above examples has very strong association?

- **Correlation**

Correlation coefficient

- It is a measure of linear association
- r is the ratio of variance together vs product of individual variances.

$$\text{Correlation coefficient } r = \frac{\text{Covariance of XY}}{\text{Sqrt(VarianceX* VarianceY)}}$$

Correlation coefficient

- Correlation varies between -1 to +1
- Correlation 0 No linear association
- Correlation 0 to 0.25 Negligible positive association
- Correlation 0.25-0.5 Weak positive association
- Correlation 0.5-0.75 Moderate positive association
- Correlation >0.75 Very Strong positive association

LAB –Correlation Calculation

- Dataset: AirPassengers\\AirPassengers.csv
- Draw scatter plot between promotional budget and number of passengers
- Find the correlation between number of passengers and promotional budget.
- Find the correlation between number of passengers and Service_Quality_Score
- Find the correlation between number of passengers and Holiday_week

Steps - Correlation Calculation

- Drag-and-drop the dataset(AirPassengers.csv) into the canvas
- In the left pane on the experiment window search for ‘Select columns from the Dataset’
- Drag-and-drop ‘Select columns from the Dataset’ into the canvas
- Connect the output of the dataset to the input of the ‘Select columns from the Dataset’
- Click on ‘Select columns from the Dataset’ and in the properties window click the ‘launch column selector’
- ‘Select columns’ window will open, select With Rules in left pane and select Begin with No Columns in right pane
- Include → Column names → Variables for which correlation is done(Passenger & Promotion_Budget) and click on ✓

Steps - Correlation Calculation cont..

- Search for ‘Compute Linear Correlation’ in left pane of the experiment window drag-and-drop it into the canvas
- Connect the output of the ‘Select columns from the Dataset’ to the input of the ‘Compute Linear Correlation’
- Click on 

and wait, after execution we can see “Finished Running’ at the top of the canvas and in properties window Status Code will be Finished
- Once this is done click on the output circle and select visualize
- Where you can see the correlation between two variables

Steps - Correlation Calculation cont..

Fig1: Add Dataset

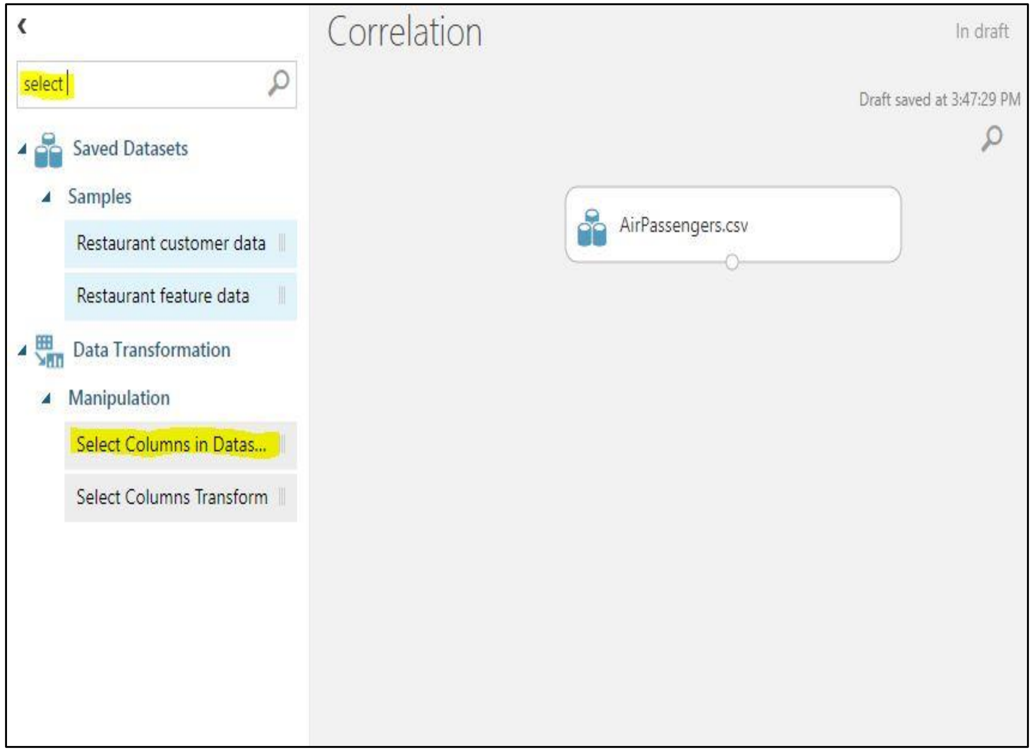
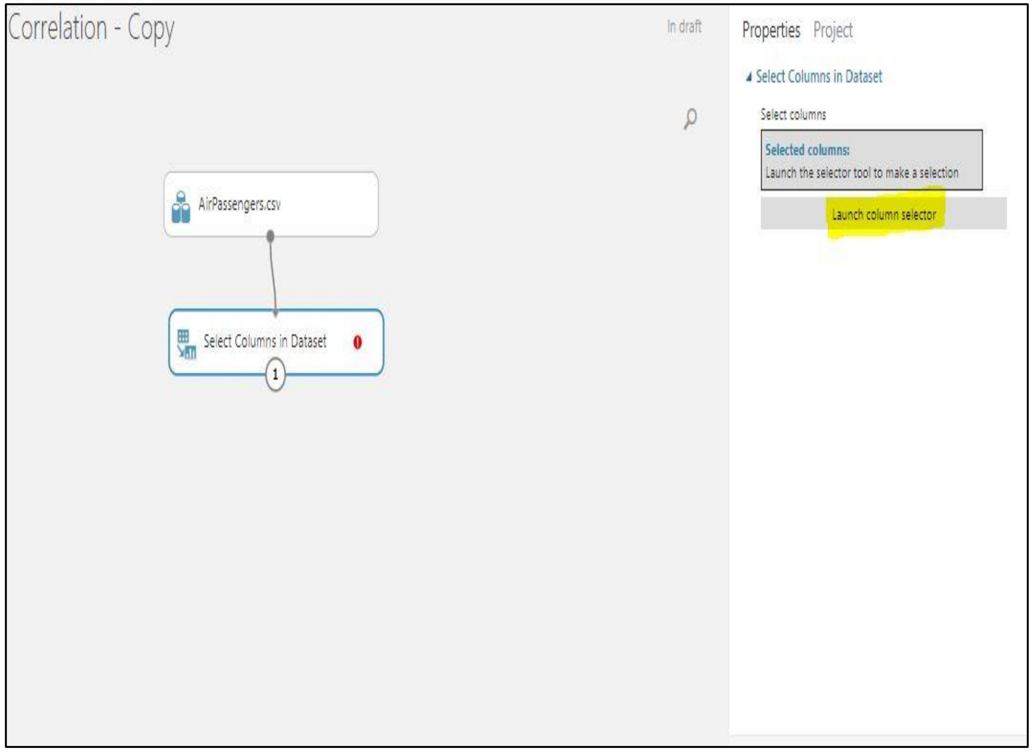


Fig2: Add Select Column



Steps - Correlation Calculation cont..

Fig3: Selecting Columns

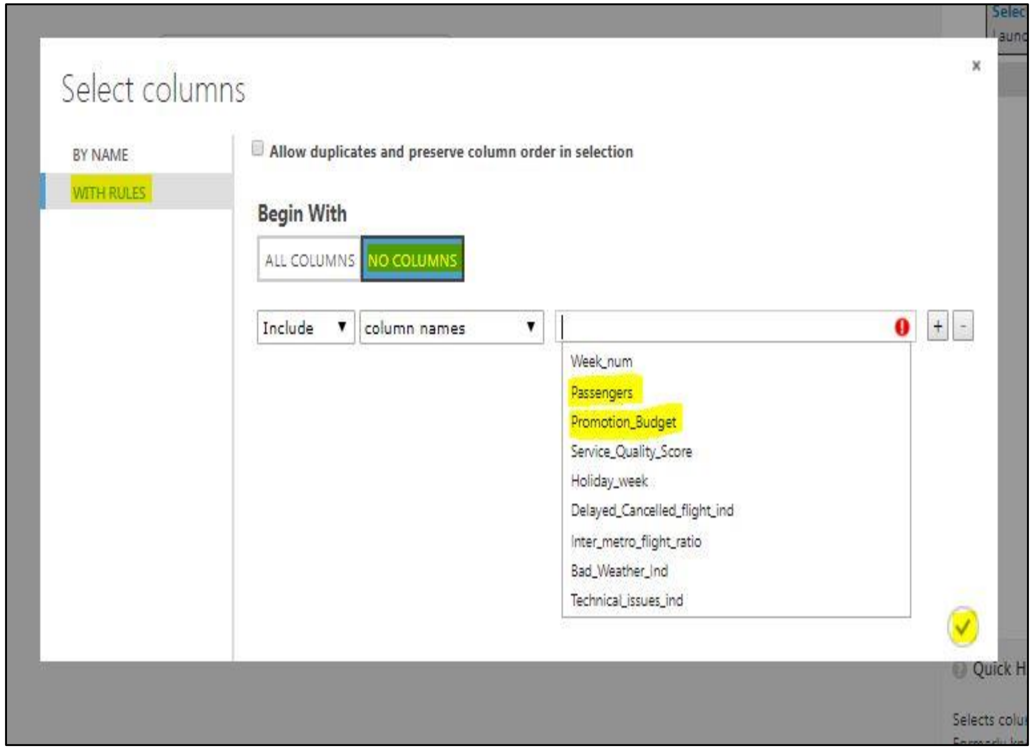
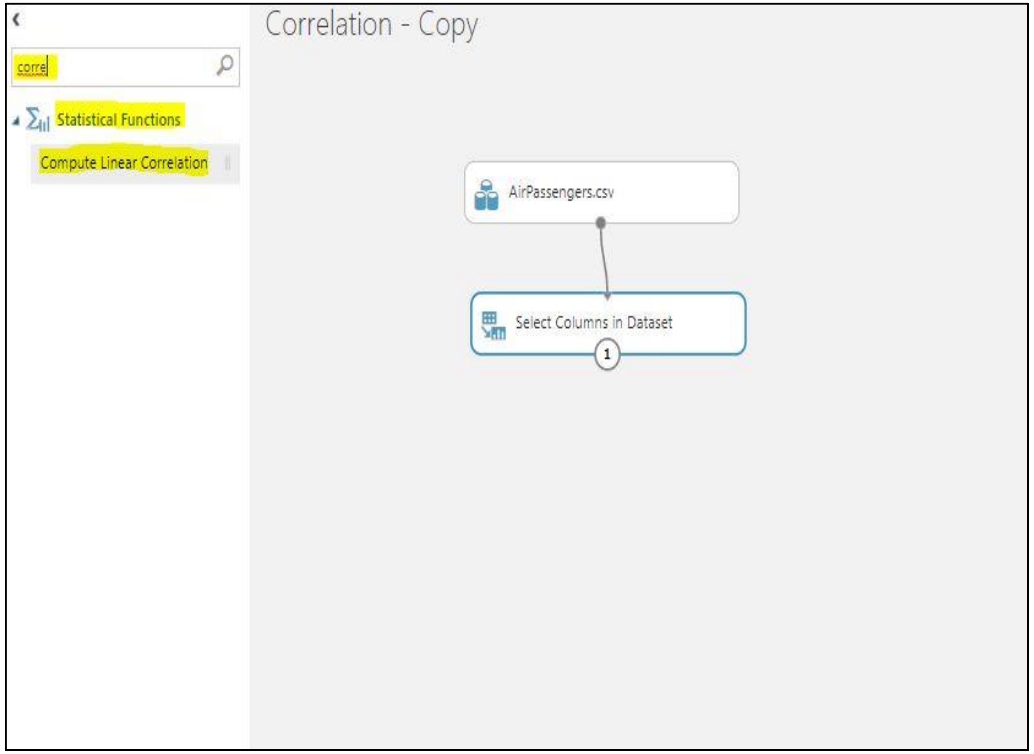


Fig4: Add Compute Linear Correlation



Steps - Correlation Calculation cont..

Fig5: Run

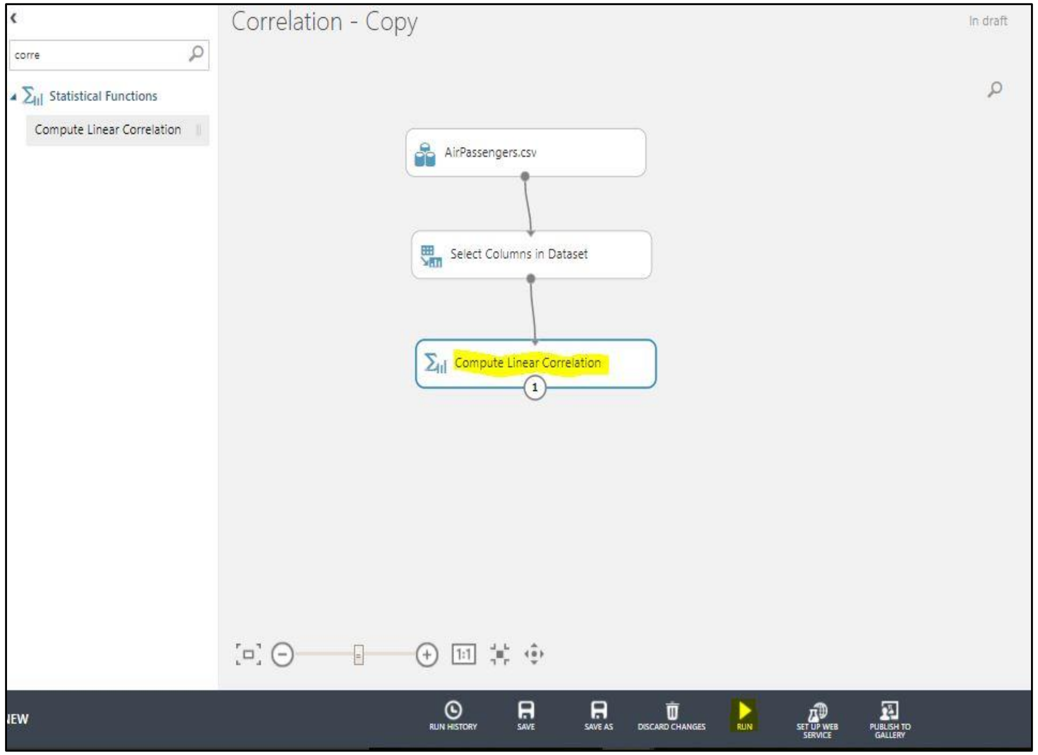
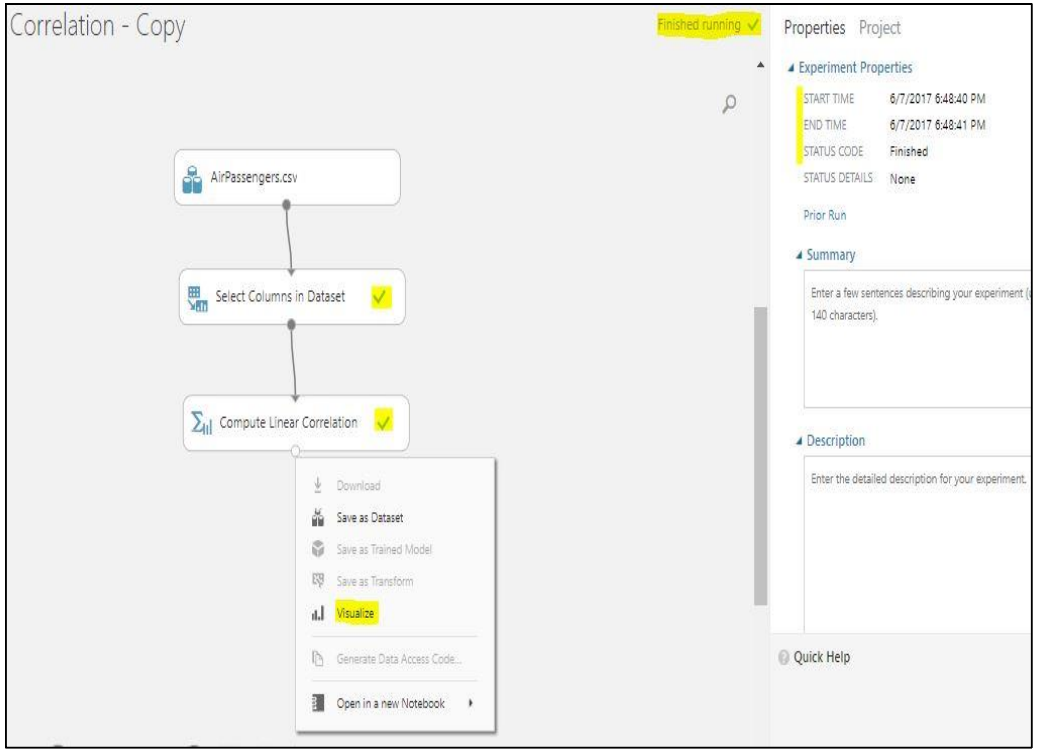
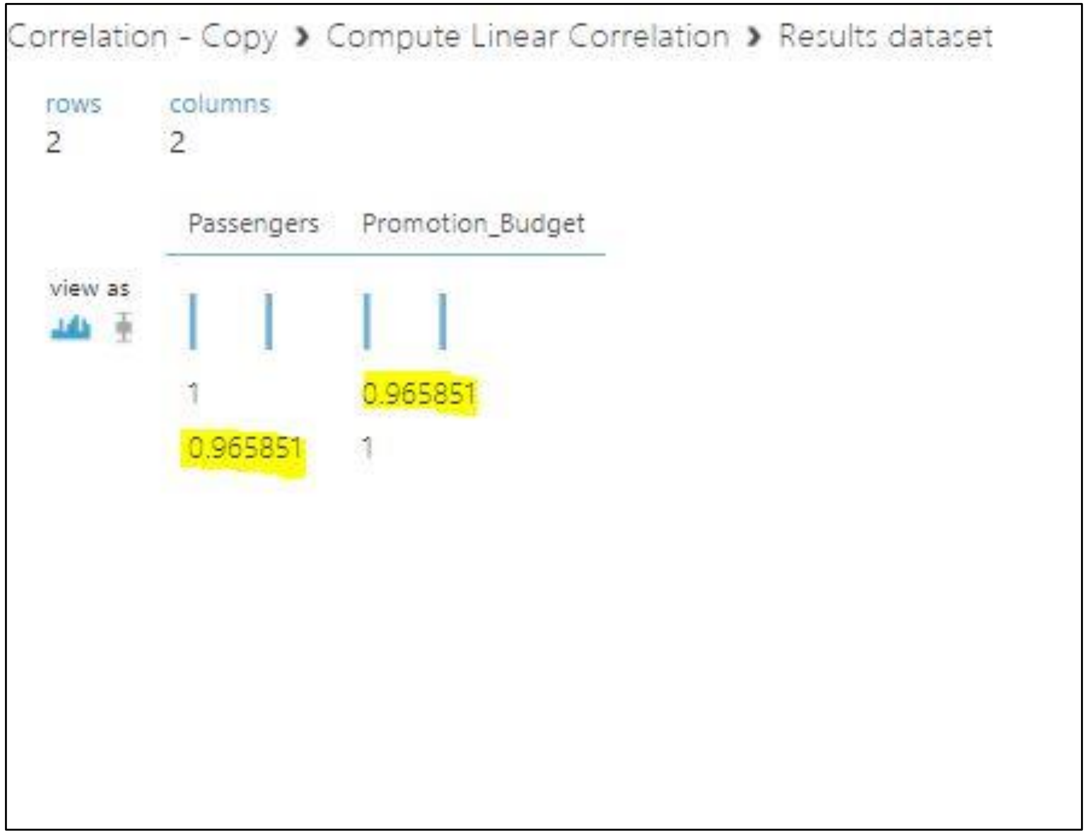


Fig6: Visualize



Steps - Correlation Calculation cont..

Fig7: Correlation between Passenger and Promotional_Budget



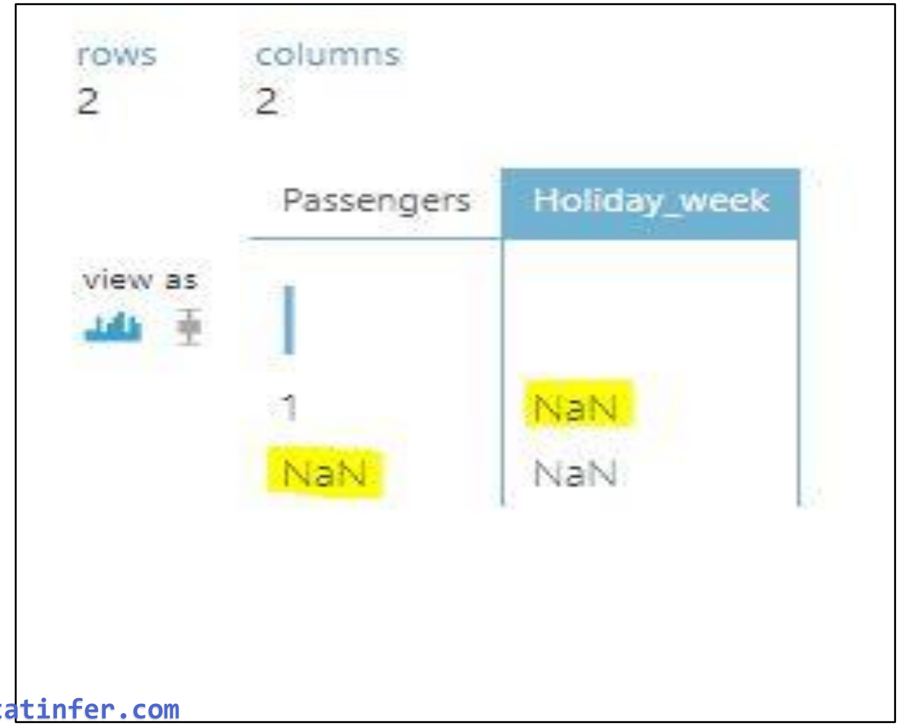
Passengers	Promotion_Budget
1	0.965851
0.965851	1

Steps - Correlation Calculation cont..

- Similarly for Service_Quality_Score and Holiday_week we can change columns and find the correlation
- Holiday_week is an Categorical variable and it cannot be compared with Passengers which is numeric so we get NaN(fig9)

Fig8: Passenger vs Service_Quality_Score

Fig9: Passenger vs Holiday_week(indicator_variable)



Beyond Pearson Correlation

Beyond Pearson Correlation

- How to find correlation between an indicator variable and continuous variable
- How to quantify the association between two indicator variables?
- How to quantify the association between two categorical variables?

Beyond Pearson Correlation

- Correlation coefficient measures for different types of data

Variable Y\X	**Quantitative /Continuous X**	**Ordinal/Ranked/Discrete X**	**Nominal/Categorical X**
Quantitative Y	Pearson r	Biserial r_b	Point Biserial r_{pb}
Ordinal/Ranked/Discrete Y	Biserial r_b	Spearman rho/Kendall's	Rank Biserial r_{rb}
Nominal/Categorical Y	Point Biserial r_{pb}	Rank Biserial r_{rb}	Phi, Contingency Coeff, V

statinfer

From Correlation to Regression

From Correlation to Regression

- In the above example promotion budget and number of passengers are highly correlated.
- Can we estimate number of passengers given the promotion budget?

From Correlation to Regression

- Correlation is just a measure of association
- It can’t be used for prediction.
- Given the predictor variable, we can’t estimate the dependent variable.
- In the air passengers example, given the promotion budget, we can’t get an estimated value of passengers
- We need a model, an equation, a fit for the data.
- That is known as regression line

statinfer

What is Regression

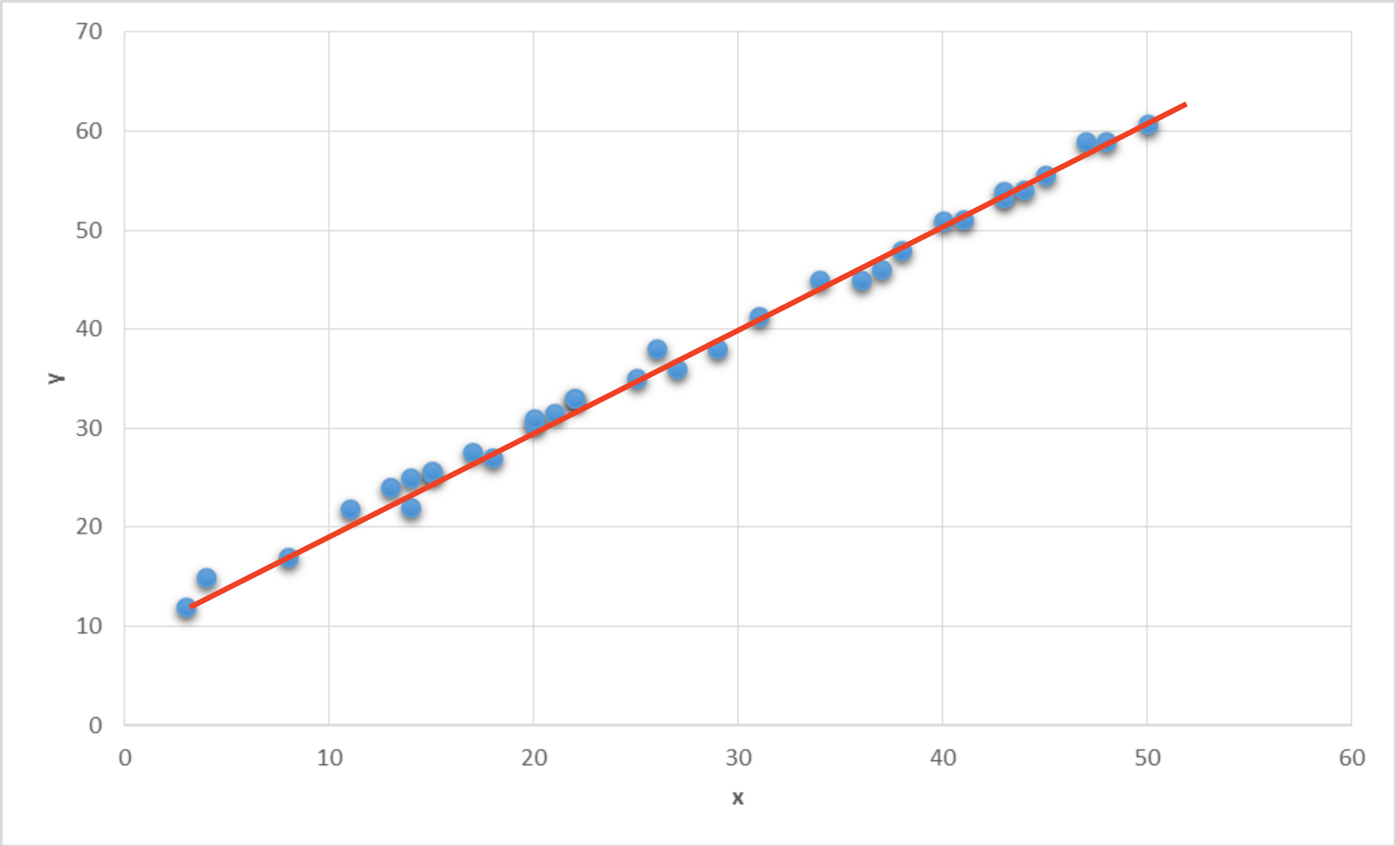
What is Regression

- A regression line is a mathematical formula that quantifies the general relation between a predictor/independent (or known variable x) and the target/dependent (or the unknown variable y)
- Below is the regression line. If we have the data of x and y then we can build a model to generalize their relation

$$\boxed{y = \beta_0 + \beta_1 x}$$

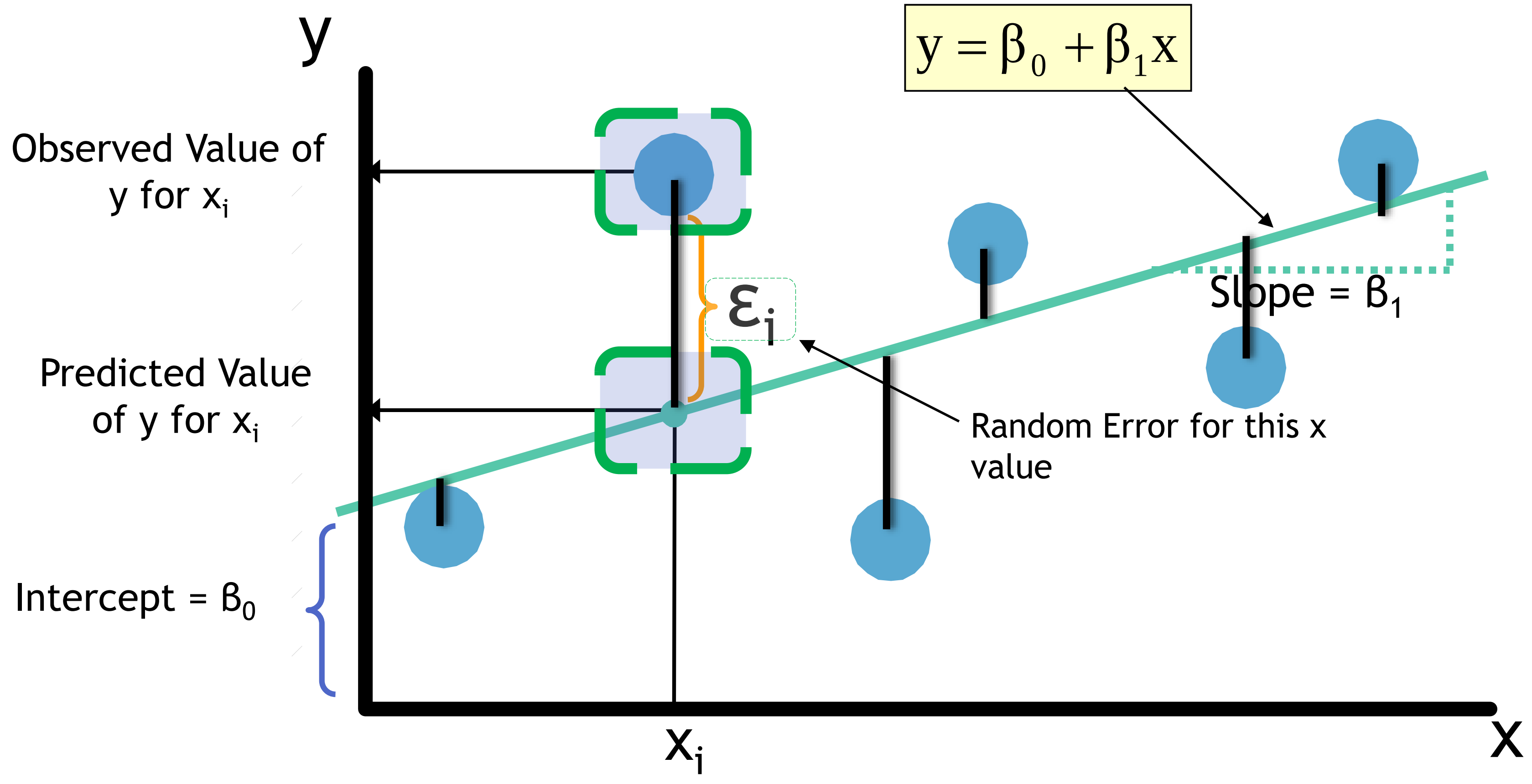
- What is the best fit for our data?
- The one which goes through the core of the data
- The one which minimizes the error

Regression

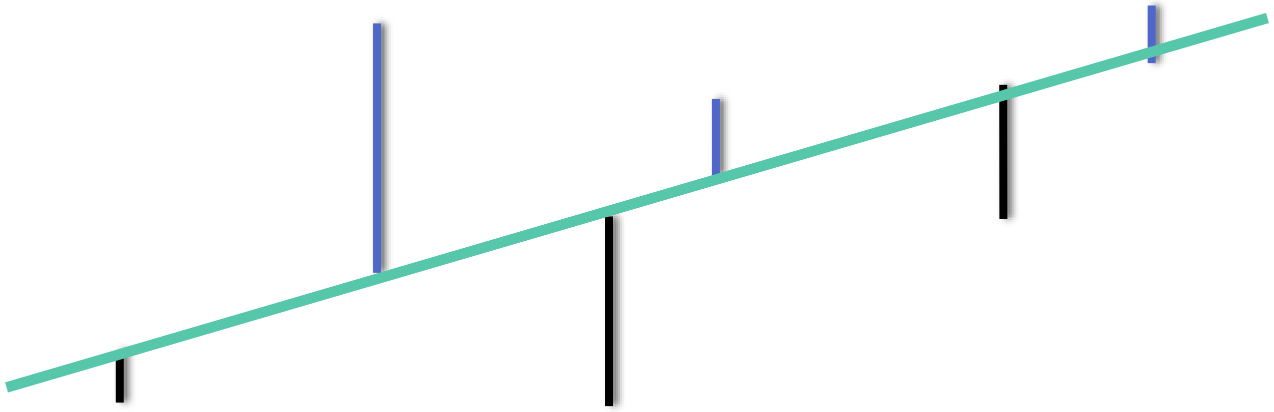


Regression Line fitting-Least Squares Estimation

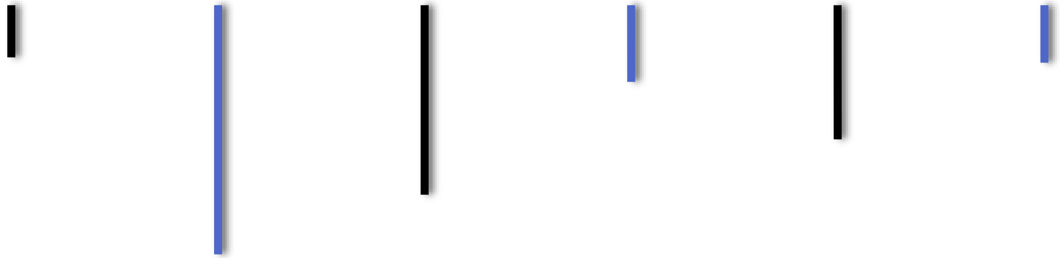
Regression Line fitting



Regression Line fitting



Minimizing the error



- The best line will have the minimum error
- Some errors are positive and some errors are negative. Taking their sum is not a good idea
- We can either minimize the squared sum of errors Or we can minimize the absolute sum of errors
- Squared sum of errors is mathematically convenient to minimize
- The method of minimizing squared sum of errors is called least squared method of regression

Least Squares Estimation

- X: x1, x2, x3, x4, x5, x6, x7,……..
- Y:y1, y2, y3, y4, y5, y6, y7…….
- Imagine a line through all the points
- Deviation from each point (residual or error)
- Square of the deviation
- Minimizing sum of squares of deviation

$$\boxed{\begin{aligned}\sum e^2 &= \sum (y - \hat{y})^2 \\ &= \sum (y - (\beta_0 + \beta_1 x))^2\end{aligned}}$$

β_0 and β_1 are obtained by minimize the sum of the squared residuals

LAB: Regression Line Fitting

- Dataset: AirPassengers\\AirPassengers.csv
- Find the correlation between Promotion_Budget and Passengers
- Draw a scatter plot between   Promotion_Budget and Passengers. Is there any pattern between Promotion_Budget and Passengers?
- If the Promotion_Budget is 650,000 how many passenger's can be expected in that week?
- Build a linear regression model and estimate the expected passengers for a Promotion_Budget is 650,000

Steps - Regression Line Fitting

- Since we have found the correlation between Passengers and Promotional_Budget (slide-15), we shall start with scatter plot
- Scatter plot between Passengers and Promotional_Budget:
  - Drag-and-drop the dataset into the canvas
  - Click on the output circle and select visualize
  - In the table click the Passenger heading, in the right pane Visualizations can be seen
  - In Compare to dropdown list select Promotional_Budget
  - The scatter plot between Passengers and Promotional_Budget appears below

Steps - Regression Line Fitting cont..

Fig10: Adding Dataset

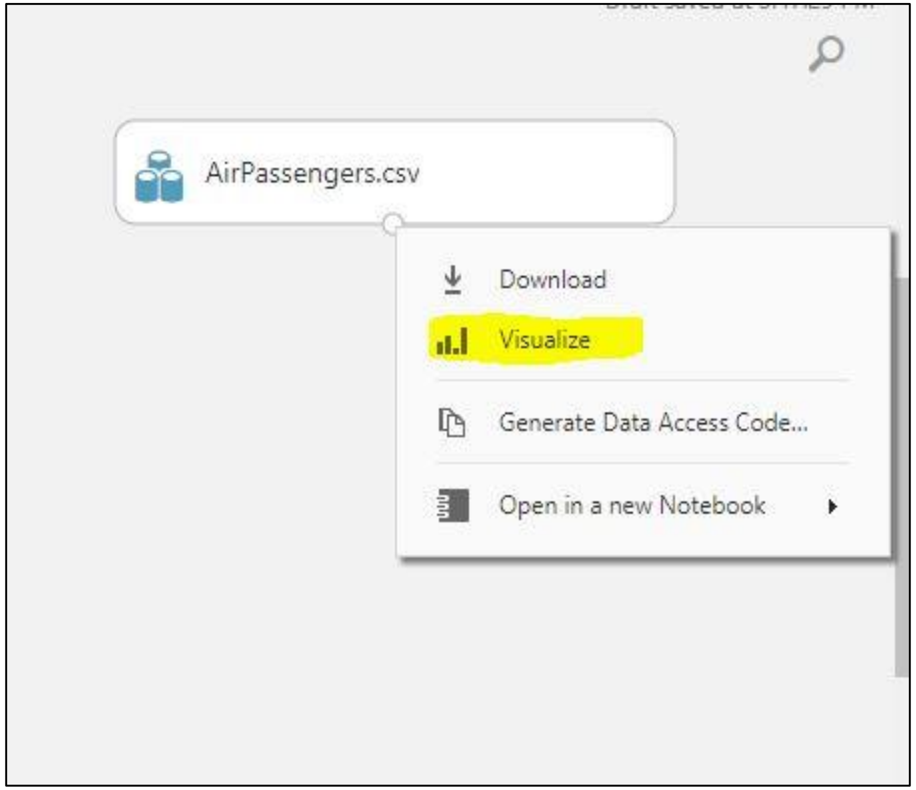
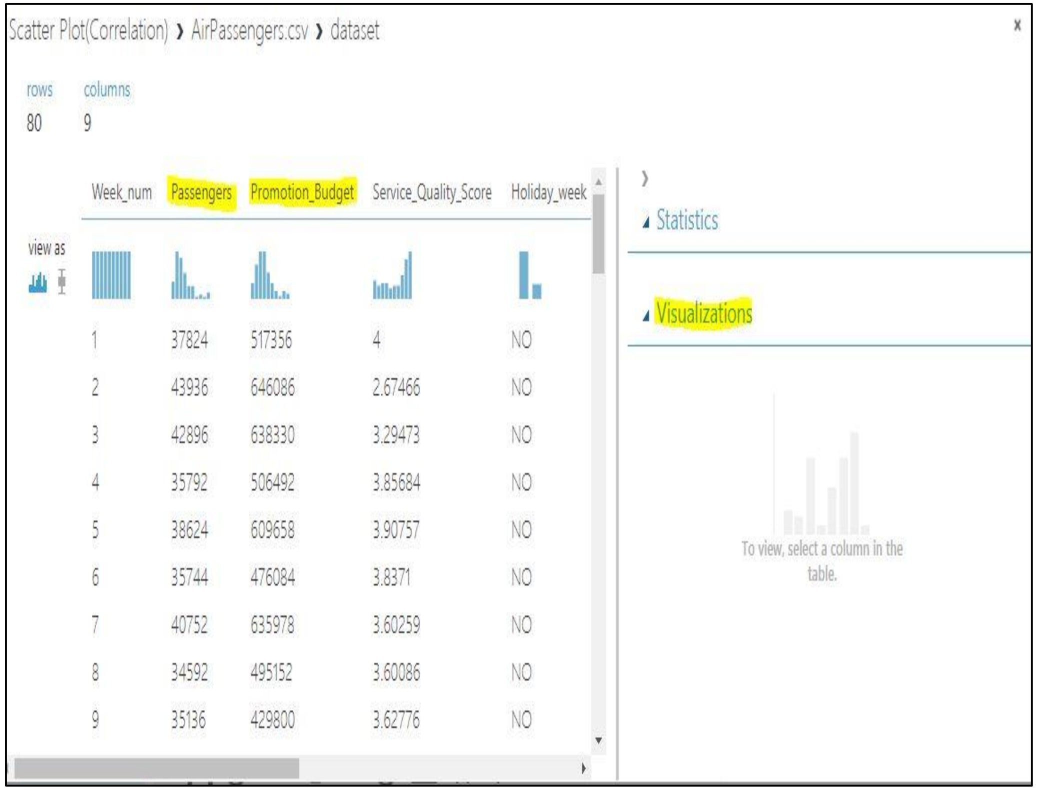
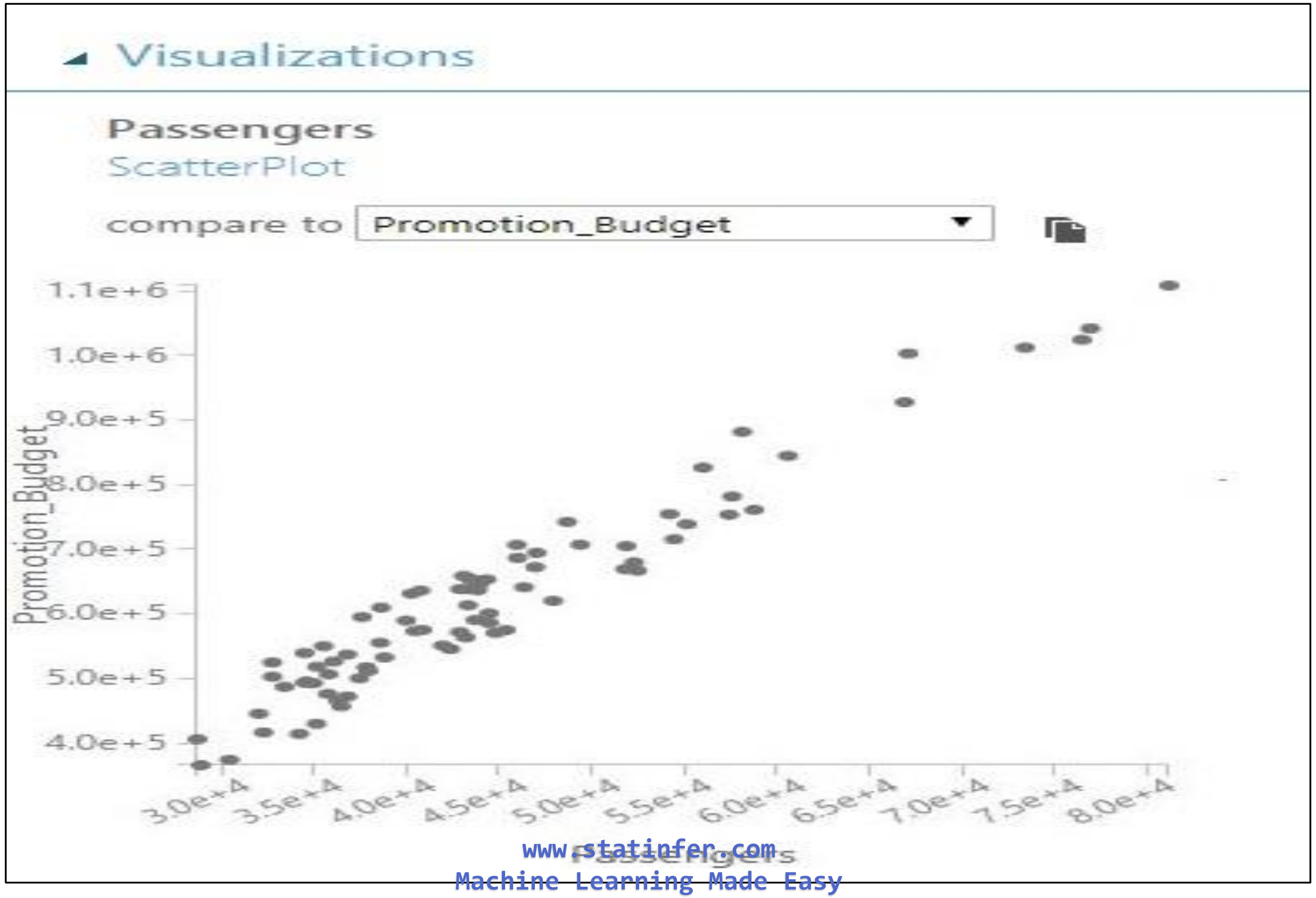


Fig11: Visualize



Steps - Regression Line Fitting cont..

Fig12: Scatter Plot between Passengers and Promotional_Budget



Steps - Regression Line Fitting cont..

- Linear Regression for Predicting the No. of Passengers
  - Drag-and-drop AirPassengers.csv dataset to the canvas
  - Drag-and-drop 'select column from dataset' and select the columns
  - Search for 'Linear Regression', drag-and-drop it into the canvas
  - Click on 'Linear Regression' make sure that in properties window 'Ordinary Least Squares' is selected for solution method
  - Search for 'Train Model', drag-and-drop it into the canvas
  - Connect the output of 'Linear Regression' to left input of the 'Train Model' 'select column from dataset' to right input of the 'Train Model'
  - Click on 'Train Model', select launch column selector in the properties window
  - Select the column(Passengers) for which the prediction to be done
  - Drag-and-drop 'Score Model' from left pane and uncheck the 'Append score column' in properties window

Steps - Regression Line Fitting cont..

- Connect the output of ‘Train Model’ to left input of the ‘Score Model’ ‘select column from dataset’ to right input of the ‘Score Model’
- Drag-and-drop ‘Evaluate Model’ from left pane
- Connect the output of ‘Score Model’ to the input of ‘Evaluate Model’
- Click on Run 

- After execution click on the output circles of ‘Train Model’, ‘Score Model’ and ‘Evaluate Model’ to see the results

Steps - Regression Line Fitting cont..

Fig13: Adding Linear Regression

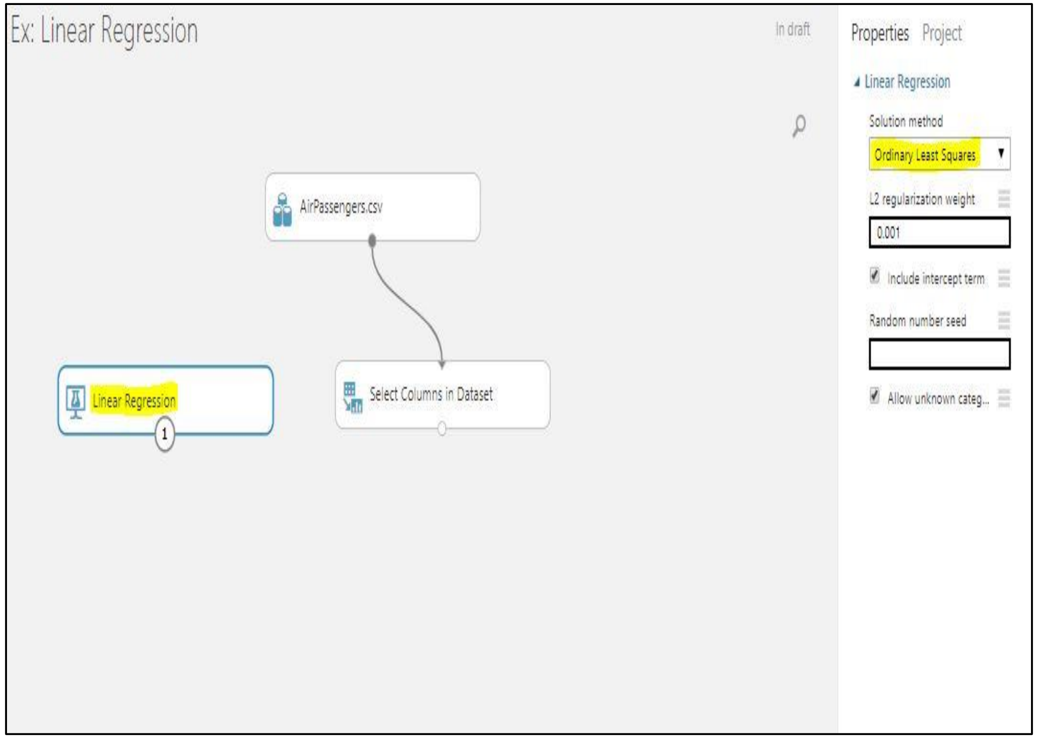
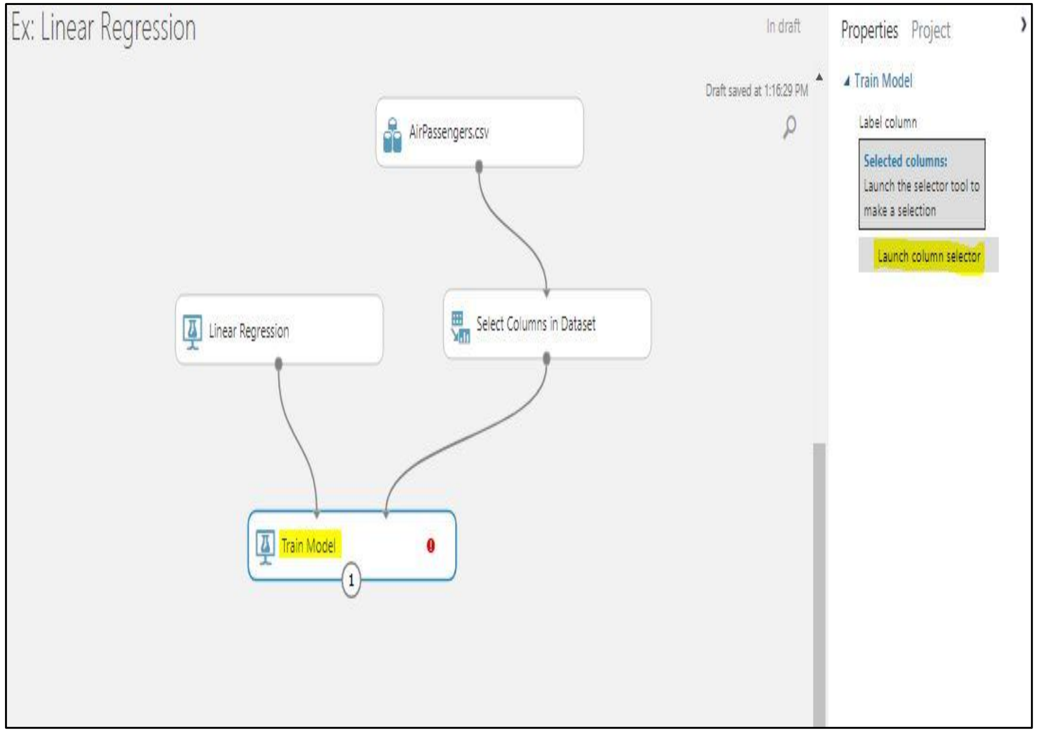


Fig14: Adding Train Model



Steps - Regression Line Fitting cont..

fig15: Variable to be Predicted

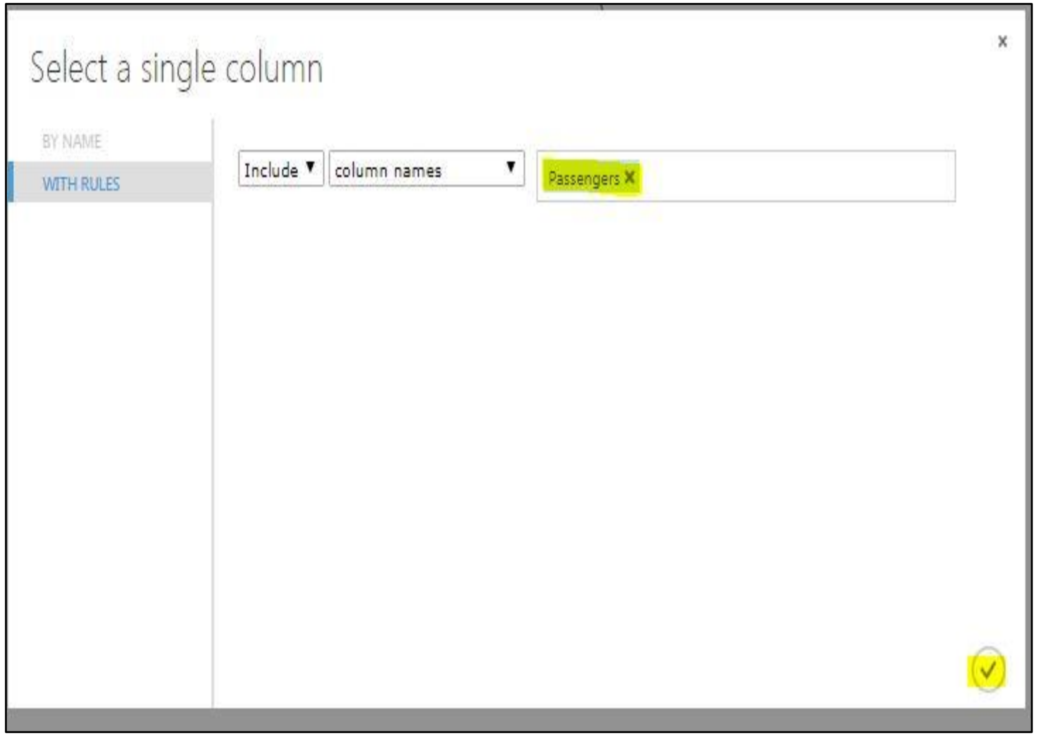
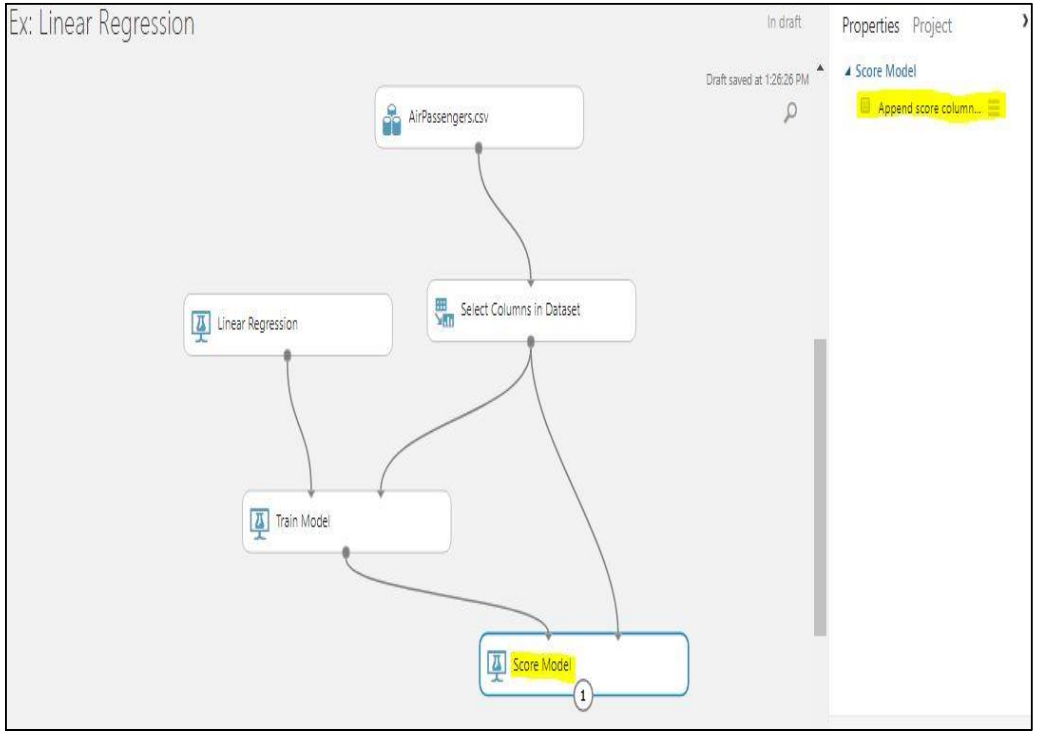


fig16: Adding Score Model



Steps - Regression Line Fitting cont..

fig17:Adding Evaluate Model

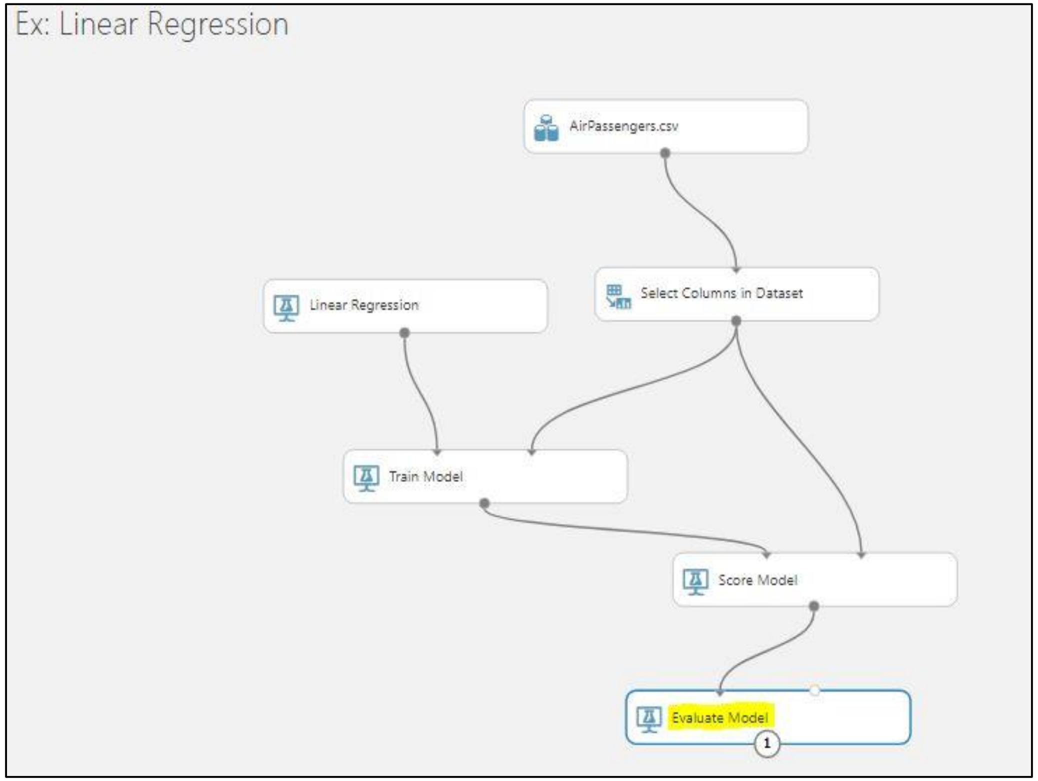
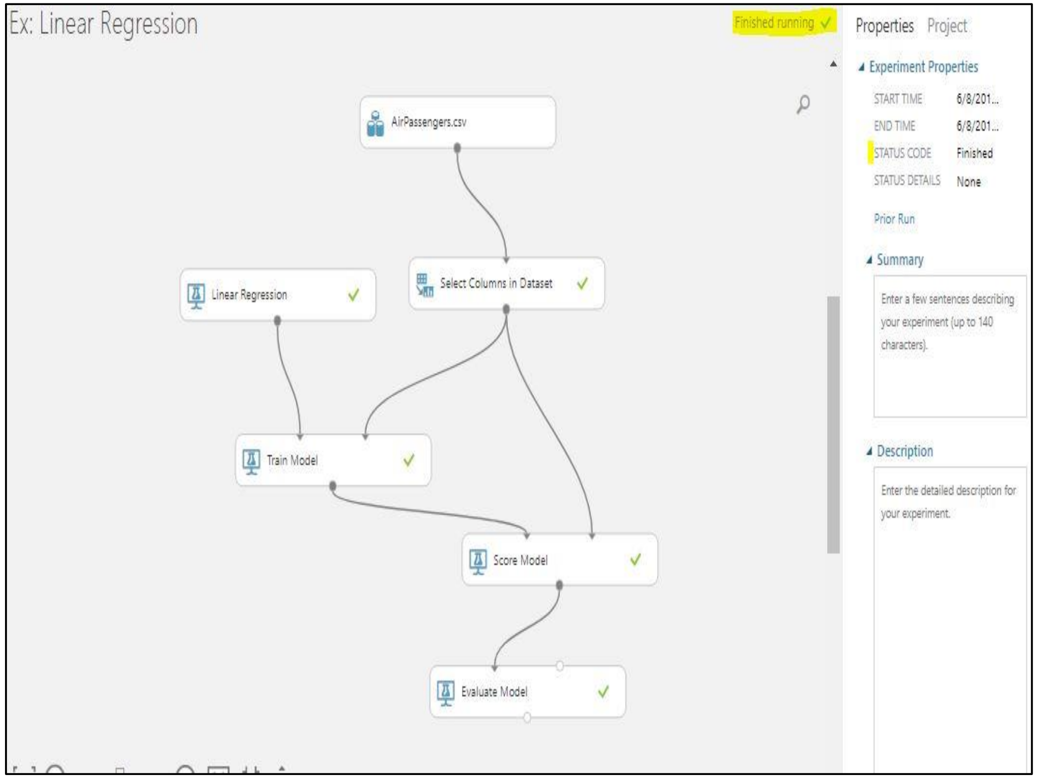
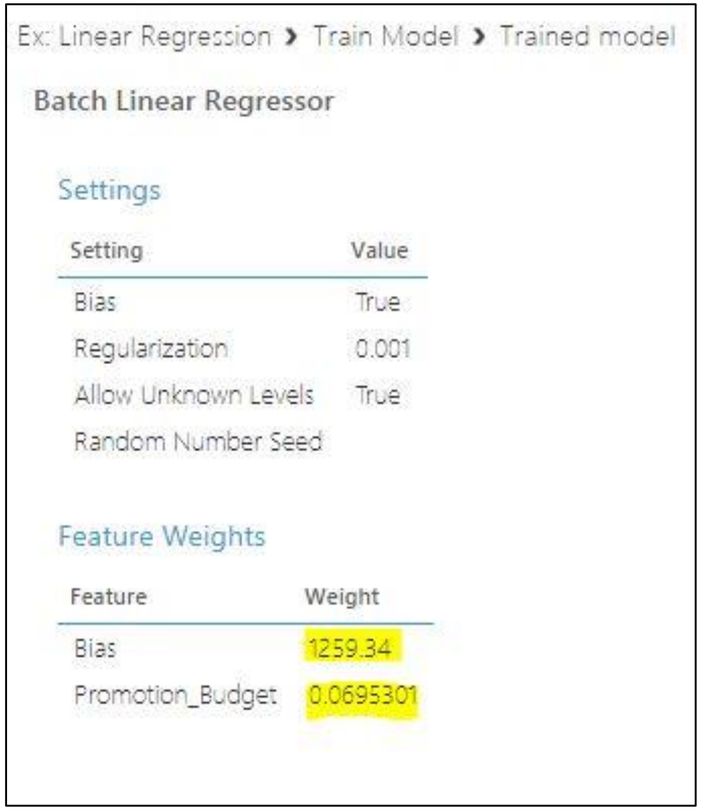


fig18: Finished Execution



Steps - Regression Line Fitting cont..

fig19: Train Model Output


Ex: Linear Regression › Train Model › Trained model

Batch Linear Regressor

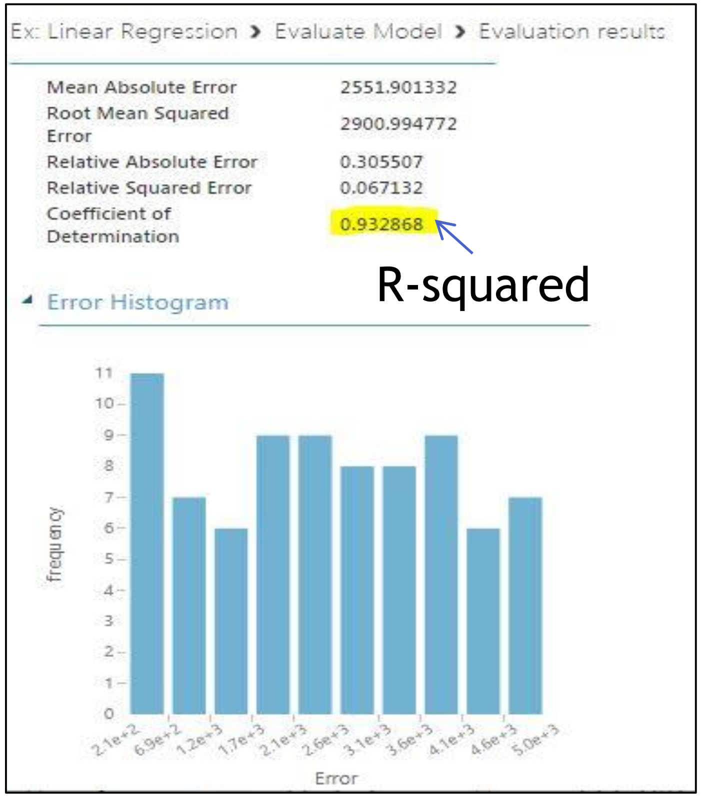
Settings

Setting	Value
Bias	True
Regularization	0.001
Allow Unknown Levels	True
Random Number Seed	

Feature Weights

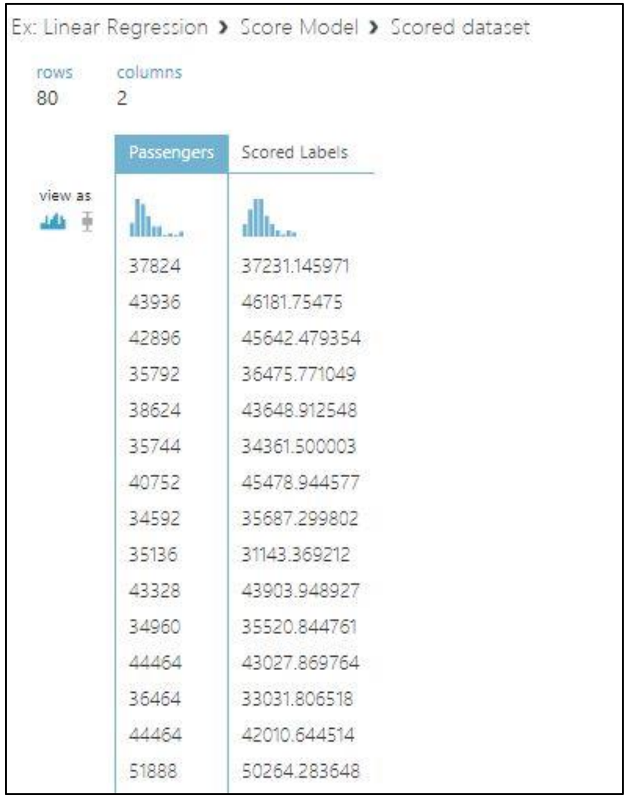
Feature	Weight
Bias	1259.34
Promotion_Budget	0.0695301

fig20: Evaluate Model Output


Ex: Linear Regression › Evaluate Model › Evaluation results

Mean Absolute Error	2551.901332
Root Mean Squared Error	2900.994772
Relative Absolute Error	0.305507
Relative Squared Error	0.067132
Coefficient of Determination	0.932868

fig21: Score Model (predicted values)


Ex: Linear Regression › Score Model › Scored dataset

rows 80 columns 2

view as

Passengers	Scored Labels
37824	37231.145971
43936	46181.75475
42896	45642.479354
35792	36475.771049
38624	43648.912548
35744	34361.500003
40752	45478.944577
34592	35687.299802
35136	31143.369212
43328	43903.948927
34960	35520.844761
44464	43027.869764
36464	33031.806518
44464	42010.644514
51888	50264.283648

Steps - Regression Line Fitting cont..

- To predict No. of passengers for a Promotion_Budget of 650,000
  - In the experiment click on  in the bottom pane
  - Select Retraining Web Service
  - Click on  to run in the bottom pane
  - After execution again click on  in the bottom pane and select Predictive Web Service
  - Again Click on  to run in the bottom pane
  - After execution click on  it will deploy and take you to the web service page
  - Click on the Test button, Enter data to predict window will open
  - In Promotion_Budget field enter 650000 and click on 
  - The prediction of No. of Passenger will be shown above the bottom pane

Steps - Regression Line Fitting cont..

fig22:Retraining Web Service

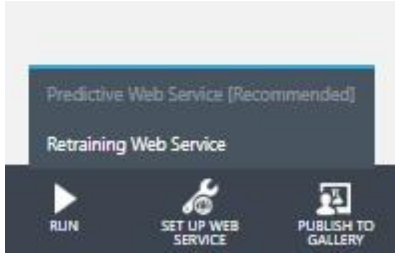
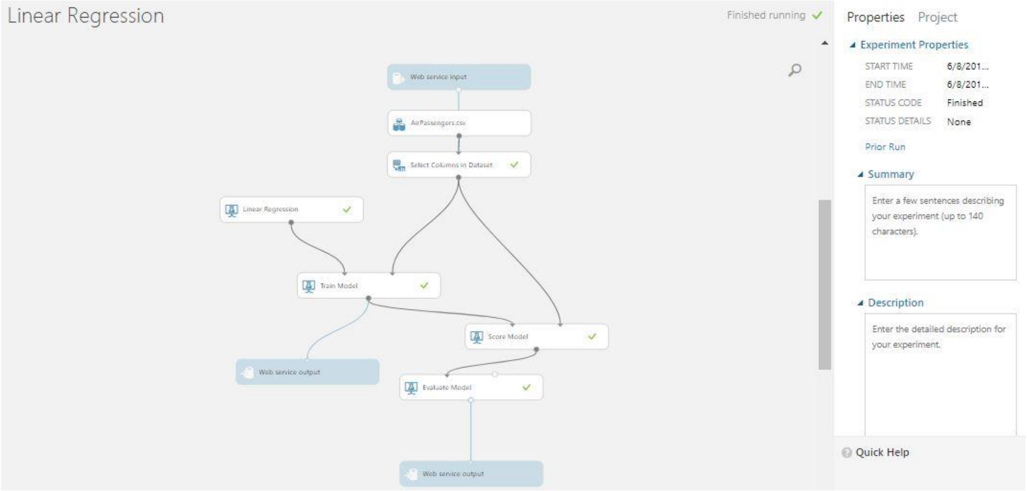


fig23: Retrain finished



Steps - Regression Line Fitting cont..

fig24: Precdictive Web Service

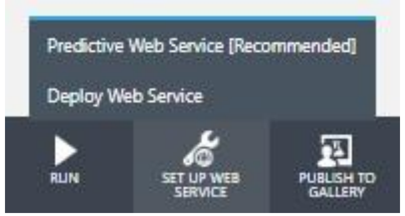
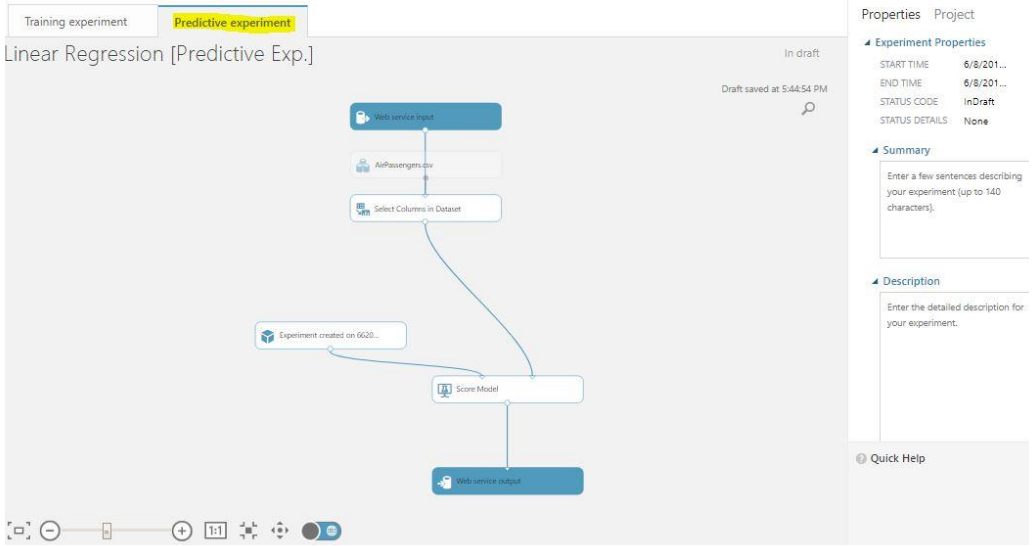


fig25: Precdictive Web Service Finished



statinfer

Steps - Regression Line Fitting cont..

fig26: Testing

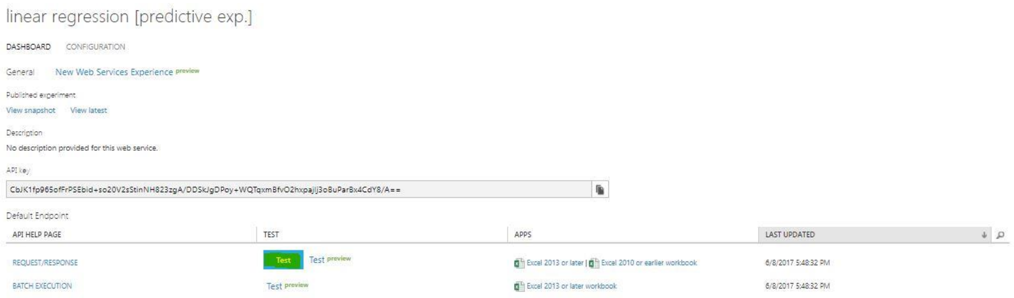
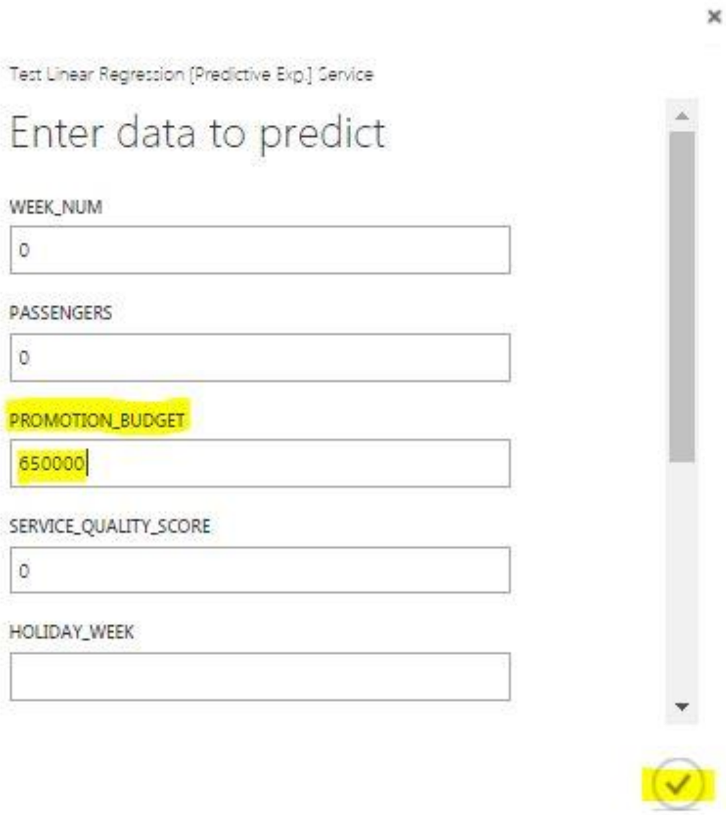


fig27: Predictor value



Steps - Regression Line Fitting cont..

fig29: Final Prediction

statinfer

How good is my regression line?

How good is my regression line?

- Take an (x,y) point from data.
- Imagine that we submitted x in the regression line, we got a prediction as y_{pred}
- If the regression line is a good fit then the we expect y_{pred}=y or (y-y_{pred}) =0
- At every point of x, if we repeat the same, then we will get multiple error values (y-y_{pred}) values
- Some of them might be positive, some of them may be negative, so we can take the square of all such errors

$$\boxed{SSE = \sum (y - \hat{y})^2}$$

How good is my regression line?

- For a good model we need SSE to be zero or near to zero
- Standalone SSE will not make any sense, For example SSE= 100, is very less when y is varying in terms of 1000's. Same value is very high when y is varying in terms of decimals.
- We have to consider variance of y while calculating the regression line accuracy

How good is my regression line?

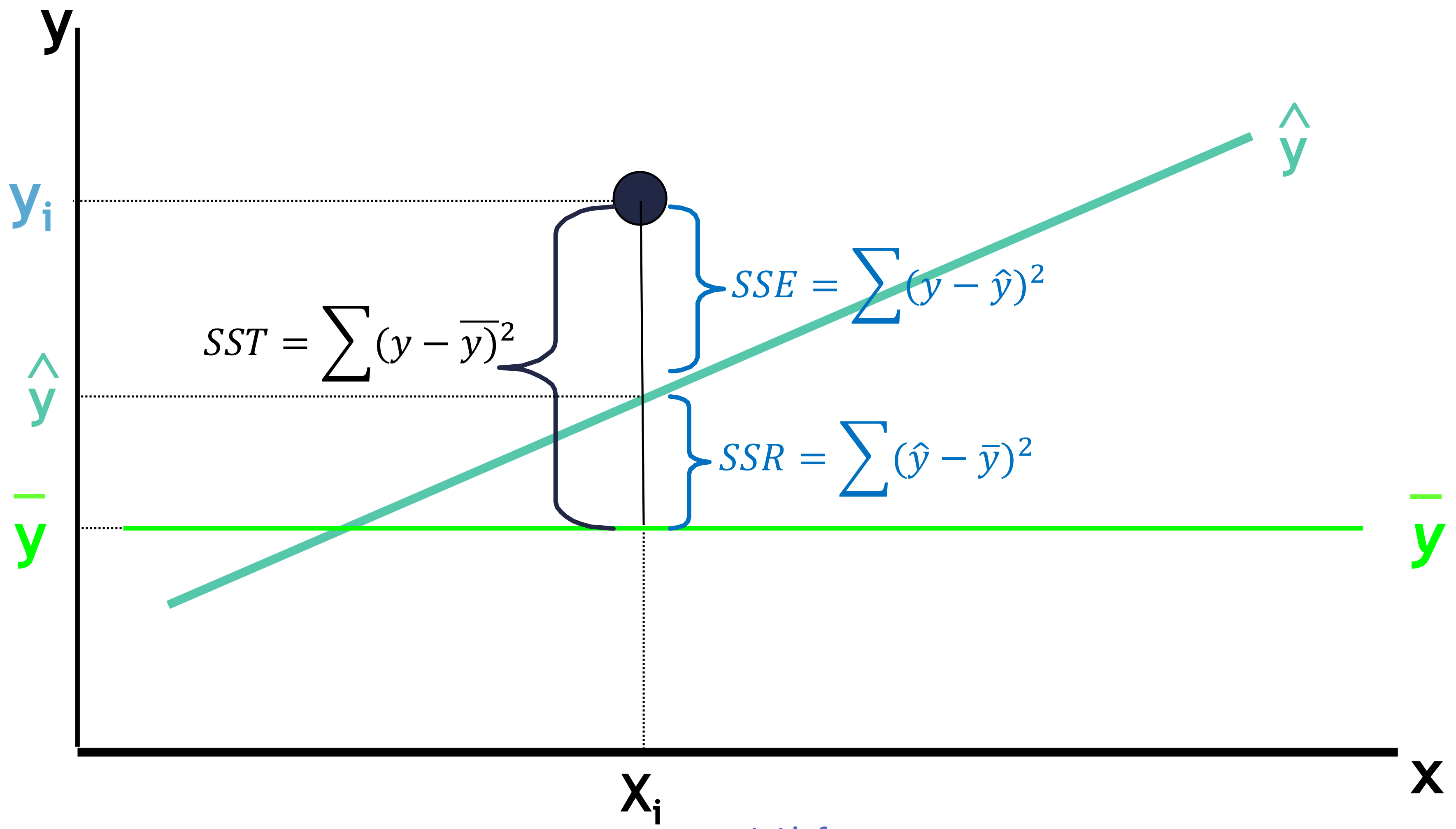
- Error Sum of squares (SSE- Sum of Squares of error)
  - $SSE = \sum(y - \hat{y})^2$
- Total Variance in Y (SST- Sum of Squares of Total)
  - $SST = \sum(y - \overline{y})^2$
  - $SST = \sum(y - \hat{y} + \hat{y} - \overline{y})^2$
  - $SST = \sum(y - \hat{y} + \hat{y} - \overline{y})^2$
  - $SST = \sum(y - \hat{y})^2 + \sum(\hat{y} - \bar{y})^2$
  - $\mathrm{SST} = \mathrm{SSE} + \sum(\hat{y} - \bar{y})^2$
  - $\mathrm{SST} = \mathrm{SSE} + SSR$

How good is my regression line?

- Total variance in Y is divided into two parts,
  - Variance that can't be explained by x (error)
  - Variance that can be explained by x, using regression

$$\boxed{SST = SSE + SSR}$$

Explained and Unexplained Variation



How good is my regression line?

- So, total variance in Y is divided into two parts,
  - Variance that can be explained by x, using regression
  - Variance that can't be explained by x

$$SST = SSE + SSR$$

SST	SSE	SSR
• Total sum of Squares	Sum of Squares Error	Sum of Squares Regression
$SST = \sum (y - \bar{y})^2$	$SSE = \sum (y - \hat{y})^2$	$SSR = \sum (\hat{y} - \bar{y})^2$

R-Squared

R-Squared

- A good fit will have
  - SSE (Minimum or Maximum?)
  - SSR (Minimum or Maximum?)
  - And we know SST= SSE + SSR
  - SSE/SST(Minimum or Maximum?)
  - SSR/SST(Minimum or Maximum?)
- The coefficient of determination is the portion of the total variation in the dependent variable that is explained by variation in the independent variable
- The coefficient of determination is also called R-squared and is denoted as R^2

$$\boxed{R^2 = \frac{SSR}{SST}} \quad \text{where} \quad \boxed{0 \le R^2 \le 1}$$

Lab: R- Squared

- What is the R-square value of Passengers vs Promotion_Budget model?
- What is the R-square value of Passengers vs Inter_metro_flight_ratio

Steps - R- Squared

- We have calculated the R-square value for Passengers vs Promotion_Budget (slide-40)
- Similarly for Passengers vs Inter_metro_flight_ratio we have to follow the same steps
- Only on change is in 'Select columns from the Dataset' select the columns Passengers and Inter_metro_flight_ratio

Steps - R- Squared cont..

fig29:Changing the columns

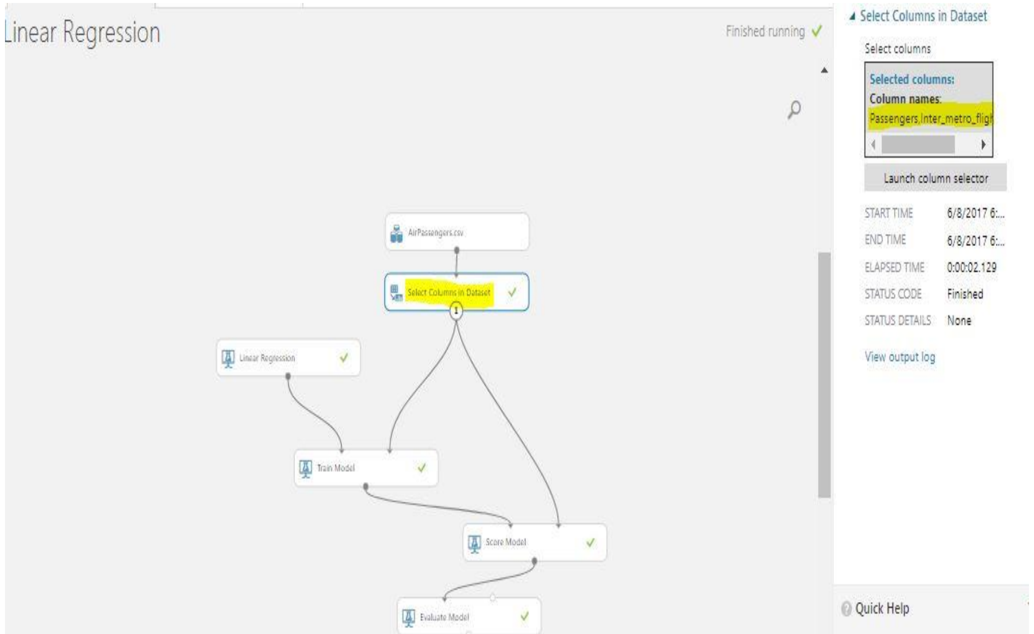
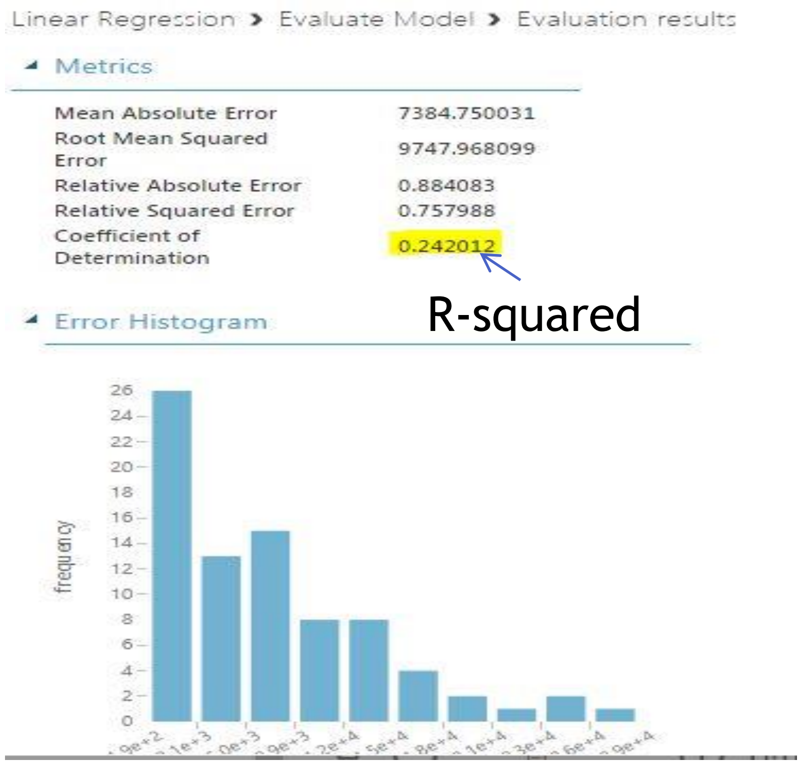


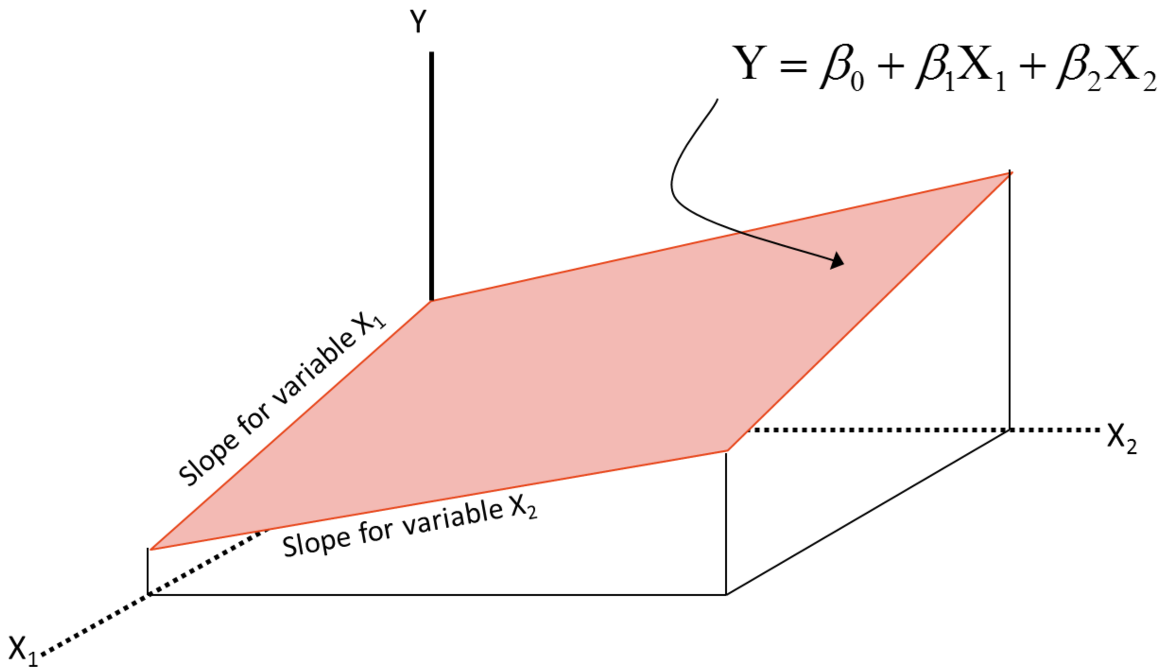
fig30: R-Square for Passenger vs Inter_Metro_Flight_Ratio



Multiple Regression

Multiple Regression

- Using multiple predictor variables instead of single variable
- We need to find a perfect plane here



LAB: Multiple Regression

- Build a multiple regression model to predict the number of passengers. Use three predictors Promotion_Budget, Inter_metro_flight_ratio and Service_Quality_Score
- What is R-square value
- Are there any predictor variables that are not impacting the dependent variable

Steps - Multiple Regression

- Multiple Regression for Predicting the No. of Passengers
  - Drag-and-drop AirPassengers.csv dataset to the canvas
  - Drag-and-drop ‘select column from dataset’ and select the columns
  - Search for ‘Linear Regression’, drag-and-drop it into the canvas
  - Click on ‘Linear Regression’ make sure that in properties window ‘Ordinary Least Squares’ is selected for solution method
  - Search for ‘Train Model’, drag-and-drop it into the canvas
  - Connect the output of ‘Linear Regression’ to left input of the ‘Train Model’ ‘select column from dataset’ to right input of the ‘Train Model’
  - Click on ‘Train Model’, select launch column selector in the properties window
  - Select the column(Passengers) for which the prediction to be done
  - Drag-and-drop ‘Score Model’ from left pane and uncheck the ‘Append score column’ in properties window

Steps - Multiple Regression cont..

- Connect the output of ‘Train Model’ to left input of the ‘Score Model’ ‘select column from dataset’ to right input of the ‘Score Model’
- Drag-and-drop ‘Evaluate Model’ from left pane
- Connect the output of ‘Score Model’ to the input of ‘Evaluate Model’
- Click on Run
- After execution click on the output circles of ‘Train Model’, ‘Score Model’ and ‘Evaluate Model’ to see the results

Steps - Multiple Regression cont..

fig31: Train Model

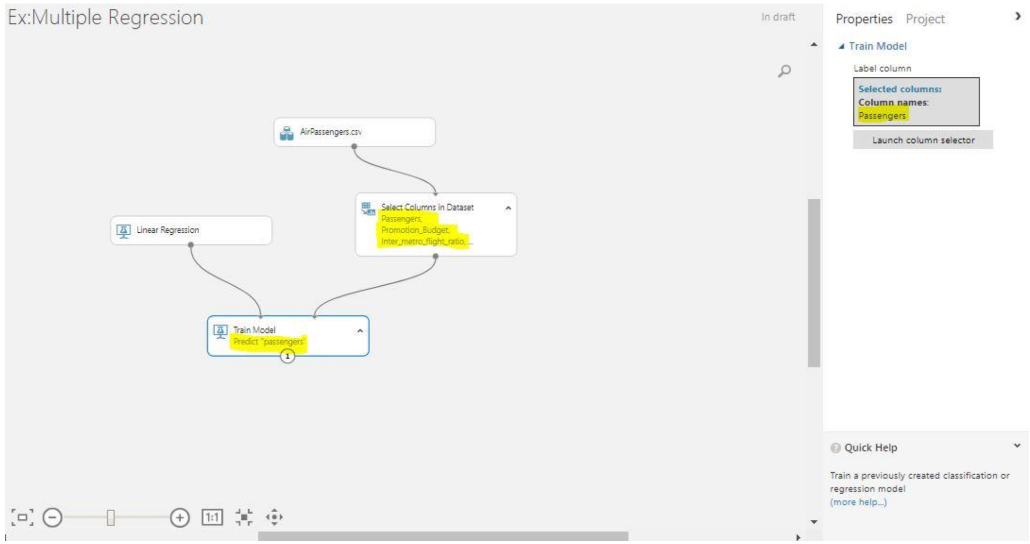
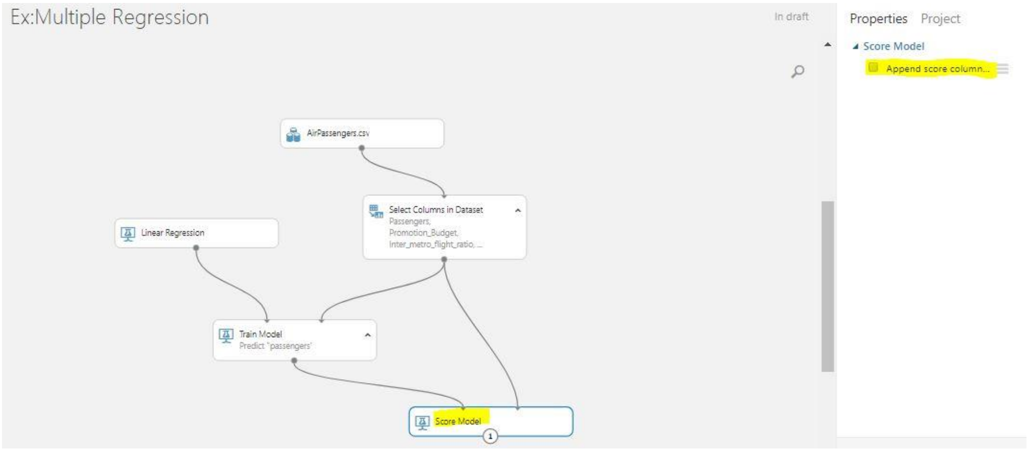


fig32: Score Model



Steps - Multiple Regression cont..

fig33: Evaluate Model

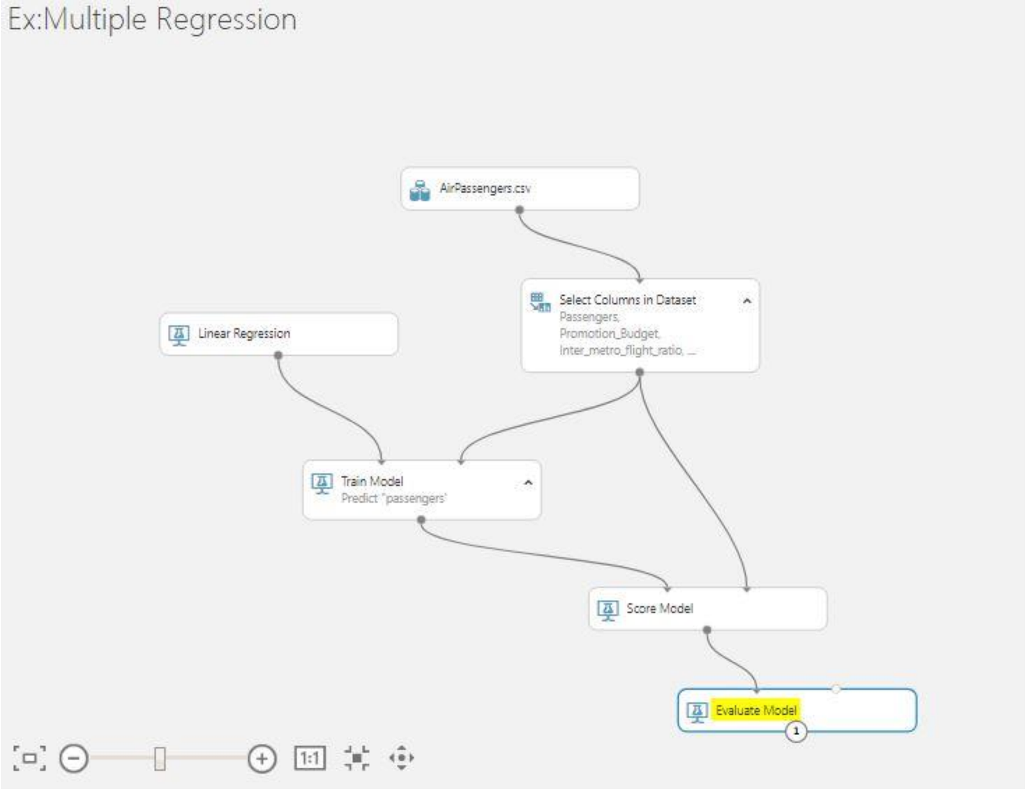
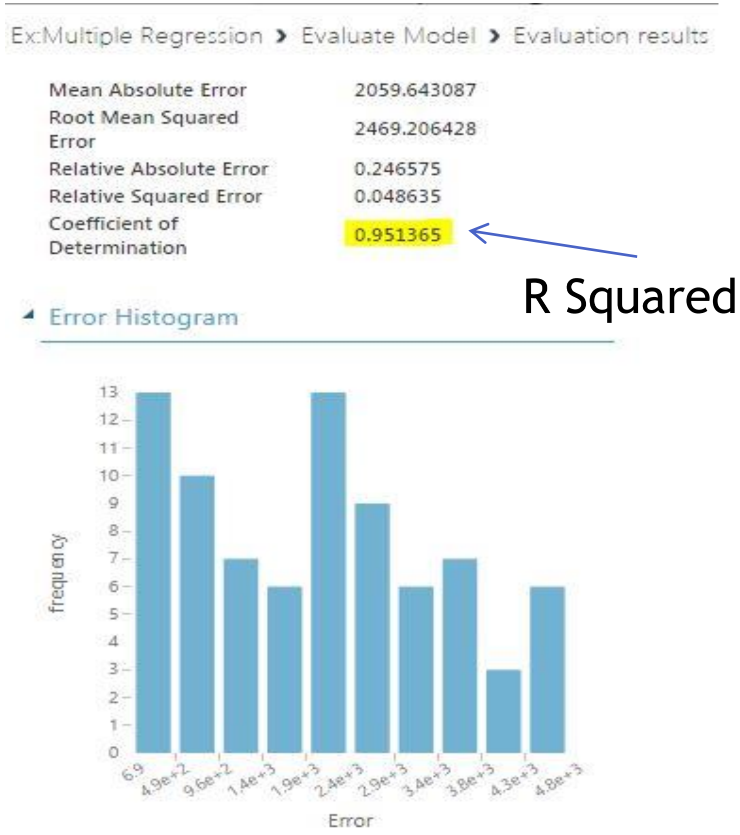


fig34: R Square



R Squared

Steps - Multiple Regression cont

fig35: Adding Execute R-Script

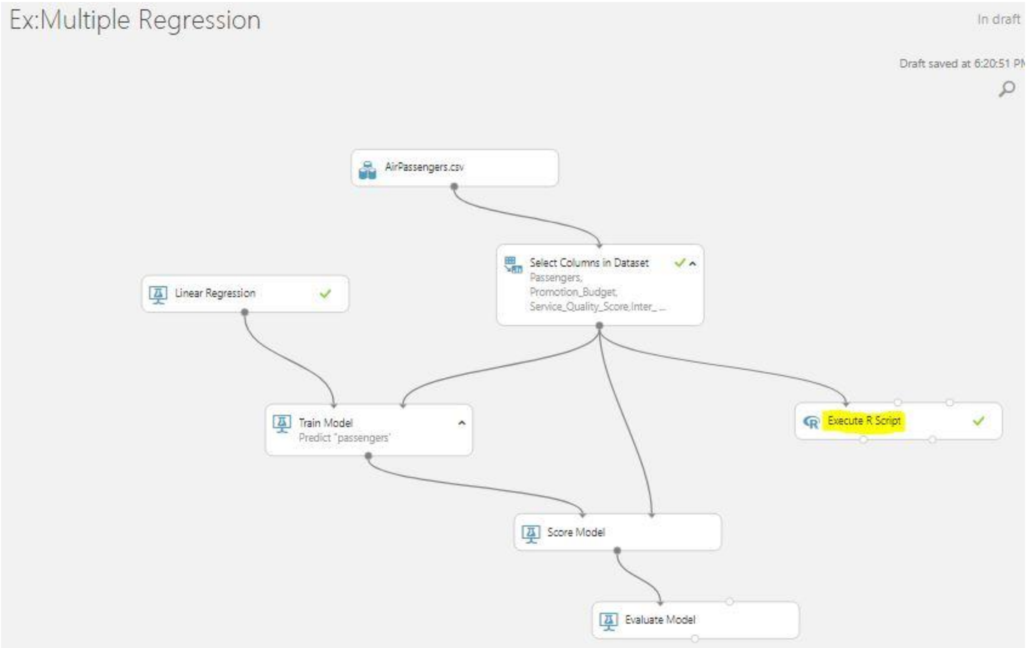


fig36: R-Script to check p-values of variables

R Script

```
dataset1 <- maml.mapInputPort(1) # class: data.frame

names<-lm(dataset1)
summary(names)

maml.mapOutputPort("dataset1");
```

Steps - Multiple Regression cont..

fig37: R Square value

Ex:Multiple Regression › Execute R Script › R Device

```
Residuals:
    Min      1Q  Median      3Q     Max
-4792.4 -1980.1    15.3  2317.9  4717.5

Coefficients:
                        Estimate Std. Error t value Pr(>|t|)
(Intercept)            1.921e+04  3.543e+03   5.424 6.68e-07 ***
Promotion_Budget       5.550e-02  3.586e-03  15.476  < 2e-16 ***
Service_Quality_Score -2.802e+03  5.304e+02  -5.283 1.17e-06 ***
Inter_metro_flight_ratio -2.003e+03  2.129e+03  -0.941     0.35
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2533 on 76 degrees of freedom
Multiple R-squared:  0.9514,    Adjusted R-squared:  0.9494
F-statistic: 495.6 on 3 and 76 DF,  p-value: < 2.2e-16
```

statinfer

Individual Impact of variables

Individual Impact of variables

- Look at the P-value
- Probability of the hypothesis being right.
- Individual variable coefficient is tested for significance
- Beta coefficients follow t distribution.
- Individual P values tell us about the significance of each variable
- A variable is significant if P value is less than 5%. Lesser the P-value, better the variable
- Note it is possible all the variables in a regression to produce great individual fits, and yet very few of the variables be individually significant.

To test

$$H_0 : \beta_i = 0$$
$$H_a : \beta_i \neq 0$$

Test statistic:

$$t = \frac{b_i}{s(b_i)}$$

Reject H_0 if

$$t > t(\frac{\alpha}{2}; n-k-1) \quad or$$
$$t < -t(\frac{\alpha}{2}; n-k-1)$$

Individual Impact of variables

- A variable is significant if P value is less than 5%.
- Lesser the P-value, better the variable
- If a variable has p-value less than 5%, if we drop that variable then we may see a drop in R-Squared value
- If a variable has p-value greater than 5%, if we drop that variable then we may not see any significant change in R-Squared value

LAB: Individual Impact of variables

- Build a multiple regression model to predict the number of passengers
- What is R-square value
- Are there any predictor variables that are not impacting the dependent variable
- Drop a low impacting variable and rebuild the model, is there any difference in R-Square?
- Drop a high impacting variable and rebuild the model, is there any difference in R-Square?

Steps - Individual Impact of variables

fig38: Drop low impacting variable (Inter_Metro_Flight_Ratio)

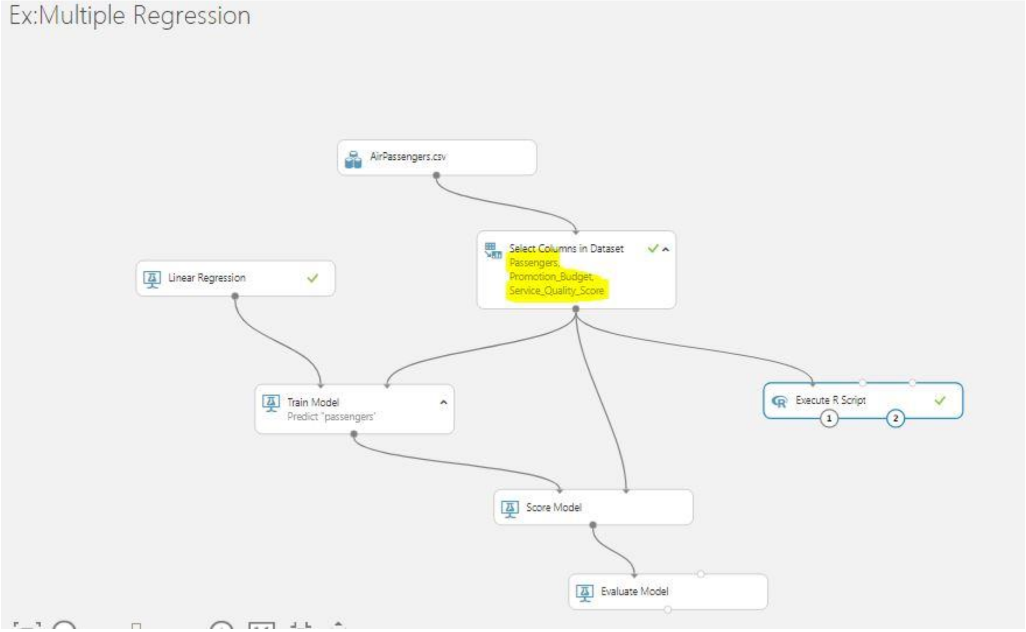


fig39: R-Squared(does not have much impact)


Ex:Multiple Regression › Execute R Script › R Device

```
Residuals:
    Min      1Q  Median      3Q     Max
-4834.1 -2191.8    34.4  2125.7  4810.9

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)           1.853e+04  3.465e+03   5.348 8.82e-07 ***
Promotion_Budget      5.440e-02  3.387e-03  16.063  < 2e-16 ***
Service_Quality_Score -2.807e+03  5.300e+02  -5.297 1.08e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2531 on 77 degrees of freedom
Multiple R-squared:  0.9508,    Adjusted R-squared:  0.9495
F-statistic:   744 on 2 and 77 DF,  p-value: < 2.2e-16
```

Steps - Individual Impact of variables

fig40: Drop high impacting variable(Promotional_Budget)

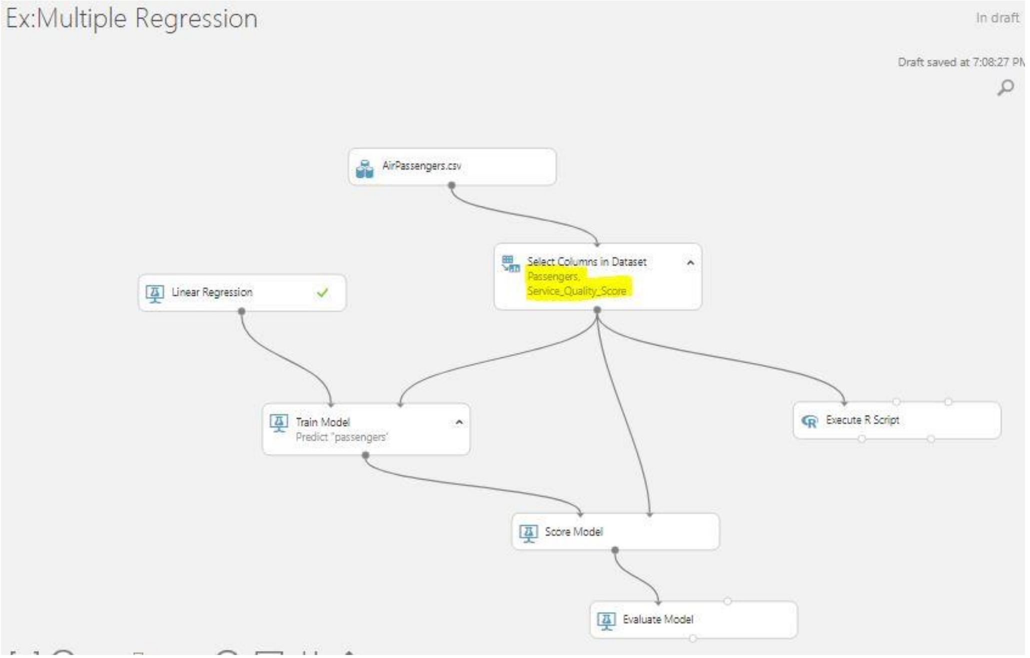


fig41: Huge impact on R-Squared value

Ex:Multiple Regression › Execute R Script › R Device

```
Residuals:
   Min     1Q Median     3Q    Max
-13158  -3376  -1117   3989  17251

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)            72519.5     1742.9   41.61   <2e-16 ***
Service_Quality_Score  -9986.6      590.1  -16.92   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5246 on 78 degrees of freedom
Multiple R-squared:  0.7859,	Adjusted R-squared:  0.7832
F-statistic: 286.4 on 1 and 78 DF,  p-value: < 2.2e-16
```

Adjusted R-Squared

Adjusted R-Squared

- Is it good to have as many independent variables as possible? Nope
- R-square is deceptive. R-squared never decreases when a new X variable is added to the model – True?
- We need a better measure or an adjustment to the original R-squared formula.

Adjusted R-Squared

- Adjusted R squared
  - Its value depends on the number of explanatory variables
  - Imposes a penalty for adding additional explanatory variables
  - It is usually written as (R-bar squared)
  - Very different from R when there are too many predictors and n is less

$$\overline{R}^2 = R^2 - \frac{k-1}{n-k}(1-R^2)$$

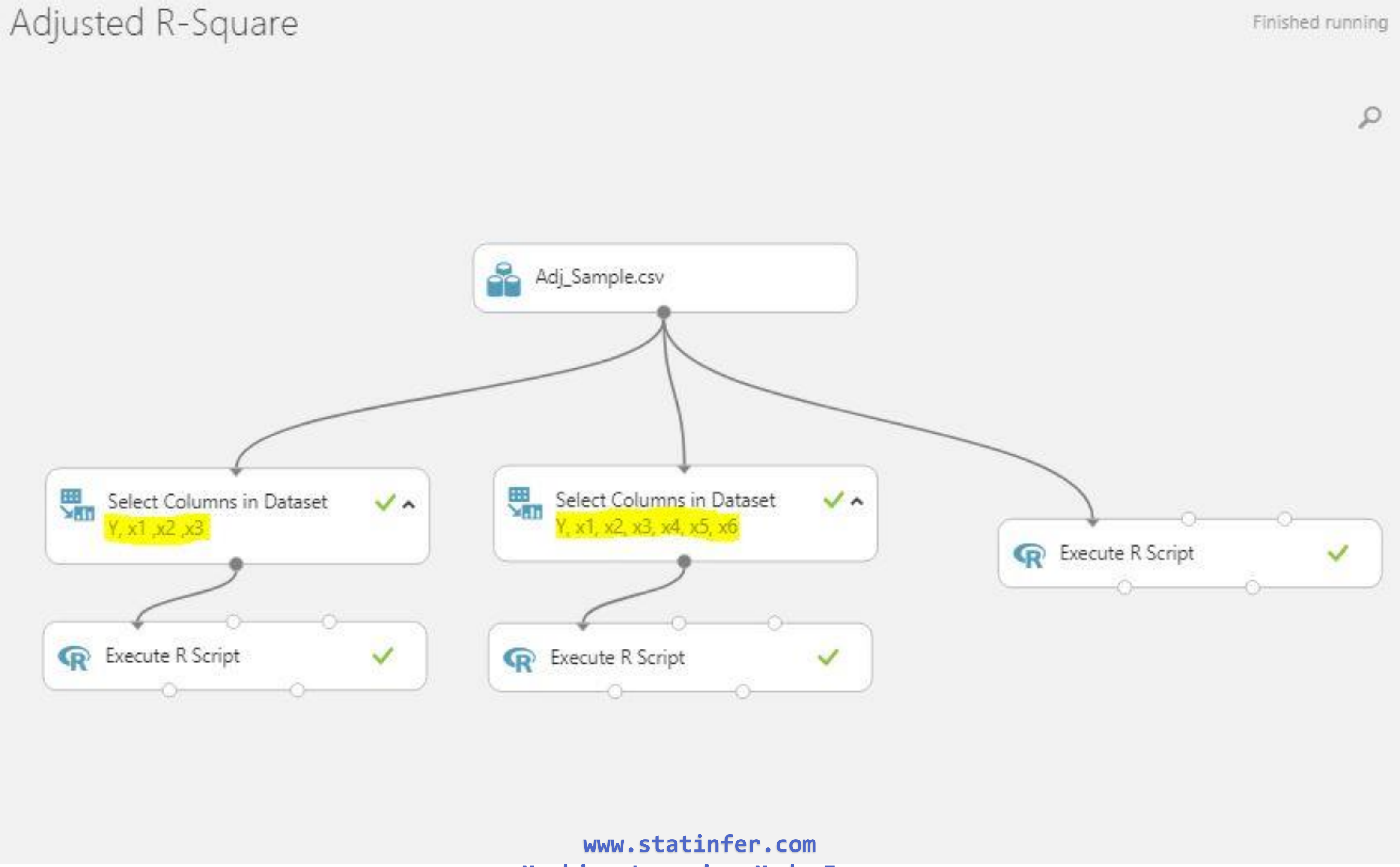
n-number of observations, k-number of parameters

LAB: Adjusted R-Square

- Dataset: "Adjusted Rsquare/ Adj_Sample.csv"
- Build a model to predict y using x1,x2 and x3. Note down R-Square and Adj R-Square values
- Build a model to predict y using x1,x2,x3,x4,x5 and x6. Note down R-Square and Adj R-Square values
- Build a model to predict y using x1,x2,x3,x4,x5,x6,x7 and x8. Note down R-Square and Adj R-Square values

Steps - Adjusted R-Square

fig42: Predicting Y with different set of variables



Steps - Adjusted R-Square cont..

fig43: Linear Regression (Y~x1+x2+x3)

fig44: Linear Regression(Y~x1+x2+x3+x4+x5+x6)

fig45: Linear Regression(with all variables)

Steps - Adjusted R-Square cont..

fig46: Linear Regression (Y~x1+x2+x3)

```
Adjusted R-Square › Execute R Script › R Device

Residuals:
     Min       1Q   Median       3Q      Max
-1.24893 -0.36289 -0.01435  0.52024  0.73439

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -2.879811   1.162727  -2.477   0.0383 *
x1          -0.489378   0.369691  -1.324   0.2222
x2           0.002854   0.001104   2.586   0.0323 *
x3           0.457233   0.176230   2.595   0.0319 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7068 on 8 degrees of freedom
Multiple R-squared:  0.6845,	Adjusted R-squared:  0.5662
F-statistic: 5.785 on 3 and 8 DF,  p-value: 0.02107
```

fig47: Linear Regression(Y~x1+x2+x3+x4+x5+x6)

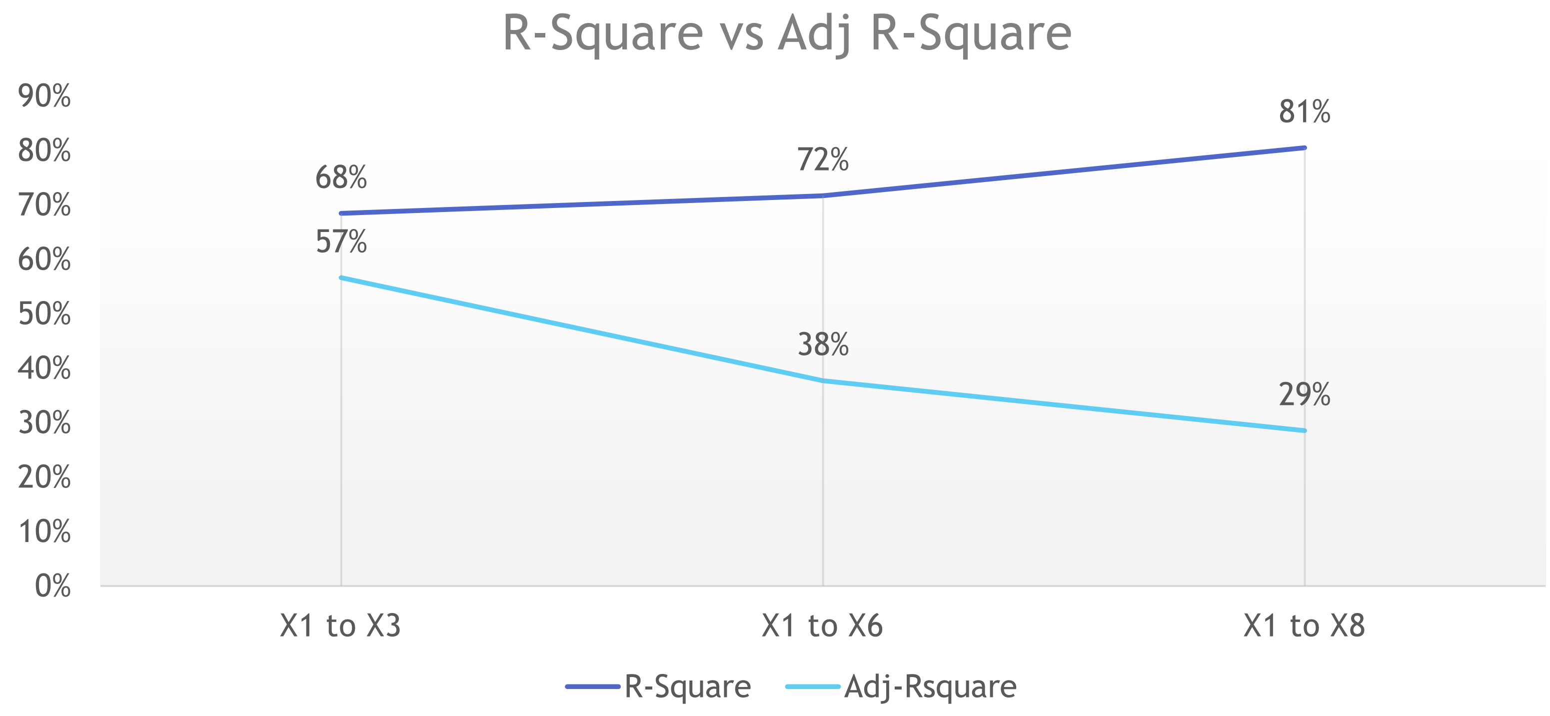
```
Adjusted R-Square › Execute R Script › R Device

Residuals:
       1        2        3        4        5        6        7        8
 0.25902  0.06800  0.45286  0.62004 -1.13449 -0.53961 -0.41898  0.52544
       9       10       11       12
-0.36028 -0.04814  0.83404 -0.25789

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -5.375099   4.686803  -1.147   0.3033
x1          -0.669681   0.536981  -1.247   0.2676
x2           0.002969   0.001518   1.956   0.1079
x3           0.506261   0.248695   2.036   0.0974 .
x4           0.037611   0.083834   0.449   0.6725
x5           0.043624   0.168830   0.258   0.8064
x6           0.051554   0.087708   0.588   0.5822
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.8468 on 5 degrees of freedom
Multiple R-squared:  0.7169,	Adjusted R-squared:  0.3773
F-statistic: 2.111 on 6 and 5 DF,  p-value: 0.2149
```

fig48: Linear Regression(with all variables)

```
Adjusted R-Square › Execute R Script › R Device

Residuals:
      1       2       3       4       5       6       7       8       9      10
 0.4989  0.4490 -0.1764  0.3267 -0.8213 -0.6679 -0.2299  0.2323 -0.2973  0.3333
     11      12
 0.6184 -0.2658

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) 17.0439629 19.9031715   0.856    0.455
x1          -0.0955943  0.7614799  -0.126    0.908
x2           0.0007376  0.0025362   0.291    0.790
x3           0.5157015  0.3062833   1.684    0.191
x4           0.0578632  0.1033356   0.560    0.615
x5           0.0858136  0.1914803   0.448    0.684
x6          -0.1746565  0.2197152  -0.795    0.485
x7          -0.0323678  0.1530067  -0.212    0.846
x8          -0.2321183  0.2065655  -1.124    0.343

Residual standard error: 0.9071 on 3 degrees of freedom
Multiple R-squared:  0.8051,	Adjusted R-squared:  0.2855
F-statistic: 1.549 on 8 and 3 DF,  p-value: 0.3927
```

R-Squared vs Adjusted R-Squared

Build three models on Adj_sample data; m1, m2 and m3 with different number of variables



Multiple Regression- issues

LAB: Multiple Regression- issues

- Import Final Exam Score data
- Build a model to predict final score using the rest of the variables.
- How are Sem2_Math & Final score related? As Sem2_Math score increases, what happens to Final score?
- Remove "Sem1_Math" variable from the model and rebuild the model
- Is there any change in R square or Adj R square
- How are Sem2_Math & Final score related now? As Sem2_Math score increases, what happens to Final score?
- Draw a scatter plot between Sem1_Math & Sem2_Math
- Find the correlation between Sem1_Math & Sem2_Math

Steps - Multiple Regression- issues

Fig49:Linear Regression(all variables)

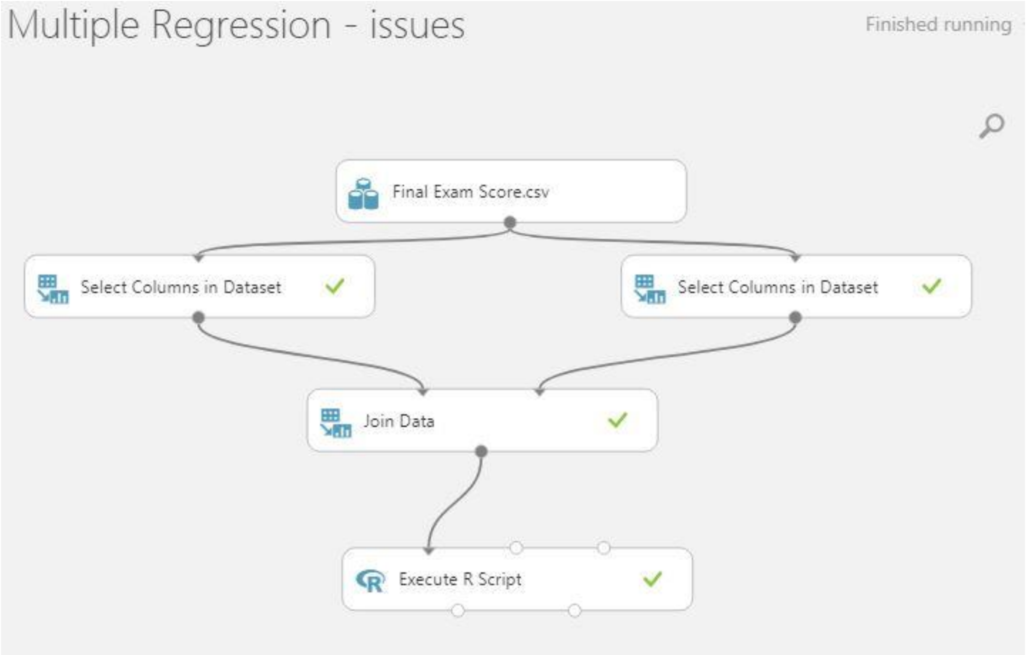
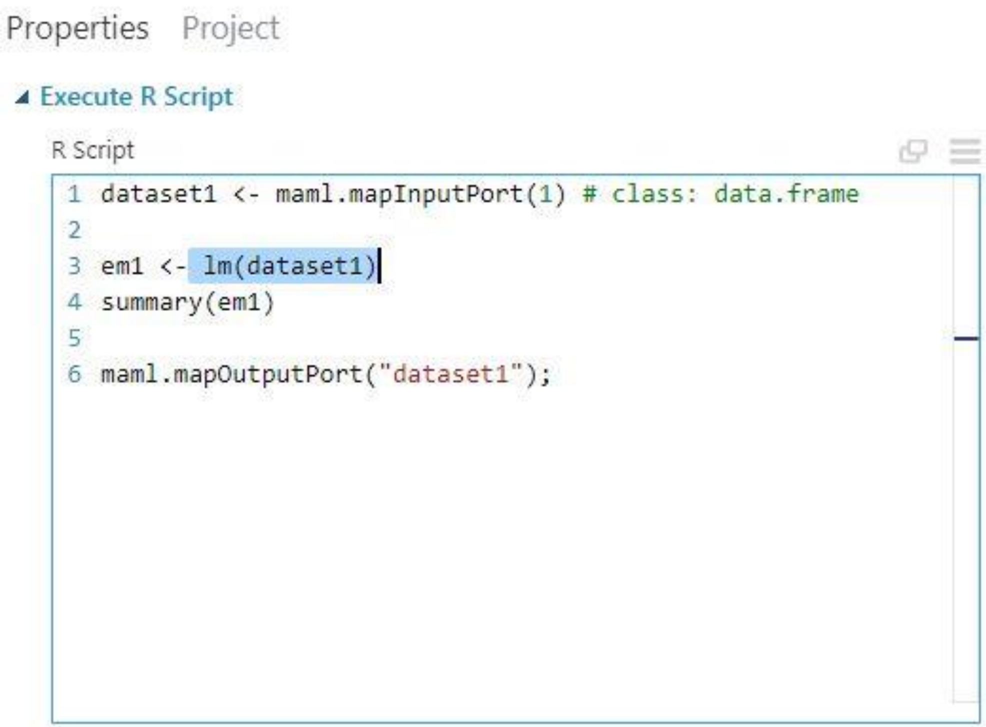


Fig50: R-Script



Steps - Multiple Regression- issues

fig51:Negative impact

Multiple Regression - issues › Execute R Script › R Device

```
Residuals:
    Min      1Q  Median      3Q     Max
-1.9199 -0.7728 -0.1456  0.3439  2.9638

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  -2.59173    1.82528  -1.420   0.1650
Sem1_Science  0.14069    0.05404   2.604   0.0137 *
Sem2_Science  0.28936    0.04104   7.051 4.54e-08 ***
Sem1_Math     0.88015    0.14943   5.890 1.33e-06 ***
Sem2_Math    -0.26064    0.14630  -1.781   0.0840 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.32 on 33 degrees of freedom
Multiple R-squared:  0.9855,	Adjusted R-squared:  0.9837
F-statistic: 560.4 on 4 and 33 DF,  p-value: < 2.2e-16
```

fig52: Positive impact(without Sem2_Math)

Multiple Regression - issues › Execute R Script › R Device

```
Residuals:
    Min      1Q  Median      3Q     Max
-2.8202 -1.4051 -0.1948  0.7619  4.3065

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  -3.62168    2.56368  -1.413   0.1668
Sem1_Science  0.16770    0.07598   2.207   0.0341 *
Sem2_Science  0.29794    0.05787   5.149 1.10e-05 ***
Sem2_Math     0.56328    0.06050   9.311 7.03e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.863 on 34 degrees of freedom
Multiple R-squared:  0.9702,	Adjusted R-squared:  0.9676
F-statistic: 369.5 on 3 and 34 DF,  p-value: < 2.2e-16
```

Steps - Multiple Regression- issues

fig53:Scatter Plot(Sem1_Math vs Sem2_Math)

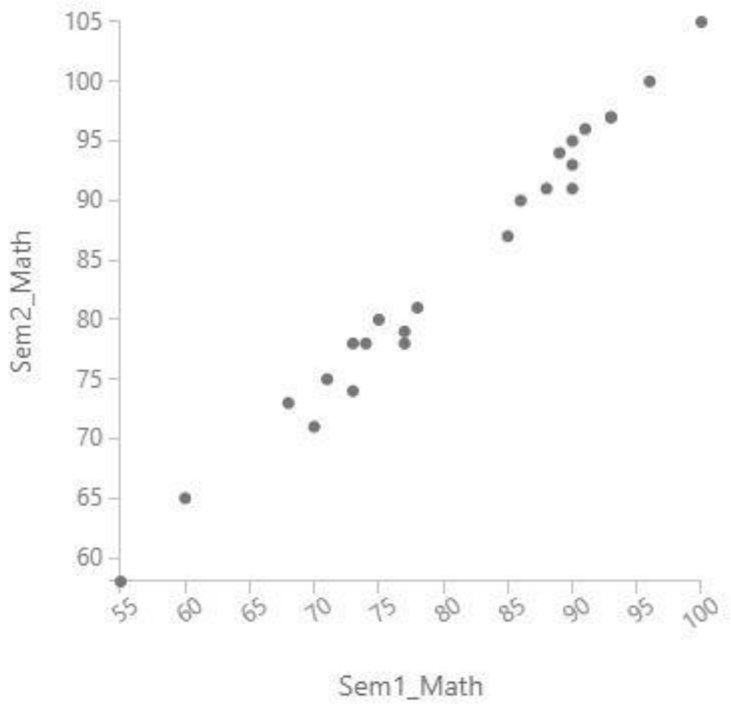


fig54:Correlation(Sem1_Math vs Sem2_Math)

Sem1_Math	Sem2_Math
1	0.992495
0.992495	1

statinfer

Multicollinearity

Multicollinearity

- Multiple regression is wonderful - In that it allows you to consider the effect of multiple variables simultaneously.
- Multiple regression is extremely unpleasant -Because it allows you to consider the effect of multiple variables simultaneously.
- The relationships between the explanatory variables are the key to understanding multiple regression.
- Multicollinearity (or inter correlation) exists when at least some of the predictor variables are correlated among themselves.

Multicollinearity

- The parameter estimates will have inflated variance in presence of multicollineraity
- Sometimes the signs of the parameter estimates tend to change
- If the relation between the independent variables grows really strong then the variance of parameter estimates tends to be infinity – Can you prove it?

Multicollinearity detection

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4$$

- Build a model X1 vs X2 X3 X4 find R square, say R1
- Build a model X2 vs X1 X3 X4 find R square, say R2
- Build a model X3 vs X1 X2 X4 find R square, say R3
- Build a model X4 vs X1 X2 X3 find R square, say R4

- For example if R3 is 95% then we don't really need X3 in the model
- Since it can be explained as liner combination of other three
- For each variable we find individual R square.

statinfer

Multicollinearity detection

- $1/(1-R^2)$ is called VIF.
- VIF option in R automatically calculates VIF values for each of the predictor variables

R Square	40%	50%	60%	70%	75%	80%	90%
VIF	1.67	2.00	2.50	3.33	4.00	5.00	10.00

LAB: Multicollinearity

- Identify the Multicollinearity in the Final Exam Score model
- Drop the variable one by one to reduce the multicollinearity
- Identify and eliminate the Multicollinearity in the Air passengers model

Steps - Multicollinearity

fig55: VIF (all variables)

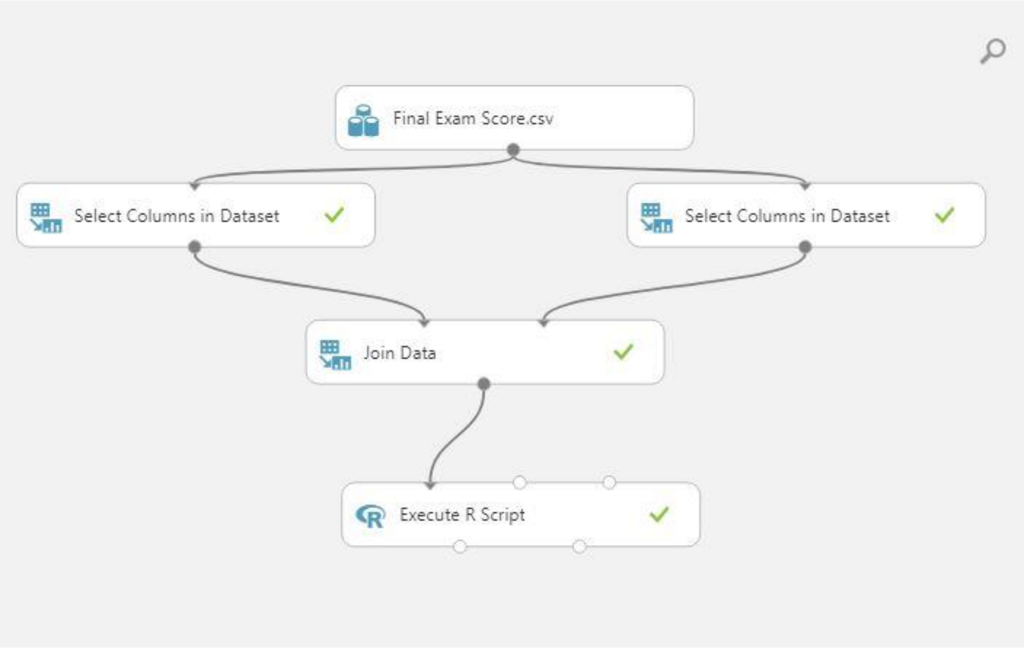
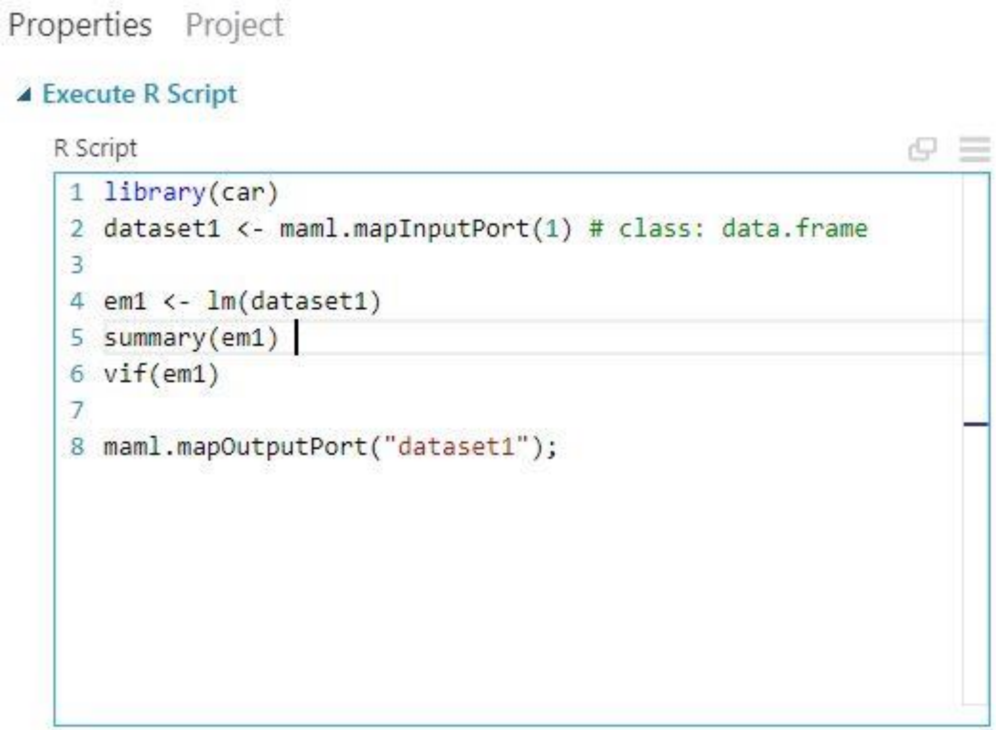


fig56: R-Script (VIF)



statinfer

Steps - Multicollinearity

fig57:Variables with high VIF Values(Sem1_Math)

Multiple Regression - issues › Execute R Script › R Device

```
Residuals:
    Min      1Q  Median      3Q     Max
-1.9199 -0.7728 -0.1456  0.3439  2.9638

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  -2.59173    1.82528  -1.420   0.1650
Sem1_Science  0.14069    0.05404   2.604   0.0137 *
Sem2_Science  0.28936    0.04104   7.051 4.54e-08 ***
Sem1_Math     0.88015    0.14943   5.890 1.33e-06 ***
Sem2_Math    -0.26064    0.14630  -1.781   0.0840 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.32 on 33 degrees of freedom
Multiple R-squared:  0.9855,	Adjusted R-squared:  0.9837
F-statistic: 560.4 on 4 and 33 DF,  p-value: < 2.2e-16

Sem1_Science Sem2_Science    Sem1_Math    Sem2_Math
    6.650747     4.007667    49.787651    49.648860
```

fig57:Variables with high VIF Values(Sem1_Science)

Multiple Regression - issues › Execute R Script › R Device

```
Residuals:
    Min      1Q  Median      3Q     Max
-2.8202 -1.4051 -0.1948  0.7619  4.3065

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  -3.62168    2.56368  -1.413   0.1668
Sem1_Science  0.16770    0.07598   2.207   0.0341 *
Sem2_Science  0.29794    0.05787   5.149 1.10e-05 ***
Sem2_Math     0.56328    0.06050   9.311 7.03e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.863 on 34 degrees of freedom
Multiple R-squared:  0.9702,	Adjusted R-squared:  0.9676
F-statistic: 369.5 on 3 and 34 DF,  p-value: < 2.2e-16

Sem1_Science Sem2_Science    Sem2_Math
    6.602866     4.002612     4.263823
```

Steps - Multicollinearity

fig57:Variables with low VIF Values

Multiple Regression - issues › Execute R Script › R Device

```
Residuals:
    Min      1Q  Median      3Q     Max 
-3.0619 -1.5087 -0.4414  1.3304  3.9521 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -3.67271    2.70167  -1.359    0.183    
Sem2_Science  0.37470    0.04875   7.687 5.08e-09 ***
Sem2_Math     0.64776    0.04938  13.119 4.49e-15 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.963 on 35 degrees of freedom
Multiple R-squared:  0.966,	Adjusted R-squared:  0.964 
F-statistic: 496.9 on 2 and 35 DF,  p-value: < 2.2e-16

Sem2_Science    Sem2_Math 
    2.557323     2.557323 
```

statinfer

Multiple Regression model building

statinfer

Lab: Multiple Regression

- Dataset: Webpage_Product_Sales/Webpage_Product_Sales.csv
- Build a model to predict sales using rest of the variables
- Drop the less impacting variables based on p-values.
- Is there any multicollinearity?
- How many variables are there in the final model?
- What is the R-squared of the final model?
- Can you improve the model using same data and variables?

Steps - Multiple Regression

fig60: Multiple Regression(Web_Products_Sales)

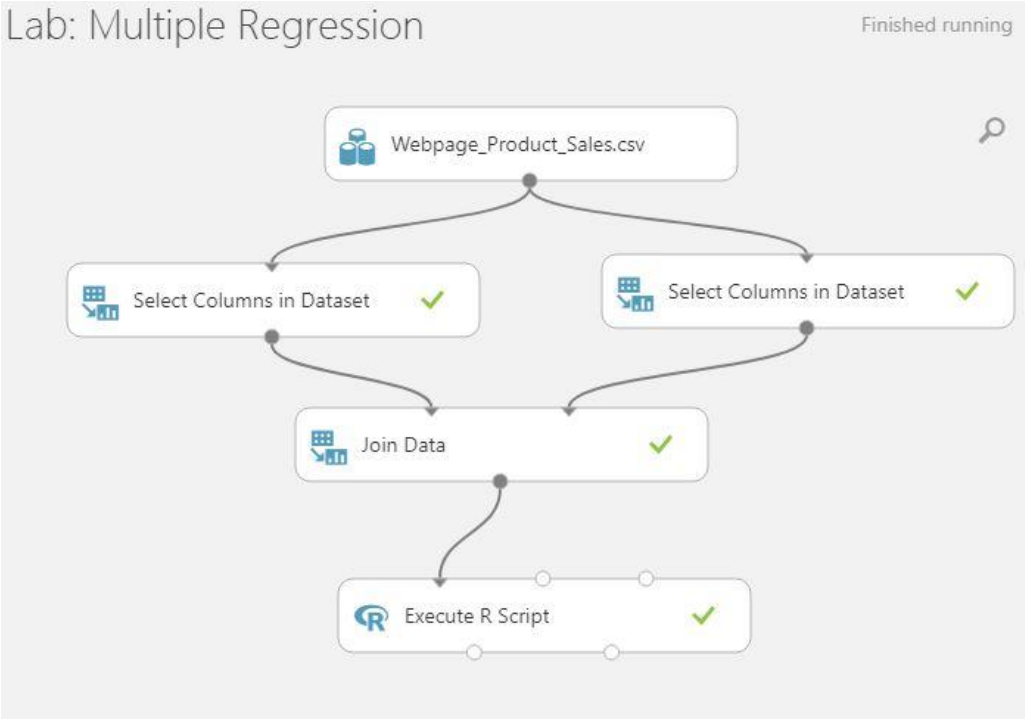
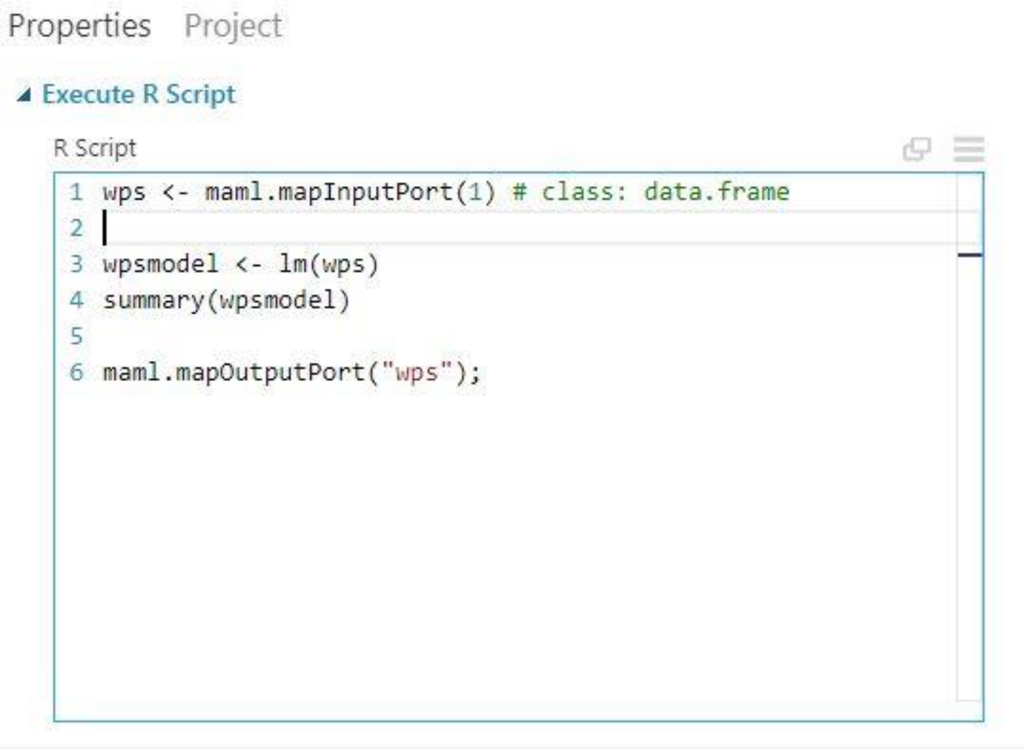


fig61:R-Script(creating Model)



Steps - Multiple Regression

fig62:Removing Variables with high P-Values

Lab: Multiple Regression › Execute R Script › R Device

```
Coefficients:
                        Estimate Std. Error t value Pr(>|t|)
(Intercept)             6.578e+03  1.269e+03   5.185 2.85e-07 ***
DayofMonth              4.705e+01  1.497e+01   3.142  0.00175 **
Weekday                 1.352e+03  6.625e+01  20.414  < 2e-16 ***
Month                   4.828e+02  4.106e+01  11.759  < 2e-16 ***
Social_Network_Ref_links  6.709e+00  4.054e-01  16.551  < 2e-16 ***
Online_Ad_Paid_ref_links  6.001e+00  9.892e-01   6.067 2.17e-09 ***
Clicks_From_Serach_Engine  2.614e-03  9.312e-01   0.003  0.99776
Special_Discount        4.661e+03  3.980e+02  11.712  < 2e-16 ***
Holiday                 1.882e+04  6.795e+02  27.691  < 2e-16 ***
Server_Down_time_Sec   -1.344e+02  1.382e+01  -9.724  < 2e-16 ***
Web_UI_Score           -5.596e+00  1.133e+01  -0.494  0.62153
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3468 on 680 degrees of freedom
Multiple R-squared:  0.8159,	Adjusted R-squared:  0.8132
F-statistic: 301.3 on 10 and 680 DF,  p-value: < 2.2e-16
```

fig63:R-Squared Value after removing

Lab: Multiple Regression › Execute R Script › R Device

```
Coefficients:
                         Estimate Std. Error t value Pr(>|t|)
(Intercept)             6136.6711   812.7562   7.550  1.4e-13 ***
DayofMonth                46.9327    14.9499   3.139  0.00177 **
Weekday                 1351.3951    66.1292  20.436  < 2e-16 ***
Month                    481.8808    40.8843  11.786  < 2e-16 ***
Social_Network_Ref_links    6.6977     0.4034  16.602  < 2e-16 ***
Online_Ad_Paid_ref_links    6.0116     0.2856  21.052  < 2e-16 ***
Special_Discount        4671.4612   395.4466  11.813  < 2e-16 ***
Holiday                18789.5172   675.0592  27.834  < 2e-16 ***
Server_Down_time_Sec    -134.4744    13.7791  -9.759  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3463 on 682 degrees of freedom
Multiple R-squared:  0.8158,	Adjusted R-squared:  0.8137
F-statistic: 377.6 on 8 and 682 DF,  p-value: < 2.2e-16
```

statinfer

Steps - Multiple Regression

fig64:R-Script(VIF)

Properties   Project

◢ Execute R Script

R Script

```
library(car)
wps <- maml.mapInputPort(1) # class: data.frame

wpsmodel <- lm(wps)
summary(wpsmodel)
vif(wpsmodel)

maml.mapOutputPort("wps");
```

fig65:Final R-Squared Value(no high values in VIF)

```
Residual standard error: 3463 on 682 degrees of freedom
Multiple R-squared:  0.8158,	Adjusted R-squared:  0.8137
F-statistic: 377.6 on 8 and 682 DF,  p-value: < 2.2e-16

          DayofMonth              Weekday                Month
            1.003920             1.004835             1.011854
Social_Network_Ref_links Online_Ad_Paid_ref_links     Special_Discount
            1.005806             1.017814             1.351502
             Holiday  Server_Down_time_Sec
            1.364391             1.017948
```

Conclusion - Regression

Conclusion - Regression

- Try adding the polynomial & interaction terms to your regression line. Sometimes they work like a charm.
- Adjusted R-squared is a good measure of training/in time sample error. We can't be sure about the final model performance based on this. We may have to perform cross-validation to get an idea on testing error.
- Outliers can influence the regression line, we need to take care of data sanitization before building the regression line.

Thank you

Part 6/12 - Logistic Regression Analysis With Azure

Venkat Reddy Konasani

Contents

Contents

- What is the need of logistic regression?
- Building logistic Regression line
- Goodness of fit measures
- Multicollinearity
- Individual Impact of variables
- Model selection

What is the need of non-linear regression?

LAB: What is the need of logistic regression?

statinfer

- Dataset: Product Sales Data/Product_sales.csv
- What are the variables in the dataset?
- Build a predictive model for Bought vs Age
- What is R-Square?
- If Age is 4 then will that customer buy the product?
- If Age is 105 then will that customer buy the product?

Steps - need of logistic regression

- Drag and drop the Data set (Product Sales Data/Product_sales.csv)
- Click on output circle and then visualize
- Check out the column names
- First find out the dimensions and of the dataset
- And then build a linear regression Model for Bought Vs Age
  - Drag-and-drop ‘select column from dataset’ and select both Bought and Age columns
  - Search for ‘Linear Regression’, drag-and-drop it into the canvas
  - Click on ‘Linear Regression’ make sure that in properties window ‘Ordinary Least Squares’ is selected for solution method
  - Search for ‘Train Model’, drag-and-drop it into the canvas
  - Connect the output of ‘Linear Regression’ to left input of the ‘Train Model’ ‘select column from dataset’ to right input of the ‘Train Model’
  - Click on ‘Train Model’, select launch column selector in the properties window
  - Select the column(Bought) for which the prediction to be done
  - Drag-and-drop ‘Score Model’ from left pane and uncheck the ‘Append score column’ in properties window

Steps - need of logistic regression

- Connect the output of ‘Train Model’ to left input of the ‘Score Model’ ‘select column from dataset’ to right input of the ‘Score Model’
- Drag-and-drop ‘Evaluate Model’ from left pane
- Connect the output of ‘Score Model’ to the input of ‘Evaluate Model’
- Click on Run 

- After execution click on the output circles of ‘Train Model’, ‘Score Model’ and ‘Evaluate Model’ to see the value of R-squared
- Now we need to predict if the Age is 4 will that customer will purchase the product or not 

Steps - need of logistic regression

- In the experiment click on 

 in the bottom pane
- Select Retraining Web Service
- Click on 

 to run in the bottom pane
- After execution again click on 

 in the bottom pane and select Predictive Web Service
- Again Click on 

 to run in the bottom pane
- After execution click on 

 it will deploy and take you to the web service page
- Click on the Test button, Enter data to predict window will open

Steps - need of logistic regression

Fig1:visualize the data

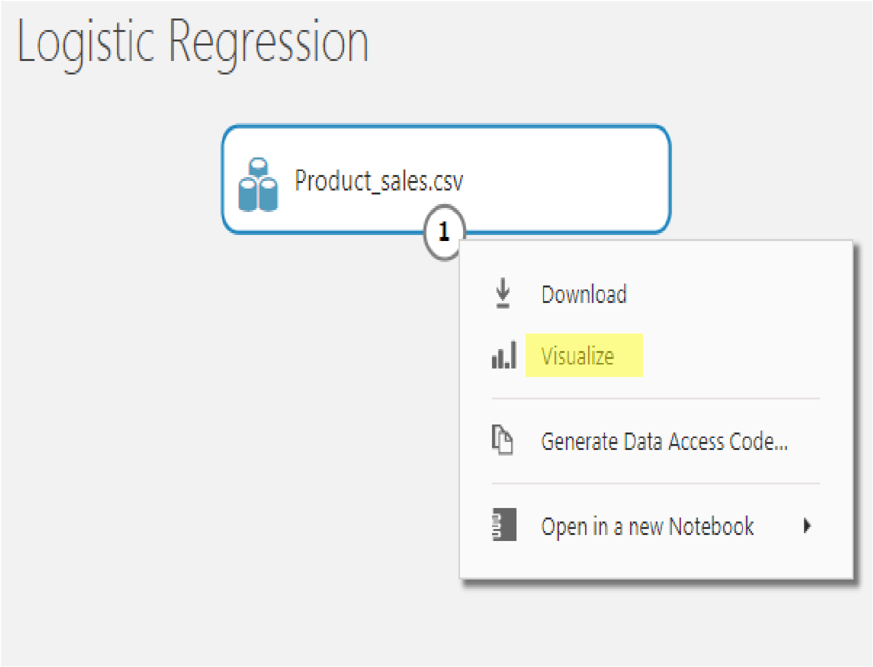
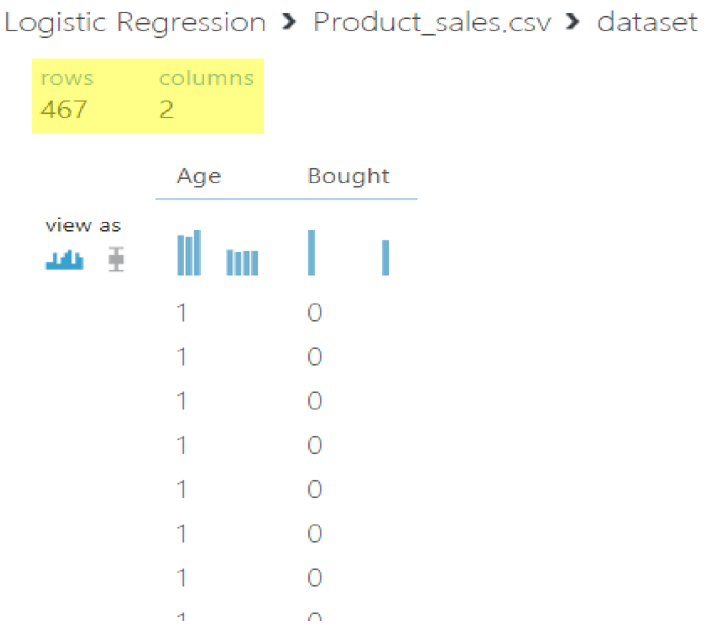


Fig2:Dimension of the dataset



Steps - need of logistic regression

Fig3: Selecting the columns (Bought and Age

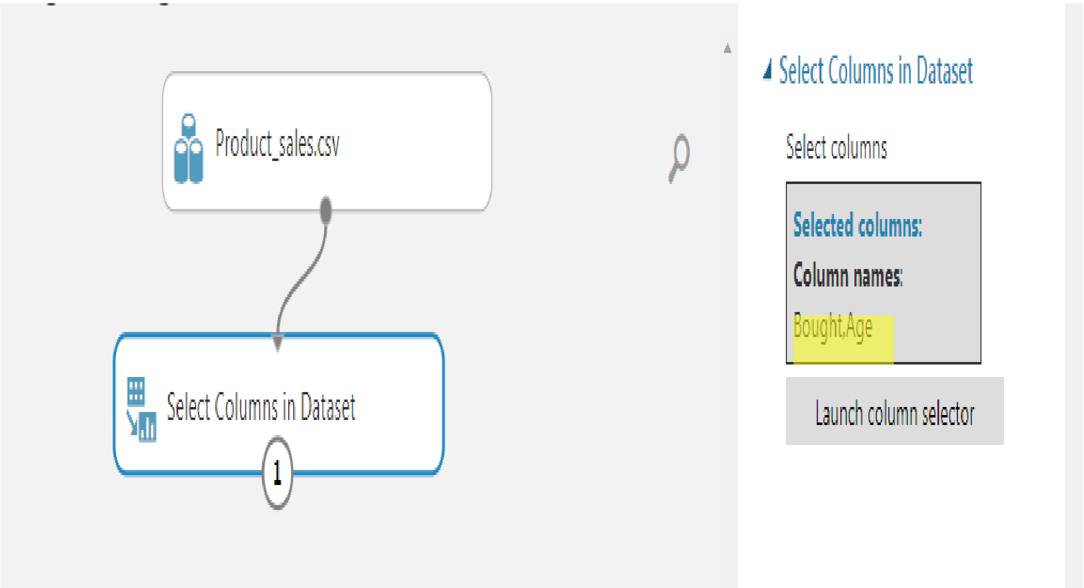
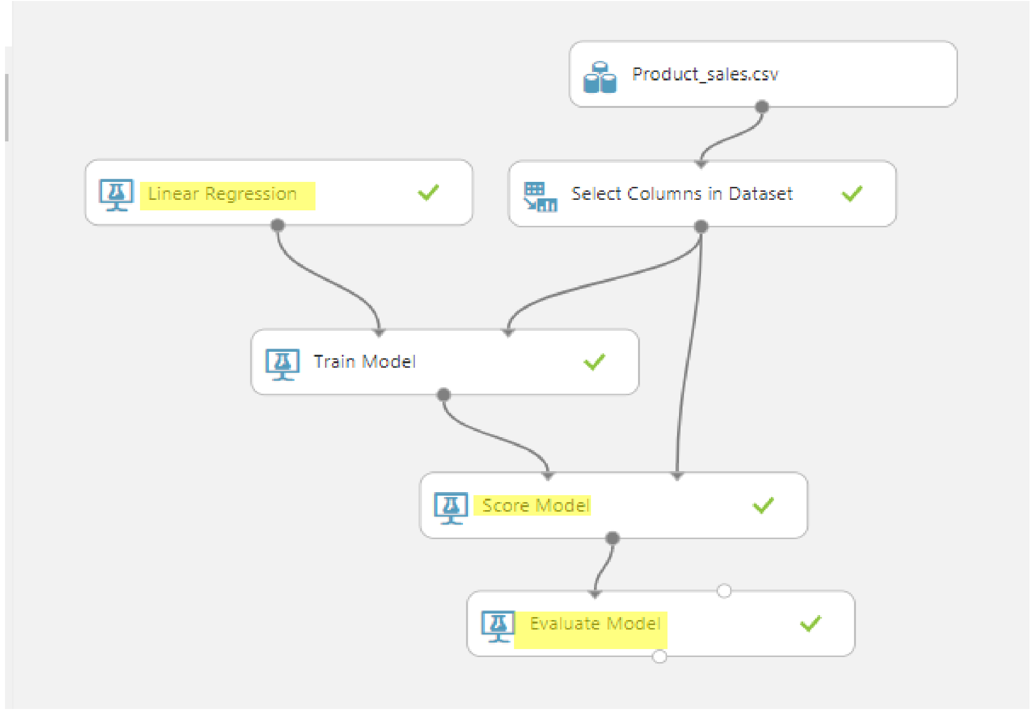


Fig4: Trained model



Steps - need of logistic regression

- Fig5: R-squared

Logistic Regression › Evaluate Model › Evaluation results

Metrics

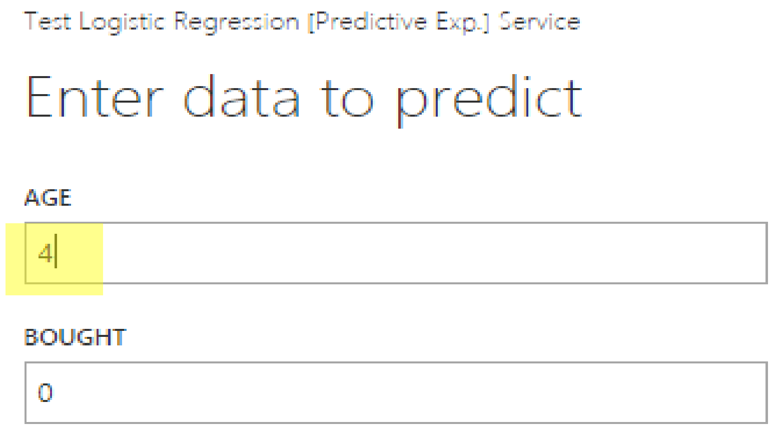
Mean Absolute Error	0.143603
Root Mean Squared Error	0.197182
Relative Absolute Error	0.291549
Relative Squared Error	0.157876
Coefficient of Determination	0.842124

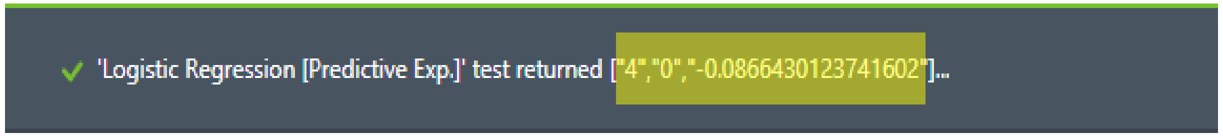
R-squared

Steps - need of logistic regression

Fig6: Enter the value for prediction for Age 4

Fig7: Prediction for Age4





Steps - need of logistic regression

Fig8: Enter the value for prediction for Age105

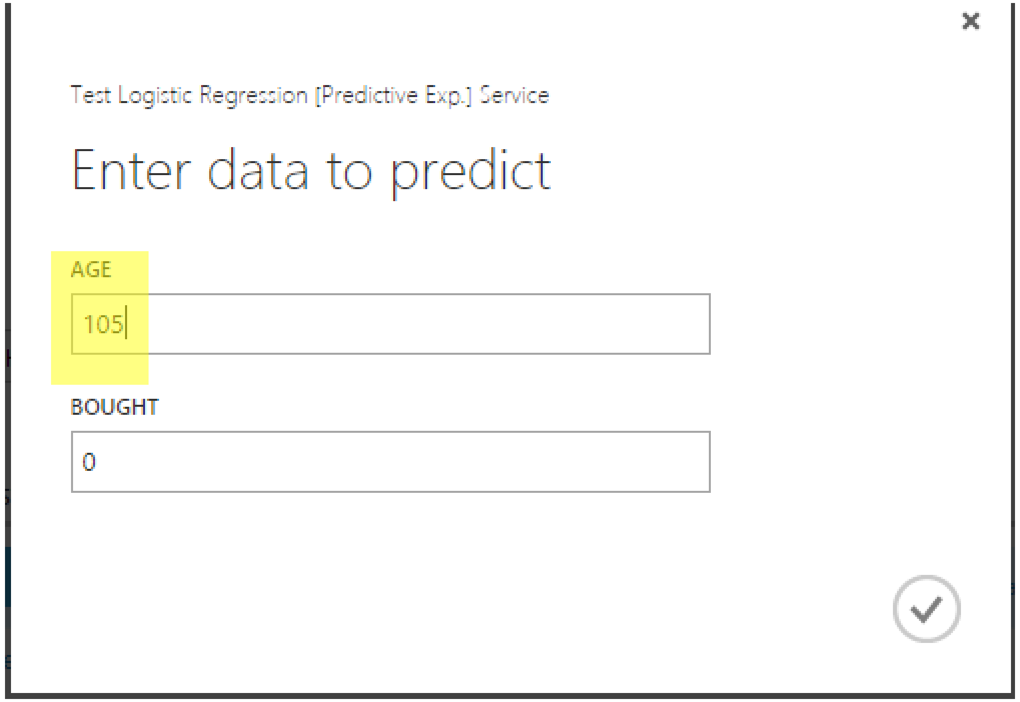
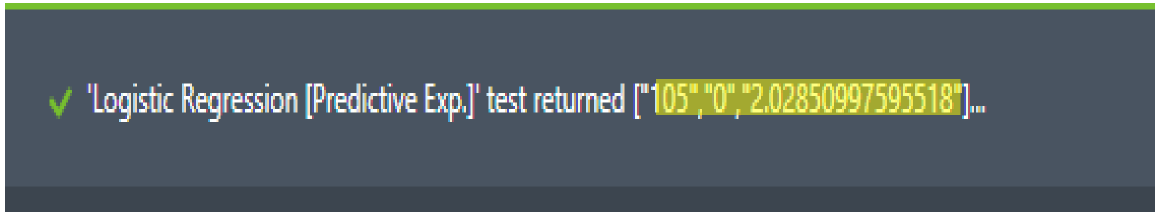


Fig9: Prediction for Age 105

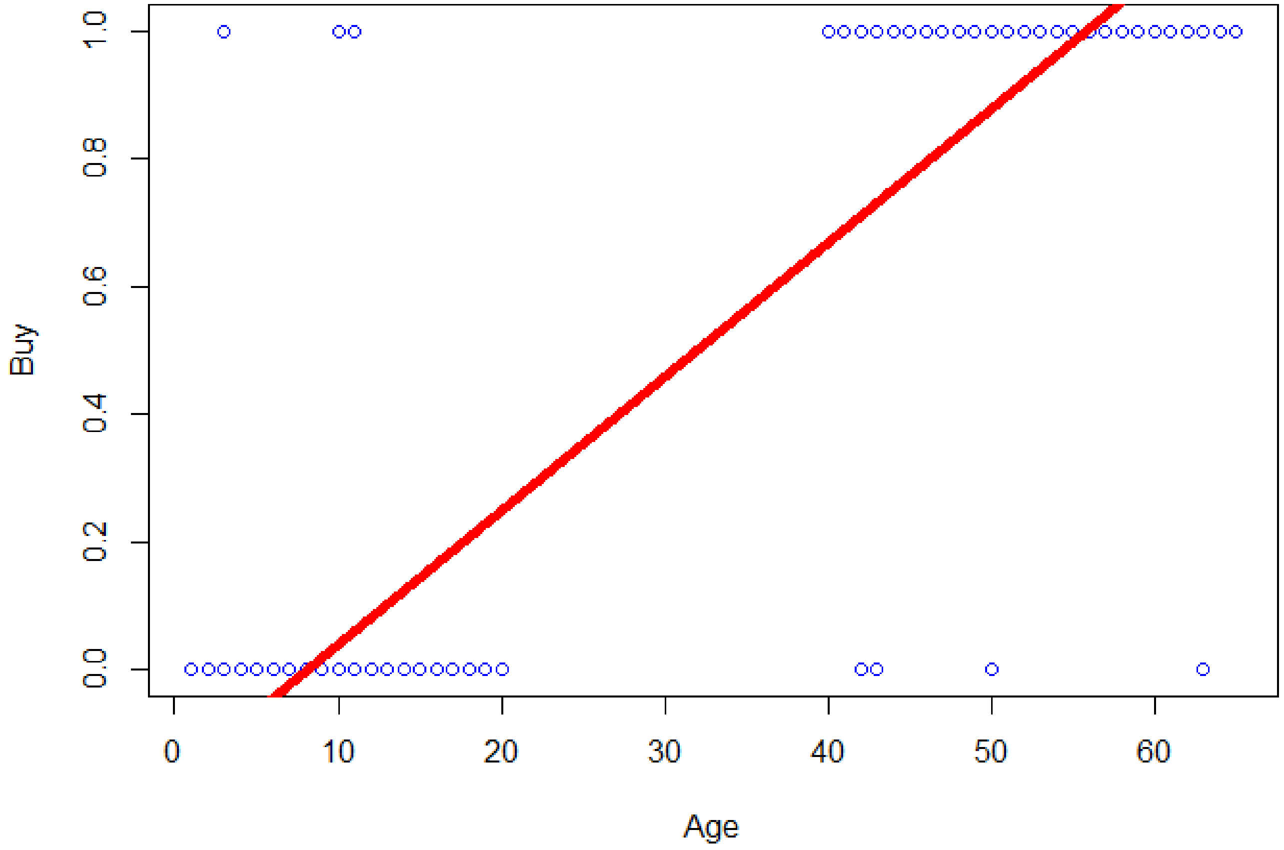


Something wrong

- The model that we built above is not right.
- There is certain issues with the type of dependent variable
- The dependent variable is not continuous it is binary
- We can’t fit a linear regression line to this data

Linear Regression line for above data

- Fig10: Enter the value for prediction for Age105

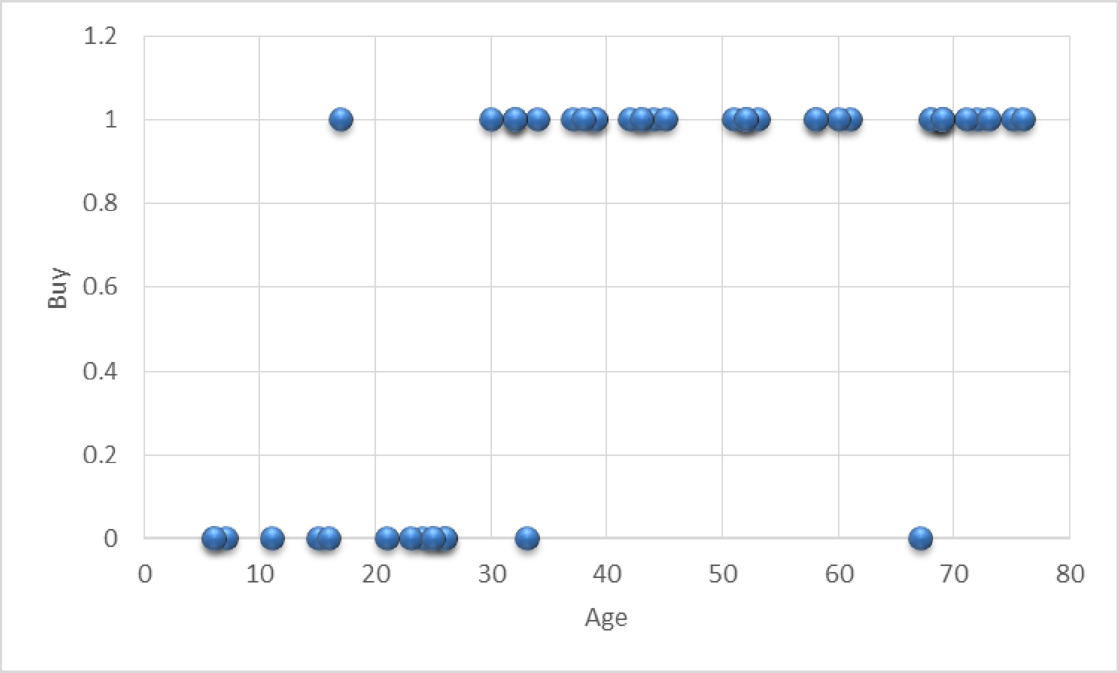


Why not linear ?

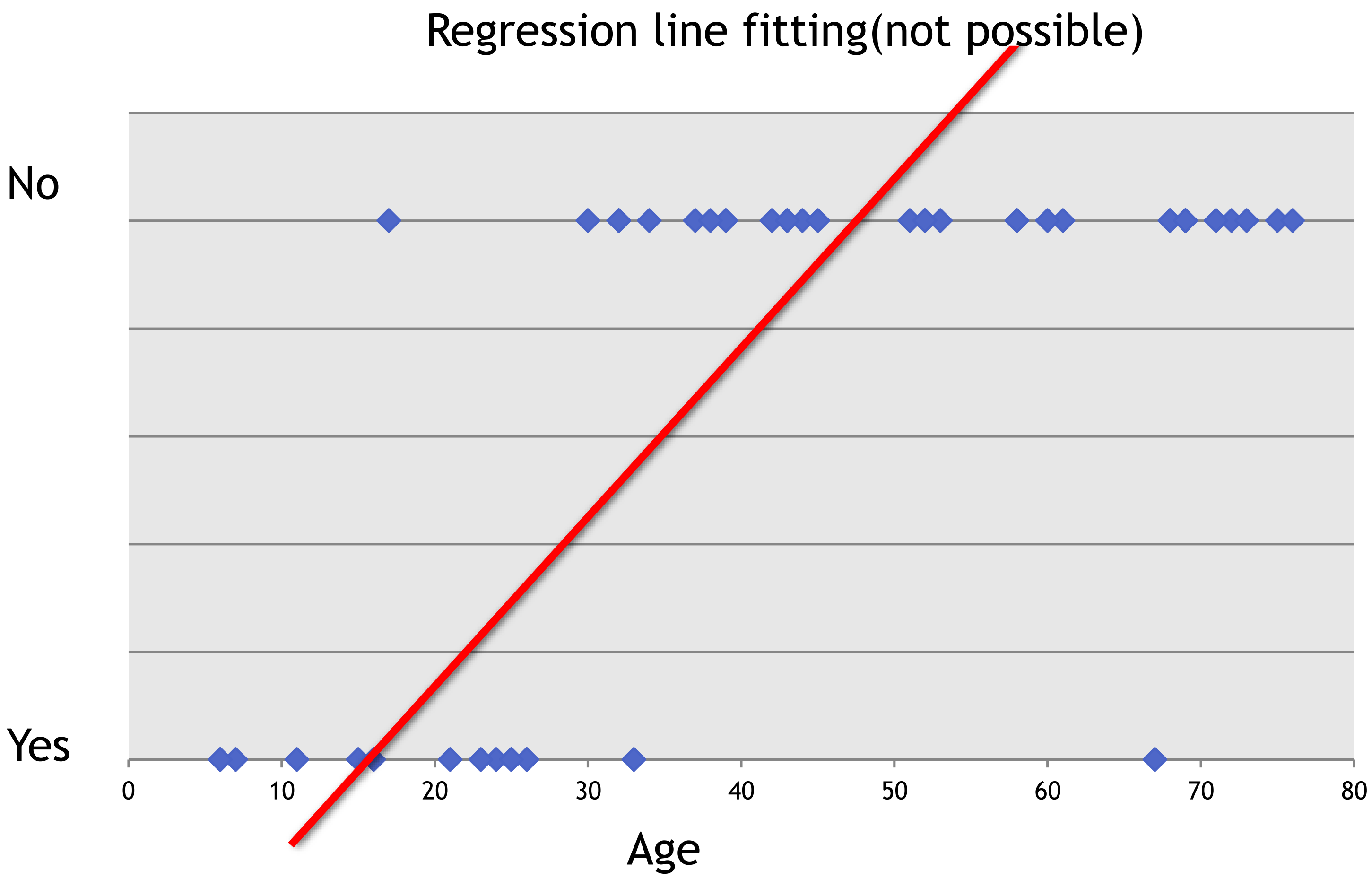
Why not linear ?

- Consider Product sales data. The dataset has two columns.
  - Age – continuous variable between 6-80
  - Buy(0- Yes ; 1-No)

Fig 11: plot between Age Vs Buy



Why not linear ?



Real-life examples

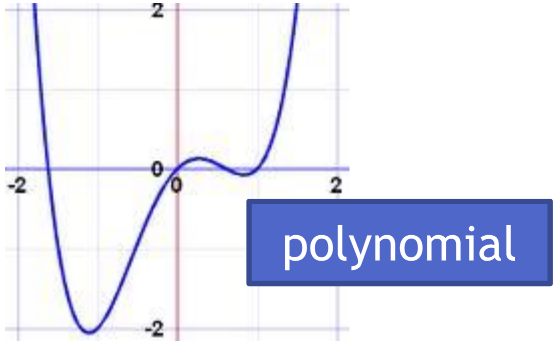
- Gaming - Win vs. Loss
- Sales - Buying vs. Not buying
- Marketing – Response vs. No Response
- Credit card & Loans – Default vs. Non Default
- Operations – Attrition vs. Retention
- Websites – Click vs. No click
- Fraud identification –Fraud vs. Non Fraud
- Healthcare –Cure vs. No Cure

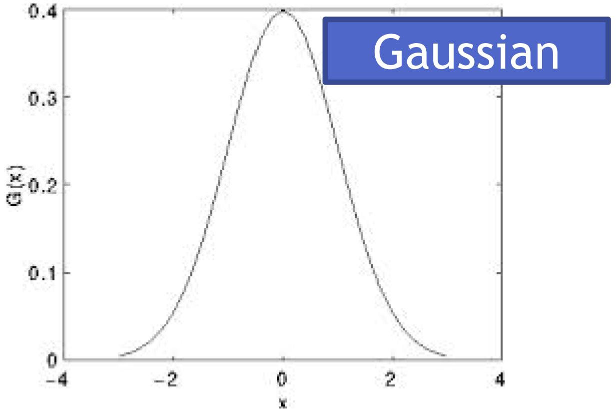
A Logistic Function

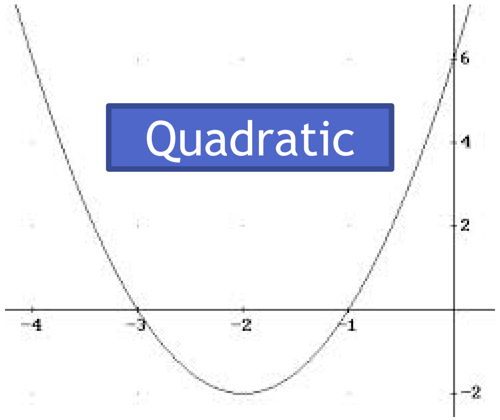
statinfer

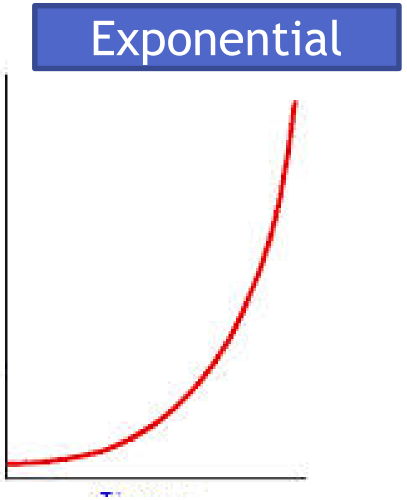
Some Nonlinear functions

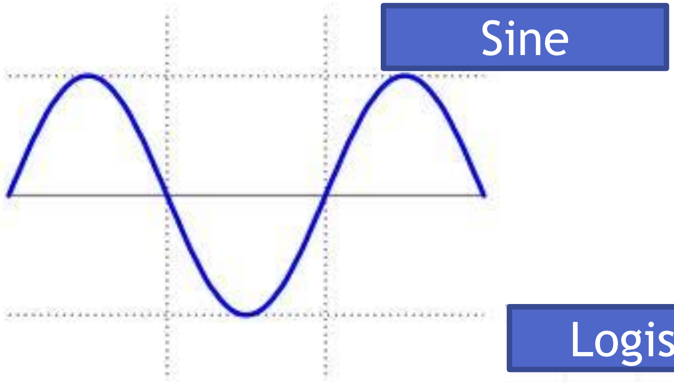
Fig12: Regression line fitting(not possible)

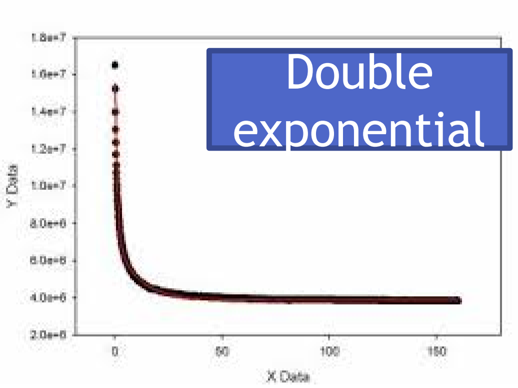


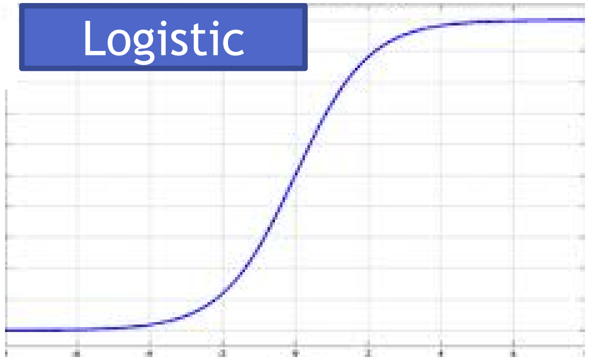




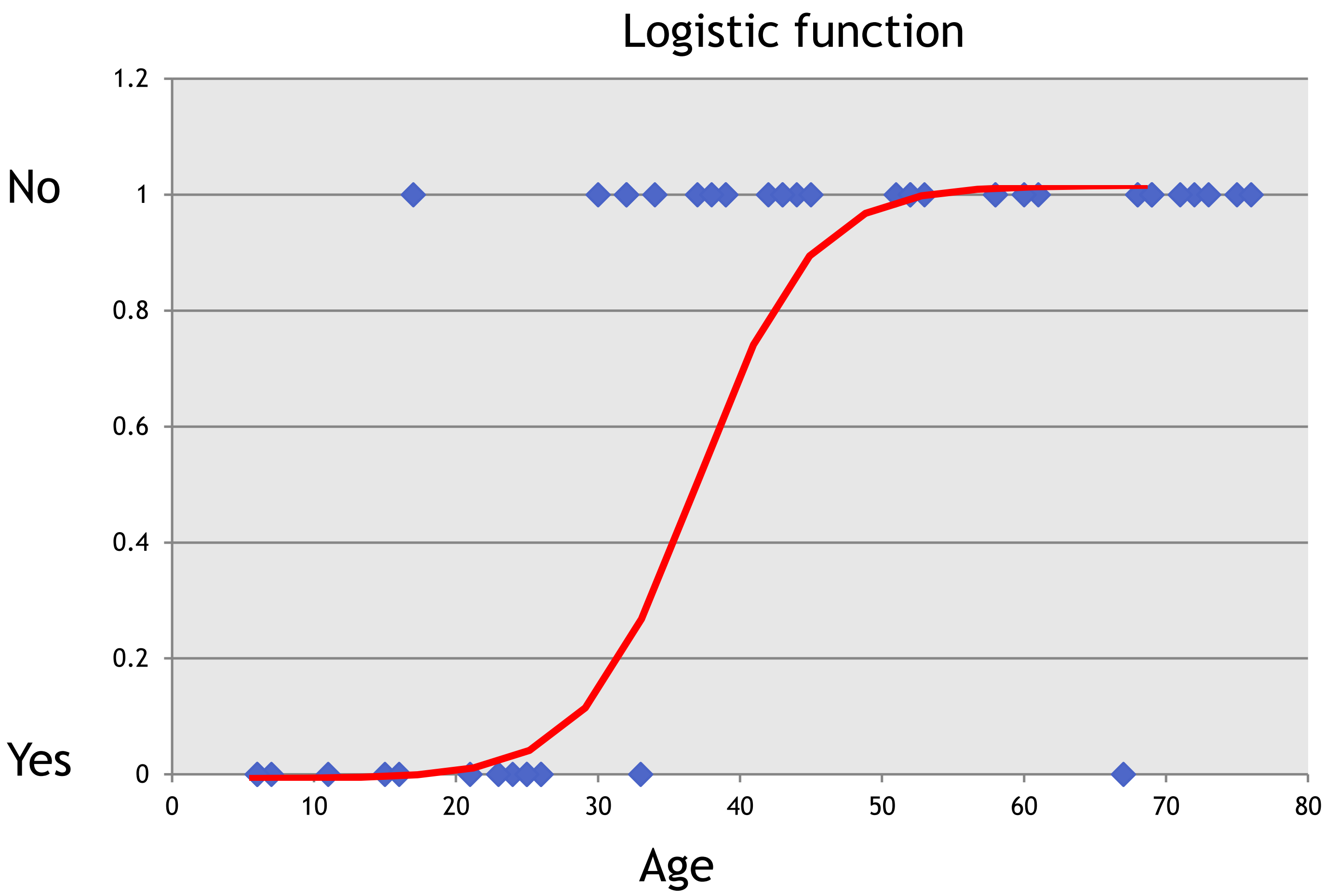








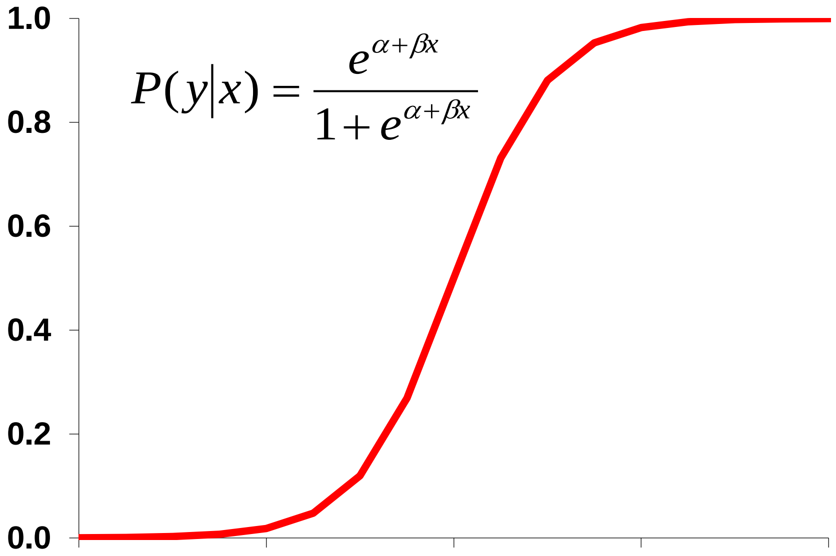
A Logistic Function



The Logistic function

- We want a model that predicts probabilities between 0 and 1, that is, S-shaped.
- There are lots of s-shaped curves. We use the logistic model:
- Probability = $\exp(\beta_0 + \beta_1 X) / [1 + \exp(\beta_0 + \beta_1 X)]$

Logistic Function



Logistic Regression Output

- In logistic regression, we try to predict the probability instead of direct values
- Y is binary, it takes only two values 1 and 0 instead of predicting 1 or 0 we predict the probability of 1 and probability of zero
- This suits aptly for the binary categorical outputs like YES vs NO; WIN vs LOSS; Fraud vs Non Fraud

Logistic Regression Line

Lab: Logistic Regression

- Dataset: Product Sales Data/Product_sales.csv
- Build a logistic Regression line between Age and buying
- A 4 years old customer, will he buy the product?
- If Age is 105 then will that customer buy the product?

Steps – logistic Regression

- Drag and drop the Data set (Product Sales Data/Product_sales.csv)
- Click on output circle and then visualize
- Check out the column names
- And then build a Logistic regression Model for Bought Vs Age
  - Drag-and-drop ‘select column from dataset’ and select both Bought and Age columns
  - Search for ‘Logistic Regression’, drag-and-drop it into the canvas
  - Search for ‘Train Model’, drag-and-drop it into the canvas
  - Connect the output of ‘Logistic Regression’ to left input of the ‘Train Model’ ‘select column from dataset’ to right input of the ‘Train Model’
  - Click on ‘Train Model’, select launch column selector in the properties window
  - Select the column(Bought) for which the prediction to be done
  - Drag-and-drop ‘Score Model’ from left pane and uncheck the ‘Append score column’ in properties window

Steps – logistic Regression

- Connect the output of ‘Train Model’ to left input of the ‘Score Model’ ‘select column from dataset’ to right input of the ‘Score Model’
- Drag-and-drop ‘Evaluate Model’ from left pane
- Connect the output of ‘Score Model’ to the input of ‘Evaluate Model’
- Click on Run 

- After execution click on the output circles of ‘Train Model’, ‘Score Model’ and ‘Evaluate Model’

statinfer

Steps – logistic Regression

- In the experiment click on  in the bottom pane
- Select Retraining Web Service
- Click on  to run in the bottom pane
- After execution again click on  in the bottom pane and select Predictive Web Service
- Again Click on  to run in the bottom pane
- After execution click on  it will deploy and take you to the web service page
- Click on the Test button, Enter data to predict window will open

Steps – Logistic Regression

Fig 13: Logistic Regression

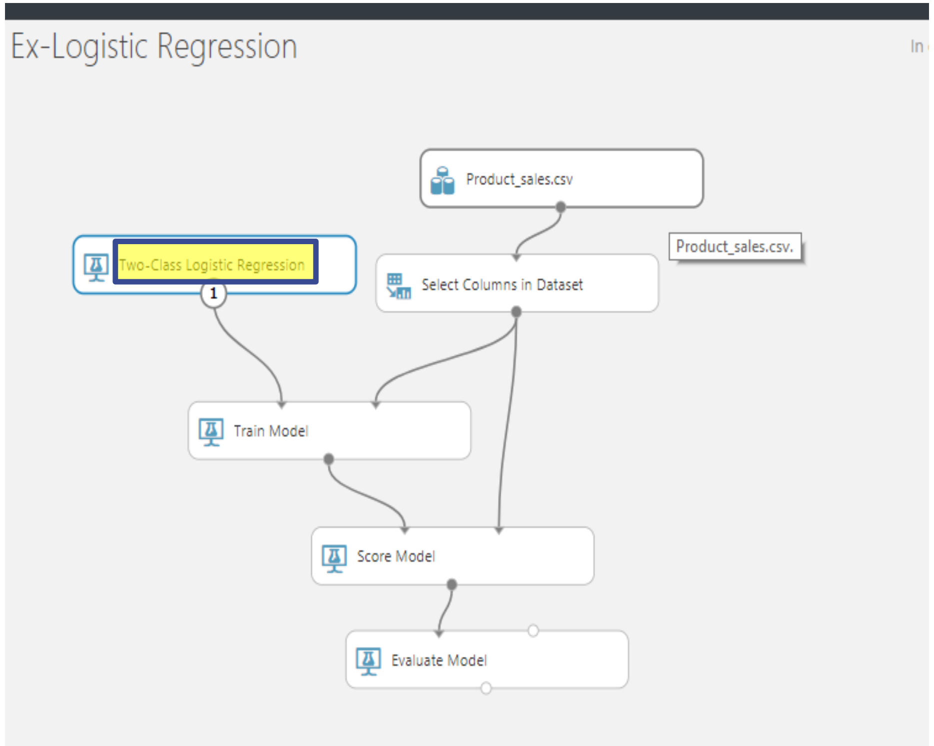
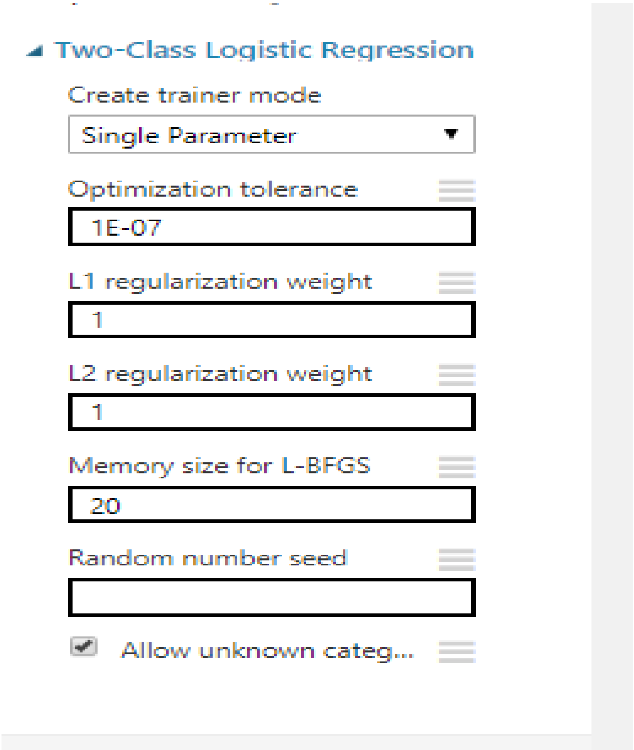


Fig 14: Two class logistic regression parameters



Steps – Logistic Regression

- **Optimization tolerance:** Set a threshold value for optimizing the model
- If the improvement falls below the specified threshold value, then the algorithm meets on a solution and then training stops
- **L1 regularization weight** and **L2 regularization weight**, Give a value to use for the regularization parameters L1 and L2. Here we required non-zero value is recommended for both
- Regularization is method for avoiding over fitting
- **Memory size for L-BFGS:** which indicates number of past and gradients to store for the computation in further steps

Steps – Logistic Regression

Fig 14-1 output of train model

Feature Weights

Feature	Weight
Bias	-0.170411
Age	0.0209421

Fig 14-2 output of evaluate model

Logistic Regression › Evaluate Model › Evaluation results

Metrics

Mean Absolute Error	0.143603
Root Mean Squared Error	0.197182
Relative Absolute Error	0.291549
Relative Squared Error	0.157876
Coefficient of Determination	0.842124

Error Histogram

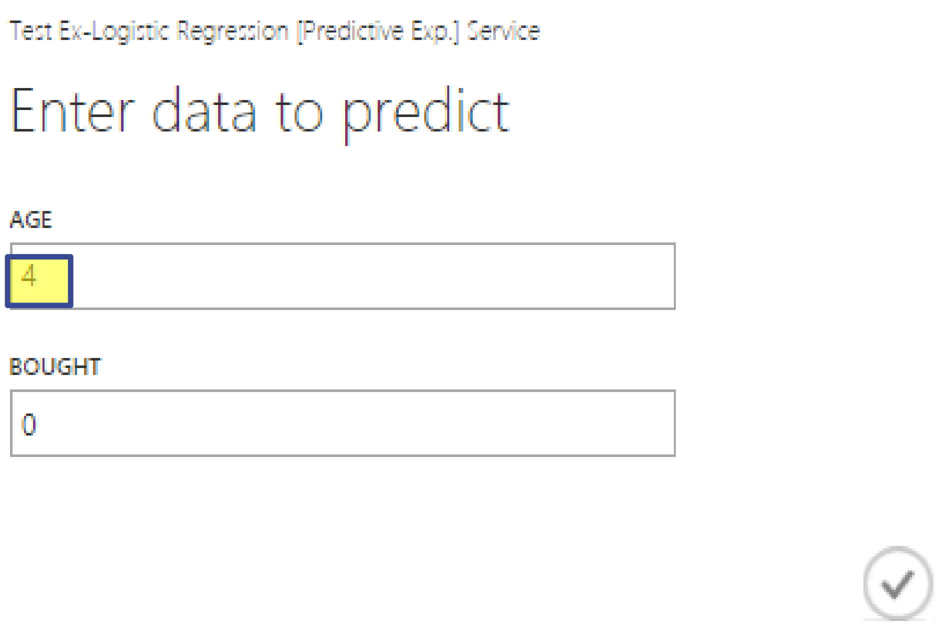
Steps – Logistic Regression

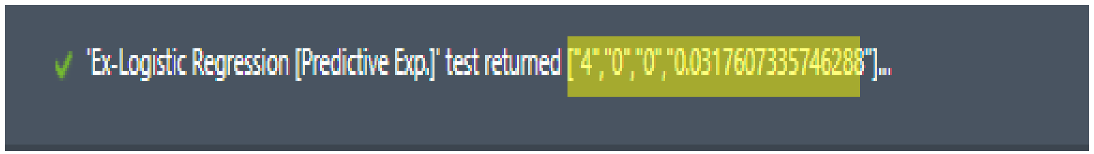
- Create predictive experiment
- Check with Age 4 and Age 105 after deploying web service

Steps – Logistic Regression

Fig 15: enter the value for prediction Age 4

Fig 16: Prediction for Age 4





Steps – Logistic Regression

Fig 17 Enter the value for prediction for Age 105

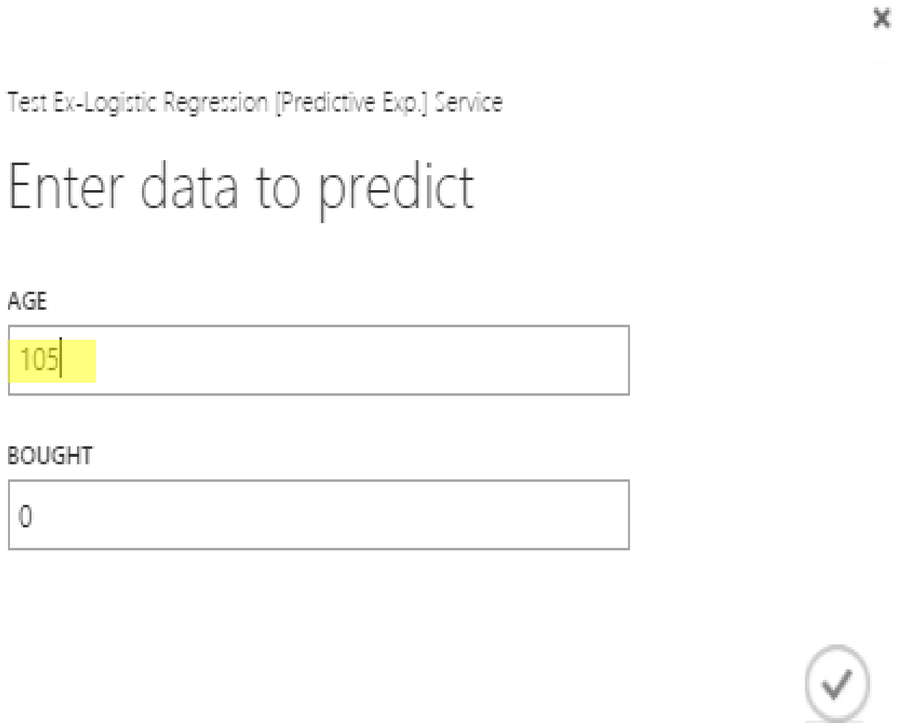
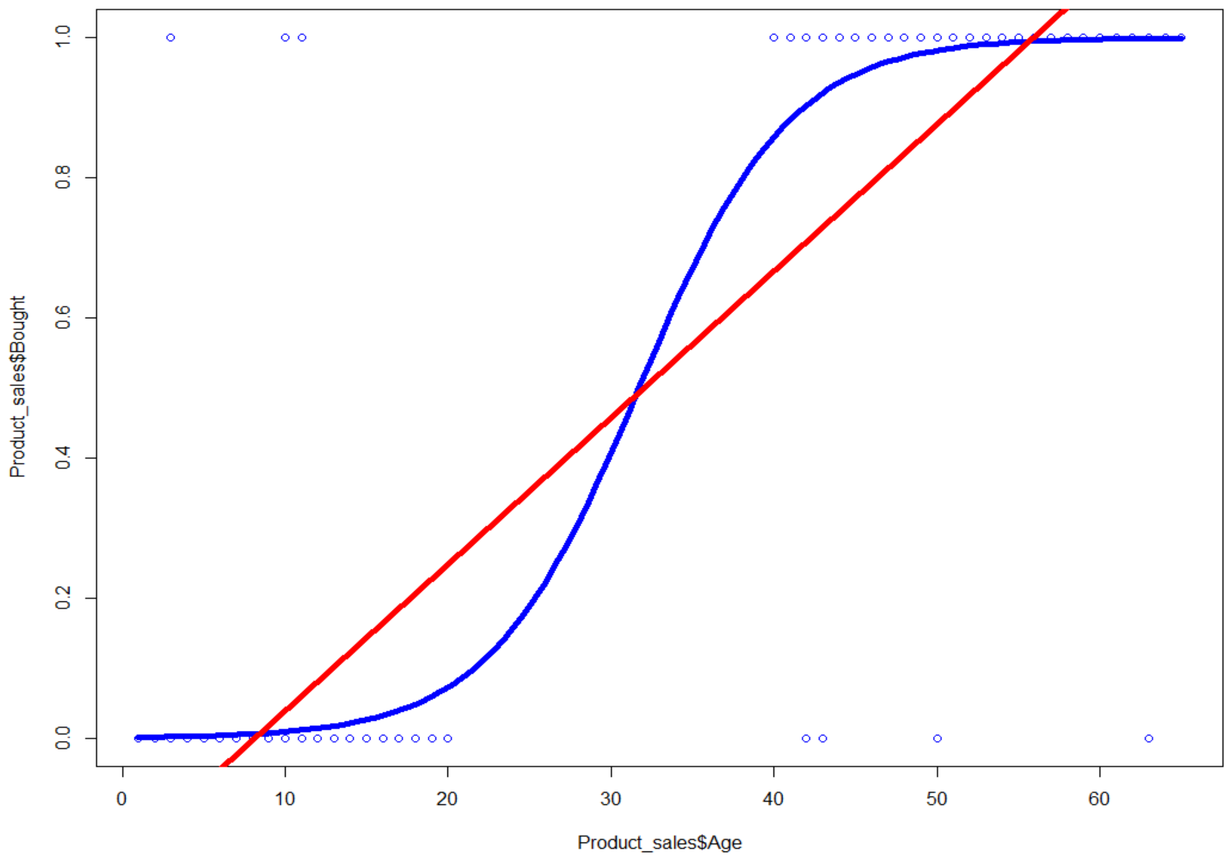


Fig 18 Prediction for Age 105

Steps – Logistic Regression

- Fig 19: Linear Vs logistic regression



Multiple Logistic Regression

Multiple Logistic Regression

statinfer

- The dependent variable is binary
- Instead of single independent/predictor variable, we have multiple predictors
- Like buying / non-buying depends on customer attributes like age, gender, place, income etc.,

LAB: Multiple Logistic Regression

- Dataset: Fiberbits/Fiberbits.csv
  - Active_cust variable indicates whether the customer is active or already left the network.
- Build a model to predict the chance of attrition for a given customer using all the features.
- How good is your model?
- What are the most impacting variables?

Steps – Multiple logistic Regression

- Drag and drop the Data set Fiberbits/Fiberbits.csv)
- Click on output circle and then visualize
- Check out the column names
- And then build a Multiple Logistic regression Model for Bought Vs Age
  - Drag-and-drop 'select column from dataset' and select both Bought and Age columns
  - Search for 'Multiple Logistic Regression', drag-and-drop it into the canvas
  - Search for 'Train Model', drag-and-drop it into the canvas
  - Connect the output of 'Multiple Logistic Regression' to left input of the 'Train Model' 'select column from dataset' to right input of the 'Train Model'
  - Click on 'Train Model', select launch column selector in the properties window
  - Select the column(Bought) for which the prediction to be done
  - Drag-and-drop 'Score Model' from left pane and uncheck the 'Append score column' in properties window

Steps – Multiple logistic Regression

- Connect the output of ‘Train Model’ to left input of the ‘Score Model’ ‘select column from dataset’ to right input of the ‘Score Model’
- Drag-and-drop ‘Evaluate Model’ from left pane
- Connect the output of ‘Score Model’ to the input of ‘Evaluate Model’
- Click on Run 

- After execution click on the output circles of ‘Train Model’, ‘Score Model’ and ‘Evaluate Model’ to see the value of R-squared
- Now we need to predict if the Age is 4 will that customer will purchase the product or not

Steps – Multiple logistic Regression

- In the experiment click on 

 in the bottom pane
- Select Retraining Web Service
- Click on 

 to run in the bottom pane
- After execution again click on 

 in the bottom pane and select Predictive Web Service
- Again Click on 

 to run in the bottom pane
- After execution click on 

 it will deploy and take you to the web service page
- Click on the Test button, Enter data to predict window will open

Steps – Multiple logistic Regression

statinfer

Feature Weights

Feature	0	1
Speed_test_result	-24.1798	24.1798
months_on_network	-2.86059	2.86063
income	-2.15482	2.15482
relocated	1.51668	-1.5167
technical_issues_per_month	1.19101	-1.19106
Num_complaints	0.999854	-0.999871
number_plan_changes	0.863475	-0.8635
monthly_bill	0.156779	-0.156793
Bias	0.00723847	-0.00683625

Fig 20: Intercept values

Goodness of fit for a logistic regression

Goodness of fit for a logistic regression

- Classification Matrix
- AIC and BIC
- ROC & AUC – Area under the curve

Classification Table & Accuracy

		Predicted	
		0	1
Actual	0	True positive (TP)	False Negatives(FN)
	1	False positive (FP)	True Negatives(TN)

- Also known as confusion matrix
- Accuracy=(TP+TN)/(TP+FN+FP+TN)

Classification Table

Sensitivity and Specificity are derived from confusion matrix

Actual Classes \ Predicted Classes	0(Positive)	1(Negative)
0(Positive)	True positive (TP) Actual condition is Positive, it is truly predicted as positive	False Negatives(FN) Actual condition is Positive, it is falsely predicted as negative
1(Negative)	False Positives(FP) Actual condition is Negative, it is falsely predicted as positive	True Negatives(TN) Actual condition is Negative, it is truly predicted as negative

- Accuracy=(TP+TN)/(TP+FP+FN+TN)
- Misclassification Rate=(FP+FN)/(TP+FP+FN+TN)

LAB: Confusion Matrix & Accuracy

- Create confusion matrix for product sales model and find the accuracy
- Create confusion matrix for Fiber bits model
- Find the accuracy value for fiber bits model
- Change try three different threshold values and note down the changes in accuracy value

Steps: Confusion Matrix & Accuracy

- Drag and drop the Data set (Product Sales Data/Product_sales.csv)
- Click on output circle and then visualize
- Check out the column names
- And then build a Logistic regression Model for Bought Vs Age
  - Drag-and-drop 'select column from dataset' and select both Bought and Age columns
  - Search for 'Logistic Regression', drag-and-drop it into the canvas
  - Search for 'Train Model', drag-and-drop it into the canvas
  - Connect the output of 'Logistic Regression' to left input of the 'Train Model' 'select column from dataset' to right input of the 'Train Model'
  - Click on 'Train Model', select launch column selector in the properties window
  - Select the column(Bought) for which the prediction to be done
  - Drag-and-drop 'Score Model' from left pane and uncheck the 'Append score column' in properties window

Steps – logistic Regression

- Connect the output of ‘Train Model’ to left input of the ‘Score Model’ ‘select column from dataset’ to right input of the ‘Score Model’
- Drag-and-drop ‘Evaluate Model’ from left pane
- Connect the output of ‘Score Model’ to the input of ‘Evaluate Model’
- Click on Run 

- After execution click on the output circles of ‘Train Model’, ‘Score Model’ and ‘Evaluate Model’
- Similarly execute same steps for fiberbits model and change threshold values

Steps: Confusion Matrix & Accuracy

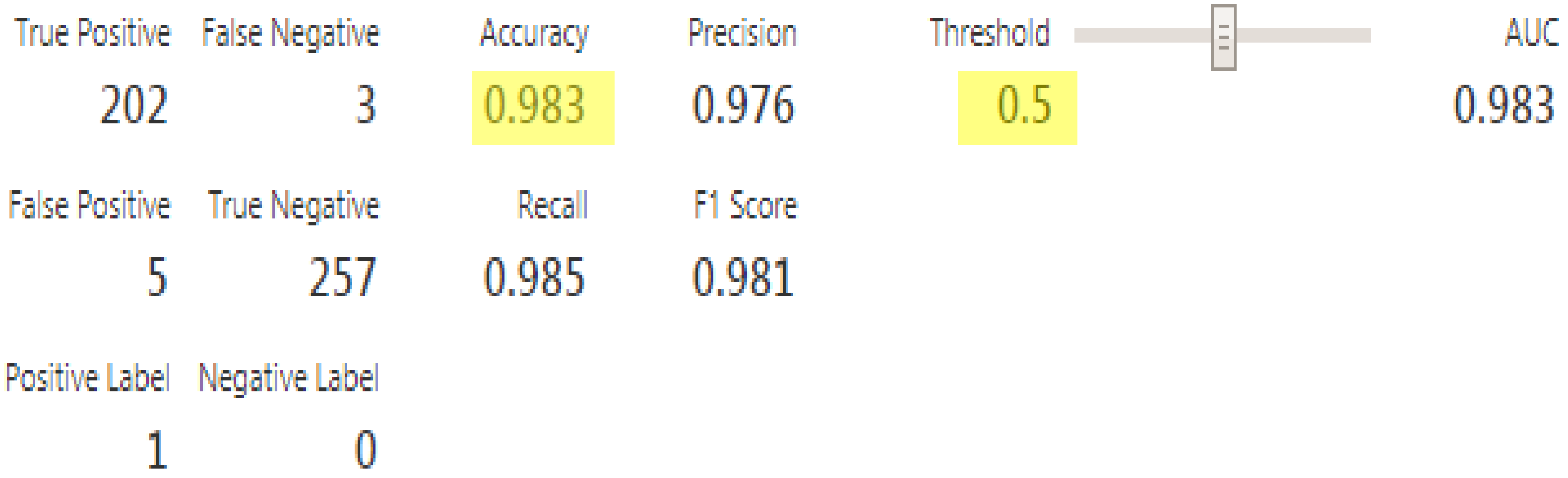


Fig 21: Accuracy and Confusion Matrix

Steps: Confusion Matrix & Accuracy

Fig 22: Accuracy and Confusion matrix when threshold is 0.5

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
48268	9591	0.770	0.782	0.5	0.830
False Positive	**True Negative**	**Recall**	**F1 Score**		
13458	28683	0.834	0.807		
Positive Label	**Negative Label**				
1	0				

Fig 23: Accuracy and Confusion matrix when threshold is 0.7

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
34872	22987	0.711	0.856	0.7	0.830
False Positive	**True Negative**	**Recall**	**F1 Score**		
5888	36253	0.603	0.707		
Positive Label	**Negative Label**				
1	0				

Steps: Confusion Matrix & Accuracy

Fig 24: Accuracy and Confusion matrix when threshold is 0.3

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
56849	1010	0.694	0.658	0.3	0.830
False Positive	True Negative	Recall	F1 Score		
29583	12558	0.983	0.788		
Positive Label	Negative Label				
1	0				

statinfer

Multicollinearity

Multicollinearity

- The relation is between X and Y is non linear, we used logistic regression
- The multicollinearity is an issue related to predictor variables.
- Multicollinearity need to be fixed in logistic regression as well.
- Otherwise the individual coefficients of the predictors will be effected by the interdependency
- The process of identification is same as linear regression

LAB-Multicollinearity

- Is there any multicollinearity in fiber bits model?
- Identify and remove multicollinearity from the model

Steps-Multicollinearity

- Here to find out multicollinearity we take ‘execute R-script’ drag and drop ‘execute R- Script’

R Script

```r
# Map 1-based optional input ports to variables
library(car)
dataset1 <- maml.mapInputPort(1) # class: data.frame
model1<-glm(dataset1$active_cust~dataset1$income
                              +dataset1$months_on_network
                              +dataset1$Num_complaints
                              +dataset1$number_plan_changes
                              +dataset1$relocated
                              +dataset1$monthly_bill
                              +dataset1$technical_issues_per_month
                              +dataset1$Speed_test_result,family=binomial())

vif(model1)
maml.mapOutputPort("dataset1");
```

Fig 25: R-script

Steps - Multicollinearity

Fig 26: Multicollinearity

Multiple logistic regression > Execute R Script > R Device

Standard Output

```
RWorker pushed "port1" to R workspace.
Beginning R Execute Script

[1] 56000
Loading objects:
  port1
[1] "Loading variable port1..."
                   dataset1$income          dataset1$months_on_network
                          4.590705                            4.641040
           dataset1$Num_complaints        dataset1$number_plan_changes
                          1.018607                            1.126892
                dataset1$relocated              dataset1$monthly_bill
                          1.145847                            1.017565
dataset1$technical_issues_per_month          dataset1$Speed_test_result
                          1.020648                            1.206999
[1] "Saving variable  dataset1  ..."
[1] "Saving the following item(s):  .maml.oport1"
```

Standard Error

```
R reported no errors.
```

Individual Impact of Variables

Individual Impact of Variables

- Out of these predictor variables, what are the important variables?
- If we have to choose the top 5 variables what are they?
- While selecting the model, we may want to drop few less impacting variables.
- How to rank the predictor variables in the order of their importance?

Individual Impact - z values & Wald chi-square

- We can simply look at the z values of the each variable. Look at their absolute values
- Or calculate the Wald chi-square, which is nearly equal to square of the z-score
- Wald Chi-Square value helps in ranking the variables

LAB: Individual Impact of Variables

- Identify top impacting and least impacting variables in fiber bits models
- Find the variable importance and order them based on their impact

Steps - Individual Impact of Variables

Fig 27: Individual impact of variables Code

R Script

```
# Map 1-based optional input ports to variables
library(car)
dataset1 <- maml.mapInputPort(1) # class: data.frame
model1<-glm(dataset1$active_cust~dataset1$income
                      +dataset1$months_on_network
                      +dataset1$Num_complaints
                      +dataset1$number_plan_changes
                      +dataset1$relocated
                      +dataset1$monthly_bill
                      +dataset1$technical_issues_per_month
                      +dataset1$Speed_test_result,family=binomial())
library(caret)
varImp(model1, scale = FALSE)
summary(model1)
vif(model1)
maml.mapOutputPort("dataset1");
```

Fig 28: Individual impact of variables

	Overall
dataset1$income	20.81981
dataset1$months_on_network	28.65421
dataset1$Num_complaints	22.81102
dataset1$number_plan_changes	24.93955
dataset1$relocated	79.92677
dataset1$monthly_bill	13.99490
dataset1$technical_issues_per_month	54.58123
dataset1$Speed_test_result	93.43471

Model Selection

How to improve model

- By adding more independent variables?
- By deriving new variables from available set?
- By transforming variables ?
- By collecting more data?
- How do we choose best model from the list of fitted models with different parameters

AIC and BIC

- AIC and BIC values are like adjusted R-squared values in linear regression
- Stand-alone model AIC has no real use, but if we are choosing between the models AIC really helps.
- Given a collection of models for the data, AIC estimates the quality of each model, relative to each of the other models
- If we are choosing between two models, a model with less AIC is preferred
- AIC is an estimate of the information lost when a given model is used to represent the process that generates the data

AIC and BIC

- AIC= -2ln(L)+ 2k
  - L be the maximum value of the likelihood function for the model
  - k is the number of independent variables
- BIC is a substitute to AIC with a slightly different formula. We will follow either AIC or BIC throughout our analysis

LAB-Logistic Regression Model Selection

- Find AIC and BIC values for the first fiber bits model(m1)
- What are the top-2 impacting variables in fiber bits model?
- What are the least impacting variables in fiber bits model?
- Can we drop any of these variables and build a new model(m2)
- Can we add any new interaction and polynomial variables to increase the accuracy of the model?(m3)
- We have three models, what the best accuracy that you can expect on this data?

Steps-Logistic Regression Model Selection

- Import ‘execute R-script’ tile
- Connect it to dataset
- Write the code for R-script (Fig no: 29)
- Click on Run
- Once finished click on second output circle to visualize

Steps-Logistic Regression Model Selection

Fig 29: Impact of variables code

Fig 30: Impact of variables

Execute R Script

R Script

```r
# Map 1-based optional input ports to variables
library(car)
dataset1 <- maml.mapInputPort(1) # class: data.frame
model1<-glm(dataset1$active_cust~dataset1$income
                       +dataset1$months_on_network
                       +dataset1$Num_complaints
                       +dataset1$number_plan_changes
                       +dataset1$relocated
                       +dataset1$monthly_bill
                       +dataset1$technical_issues_per_month
                       +dataset1$Speed_test_result,family=binomial())
library(caret)
varImp(model1, scale = FALSE)
#summary(model1)
#vif(model1)

library(stats)
AIC(model1)
BIC(model1)

#summary(model1)
```

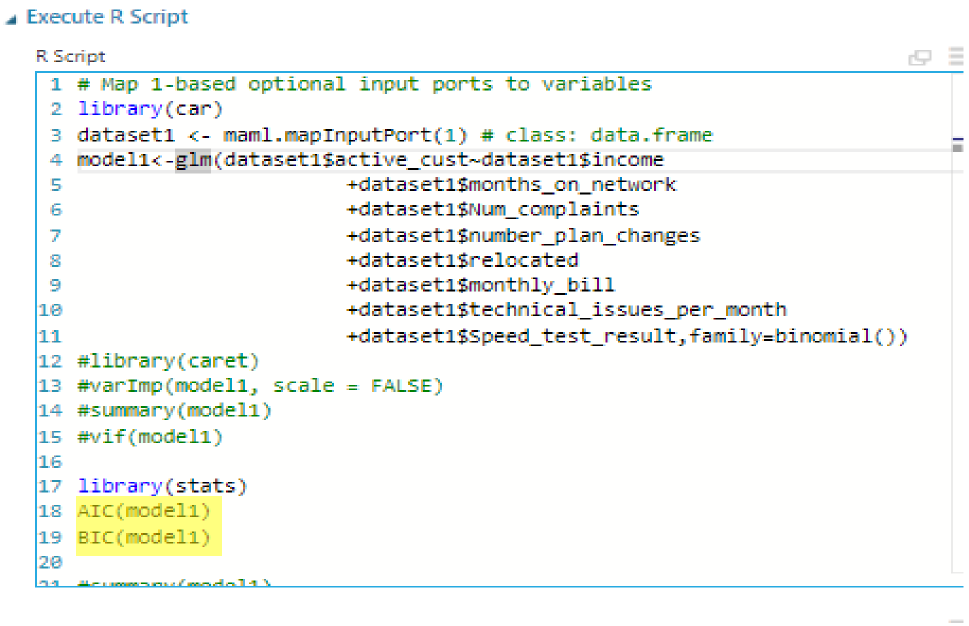
Multiple logistic regression › Execute R Script › R Device

Standard Output

```
RWorker pushed "port1" to R workspace.
Beginning R Execute Script

[1] 56000
Loading objects:
  port1
[1] "Loading variable port1..."
                                    Overall
dataset1$income                     20.81981
dataset1$months_on_network          28.65421
dataset1$Num_complaints             22.81102
dataset1$number_plan_changes        24.93955
dataset1$relocated                  79.92677
dataset1$monthly_bill               13.99490
dataset1$technical_issues_per_month 54.58123
dataset1$Speed_test_result          93.43471
[1] 98377.36
[1] 98462.97
[1] "Saving variable  dataset1  ..."
[1] "Saving the following item(s):  .maml.oport1"
```

Steps-Logistic Regression Model Selection

Fig 31: AIC and BIC Code

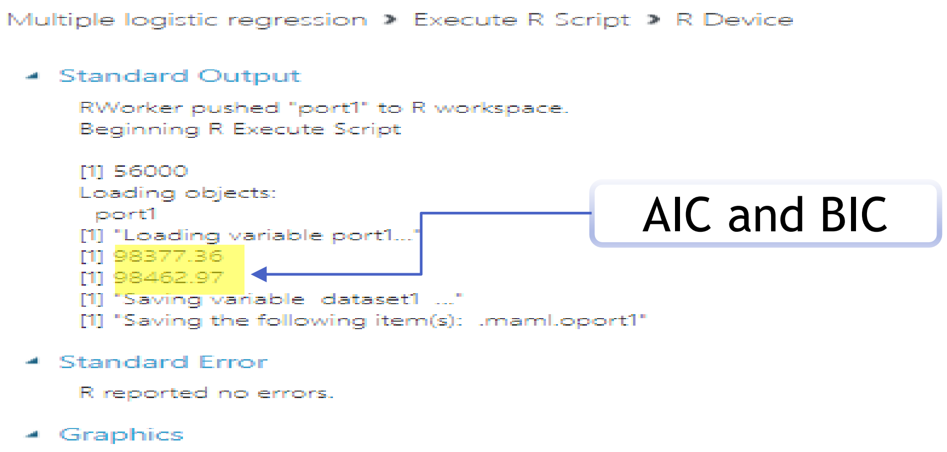


Execute R Script

R Script

```r
# Map 1-based optional input ports to variables
library(car)
dataset1 <- maml.mapInputPort(1) # class: data.frame
model1<-glm(dataset1$active_cust~dataset1$income
                         +dataset1$months_on_network
                         +dataset1$Num_complaints
                         +dataset1$number_plan_changes
                         +dataset1$relocated
                         +dataset1$monthly_bill
                         +dataset1$technical_issues_per_month
                         +dataset1$Speed_test_result,family=binomial())
#library(caret)
#varImp(model1, scale = FALSE)
#summary(model1)
#vif(model1)

library(stats)
AIC(model1)
BIC(model1)

#summary(model1)
```

Fig 32: AIC and BIC values



Steps-Logistic Regression Model Selection

Fig 33 AIC and BIC (impact of variable code)

R Script

```r
# Map 1-based optional input ports to variables
library(car)
dataset1 <- maml.mapInputPort(1) # class: data.frame
model1<-glm(dataset1$active_cust~
                          dataset1$months_on_network
                          +dataset1$Num_complaints
                          +dataset1$number_plan_changes
                          +dataset1$relocated
                          +dataset1$monthly_bill
                          +dataset1$technical_issues_per_month
                          +dataset1$Speed_test_result,family=binomial())
library(caret)
varImp(model1, scale = FALSE)
#summary(model1)
#vif(model1)

library(stats)
AIC(model1)
BIC(model1)

#summary(model1)
```

Fig 34: AIC and BIC (with impact of variable)

Multiple logistic regression › Execute R Script › R Device

Standard Output

```
RWorker pushed "port1" to R workspace.
Beginning R Execute Script

[1] 56000
Loading objects:
  port1
[1] "Loading variable port1..."
                                         Overall
dataset1$months_on_network           21.62375
dataset1$Num_complaints              23.65057
dataset1$number_plan_changes         26.62771
dataset1$relocated                   79.65556
dataset1$monthly_bill                14.38908
dataset1$technical_issues_per_month 55.44575
dataset1$Speed_test_result           94.15623
[1] 99076.27
[1] 99152.37
[1] "Saving variable  dataset1  ..."
[1] "Saving the following item(s):  .maml.oport1"
```

Here we discarded less impact variable (income)

Steps-Logistic Regression Model Selection

Fig 35: AIC and BIC code

R Script

```
# map 1-based optional input ports to variables
library(car)
dataset1 <- maml.mapInputPort(1) # class: data.frame
model1<-glm(dataset1$active_cust~dataset1$income
            +dataset1$months_on_network
            +dataset1$Num_complaints
            +dataset1$number_plan_changes
            +dataset1$relocated
            +dataset1$monthly_bill
            +(dataset1$technical_issues_per_month*dataset1$number_plan_changes)
            +dataset1$technical_issues_per_month
                      +(dataset1$Speed_test_result^2)
            +dataset1$Speed_test_result,family=binomial())
library(caret)
varImp(model1, scale = FALSE)
#summary(model1)
#vif(model1)

library(stats)
AIC(model1)
BIC(model1)
```

Fig 36: AIC and BIC impact of variable

Multiple logistic regression › Execute R Script › R Device

Standard Output

```
RWorker pushed "port1" to R workspace.
Beginning R Execute Script

[1] 56000
Loading objects:
  port1
[1] "Loading variable port1..."
                                                         Overall
dataset1$income                                          20.81519
dataset1$months_on_network                               29.04079
dataset1$Num_complaints                                  22.83986
dataset1$number_plan_changes                             21.27366
dataset1$relocated                                       80.37997
dataset1$monthly_bill                                    13.97731
dataset1$technical_issues_per_month                      49.24918
dataset1$Speed_test_result                               91.27237
dataset1$number_plan_changes:dataset1$technical_issues_per_month 13.71783
[1] 98226.04
[1] 98321.17
[1] "Saving variable  dataset1  ..."
[1] "Saving the following item(s):  .maml.oport1"
```

statinfer

Conclusion: Logistic Regression

Conclusion: Logistic Regression

- Logistic Regression is a good foundation for all classification algorithms
- A good understanding on logistic regression and goodness of fit measures will really help in understanding complex machine learning algorithms like neural networks and SVMs
- One has to be careful while selecting the model, all the goodness of fit measures are calculated on training data. We may have to do cross validation to get an idea on the test error

statinfer

Thank you

statinfer

Part 7/12 - Decision Trees with Azure

Venkat Reddy

statinfer

Introduction

Contents

- What is segmentation
- What is a Decision tree
- Decision Trees Algorithm
- Building decision Trees
- Tree validation
- Pruning
- Prediction using the model

statinfer

What is Segmentation?

What is Segmentation?

- Imagine a scenario where we want to run a SMS marketing campaign to attract more customers in the next quarter
  - Some customers like to see high discount
  - Some customers want to see a large collection of items
  - Some customers are fans of particular brands
  - Some customers are Male some are Female
- Divide them based on their demographics, buying patterns and profile related attributes

What is Segmentation?

- One size doesn't fit all
- Divide the population in such a way that
  - Customers inside a group are homogeneous
  - Customers across groups are heterogeneous
- Is there any statistical way of dividing them correctly based on the data

Segmentation Business Problem

The Business Problem

Old Data		
Gender	Marital Status	Ordered the product
M	Married	No
F	Unmarried	Yes
M	Married	No
M	Married	No
M	Married	No
M	Married	No
F	Unmarried	Yes
M	Unmarried	Yes
F	Married	No
M	Married	No
F	Married	No
M	Unmarried	No
F	Married	No
F	Unmarried	Yes

New Data		
Gender	Marital Status	Product order
M	Married	??
F	Unmarried	??

The Business Problem

Old Data

Sr No	Gender	Marital Status	Ordered the product
1	M	Married	No
2	F	Unmarried	Yes
3	M	Married	No
4	M	Married	No
5	M	Married	No
6	M	Married	No
7	F	Unmarried	Yes
8	M	Unmarried	Yes
9	F	Married	No
10	M	Married	No
11	F	Married	No
12	M	Unmarried	No
13	F	Married	No
14	F	Unmarried	Yes

New Data

Gender	Marital Status	Product order
M	Married	??
F	Unmarried	??

The Decision Tree Philosophy

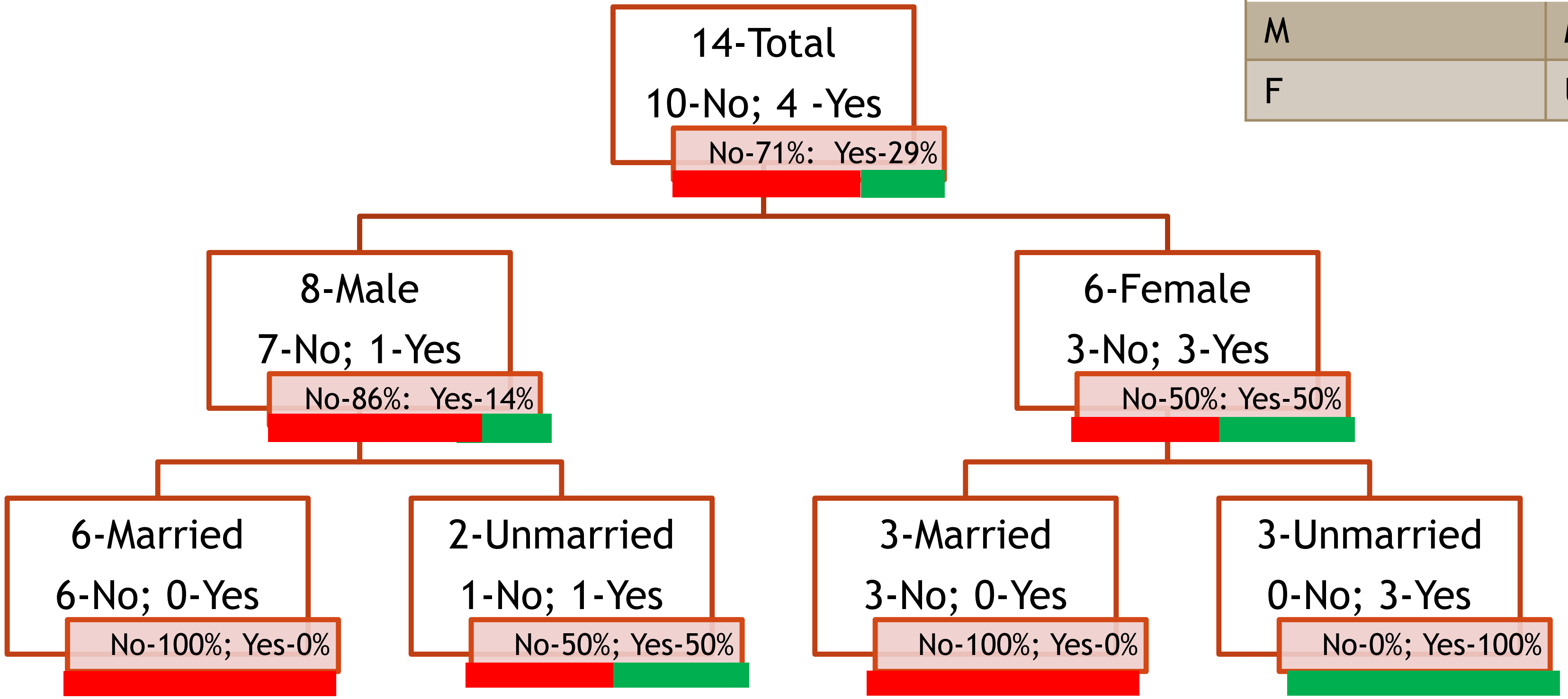
The Data

statinfer

Old Data			
Sr No	Gender	Marital Status	Ordered the product
1	M	Married	No
2	F	Unmarried	Yes
3	M	Married	No
4	M	Married	No
5	M	Married	No
6	M	Married	No
7	F	Unmarried	Yes
8	M	Unmarried	Yes
9	F	Married	No
10	M	Married	No
11	F	Married	No
12	M	Unmarried	No
13	F	Married	No
14	F	Unmarried	Yes

New Data		
Gender	Marital Status	Product order
M	Married	??
F	Unmarried	??

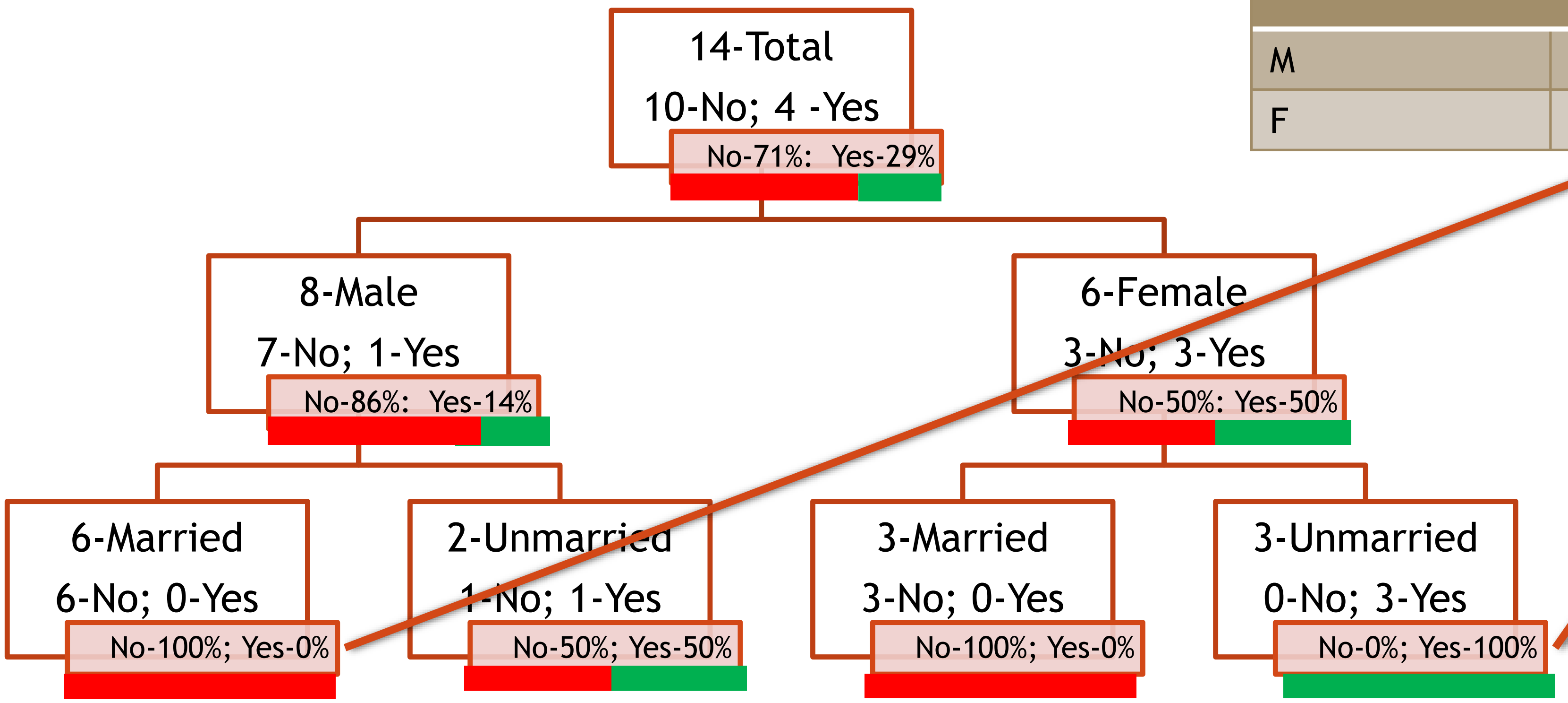
Re-Arranging the data



New Data

Gender	Marital Status	Product order
M	Married	??
F	Unmarried	??

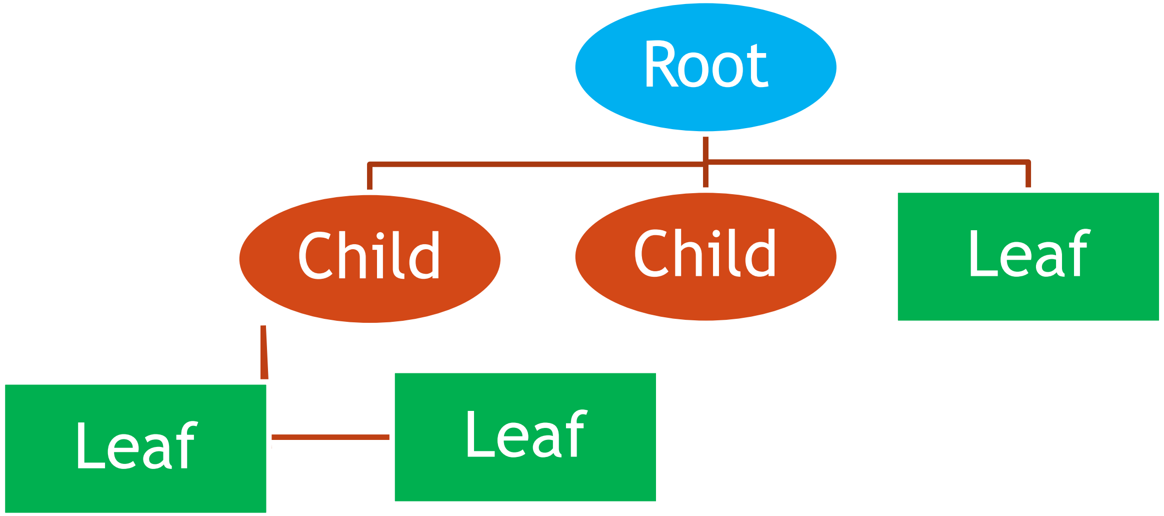
Re-Arranging the data



New Data

Gender	Marital Status	Product order
M	Married	??
F	Unmarried	??

The Decision Tree Vocabulary

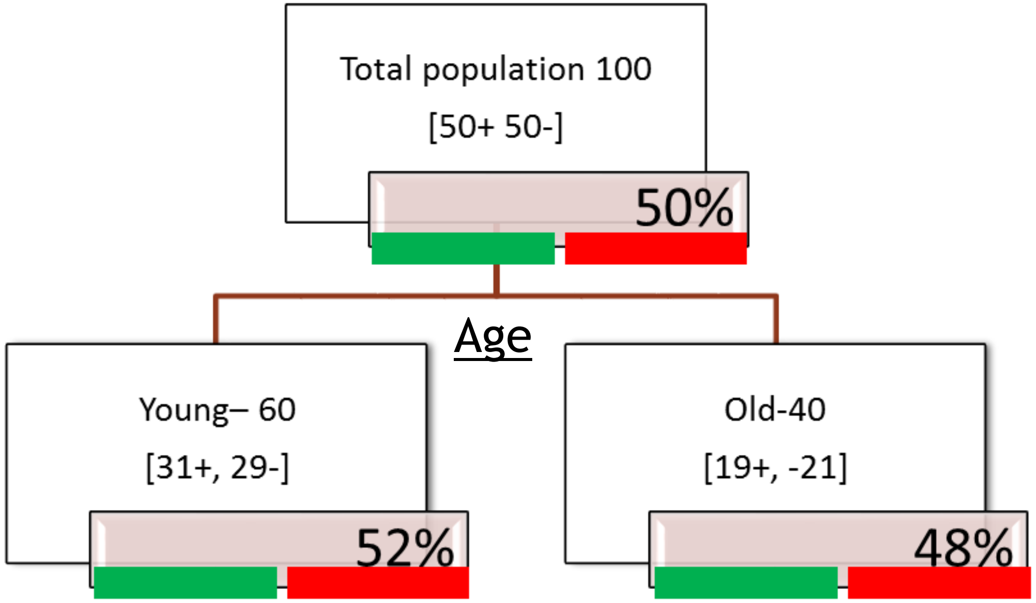


The Decision Tree Approach

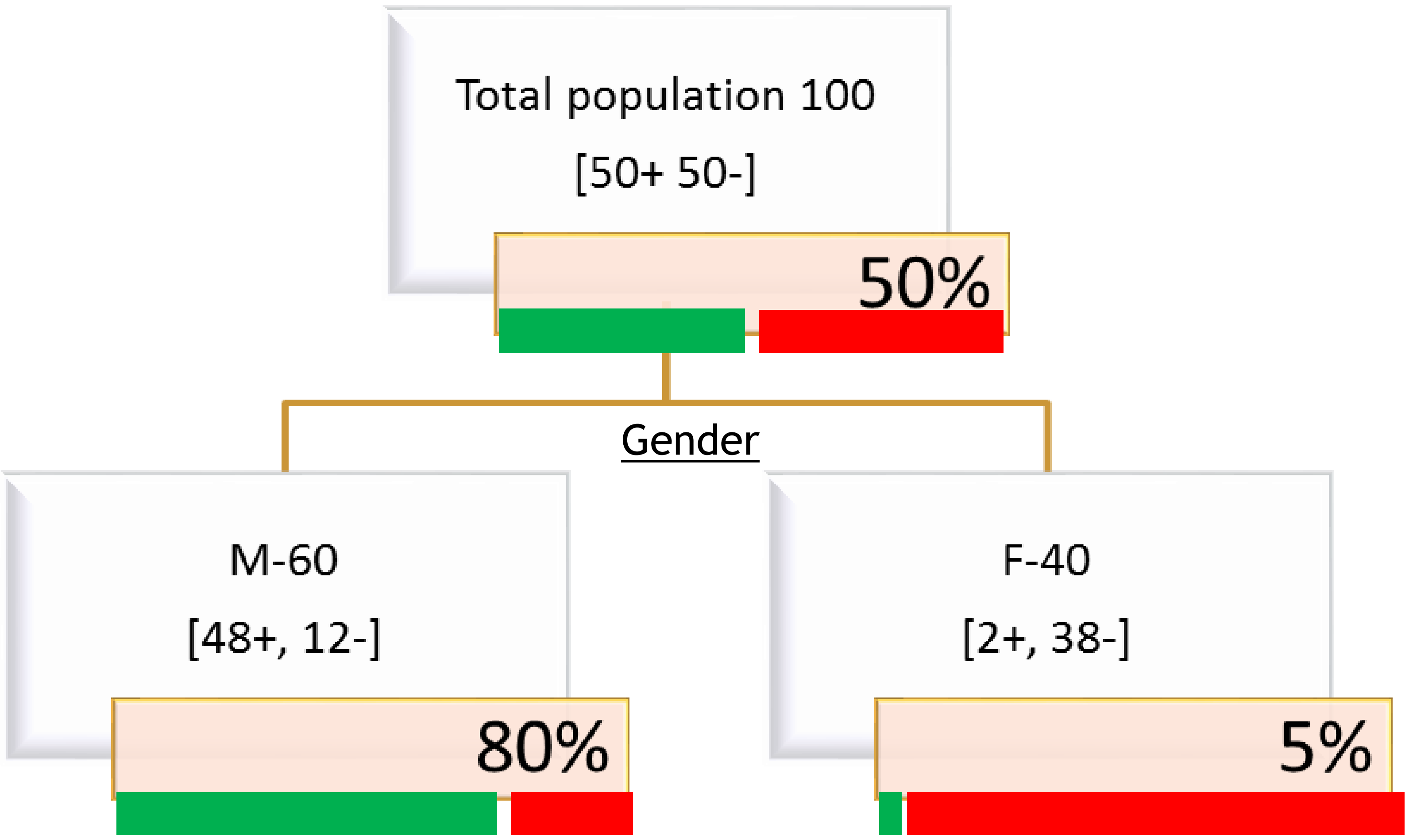
The Decision Tree Approach

- The aim is to divide the whole population or the data set into segments
- The segmentation need to be useful for business decision making.
- If one class is really dominating in a segment
  - Then it will be easy for us to classify the unknown items
  - Then its very easy for applying business strategy
- For example:
  - It takes no great skill to say that the customers have 50% chance to buy and 50% chance to not buy.
  - A good splitting criterion segments the customers with 90% -10% buying probability, say Gender="Female" customers have 5% buying probability and 95% not buying

Example Sales Segmentation Based on Age



Example Sales Segmentation Based on Gender



Main questions

- Ok, we are looking for pure segments
- Dataset has many attributes
- Which is the right attribute for pure segmentation?
- Can we start with any attribute?
- Which attribute to start? – The best separating attribute
- Customer Age can impact the sales, gender can impact sales , customer place and demographics can impact the sales. How to identify the best attribute and the split?

statinfer

The Splitting Criterion

The Splitting Criterion

- The best split is
  - The split does the best job of separating the data into groups
    - Where a single class(either 0 or 1) predominates in each group

The Decision tree Algorithm

The Decision tree Algorithm

- The major step is to identify the best split variables and best split criteria
- Once we have the split then we have to go to segment level and drill down further

LAB: Decision Tree Building

LAB: Decision Tree Building

- Data:Ecom_Cust_Relationship_Management/Ecom_Cust_Survey.csv
- How many customers have participated in the survey?
- Overall most of the customers are satisfied or dis-satisfied?
- Can you segment the data and find the concentrated satisfied and dis-satisfied customer segments ?
- What are the major characteristics of satisfied customers?
- What are the major characteristics of dis-satisfied customers?
- What are the final rules of the tree

Steps - Decision Tree Building

- Drag and drop the dataset into the canvas
- No. of customers participated in the survey:
  - Visualize the data, No. of rows is the value for No. of customers
- No. of satisfied and dis-satisfied customers
  - Drag and drop Split Data, connect it to the dataset
  - Select Splitting mode as Regular Expression
  - expression = \"Overall_Satisfaction" ^Dis Satisfied
  - Click run
  - Visualize the first output circle of Split Data No. of rows is the No. of dis-satisfied customers
  - Visualize the second output circle of Split Data No. of rows is the No. of satisfied customers

statinfer

Steps - Decision Tree Building

Fig1: Total No. of Customers

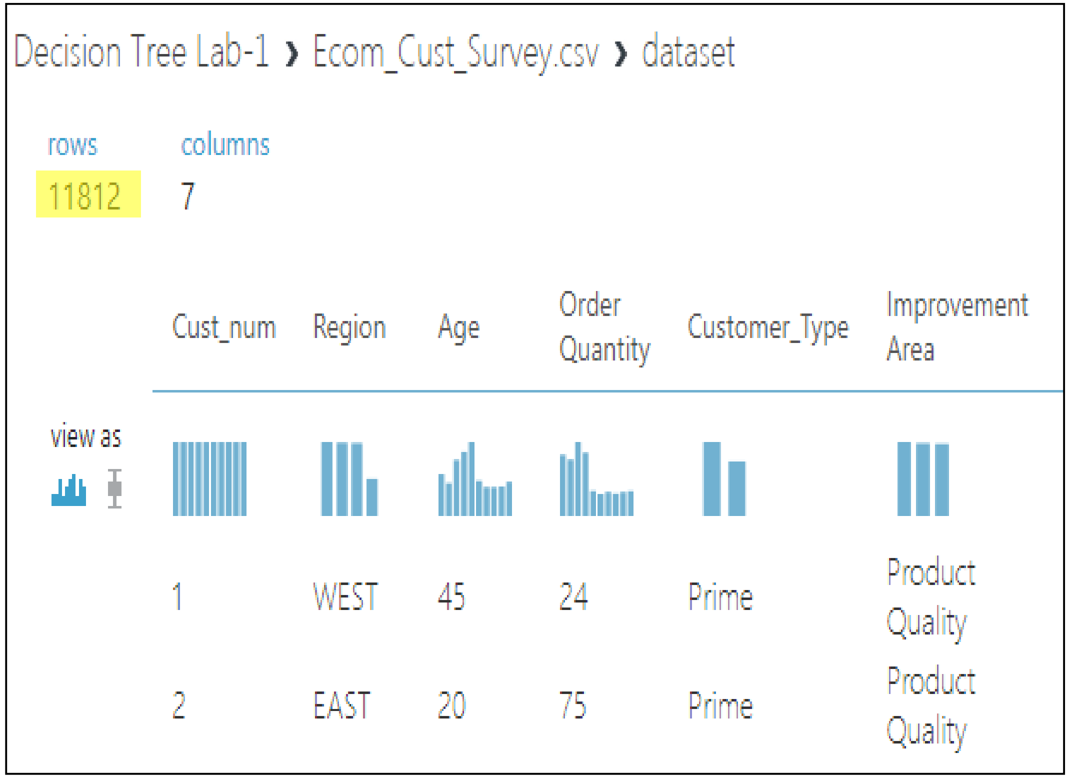
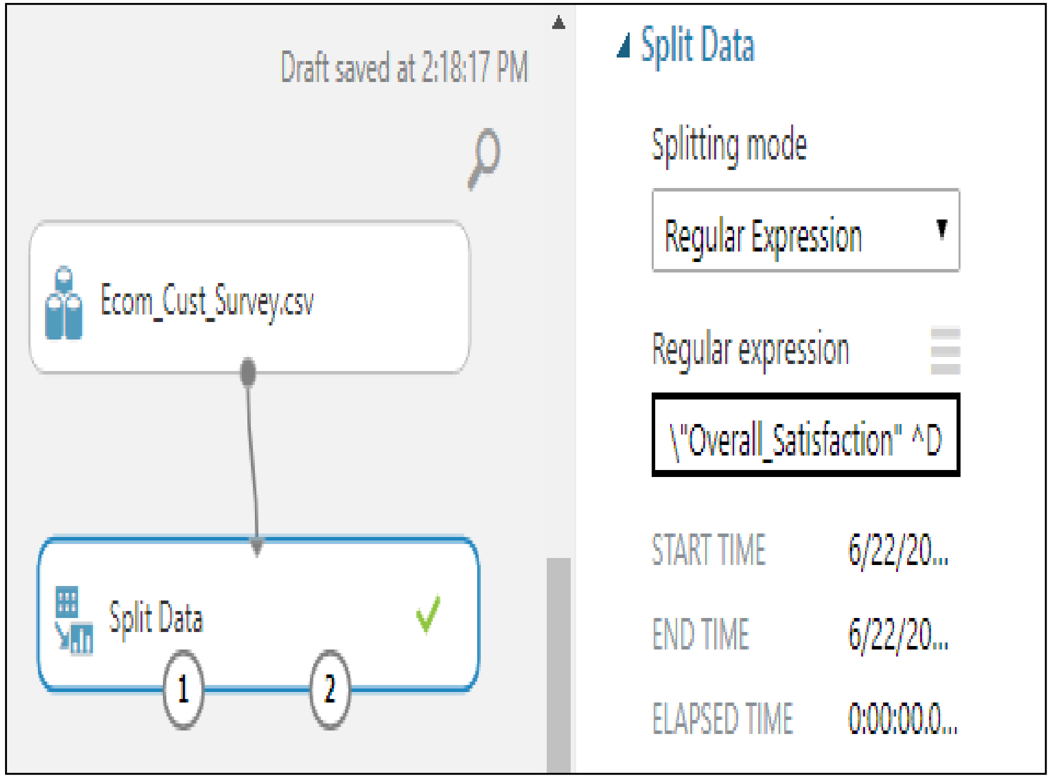
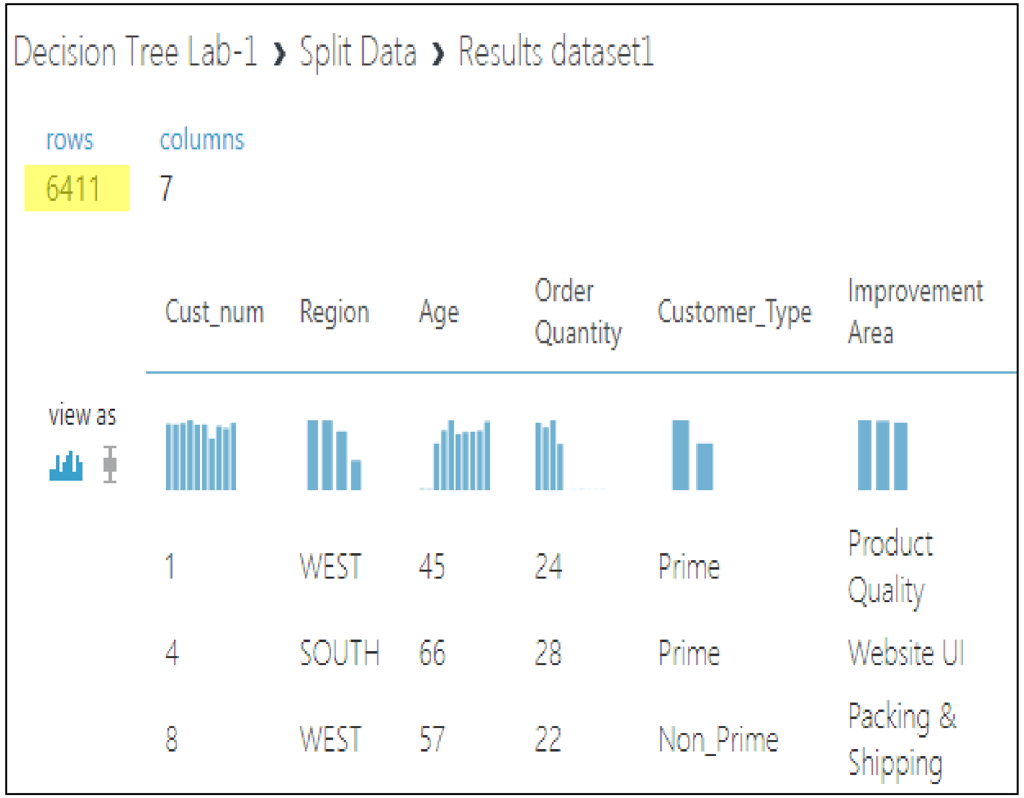


Fig2: Splitting Data



Steps - Decision Tree Building

Fig3: Total No. of Dis-Satisfied Customers

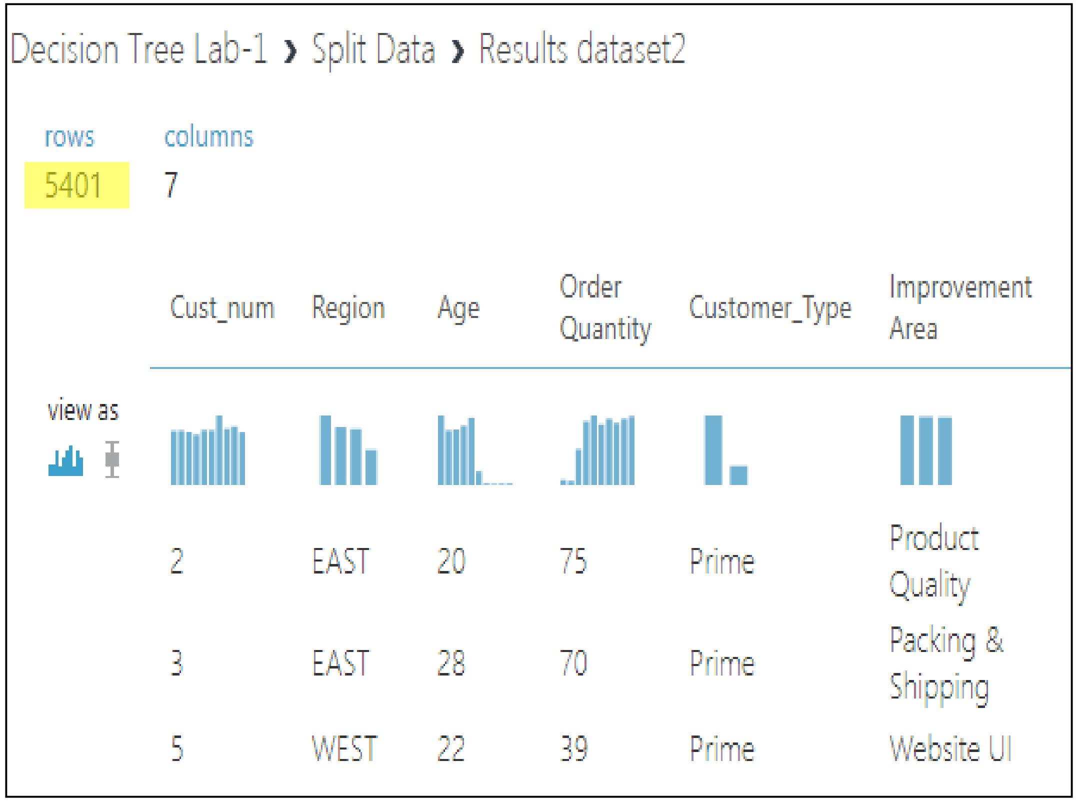


Decision Tree Lab-1 › Split Data › Results dataset1

rows	columns
6411	7

Cust_num	Region	Age	Order Quantity	Customer_Type	Improvement Area
1	WEST	45	24	Prime	Product Quality
4	SOUTH	66	28	Prime	Website UI
8	WEST	57	22	Non_Prime	Packing & Shipping

Fig4: Total No. of Satisfied Customers



Decision Tree Lab-1 › Split Data › Results dataset2

rows	columns
5401	7

Cust_num	Region	Age	Order Quantity	Customer_Type	Improvement Area
2	EAST	20	75	Prime	Product Quality
3	EAST	28	70	Prime	Packing & Shipping
5	WEST	22	39	Prime	Website UI

Steps - Decision Tree Building

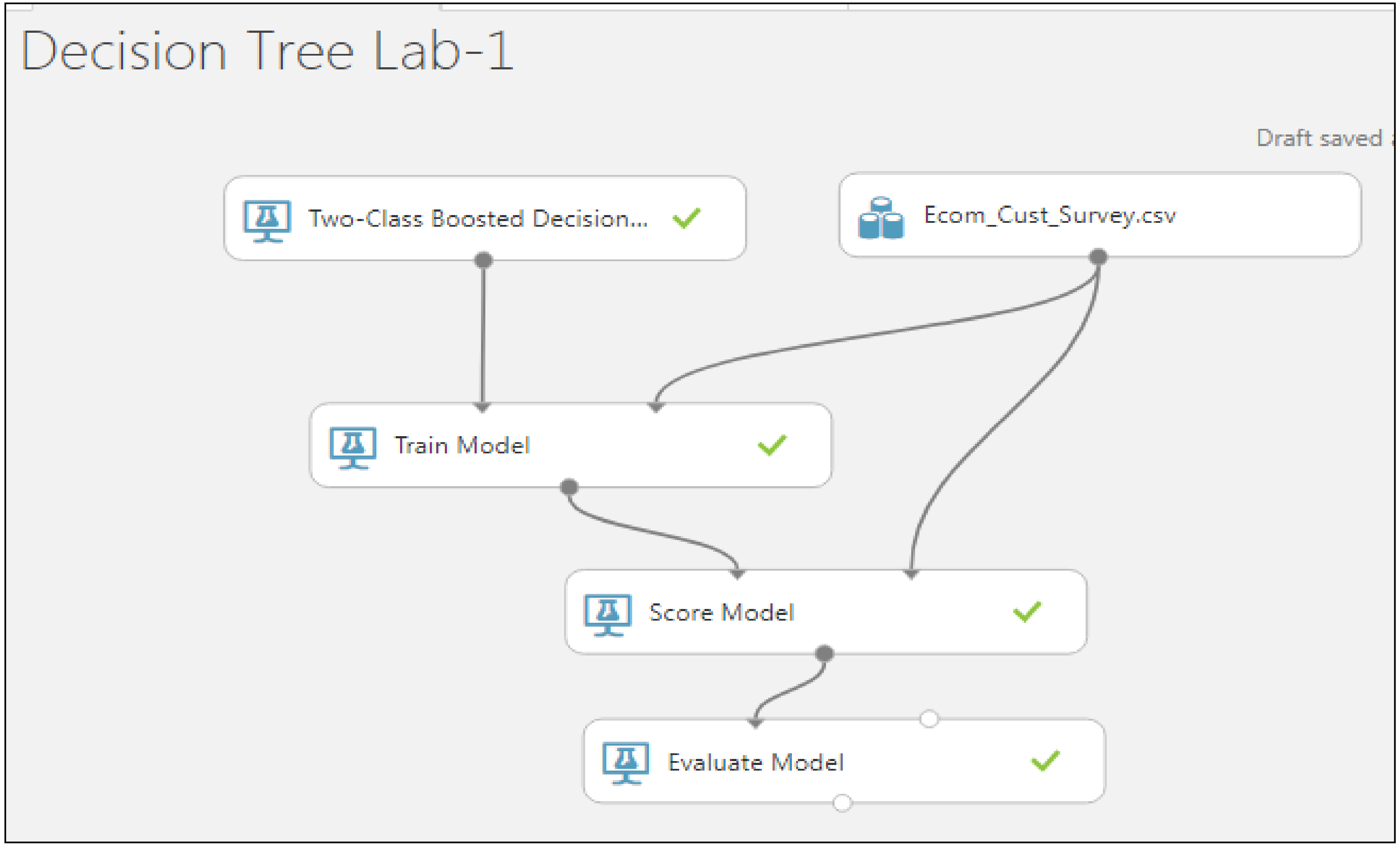
- Building Decision Tree :
  - Drag and drop the Dataset into the canvas
  - Drag and drop **Two-Class Boosted Decision Tree, Train Model, Score Model** and **Evaluate Model**
  - Connect **Two-Class Boosted Decision Tree** to the first input of **Train Model** and **Dataset** to the Second input of **Train Model**
  - Connect the output of **Train Model** first input of **Score Model** and **Dataset** to the Second input of **Score Model**
  - Connect the output of **Score Model** to the input of **Evaluate Model**

Steps - Decision Tree Building

- Click on **Two-Class Boosted Decision Tree** and select the following:
  - Create trainer mode → Single Parameter
  - Maximum number of leaves per tree → 3
  - Minimum number of samples per leaf node → 30
  - Learning rate → 0.2
  - Number of trees constructed → 1
- Click on Train Model and select the column for which the prediction is done(Overall_Satisfaction)
- Click run and visualize the output of Train Model and Evaluate Model

Steps - Decision Tree Building

Fig5: Decision Tree Model



Steps - Decision Tree Building

Fig6: Properties(Two-Class Boosted Decision)

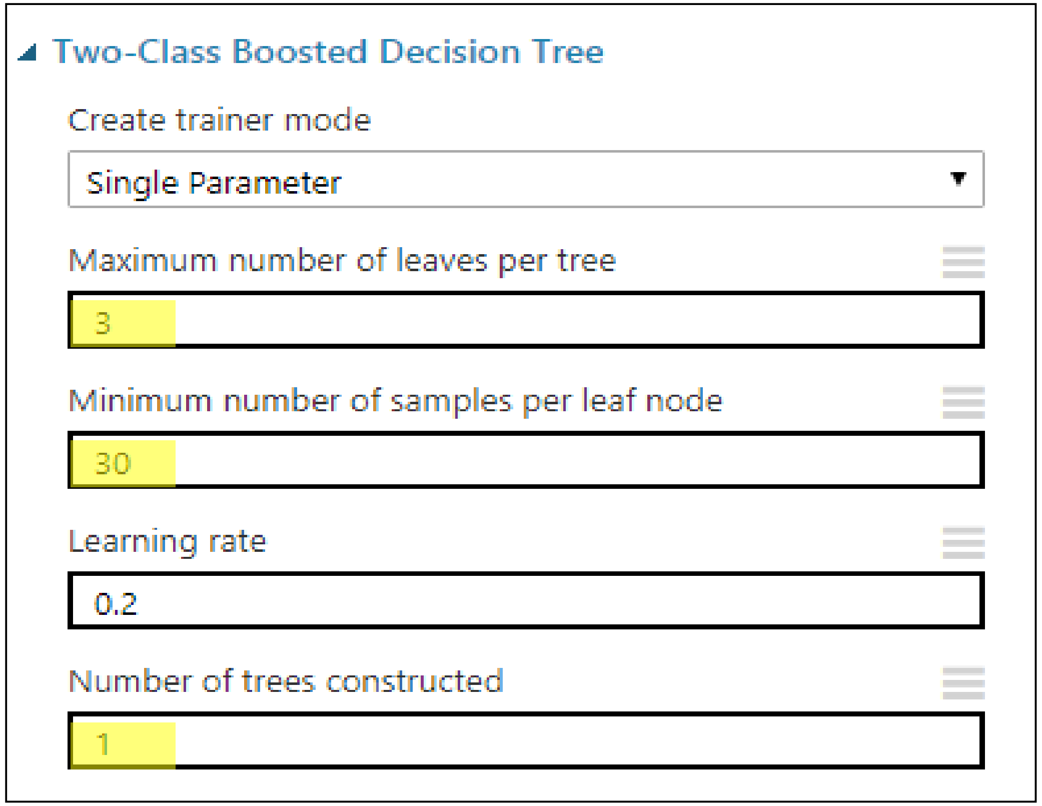
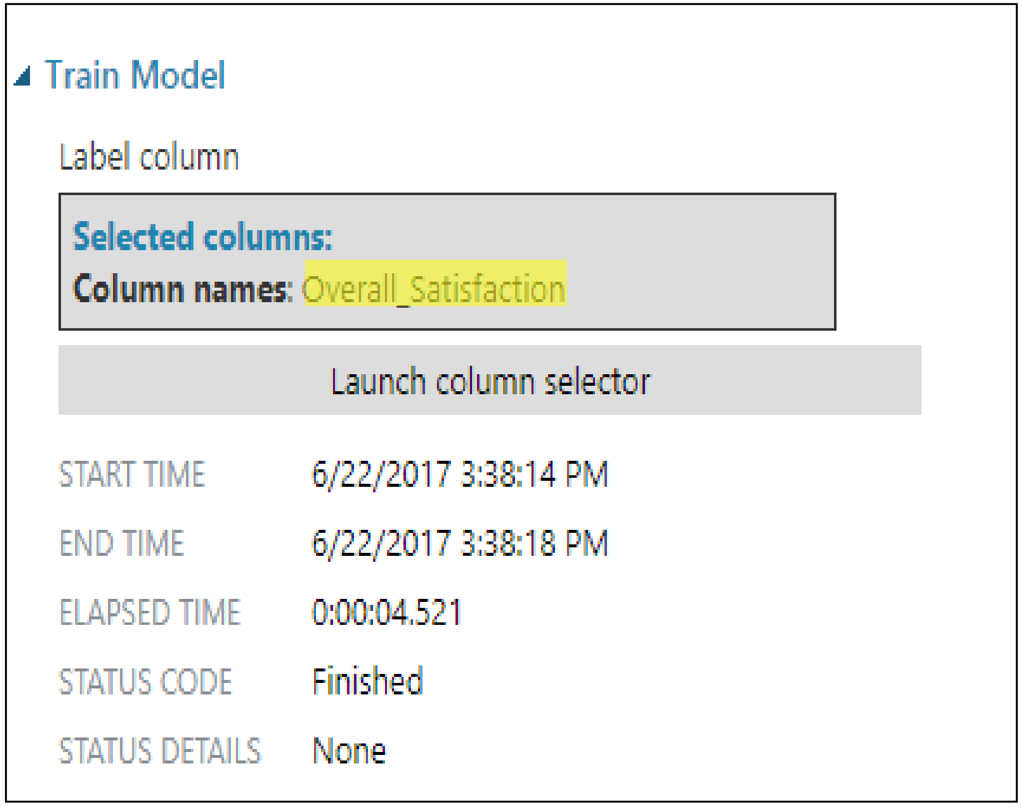


Fig7: Properties(Train Model)



Steps - Decision Tree Building

Fig8: Decision Tree (Satisfied Rule)

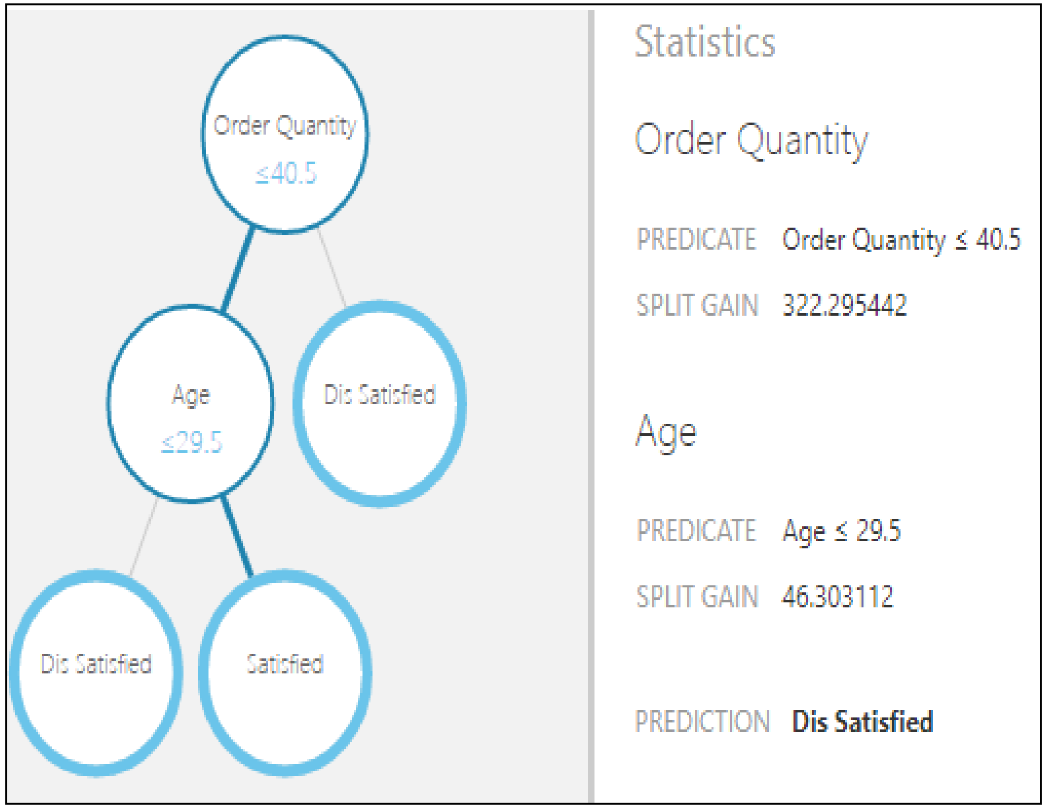
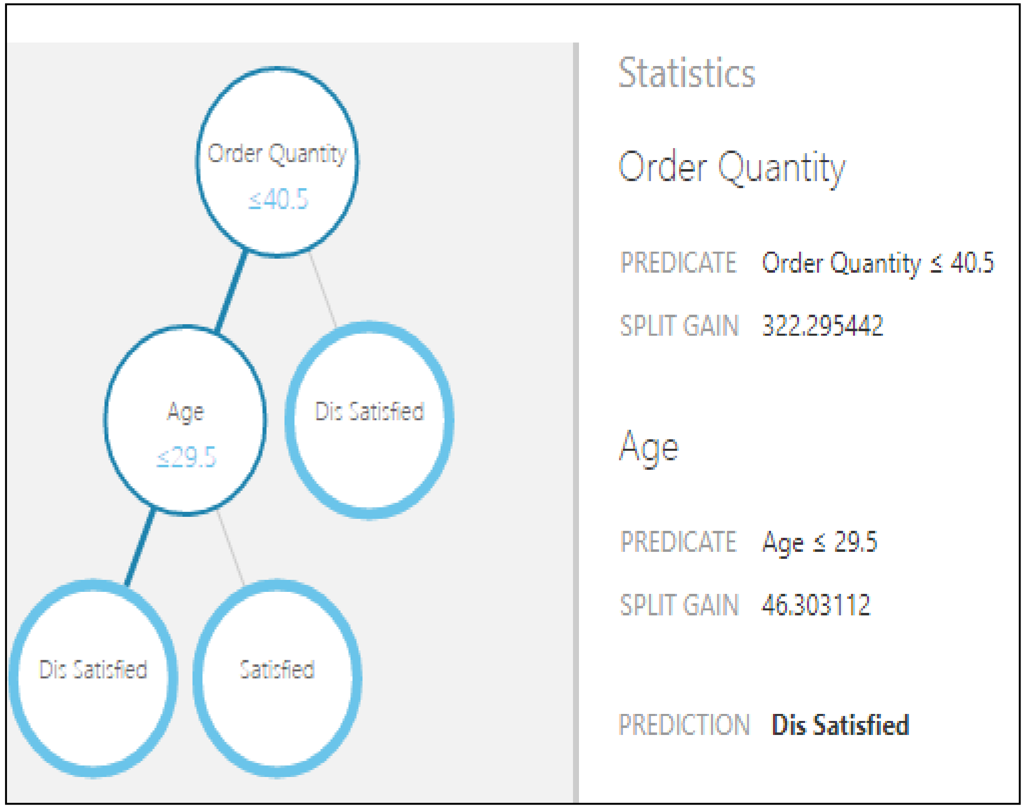


Fig9: Decision Tree(Dis-Satisfied Rule)



Note: Prediction - Satisfied is wrongly printed as Dis-Satisfied

Tree Validation

Classification Table & Accuracy

		Predicted Classes	
		0(Positive)	1(Negative)
Actual Classes	0(Positive)	True positive (TP) Actual condition is Positive, it is truly predicted as positive	False Negatives(FN) Actual condition is Positive, it is falsely predicted as negative
	1(Negative)	False Positives(FP) Actual condition is Negative, it is falsely predicted as positive	True Negatives(TN) Actual condition is Negative, it is truly predicted as negative

- Accuracy=(TP+TN)/(TP+FP+FN+TN)
- Misclassification Rate=(FP+FN)/(TP+FP+FN+TN)

statinfer

LAB: Tree Validation

LAB: Tree Validation

- Create the confusion matrix for the model
- Find the accuracy of the classification for the Ecom_Cust_Survey model

Steps - Tree Validation

Fig10: Decision Tree (Evaluation)

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
4749	648	0.942	0.992	0.5	0.937
False Positive	True Negative	Recall	F1 Score		
38	6370	0.880	0.933		
Positive Label	Negative Label				
Satisfied	Dis Satisfied				

statinfer

The Problem of Overfitting

LAB: The Problem of Overfitting

- Dataset: “Buyers Profiles/Train_data.csv”
- Import both test and training data
- Build a decision tree model on training data
- Find the accuracy on training data
- Find the predictions for test data
- What is the model prediction accuracy on test data?

Steps - The Problem of Overfitting

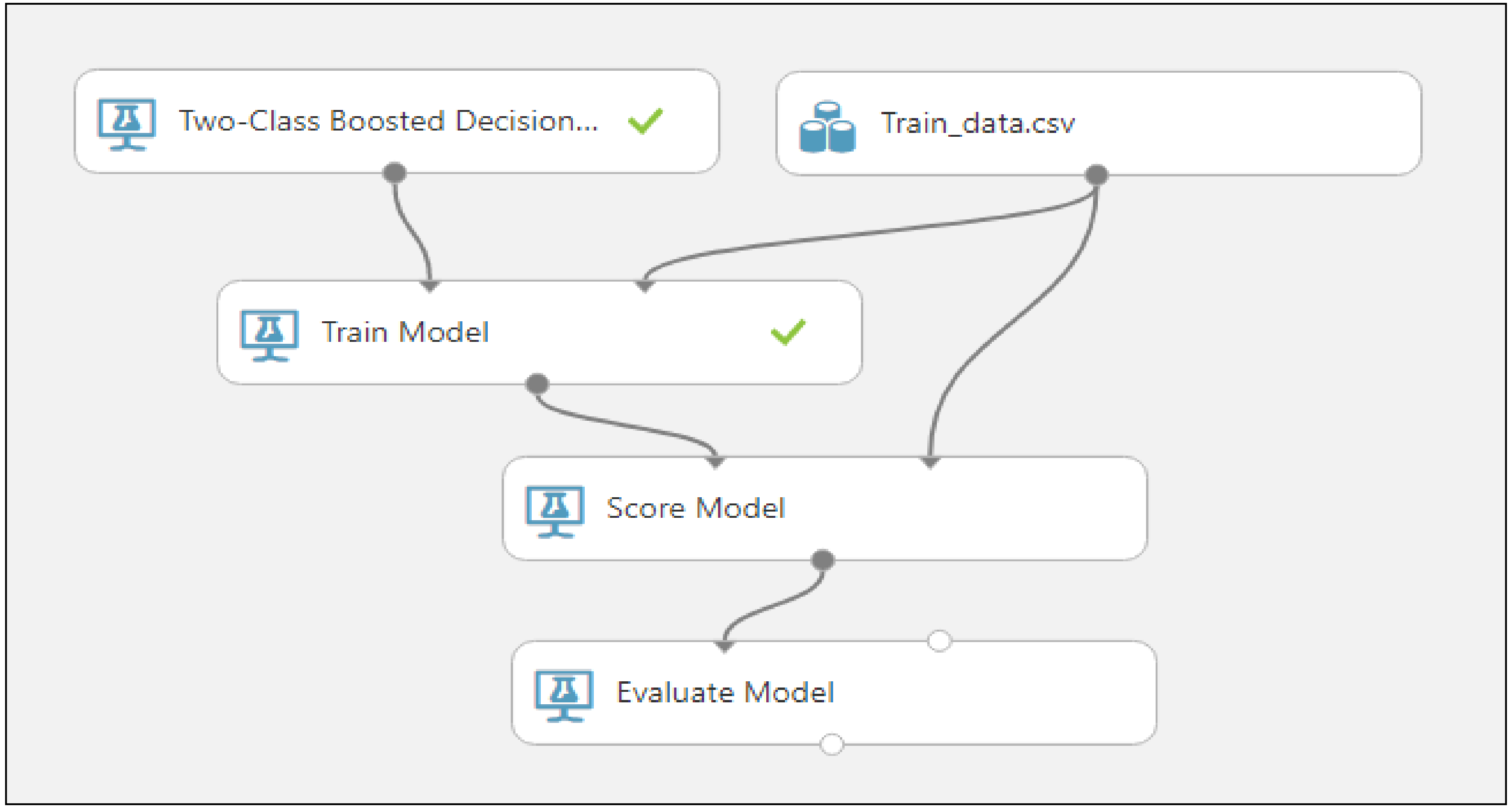
- Building Decision Tree with Training Data :
  - Drag and drop the Dataset into the canvas
  - Drag and drop **Two-Class Boosted Decision Tree, Train Model, Score Model** and **Evaluate Model**
  - Connect **Two-Class Boosted Decision Tree** to the first input of **Train Model** and **Dataset** to the Second input of **Train Model**
  - Connect the output of **Train Model** first input of **Score Model** and **Dataset** to the Second input of **Score Model**
  - Connect the output of **Score Model** to the input of **Evaluate Model**

Steps - The Problem of Overfitting

- Click on **Two-Class Boosted Decision Tree** and select the following:
  - Create trainer mode → Single Parameter
  - Maximum number of leaves per tree → 6
  - Minimum number of samples per leaf node → 1
  - Learning rate → 0.2
  - Number of trees constructed → 1
- Click on Train Model and select the column for which the prediction is done(Bought)
- Click run and visualize the output of Train Model and Evaluate Model

Steps - The Problem of Overfitting

Fig11: Model With Training Data



Steps - The Problem of Overfitting

Fig12: Properties(Two-Class Boosted Decision)

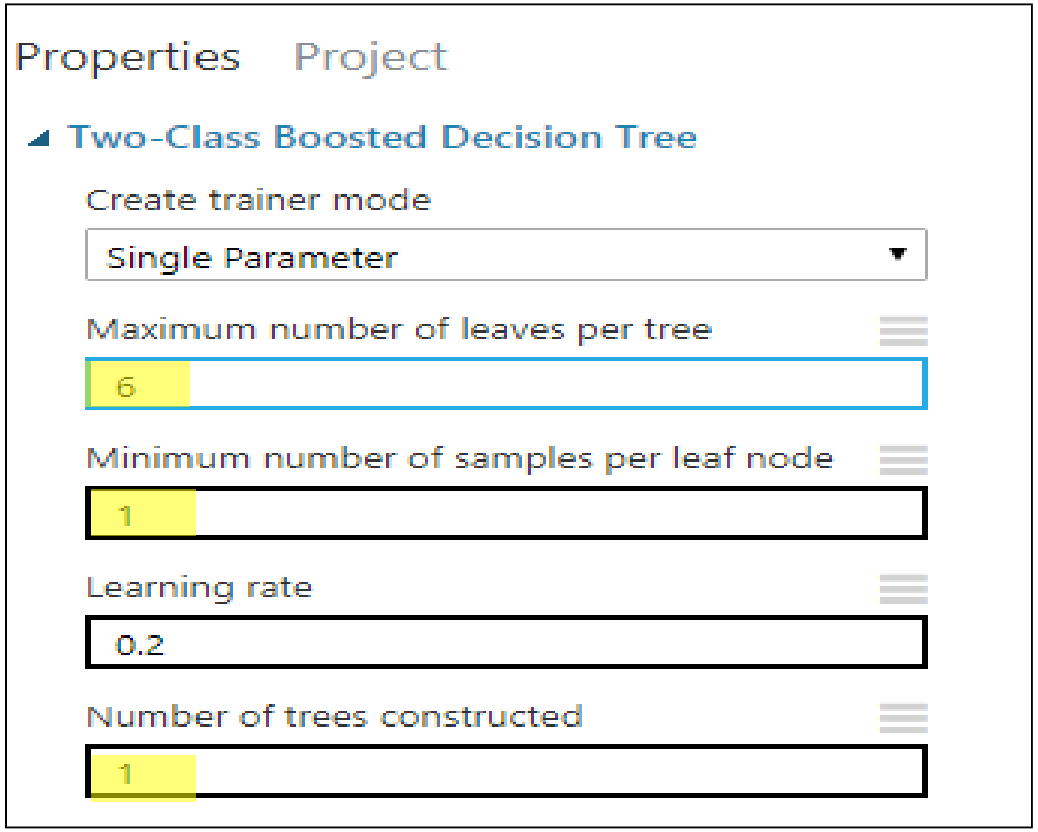
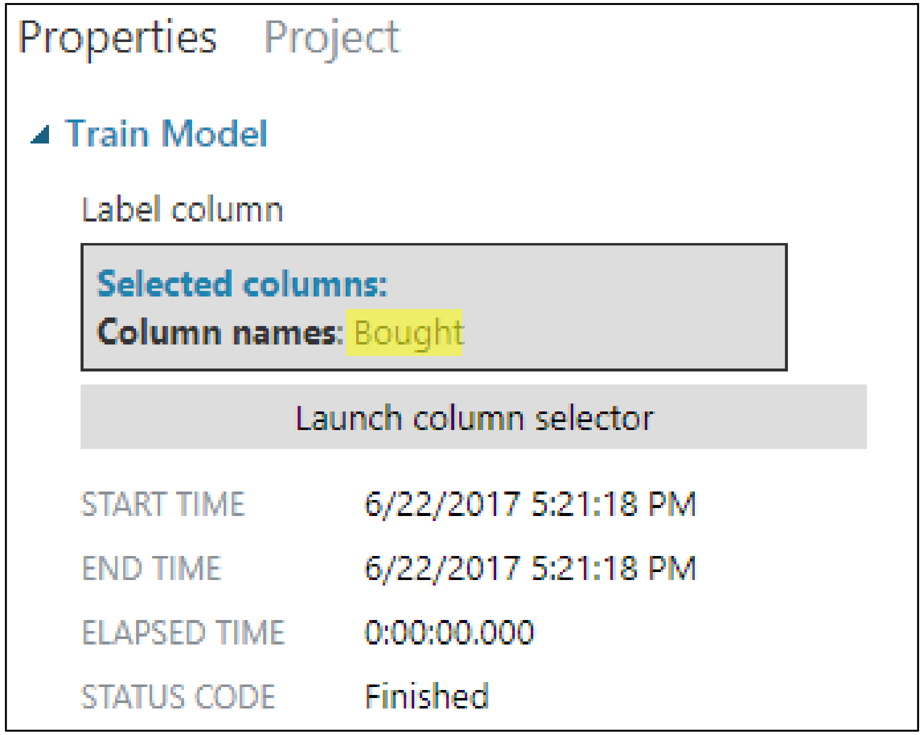


Fig13: Properties(Train Model)



Steps - The Problem of Overfitting

Fig14: Decision Tree Prediction(Training)

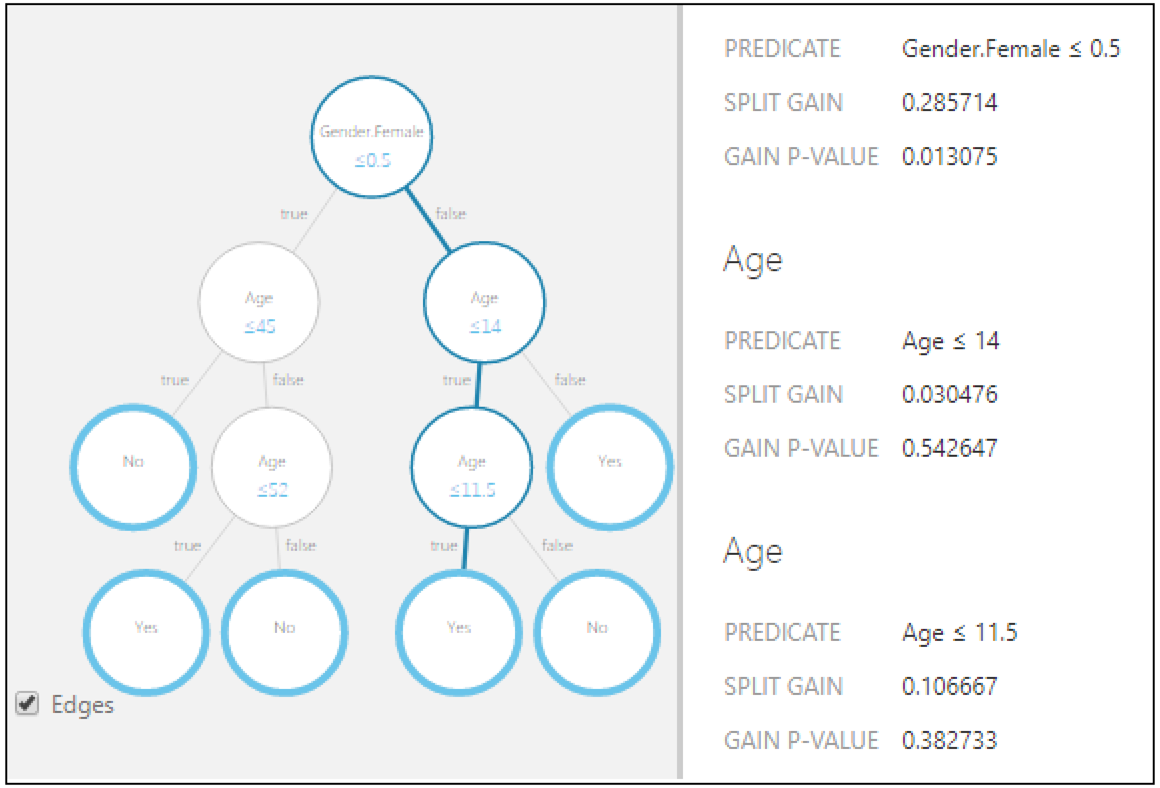


Fig15: Accuracy(Training)

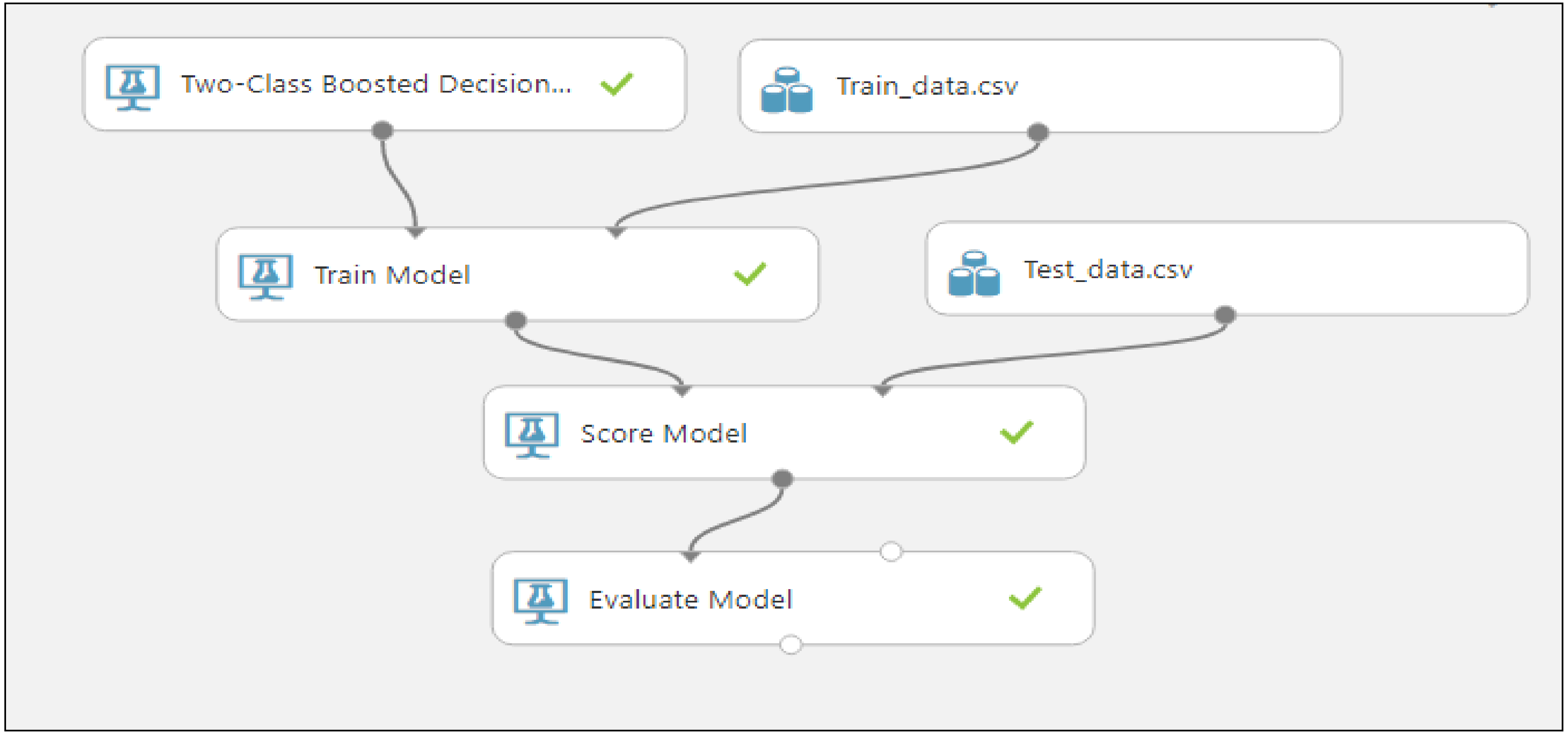
True Positive	False Negative	Accuracy	Precision
7	0	1.000	1.000
False Positive	**True Negative**	**Recall**	**F1 Score**
0	7	1.000	1.000
Positive Label	**Negative Label**		
Yes	No		

Steps - The Problem of Overfitting

- Building Decision Tree with Testing Data :
  - With the same model instead of passing Training Dataset to the Score Model now pass the Test Data
  - Click run

Steps - The Problem of Overfitting

Fig16: Model With Test Data



Steps - The Problem of Overfitting

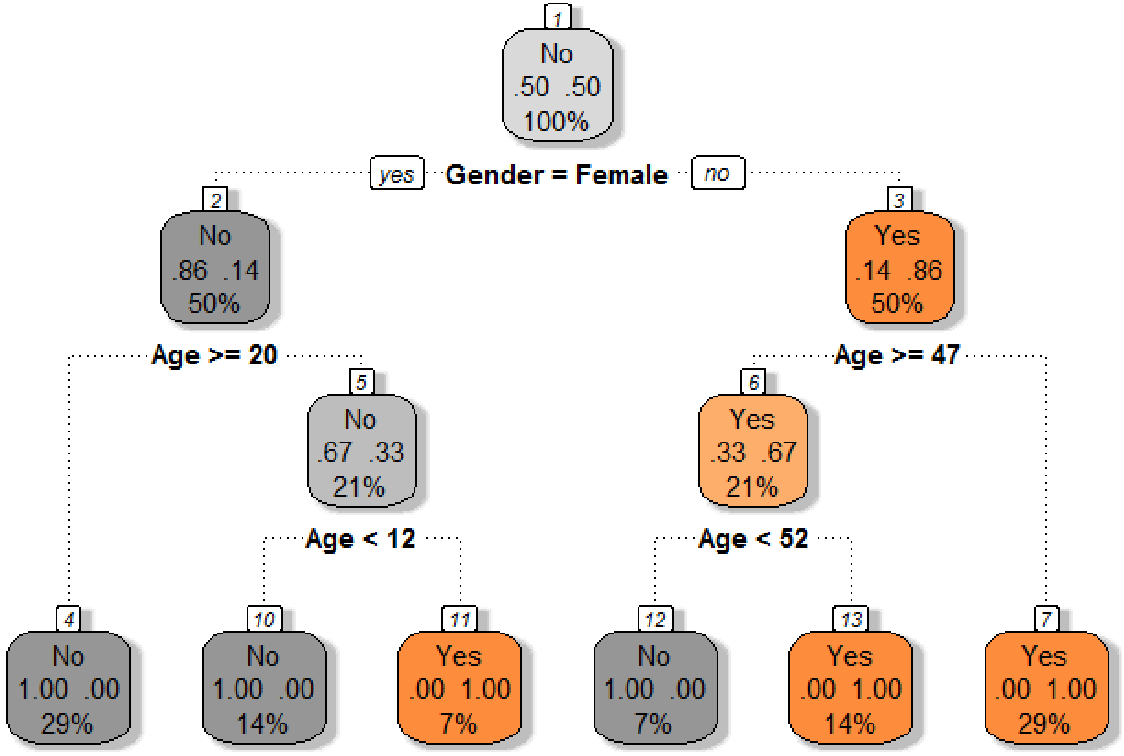
Fig17: Accuracy(Test Data)

True Positive 1	False Negative 3	Accuracy 0.333	Precision 0.500
False Positive 1	True Negative 1	Recall 0.250	F1 Score 0.333
Positive Label Yes	Negative Label No		

The Problem of Overfitting

- Build a decision tree on Prune_Sample.csv

Age	Gender	Bought
29	Male	Yes
34	Male	Yes
13	Female	Yes
27	Female	No
10	Female	No
68	Male	Yes
15	Male	Yes
53	Male	Yes
51	Male	No
48	Female	No
63	Female	No
43	Male	Yes
8	Female	No
47	Female	No



The Final Tree with Rules

4) Gender=Female & Age>=20 No *

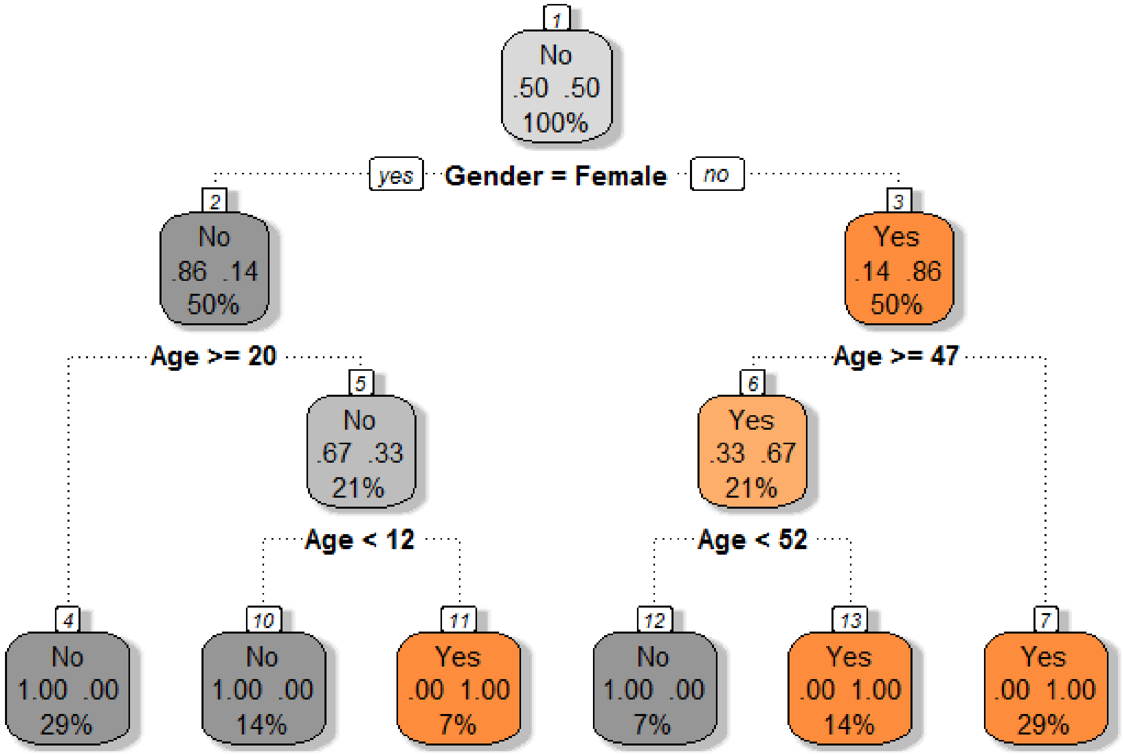
10) Gender=Female & Age< 20 & Age< 11.5 No *

11) Gender=Female & Age< 20 & Age>=11.5 Yes *

12) Gender=Male & Age>=47 & Age< 52 No *

13) Gender=Male & Age>=47 & Age>=52 Yes *

7) Gender=Male & Age< 47 Yes *



The Problem of Overfitting

Age	Gender	Bought
29	Male	Yes
34	Male	Yes
13	Female	Yes
27	Female	No
10	Female	No
68	Male	Yes
15	Male	Yes
53	Male	Yes
51	Male	No
48	Female	No
63	Female	No
43	Male	Yes
8	Female	No
47	Female	No

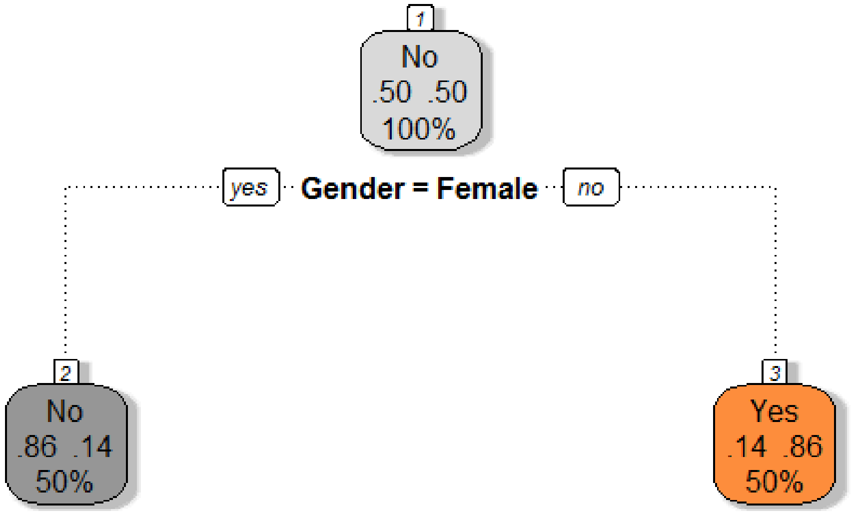
- If we further grow the tree we might even see each row of the input data table as the final rules
- The model will be really good on the training data but it will fail to validate on the test data
- Growing the tree beyond a certain level of complexity leads to overfitting
- A really big tree is very likely to suffer from overfitting.

Pruning

Pruning

Age	Gender	Bought
29	Male	Yes
34	Male	Yes
13	Female	Yes
27	Female	No
10	Female	No
68	Male	Yes
15	Male	Yes
53	Male	Yes
51	Male	No
48	Female	No
63	Female	No
43	Male	Yes
8	Female	No
47	Female	No

- Growing the tree beyond a certain level of complexity leads to overfitting
- In our data, age doesn't have any impact on the target variable.
- Growing the tree beyond Gender is not going to add any value. Need to cut it at Gender
- This process of trimming trees is called Pruning



Pruning with Value of Split Gain

- Pruning helps us to avoid overfitting
- Generally it is preferred to have a simple model, it avoids overfitting issue
- Any additional split that does not add significant value is not worth while.
- The value of split gain gives an idea about the split by seeing this we can decide weather the split is required or not

LAB: Pruning

LAB: Pruning

- Rebuild the model for above data
- Check the Split Gain at each node
- Change the value of 'Maximum number of leaves per tree' to achieve an optimal level tree
- Prune the decision tree
- calculate the training and test Accuracy
- Check whether there is an issue of overfitting in the final model

Steps - Pruning

- At the second node we can find that the split gain is only 3%
- Change the value of Maximum number of leaves per tree to 2
- Click on run
- Check the values accuracy and confusion matrix

Steps - Pruning

Fig18: Properties(Two-Class Boosted Decision Tree)

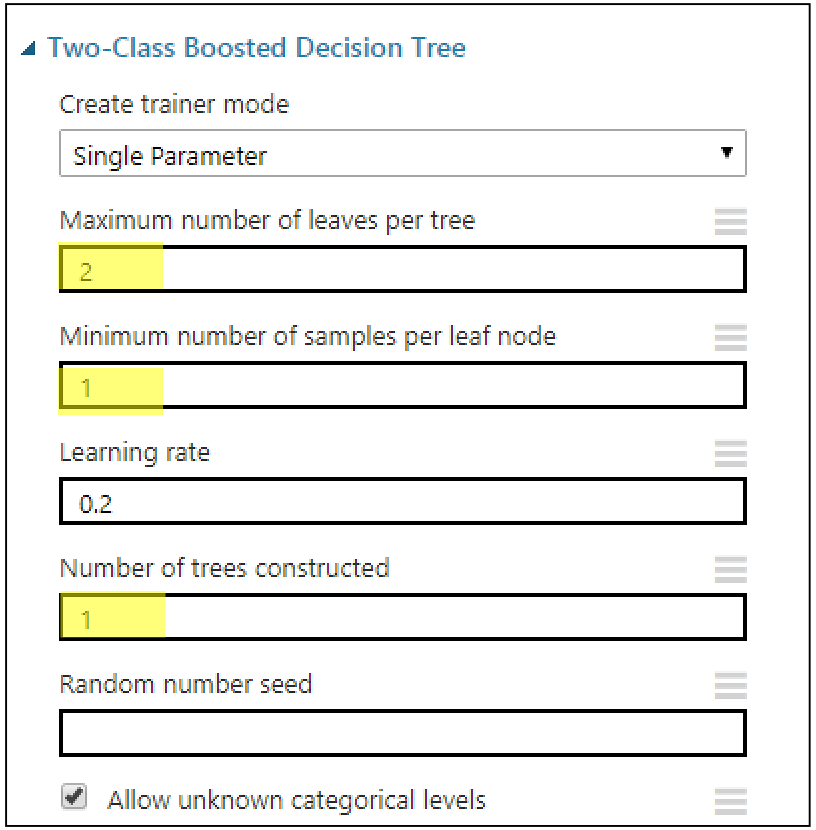
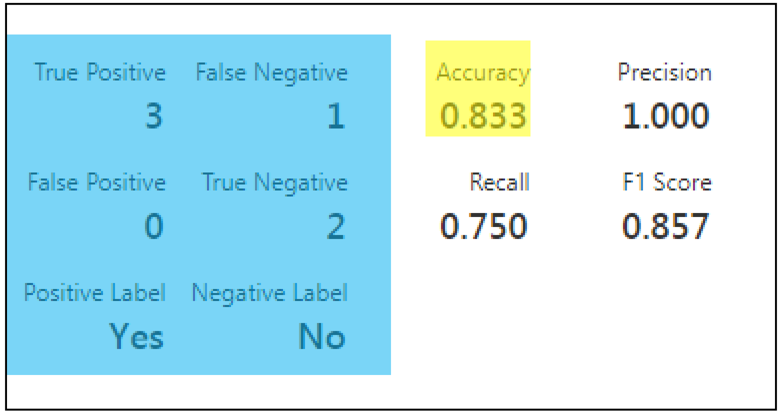


Fig19: Accuracy and Confusion Matrix



Two types of pruning

Two types of pruning

- **Pre-Pruning:**
  - Building the tree by mentioning Cp value upfront
- **Post-pruning:**
  - Grow decision tree to its entirety, trim the nodes of the decision tree in a bottom-up fashion

LAB: Tree Building & Model Selection

LAB: Tree Building & Model Selection

- Import fiber bits data. This is internet service provider data. The idea is to predict the customer attrition based on some independent factors
- Build a decision tree model for fiber bits data
- Prune the tree if required
- Find out the final accuracy
- Is there any 100% active/inactive customer segment?

Steps - Tree Building & Model Selection

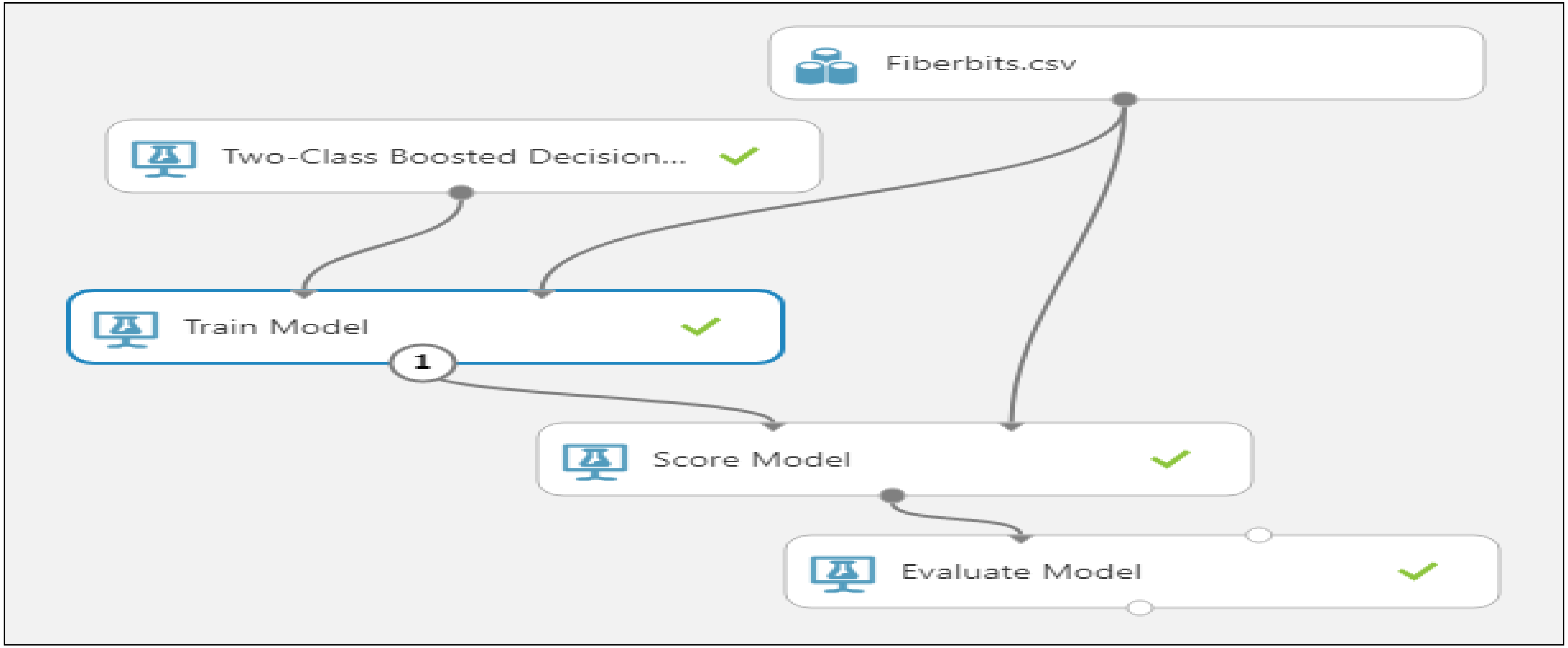
- Building Decision Tree with FiberBits Data :
  - Drag and drop the Dataset into the canvas
  - Drag and drop **Two-Class Boosted Decision Tree, Train Model, Score Model** and **Evaluate Model**
  - Connect **Two-Class Boosted Decision Tree** to the first input of **Train Model** and **Dataset** to the Second input of **Train Model**
  - Connect the output of **Train Model** first input of **Score Model** and **Dataset** to the Second input of **Score Model**
  - Connect the output of **Score Model** to the input of **Evaluate Model**

Steps - Tree Building & Model Selection

- Click on **Two-Class Boosted Decision Tree** and select the following:
  - Create trainer mode → Single Parameter
  - Maximum number of leaves per tree → 15
  - Minimum number of samples per leaf node → 30
  - Learning rate → 0.09
  - Number of trees constructed → 1
- Click on Train Model and select the column for which the prediction is done(active_cust)
- Click run and visualize the output of Train Model and Evaluate Model

Steps - Tree Building & Model Selection

Fig20: Decision Tree Modal(FiberBits)



Steps - Tree Building & Model Selection

Fig21: Properties(Two-Class Boosted Decision Tree)

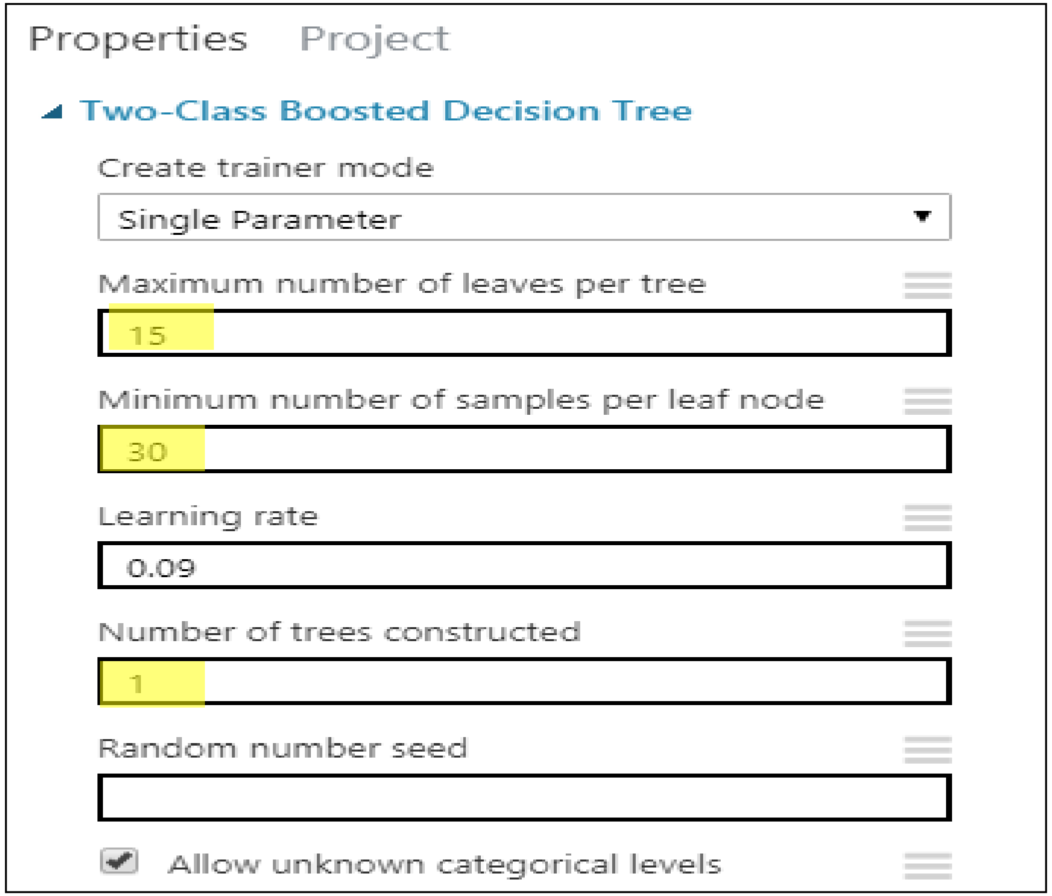
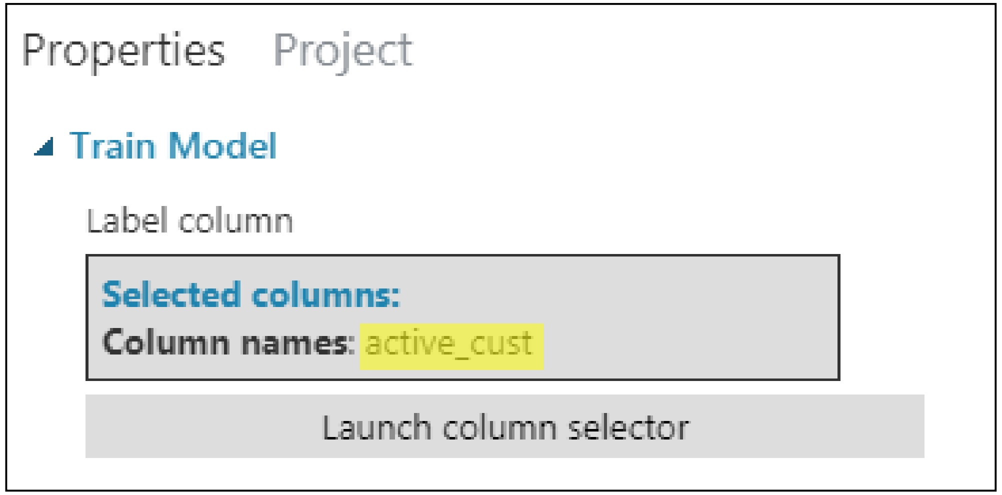
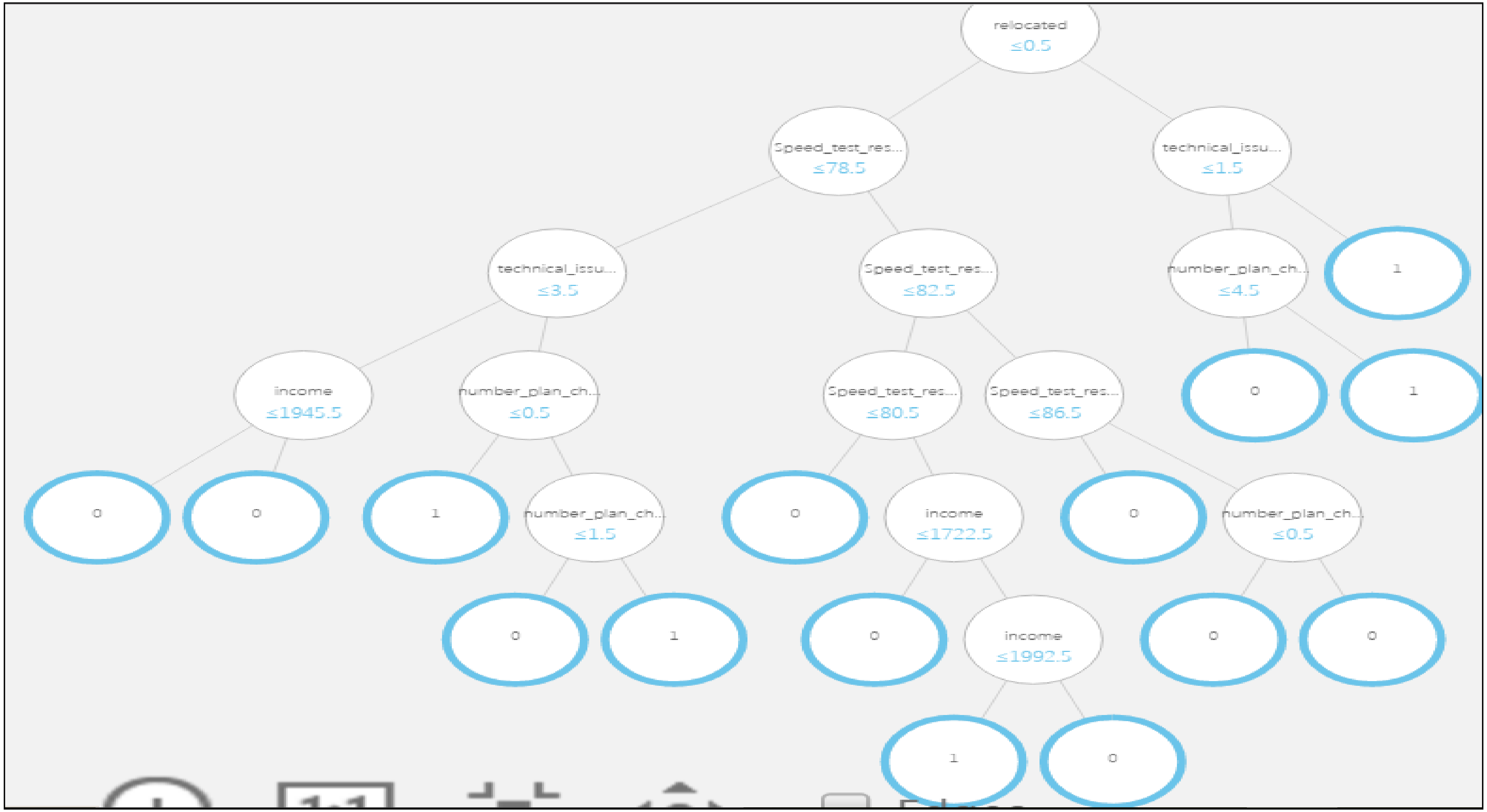


Fig22: Properties(Train Model)



Steps - Tree Building & Model Selection

Fig23: Decision Tree



Steps - Tree Building & Model Selection

Fig24: Statistics(income ≤ 1992.5)

Statistics

relocated

PREDICATE	relocated ≤ 0.5
SPLIT GAIN	114.717084

Speed_test_result

PREDICATE	Speed_test_result ≤ 78.5
SPLIT GAIN	98.127054

Speed_test_result

PREDICATE	Speed_test_result ≤ 82.5
SPLIT GAIN	26.649203

Fig25: Statistics(income ≤ 1992.5)

Speed_test_result

PREDICATE	Speed_test_result ≤ 80.5
SPLIT GAIN	23.578085

income

PREDICATE	income ≤ 1722.5
SPLIT GAIN	27.590679

income

PREDICATE	income ≤ 1992.5
SPLIT GAIN	7.163877

Steps - Tree Building & Model Selection

Fig26: Accuracy and Confusion Matrix(with 15 leaf nodes)

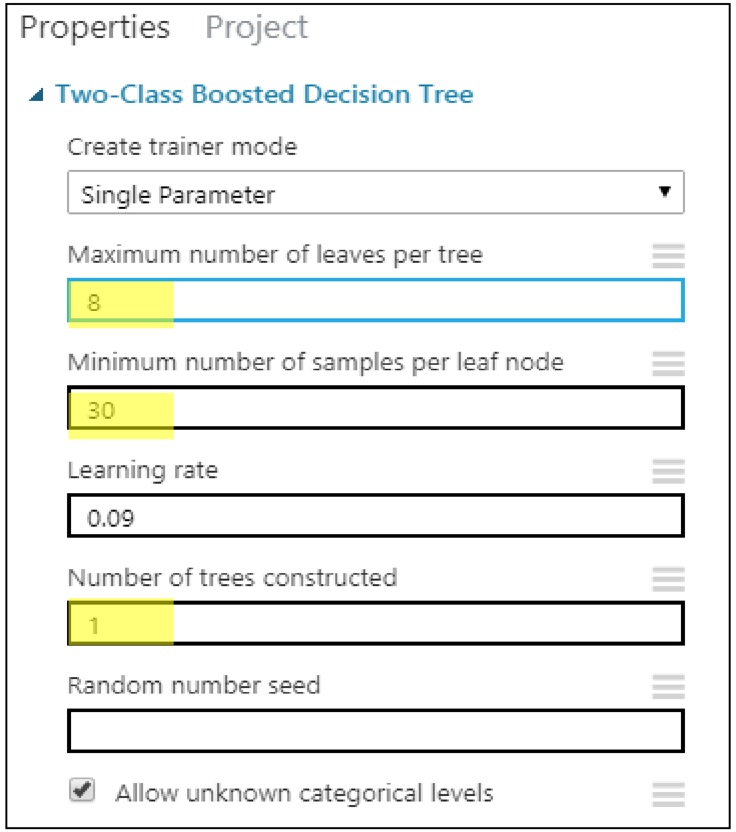
True Positive	False Negative	Accuracy	Precision
53910	3949	0.852	0.832
False Positive	True Negative	Recall	F1 Score
10871	31270	0.932	0.879
Positive Label	Negative Label		
1	0		

Steps - Tree Building & Model Selection

- The Split Gain value of the last level nodes are less than 10% and we can remove them by changing the values of the leaf nodes
- We do this by reducing the value of Maximum number of leaves per tree from 15 to 8
- Click run
- Check the accuracy of the modal if it is reducing to a larger amount then increase the value of Maximum number of leaves per tree
- If accuracy is not reducing much try to find out any other split has lesser gain and can be removed without having much impact on the accuracy
- If any then remove it and run the modal again

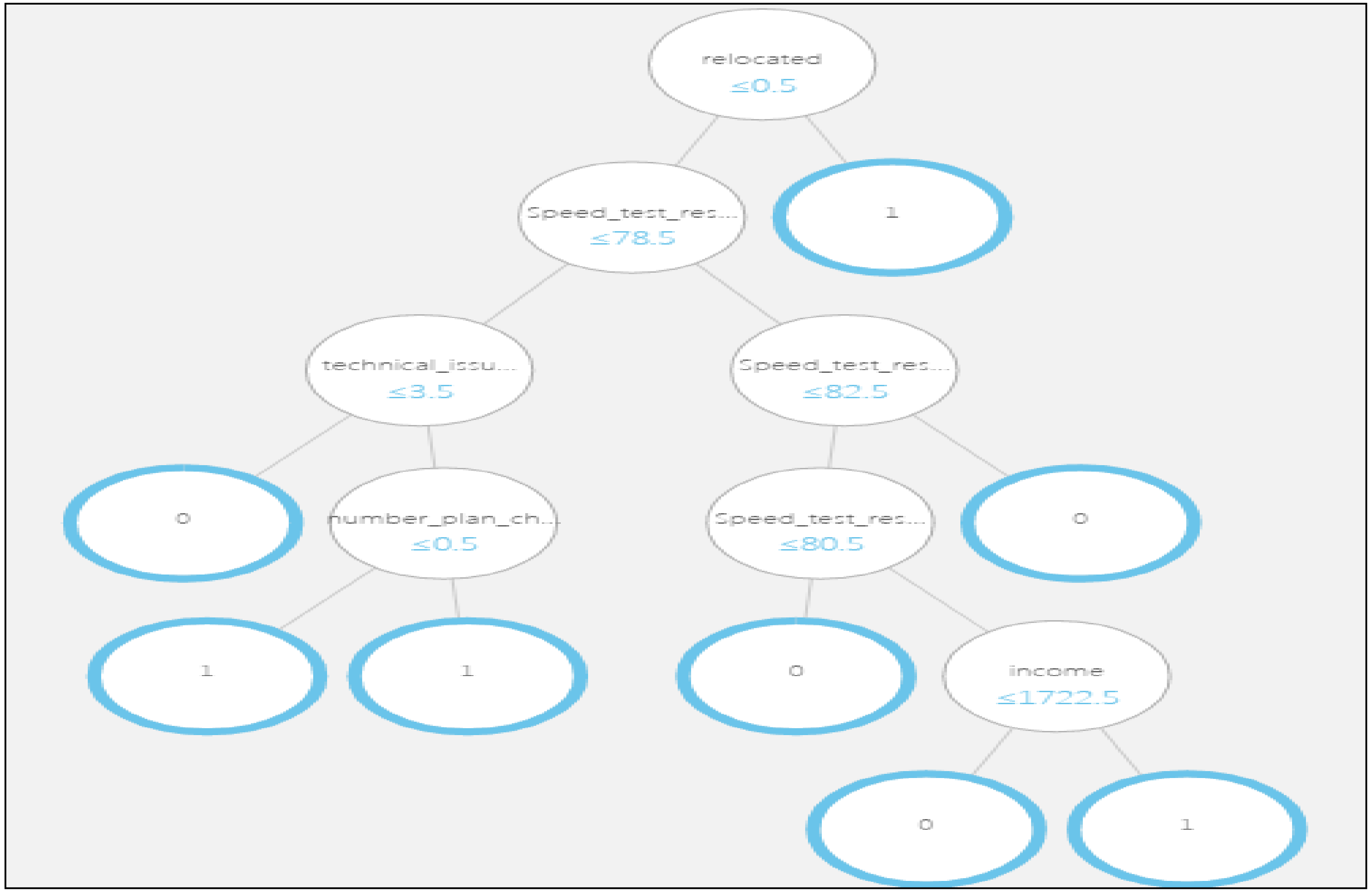
Steps - Tree Building & Model Selection

Fig27: Properties(**Two-Class Boosted Decision Tree)**



Steps - Tree Building & Model Selection

Fig28: **Decision Tree with 8 Leaf Nodes**



Steps - Tree Building & Model Selection

Fig29: Statistics(income ≤ 1722.5)

Statistics

relocated

PREDICATE	relocated ≤ 0.5
SPLIT GAIN	114.717084

Speed_test_result

PREDICATE	Speed_test_result ≤ 78.5
SPLIT GAIN	98.127054

Speed_test_result

PREDICATE	Speed_test_result ≤ 82.5
SPLIT GAIN	26.649203

Fig30: Statistics(income ≤ 1722.5)

Speed_test_result

PREDICATE	Speed_test_result ≤ 80.5
SPLIT GAIN	23.578085

income

PREDICATE	income ≤ 1722.5
SPLIT GAIN	27.590679

Steps - Tree Building & Model Selection

Fig31: **Accuracy and Confusion Matrix(with 8 leaf nodes)**

True Positive	False Negative	Accuracy	Precision
49815	8044	0.826	0.842
False Positive	**True Negative**	**Recall**	**F1 Score**
9344	32797	0.861	0.851
Positive Label	**Negative Label**		
1	0		

Conclusion

Conclusion

- Decision trees are powerful and very simple to represent and understand.
- One need to be careful with the size of the tree. Decision trees are more prone to overfitting than other algorithms
- Can be applied to any type of data, especially with categorical predictors
- One can use decision trees to perform a basic customer segmentation and build a different predictive model on the segments

Thank you

statinfer

We offer training on

www.statinfer.com

- Data Analytics
- Data Visualization
- Predictive Modelling
- Data Science
- Machine Learning
- Deep Learning
- R
- Python
- TensorFlow

Part 8/12 - Model Selection and Cross Validation with Azure

Venkat Reddy

Contents

- How to validate a model?
- What is a best model ?
- Types of data
- Types of errors
- The problem of over fitting
- The problem of under fitting
- Bias Variance Tradeoff
- Cross validation
- Boot strapping

statinfer

Model Validation Metrics

Model Validation

- Checking how good is our model
- It is very important to report the accuracy of the model along with the final model
- The model validation in regression is done through R square and Adj R-Square
- Logistic Regression, Decision tree and other classification techniques have the very similar validation measures.
- Till now we have seen confusion matrix and accuracy. There are many more validation and model accuracy metrics for classification models

Classification-Validation measures

- Confusion matrix, Specificity, Sensitivity
- ROC, AUC
- KS, Gini
- Concordance and discordance
- Chi-Square, Hosmer and Lemeshow Goodness-of-Fit Test
- Lift curve

All of them are measuring the model accuracy only. Some metrics work really well for certain class of problems. Confusion matrix, ROC and AUC will be sufficient for most of the business problems

Sensitivity and Specificity

Classification Table

Sensitivity and Specificity are derived from confusion matrix

		Predicted Classes	
		0(Positive)	1(Negative)
Actual Classes	0(Positive)	True positive (TP) Actual condition is Positive, it is truly predicted as positive	False Negatives(FN) Actual condition is Positive, it is falsely predicted as negative
	1(Negative)	False Positives(FP) Actual condition is Negative, it is falsely predicted as positive	True Negatives(TN) Actual condition is Negative, it is truly predicted as negative

- Accuracy=(TP+TN)/(TP+FP+FN+TN)
- Misclassification Rate=(FP+FN)/(TP+FP+FN+TN)

Sensitivity and Specificity

- Sensitivity : Percentage of positives that are successfully classified as positive
- Specificity : Percentage of negatives that are successfully classified as negatives

Predicted Classes

Actual Classes

	0(Positive)	1(Negative)	
0(Positive)	True positive (TP) Actual condition is Positive, it is truly predicted as positive	False Negatives(FN) Actual condition is Positive, it is falsely predicted as negative	Sensitivity= TP/(TP+FN) or TP/ Overall Positives
1(Negative)	False Positives(FP) Actual condition is Negative, it is falsely predicted as positive	True Negatives(TN) Actual condition is Negative, it is truly predicted as negative	Specificity = TN/(TN+FP) or TN/ Overall Negatives

statinfer

Calculating Sensitivity and Specificity

LAB - Sensitivity and Specificity

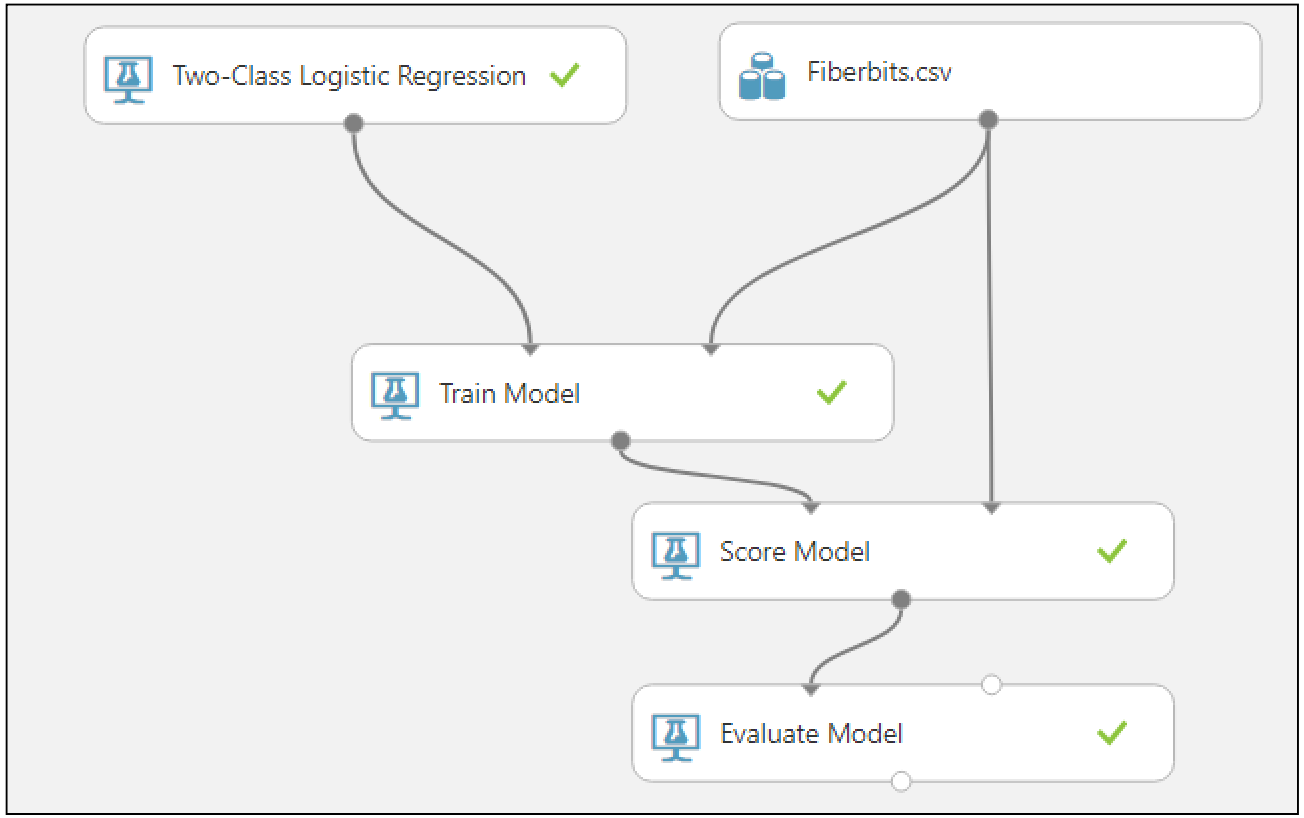
- Build a logistic regression model on fiber bits data
- Create the confusion matrix
- Find the accuracy
- Calculate Specificity
- Calculate Sensitivity

Steps - Sensitivity and Specificity

- Drag and drop the Dataset into the canvas
- Drag and drop **Two-Class Logistic Regression, Train Model, Score Model** and **Evaluate Model**
- Connect **Two-Class Boosted Logistic Regression** to the first input of **Train Model** and **Dataset** to the Second input of **Train Model**
- Connect the output of **Train Model** first input of **Score Model** and **Dataset** to the Second input of **Score Model**
- Connect the output of **Score Model** to the input of **Evaluate Model**
- Click on Train Model and select the column for which the prediction is done(active_cust)
- Click run and visualize the output of Evaluate Model

Steps - Sensitivity and Specificity

Fig1: Logistic Regression (Fiberbits.csv)



Steps - Sensitivity and Specificity

Fig2: Sensitivity and Specificity(Threshold=0.5)

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
48268	9591	0.770	0.782	0.5	0.830
False Positive	**True Negative**	**Recall**	**F1 Score**		
13458	28683	0.834	0.807		
Positive Label	**Negative Label**				
1	0				

Fig3: Sensitivity and Specificity(Threshold=0.8)

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
22443	35416	0.607	0.854	0.8	0.830
False Positive	**True Negative**	**Recall**	**F1 Score**		
3847	38294	0.388	0.533		
Positive Label	**Negative Label**				

Fig4: Sensitivity and Specificity(Threshold=0.3)

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
56849	1010	0.694	0.658	0.3	0.830
False Positive	**True Negative**	**Recall**	**F1 Score**		
29583	12558	0.983	0.788		
Positive Label	**Negative Label**				
1	0				

Sensitivity vs Specificity

Sensitivity and Specificity

- By changing the threshold, the good and bad customers classification will be changed hence the sensitivity and specificity will be changed
- Which one of these two we should maximize? What should be ideal threshold?
- Ideally we want to maximize both Sensitivity & Specificity. But this is not possible always. There is always a tradeoff.
- Sometimes we want to be 100% sure on Predicted negatives, sometimes we want to be 100% sure on Predicted positives.
- Sometimes we simply don’t want to compromise on sensitivity sometimes we don’t want to compromise on specificity
- The threshold is set based on business problem

When Sensitivity is a high priority

When Sensitivity is a high priority

- Predicting a bad customers or defaulters before issuing the loan

		Predicted Classes		
		0(Yes-Defaulter)	1(Non-Defaulter)	
Actual Classes	0(Yes-Defaulter)	True positive (TP) Actual customer is bad and model is predicting them as bad	False Negatives(FN) Actual customer is bad and model is predicting them as good	Sensitivity= TP/(TP+FN) or TP/ Overall Positives
	1(Non-Defaulter)	False Positives(FP) Actual customer is good and model is predicting them as bad	True Negatives(TN) Actual customer is good and model is predicting them as good	Specificity = TN/(TN+FP) or TN/ Overall Negatives

When Sensitivity is a high priority

- Predicting a bad defaulters before issuing the loan

		Predicted Classes		
		0(Yes-Defaulter)	1(Non-Defaulter)	
Actual Classes	0(Yes-Defaulter)	True positive (TP) Actual customer is bad and model is predicting them as bad. **Rejected a Loan of 100,000**	False Negatives(FN) Actual customer is bad and model is predicting them as good **Issued a loan of 100,00**	Sensitivity= TP/(TP+FN) or TP/ Overall Positives
	1(Non-Defaulter)	False Positives(FP) Actual customer is good and model is predicting them as bad. **Rejected a Loan of 100,000**	True Negatives(TN) Actual customer is good and model is predicting them as good. **Issued a loan of 100,00**	Specificity = TN/(TN+FP) or TN/ Overall Negatives

When Sensitivity is a high priority

- The profit on good customer loan is not equal to the loss on one bad customer loan
- The loss on one bad loan might eat up the profit on 100 good customers
- In this case one bad customer is not equal to one good customer.
- If p is probability of default then we would like to set our threshold in such a way that we don't miss any of the bad customers.
- We set the threshold in such a way that Sensitivity is high
- We can compromise on specificity here. If we wrongly reject a good customer, our loss is very less compared to giving a loan to a bad customer.
- We don't really worry about the good customers here, they are not harmful hence we can have less Specificity

statinfer

When Specificity is a high priority

When Specificity is a high priority

statinfer

- Testing a medicine is good or poisonous

Actual Classes	Predicted Classes: 0(Yes-Good)	Predicted Classes: 1(Poisonous)	
0(Yes-Good)	True positive (TP) Actual medicine is good and model is predicting them as good	False Negatives(FN) Actual medicine is good and model is predicting them as poisonous	Sensitivity= TP/(TP+FN) or TP/ Overall Positives
1(Poisonous)	False Positives(FP) Actual medicine is poisonous and model is predicting them as good	True Negatives(TN) Actual medicine is poisonous and model is predicting them as poisonous	Specificity = TN/(TN+FP) or TN/ Overall Negatives

When Specificity is a high priority

- Testing a medicine is good or poisonous

Actual Classes \ Predicted Classes	0(Yes-Good)	1(Poisonous)	
0(Yes-Good)	True positive (TP) Actual medicine is good and model is predicting them as good. **Recommended for use**	False Negatives(FN) Actual medicine is good and model is predicting them as poisonous. **Banned the usage**	Sensitivity= TP/(TP+FN) or TP/ Overall Positives
1(Poisonous)	False Positives(FP) Actual medicine is poisonous and model is predicting them as good. **Recommended for use**	True Negatives(TN) Actual medicine is poisonous and model is predicting them as poisonous. **Banned the usage**	Specificity = TN/(TN+FP) or TN/ Overall Negatives

When Specificity is a high priority

- In this case, we have to really avoid cases like , Actual medicine is poisonous and model is predicting them as good.
- We can’t take any chance here.
- The specificity need to be near 100.
- The sensitivity can be compromised here. It is not very harmful not to use a good medicine when compared with vice versa case

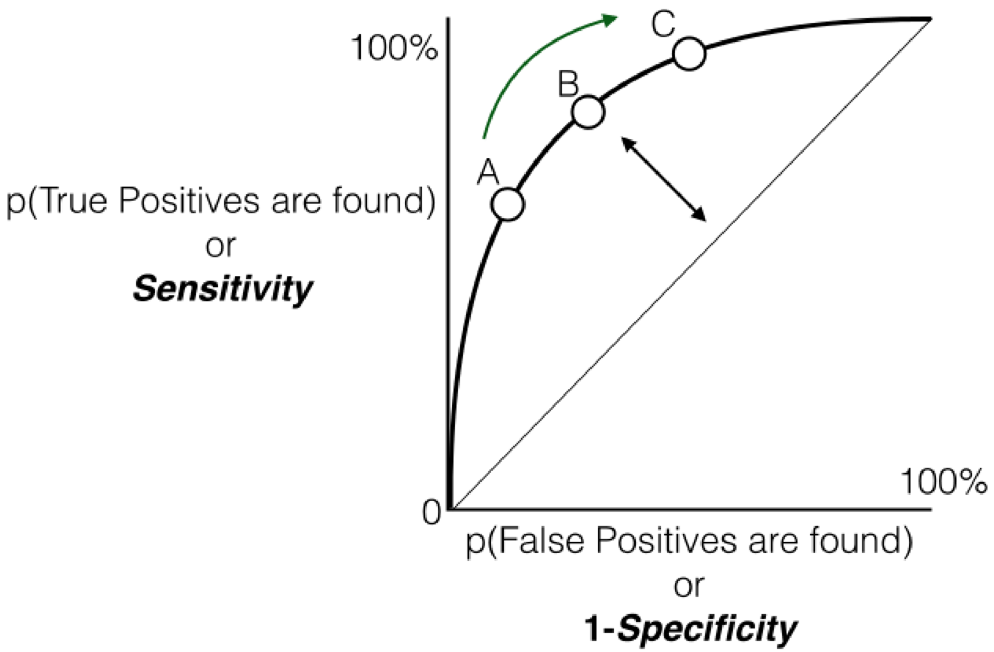
Sensitivity vs Specificity - Importance

- There are some cases where Sensitivity is important and need to be near to 1
- There are business cases where Specificity is important and need to be near to 1
- We need to understand the business problem and decide the importance of Sensitivity and Specificity

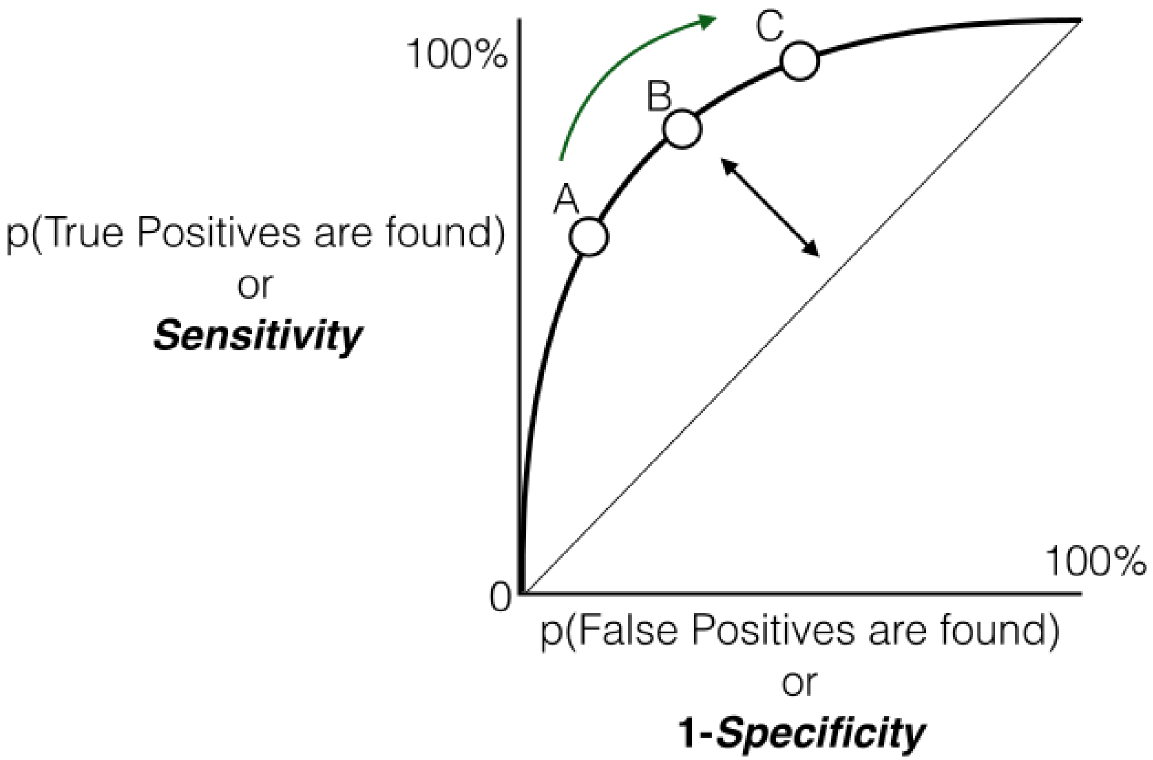
ROC Curve

ROC Curve

- If we consider all the possible threshold values and the corresponding specificity and sensitivity rate what will be the final model accuracy.
- ROC(Receiver operating characteristic) curve is drawn by taking False positive rate on X-axis and True positive rate on Y- axis
- ROC tells us, how many mistakes are we making to identify all the positives?



ROC Curve - Interpretation

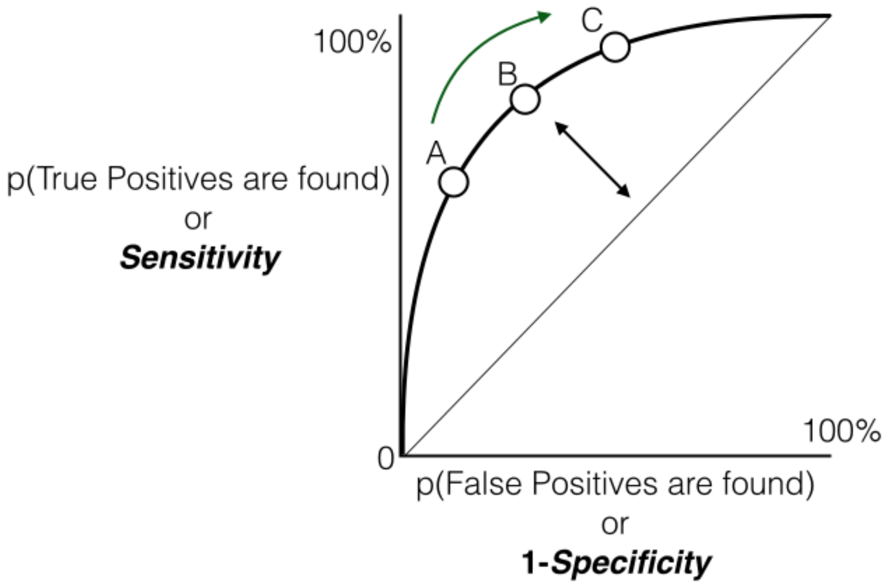


- How many mistakes are we making to identify all the positives?
- How many mistakes are we making to identify 70%, 80% and 90% of positives?
- 1-Specificty(false positive rate) gives us an idea on mistakes that we are making
- We would like to make 0% mistakes for identifying 100% positives
- We would like to make very minimal mistakes for identifying maximum positives
- We want that curve to be far away from straight line
- Ideally we want the area under the curve as high as possible

AUC

ROC and AUC

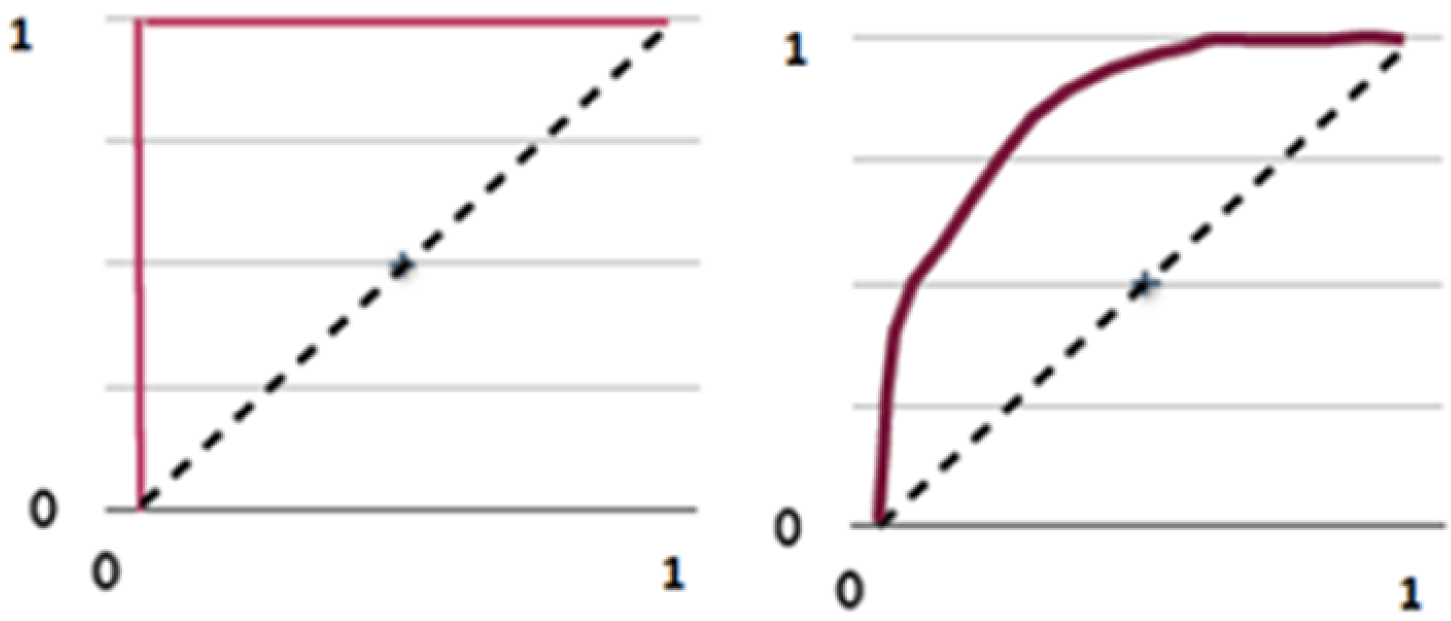
- We want that curve to be far away from straight line. Ideally we want the area under the curve as high as possible
- ROC comes with a connected topic, AUC. Area Under
- ROC Curve Gives us an idea on the performance of the model under all possible values of threshold.
- We want to make almost 0% mistakes while identifying all the positives, which means we want to see AUC value near to 1



AUC

- AUC is near to 1 for a good model

AUC=1 AUC=0.82



statinfer

ROC and AUC Calculation

LAB: ROC and AUC

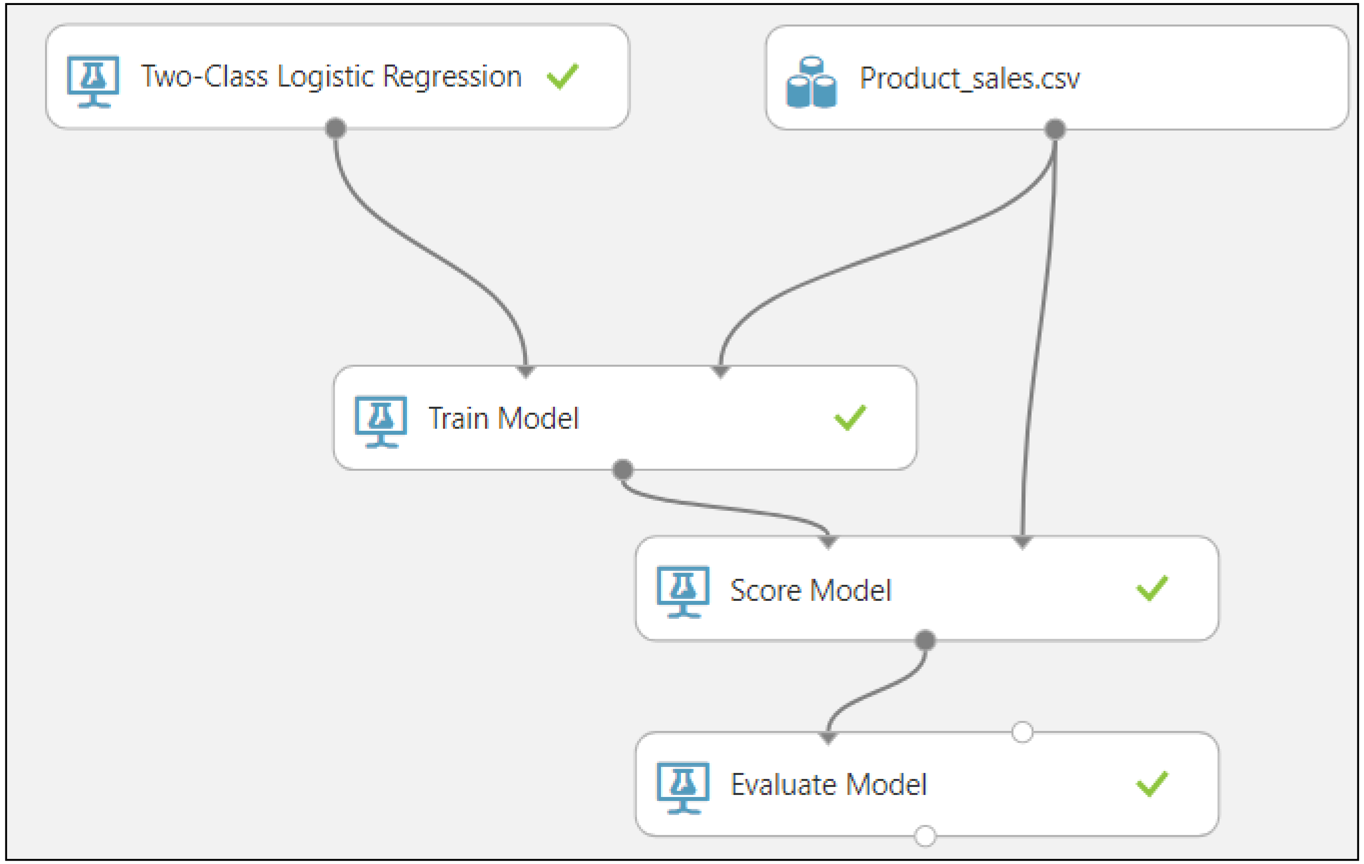
- Calculate ROC and AUC for Product Sales Data/Product_sales.csv logistic regression model
- Calculate ROC and AUC for fiber bits logistic regression model

Steps - ROC and AUC

- Drag and drop the Dataset into the canvas(Product_sales.csv)
- Drag and drop **Two-Class Logistic Regression, Train Model, Score Model** and **Evaluate Model**
- Connect **Two-Class Boosted Decision Tree** to the first input of **Train Model** and **Dataset** to the Second input of **Train Model**
- Connect the output of **Train Model** first input of **Score Model** and **Dataset** to the Second input of **Score Model**
- Connect the output of **Score Model** to the input of **Evaluate Model**
- Click on Train Model and select the column for which the prediction is done(active_cust)
- Click run and visualize the output of Evaluate Model for ROC and AUC
- Follow the same steps for the Fiberbits data

Steps - ROC and AUC

Fig5: Logistic Regression(Product_sales.csv)



Steps - ROC and AUC

Fig6: ROC(Product_sales.csv)

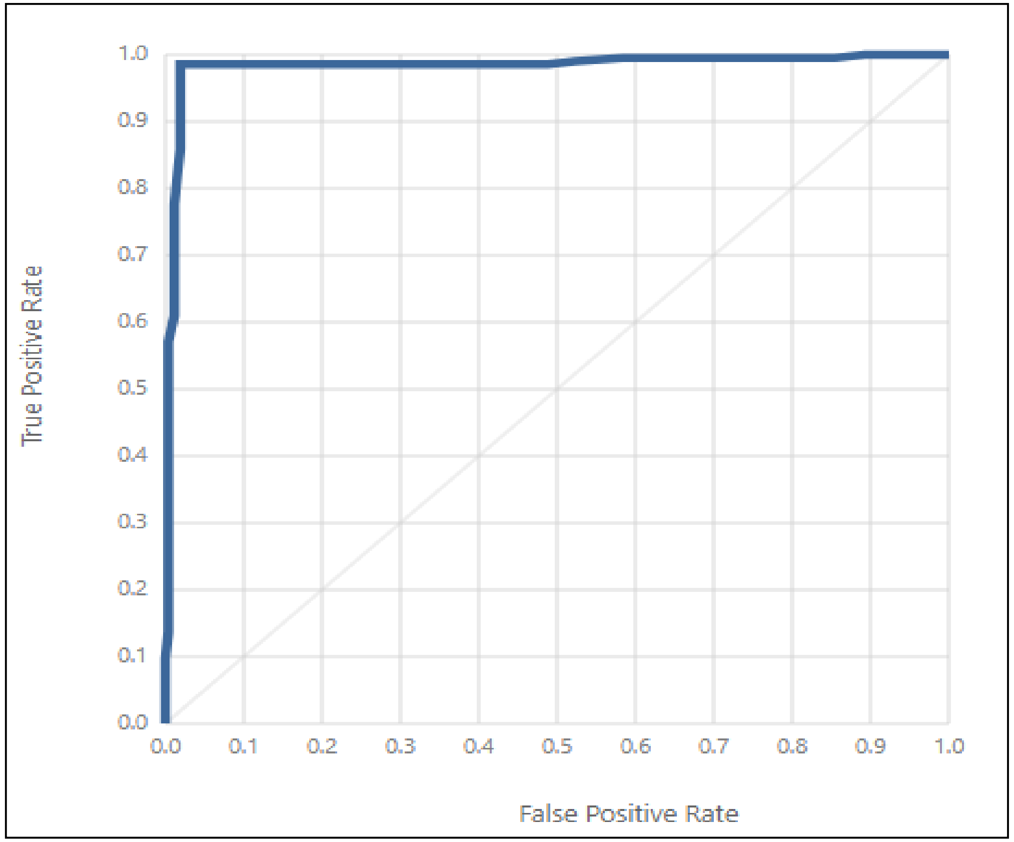


Fig7: AUC(Product_sales.csv)

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
202	3	0.983	0.976	0.5	0.983
False Positive	**True Negative**	**Recall**	**F1 Score**		
5	257	0.985	0.981		
Positive Label	**Negative Label**				
1	0				

Steps - ROC and AUC

Fig8: ROC(Fiberbits.csv)

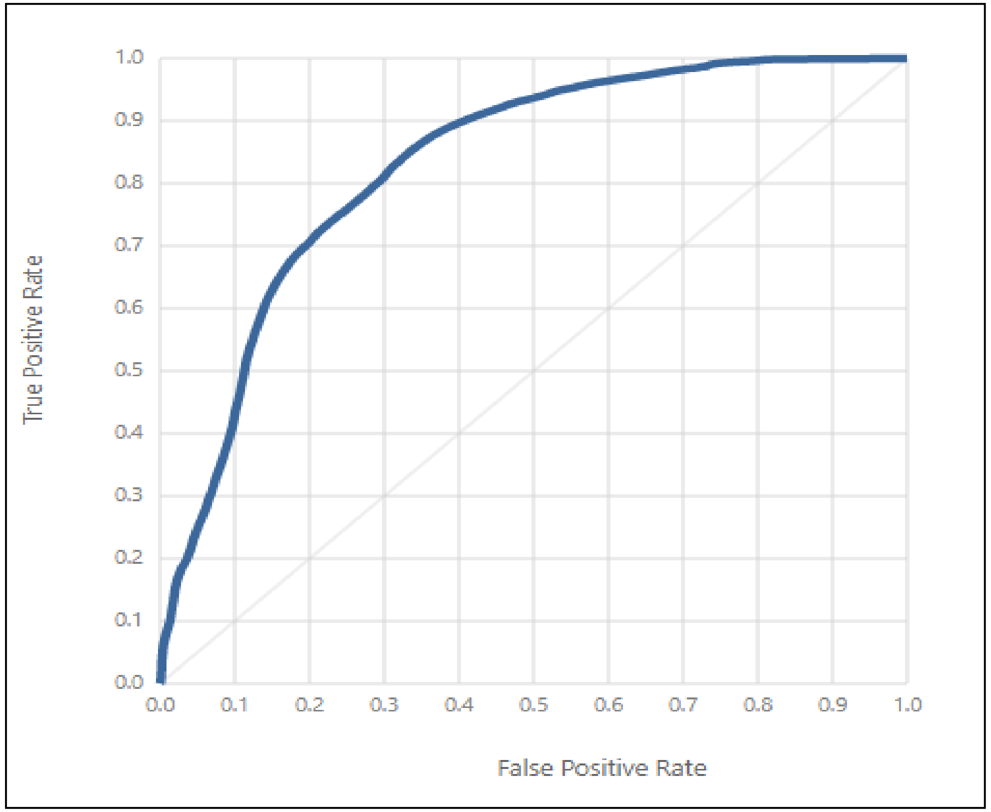


Fig9: AUC(Fiberbits.csv)

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
48268	9591	0.770	0.782	0.5	0.830
False Positive	**True Negative**	**Recall**	**F1 Score**		
13458	28683	0.834	0.807		
Positive Label	**Negative Label**				
1	0				

The best model

What is a best model? How to build?

- A model with maximum accuracy /least error
- A model that uses maximum information available in the given data
- A model that has minimum squared error
- A model that captures all the hidden patterns in the data
- A model that produces the best perdition results

Model Selection

- How to build/choose a best model?
- Error on the training data is not a good meter of performance on future data
- How to select the best model out of the set of available models ?
- Are there any methods/metrics to choose best model?
- What is training error? What is testing error? What is hold out sample error?

LAB: The most accurate model

LAB: The most accurate model

- Data: Fiberbits/Fiberbits.csv
- Build a decision tree to predict active_user
- What is the accuracy of your model?
- Grow the tree as much as you can and achieve 95% accuracy.

Steps - The most accurate model

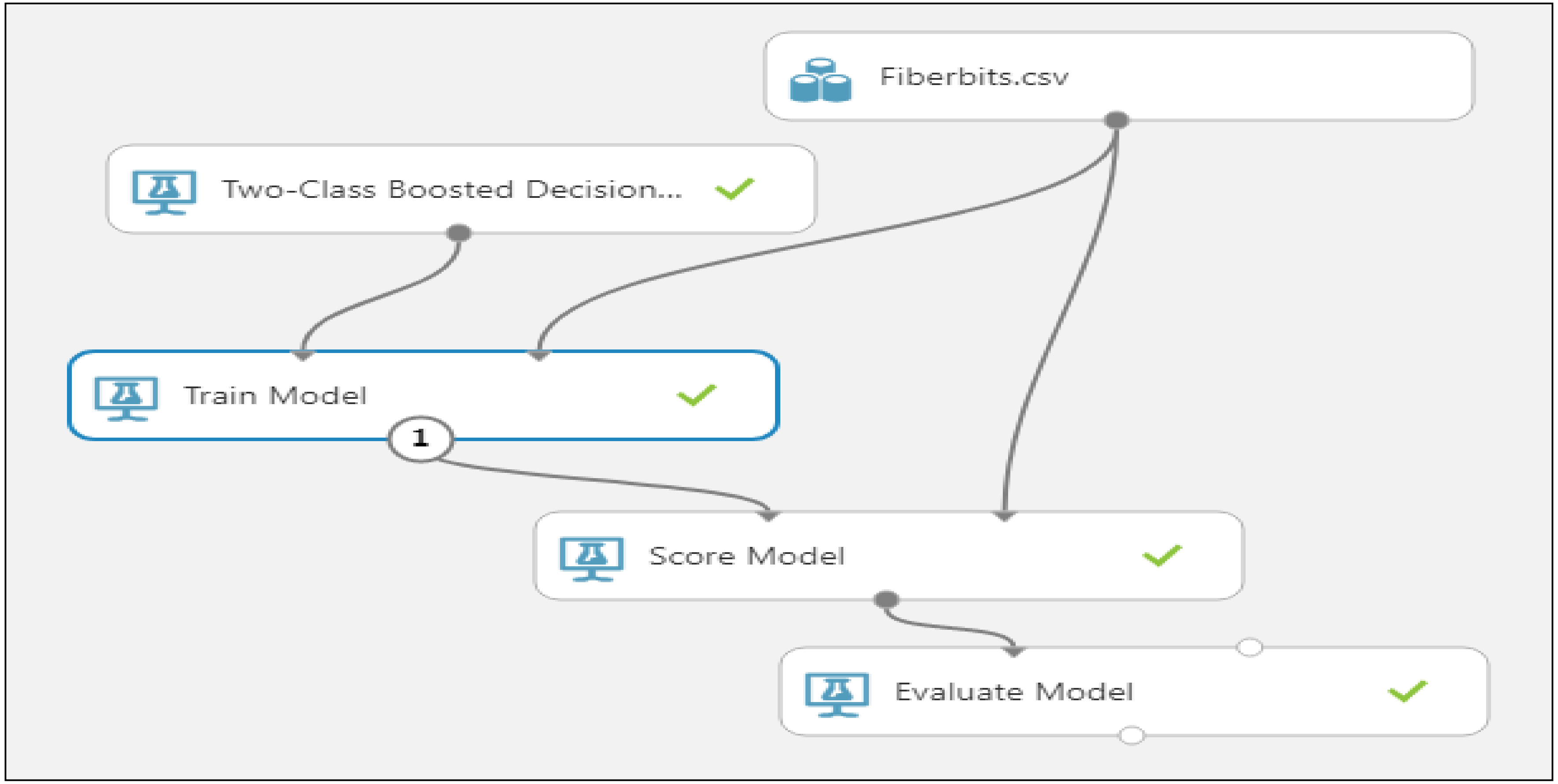
- Building Decision Tree with FiberBits Data :
  - Drag and drop the Dataset into the canvas
  - Drag and drop **Two-Class Boosted Decision Tree, Train Model, Score Model** and **Evaluate Model**
  - Connect **Two-Class Boosted Decision Tree** to the first input of **Train Model** and **Dataset** to the Second input of **Train Model**
  - Connect the output of **Train Model** first input of **Score Model** and **Dataset** to the Second input of **Score Model**
  - Connect the output of **Score Model** to the input of **Evaluate Model**

Steps - The most accurate model

- Click on **Two-Class Boosted Decision Tree** and select the following:
  - Create trainer mode → Single Parameter
  - Maximum number of leaves per tree → 8
  - Minimum number of samples per leaf node → 30
  - Learning rate → 0.09
  - Number of trees constructed → 1
- Click on Train Model and select the column for which the prediction is done(active_cust)
- Click run and visualize the output of Train Model and Evaluate Model

Steps - The most accurate model

Fig10: Decision Tree Modal(FiberBits)



Steps - The most accurate model

Fig11: Properties(**Two-Class Boosted Decision Tree)**

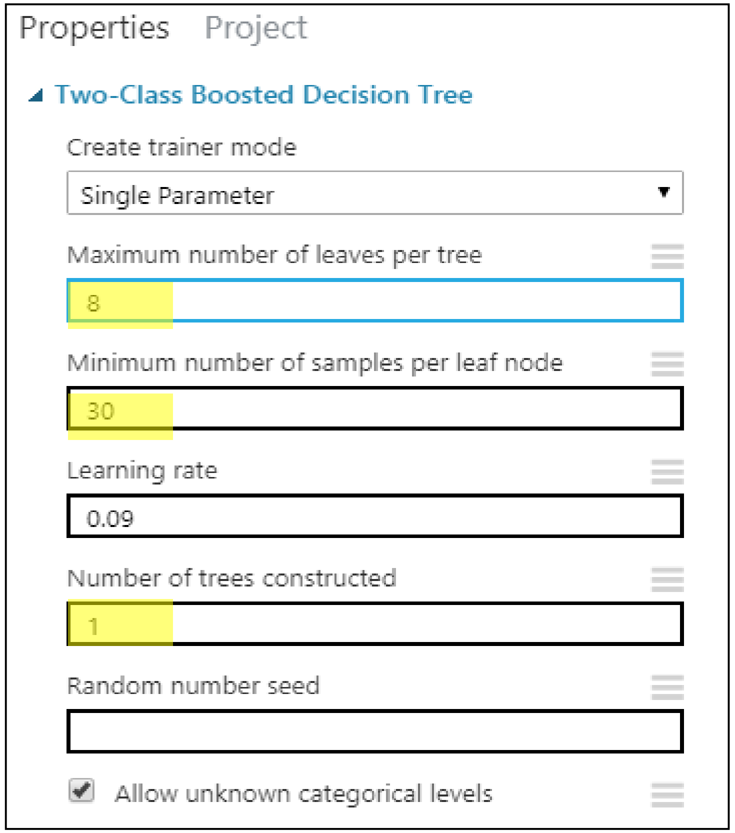
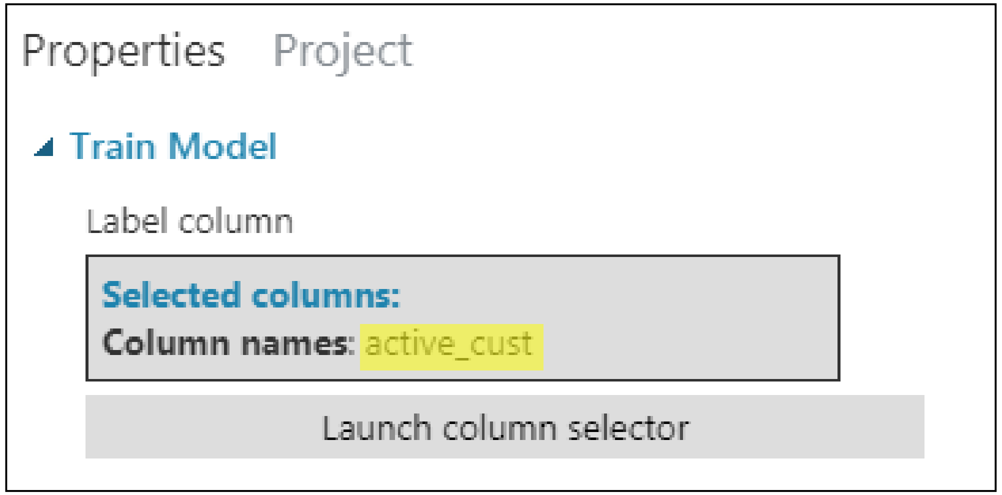


Fig12: Properties(Train Model)



statinfer

Steps - The most accurate model

Fig13: Accuracy(active_cust)

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
49815	8044	0.826	0.842	0.5	0.859
False Positive	True Negative	Recall	F1 Score		
9344	32797	0.861	0.851		
Positive Label	Negative Label				
1	0				

Steps - The most accurate model

- To achieve 95% accuracy :
  - Click on **Two-Class Boosted Decision Tree** and select the following:
    - Create trainer mode → Single Parameter
    - Maximum number of leaves per tree → 5750
    - Minimum number of samples per leaf node → 1
    - Learning rate → 0.09
    - Number of trees constructed → 1
  - Click run and visualize the output of Train Model and Evaluate Model

Steps - The most accurate model

Fig14: Properties(**Two-Class Boosted Decision Tree**)

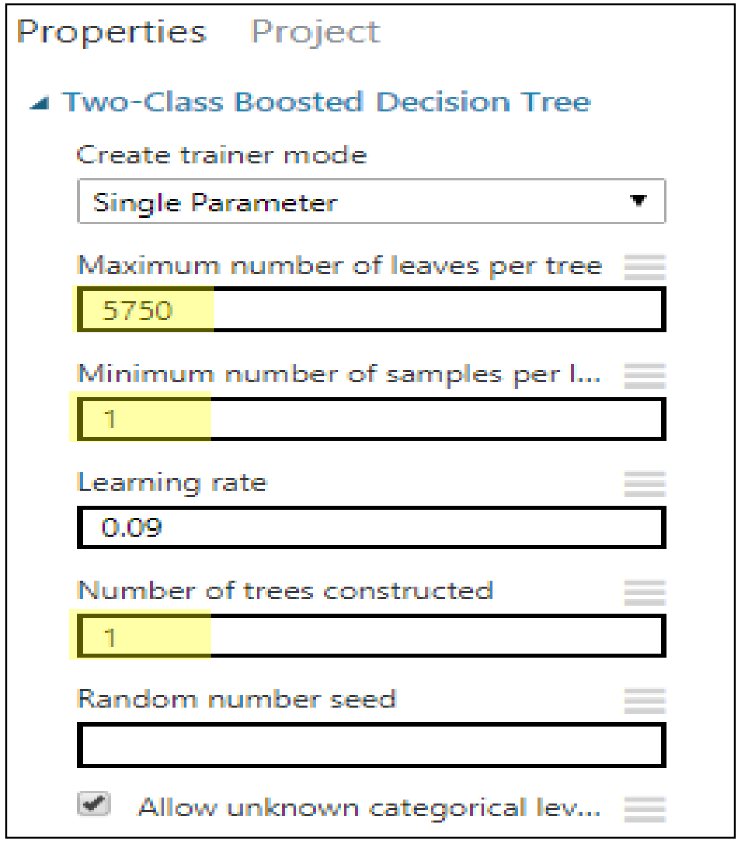
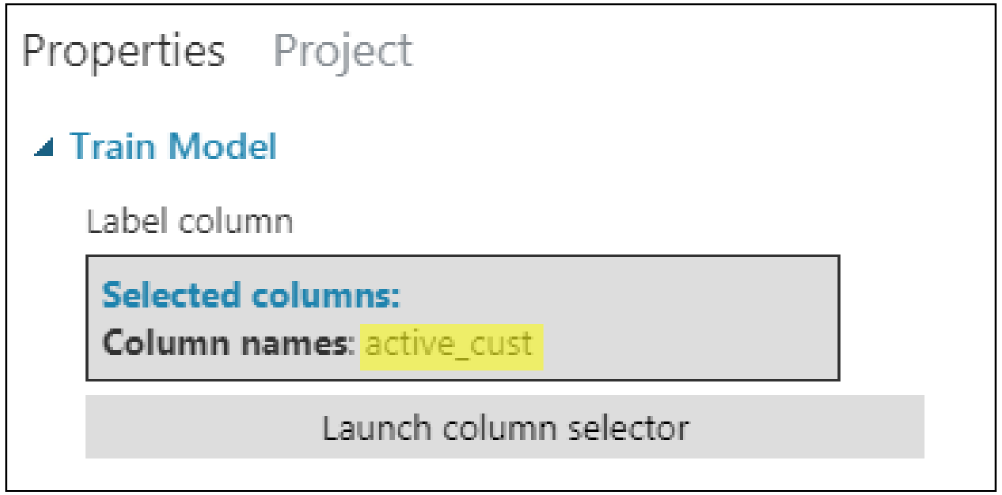


Fig15: Properties(Train Model)



Steps - The most accurate model

Fig16: Accuracy(active_cust)

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
56052	1807	0.950	0.946	0.5	0.982
False Positive	True Negative	Recall	F1 Score		
3191	38950	0.969	0.957		
Positive Label	Negative Label				
1	0				

statinfer

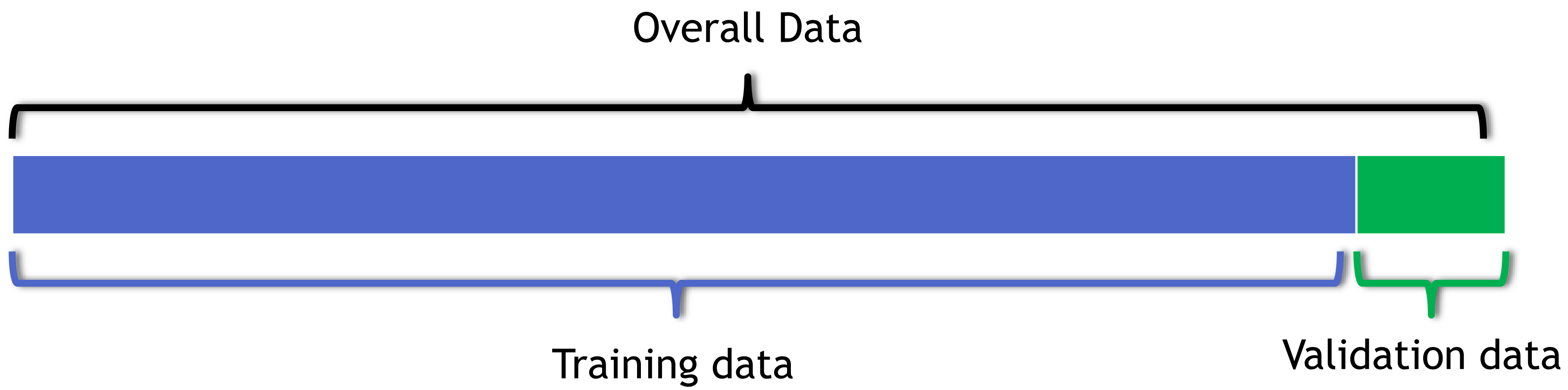
Different type of datasets and errors

The Training Error

- The accuracy of our best model is 95%. Is the 5% error model really good?
- The error on the training data is known as training error.
- A low error rate on training data may not always mean the model is good.
- What really matters is how the model is going to perform on unknown data or test data.
- We need to find out a way to get an idea on error rate of test data.
- We may have to keep aside a part of the data and use it for validation.
- There are two types of datasets and two types of errors

Two types of datasets

- There are two types of datasets
  - **Training set:** This is used in model building. The input data
  - **Test set:** The unknown dataset. This dataset is gives the accuracy of the final model
- We may not have access to these two datasets for all machine learning problems. In some cases, we can take 90% of the available data and use it as training data and rest 10% can be treated as validation data
    - **Validation set:** This dataset kept aside for model validation and selection. This is a temporary subsite to test dataset. It is not third type of data
- We create the validation data with the hope that the error rate on validation data will give us some basic idea on the test error



Types of errors

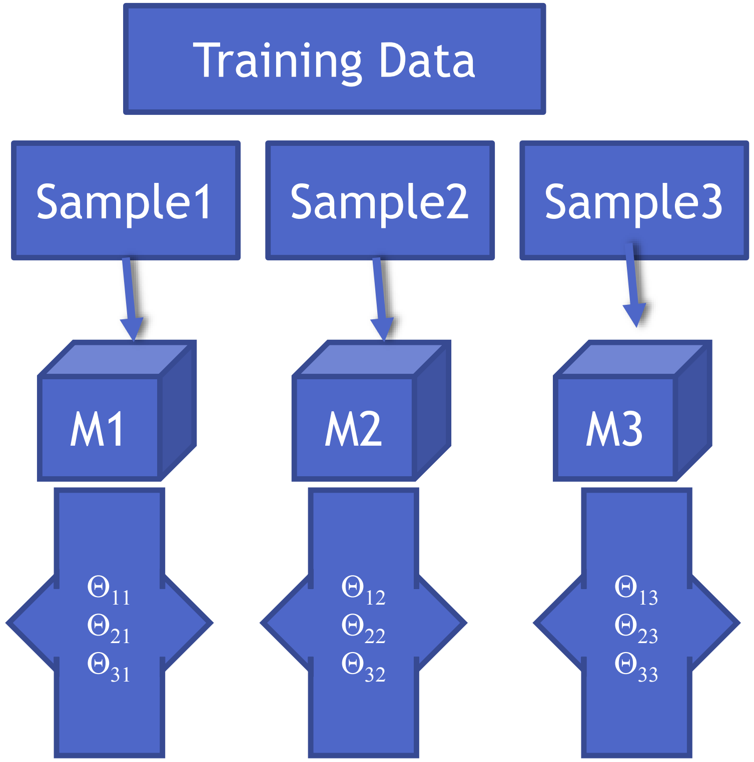
- The training error
  - The error on training dataset
  - In-time error
  - Error on the known data
  - Can be reduced while building the model
- The test error
  - The error that matters
  - Out-of-time error
  - The error on unknown/new dataset.

"A good model will have both training and test error very near to each other and close to zero"

The problem of over fitting

The problem of over fitting

- In search of the best model on the given data we add many predictors, polynomial terms, Interaction terms, variable transformations, derived variables, indicator/dummy variables etc.,
- Most of the times we succeed in reducing the error. What error is this?
- So by complicating the model we fit the best model for the training data.
- Sometimes the error on the training data can reduce to near zero
- But the same best model on training data fails miserably on test data.
- Imagine building multiple models with small changes in training data. The resultant set of models will have huge variance in their parameter estimates.



The problem of over fitting

- The model is made really complicated, that it is very sensitive to minimal changes
- By complicating the model the variance of the parameters estimates inflates
- Model tries to fit the irrelevant characteristics in the data
- Over fitting
  - The model is super good on training data but not so good on test data
  - We fit the model for the noise in the data
  - Less training error, high testing error
  - The model is over complicated with too many predictors
  - Model need to be simplified
  - A model with lot of **variance**

LAB: Model with huge Variance

LAB: Model with huge Variance

- Data: Fiberbits/Fiberbits.csv
- Take initial 90% of the data. Consider it as training data. Keep the final 10% of the records for validation.
- Build the best model(5% error) model on training data.
- Use the validation data to verify the error rate. Is the error rate on the training data and validation data same?

Steps - Model with huge Variance

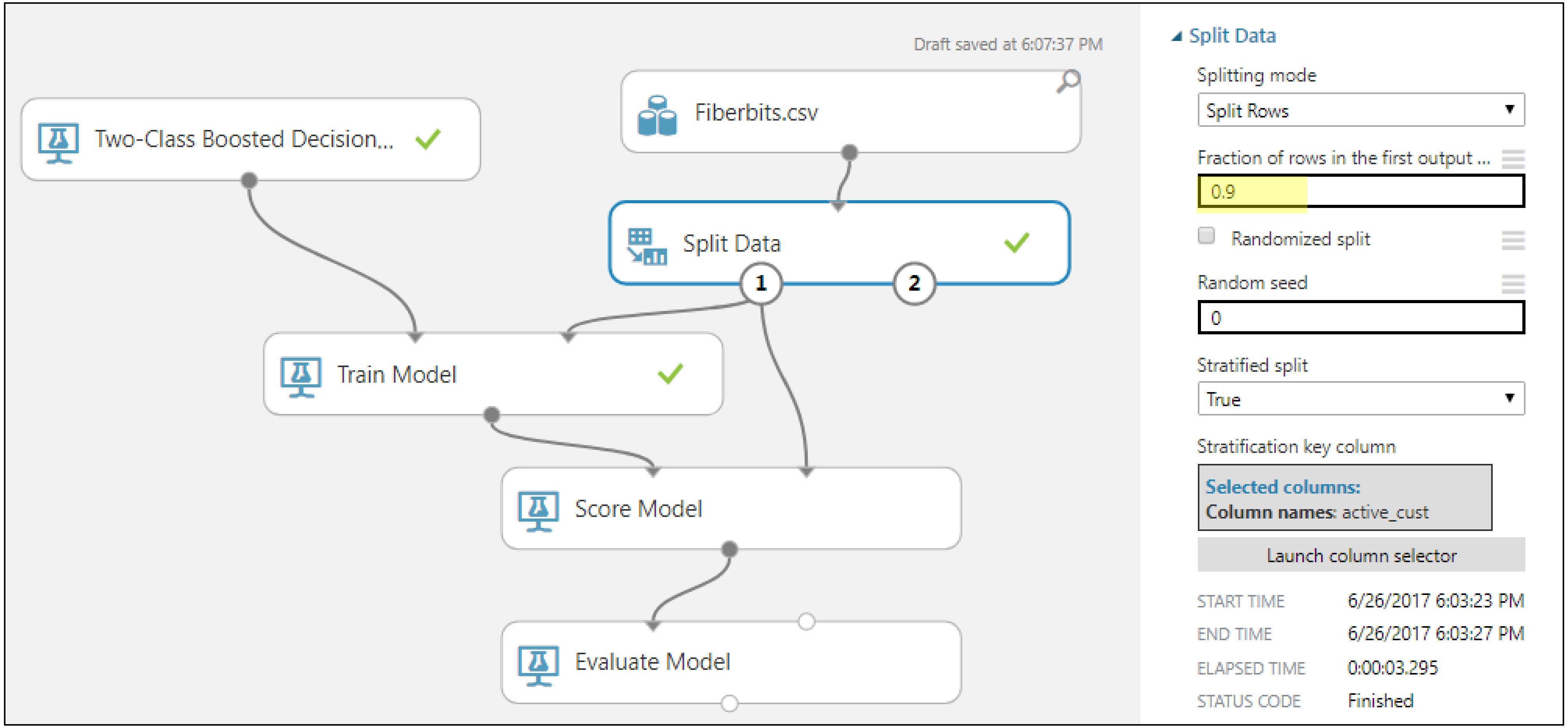
- Building Decision Tree with FiberBits Data :
  - Drag and drop the Dataset into the canvas
  - Drag and drop Split Data, connect it to the dataset
    - Select the properties:
      - Splitting mode → Split Rows
      - Fraction of Rows → 0.9
      - Uncheck Randomized Split
  - Drag and drop **Two-Class Boosted Decision Tree, Train Model, Score Model** and **Evaluate Model**
  - Connect **Two-Class Boosted Decision Tree** to the first input of **Train Model** and **Training Data** to the Second input of **Train Model**
  - Connect the output of **Train Model** first input of **Score Model** and **Training Data** to the Second input of **Score Model**
  - Connect the output of **Score Model** to the input of **Evaluate Model**

Steps - Model with huge Variance

- Click on **Two-Class Boosted Decision Tree** and select the following:
  - Create trainer mode → Single Parameter
  - Maximum number of leaves per tree → 5750
  - Minimum number of samples per leaf node → 1
  - Learning rate → 0.09
  - Number of trees constructed → 1
- Click on Train Model and select the column for which the prediction is done(active_cust)
- Click run and visualize the output of Evaluate Model
- Repeat the same by passing Test Data(Second output of Split Data) to the score model

Steps - Model with huge Variance

Fig17: Decision Tree Modal with Training data



Steps - Model with huge Variance

Fig18: Properties(**Two-Class Boosted Decision Tree)**

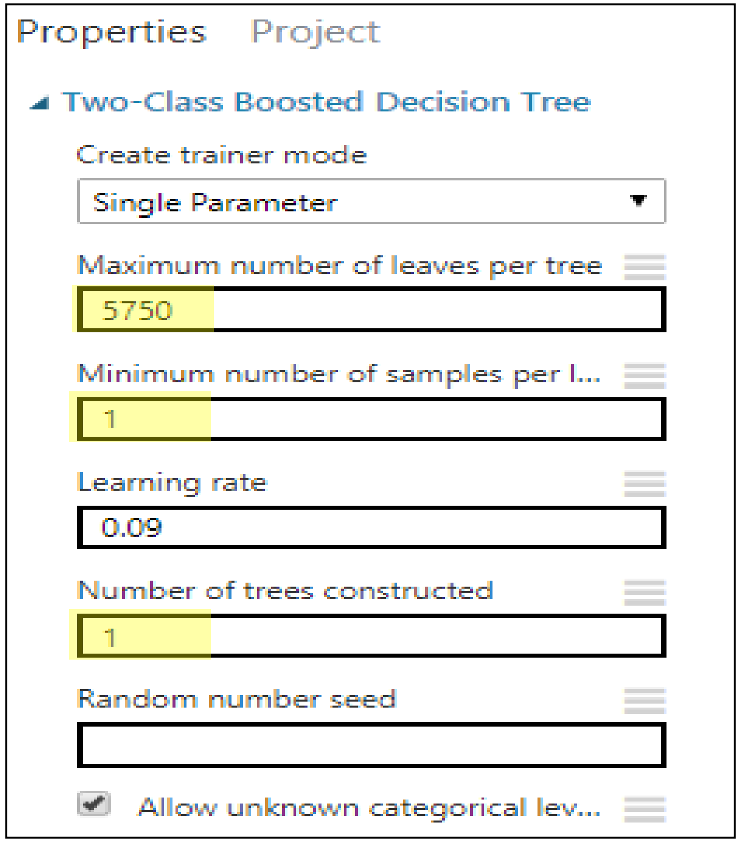
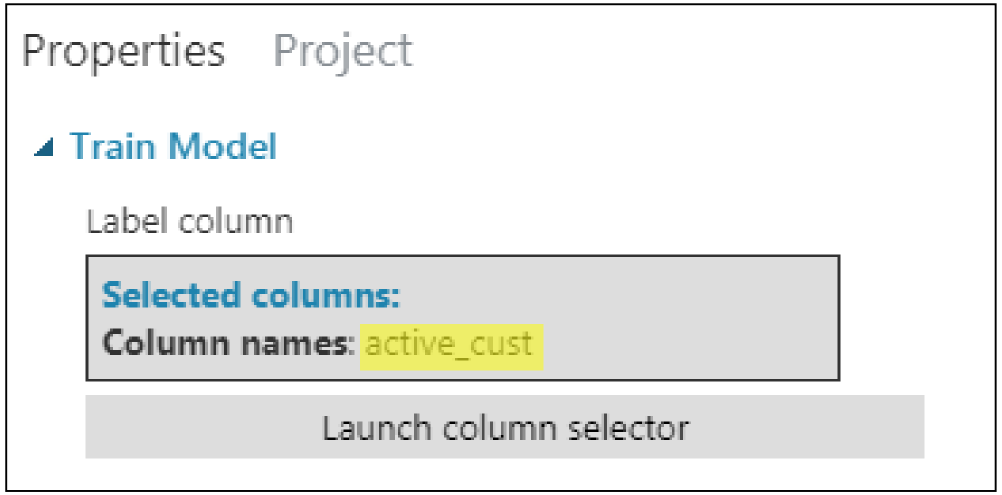


Fig19: Properties(Train Model)



Steps - Model with huge Variance

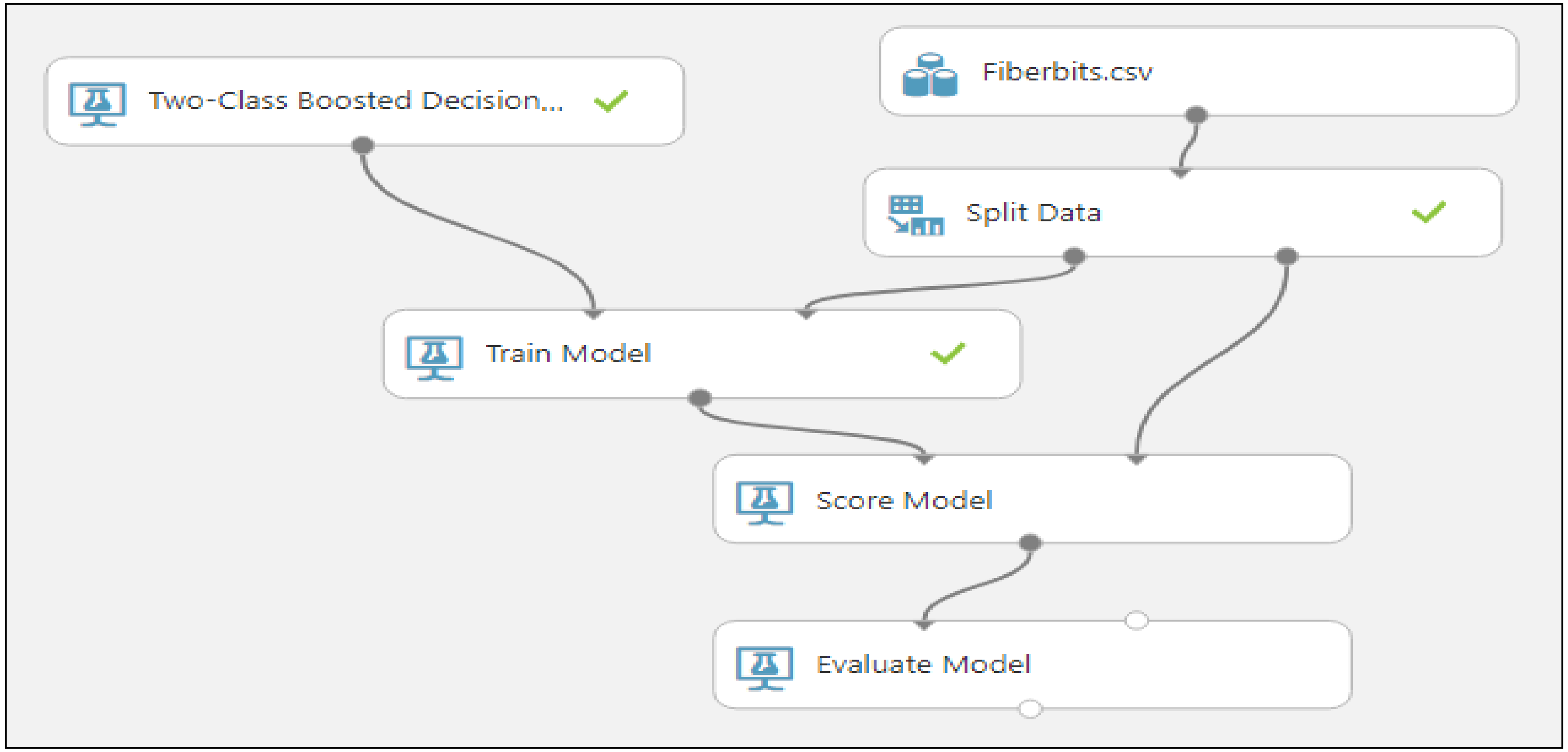
Fig20: Accuracy(Training Data)

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
50773	1300	0.957	0.951	0.5	0.984
False Positive	**True Negative**	**Recall**	**F1 Score**		
2613	35314	0.975	0.963		
Positive Label	**Negative Label**				
1	0				

statinfer

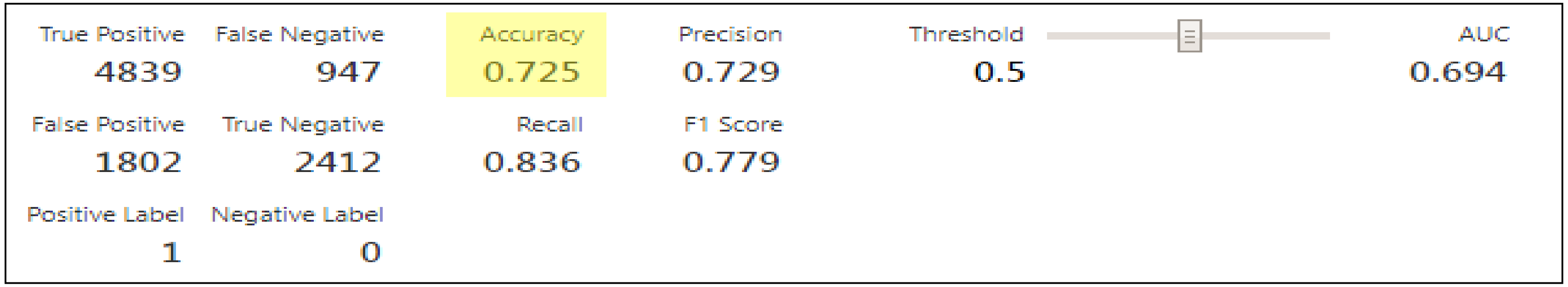
Steps - Model with huge Variance

Fig21: Decision Tree validation Test data



Steps - Model with huge Variance

Fig22: Accuracy(Test Data)



statinfer

The problem of under fitting

The problem of under-fitting

- Simple models are better. Its true but is that always true? May not be always true.
- We might have given it up too early. Did we really capture all the information?
- Did we do enough research and future reengineering to fit the best model? Is it the best model that can be fit on this data?
- By being over cautious about variance in the parameters, we might miss out on some patterns in the data.
- Model need to be complicated enough to capture all the information present.

The problem of under-fitting

- If the training error itself is high, how can we be so sure about the model performance on unknown data?
- Most of the accuracy and error measuring statistics give us a clear idea on training error, this is one advantage of under fitting, we can identify it confidently.
- Under fitting
  - A model that is too simple
  - A mode with a scope for improvement
  - A model with lot of **bias**

LAB: Model with huge Bias

LAB: Model with huge Bias

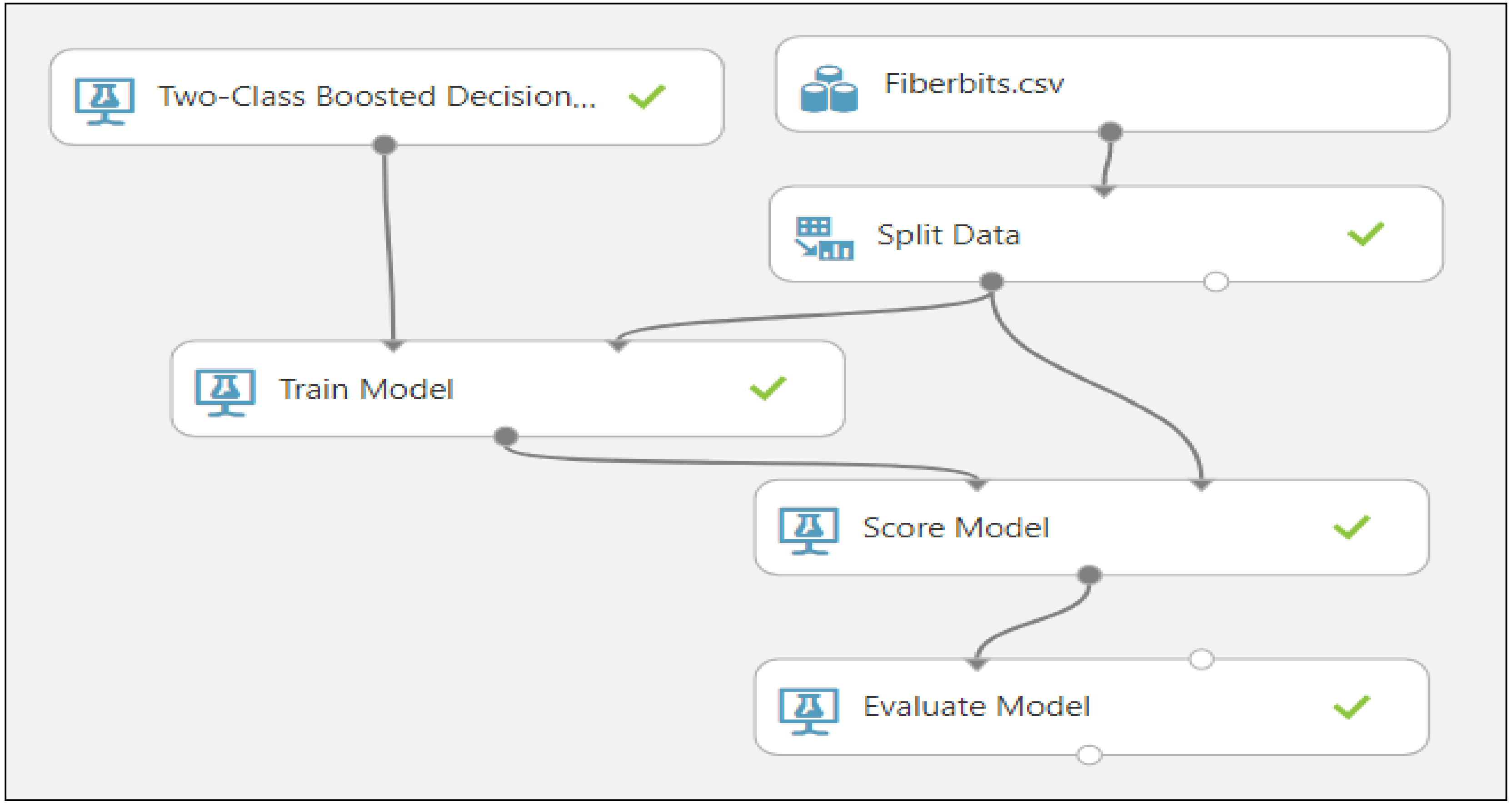
- Lets simplify the model.
- Take the high variance model and prune it.
- Make it as simple as possible.
- Find the training error and validation error.

Steps - Model with huge Bias

- In the previous model change the parameters of **Two-Class Boosted Decision Tree**
  - select the following:
    - Create trainer mode → Single Parameter
    - Maximum number of leaves per tree → 3
    - Minimum number of samples per leaf node → 30
    - Learning rate → 0.09
    - Number of trees constructed → 1
- Click run and visualize the output of Evaluate Mode
- Repeat the same with Test Data

Steps - Model with huge Bias

Fig23: Decision Tree Modal with Training data



Steps - Model with huge Bias

Fig24: Properties(**Two-Class Boosted Decision Tree)**

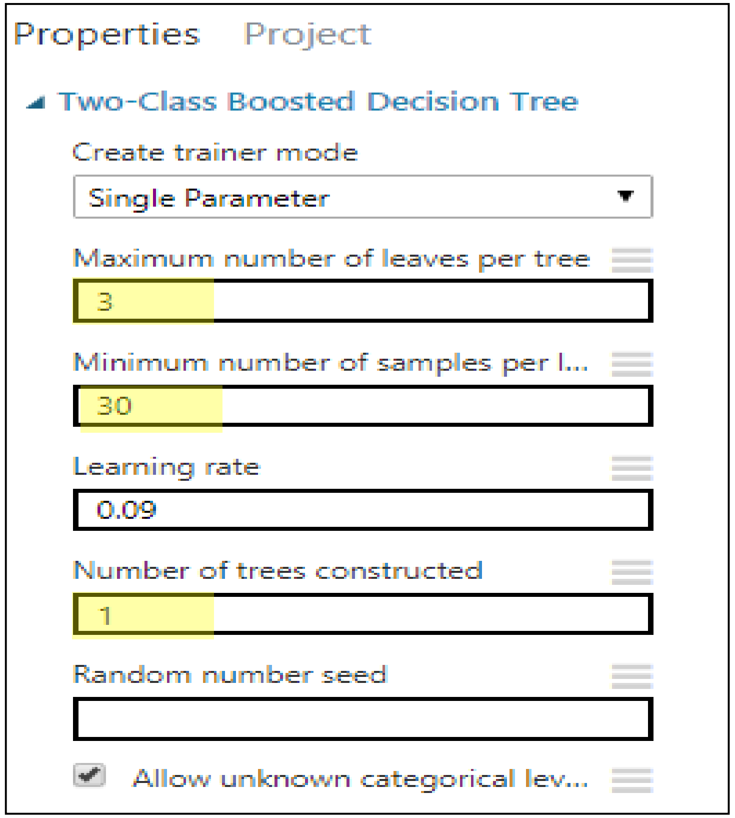
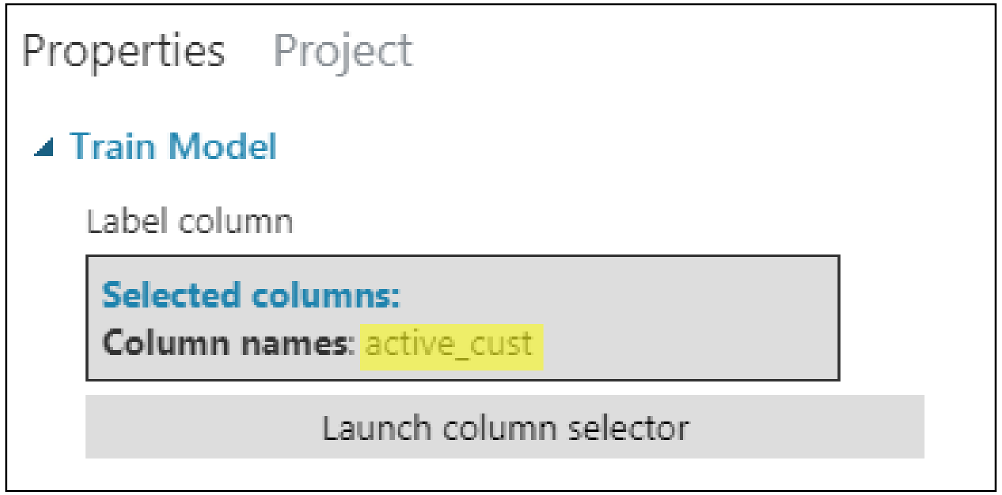
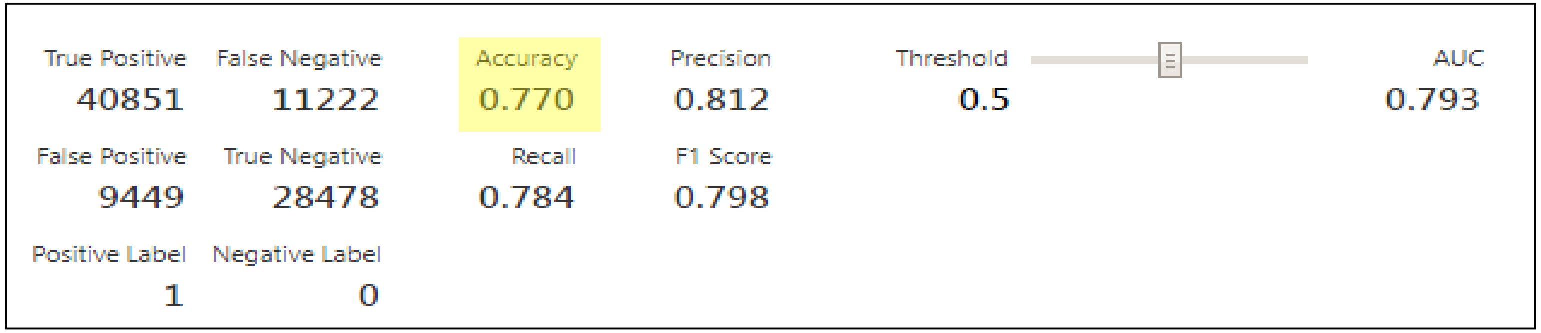


Fig25: Properties(Train Model)



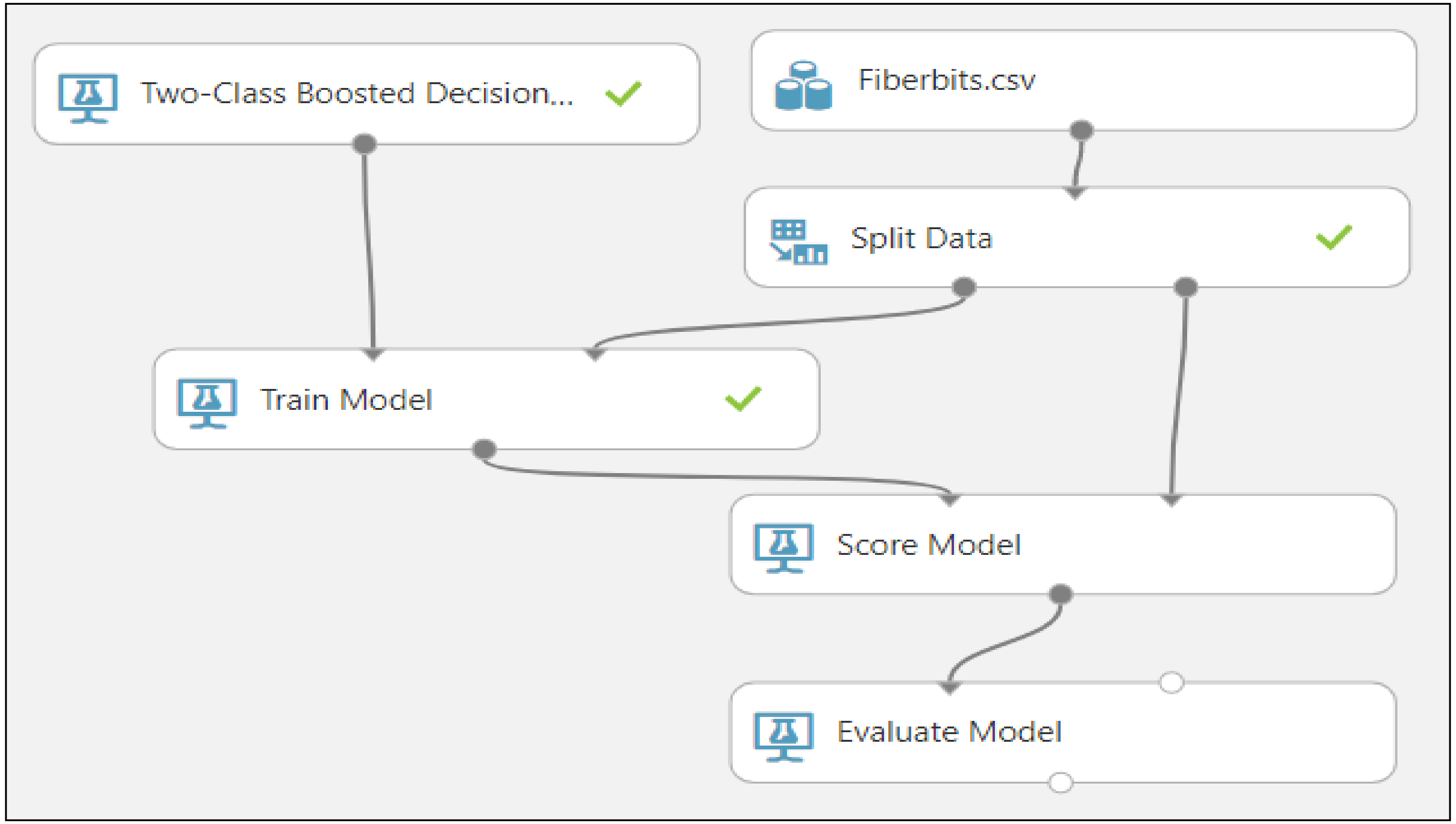
Steps - Model with huge Bias

Fig26: Accuracy(Training Data)



Steps - Model with huge Bias

Fig27: Decision Tree validation Test data



Steps - Model with huge Bias

Fig28: Accuracy(Test Data)

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
5751	35	0.588	0.585	0.5	0.512
False Positive	**True Negative**	**Recall**	**F1 Score**		
4084	130	0.994	0.736		
Positive Label	**Negative Label**				
1	0				

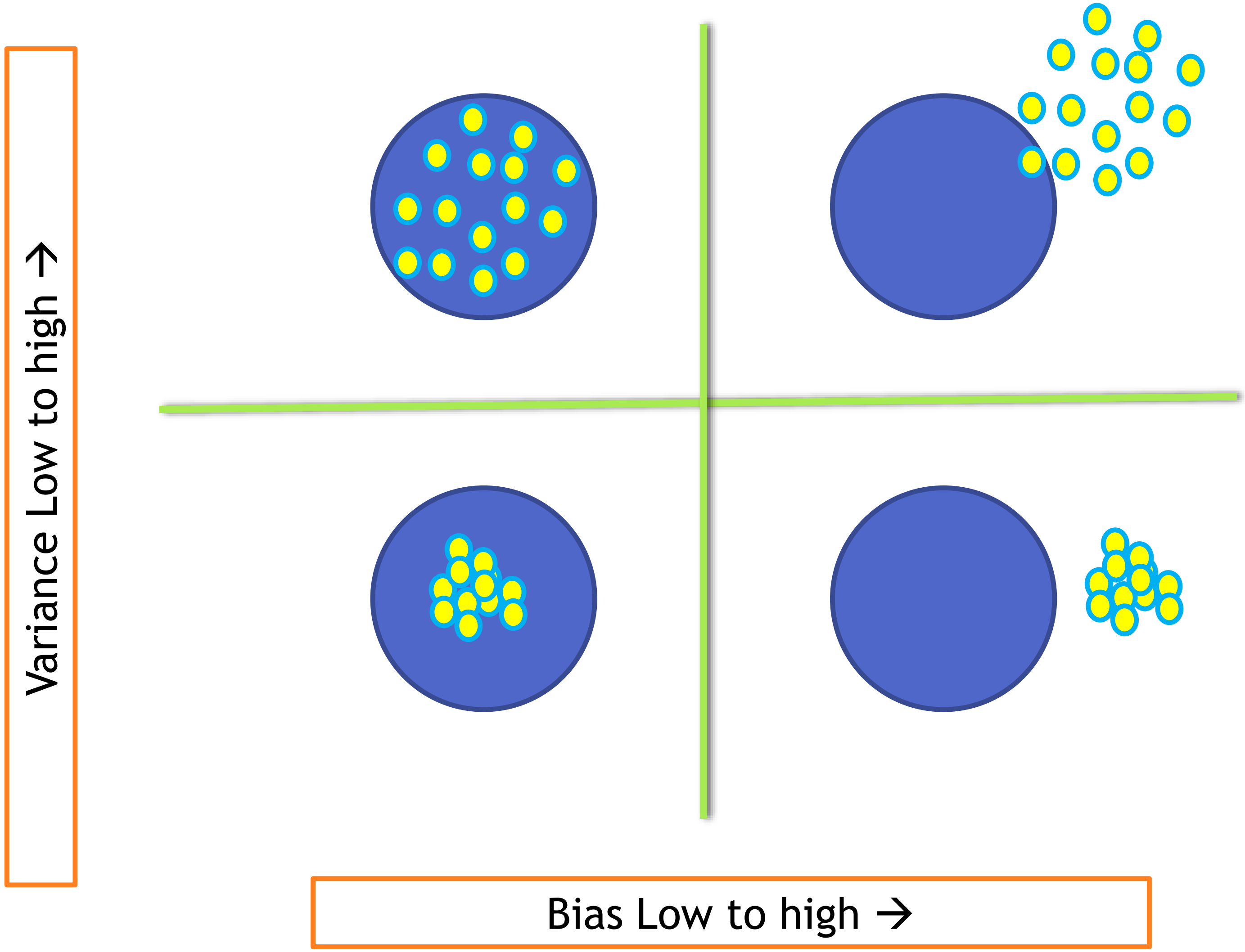
statinfer

Model Bias and Variance

Model Bias and Variance

- Over fitting
  - Low Bias with High Variance
  - Low training error – 'Low Bias'
  - High testing error
  - Unstable model – 'High Variance'
  - The coefficients of the model change with small changes in the data
- Under fitting
  - High Bias with low Variance
  - High training error – 'high Bias'
  - testing error almost equal to training error
  - Stable model – 'Low Variance'
  - The coefficients of the model doesn't change with small changes in the data

Model Bias and Variance



Model aim is to hit the center of circle

The Bias-Variance Decomposition

$$Y = f(X) + \varepsilon$$

$$Var(\varepsilon) = \sigma^2$$

$$
\begin{aligned}
SquaredError &= E[(Y - \hat{f}(x_0))^2 \mid X = x_0] \\
&= \sigma^2 + [E\hat{f}(x_0) - f(x_0)]^2 + E[\hat{f}(x_0) - E\hat{f}(x_0)]^2 \\
&= \sigma^2 + Bias^2(\hat{f}(x_0)) + Var(\hat{f}(x_0))
\end{aligned}
$$

Overall Model Squared Error = Irreducible Error + Bias^2 + Variance

Bias-Variance Decomposition

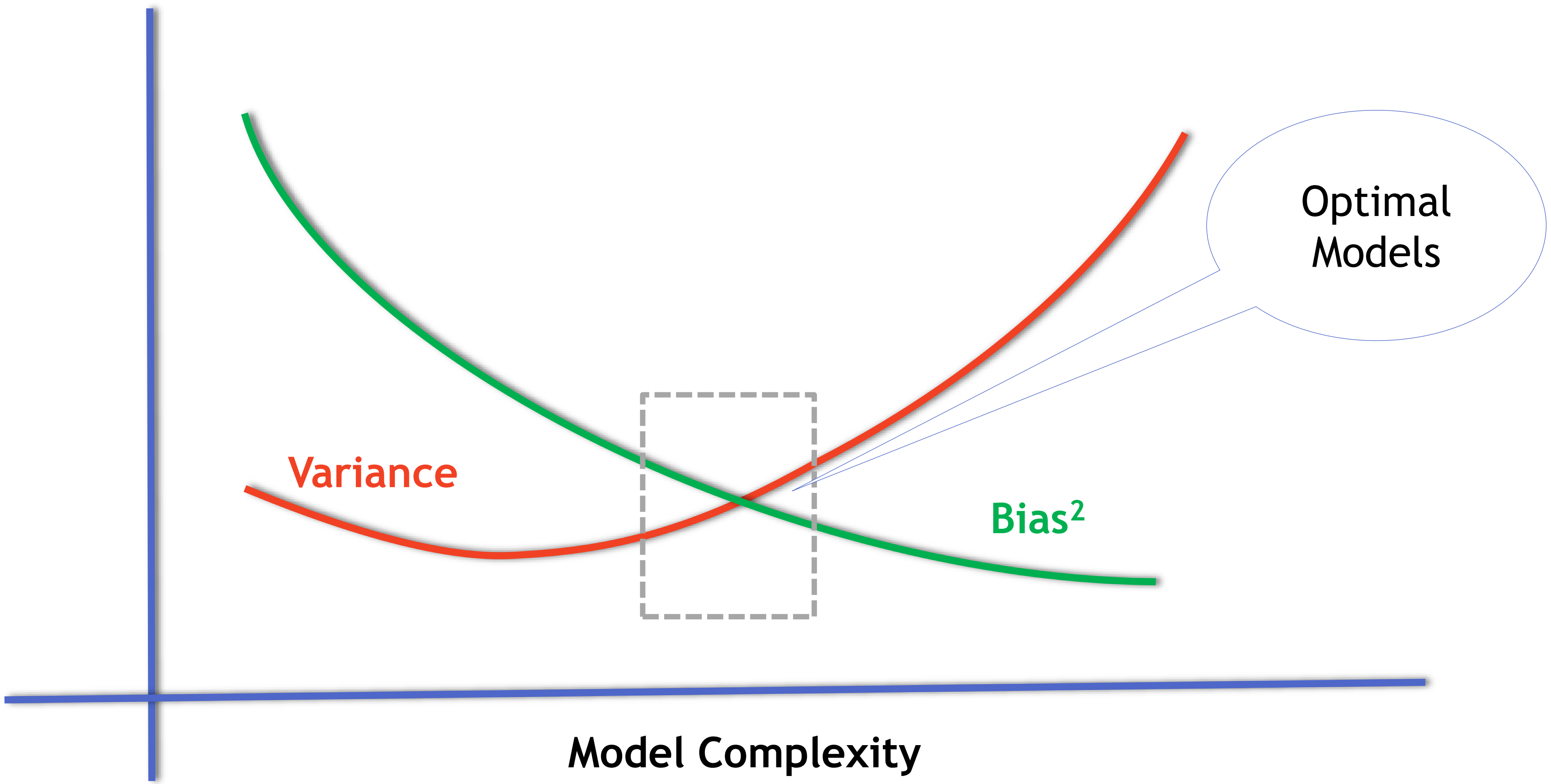
- **Overall Model Squared Error = Irreducible Error + Bias^2 + Variance**
- Overall error is made by bias and variance together
- High bias low variance, Low bias and high variance, both are bad for the overall accuracy of the model
- A good model need to have low bias and low variance or at least an optimal where both of them are jointly low
- How to choose such optimal model. How to choose that optimal model complexity

Choosing optimal model-Bias Variance Tradeoff

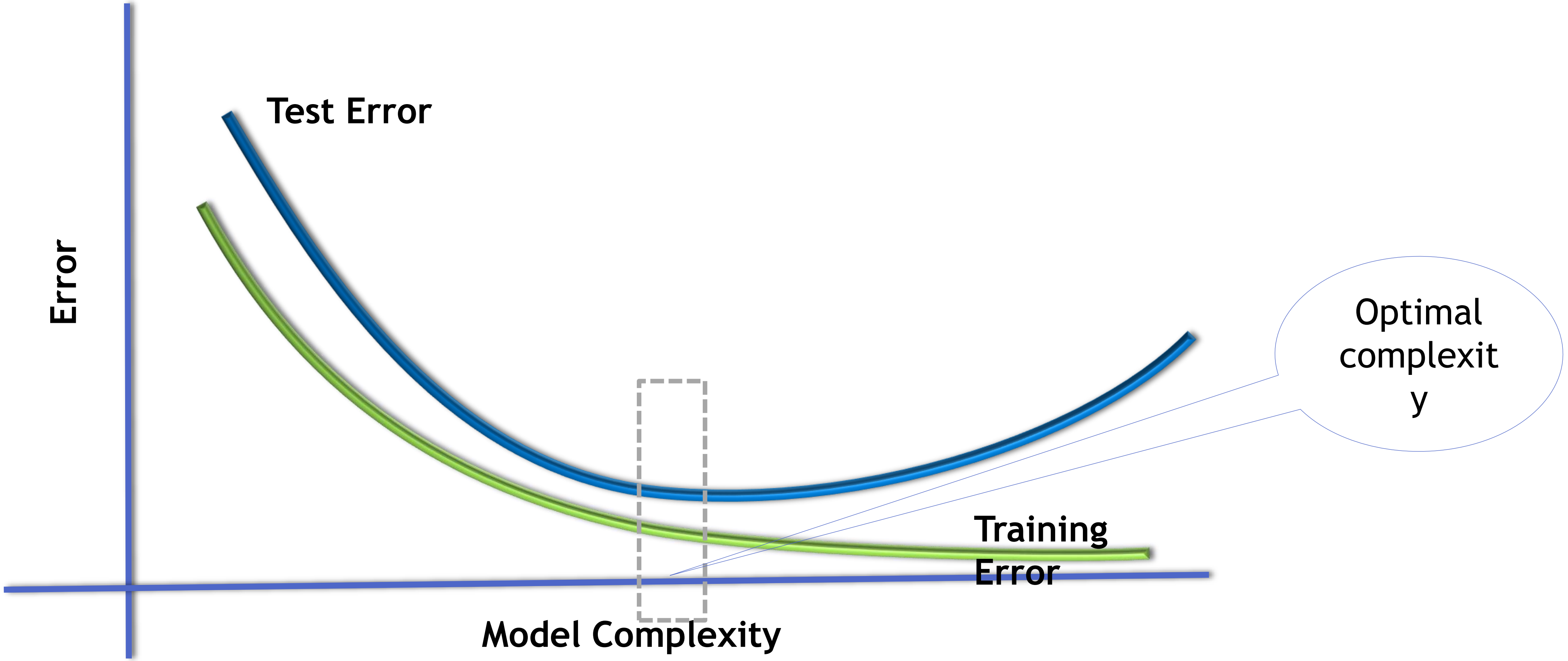
Two ways of reading bias and variance

- Variance and bias vs Model Complexity
- Testing and Training Error vs Model Complexity

Bias Variance Tradeoff



Test and Training error



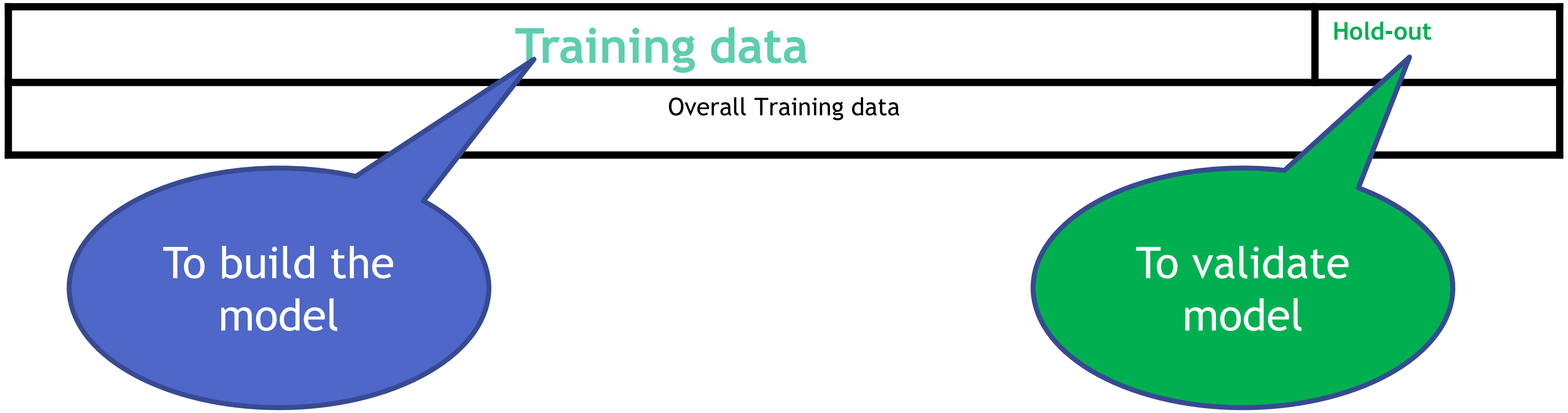
Choosing optimal model

- Unfortunately There is no scientific method
  - How to choose optimal model complexity that gives minimum test error?
  - Training error is not a good estimate of the test error.
  - There is always bias-variance tradeoff in choosing the appropriate complexity of the model.
  - We can use cross validation methods, boot strapping and bagging to choose the optimal and consistent model

Holdout data Cross validation

Holdout data Cross validation

- The best solution is out of time validation. Or the testing error should be given high priority over the training error.
- A model that is performing good on training data and equally good on testing is preferred.
- We may not have to test data always. How do we estimate test error?
- We take the part of the data as training and keep aside some potion for validation. May be 80%-20% or 90%-10%
- Data splitting is a very basic intuitive method



Lab: Holdout data Cross validation

- Data: Fiberbits/Fiberbits.csv
- Take a random sample with 80% data as training sample
- Use rest 20% as holdout sample.
- Build a model on 80% of the data. Try to validate it on holdout sample.
- Try to increase or reduce the complexity and choose the best model that performs well on training data as well as holdout data

Steps - Holdout data Cross validation

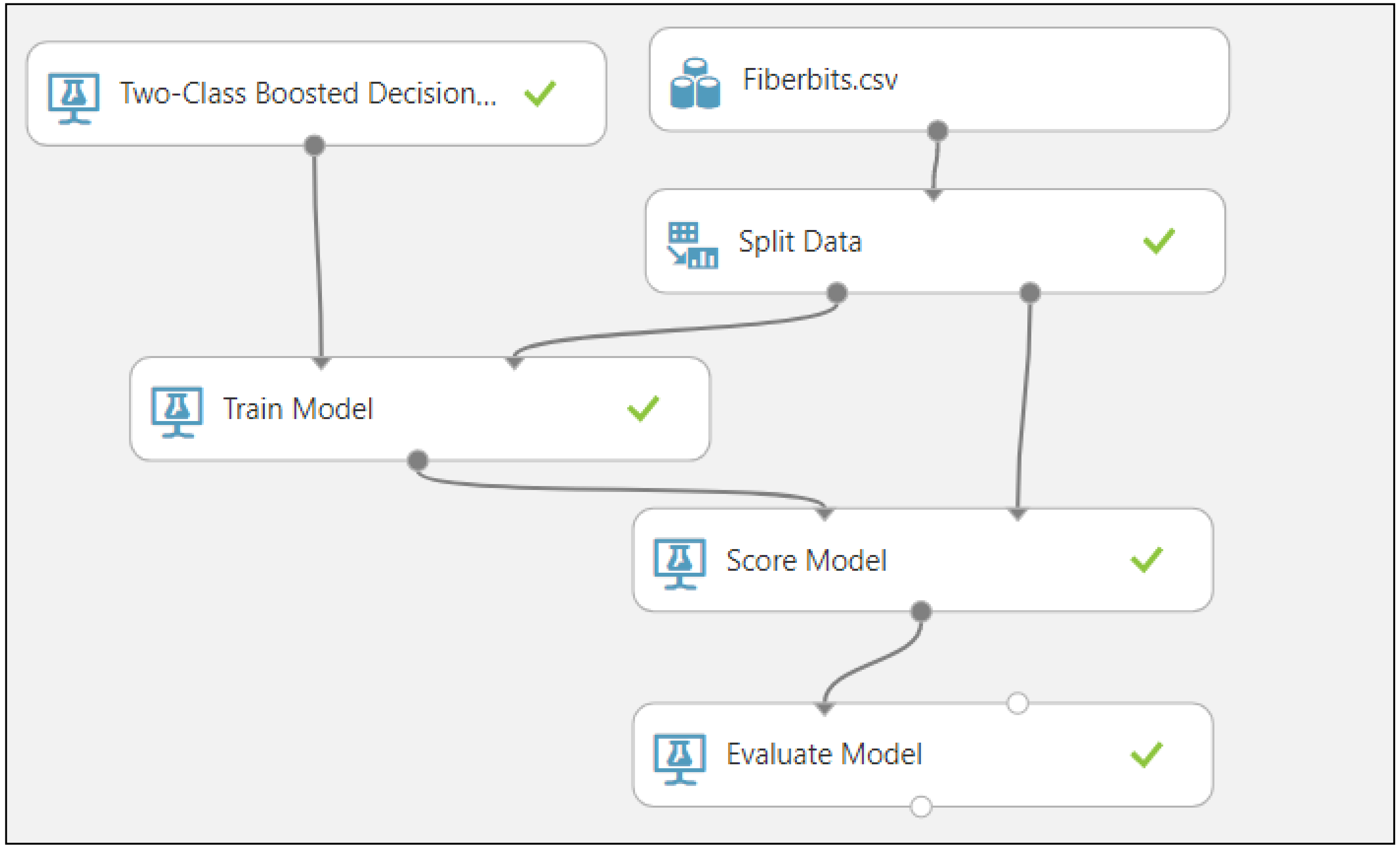
- Building Decision Tree with FiberBits Data :
  - Drag and drop the Dataset into the canvas
  - Drag and drop Split Data, connect it to the dataset and Select the properties:
    - Splitting mode → Split Rows
    - Fraction of Rows → 0.8
    - check Randomized Split
    - Random Seed → 20(any positive integer)
  - Drag and drop **Two-Class Boosted Decision Tree, Train Model, Score Model** and **Evaluate Model**
  - Connect **Two-Class Boosted Decision Tree** to the first input of **Train Model** and **Training Data** to the Second input of **Train Model**
  - Connect the output of **Train Model** first input of **Score Model** and **Training Data** to the Second input of **Score Model**
  - Connect the output of **Score Model** to the input of **Evaluate Model**

Steps - Model with huge Variance

- Click on **Two-Class Boosted Decision Tree** and select the following:
  - Create trainer mode → Single Parameter
  - Maximum number of leaves per tree → 5
  - Minimum number of samples per leaf node → 30
  - Learning rate → 0.09
  - Number of trees constructed → 1
- Click on Train Model and select the column for which the prediction is done(active_cust)
- Click run and visualize the output of Evaluate Model
- Repeat the same by passing Test Data(Second output of Split Data) to the score model

Steps - Holdout data Cross validation

Fig29: Decision Tree Modal



Steps - Holdout data Cross validation

Fig30: Properties(Two-Class Boosted Decision Tree-Modal1)

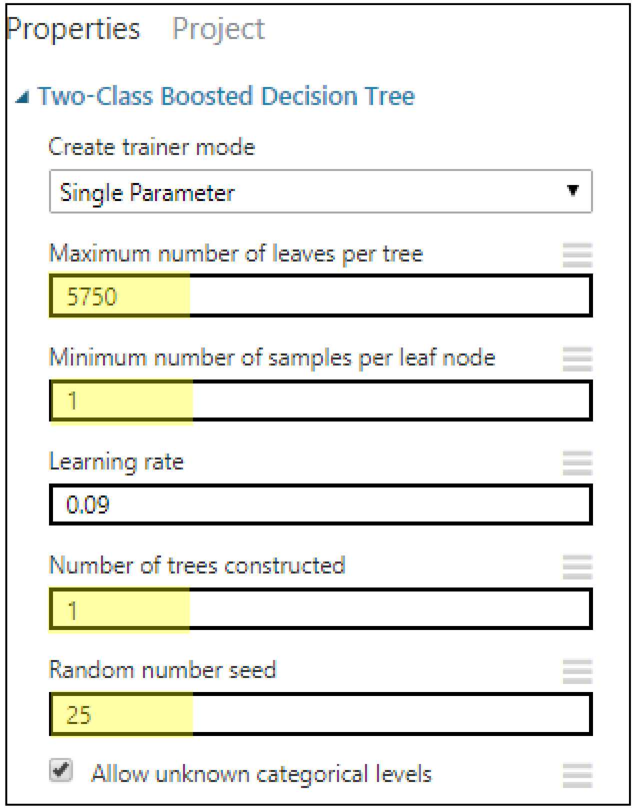
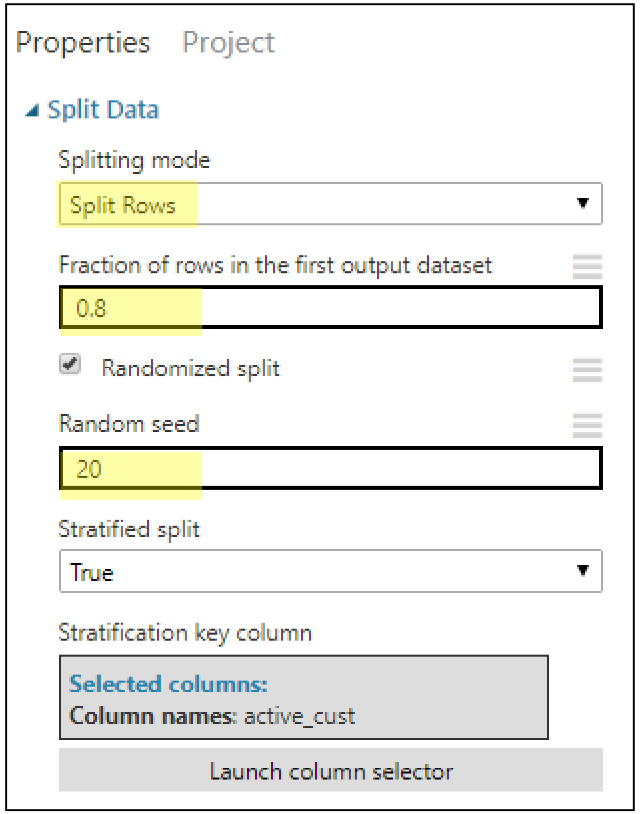


Fig31:Properties(Split Data)



Steps - Holdout data Cross validation

Fig32: Accuracy with Training Data(Modal1)

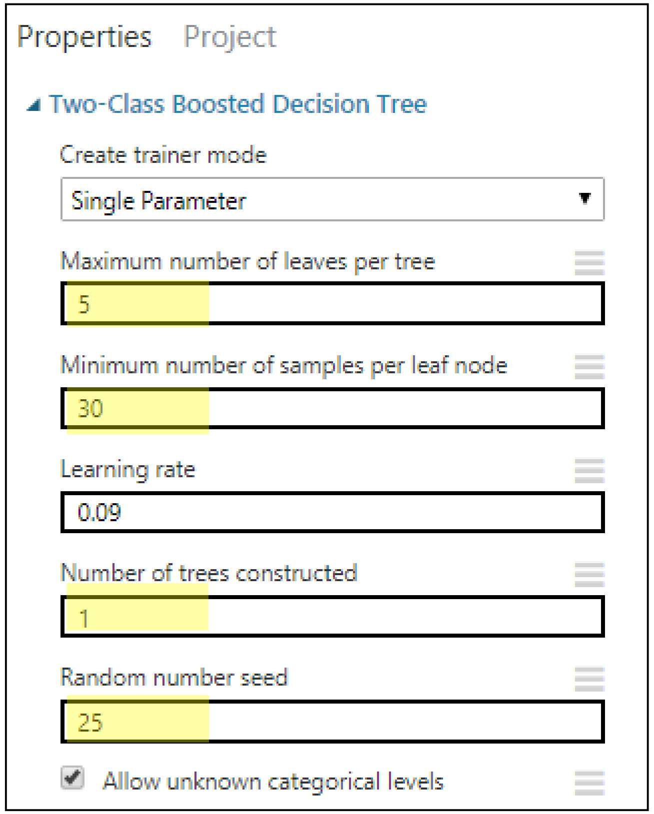
True Positive	False Negative	Accuracy	Precision	Threshold	AUC
45224	1063	0.959	0.954	0.5	0.985
False Positive	True Negative	Recall	F1 Score		
2193	31520	0.977	0.965		
Positive Label	Negative Label				
1	0				

Fig33: Accuracy with Test Data(Modal1)

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
10337	1235	0.866	0.877	0.5	0.853
False Positive	True Negative	Recall	F1 Score		
1451	6977	0.893	0.885		
Positive Label	Negative Label				
1	0				

Steps - Holdout data Cross validation

Fig34: Properties(Two-Class Boosted Decision Tree-Modal2)



Steps - Holdout data Cross validation

Fig35: Accuracy with Training Data(Modal2)

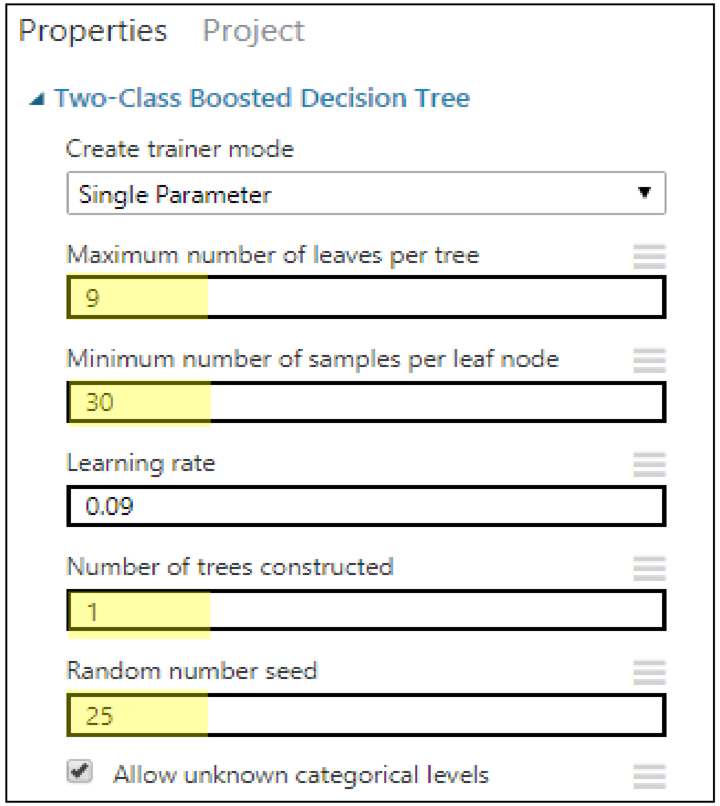
True Positive	False Negative	Accuracy	Precision	Threshold	AUC
40882	5405	0.789	0.781	0.5	0.832
False Positive	**True Negative**	**Recall**	**F1 Score**		
11451	22262	0.883	0.829		
Positive Label	**Negative Label**				
1	0				

Fig36: Accuracy with Test Data(Modal2)

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
10232	1340	0.788	0.779	0.5	0.830
False Positive	**True Negative**	**Recall**	**F1 Score**		
2900	5528	0.884	0.828		
Positive Label	**Negative Label**				
1	0				

Steps - Holdout data Cross validation

Fig37: Properties(Two-Class Boosted Decision Tree-Modal3)



Steps - Holdout data Cross validation

Fig38: Accuracy with Training Data(Modal3)

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
42488	3799	0.847	0.834	0.5	0.864
False Positive	**True Negative**	**Recall**	**F1 Score**		
8453	25260	0.918	0.874		
Positive Label	**Negative Label**				
1	0				

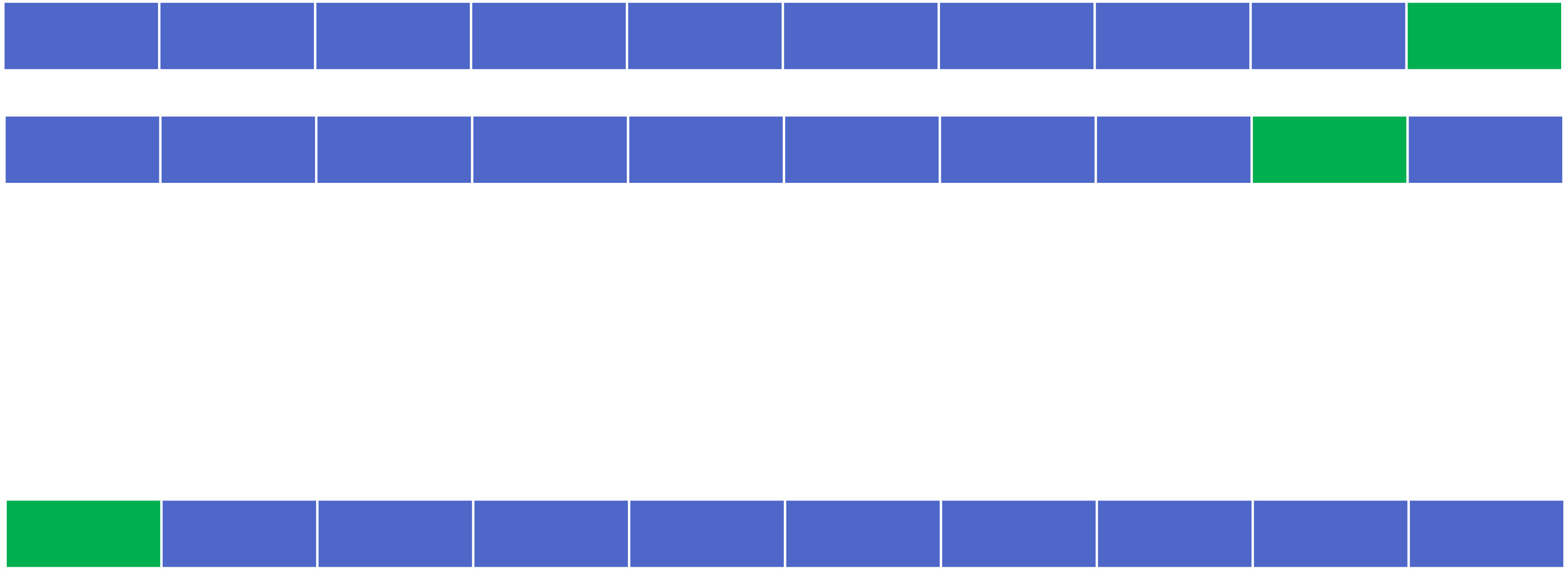
Fig39: Accuracy with Test Data(Modal3)

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
10601	971	0.844	0.832	0.5	0.862
False Positive	**True Negative**	**Recall**	**F1 Score**		
2147	6281	0.916	0.872		
Positive Label	**Negative Label**				
1	0				

Ten-fold Cross - Validation

Ten-fold Cross - Validation

- Divide the data into 10 parts(randomly)
- Use 9 parts as training data(90%) and the tenth part as holdout data(10%)
- We can repeat this process 10 times
- Build 10 models, find average error on 10 holdout samples. This gives us an idea on testing error



K-fold - Validation

K-fold Cross Validation

- A generalization of cross validation.
- Divide the whole dataset into k equal parts
- Use k^{th} part of the data as the holdout sample, use remaining k-1 parts of the data as training data
- Repeat this K times, build K models. The average error on holdout sample gives us an idea on the testing error
- Which model to choose?
  - Choose the model with least error and least complexity
  - Or the model with less than average error and simple (less parameters)
  - Finally use complete data and build a model with the chosen number of parameters
- Note: Its better to choose K between 5 to 10. Which gives 80% to 90% training data and rest 20% to 10% is holdout data

LAB- K-fold Cross Validation

LAB- K-fold Cross Validation

- Build a tree model on the fiber bits data.
- Try to build the best model by making all the possible adjustments to the parameters.
- What is the accuracy of the above model?
- Perform 10 –fold cross validation. What is the final accuracy?
- Perform 20 –fold cross validation. What is the final accuracy?
- What can be the expected accuracy on the unknown dataset?

Steps - K-fold Cross Validation

- Create a Decision tree modal using Two-Class Boosted Decision Tree with the following properties:
  - Create trainer mode → Single Parameter
  - Maximum number of leaves per tree → 3000
  - Minimum number of samples per leaf node → 1
  - Learning rate → 0.09
  - Number of trees constructed → 1
- Click on Train Model and select the column for which the prediction is done(active_cust)
- Click run and visualize the output of Evaluate Model

Steps - K-fold Cross Validation

Fig40: Decision Tree Modal

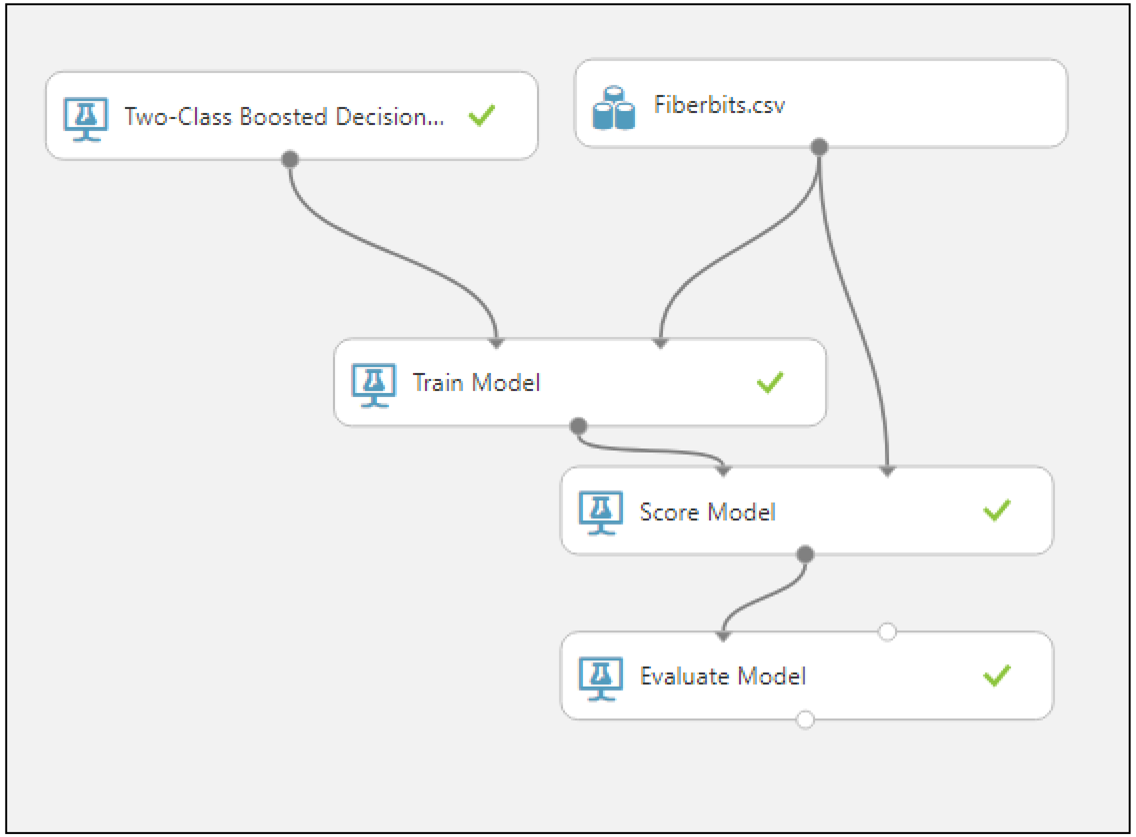
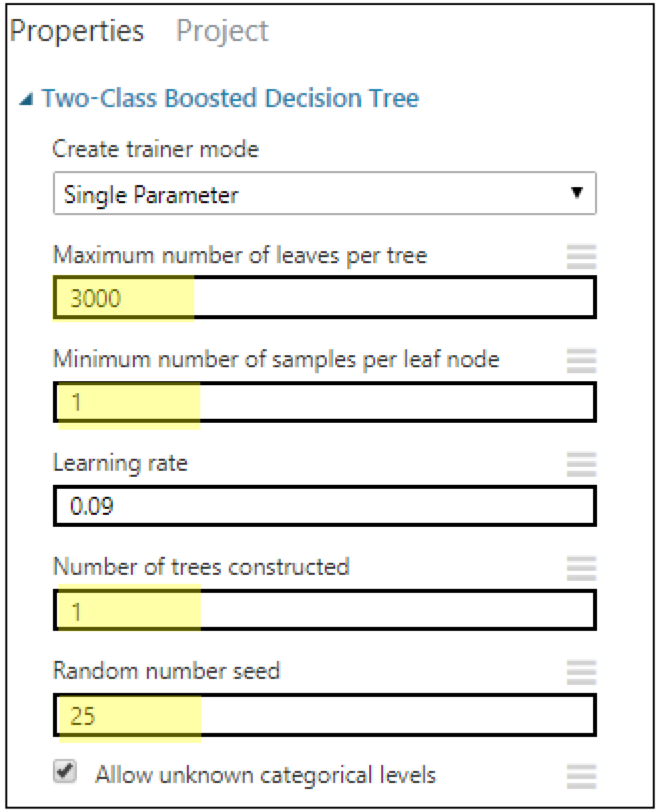
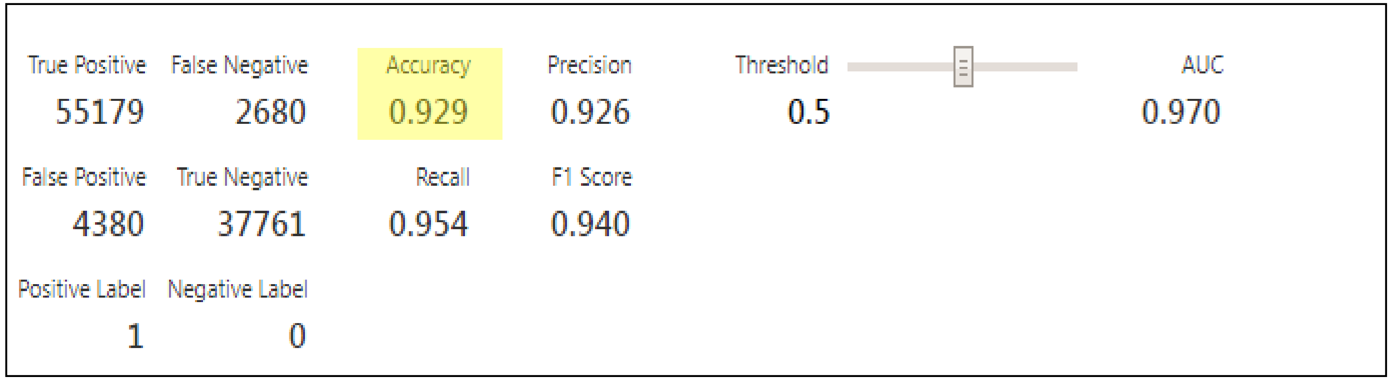


Fig41: Properties(Two-Class Boosted Decision Tree)



Steps - K-fold Cross Validation

Fig42: Accuracy



Steps - K-fold Cross Validation

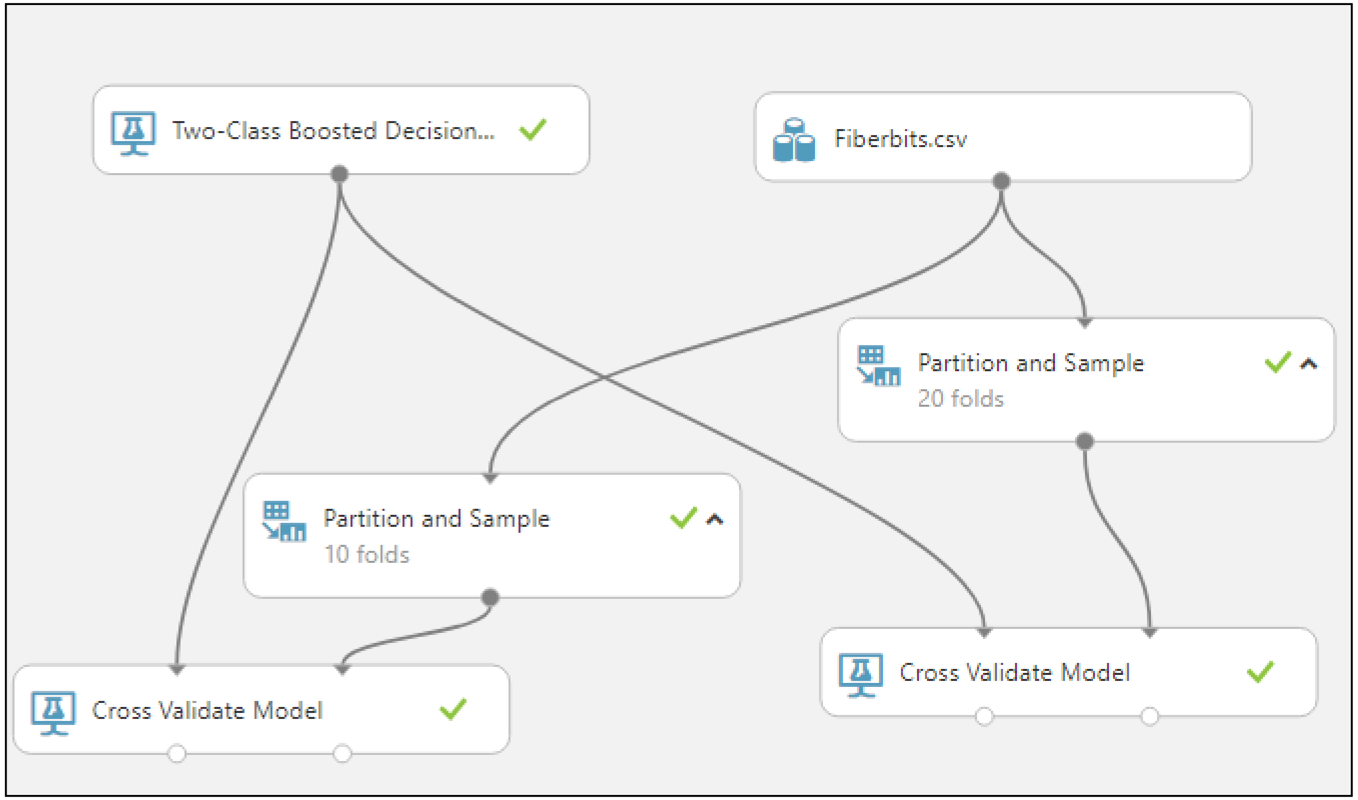
- Drag and drop **Partition and Sample** into the canvas, connect it to the dataset and select the following Properties:
  - Partition or sample mode → Assign to Folds
  - Uncheck Use replacement in the partitioning
  - Check Randomized split
  - Random seed → 20(any positive integer)
  - Specify the partitioner method → Partition Evenly
  - Specify number of folds to split evenly into →10
  - Stratified split → True
  - Stratification key column →active_cust
- Drag and drop **Cross Validate Model** in to the canvas

Steps - K-fold Cross Validation

- Connect **Two-Class Boosted Decision Tree** to the first input of **Cross Validate Model** and **Partition and sample** to the second input of **Cross Validate Model**
- **In Cross Validate Model** select label column as active_cust
- Click on run and visualize the second output circle of **Cross Validate Model**
- Click on accuracy column after the fold values, we can see a row for the Mean, check the Mean value of the accuracy column
- Follow the same by changing 'Specify number of folds to split evenly into → 20' and check the Mean of accuracy column

Steps - K-fold Cross Validation

Fig43: K-Fold Cross validation with 10 folds and 20 folds



Steps - K-fold Cross Validation

Fig44: Properties-Partition and Sample(10 folds)

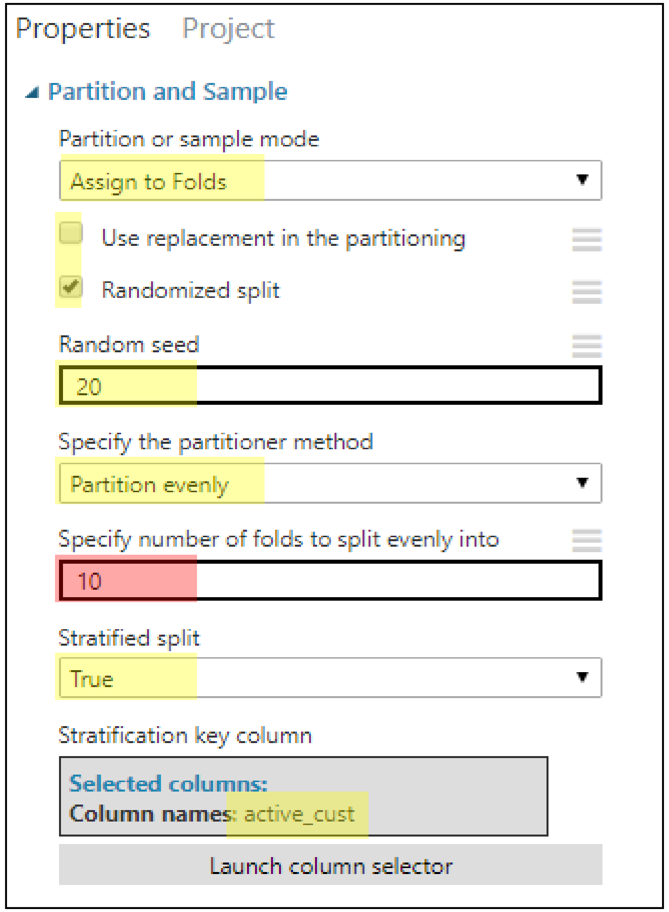
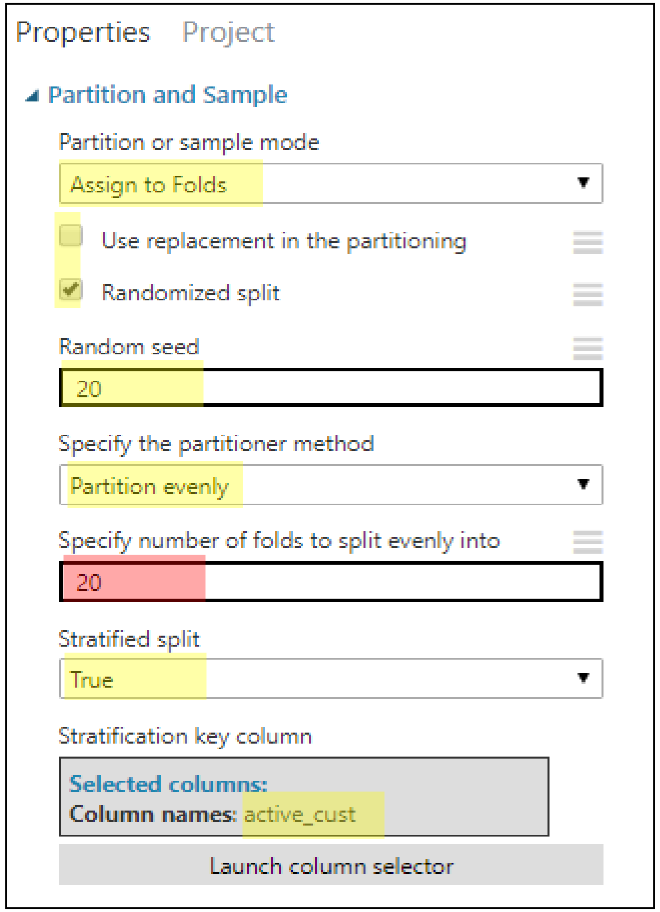


Fig45: Properties-Partition and Sample(20 folds)



Steps - K-fold Cross Validation

Fig46: Mean of Accuracy Column(10-folds)

K-Fold Cross Validat › Cross Validate Model › Evaluation results by fold

rows 12 columns 10

		Classification					
7	10000	FastTree (Boosted Trees) Classification	0.8206	0.835575	0.85897	0.847111	0.855598
8	10000	FastTree (Boosted Trees) Classification	0.8225	0.839168	0.857587	0.848278	0.854639
9	10000	FastTree (Boosted Trees) Classification	0.8268	0.841131	0.863809	0.852319	0.859631
Mean	100000	FastTree (Boosted Trees) Classification	0.826	0.842078	0.860696	0.851281	0.858212
Standard Deviation	100000	FastTree (Boosted Trees) Classification	0.002877	0.003388	0.002892	0.002358	0.002454

Steps - K-fold Cross Validation

Fig47: Mean of Accuracy Column(20-folds)

K-Fold Cross Validat › Cross Validate Model › Evaluation results by fold

rows 22 columns 10

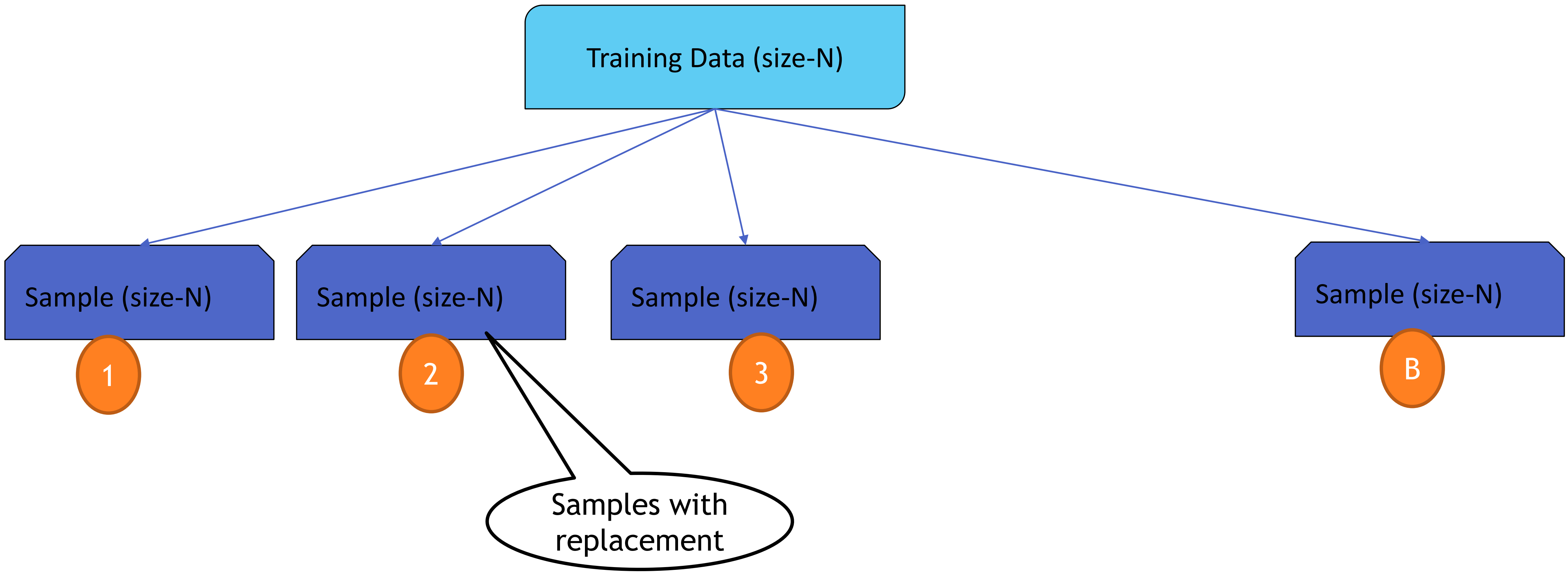
		Classification					
17	5000	FastTree (Boosted Trees) Classification	0.8302	0.844804	0.865538	0.855045	0.863621
18	5000	FastTree (Boosted Trees) Classification	0.8212	0.83529	0.860698	0.847804	0.853229
19	5000	FastTree (Boosted Trees) Classification	0.8284	0.846206	0.859661	0.852881	0.860726
Mean	100000	FastTree (Boosted Trees) Classification	0.82601	0.842078	0.860713	0.851278	0.858304
Standard Deviation	100000	FastTree (Boosted Trees) Classification	0.004824	0.003701	0.007431	0.004486	0.005856

Bootstrap Cross Validation

Bootstrap Methods

- Boot strapping is a powerful tool to get an idea on accuracy of the model and the test error
- Can estimate the likely future performance of a given modeling procedure, on new data not yet realized.
- The Algorithm
  - We have a training data is of size N
  - Draw random sample with replacement of size N – This gives a new dataset, it might have repeated observations, some observations might not have even appeared once.
  - Create B such new datasets. These are called boot strap datasets
  - Build the model on these B datasets, we can test the models on the original training dataset.

Bootstrap Method



Bootstrap Example

- Example
  1. We have a training data is of size 500
  2. Boot Strap Data-1:
  - Create a dataset of size 500. To create this dataset, draw **a random point**, note it down, then replace it back. Again draw another sample point. Repeat this process 500 times. This makes a dataset of size 500. Call this as Boot Strap Data-1
  3. Multiple Boot Strap datasets
  - Repeat the procedure in step -2 multiple times. Say 200 times. Then we have 200 Boot Strap datasets
  4. We can build the models on these 200 boost strap datasets and the average error gives a good idea on overall error. We can even use the original training data as the test data for each of the models

statinfer

LAB: Bootstrap Cross Validation

LAB: Bootstrap cross validation

- Draw a boot strap sample with sufficient sample size
- Build a tree model and get an estimate on true accuracy of the model

Steps - Bootstrap cross validation

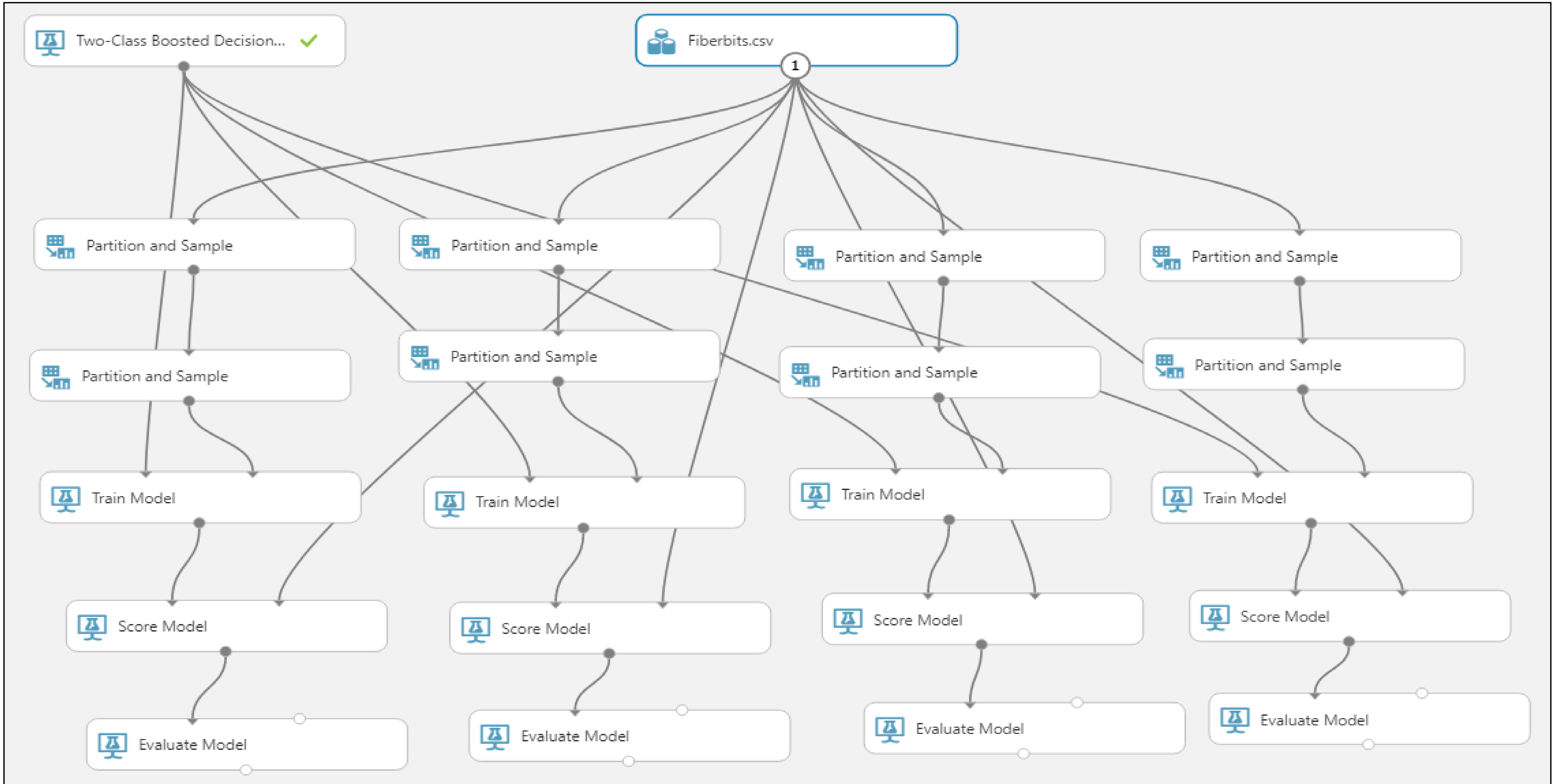
- Drag and drop the dataset into the canvas
- Drag and drop four(B) Partition and Sample for Assigning the folds, connect it to the dataset and select the properties(Fig:49)
- Drag and drop another four Partition and Sample for picking the folds, connect it to the previous Partition and Sample and select the properties(Fig:50)
- Drag and drop Two class Boosted Decision Tree in to the canvas
- Drag and drop four Train Model and give the following connection
  - First input to Two class Boosted Decision Tree
  - Second input to Partition and Sample for picking the folds

Steps - Bootstrap cross validation

- Drag and drop four Score Model and give the following connection
  - First input to Train Model
  - Second input to Original Dataset
- Drag and drop four Evaluate Model and connect it to the Score Model
- Click on run and visualize the Evaluate model to view the accuracy

Steps - Bootstrap cross validation

Fig48: Bootstrap Cross Validation Model with Four Samples



Steps - Bootstrap cross validation

Fig 49: Properties - Partition and Sample at level1(different Seeds)

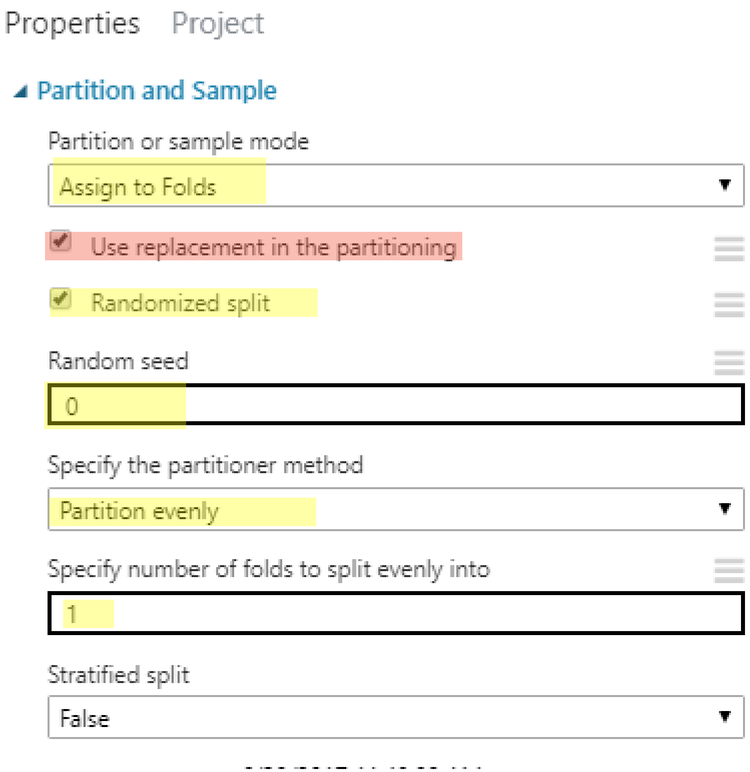
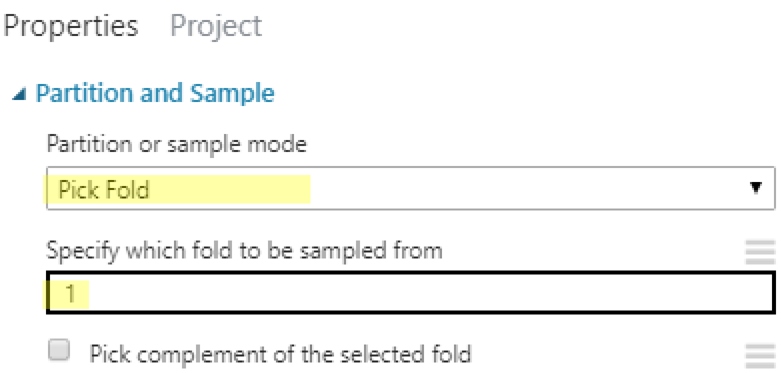


Fig50: Properties - Partition and Sample at level2



Steps - Bootstrap cross validation

Fig51:Accuracy (Sample-1)

True Positive	False Negative	Accuracy	Precision
49815	8044	0.826	0.842
False Positive	True Negative	Recall	F1 Score
9344	32797	0.861	0.851
Positive Label	Negative Label		
1	0		

Fig52:Accuracy (Sample-2)

True Positive	False Negative	Accuracy	Precision
49815	8044	0.826	0.842
False Positive	True Negative	Recall	F1 Score
9344	32797	0.861	0.851
Positive Label	Negative Label		
1	0		

Fig53:Accuracy (Sample-3)

True Positive	False Negative	Accuracy	Precision
49815	8044	0.826	0.842
False Positive	True Negative	Recall	F1 Score
9344	32797	0.861	0.851
Positive Label	Negative Label		
1	0		

Fig54:Accuracy (Sample-4)

True Positive	False Negative	Accuracy	Precision
49815	8044	0.826	0.842
False Positive	True Negative	Recall	F1 Score
9344	32797	0.861	0.851
Positive Label	Negative Label		
1	0		

Accuracy(average): 0.826

statinfer

Conclusion

Conclusion

- We studied
  - Validating a model, Types of data & Types of errors
  - The problem of over fitting & The problem of under fitting
  - Bias Variance Tradeoff
  - Cross validation & Boot strapping
- Training error is what we see and that is not the true performance metric
- Test error plays vital role in model selection
- R-square, Adj-R-square, Accuracy, ROC, AUC, AIC and BIC can be used to get an idea on training error
- Cross Validation and Boot strapping techniques give us an idea on test error
- Choose the model based on the combination of AIC, Cross Validation and Boot strapping results
- Bootstrap is widely used in ensemble models & random forests.

Thank you

statinfer

Our e-Learning Modules

www.statinfer.com

Part 9/12 - Neural Networks With Azure

Venkat Reddy

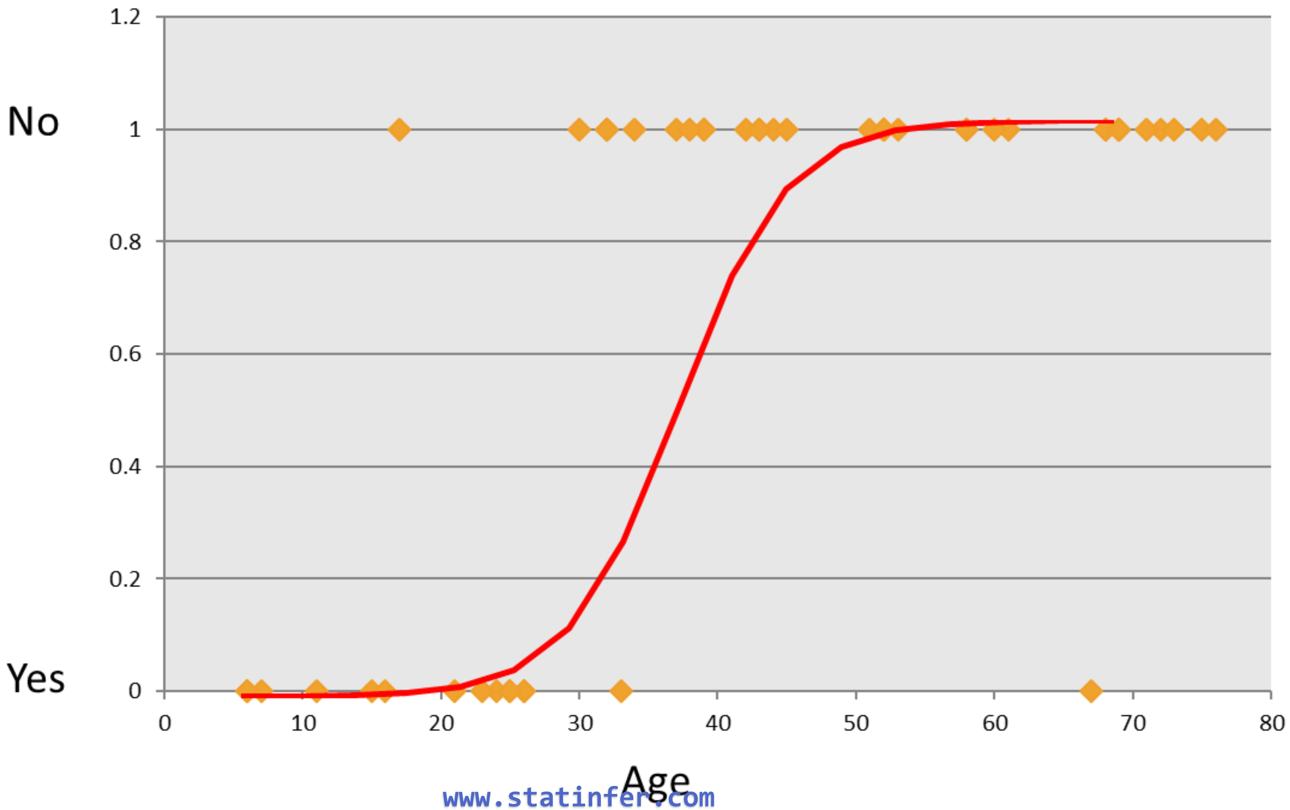
Contents

Contents

Recap of Logistic Regression

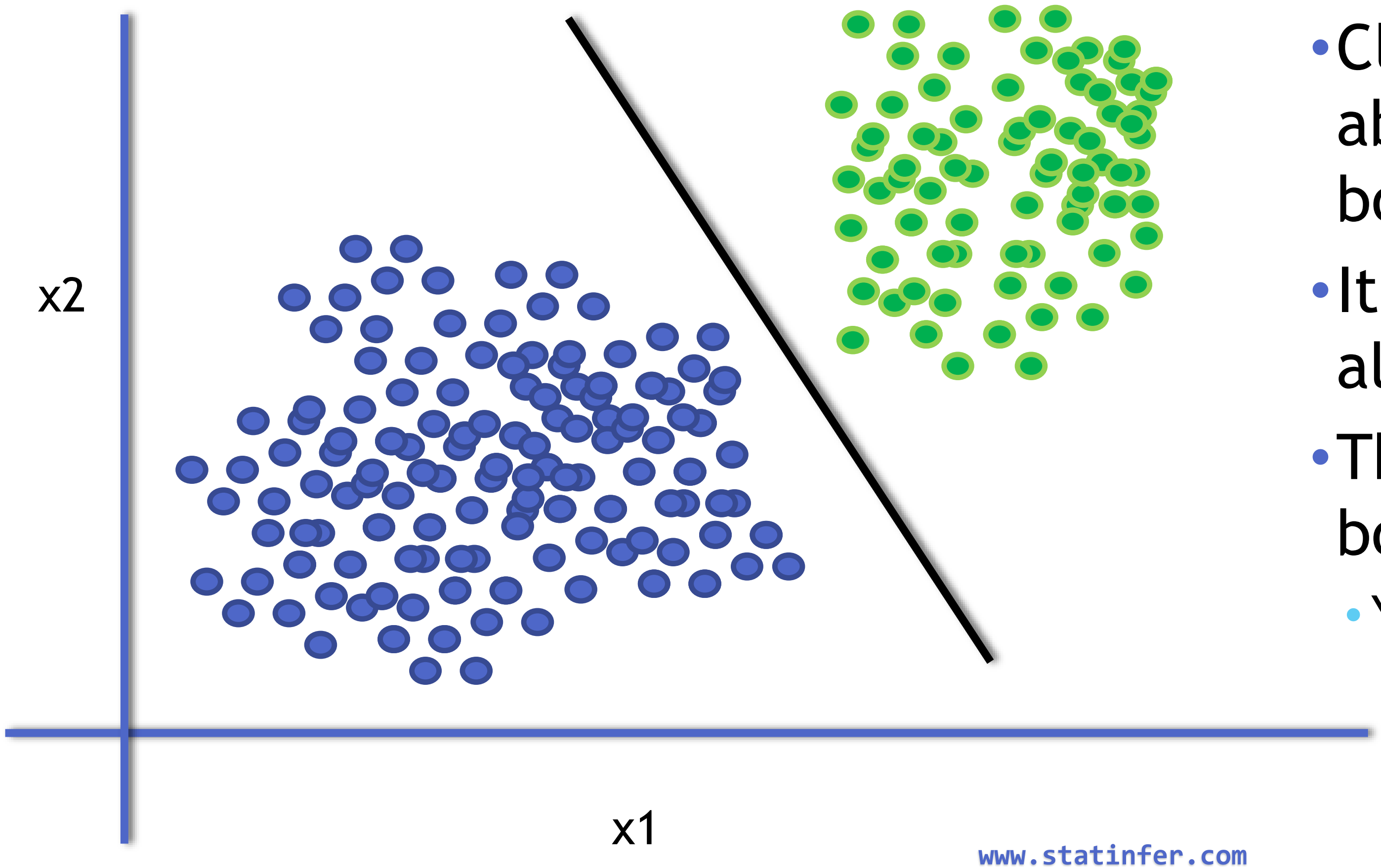
Recap of Logistic Regression

- Categorical output YES/NO type
- Using the predictor variables to predict the categorical output



Decision Boundary

Decision Boundary – Logistic Regression



- The line or margin that separates the classes
- Classification algorithms are all about finding the decision boundaries
- It need not be straight line always
- The final function of our decision boundary looks like
  - $Y=1$ if $w^T x+w_0>0$; else $Y=0$

Decision Boundary – Logistic Regression

- In logistic regression, Decision Boundary can be derived from the logistic regression coefficients and the threshold.
  - Imagine the logistic regression line $p(y)=e^{(b0+b1x1+b2x2)}/1+exp^{(b0+b1x1+b2x2)}$
  - Suppose if p(y)>0.5 then class-1 or else class-0
    - $\log(y/1-y)=b_0+b_1x_1+b_2x_2$
    - $\mathrm{Log}(0.5/0.5)=b_0+b_1x_1+b_2x_2$
    - $0=b_0+b_1x_1+b_2x_2$
    - $b_0+b_1x_1+b_2x_2=0$ is the line

Decision Boundary – Logistic Regression

- Rewriting it in mx+c form
  - $X_2=(-b_1/b_2)X_1+(-b_0/b_2)$
- Anything above this line is class-1, below this line is class-0
  - $X_2>(-b_1/b_2)X_1+(-b_0/b_2)$is class-1
  - $X_2<(-b_1/b_2)X_1+(-b_0/b_2)$ is class-0
  - $X_2=(-b_1/b_2)X_1+(-b_0/b_2)$ tie probability of 0.5
- We can change the decision boundary by changing the threshold value(here 0.5)

LAB: Logistic Regression and Decision Boundary

LAB: Logistic Regression

- Dataset: Emp_Productivity/Emp_Productivity.csv
- Filter the data and take a subset from above dataset . Filter condition is Sample_Set<3
- Draw a scatter plot that shows Age on X axis and Experience on Y-axis. Try to distinguish the two classes with colors or shapes (visualizing the classes)
- Build a logistic regression model to predict Productivity using age and experience
- Create the confusion matrix
- Calculate the accuracy and error rates

LAB: Decision Boundary

- Draw a scatter plot that shows Age on X axis and Experience on Y-axis. Try to distinguish the two classes with colors or shapes (visualizing the classes)
- Build a logistic regression model to predict Productivity using age and experience
- Finally draw the decision boundary for this logistic regression model

Steps - Logistic Regression

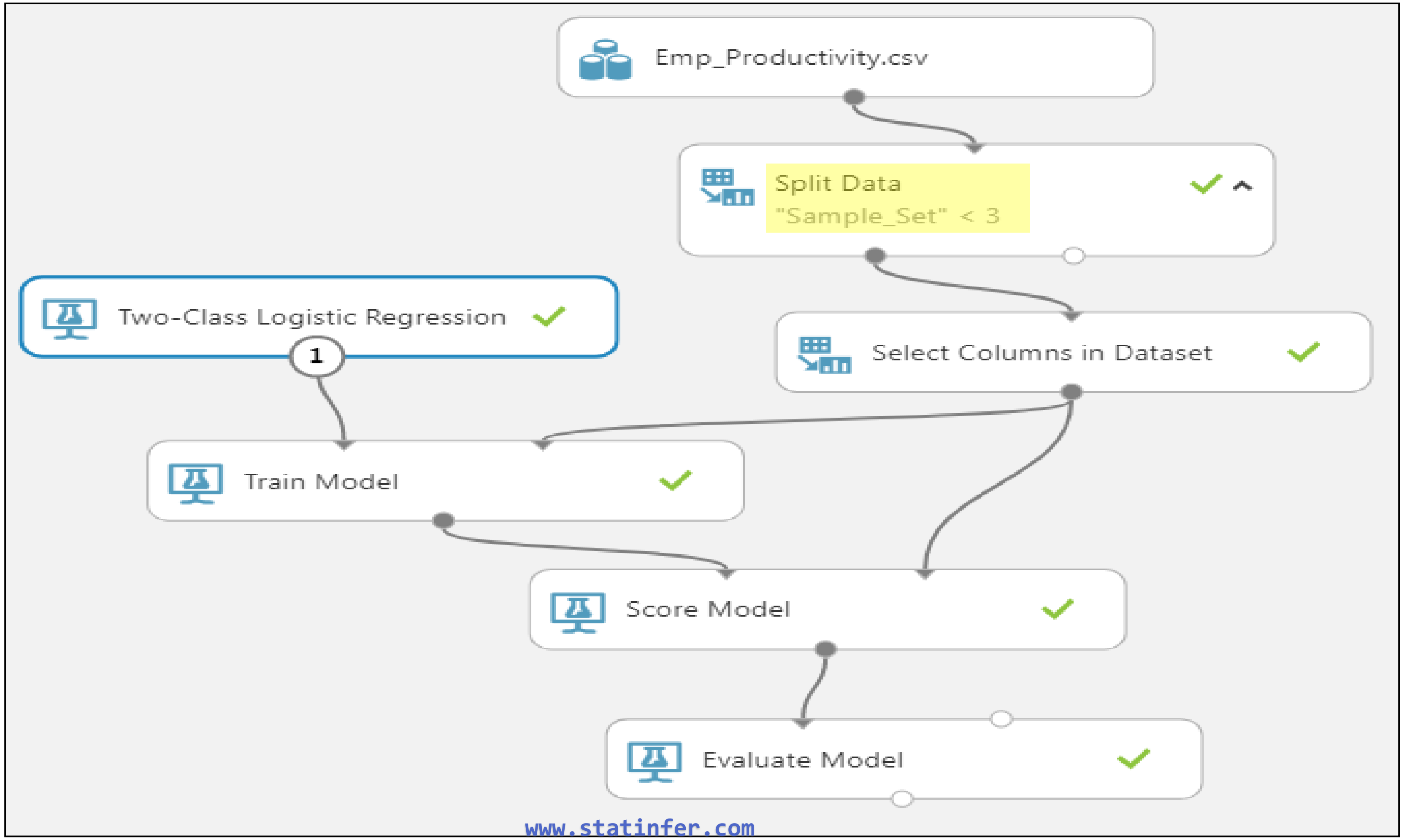
- Drag and drop the Dataset into the canvas
- Drag and drop the Split Data and connect it to the dataset
- In Split Data properties, select
  - Mode → Relative Expression
  - Expression → \"Sample_Set" < 3
- Drag and drop the Select Columns and select the columns(Age, Experience, Productivity)
- Drag and drop **Two-Class Logistic Regression, Train Model, Score Model** and **Evaluate Model**
- Connect **Two-Class Boosted Logistic Regression** to the first input of **Train Model** and **Select Columns** to the Second input of **Train Model**

Steps - Logistic Regression

- Connect the output of **Train Model** first input of **Score Model** and **Select Columns** to the Second input of **Score Model**
- Connect the output of **Score Model** to the input of **Evaluate Model**
- Click on Train Model and select the column for which the prediction is done(Productivity)
- Click run and visualize the output of Evaluate Model

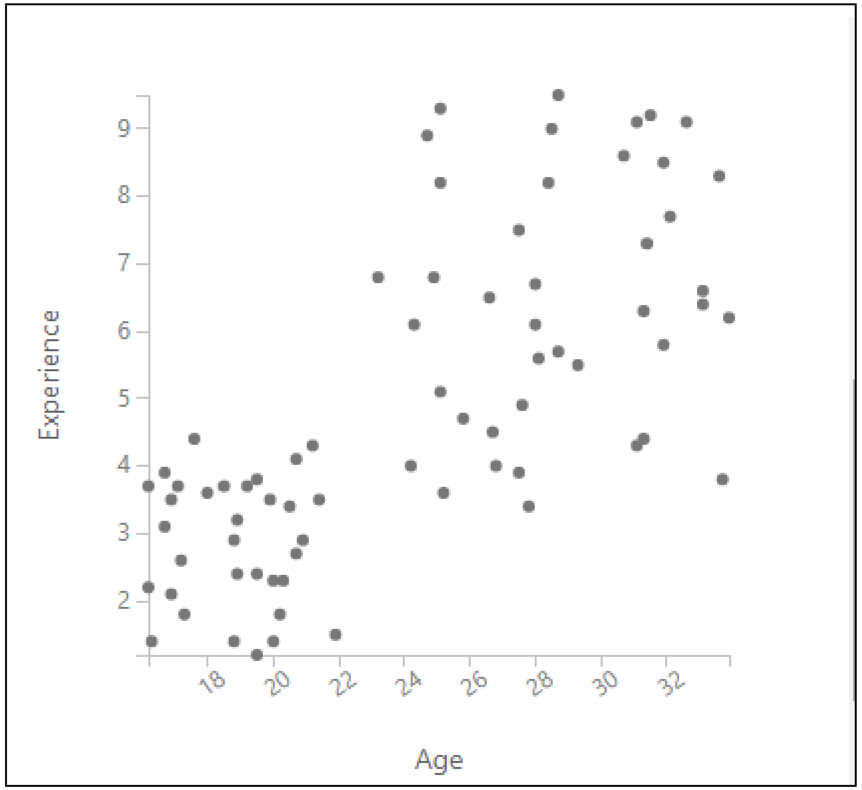
Steps - Logistic Regression

Fig1: Logistic Regression (Emp_Productivity.csv)



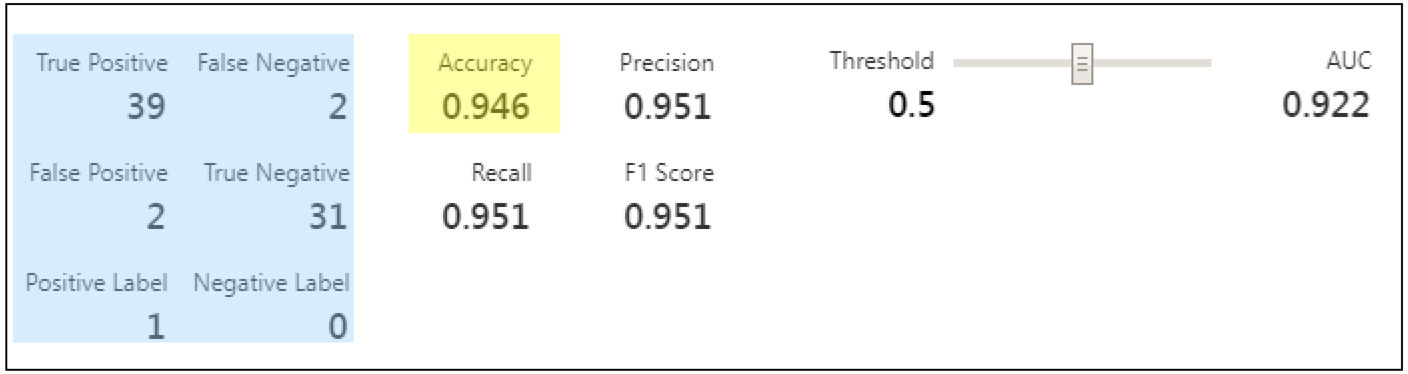
Steps - Logistic Regression

Fig2: Scatter Plot – Age vs Experience



Steps - Logistic Regression

Fig3: Accuracy and Confusion Matrix (Emp_Productivity.csv)

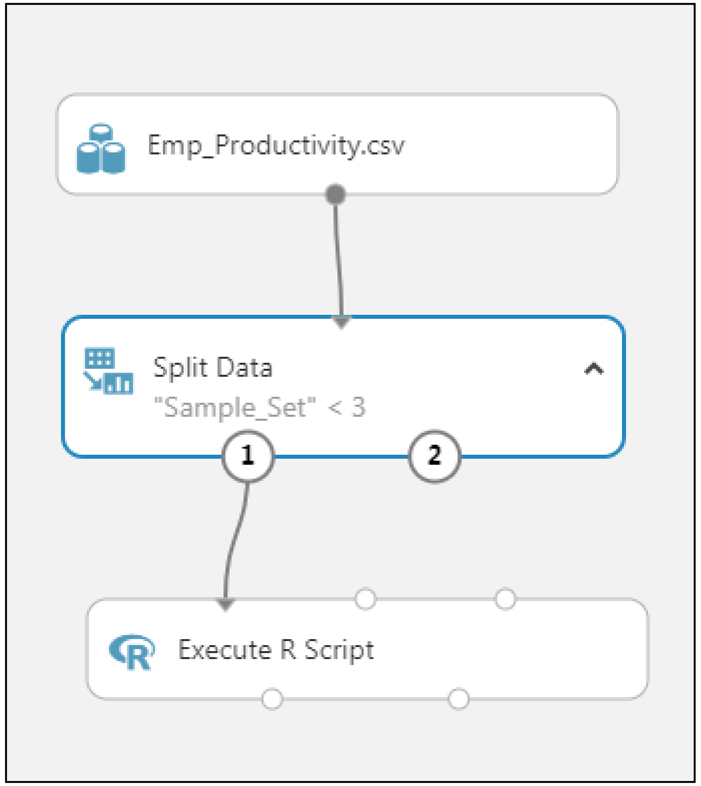


Steps - Decision Boundary

- Drag and drop the Dataset into the canvas
- Drag and drop the Split Data and connect it to the dataset
- In Split Data properties, select
  - Mode → Relative Expression
  - Expression → \"Sample_Set" < 3
- Drag and drop **Execute R Script** and connect Split Data to the first input circle
- Click on **Execute R Script**, in Properties write the code in the fig-4
- Click on run and visualize the Second Output circle of **Execute R Script**

Steps - Decision Boundary

Fig4: R-Script Logistic Regression(Emp_Productivity.csv)



statinfer

Steps - Decision Boundary

Fig5: R-Code for Logistic Regression

R Script

```
dataset1 <- maml.mapInputPort(1) # class: data.frame

library(ggplot2)
ggplot(dataset1)+geom_point(aes(x=dataset1$Age,y=dataset1$Experience,color=factor(dataset1$Productivity),
    shape=factor(dataset1$Productivity)),size=5)      #Scatter plot without decision boundary

Emp_Productivity_logit<-glm(dataset1$Productivity~dataset1$Age+dataset1$Experience, family=binomial())
coef(Emp_Productivity_logit)

slope1 <- coef(Emp_Productivity_logit)[2]/(-coef(Emp_Productivity_logit)[3])
intercept1 <- coef(Emp_Productivity_logit)[1]/(-coef(Emp_Productivity_logit)[3])

library(ggplot2)
base<-ggplot(dataset1)+geom_point(aes(x=dataset1$Age,y=dataset1$Experience,color=factor(dataset1$Productivity),
    shape=factor(dataset1$Productivity)),size=5)
base+geom_abline(intercept = intercept1 , slope = slope1, color = "red", size = 2) #Base is the scatter plot.
                                                                                   #Then we are adding the decision boundary

maml.mapOutputPort("dataset1");
```

Scatter plot without Decision boundary

Logistic Regression

Scatter plot with Decision boundary

Steps - Decision Boundary

Fig6: Scatter plot without Decision boundary(R-Output)

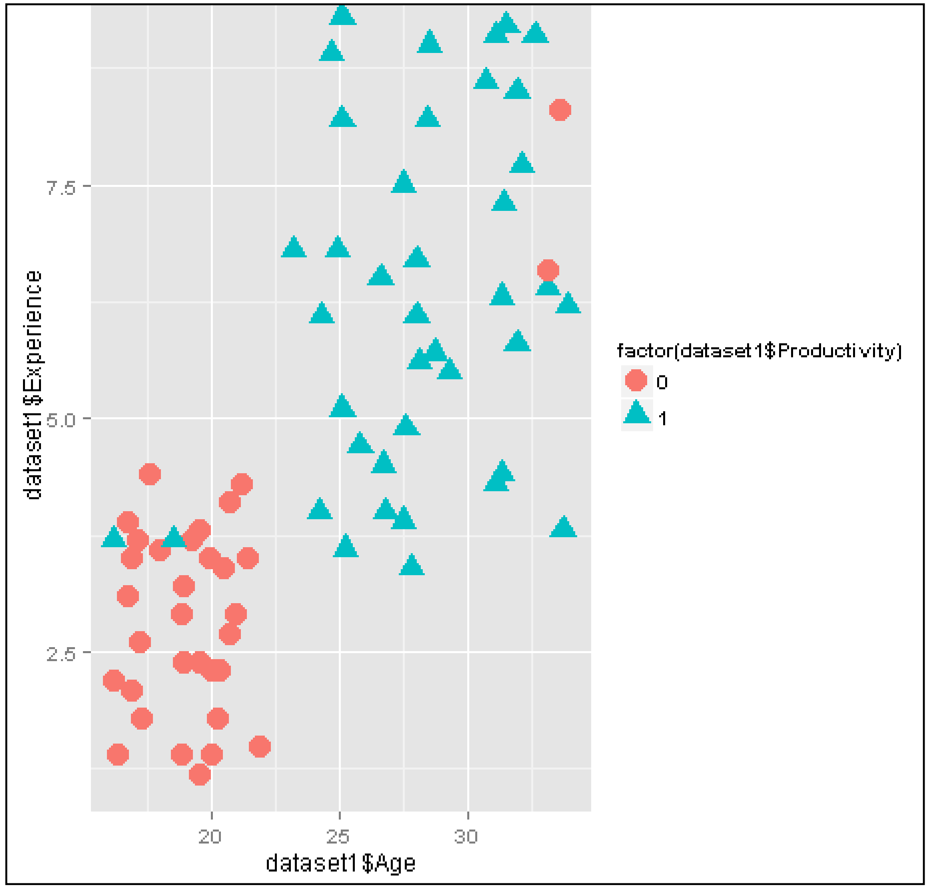
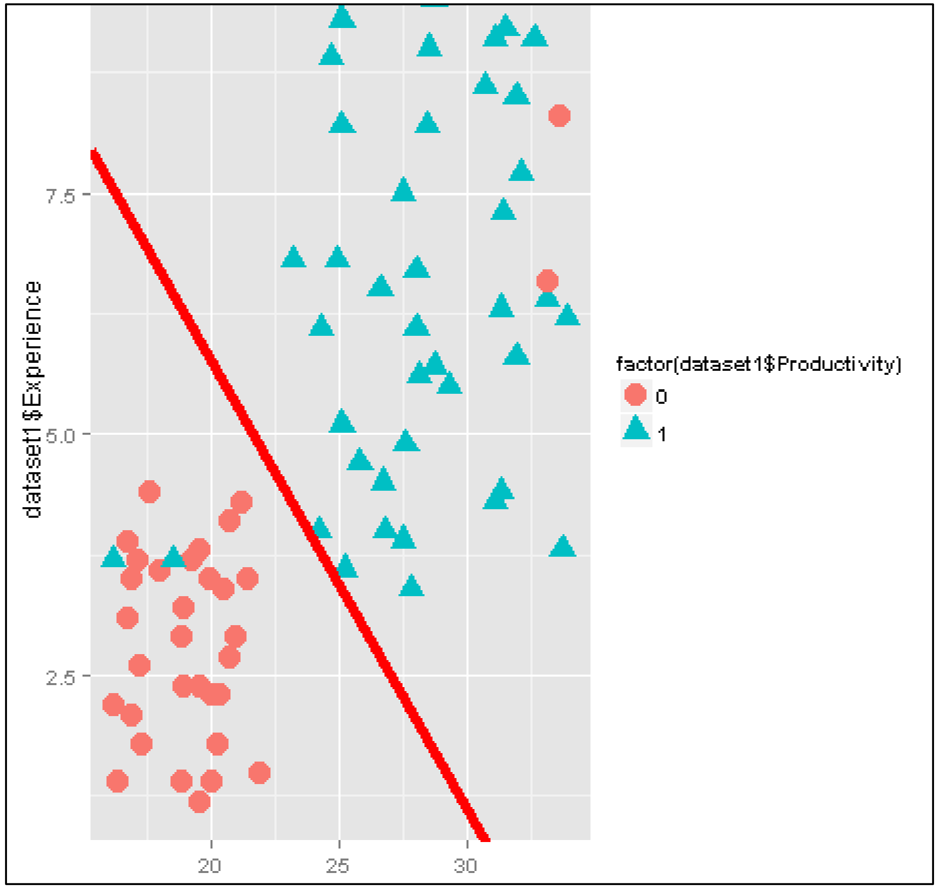


Fig7: Scatter plot with Decision boundary(R-Output)



New representation for logistic regression

statinfer

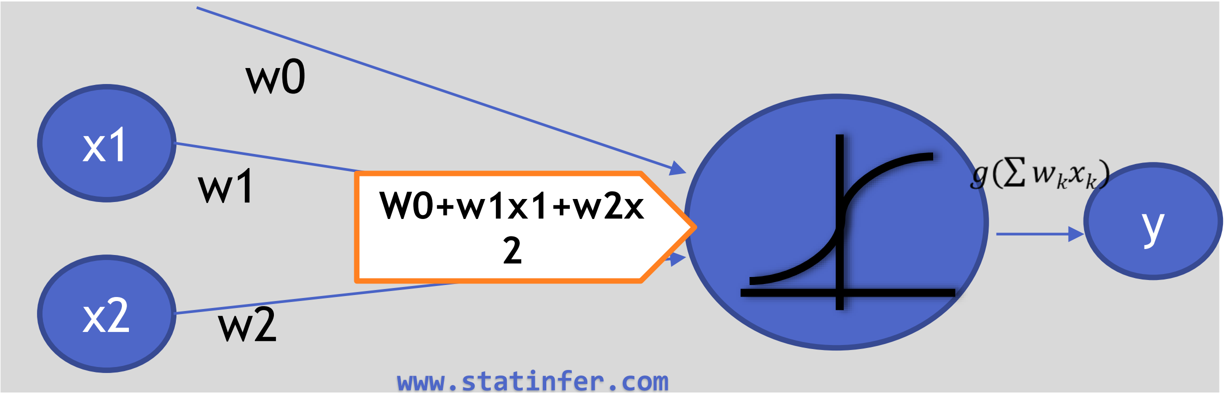
New representation for logistic regression

$$y = \frac{e^{\beta 0+\beta 1x1+\beta 2x2}}{1+e^{\beta 0+\beta 1x1+\beta 2x2}}$$

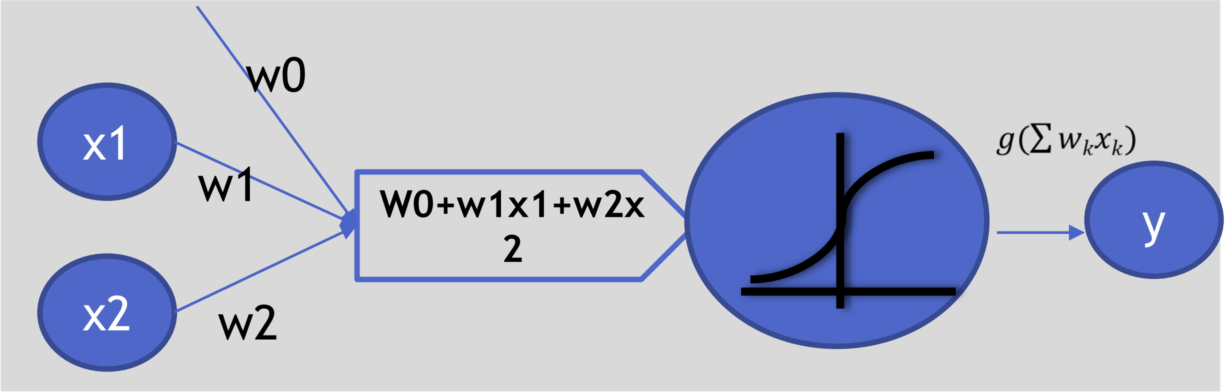
$$y = \frac{1}{1+e^{-(\beta 0+\beta 1x1+\beta 2x2)}}$$

$$y = g(w_0 + w_1x_1 + w_2x_2) \quad where \quad g(x) = \frac{1}{1+e^{-x}}$$

$$y = g(\sum w_k x_k)$$



Finding the weights in logistic regression



$$out(x) = g(\sum w_k x_k)$$

The above output is a non linear function of linear combination of inputs – A typical multiple logistic regression line

$$We\ find\ \ w\ to\ minimize\ \sum_{i=1}^{n}[y_i - g(\sum w_k x_k)]^2$$

LAB: Non-Linear Decision Boundaries

LAB: Non-Linear Decision Boundaries

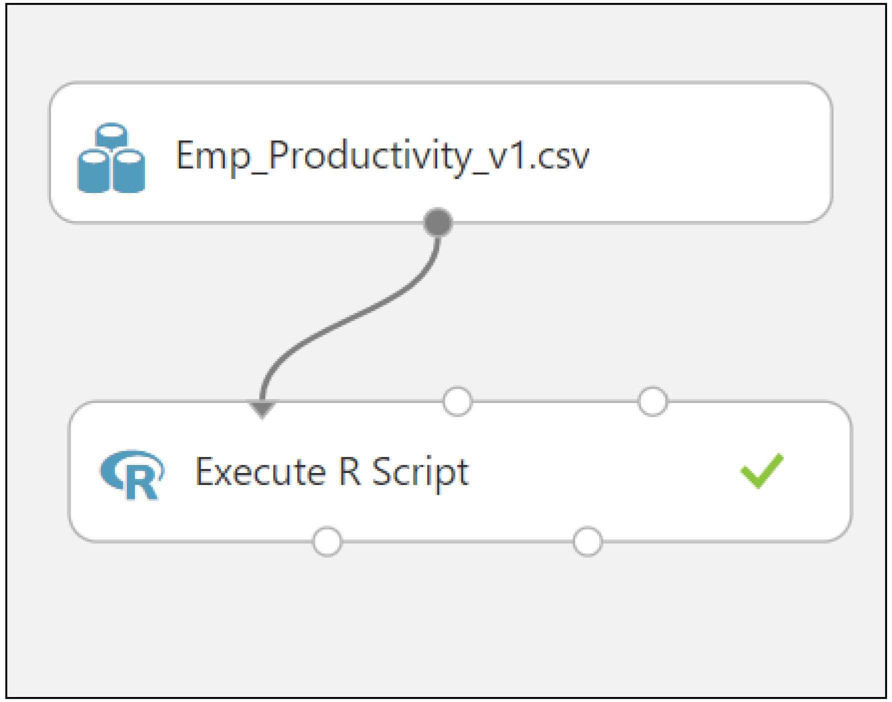
- Dataset: "Emp_Productivity/ Emp_Productivity_v1.csv"
- Draw a scatter plot that shows Age on X axis and Experience on Y-axis. Try to distinguish the two classes with colors or shapes (visualizing the classes)
- Build a logistic regression model to predict Productivity using age and experience
- Finally draw the decision boundary for this logistic regression model
- Create the confusion matrix
- Calculate the accuracy and error rates

Steps - Non-Linear Decision Boundaries

- Drag and drop the Dataset into the canvas
- Drag and drop **Execute R Script** and connect Dataset to the first input circle
- Click on **Execute R Script**, in Properties write the code in the fig-8
- Click on run and visualize the Second Output circle of **Execute R Script**

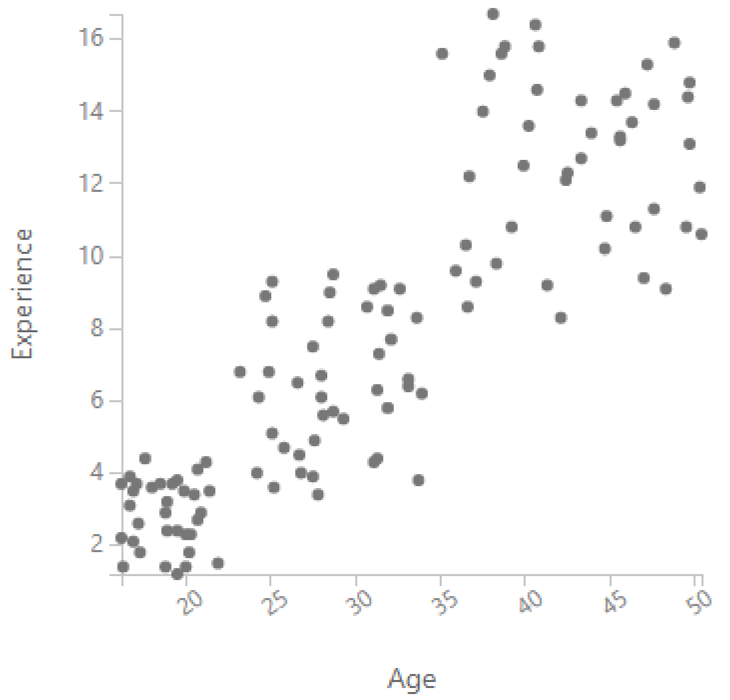
Steps - Non-Linear Decision Boundaries

Fig8: Logistic Regression with R-Script



Steps - Non-Linear Decision Boundaries

Fig9: Scatter Plot – Age vs Experience



Steps - Non-Linear Decision Boundaries

R Script

Fig10: R-Code for Logistic Regression

```r
dataset1 <- maml.mapInputPort(1) # class: data.frame
library(ggplot2)
ggplot(dataset1)+geom_point(aes(x=dataset1$Age,y=dataset1$Experience,color=factor(dataset1$Productivity),
    shape=factor(dataset1$Productivity)),size=5)

Emp_Productivity_logit_overall<-glm(Productivity~Age+Experience,data=dataset1, family=binomial())

slope2 <- coef(Emp_Productivity_logit_overall)[2]/(-coef(Emp_Productivity_logit_overall)[3])
intercept2 <- coef(Emp_Productivity_logit_overall)[1]/(-coef(Emp_Productivity_logit_overall)[3])

library(ggplot2)
base<-ggplot(dataset1)+geom_point(aes(x=dataset1$Age,y=dataset1$Experience,
            color=factor(dataset1$Productivity),shape=factor(dataset1$Productivity)),size=5)
base+geom_abline(intercept = intercept2 , slope = slope2, colour = "blue", size = 2)

predicted_values<-round(predict(Emp_Productivity_logit_overall,type="response"),0)

conf_matrix<-table(predicted_values,Emp_Productivity_logit_overall$y)
conf_matrix

accuracy<-(conf_matrix[1,1]+conf_matrix[2,2])/(sum(conf_matrix))
accuracy
maml.mapOutputPort("dataset1");
```

Steps - Non-Linear Decision Boundaries

Fig11: Accuracy and Confusion Matrix

```
predicted_values   0   1
               0 152  92
               1   5   1
[1] 0.612
```

Confusion Matrix

Accuracy

Steps - Non-Linear Decision Boundaries

Fig12: Scatter plot without Decision boundary(R-Output)

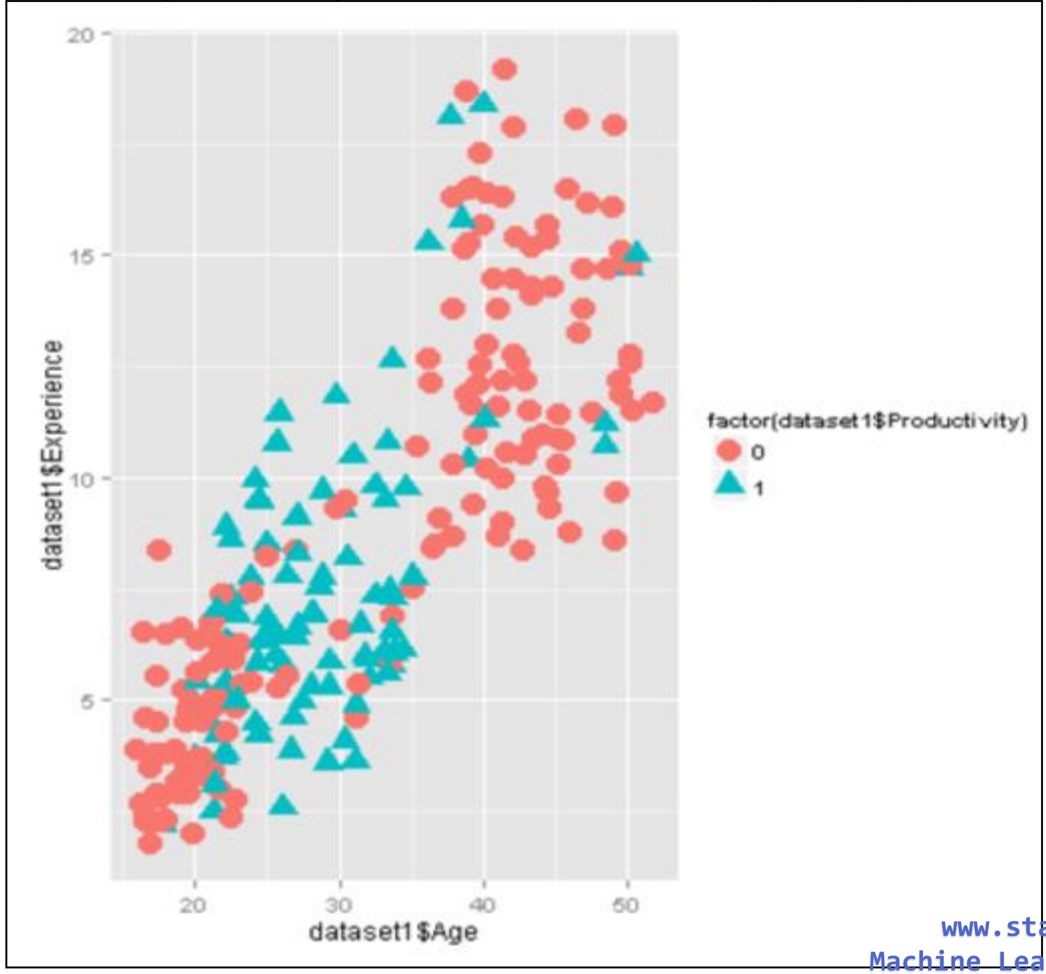
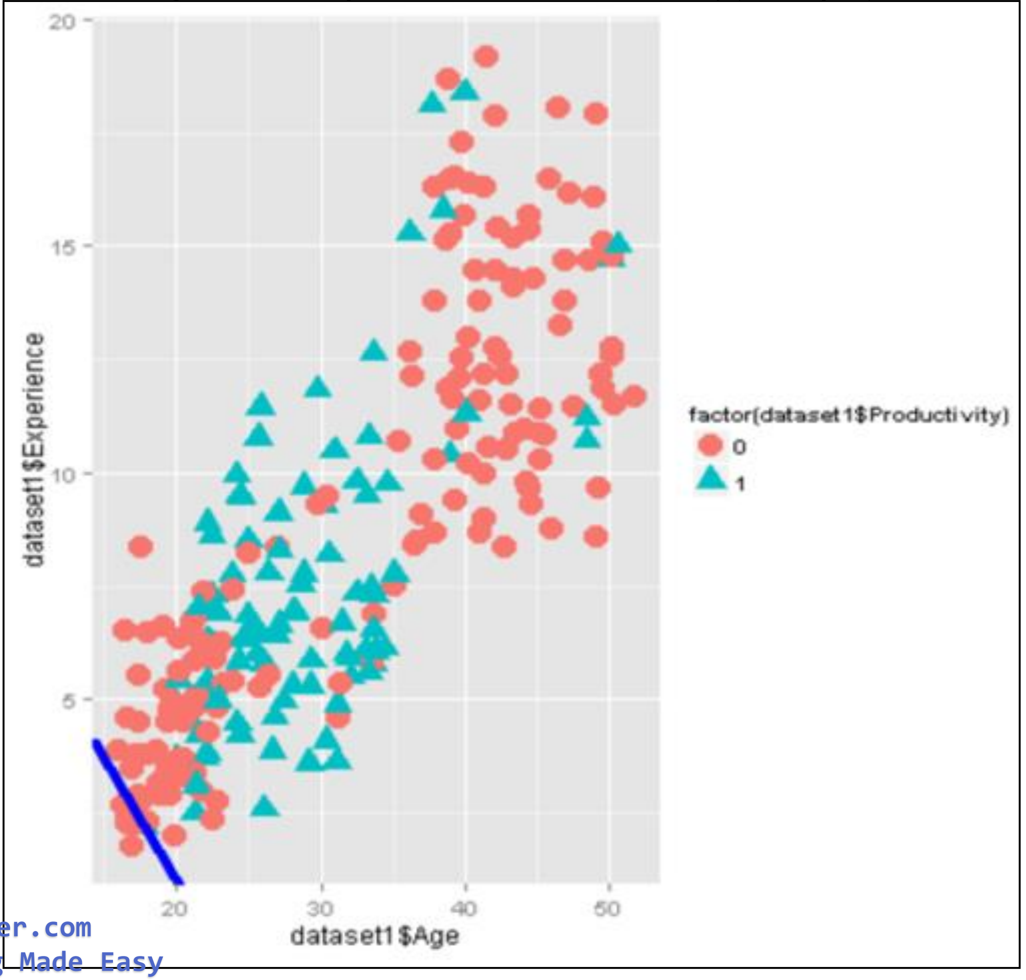
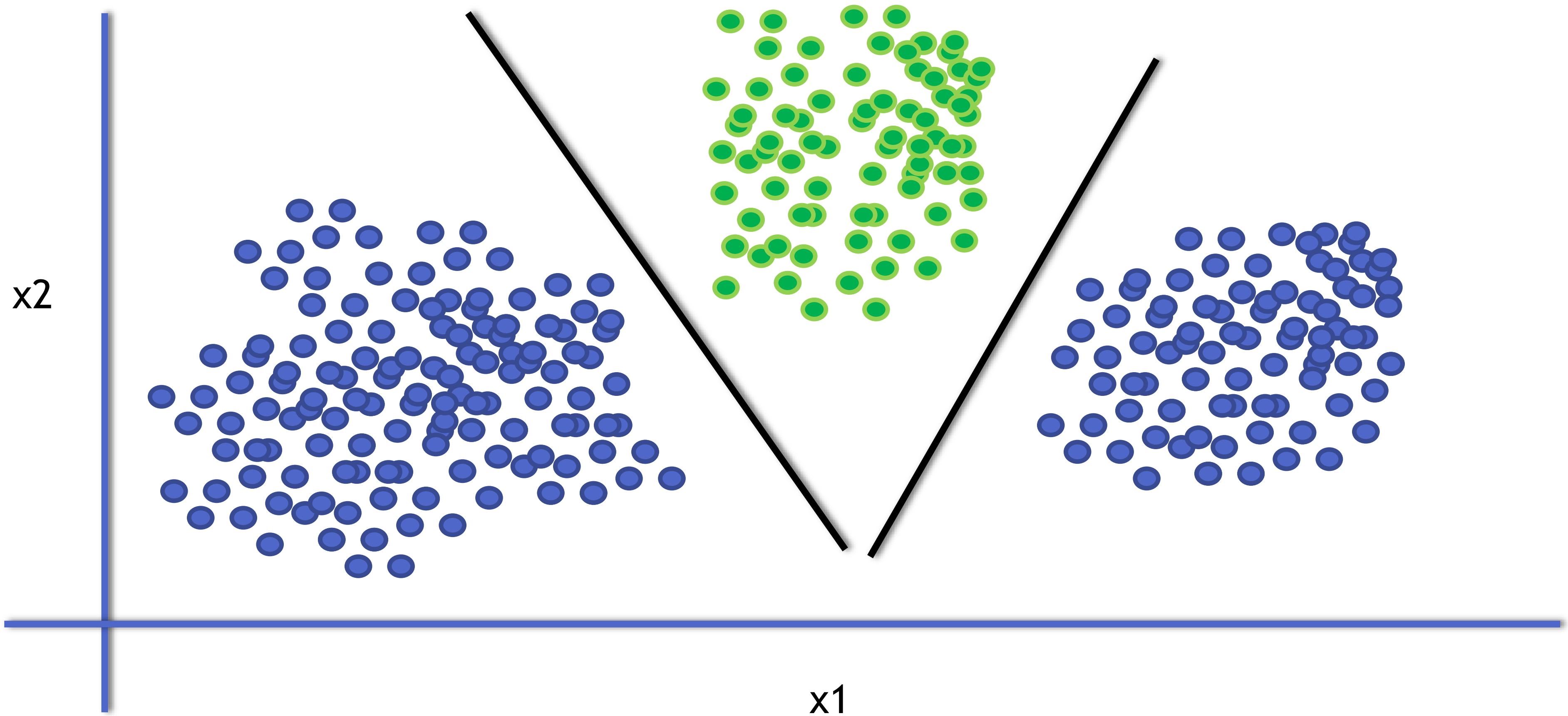


Fig13: Scatter plot with Decision boundary(R-Output)



Non-Linear Decision Boundaries-Issue

Non-Linear Decision Boundaries

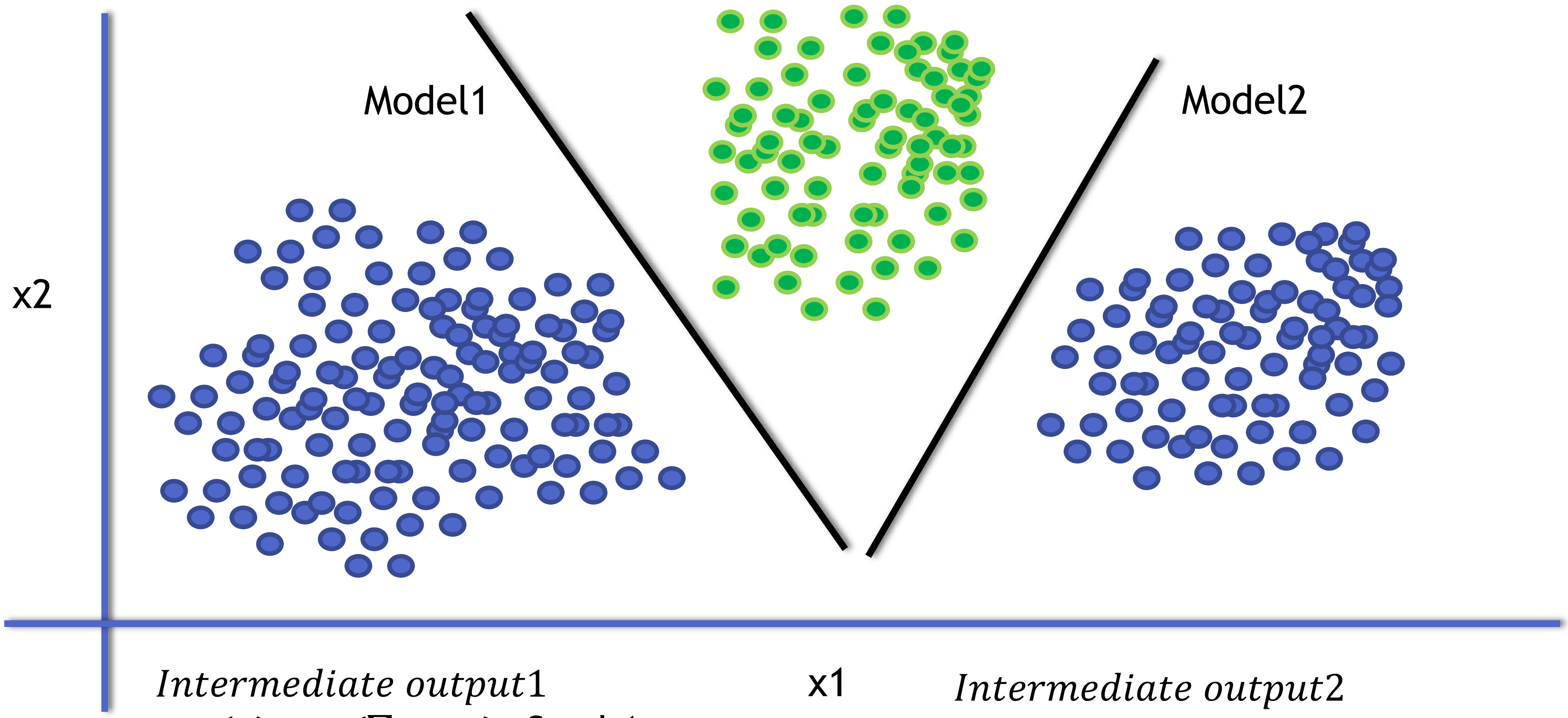


Non-Linear Decision Boundaries-issues

- Logistic Regression line doesn’t seam to be a good option when we have non-linear decision boundaries

Non-Linear Decision Boundaries-Solution

Intermediate outputs

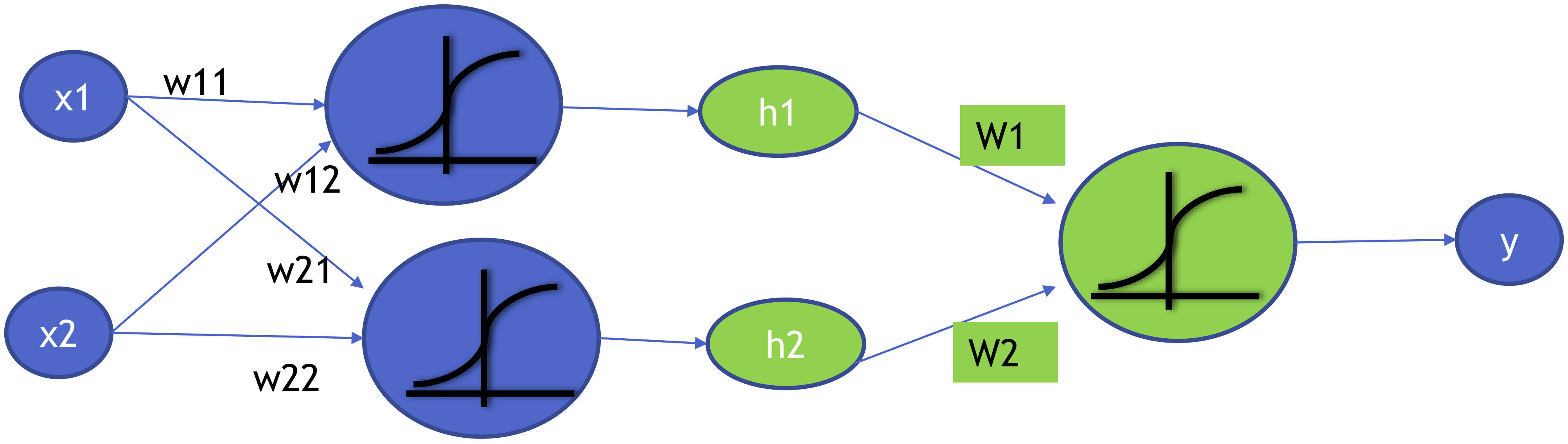


$Intermediate\ output1$
$out(x) = g(\sum w_k x_k)$,Say h1

$Intermediate\ output2$
$out(x) = g(\sum w_k x_k)$, Say h2

The Intermediate output

- Using the x's Directly predicting y is challenging.
- We can predict h, the intermediate output, which will indeed predict Y

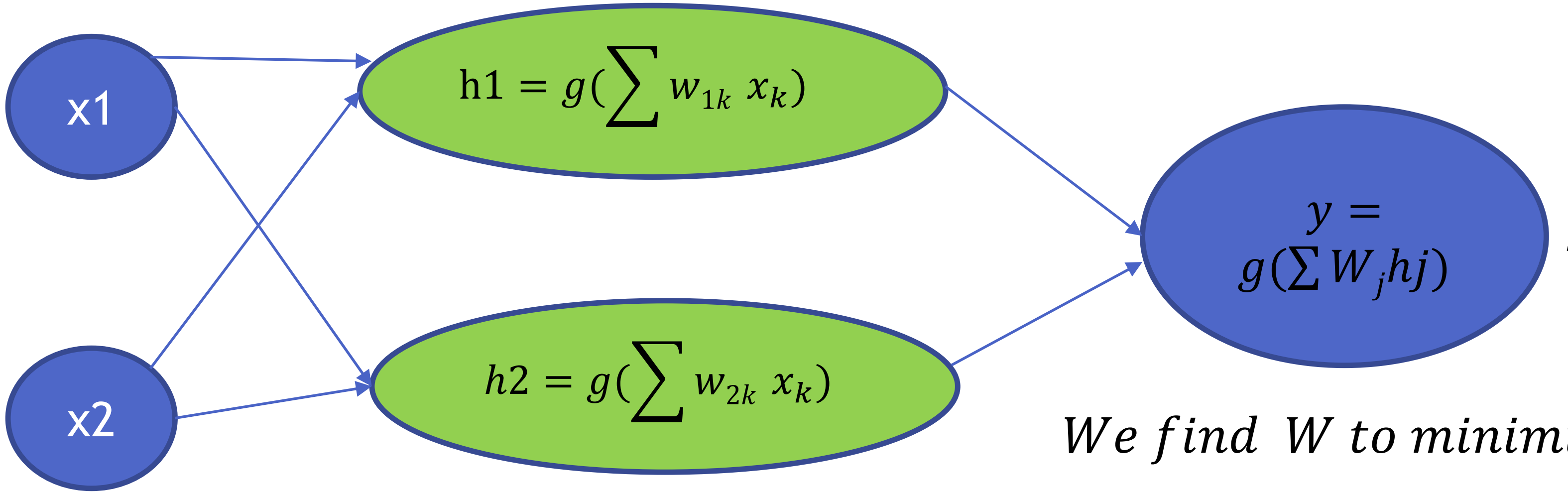


Finding the weights for intermediate outputs

Intermediate output1

$h1 = out(x) = g(\sum w_{1k}\ x_k)$

$We\ find\ \ w_1\ to\ minimize\ \sum_{i=1}^{n}[h_{1i} - g(\sum w_{1k}\ x_k)]^2$



Final output

$y = out(h) = g(\sum W_j hj)$

$We\ find\ \ W\ to\ minimize\ \sum_{i=1}^{n}[y_i - g(\sum W_j h_{ji})]^2$

Intermediate output2

$h2 = out(x) = g(\sum w_{2k}\ x_k)$

$We\ find\ \ w_2\ to\ minimize\ \sum_{i=1}^{n}[h_{2i} - g(\sum w_{1k}\ x_k)]^2$

LAB: Intermediate output

LAB: Intermediate output

- Dataset: Emp_Productivity/ Emp_Productivity_v1.csv
- Filter the data and take first 74 observations from above dataset(Modal1)
- Build a logistic regression model to predict Productivity using age and experience
- Calculate the prediction probabilities for all the inputs. Store the probabilities in inter1 variable
- Filter the data and take observations from row 34 onwards(Modal2)
- Build a logistic regression model to predict Productivity using age and experience
- Calculate the prediction probabilities for all the inputs. Store the probabilities in inter2 variable
- Build a consolidated model to predict productivity using inter-1 and inter-2 variables(Intermediate Modal)
- Create the confusion matrix and find the accuracy and error rates for the consolidated model

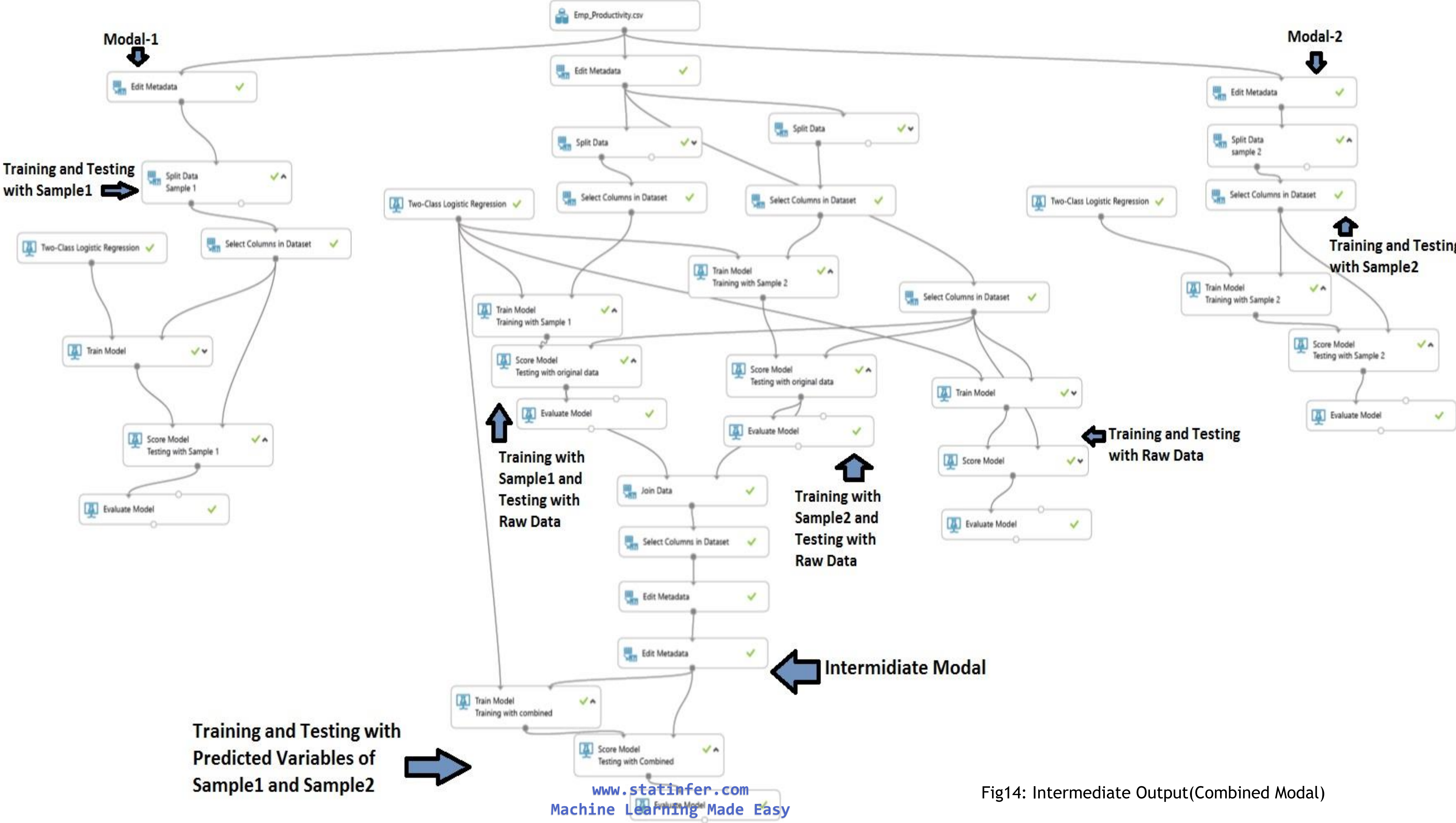


Fig14: Intermediate Output(Combined Modal)

Steps - Intermediate output(Raw Data)

- Drag and drop the **Dataset** into the canvas
- Drag and drop **Edit Metadata** and connect it to the **Dataset**, make Productivity Column as categorical
- Drag and drop the **Select Columns from Dataset, connect it Edit Metadata** and select the columns(Age, Experience, Productivity)
- Drag and drop **Two-Class Logistic Regression, Train Model, Score Model** and **Evaluate Model**
- Connect **Two-Class Boosted Logistic Regression** to the first input of **Train Model** and **Select Columns** to the Second input of **Train Model**

Steps - Intermediate output(Raw Data)

- Connect the output of **Train Model** first input of **Score Model** and **Select Columns** to the Second input of **Score Model**
- Connect the output of **Score Model** to the input of **Evaluate Model**
- Click on **Train Model** and select the column for which the prediction is done(Productivity)
- Click run and visualize the output of **Evaluate Model**

Steps - Intermediate output(Modal1)

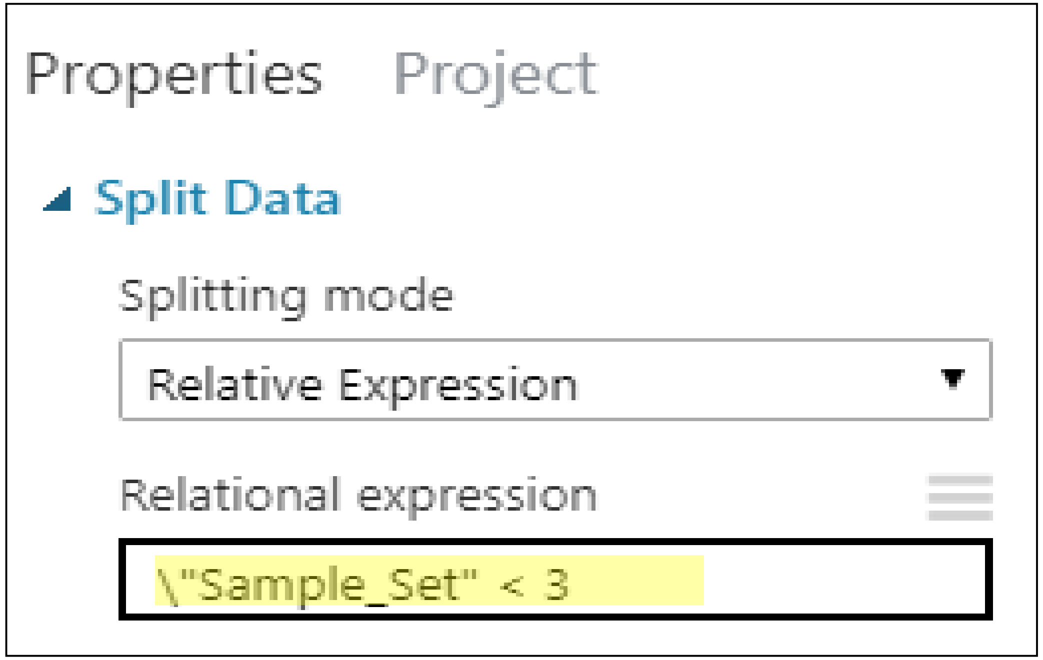
- Drag and drop the **Dataset** into the canvas
- Drag and drop **Edit Metadata** and connect it to the **Dataset**, make Productivity Column as categorical
- Drag and drop the **Split Data** and connect it to the Edit Metadata
- In Split Data properties, select
  - Mode → Relative Expression
  - Expression → \"Sample_Set" < 3
- Drag and drop the **Select Columns from Dataset,** connect it to first output of **Split Data** and select the columns(Age, Experience, Productivity)
- Drag and drop **Two-Class Logistic Regression, Train Model, Score Model** and **Evaluate Model**
- Connect **Two-Class Boosted Logistic Regression** to the first input of **Train Model** and **Select Columns** to the Second input of **Train Model**

Steps - Intermediate output(Modal1)

- Connect the output of **Train Model** first input of **Score Model** and **Select Columns** to the Second input of **Score Model**
- Connect the output of **Score Model** to the input of **Evaluate Model**
- Click on **Train Model** and select the column for which the prediction is done(Productivity)
- Click run and visualize the output of **Evaluate Model**

Steps - Intermediate output(Modal1)

Fig15: Split Data(Modal1)



Steps - Intermediate output(Modal2)

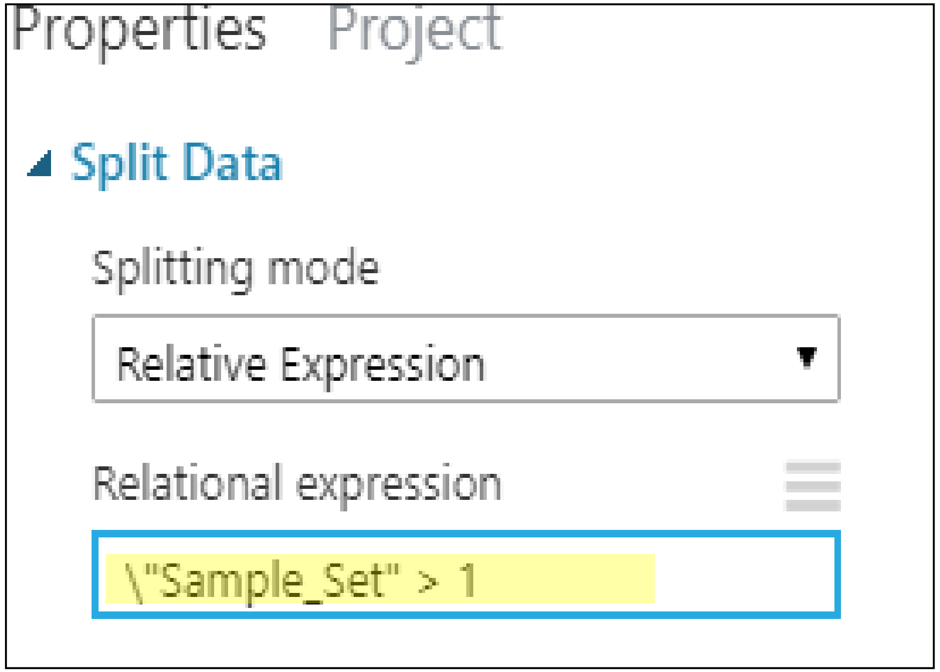
- Drag and drop the **Dataset** into the canvas
- Drag and drop **Edit Metadata** and connect it to the dataset, make Productivity Column as categorical
- Drag and drop the **Split Data** and connect it to the **Edit Metadata**
- In **Split Data** properties, select
  - Mode → Relative Expression
  - Expression → \"Sample_Set" > 1
- Drag and drop the **Select Columns from Dataset** connect it to the first out put of the **Split Data** and select the columns(Age, Experience, Productivity)
- Drag and drop **Two-Class Logistic Regression, Train Model, Score Model** and **Evaluate Model**
- Connect **Two-Class Boosted Logistic Regression** to the first input of **Train Model** and **Select Columns from Dataset** to the Second input of **Train Model**

Steps - Intermediate output(Modal2)

- Connect the output of **Train Model** first input of **Score Model** and **Select Columns** to the Second input of **Score Model**
- Connect the output of **Score Model** to the input of **Evaluate Model**
- Click on **Train Model** and select the column for which the prediction is done(Productivity)
- Click run and visualize the output of **Evaluate Model**

Steps - Intermediate output(Modal2)

Fig16: Split Data(Modal2)



Steps - Intermediate output(Combined)

- Create Modal1 and Modal2, test it with the Raw Data by passing it to the **Score Modal**
- Drag and drop **Join Data** connect the output of **Score Modal of** Modal1 and Modal2 and select the properties as in figure
- Drag and drop **Select column from Dataset** connect it to the **Join Data** and select the columns(Productivity, Scored Labels, Scored Labels (2))
- Drag and drop **Edit Metadata** connect it to the **Select column from Dataset** and select the properties as in figure
- Drag and drop **Edit Metadata** connect it to previous **Edit Metadata** and select the properties as in figure
- Drag and drop **Two-Class Logistic Regression, Train Model, Score Model** and **Evaluate Model**
- Connect **Two-Class Boosted Logistic Regression** to the first input of **Train Model** and second **Edit Metadata** to the Second input of **Train Model**

Steps - Intermediate output(Combined)

- Connect the output of **Train Model** first input of **Score Model** and second **Edit Metadata** to the Second input of **Score Model**
- Connect the output of **Score Model** to the input of **Evaluate Model**
- Click on **Train Model** and select the column for which the prediction is done(Productivity)
- Click run and visualize the output of **Evaluate Model**

Steps - Intermediate output

Fig17: Properties - Edit Metadata
(common to all before join data)

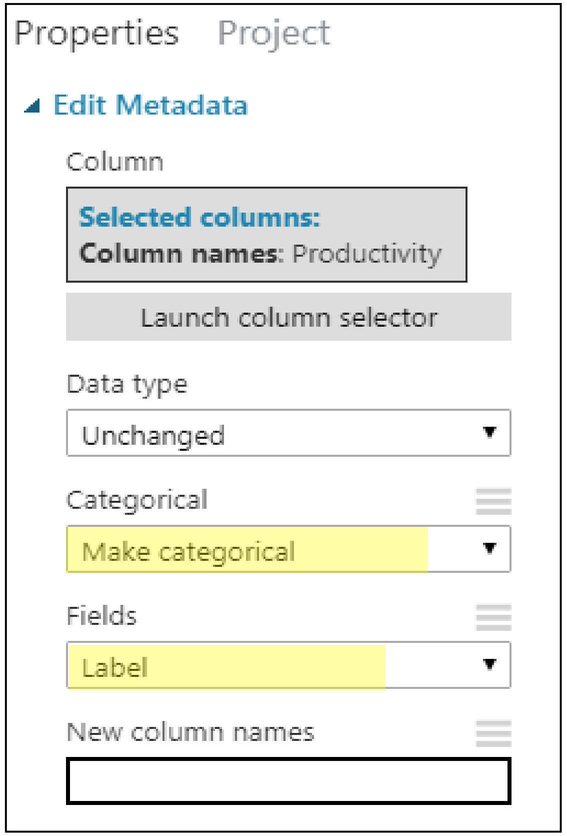
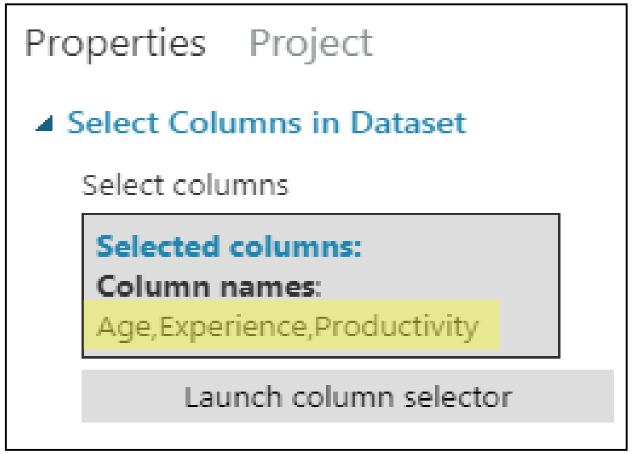


Fig18: Properties - Select Columns in Dataset
(common to all before join data)



Steps - Intermediate output

Fig19: Properties - Two-Class Logistic Regression
(common to all)

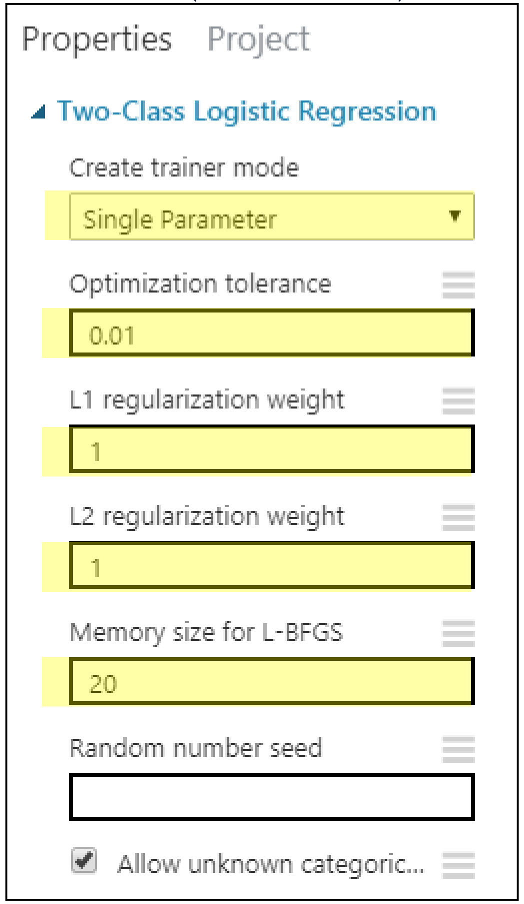
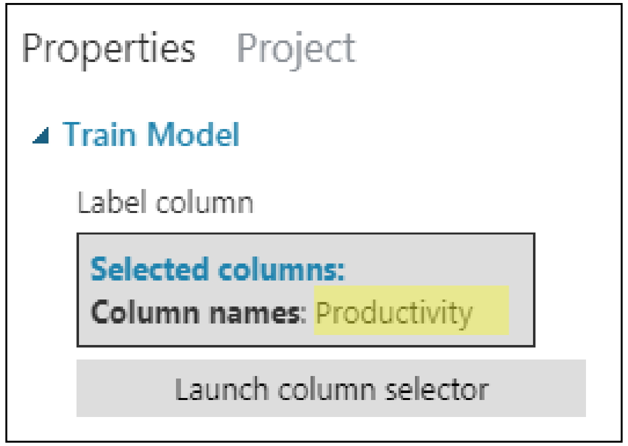


Fig20: Properties - Train Model
(common to all)



Steps - Intermediate output

Fig21: Properties - Join Data

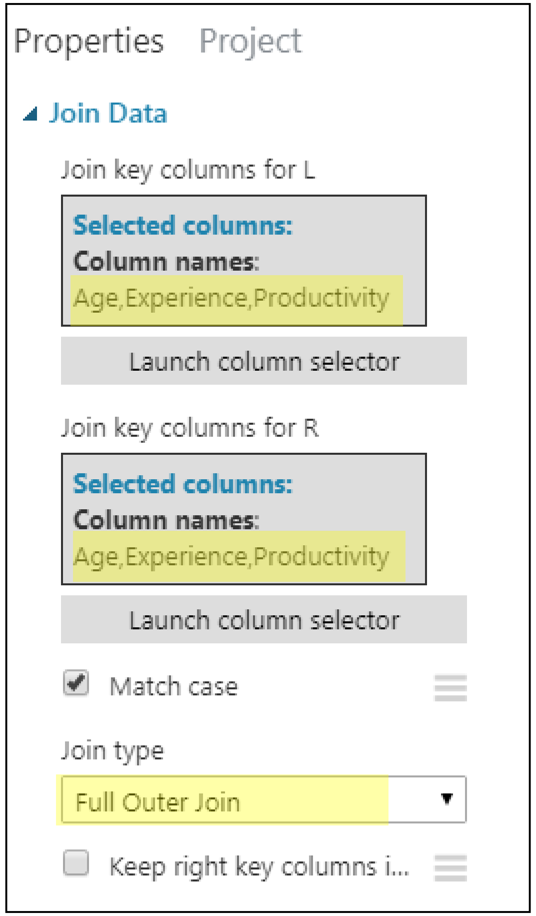
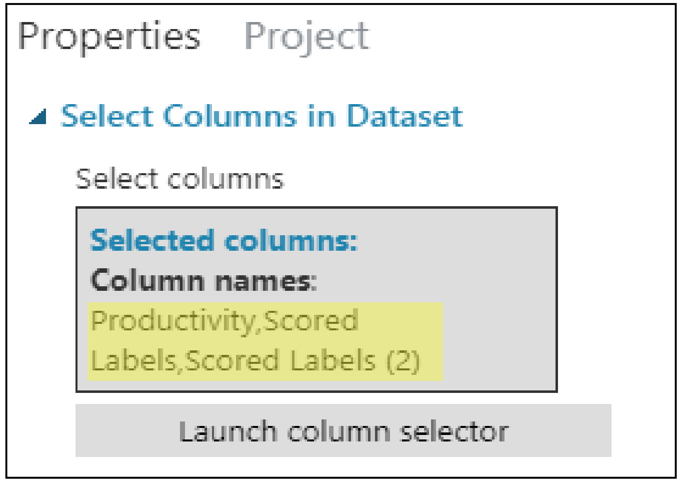


Fig22: Properties - Select Columns in Dataset(After Join Data)



Steps - Intermediate output

Fig23: Properties - Edit Metadata
(First after Join Data)

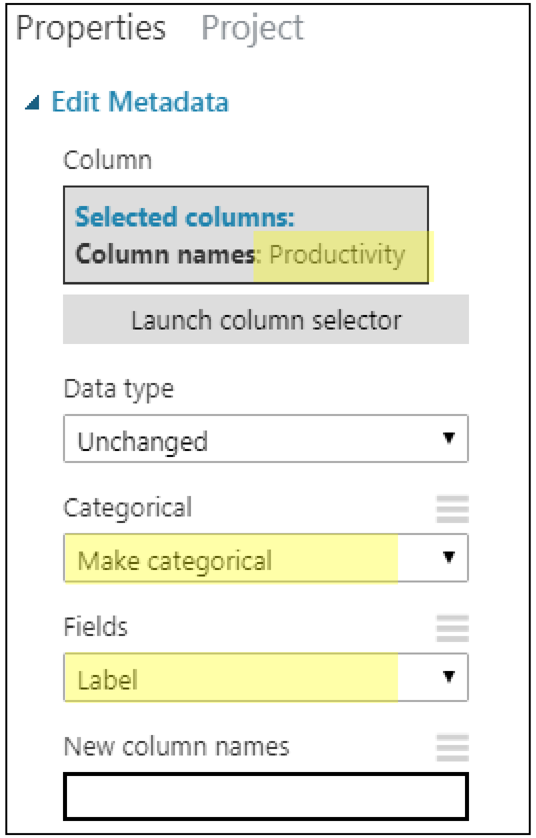
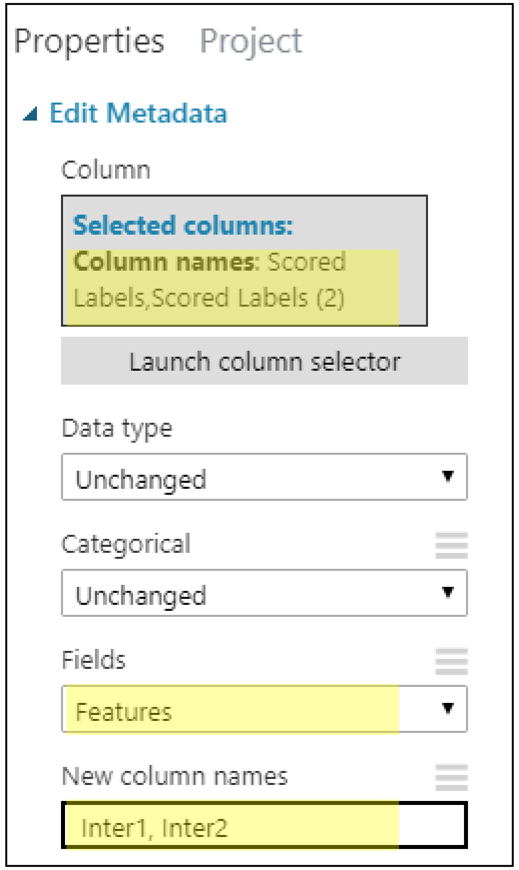


Fig24: Properties - Edit Metadata
(Second after Join Data)



Steps - Intermediate output

Fig25: Accuracy(Train and Test with Modal1)

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
39	2	0.946	0.951	0.5	0.922
False Positive	**True Negative**	**Recall**	**F1 Score**		
2	31	0.951	0.951		
Positive Label	**Negative Label**				
1	0				

Fig26: Accuracy(Train and Test with Modal2)

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
39	2	0.942	0.929	0.5	0.969
False Positive	**True Negative**	**Recall**	**F1 Score**		
3	42	0.951	0.940		
Positive Label	**Negative Label**				
1	0				

Fig27: Accuracy(Train with Modal1 and Test with Raw Data)

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
41	2	0.605	0.477	0.5	0.415
False Positive	**True Negative**	**Recall**	**F1 Score**		
45	31	0.953	0.636		
Positive Label	**Negative Label**				
1	0				

Fig28: Accuracy(Train with Modal2 and Test with Raw Data)

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
41	2	0.697	0.547	0.5	0.585
False Positive	**True Negative**	**Recall**	**F1 Score**		
34	42	0.953	0.695		
Positive Label	**Negative Label**				
1	0				

Fig29: Accuracy(Train and Test with Raw Data)

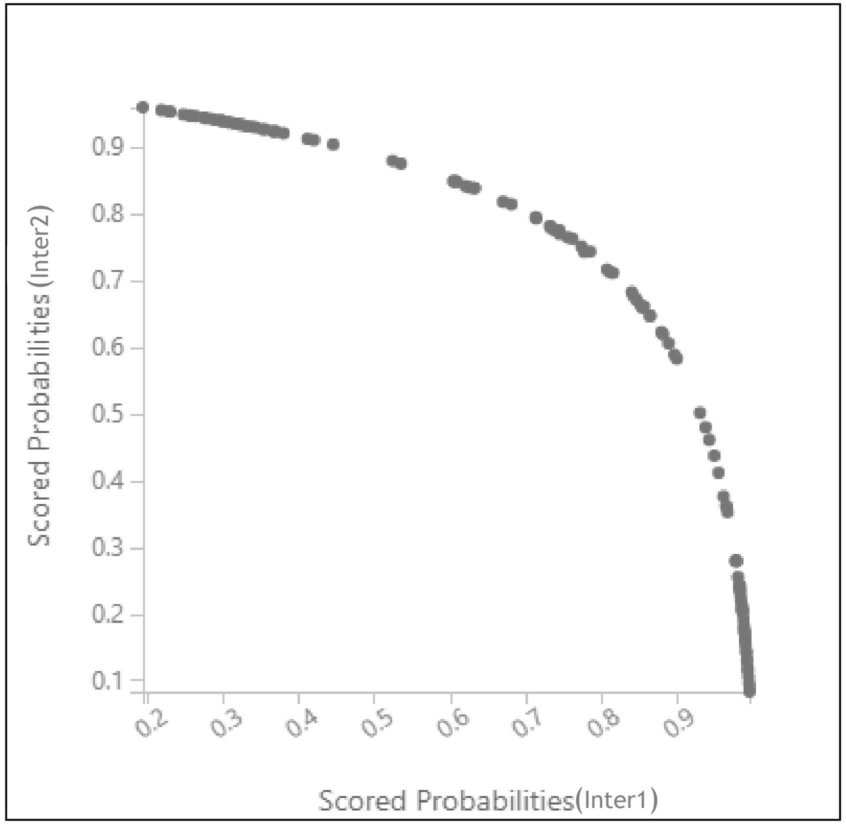
True Positive	False Negative	Accuracy	Precision	Threshold	AUC
0	43	0.639	1.000	0.5	0.585
False Positive	**True Negative**	**Recall**	**F1 Score**		
0	76	0.000	0.000		
Positive Label	**Negative Label**				
1	0				

Fig30: Accuracy(Train and Test with Intermediate Data)

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
39	4	0.941	0.929	0.5	0.937
False Positive	**True Negative**	**Recall**	**F1 Score**		
3	73	0.907	0.918		
Positive Label	**Negative Label**				
1	0				

Steps - Intermediate output

Fig31: Scatter Plot - Scored Probabilities of Inter1 vs Scored Probabilities of Inter1



statinfer

Neural Network intuition

Neural Network intuition

$$\textit{Final output}$$
$$y = out(h) = g(\sum W_j h_j)$$

$$h_j = out(x) = g(\sum w_{jk} x_k)$$

$$y = out(h) = g(\sum W_j g(\sum w_{jk} x_k))$$

- So h is a non linear function of linear combination of inputs – A multiple logistic regression line
- Y is a non linear function of linear combination of outputs of logistic regressions
- Y is a non linear function of linear combination of non linear functions of linear combination of inputs

$$We\ find\ \ W\ to\ minimize\ \textstyle\sum_{i=1}^{n}[y_i - g(\sum W_j h_j)]^2$$

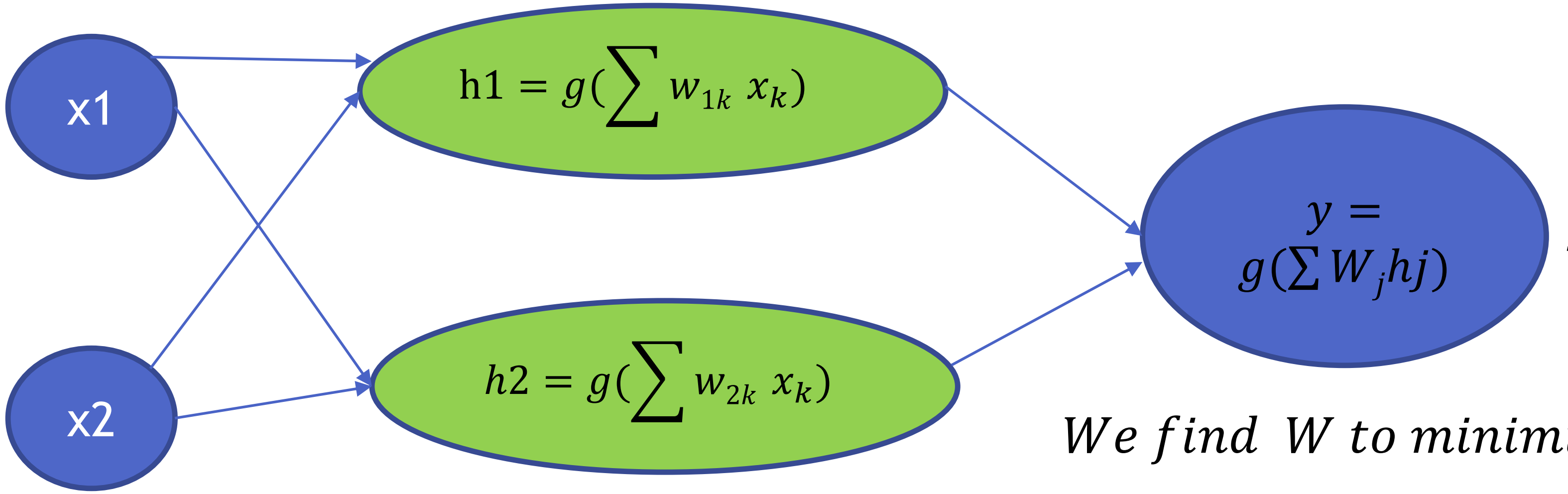
$$We\ find\ \{W_j\}\ \&\ \{w_{jk}\}\ to\ minimize\ \textstyle\sum_{i=1}^{n}[y_i - g(\sum W_j g(\sum w_{jk} x_k))]^2$$

Neural networks is all about finding the sets of weights {Wj,} and $\{w_{jk}\}$ using **Gradient Descent Method**

Neural Network intuition

Intermediate output1
$h1 = out(x) = g(\sum w_{1k}\ x_k)$

$$We\ find\ \ w_1\ to\ minimize\ \sum_{i=1}^{n}[h_{1i} - g(\sum w_{1k}\ x_k)]^2$$



Final output
$y = out(h) = g(\sum W_j hj)$

$$We\ find\ \ W\ to\ minimize\ \sum_{i=1}^{n}[y_i - g(\sum W_j h_{ji})]^2$$

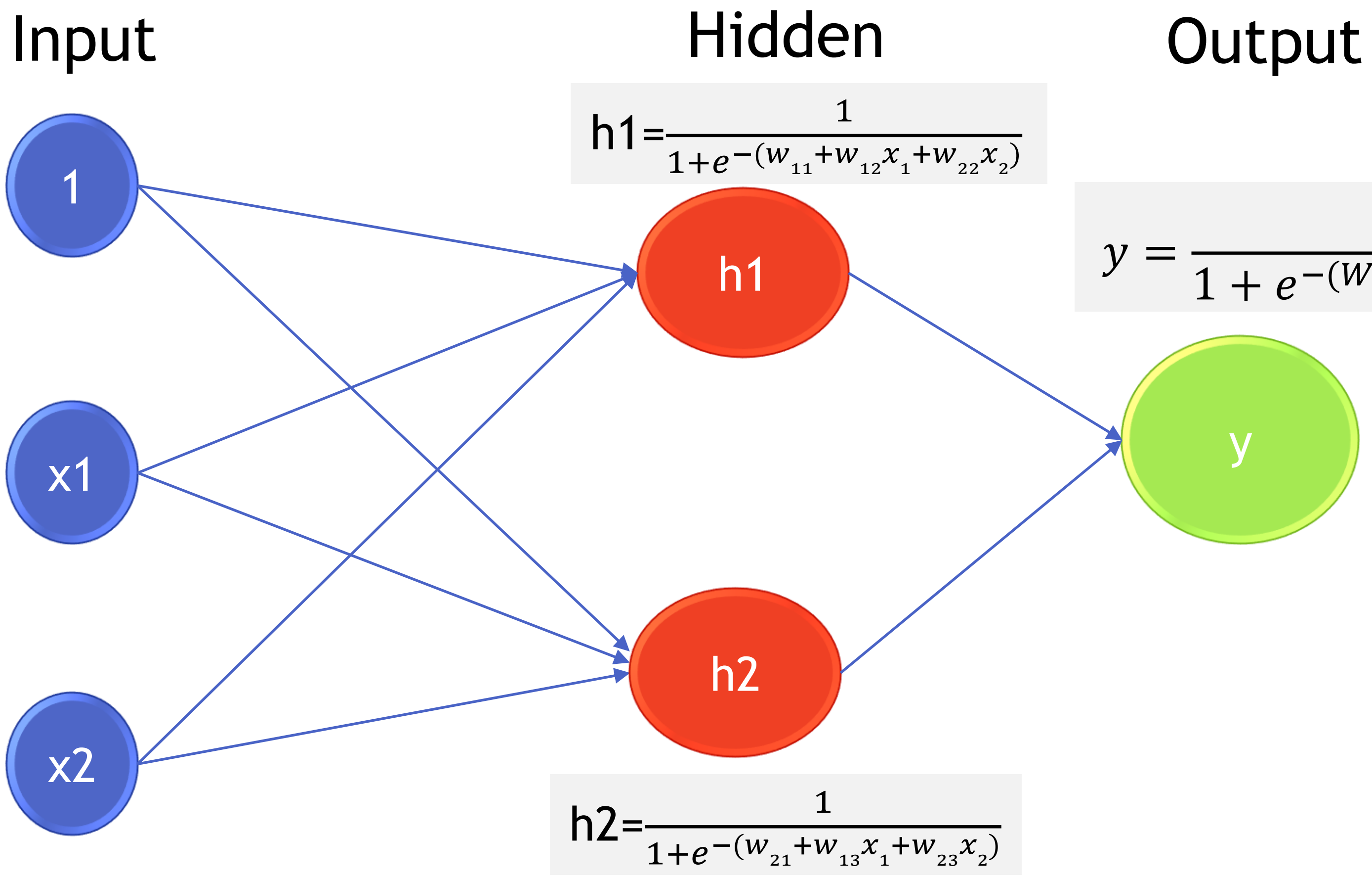
Intermediate output2
$h2 = out(x) = g(\sum w_{2k}\ x_k)$

$$We\ find\ \ w_2\ to\ minimize\ \sum_{i=1}^{n}[h_{2i} - g(\sum w_{1k}\ x_k)]^2$$

The Neural Networks

- The neural networks methodology is similar to the intermediate output method explained above.
- But we will not manually subset the data to create the different models.
- The neural network technique automatically takes care of all the intermediate outputs using hidden layers
- It works very well for the data with non-linear decision boundaries
- The intermediate output layer in the network is known as hidden layer
- In Simple terms, neural networks are multi layer nonlinear regression models.
- If we have sufficient number of hidden layers, then we can estimate any complex non-linear function

Neural network and vocabulary



$$y = \frac{1}{1+e^{-(W0+W1h1+W2h2)}}$$

- X1,X2 →inputs
- 1 → bias term
- W’s are weights
- 1/(1+e^{-u}) is the sigmoid function
- Y is output

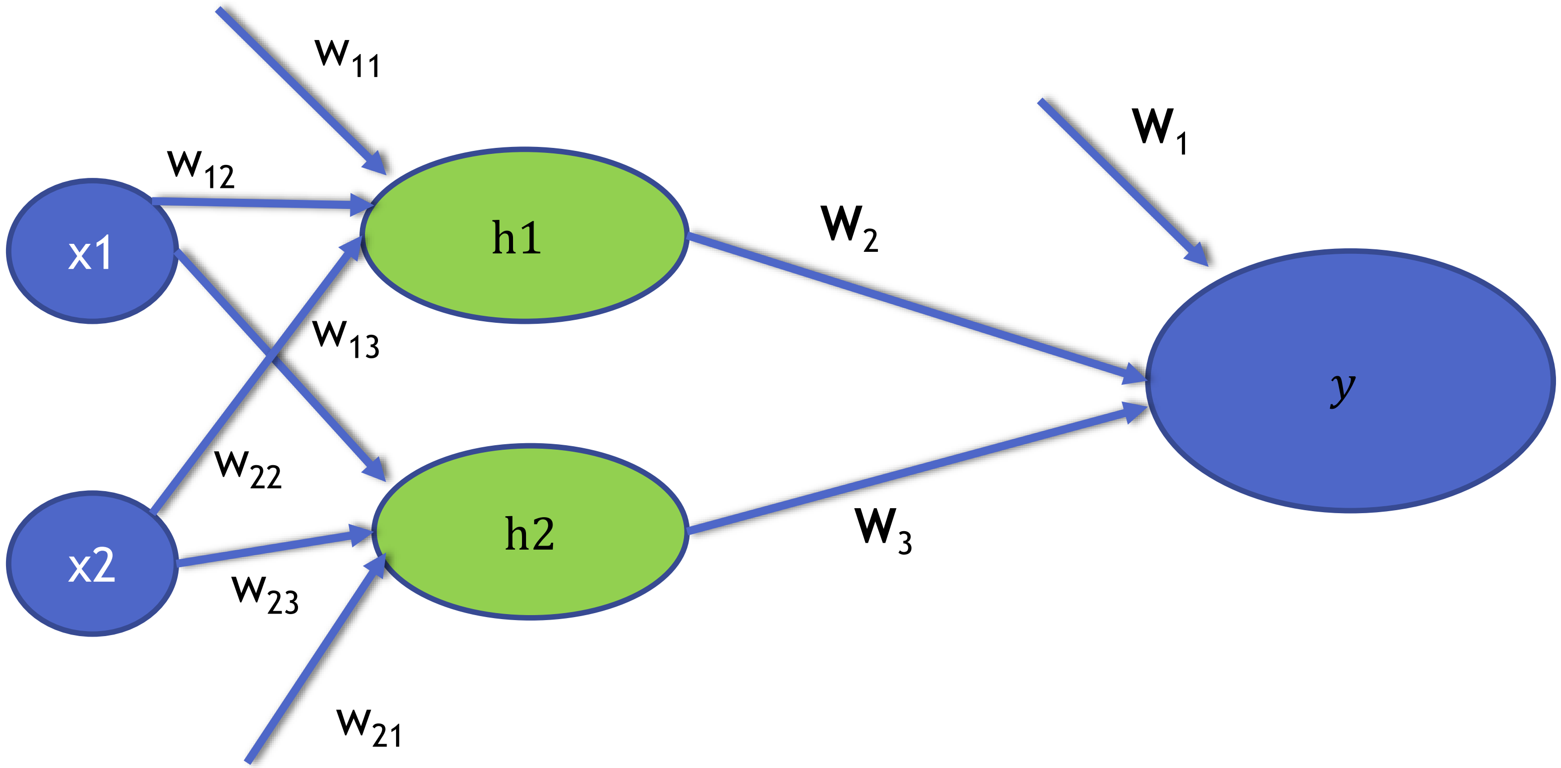
Why are they called hidden layers?

- A hidden layer “hides” the desired output.
- Instead of predicting the actual output using a single model, build multiple models to predict intermediate output
- There is no standard way of deciding the number of hidden layers.

statinfer

The Neural network Algorithm

Algorithm for Finding weights

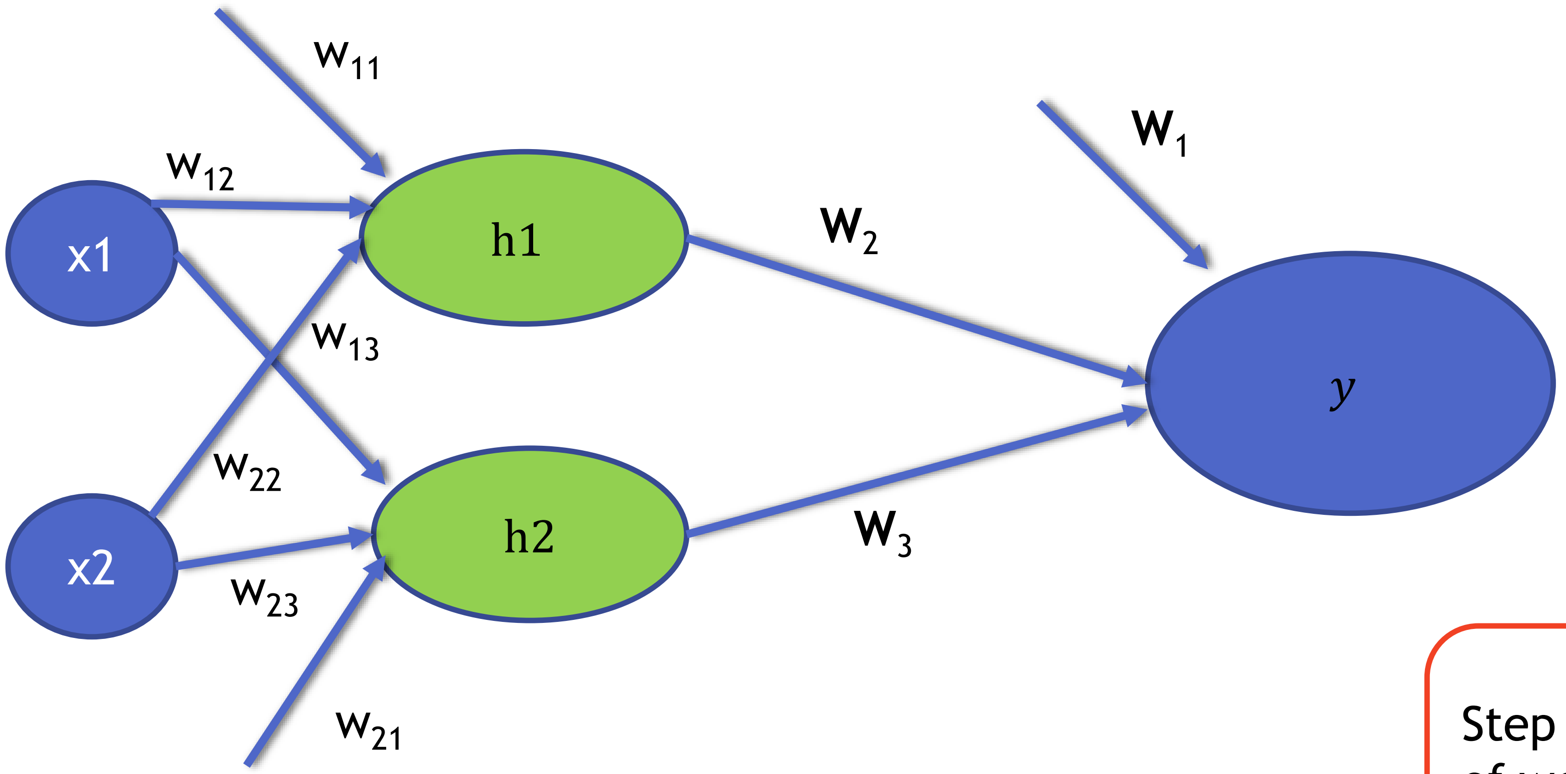


- Algorithm is all about finding the weights/coefficients
- We randomly initialize some weights; Calculate the output by supplying training input; If there is an error the weights are adjusted to reduce this error.

The Neural Network Algorithm

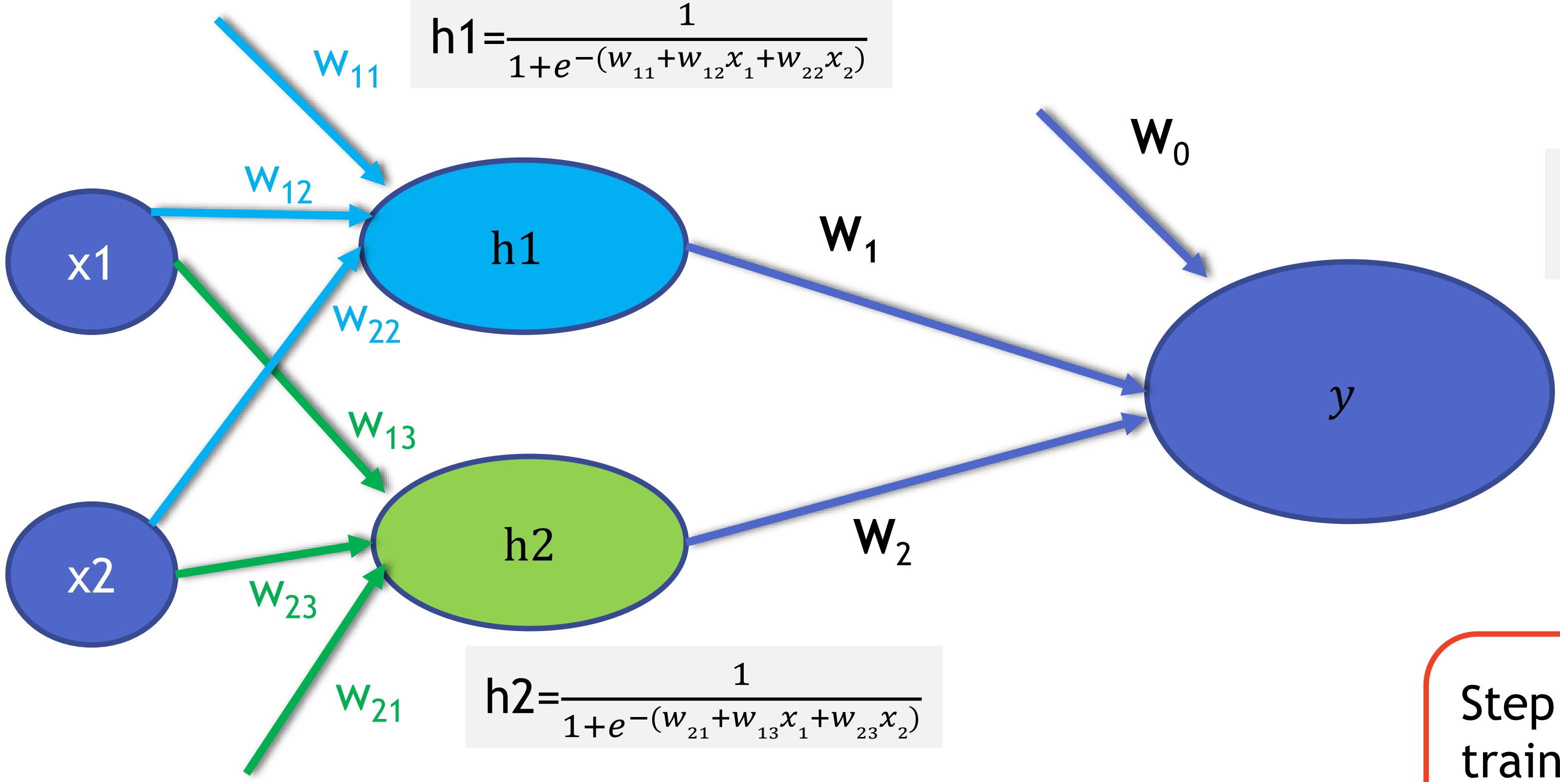
- **Step 1: Initialization of weights:** Randomly select some weights
- **Step 2 : Training & Activation:** Input the training values and perform the calculations forward.
- **Step 3 : Error Calculation:** Calculate the error at the outputs. Use the output error to calculate error fractions at each hidden layer
- **Step 4: Weight training :** Update the weights to reduce the error, recalculate and repeat the process of training & updating the weights for all the examples.
- **Step 5: Stopping criteria:** Stop the training and weights updating process when the minimum error criteria is met

Randomly initialize weights



Step 1: Initialization of weights: Randomly select some weights

Training & Activation



$$h1=\frac{1}{1+e^{-(w_{11}+w_{12}x_1+w_{22}x_2)}}$$

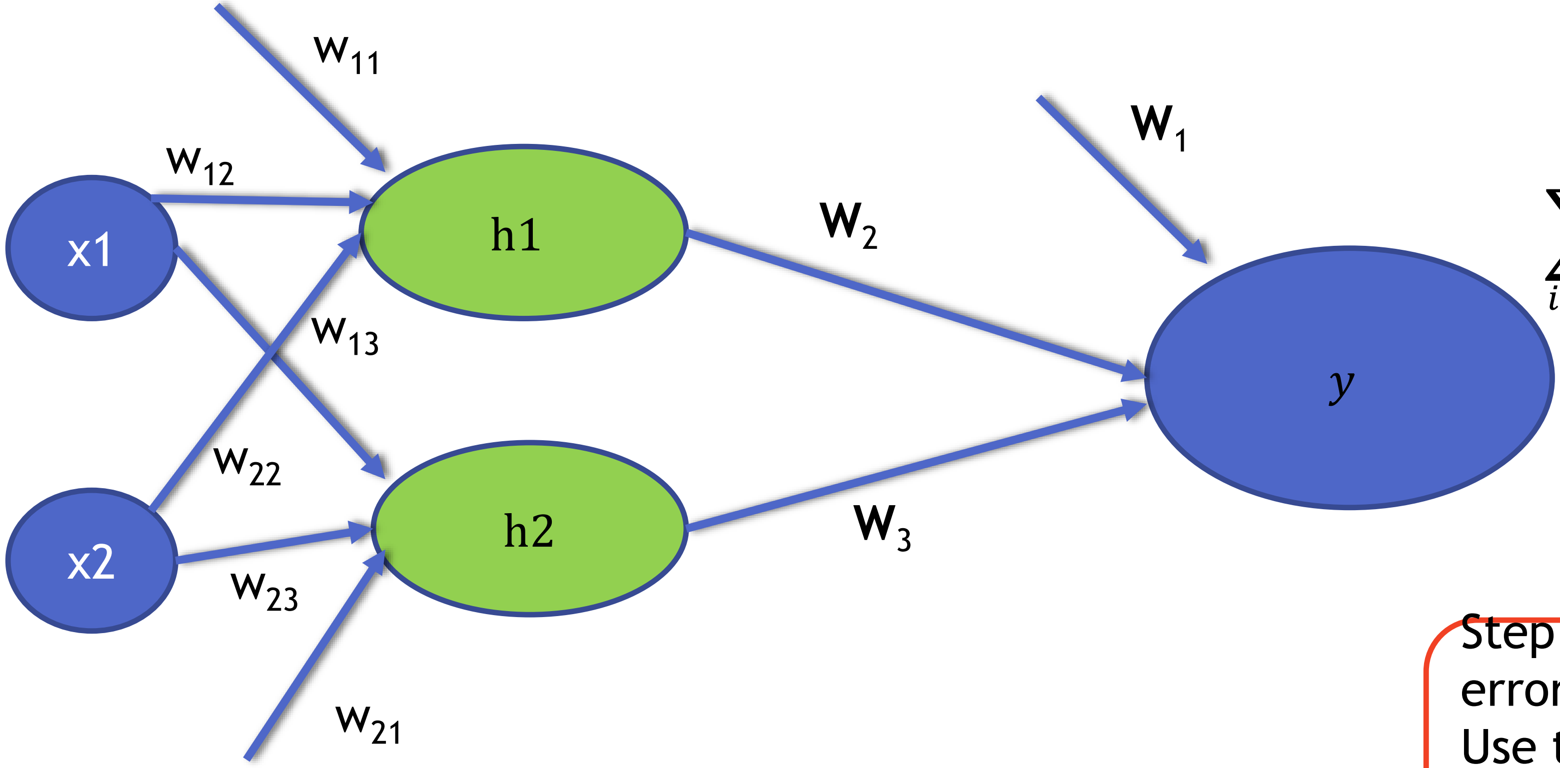
$$h2=\frac{1}{1+e^{-(w_{21}+w_{13}x_1+w_{23}x_2)}}$$

$$y=\frac{1}{1+e^{-(W0+W1h1+W2h2)}}$$

Step 2 : Input the training values and perform the calculations forward

Training input & calculations - **Feed Forward**

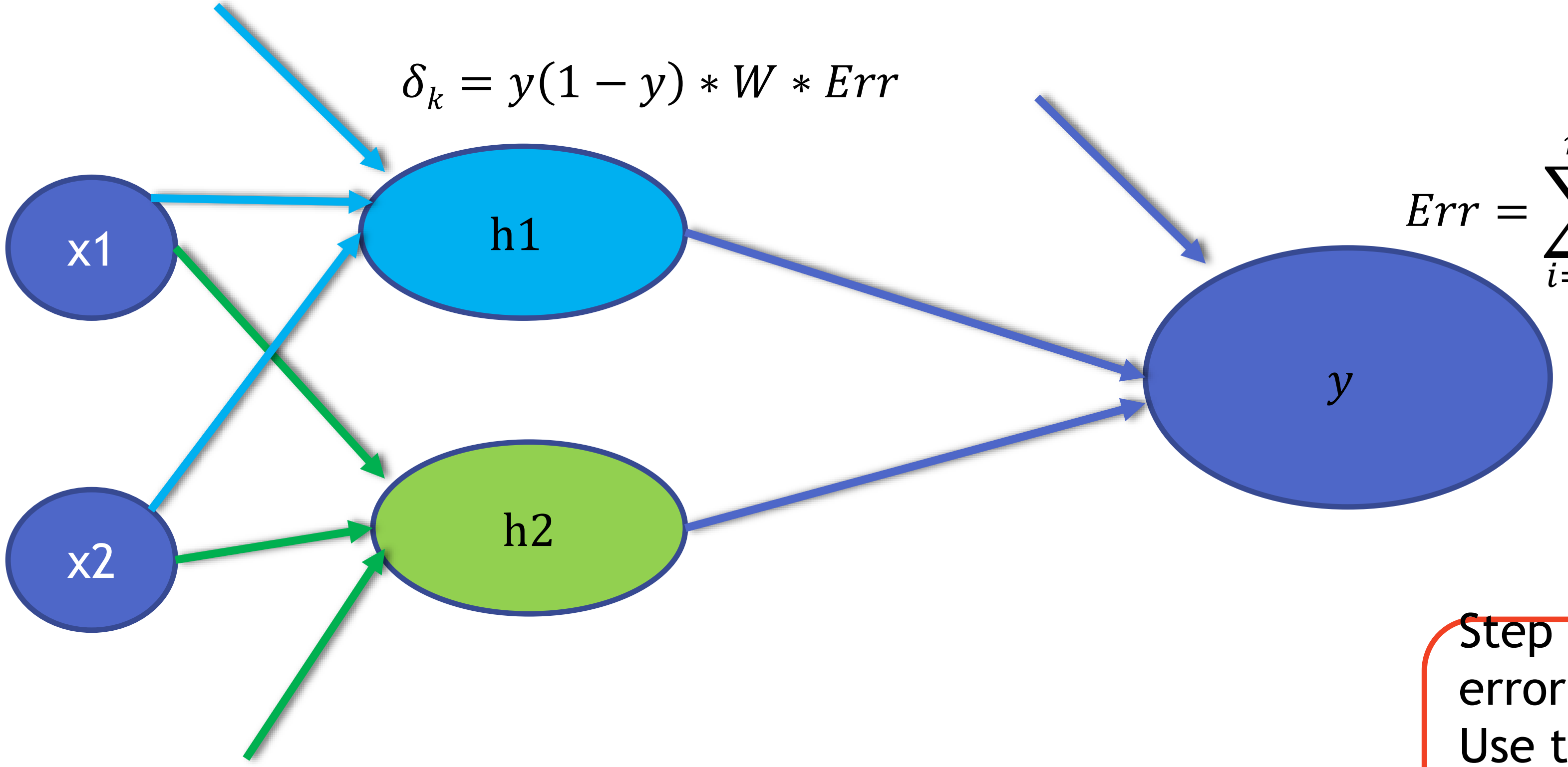
Error Calculation at Output



$$\sum_{i=1}^{n}\left[y_i - g\left(\sum_{k=1}^{m} w_k h_{ki}\right)\right]^2$$

Step 3: Calculate the error at the outputs. Use the output error to calculate error fractions at each hidden layer

Error Calculation at hidden layers



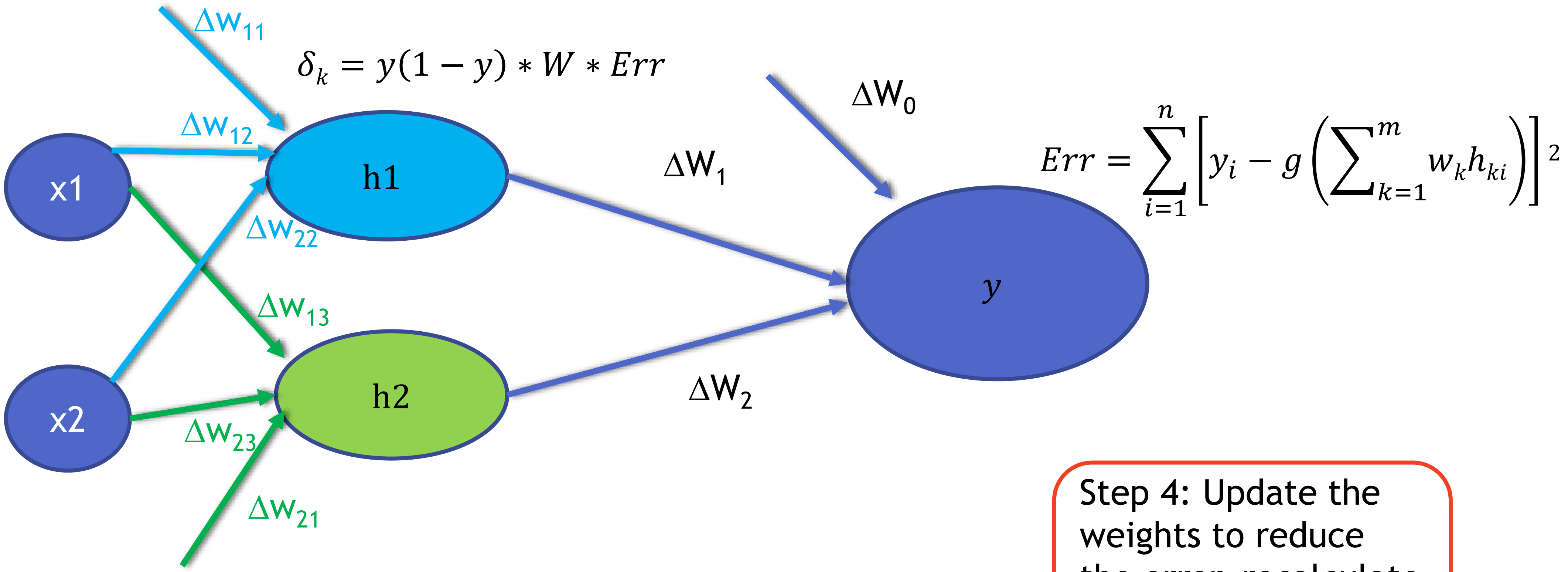
$$\delta_k = y(1-y) * W * Err$$

$$Err = \sum_{i=1}^{n}\left[y_i - g\left(\sum_{k=1}^{m} w_k h_{ki}\right)\right]^2$$

Back Propagation - Calculate Errors signals backwards

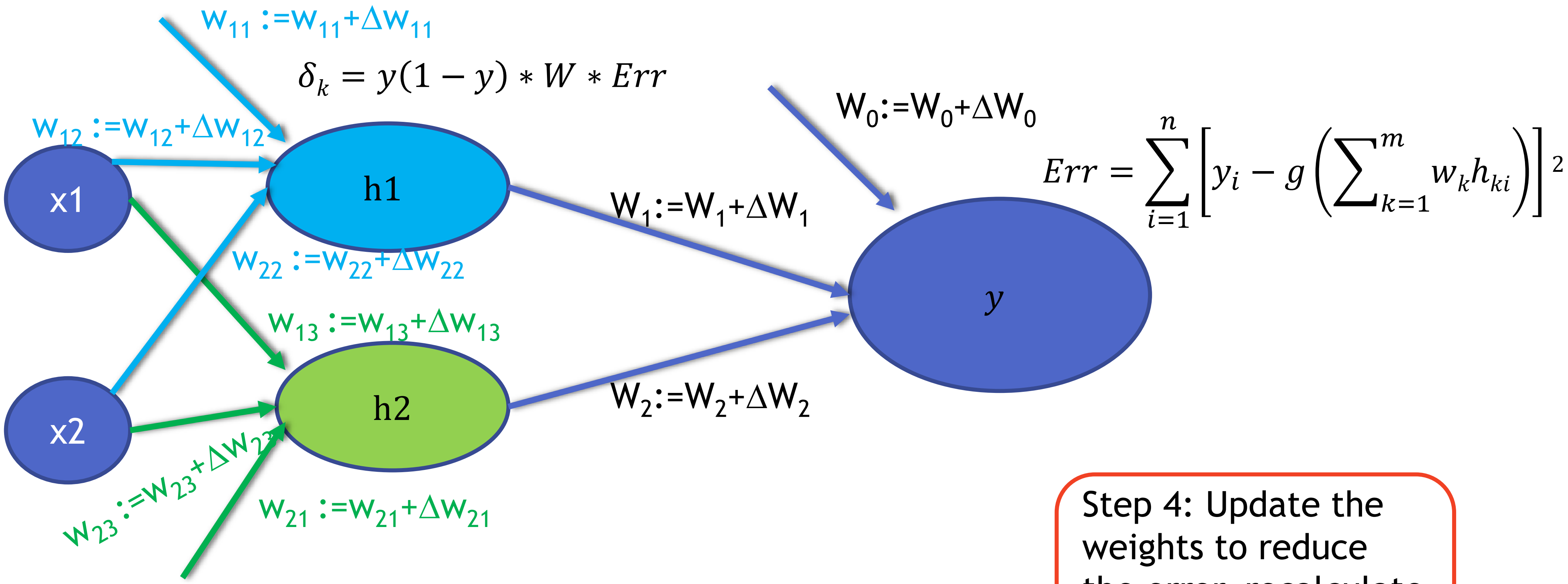
Step 3: Calculate the error at the outputs. Use the output error to calculate error fractions at each hidden layer

Calculate weight corrections



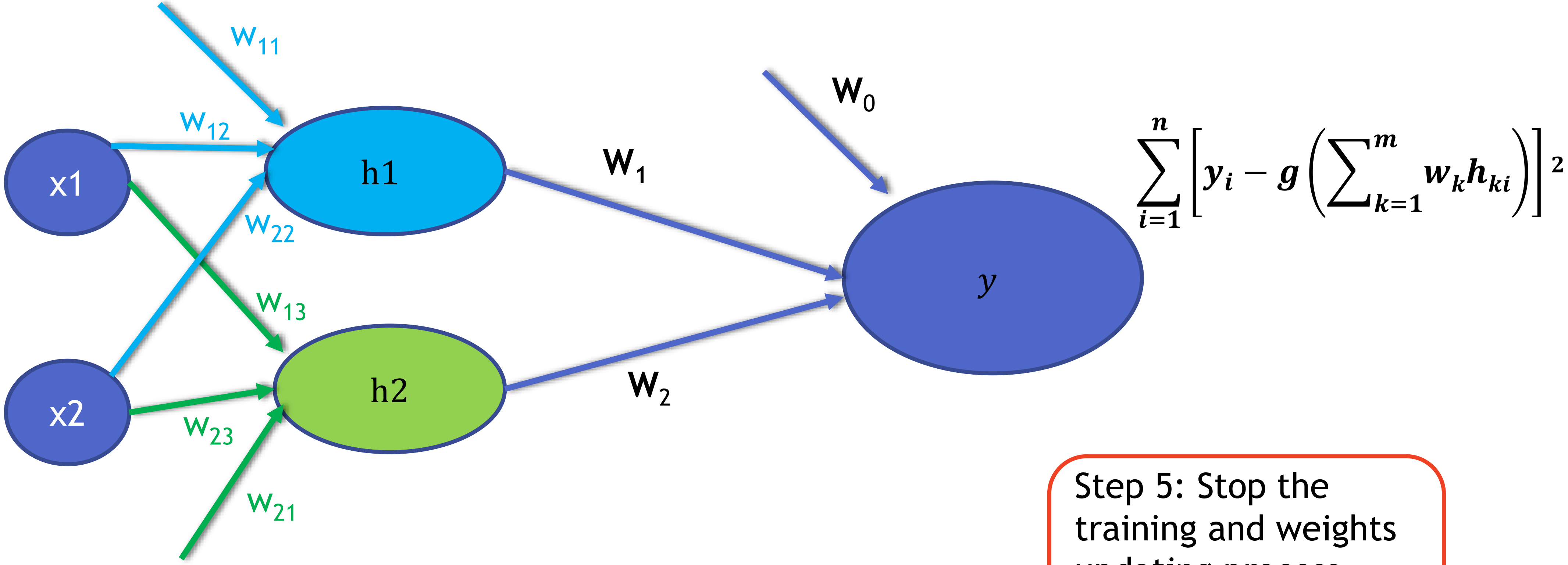
Step 4: Update the weights to reduce the error, recalculate and repeat the process

Update Weights



Step 4: Update the weights to reduce the error, recalculate and repeat the process

Stopping Criteria



Step 5: Stop the training and weights updating process when the minimum error criteria is met

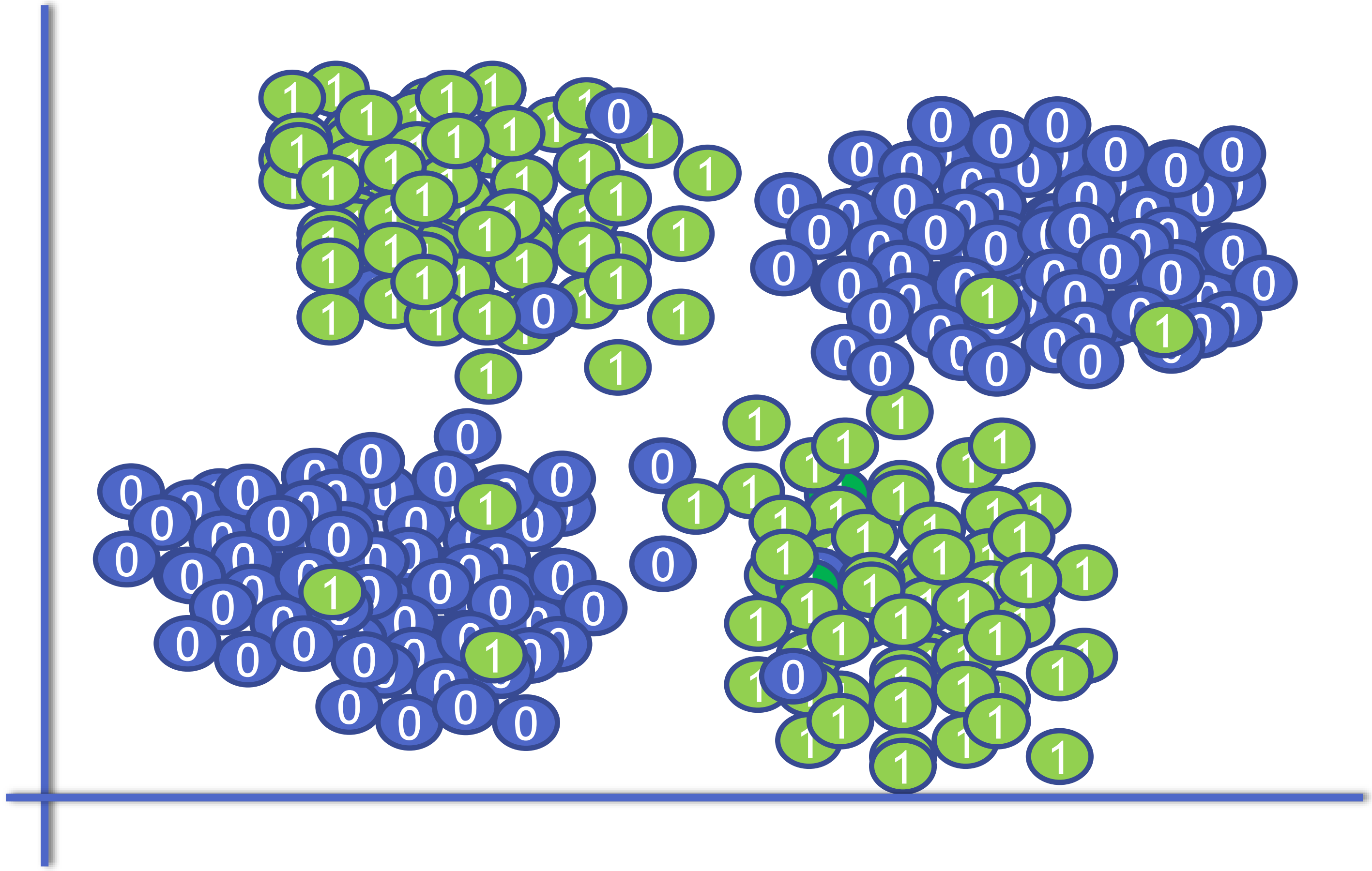
Once Again ….Neural network Algorithm

- Step 1: Initialization of weights: Randomly select some weights
- Step 2 : Training & Activation: Input the training values and perform the calculations forward.
- Step 3 : Error Calculation: Calculate the error at the outputs. Use the output error to calculate error fractions at each hidden layer
- Step 4: Weight training : Update the weights to reduce the error, recalculate and repeat the process of training & updating the weights for all the examples.
- Step 5: Stopping criteria: Stop the training and weights updating process when the minimum error criteria is met

statinfer

Neural network Algorithm-Demo

Neural network Algorithm-Demo

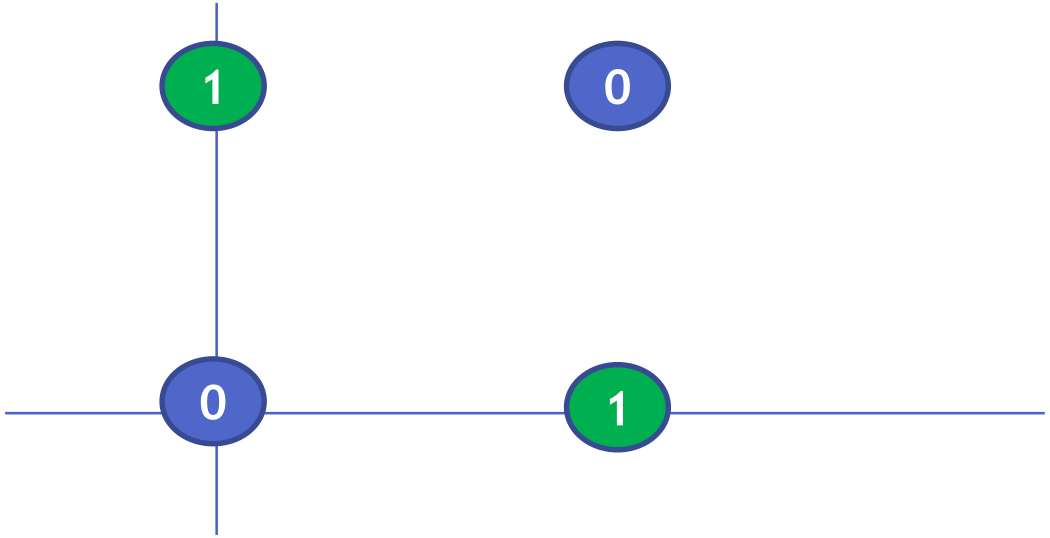


Looks like a dataset that can't be separated by using single linear decision boundary/perceptron

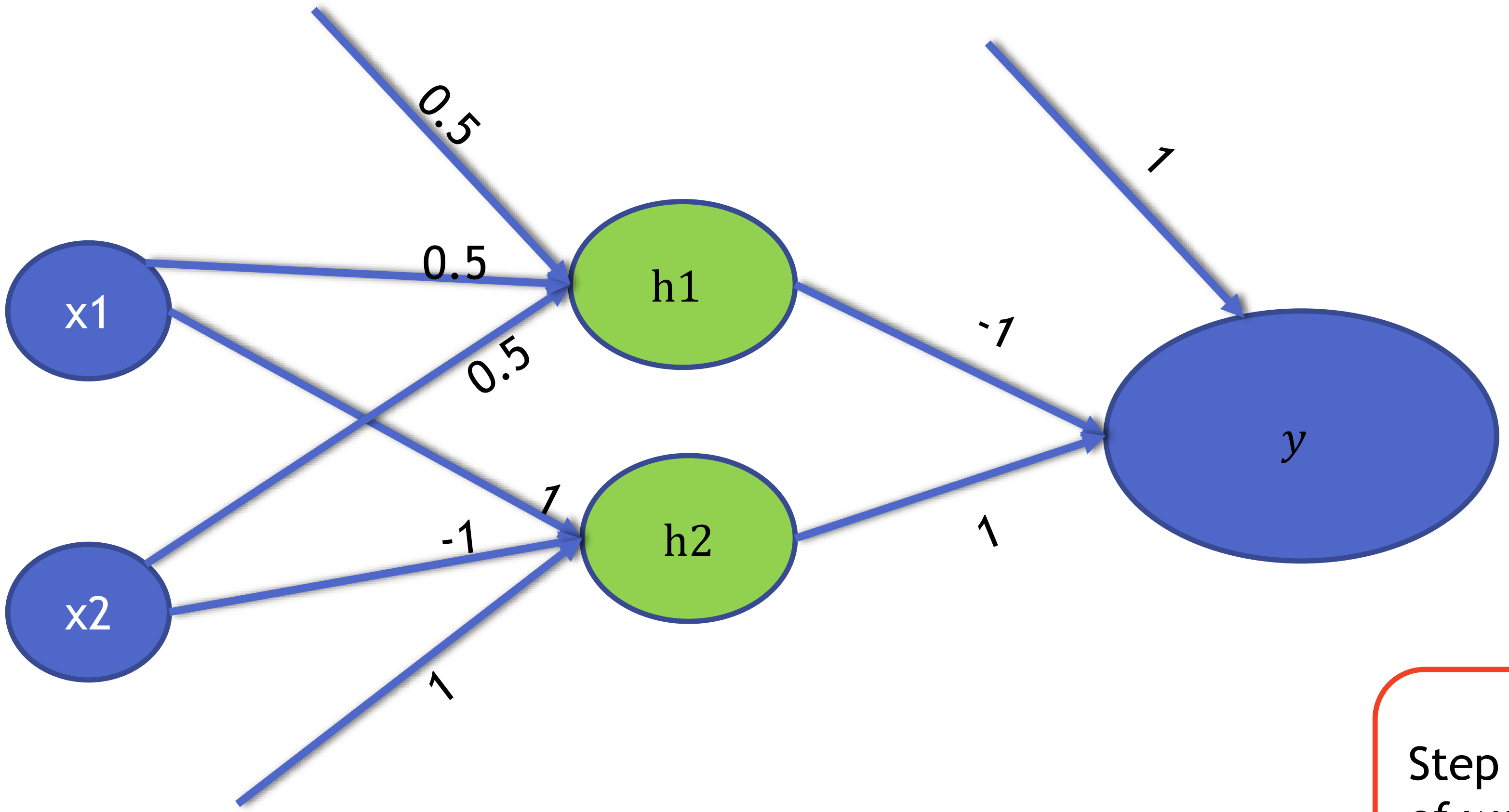
Neural network Algorithm-Demo

- Lets consider a similar but simple classification example
- XOR Gate Dataset

Input1(x1)	Input2(x2)	Output(y)
1	1	0
1	0	1
0	1	1
0	0	0



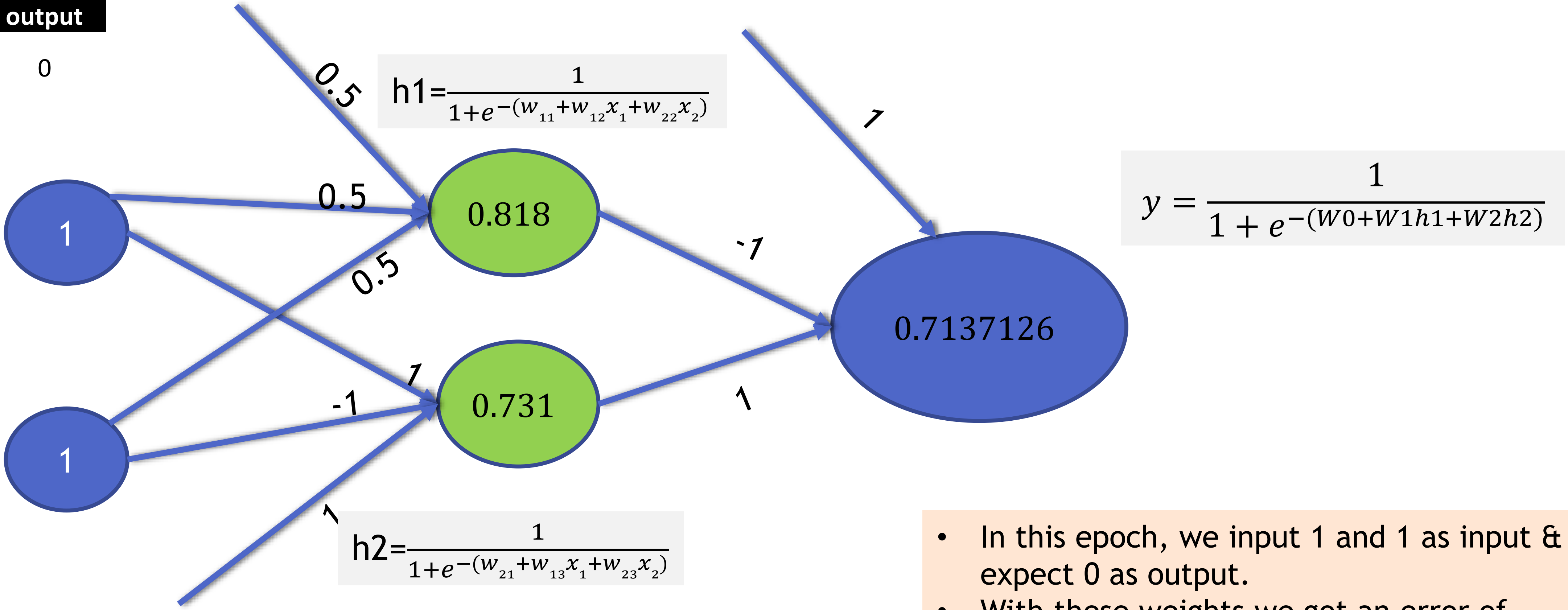
Randomly initialize weights



Step 1: Initialization of weights: Randomly select some weights

Activation

input1	input2	output
1	1	0



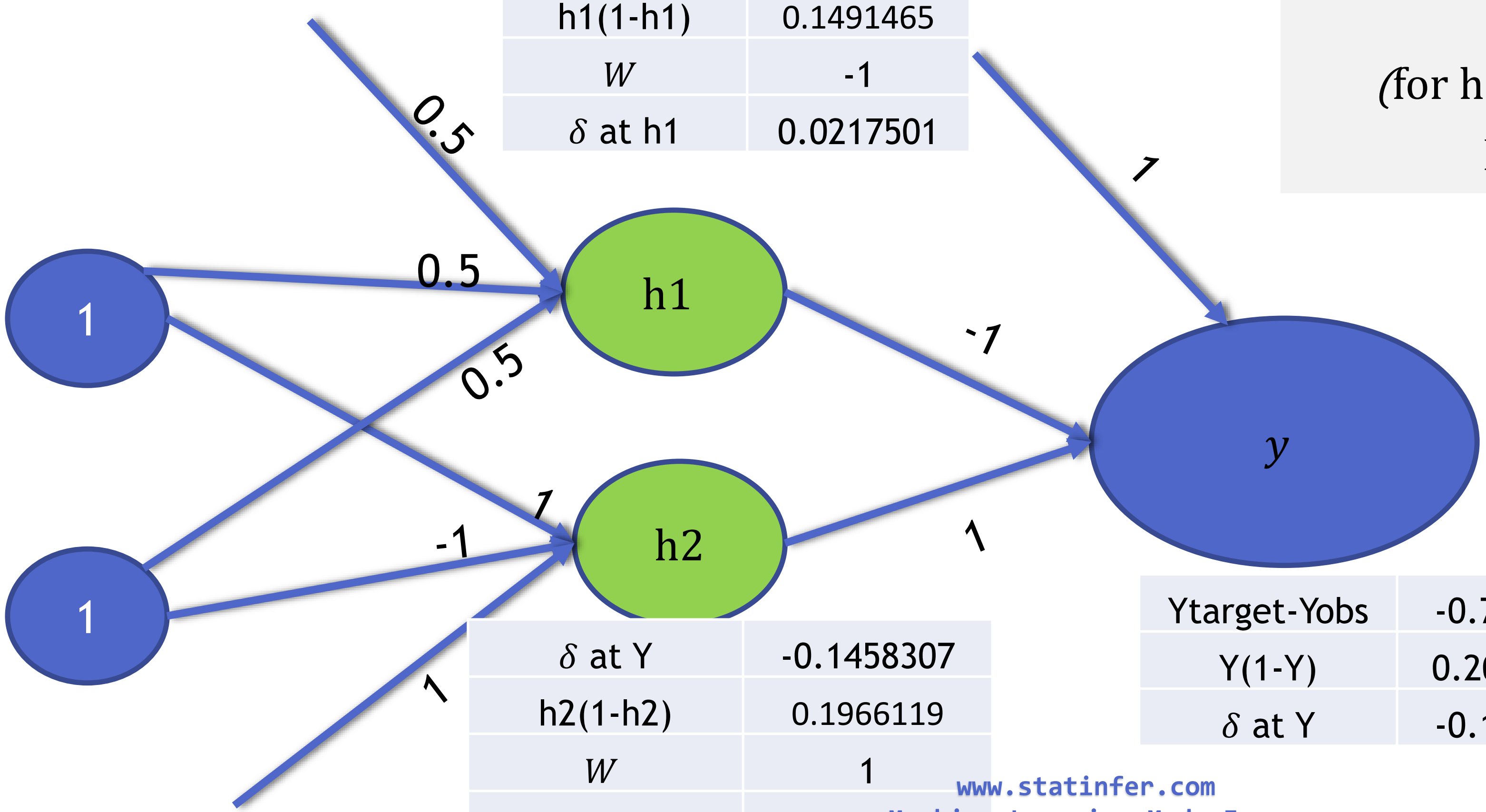
- In this epoch, we input 1 and 1 as input & expect 0 as output.
- With these weights we got an error of -0.714 at output layer
- We need to adjust weights

statinfer

Back-Propagate Errors

$$W_{jk} := W_{jk} + \Delta W_{jk}$$
$$where\ \Delta W_{jk} = \eta . y_j \delta_k$$
η is the learning parameter
$$\delta_k = y_k(1 - y_k) * Err$$
$$(\text{for hidden layers}\delta_k = y_k(1 - y_k) * w_j * Err)$$
Err=Expected output-Actual output

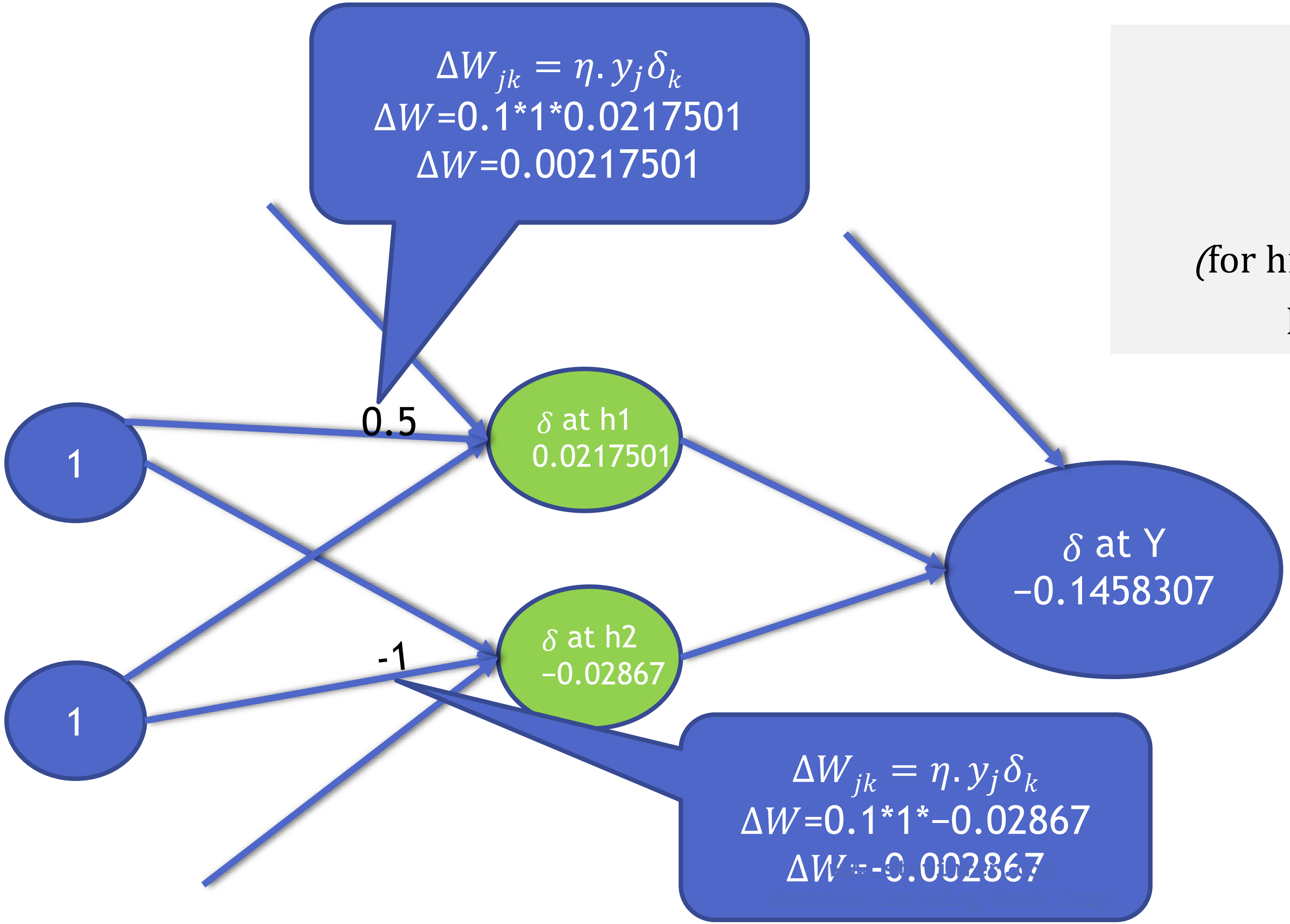
δ at Y	-0.1458307
h1(1-h1)	0.1491465
W	-1
δ at h1	0.0217501



δ at Y	-0.1458307
h2(1-h2)	0.1966119
W	1
δ at h2	-0.0286721

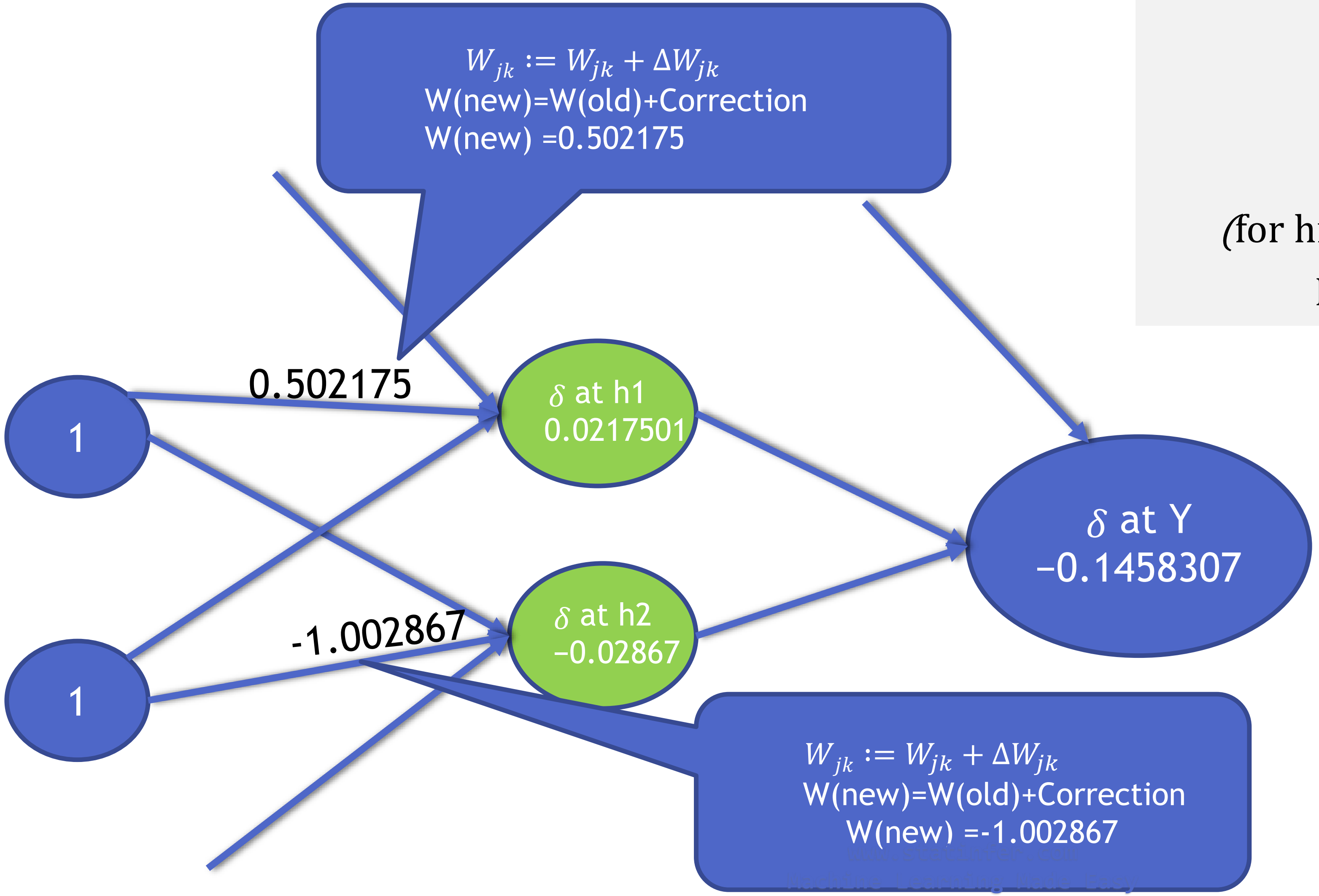
Ytarget-Yobs	-0.7137126
Y(1-Y)	0.20432693
δ at Y	-0.1458307

Calculate Weight Corrections



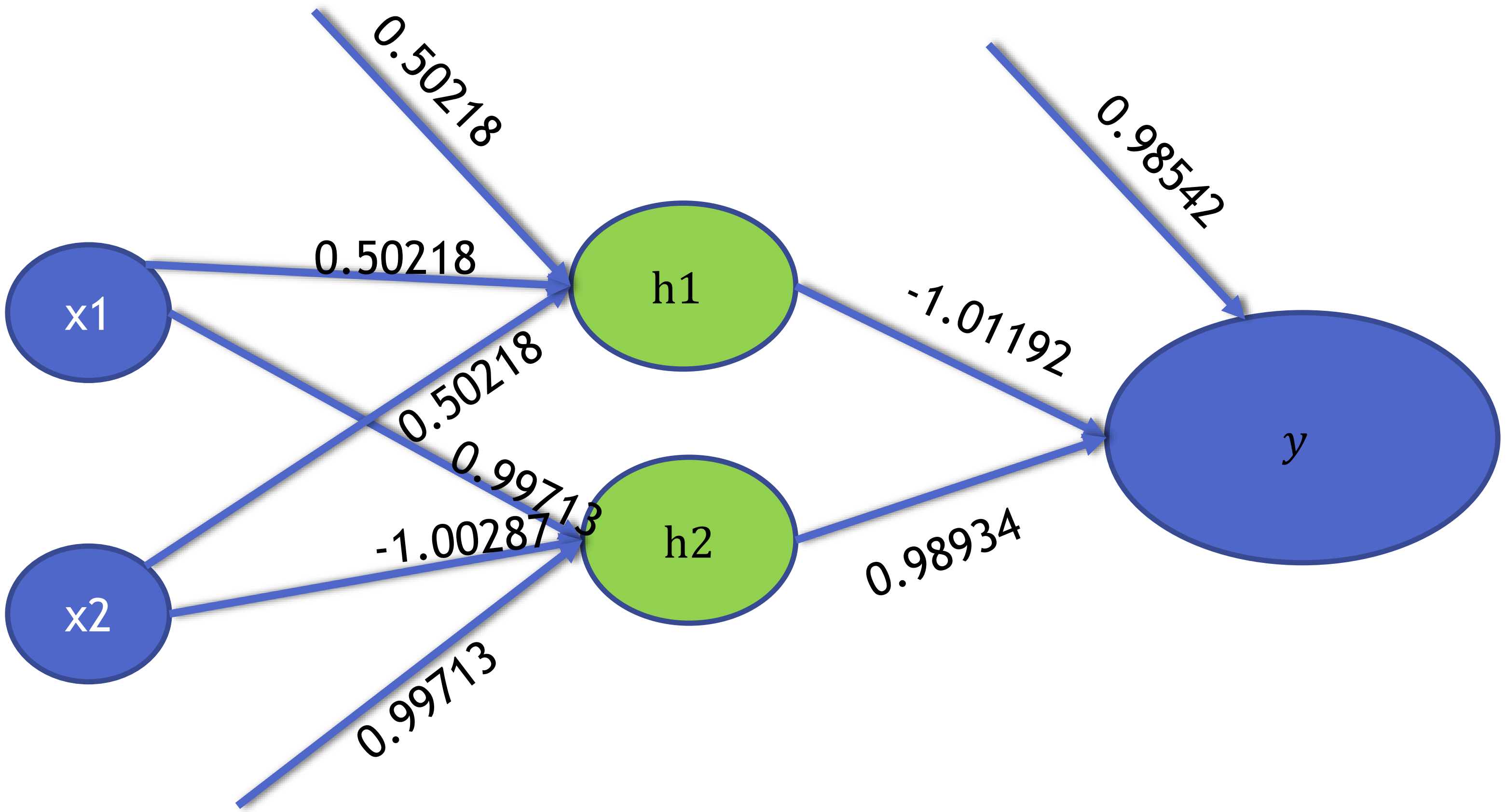
$$W_{jk} := W_{jk} + \Delta W_{jk}$$
$$where\ \Delta W_{jk} = \eta . y_j \delta_k$$
η is the learning parameter
$$\delta_k = y_k(1 - y_k) * Err$$
$$(\text{for hidden layers} \delta_k = y_k(1 - y_k) * w_j * Err)$$
Err=Expected output-Actual output

Updated Weights



$$W_{jk} := W_{jk} + \Delta W_{jk}$$
$$where\ \Delta W_{jk} = \eta . y_j \delta_k$$
η is the learning parameter
$$\delta_k = y_k(1 - y_k) * Err$$
$$(\text{for hidden layers}\delta_k = y_k(1 - y_k) * w_j * Err)$$
Err=Expected output-Actual output

Updated Weights..contd



Iterations and Stopping Criteria

- This iteration is just for one training example (1,1,0). This is just the first epoch.
- We repeat the same process of training and updating of weights for all the data points
- We continue and update the weights until we see there is no significant change in the error or when the maximum permissible error criteria is met.
- By updating the weights in this method, we reduce the error slightly. When the error reaches the minimum point the iterations will be stopped and the weights will be considered as optimum for this training set

statinfer

LAB: Building the neural network

LAB: Building the neural network

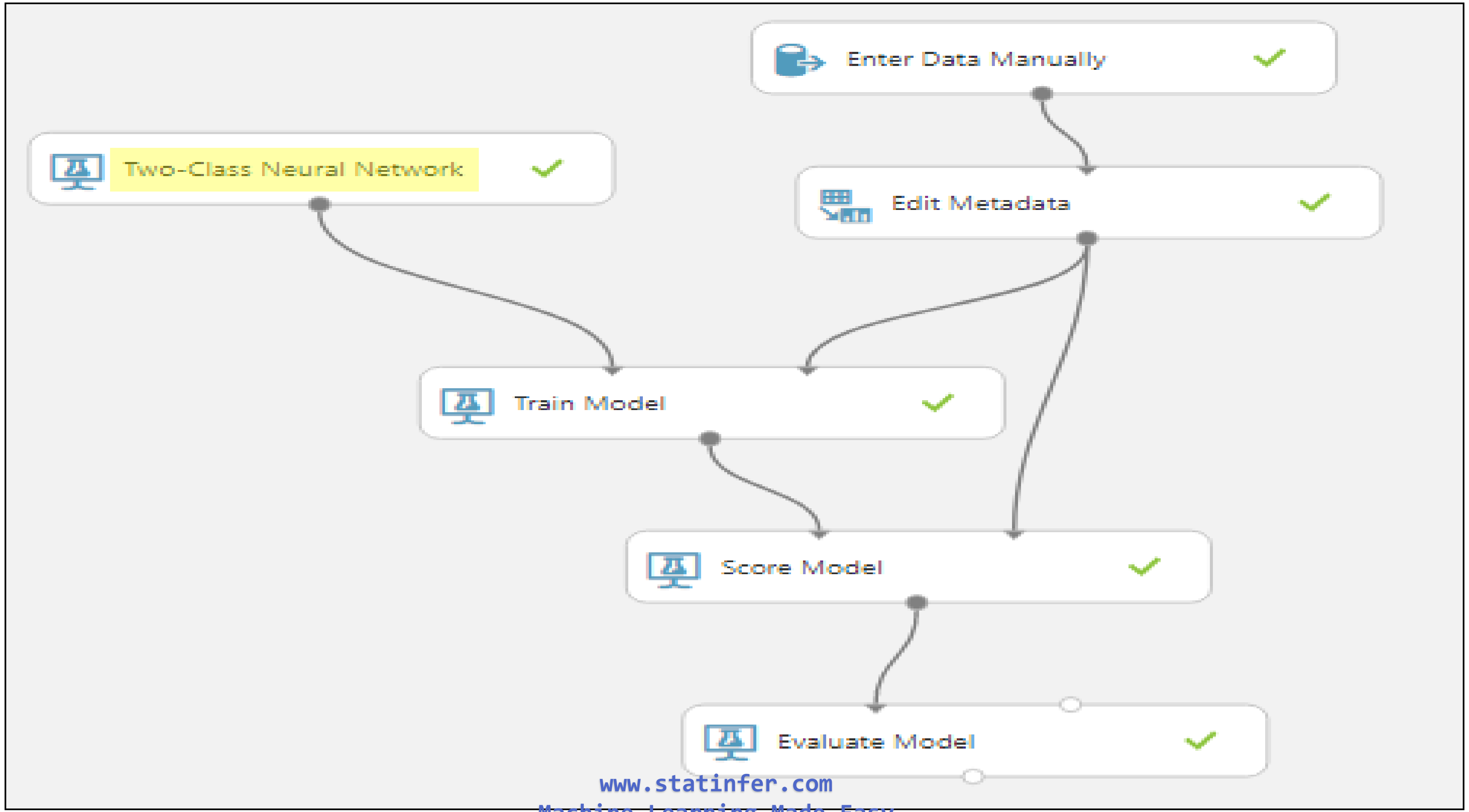
- Build a neural network for XOR data

Steps - Neural Network on XOR Data

- Drag and drop the **Enter data Manually** into the canvas and fill up with the data as in the figure
- Drag and drop **Edit Metadata** and select the properties as in the figure
- Drag and drop **Two-Class Neural Network**, **Train Model, Score Model** and **Evaluate Model**
- Connections:
  - Connect **Enter data Manually** to **Edit Metadata**
  - Connect the **Two-Class Neural Network** to the first input of the **Train Model** and **Edit Metadata** to the second input of the **Train Model**
  - Connect the **Train Model** to the first input of the **Score Model** and **Test dataset** to the second input of the **Score Model**
  - Connect the output of the **Score Model** to the **Evaluate Model**
- Fill the properties of the **Multiclass Neural Network** and **Train Model** as in the figure
- Click run and visualize the output of **Evaluate Model,** check the accuracy

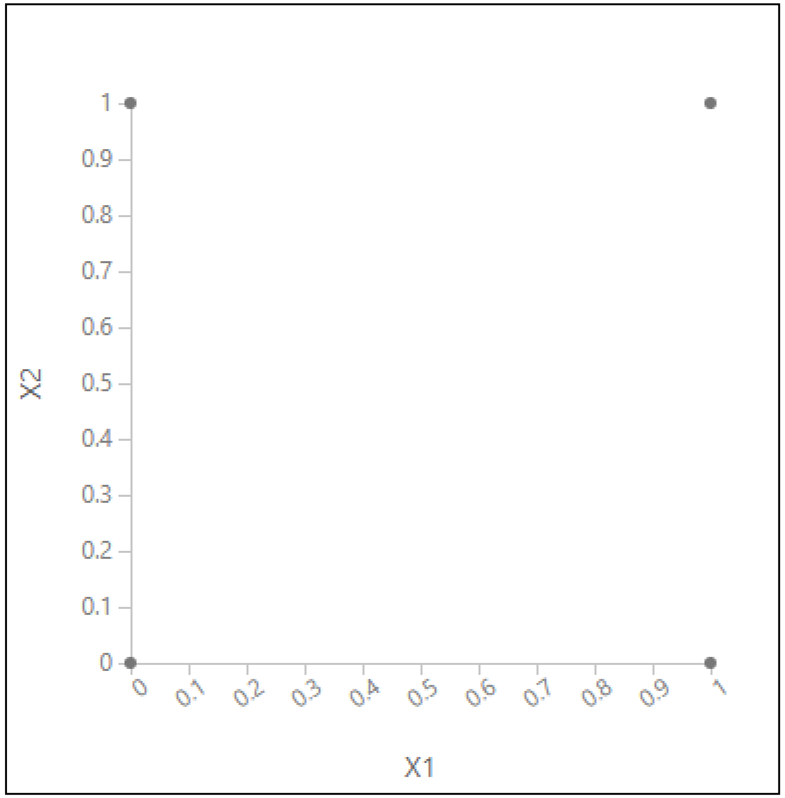
Steps - Neural Network on XOR Data

Fig32: Neural Network on XOR Data



Steps - Neural Network on XOR Data

Fig33: Scatter Plot – X1 vs X2



Steps - Neural Network on XOR Data

Fig34: Properties - Enter Data Manually

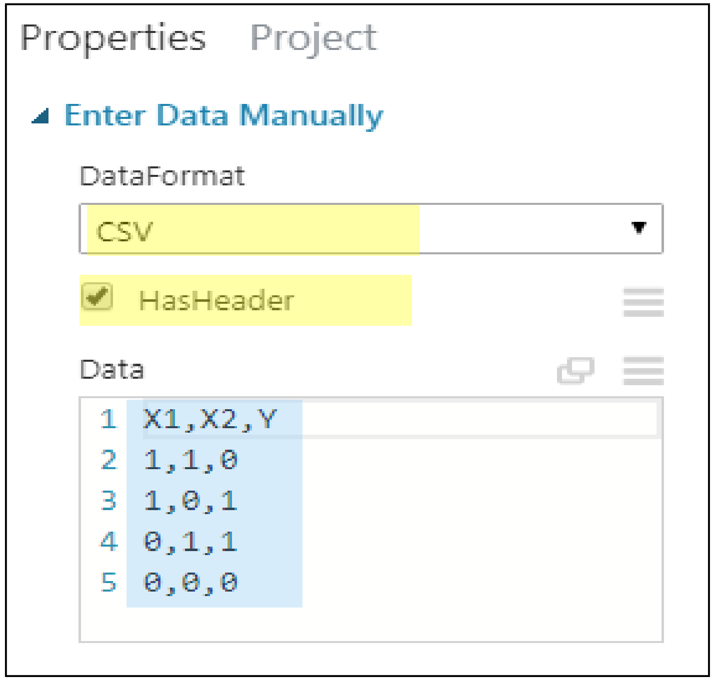
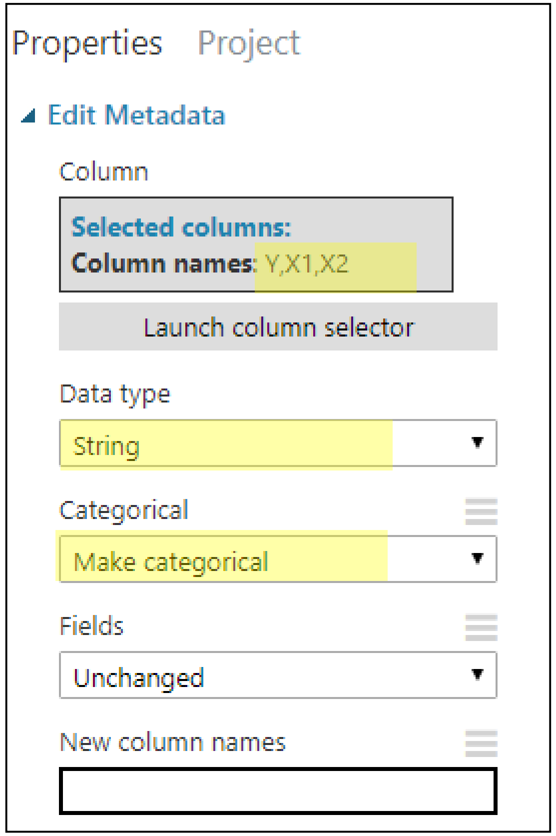


Fig:35 Properties - Edit Metadata



Steps - Neural Network on XOR Data

Fig36: Properties - Two Class Neural Network

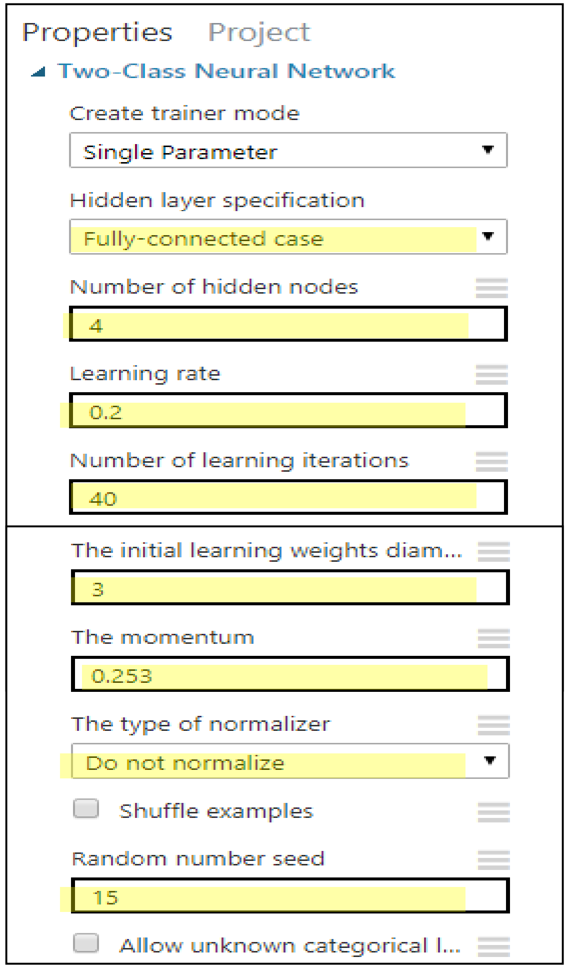
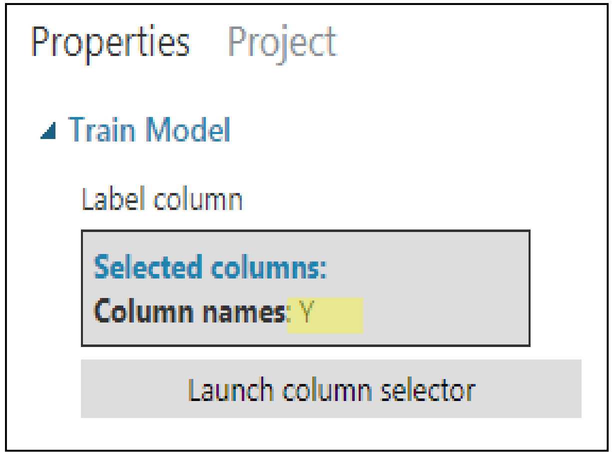
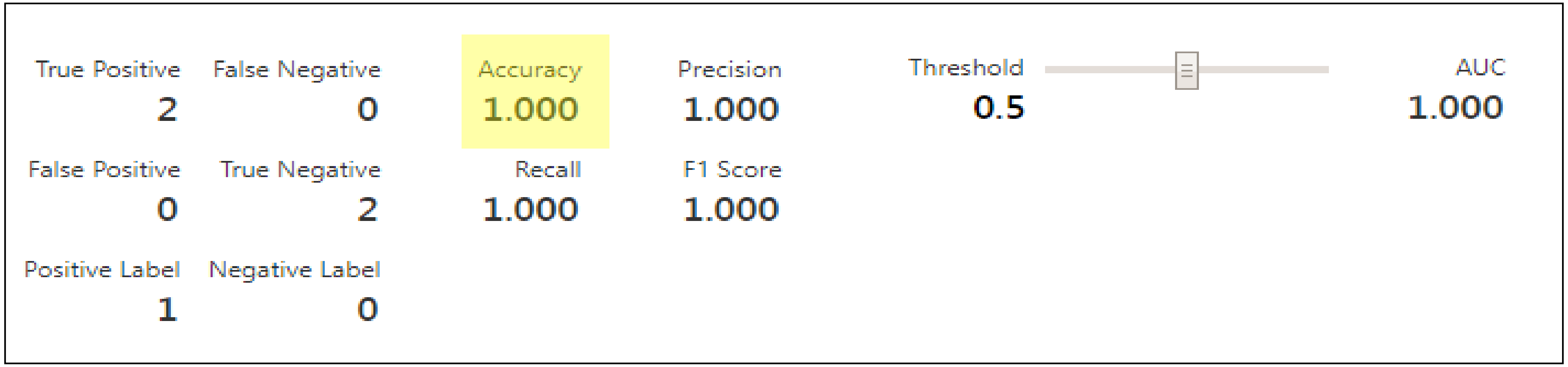


Fig37: Properties - Train Model



Steps - Neural Network on XOR Data

Fig38: Accuracy(XOR Data)



Lab: Building Neural network on Employee productivity data

- Dataset: Emp_Productivity/Emp_Productivity.csv
- Draw a 2D graph between age, experience and productivity
- Build neural network algorithm to predict the productivity based on age and experience
- Plot the neural network with final weights
- Increase the hidden layers and see the change in accuracy

Steps - Neural network on Employee productivity data

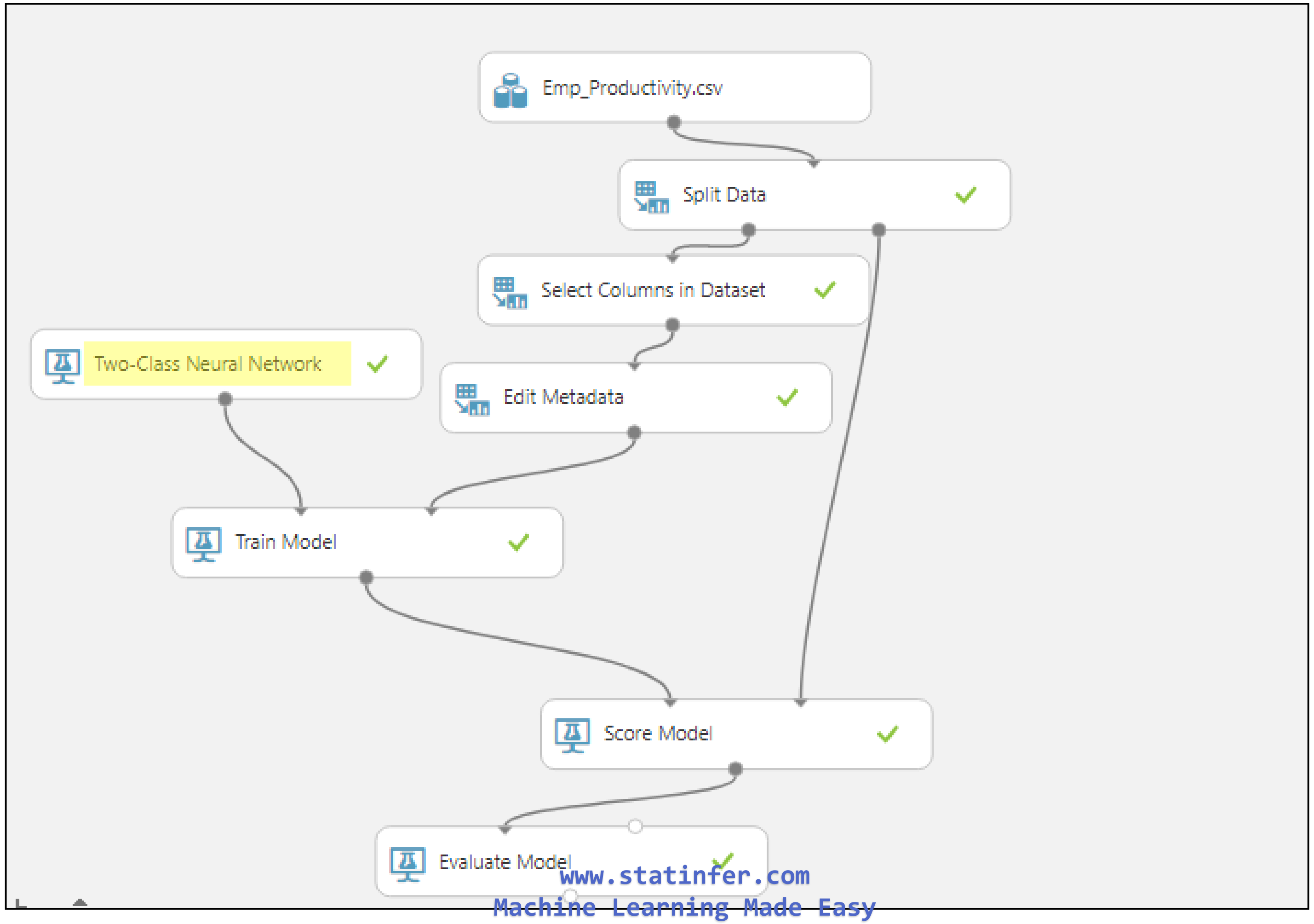
- Drag and drop the **dataset** into the canvas
- Drag and drop the **Split Data** into the canvas and select the properties as in the figure and select the properties as in the figure
- Drag and drop **Select columns from Dataset** and select the properties as in the figure
- Drag and drop **Edit Metadata** and select the properties as in the figure
- Drag and drop **Two-Class Neural Network**, **Train Model**, **Score Model** and **Evaluate Model**

Steps - Neural network on Employee productivity data

- Connections:
  - Connect **dataset** to the **Split Data**
  - Connect **Split Data** to the **Select columns from Dataset**
  - Connect **Select columns from Dataset** to the **Edit Metadata**
  - Connect the **Two-Class Neural Network** to the first input of the **Train Model** and **Edit Metadata** to the second input of the **Train Model**
  - Connect the **Train Model** to the first input of the **Score Model** and **Test dataset** to the second input of the **Score Model**
  - Connect the output of the **Score Model** to the **Evaluate Model**
- Fill the properties of the **Multiclass Neural Network** and **Train Model** as in the figure
- Click run and visualize the output of **Evaluate Model**, check the accuracy

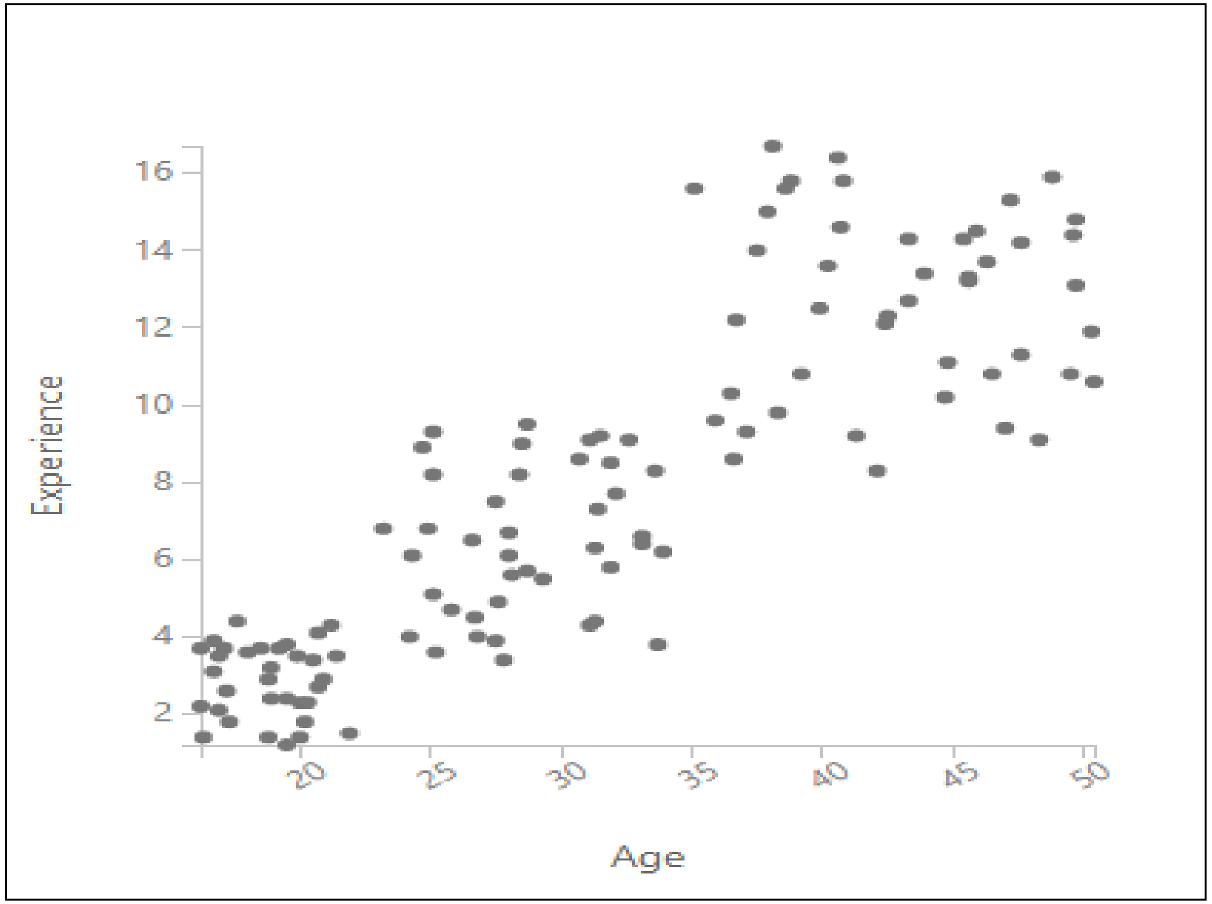
Steps - Neural network on Employee productivity data

Fig39: Neural Network on Employee Productivity Data



Steps - Neural network on Employee productivity data

Fig40: Scatter Plot – Age vs Experience



Steps - Neural network on Employee productivity data

Fig41: Properties - Split Data

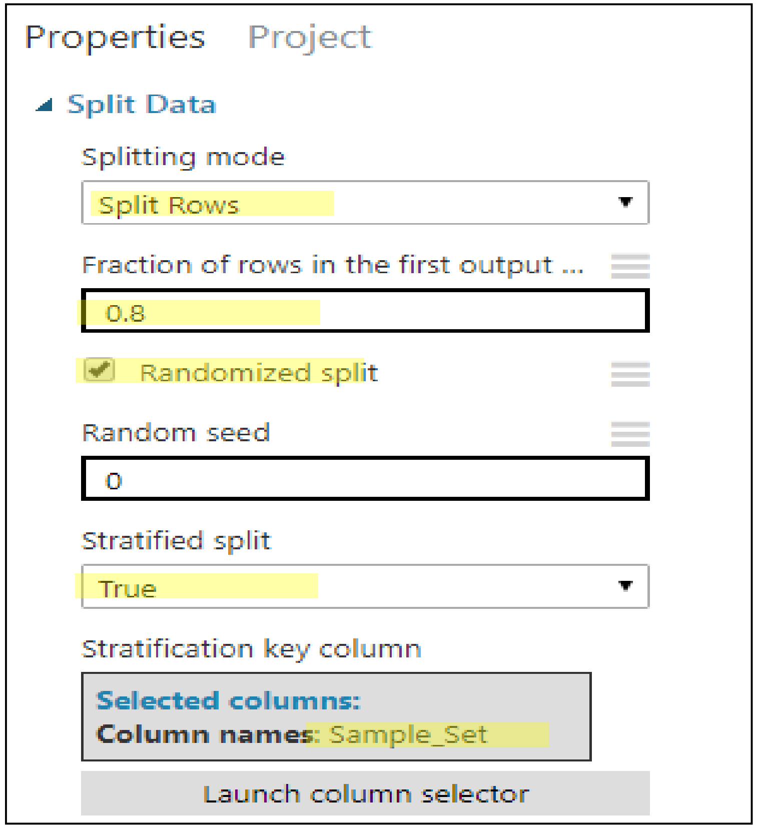
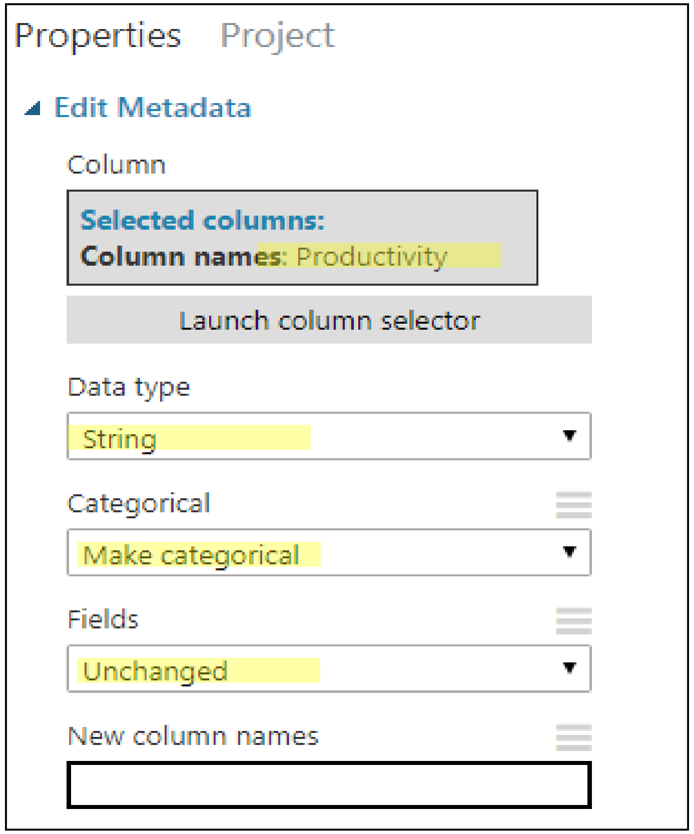


Fig42: Properties - Edit Metadata



Steps - Neural network on Employee productivity data

Fig43: Properties - Two Class Neural Network

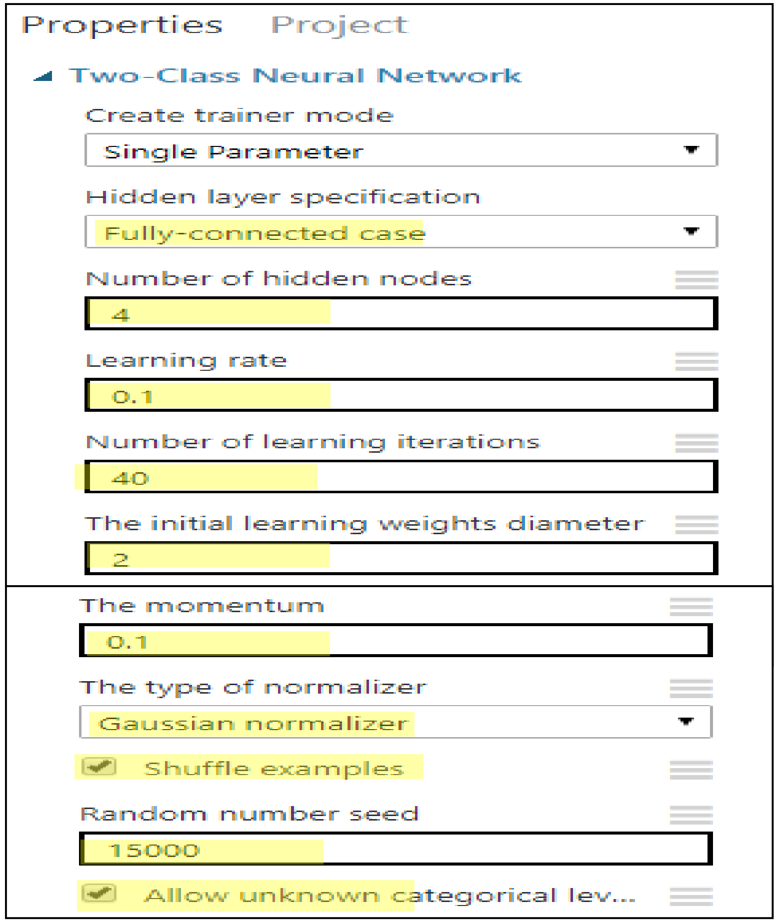
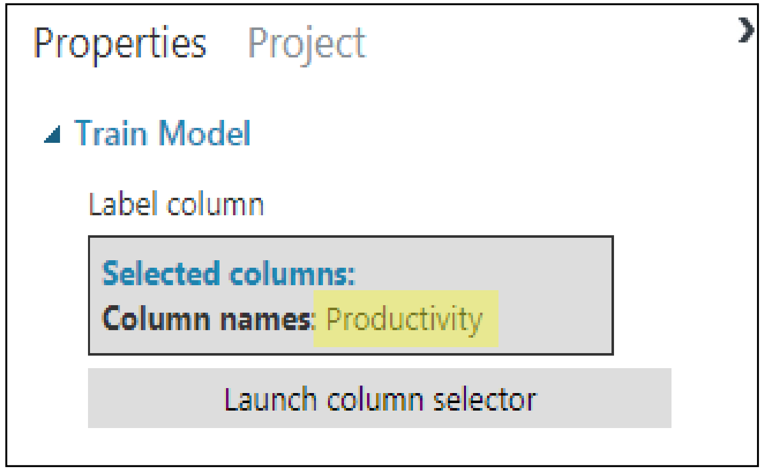


Fig44: Properties - Train Modal



Steps - Neural network on Employee productivity data

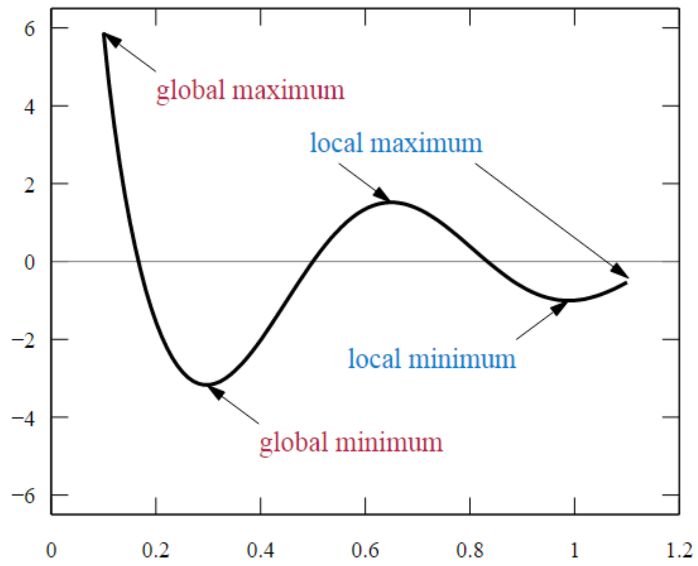
Fig45: Accuracy and Confusion Matrix

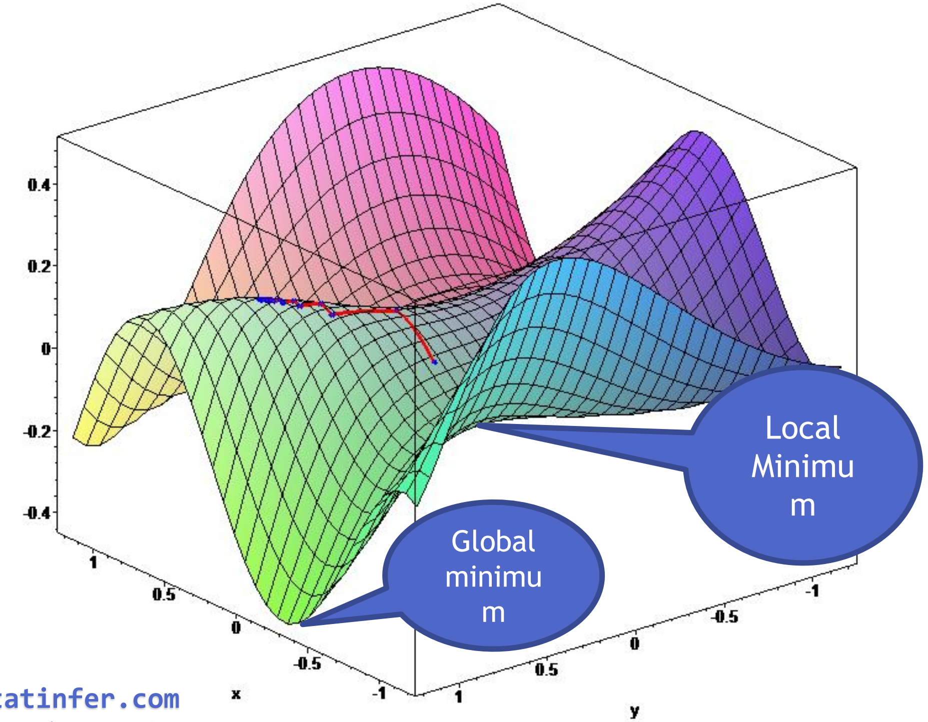
True Positive	False Negative	Accuracy	Precision	Threshold	AUC
8	0	1.000	1.000	0.5	1.000
False Positive	True Negative	Recall	F1 Score		
0	16	1.000	1.000		
Positive Label	Negative Label				
1	0				

Local vs. Global Minimum

Local vs. Global Minimum

- The neural network might give different results with different start weights.
- The algorithm tries to find the local minima rather than global minima.
- There can be many local minima’s, which means there can be many solutions to neural network problem
- We need to perform the validation checks before choosing the final model.

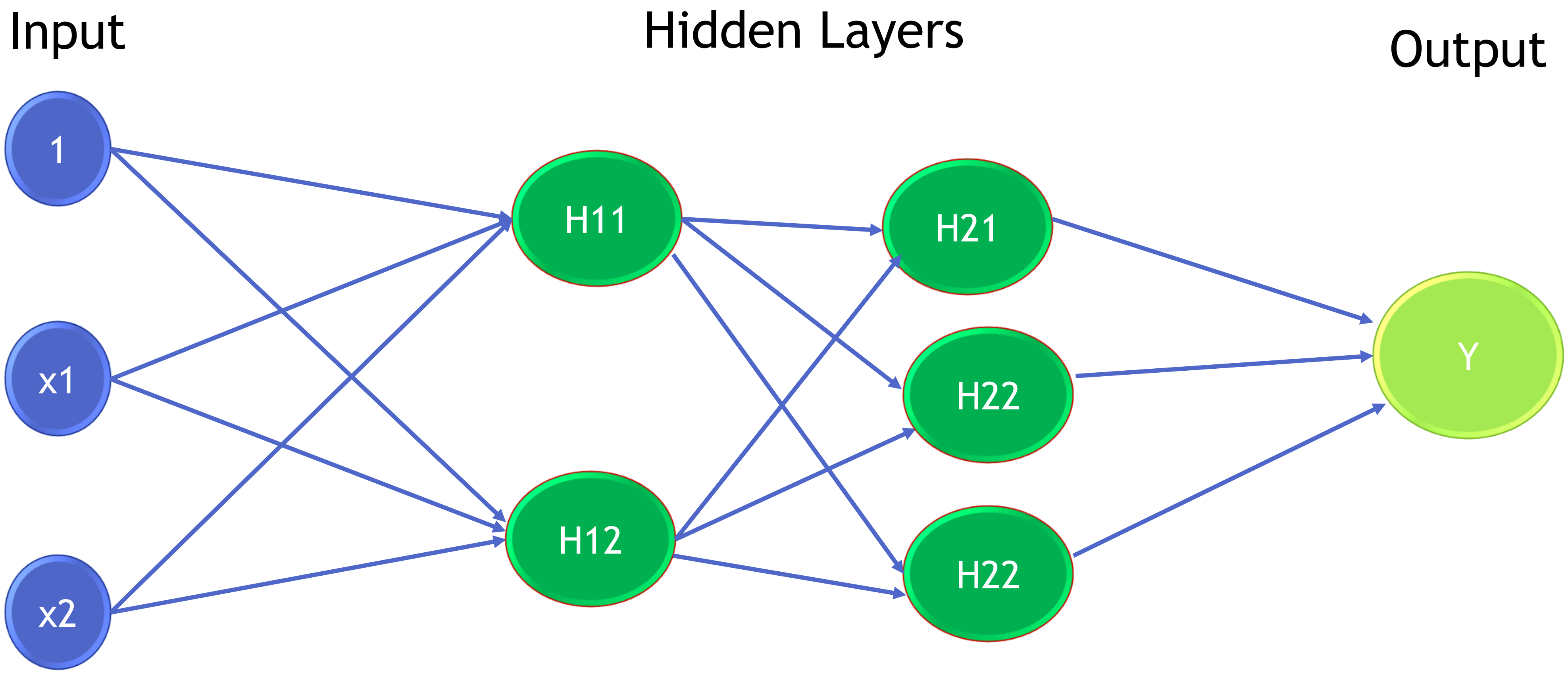




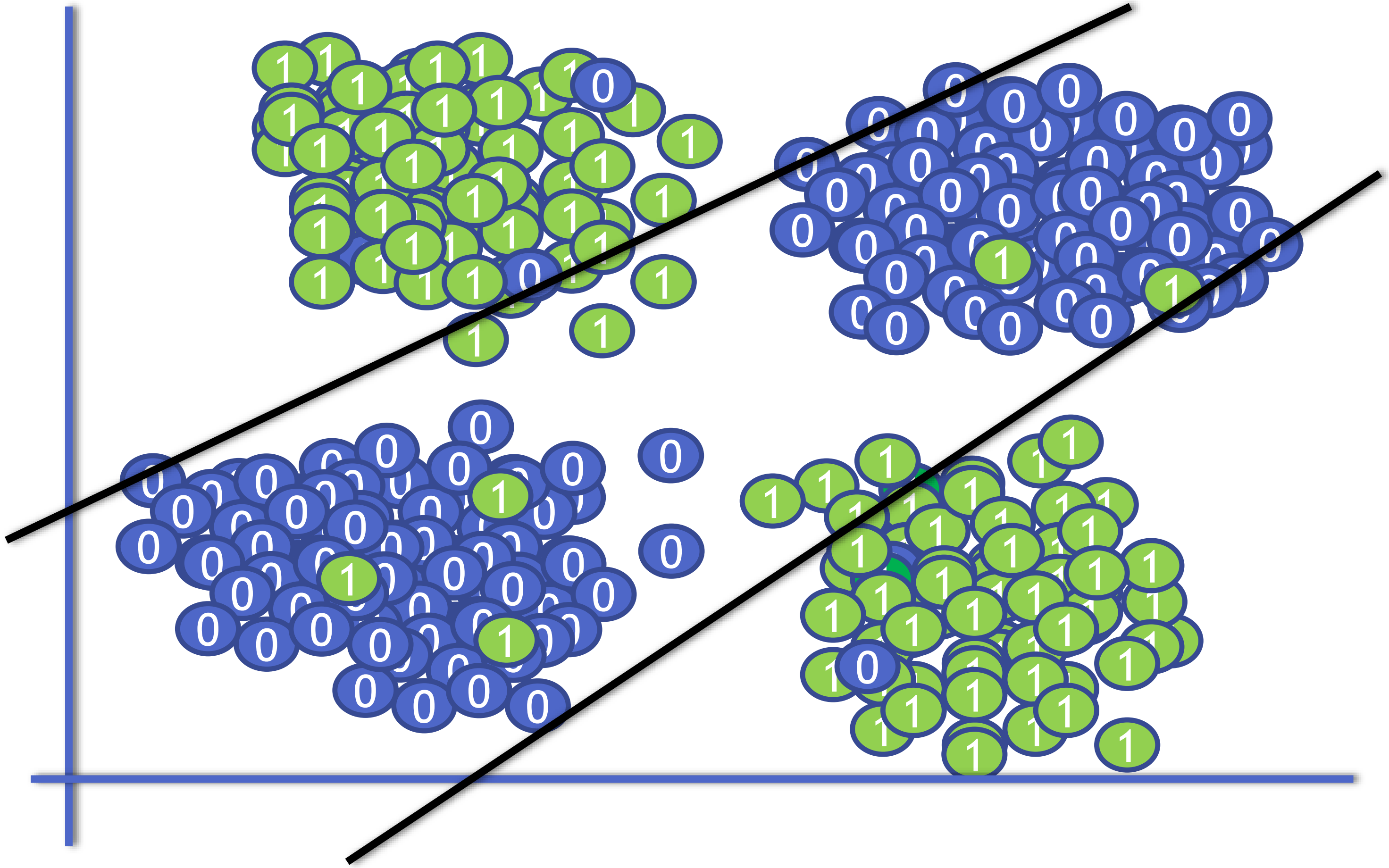
statinfer

Hidden layers and their role

Multi Layer Neural Network

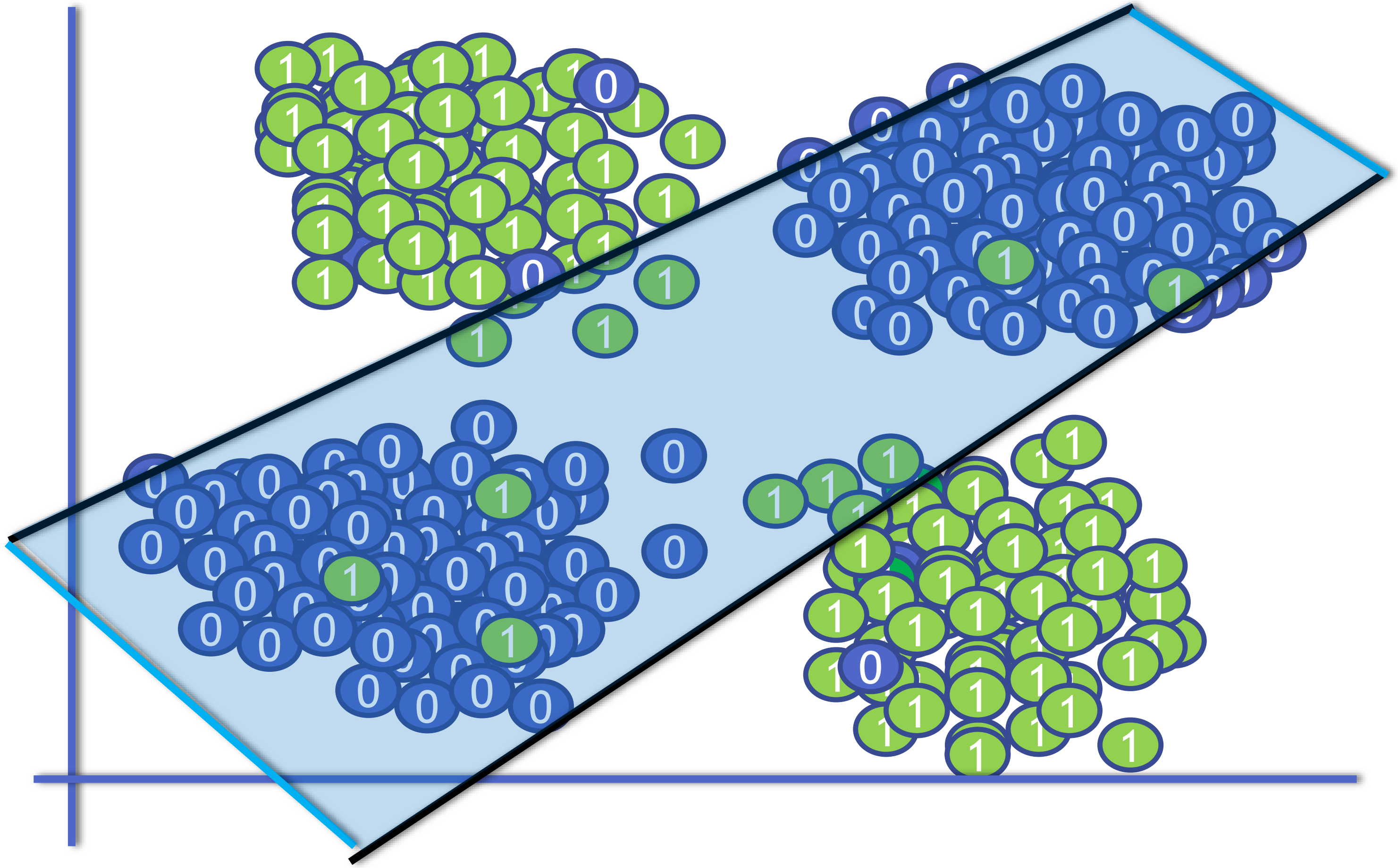


The role of hidden layers



- The First hidden layer
- The first layer is nothing but the liner decision boundaries
- The simple logistic regression line outputs
- We can see them as multiple lines on the decision space

The role of hidden layers



- The Second hidden layer
- The Second layer combines these lines and forms simple decision boundary shapes
- The third hidden layer forms even complex shapes within the boundaries generated by second layer.
- You can imagine All these layers together divide the whole objective space into multiple decision boundary shapes, the cases within the shape are class-1 outside the shape are class-2

The Number of hidden layers

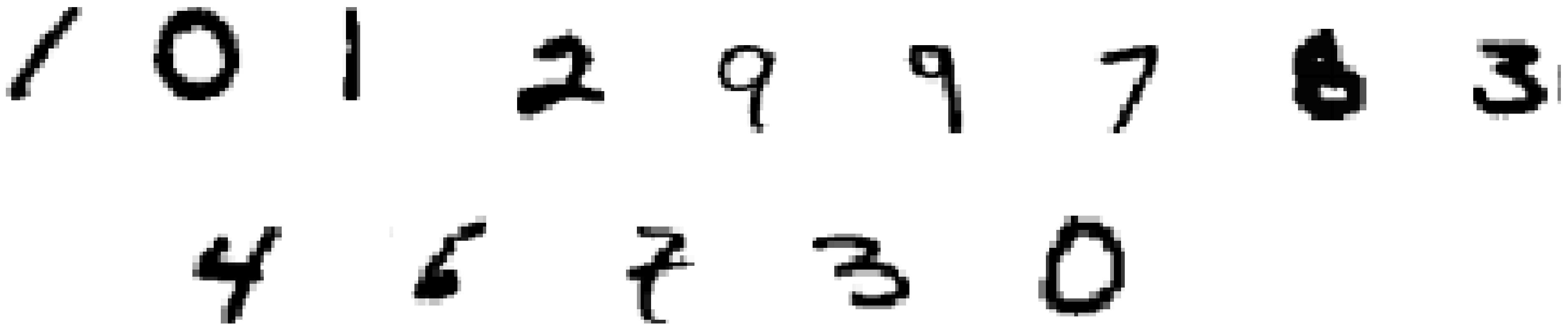
The Number of hidden layers

- There is no concrete rule to choose the right number. We need to choose by trail and error validation
- Too few hidden layers might result in imperfect models. The error rate will be high
- High number of hidden layers might lead to over-fitting, but it can be identified by using some validation techniques
- The final number is based on the number of predictor variables, training data size and the complexity in the target.
- When we are in doubt, its better to go with many hidden nodes than few. It will ensure higher accuracy. The training process will be slower though
- Cross validation and testing error can help us in determining the model with optimal hidden layers

LAB: Digit Recognizer

LAB: Digit Recognizer

- Take an image of a handwritten single digit, and determine what that digit is.
- Normalized handwritten digits, automatically scanned from envelopes by the U.S. Postal Service. The original scanned digits are binary and of different sizes and orientations; the images here have been de slanted and size normalized, resultingin 16 x 16 grayscale images (Le Cun et al., 1990).
- The data are in two gzipped files, and each line consists of the digitid (0-9) followed by the 256 grayscale values.
- Build a neural network model that can be used as the digit recognizer
- Use the test dataset to validate the true classification power of the model
- What is the final accuracy of the model?

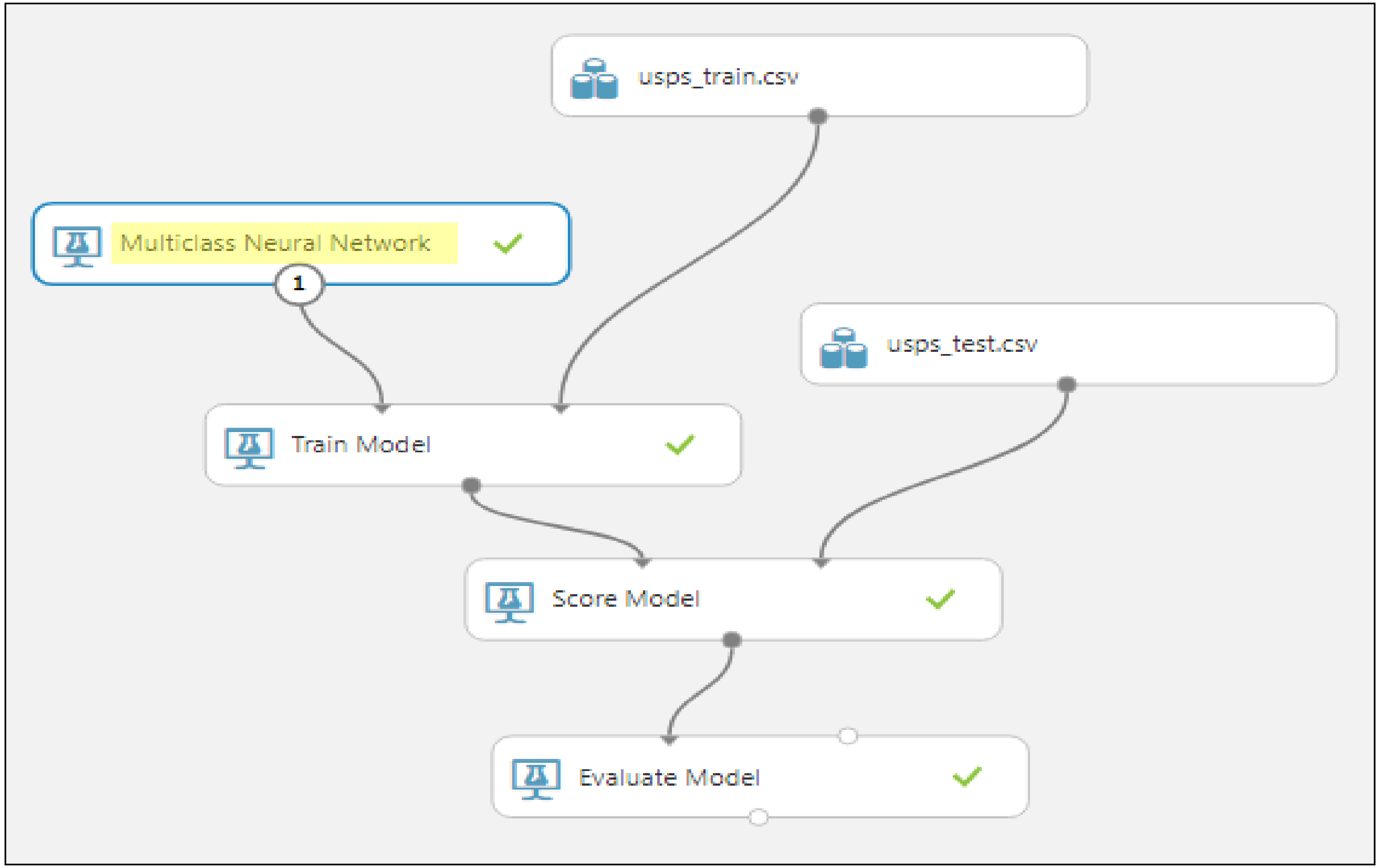


Steps - Digit Recognizer

- Drag and drop the **Training dataset** into the canvas
- Drag and drop the **Test dataset** into the canvas
- Drag and drop **Multiclass Neural Network**, **Train Model, Score Model** and **Evaluate Model**
- Connections:
  - Connect the **Multiclass Neural Network** to the first input of the **Train Model** and **Training dataset** to the second input of the **Train Model**
  - Connect the **Train Model** to the first input of the **Score Model** and **Test dataset** to the second input of the **Score Model**
  - Connect the output of the **Score Model** to the **Evaluate Model**
- Fill the properties of the **Multiclass Neural Network** and **Train Model** as in the figure
- Click run and visualize the output of **Evaluate Model,** check the accuracy

Steps - Digit Recognizer

Fig46: Digit Recognition using Multiclass Neural Network



Steps - Digit Recognizer

Fig47: Properties - Multiclass Neural Network

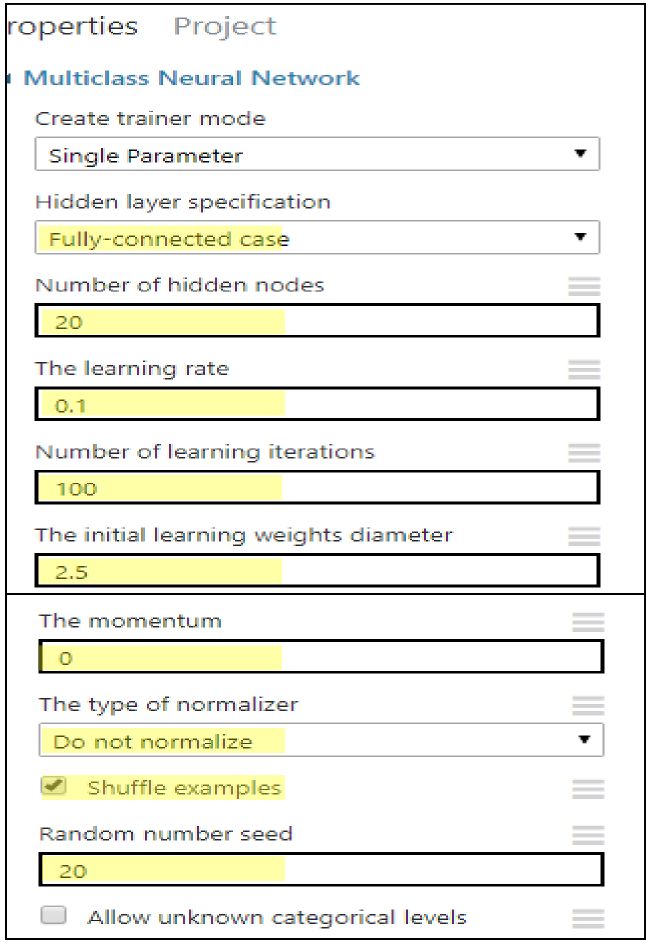
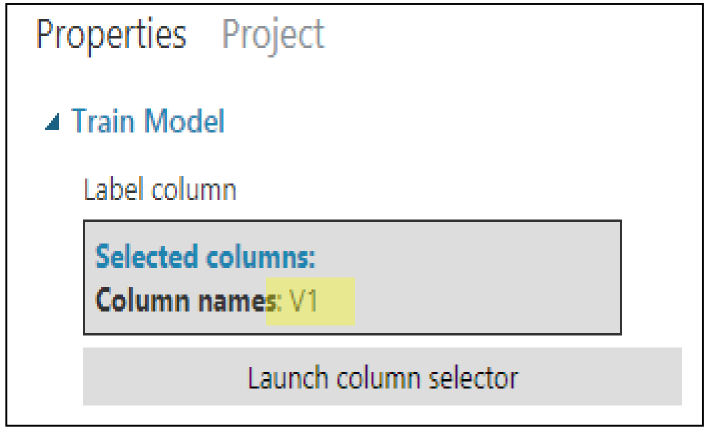


Fig48: Properties - Train Model



Steps - Digit Recognizer

Fig49: Accuracy - Digit Recognition

Metrics

Overall accuracy	0.918286
Average accuracy	0.983657
Micro-averaged precision	0.918286
Macro-averaged precision	0.915908
Micro-averaged recall	0.918286
Macro-averaged recall	0.910863

Steps - Digit Recognizer

Fig50: Confusion Matrix

Predicted Class

Actual Class	0	1	2	3	4	5	6	7	8	9
0	96.7%	0.6%	0.6%		0.6%	0.3%	0.6%	0.3%	0.6%	
1		95.5%	0.4%		1.9%	0.4%	1.5%		0.4%	
2	2.5%	0.5%	88.9%	1.0%	2.5%	1.0%	0.5%	1.0%	2.0%	
3	1.8%	0.6%	2.4%	82.5%		7.8%		1.2%	2.4%	1.2%
4		0.5%	2.0%		92.0%	2.0%	0.5%	0.5%		2.5%
5	3.8%	0.6%		1.3%	0.6%	92.5%				1.3%
6	1.2%		0.6%		1.8%	1.8%	94.1%		0.6%	
7				1.4%	4.1%	0.7%		91.2%		2.7%
8	3.0%	1.2%	1.2%	1.8%	3.0%	3.6%		0.6%	84.3%	1.2%
9		0.6%	0.6%		3.4%	0.6%		1.7%		93.2%

Real-world applications

Real-world applications

- Self driving car by taking the video as input
- Speech recognition
- Face recognition
- Cancer cell analysis
- Heart attack predictions
- Currency predictions and stock price predictions
- Credit card default and loan predictions
- Marketing and advertising by predicting the response probability
- Weather forecasting and rainfall prediction

Drawbacks of Neural Networks

Drawbacks of Neural Networks

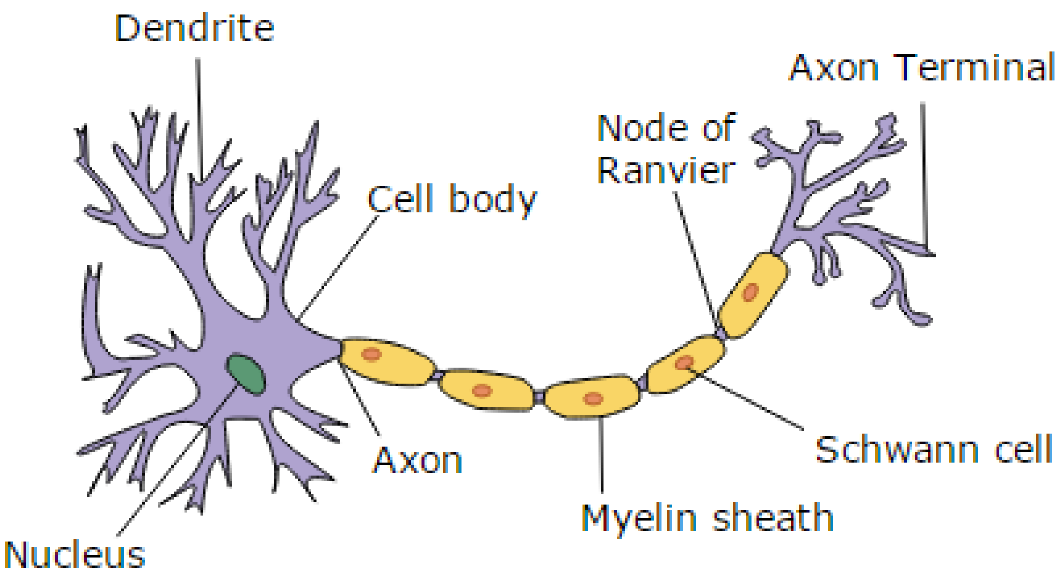
- No real theory that explains how to choose the number of hidden layers
- Takes lot of time when the input data is large, needs powerful computing machines
- Difficult to interpret the results. Very hard to interpret and measure the impact of individual predictors
- Its not easy to choose the right training sample size and learning rate.
- The local minimum issue. The gradient descent algorithm produces the optimal weights for the local minimum, the global minimum of the error function is not guaranteed

Why the name neural network?

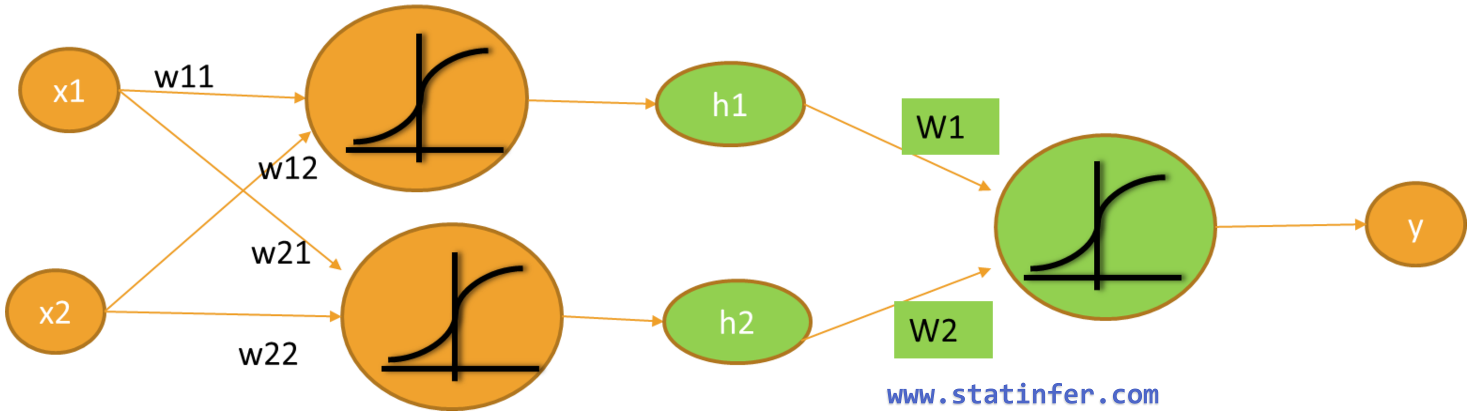
Why the name neural network?

- The neural network algorithm for solving complex learning problems is inspired by human brain
- Our brains are a huge network of processing elements. It contains a network of billions of neurons.
- In our brain, a neuron receives input from other neurons. Inputs are combined and send to next neuron
- The artificial neural network algorithm is built on the same logic.

Why the name neural network?



Dendrites→ Input(X)
Cell body → Processor(Σwx)
Axon → Output(Y)



Conclusion

Conclusion

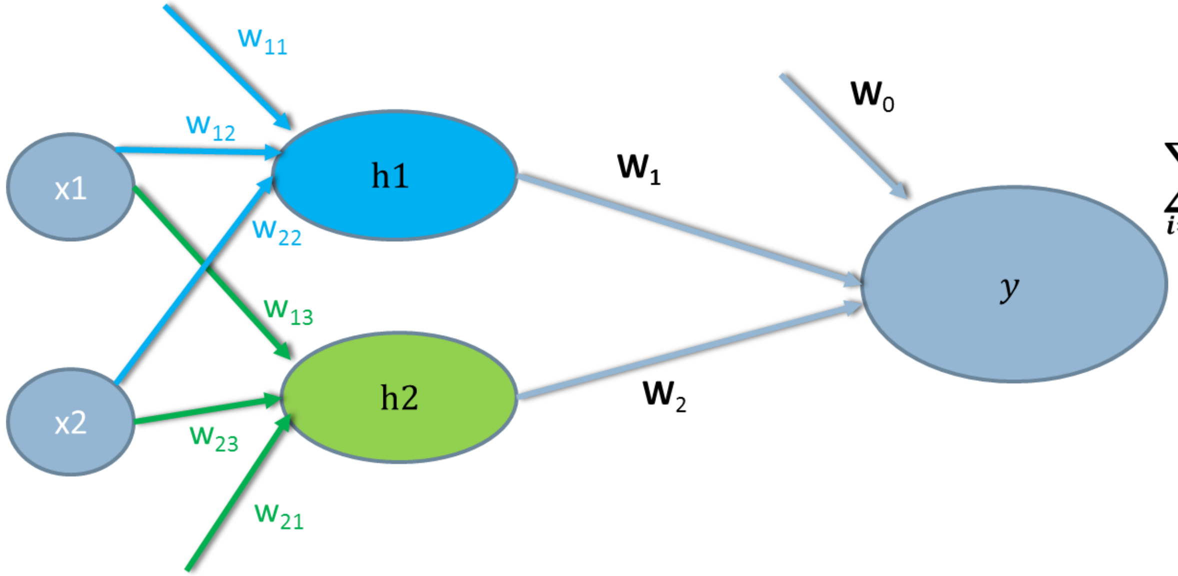
- Neural network is a vast subject. Many data scientists solely focus on only Neural network techniques
- In this session we practiced the introductory concepts only. Neural Networks has much more advanced techniques. There are many algorithms other than back propagation.
- Neural networks particularly work well on some particular class of problems like image recognition.
- The neural networks algorithms are very calculation intensive. They require highly efficient computing machines. Large datasets take significant amount of runtime.
- Currently there is a lot of exciting research is going on, around neural networks.
- After gaining sufficient knowledge in this basic session, you may want to explore reinforced learning, deep learning etc.,

Appendix

statinfer

Math- How to update the weights?

Math- How to update the weights?



$$\sum_{i=1}^{n}\left[y_i - g\left(\sum_{k=1}^{m} w_k h_{ki}\right)\right]^2$$

- We update the weights backwards by iteratively calculating the error
- The formula for weights updating is done using gradient descent method or delta rule also known as Widrow-Hoff rule
- First we calculate the weight corrections for the output layer then we take care of hidden layers

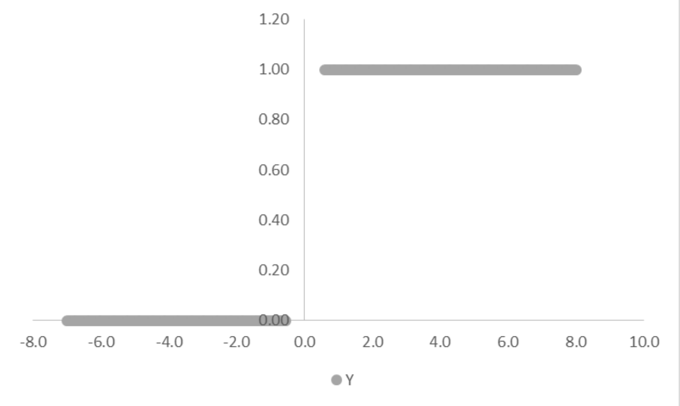
Math- How to update the weights?

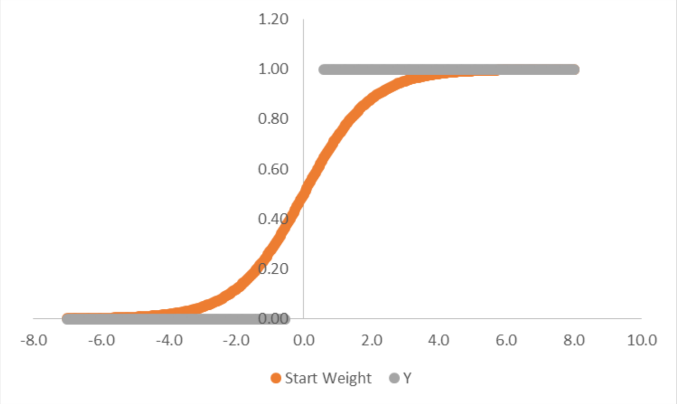
- $W_{jk} := W_{jk} + \Delta W_{jk}$
  - $where\ \Delta W_{jk} = \eta . y_{\mathrm{j}} \delta_k$
  - η is the learning parameter
  - $\delta_k = y_k(1 - y_k) * Err$ (for hidden layers $\delta_k = y_k(1 - y_k) * w_j * Err$)
  - Err=Expected output-Actual output
- The weight corrections is calculated based on the error function
- The new weights are chosen in such way that the final error in that network is minimized

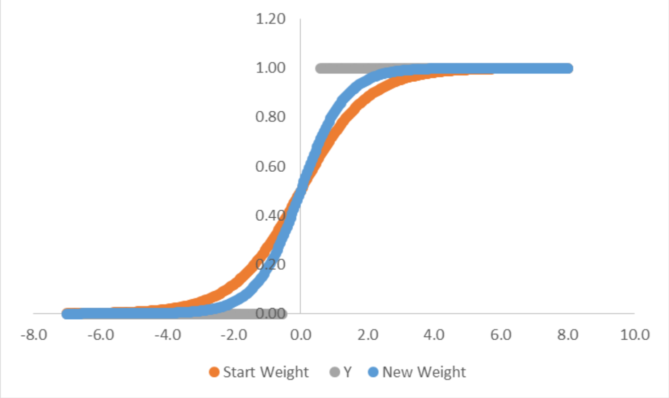
Math-How does the delta rule work?

How does the delta rule work?

- Lets consider a simple example to understand the weight updating using delta rule.





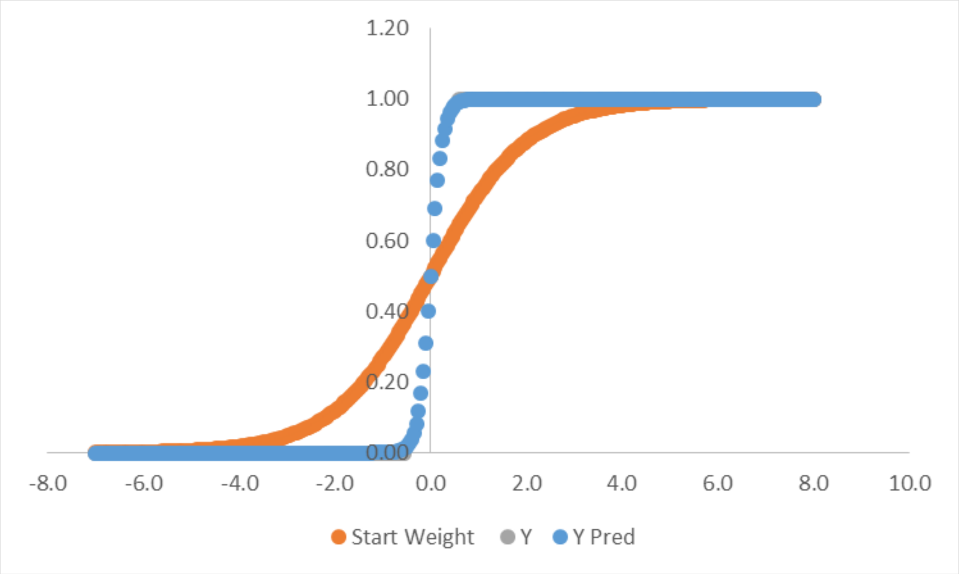


- If we building a simple logistic regression line. We would like to find the weights using weight update rule
- $Y=1/(1+e^{-wx})$ is the equation
- We are searching for the optimal w for our data

- Let w be 1
- $Y=1/(1+e^{-x})$ is the initial equation
- The error in our initial step is 3.59
- To reduce the error we will add a delta to w and make it 1.5

- Now w is 1.5 (blue line)
- $Y=1/(1+e^{-1.5x})$ the updated equation
- With the updated weight, the error is 1.57
- We can further reduce the error by increasing w by delta

How does the delta rule work?



- If we repeat the same process of adding delta and updating weights, we can finally end up with minimum error
- The weight at that final step is the optimal weight
- In this example the weight is 8, and the error is 0
- $Y=1/(1+e^{-8x})$ is the final equation

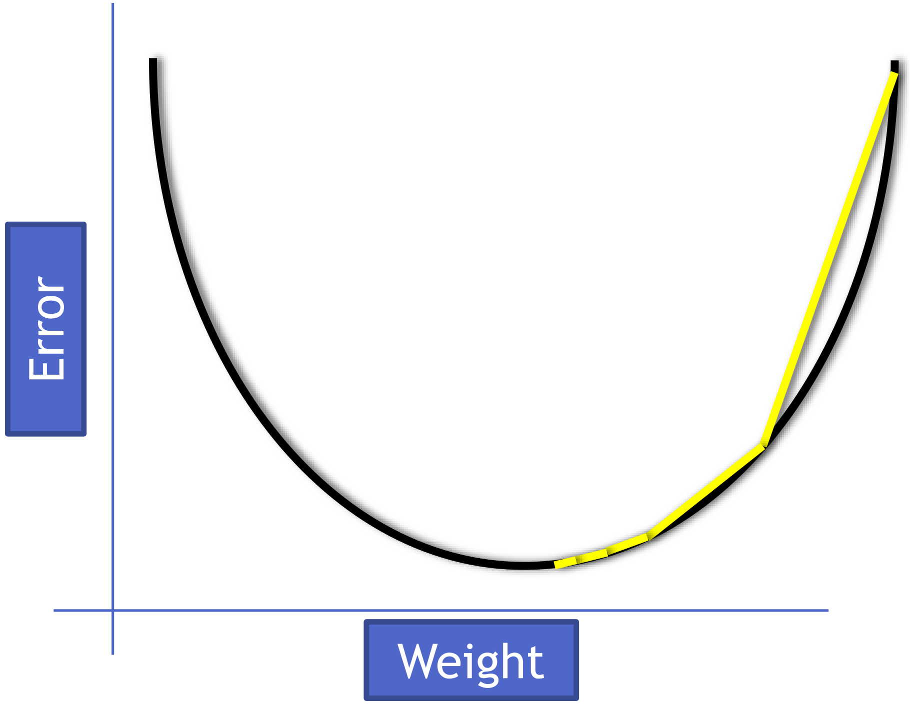
- In this example, we manually changed the weights to reduce the error. This is just for intuition, manual updating is not feasible for complex optimization problems.
- In gradient descent is a scientific optimization method. We update the weights by calculating gradient of the function.

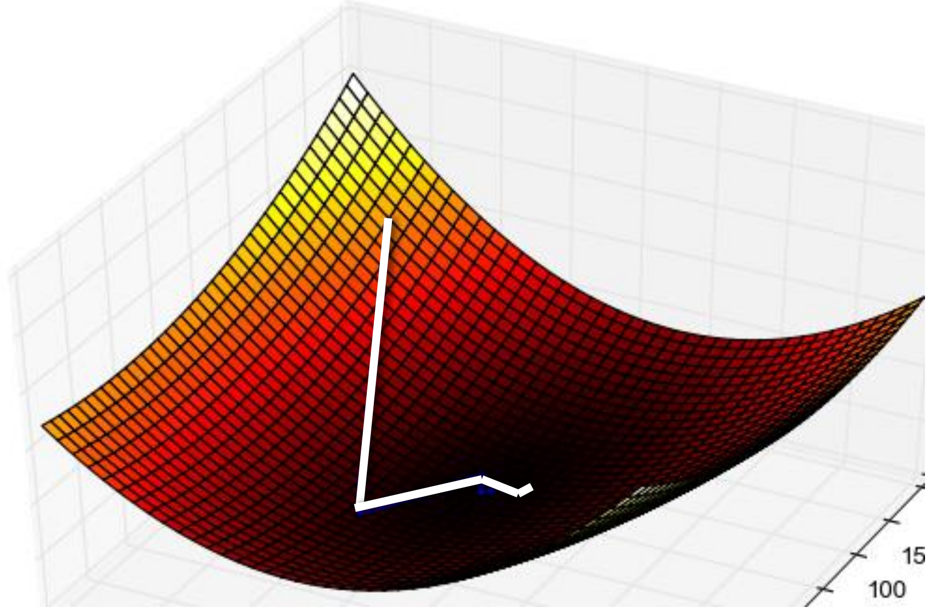
statinfer

Math-How does gradient descent work?

How does gradient descent work?

- Gradient descent is one of the famous ways to calculate the local minimum
- By Changing the weights we are moving towards the minimum value of the error function. The weights are changed by taking steps in the negative direction of the function gradient(derivative)



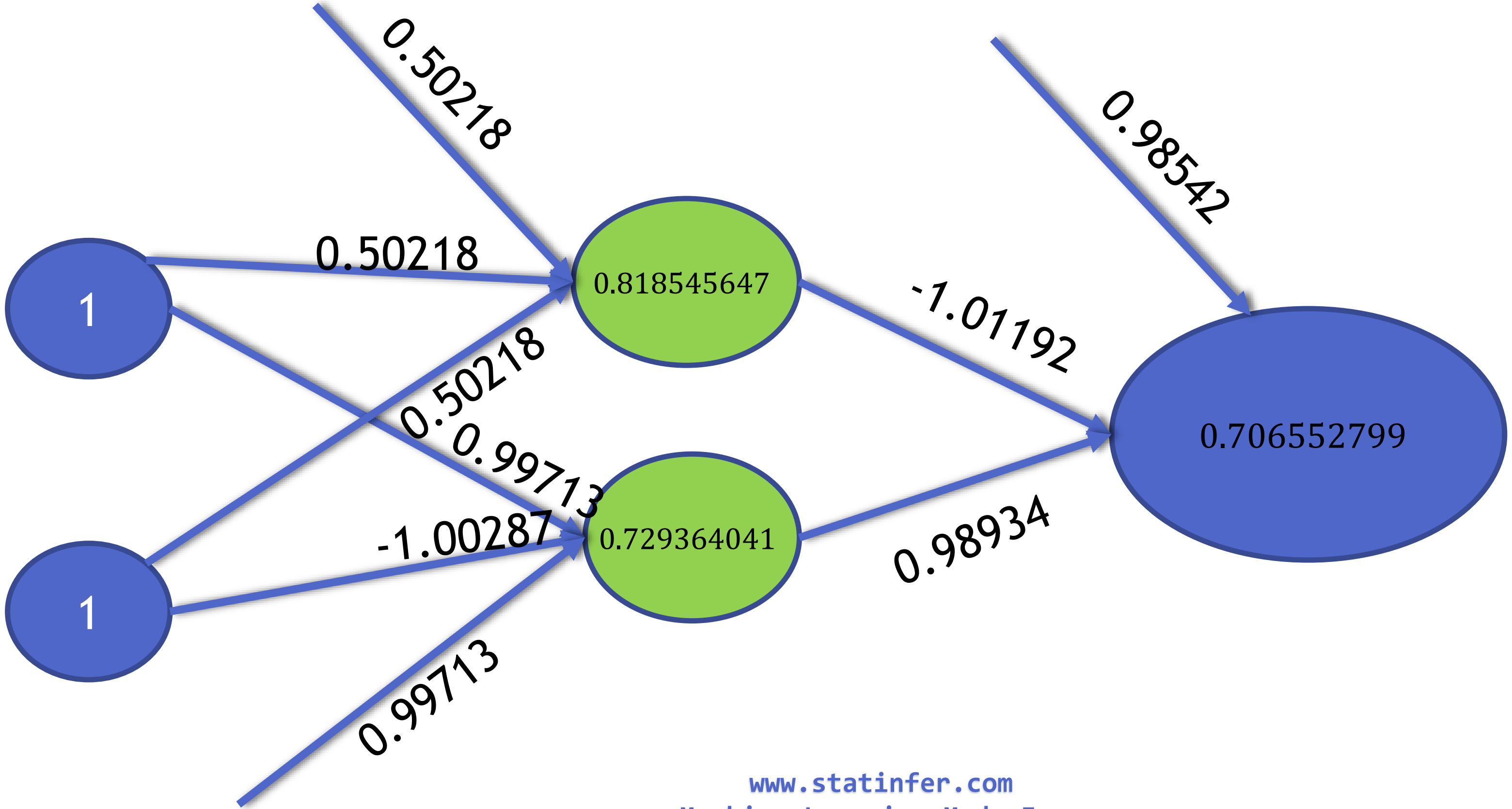


statinfer

Demo-How does gradient descent work?

Does this method really work?

- We changed the weights did it reduce the overall error?
- Lets calculate the error with new weights and see the change



Gradient Descent method validation

- With our initial set of weights the overall error was 0.7137,Y Actual is 0, Y Predicted is 0.7137 error =0.7137
- The new weights give us a predicted value of 0.70655
- In one iteration, we reduced the error from 0.7137 to 0.70655
- The error is reduced by 1%. Repeat the same process with multiple epochs and training examples, we can reduce the error further.

	input1	input2	Output(Y-Actual)	Y Predicted	Error
Old Weights	1	1	0	0.71371259	0.71371259
Updated Weights	1	1	0	0.706552799	0.706552799

Thank you

Part 10/12 -Support Vector Machines in Azure

Venkat Reddy

Contents

Contents

- Introduction
- The decision boundary with largest margin
- SVM- The large margin classifier
- SVM algorithm
- The kernel trick
- Building SVM model
- Conclusion

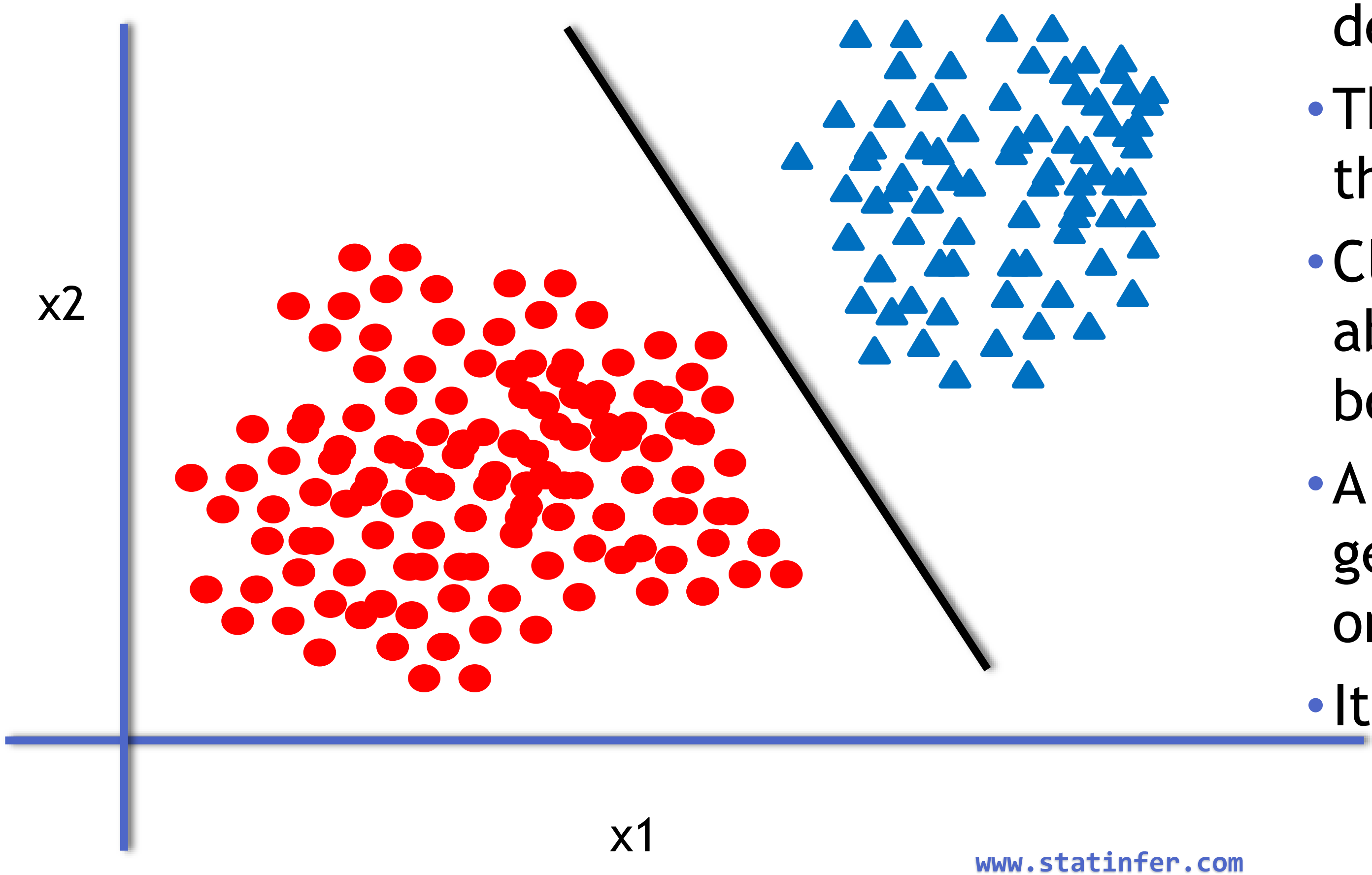
Introduction

Introduction

- SVM is another black box method in Machine Learning space
- Compared to other ml algorithms, SVM totally a different approach to learning.
- The in-depth theory and mathematics of SVM needs great knowledge in vector algebra and numerical analysis
- We will try to learn the basic principal, philosophy, implementation of SVM
- SVM was first introduced by Vapnik and Chervonenkis
- Neural networks try to reduce the squared error and often suffer from overfitting.
- SVM algorithm has better generalization ability. There are many applications where SVM works better than neural networks

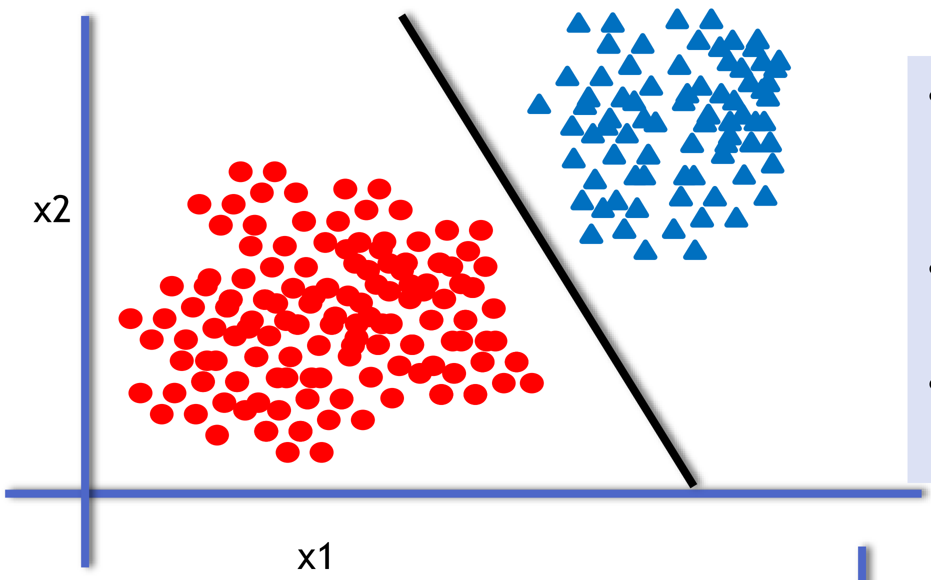
The Classifier

The Classifier

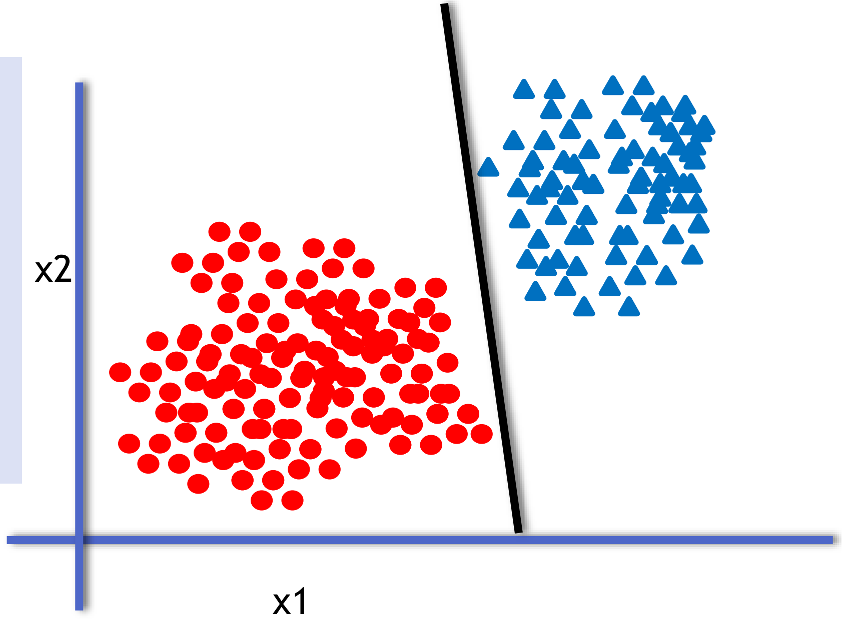


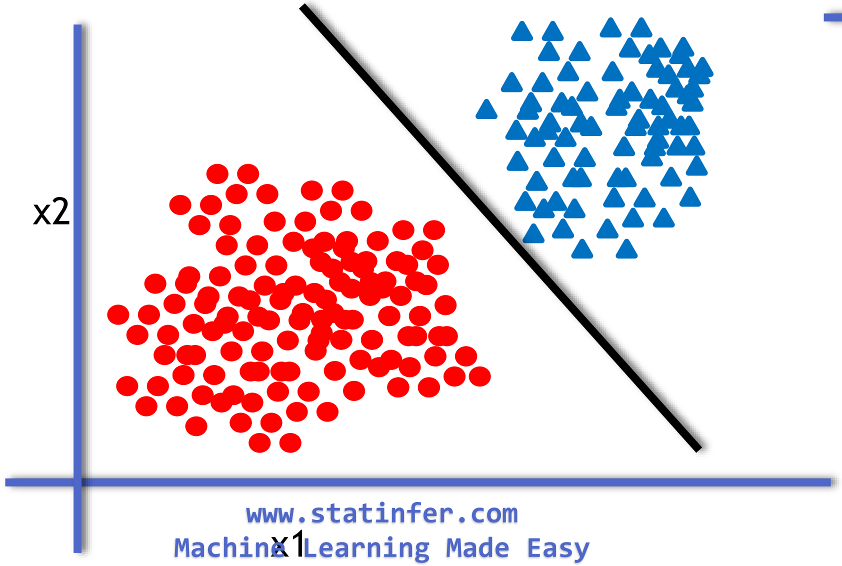
- To understand the SVM algorithm easily, we will start with the decision boundary
- The line or margin that separates the classes
- Classification algorithms are all about finding the decision boundaries
- A good classifier is the one that generalizes well. It should work well on both training and testing data
- It need not be a straight line always

Many Classifiers



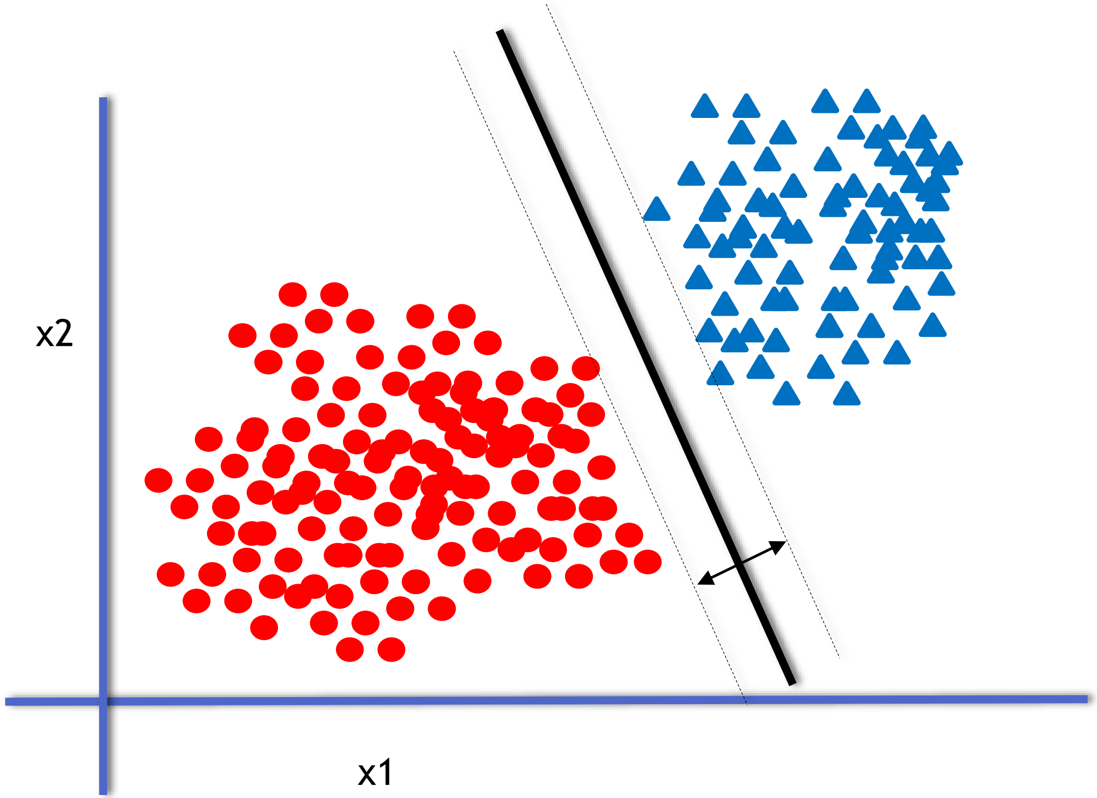
- There can be many classifiers. That may work for a given dataset.
- They might even have same level of accuracy
- How to choose the best classifier ?

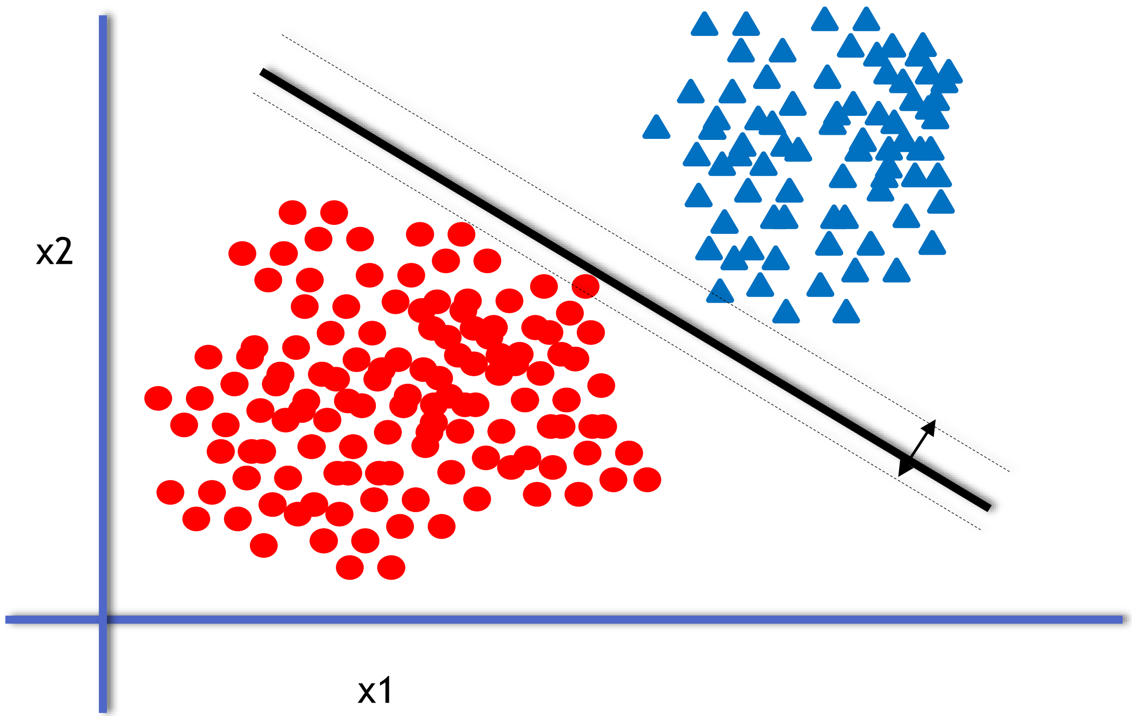




The Margin of classifier

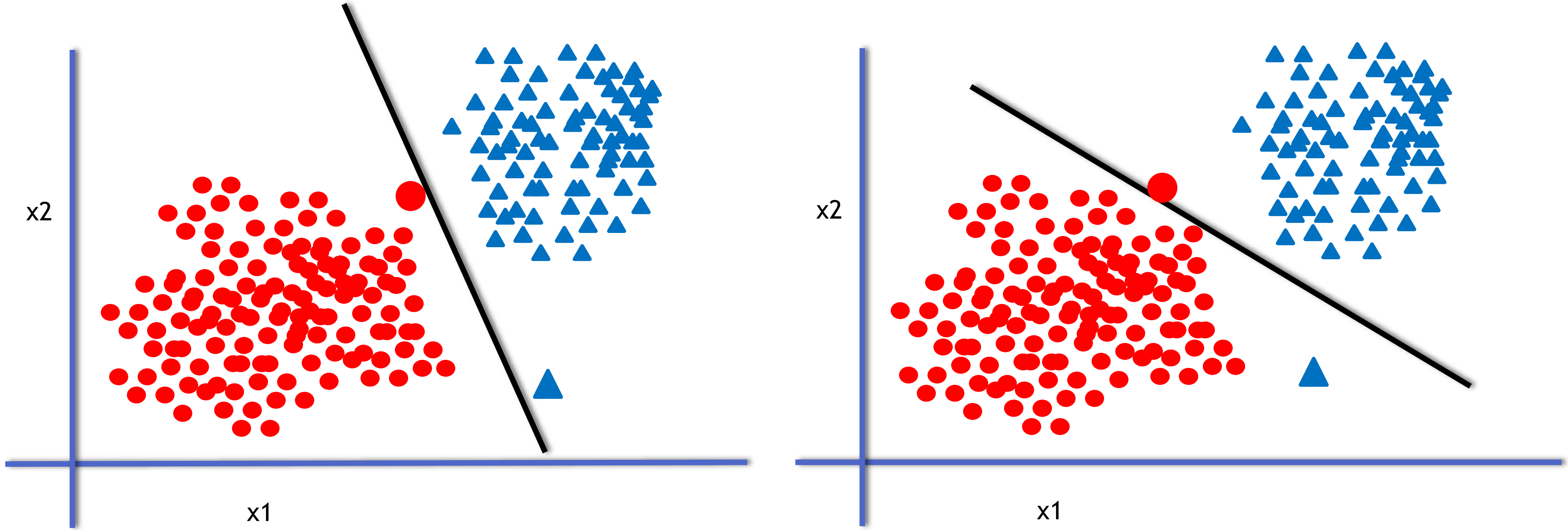
The margin of classifier





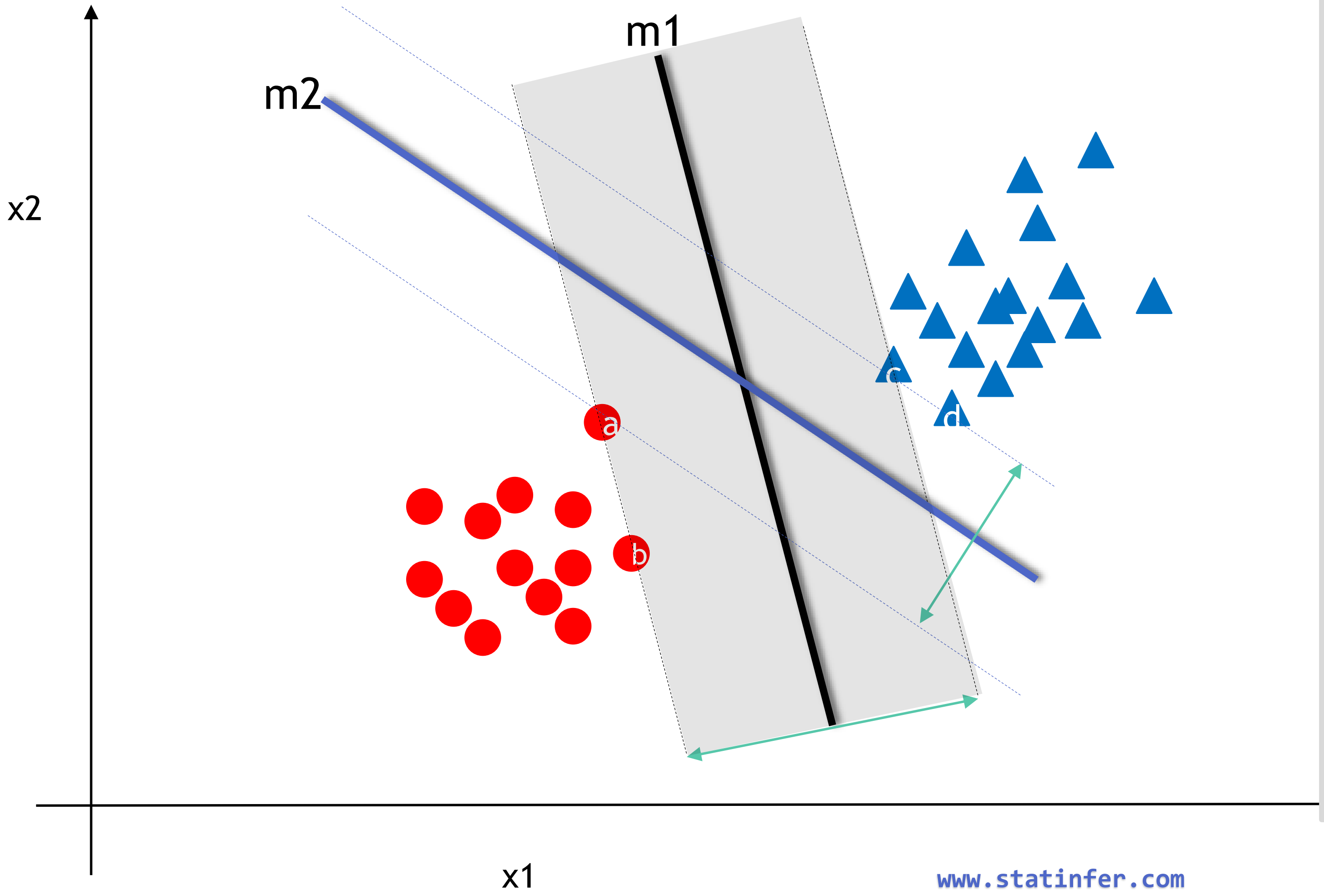
Out all the classifiers, the one that has maximum margin will generalize well. But why?

The best decision boundary



- Out all the classifiers, the one that has maximum margin will generalize well. But why?
- Imagine two more data points. The classifier with maximum margin will be able to classify them more accurately.

The Maximum Margin Classifier



- So, the best classifier has maximum margin
- The classifier that maximizes the distance between itself and the nearest training data
- In our example a,b,c are the training data points that are near to m1, and a,c,d are the training examples that are near to model m2.
- The model m1 has maximum margin
- The model m1 works well with the unseen examples
- The model m1 does good generalization
- For a given dataset, if we can find a classifier that has maximum margin, then it will assure maximum accuracy.

LAB: Simple Classifiers

LAB: Simple Classifiers

- Dataset: Fraud Transaction/Transactions_sample.csv
- Draw a classification graph that shows all the classes
- Build a logistic regression classifier
- Draw the classifier on the data plot

Steps - Simple Classifiers

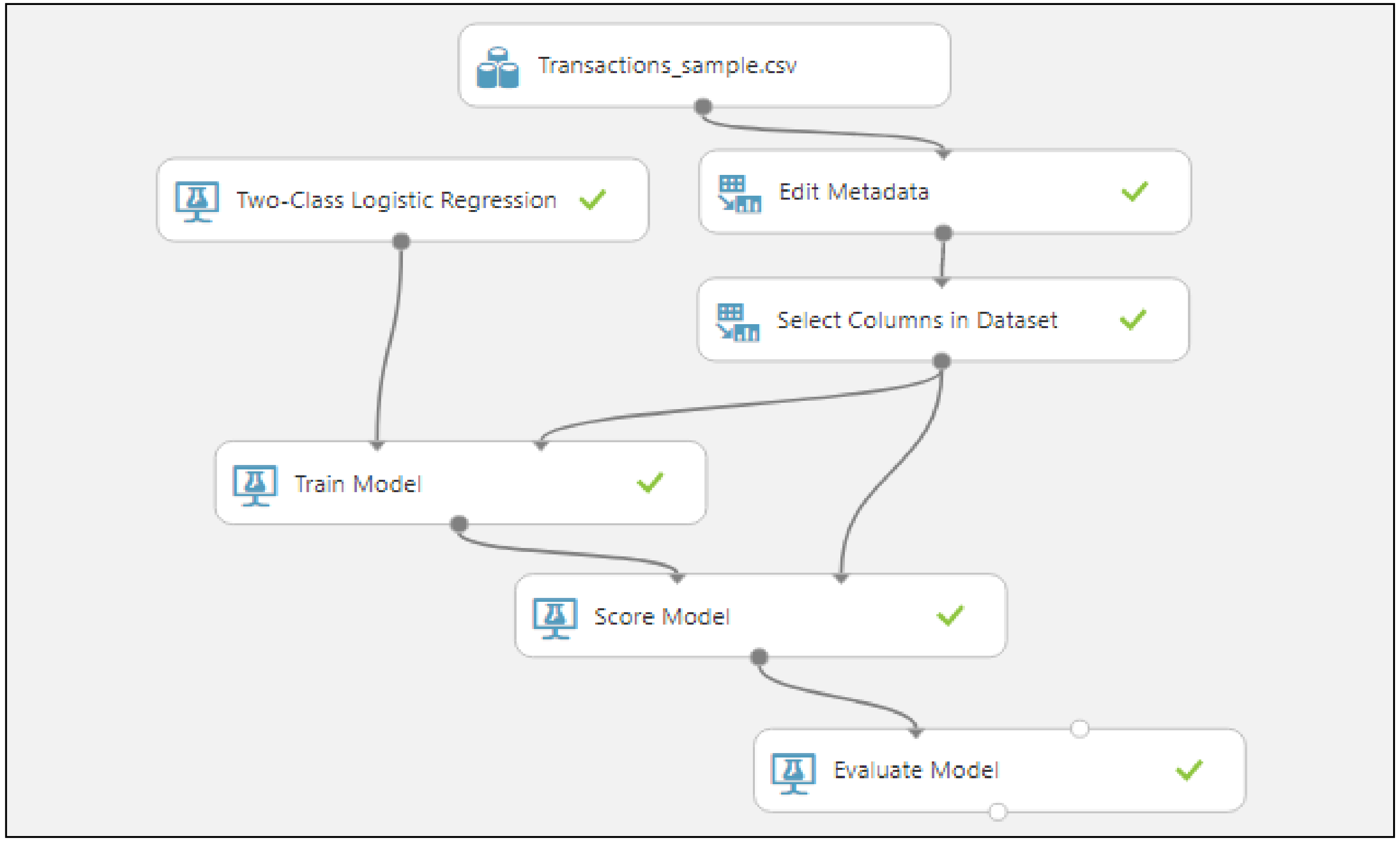
- Drag and drop the **Dataset** into the canvas
- Drag and drop the **Edit Metadata** and connect it to the dataset
- Drag and drop the **Select Columns from the Dataset** and select the columns, connect it to the **Edit Metadata**
- Drag and drop **Two-Class Logistic Regression, Train Model, Score Model** and **Evaluate Model**
- Connect **Two-Class Boosted Logistic Regression** to the first input of **Train Model** and **Select Columns from the Dataset** to the Second input of **Train Model**

Steps - Simple Classifiers

- Connect the output of **Train Model** first input of **Score Model** and **Select Columns from the Dataset** to the Second input of **Score Model**
- Connect the output of **Score Model** to the input of **Evaluate Model**
- Click on Train Model and select the column for which the prediction is done(Fraud_id)
- Click run and visualize the output of **Evaluate Model**

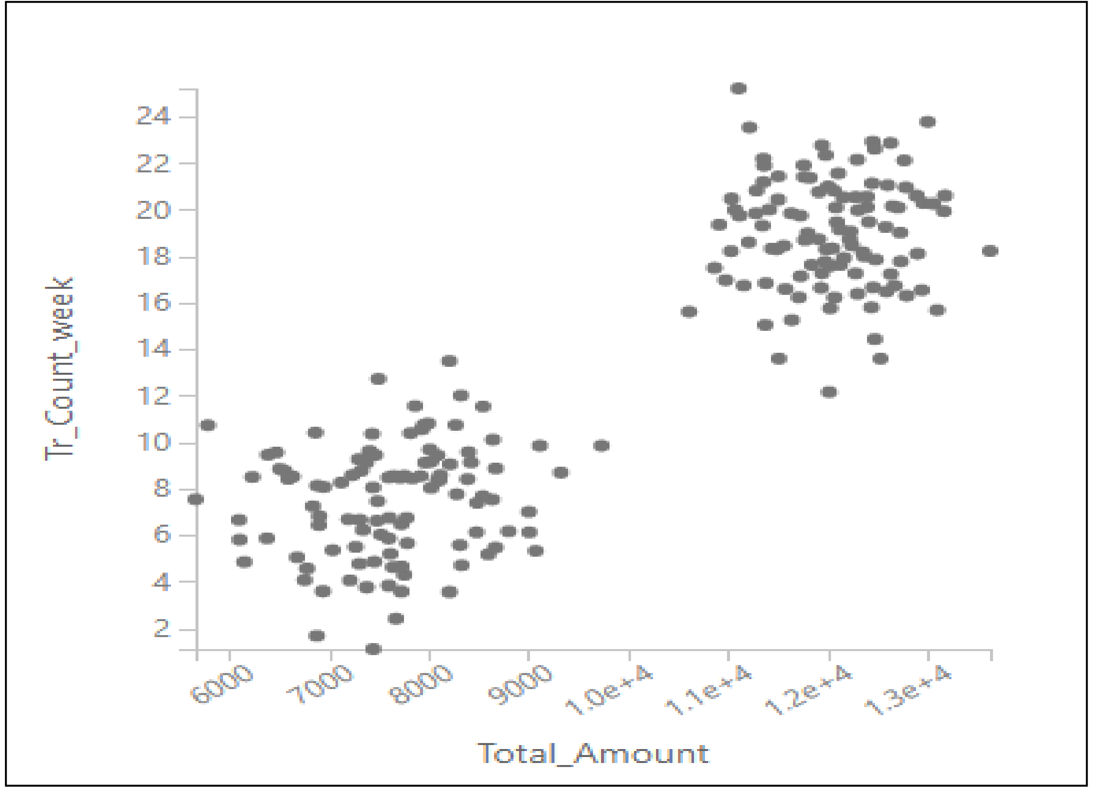
Steps - Simple Classifiers

Fig1: Logistic Regression (Transaction_sample)



Steps - Simple Classifiers

Fig2: Scatter plot – Total_Amount vs Tr_Count_week (Classification Graph)



Steps - Simple Classifiers

Fig3: Properties - Edit Metadata

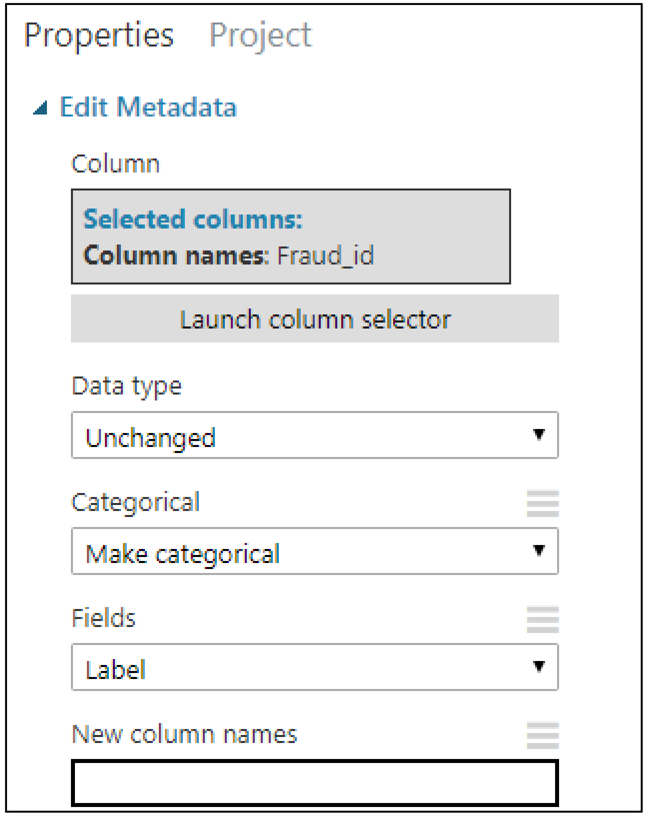
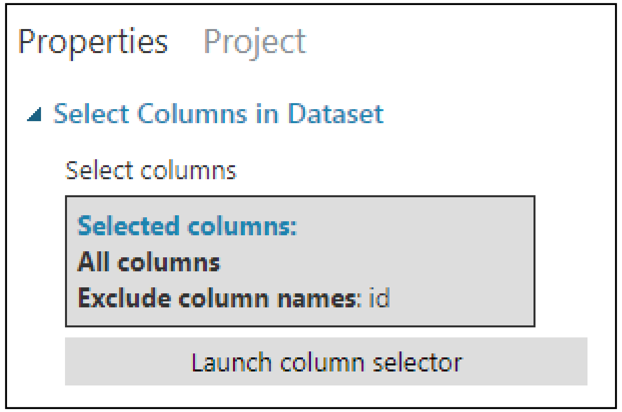


Fig4: Properties - Select Columns from Dataset



Steps - Simple Classifiers

Fig5: Properties - Logistic Regression

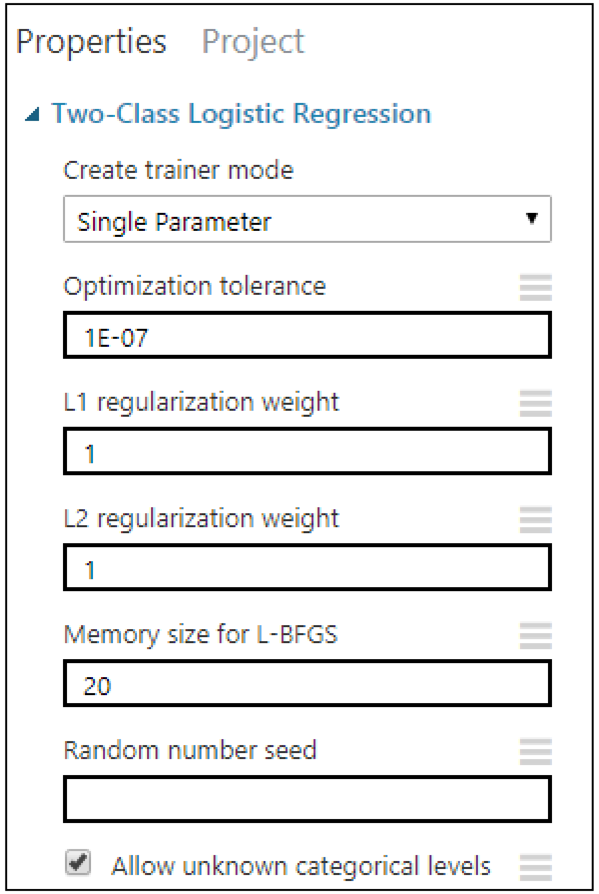
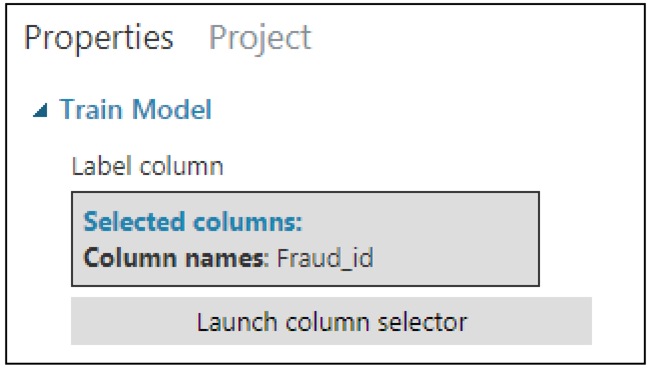


Fig6: Properties - Train Model



Steps - Simple Classifiers

Fig7: Accuracy

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
105	1	0.995	1.000	0.5	0.992
False Positive	True Negative	Recall	F1 Score		
0	104	0.991	0.995		
Positive Label	Negative Label				
1	0				

statinfer

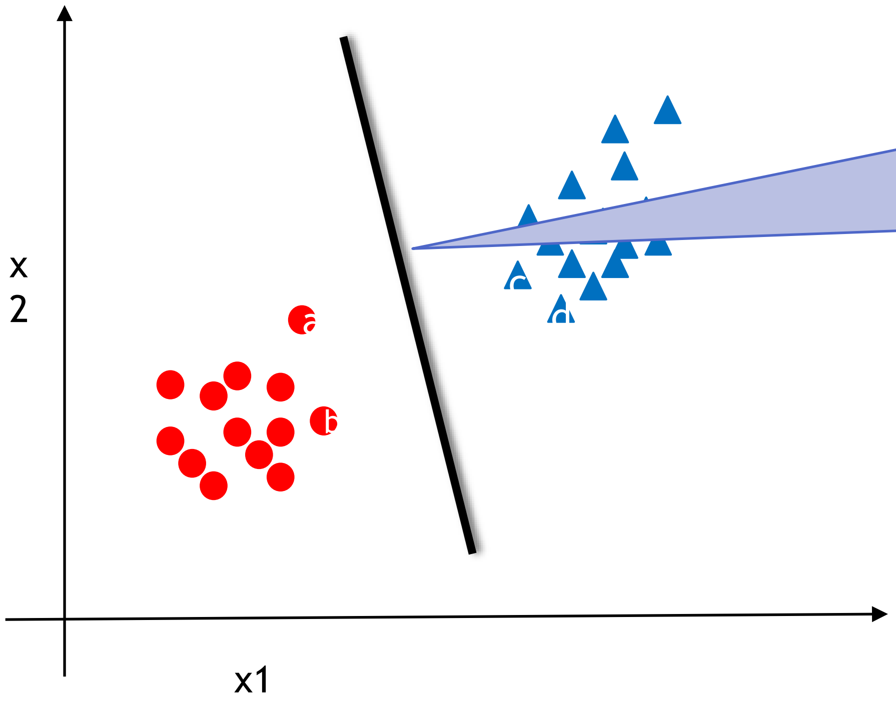
SVM- The large margin classifier

SVM- The large margin classifier

- SVM is all about finding the maximum-margin Classifier.
- Classifier is a generic name, its actually called the hyper plane
  - **Hyper plane:** In 3-dimensional system hyperplanes are the 2-dimensional planes, in 2-dimensional space its hyperplanes are the 1-dimensional lines.
- SVM algorithm makes use of the nearest training examples to derive the classifier with maximum margin
- Each data point is considered as a p-dimensional vector (a list of p numbers)
- SVM uses vector algebra and mathematical optimization to find the optimal hyperplane that has maximum margin

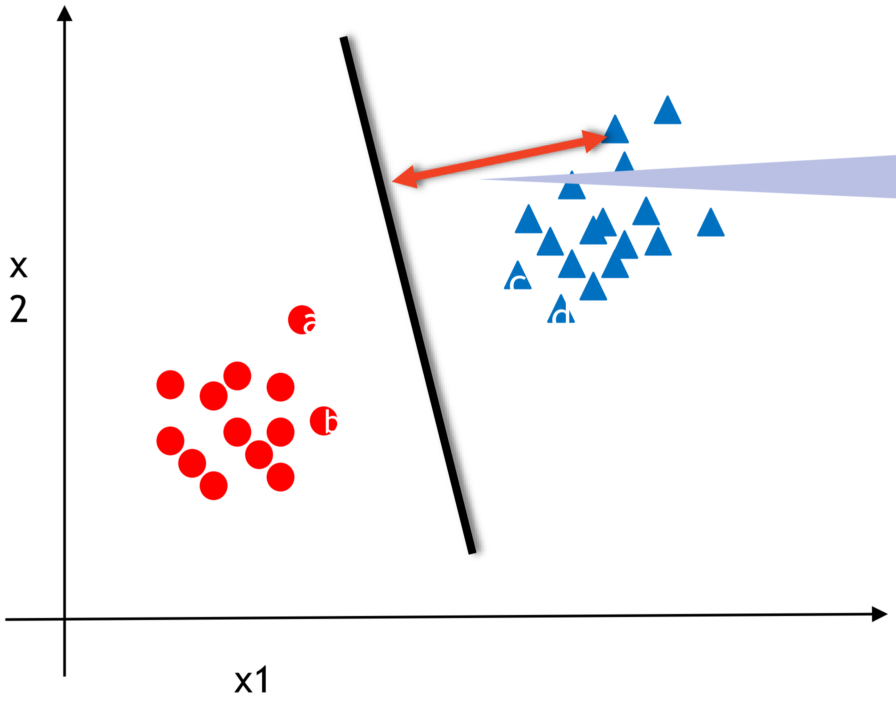
The SVM Algorithm

The SVM Algorithm



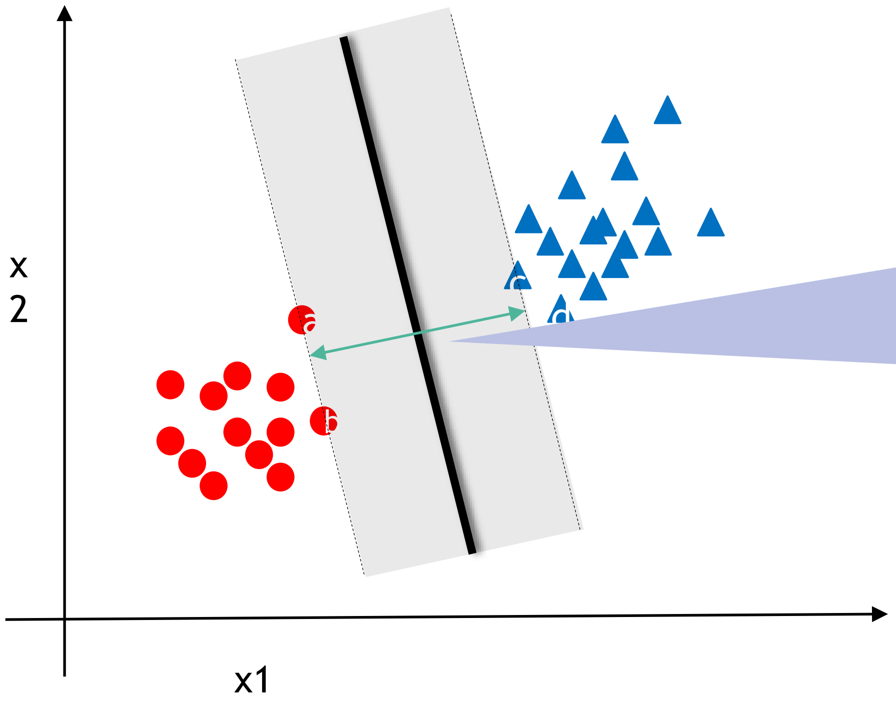
- If a dataset is linearly separable then we can always find a hyperplane f(x) such that
  - For all negative labeled records f(x)<0
  - For all positive labeled records f(x)>0
  - This hyper plane f(x) is nothing but the linear classifier
  - $f(x)=w_1x_1+ w_2x_2 +b$
  - $f(x)=w^Tx+b$

The SVM Algorithm



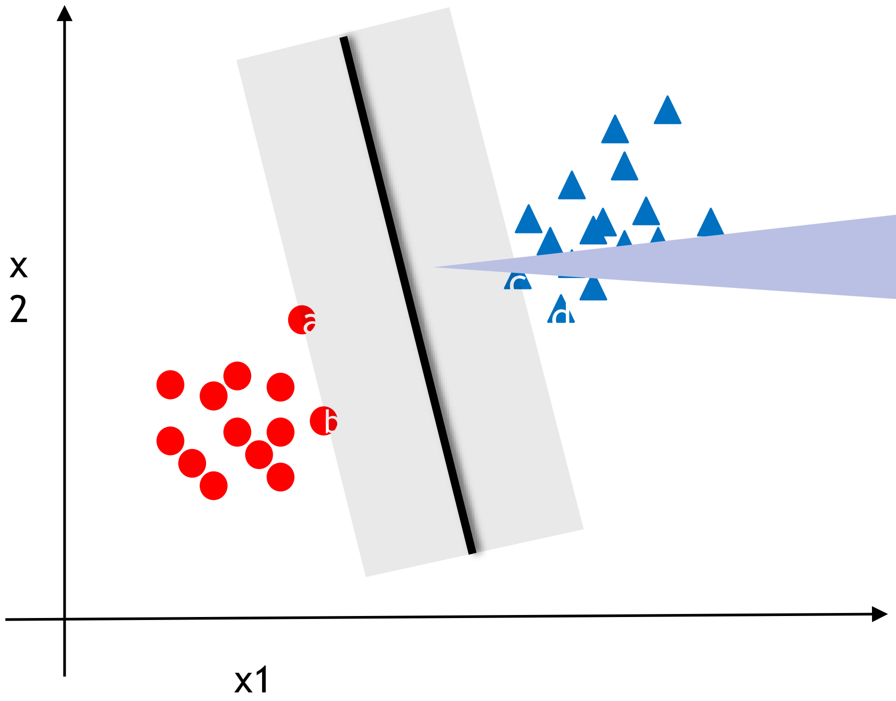
Given that plane, we can calculate the distance between each data vector(point)

The SVM Algorithm



- Some points on the edges are close to the hyperplane
- The distance between these points and the hyperplane is known as margin of the classifier

The SVM Algorithm



- All those data points(vectors) that are on the edge, that constitute to the margin of classifier are called support vectors
- Vectors a, b & c are support vectors
- Based on these hyperplane with maximum margin will be calculated

statinfer

Math behind SVM Algorithm

SVM Algorithm – The Math

If you have already understood the SVM technique and If you find this slide is too technical, you may want to skip it. The tool will take care of this optimization

1. $f(x)=w^Tx+b$
2. $w^Tx^+ +b=1$ and $w^Tx^- +b=-1$
3. $x^+=x^- + \lambda w$
4. $w^Tx^+ +b=1$
   - $w^T(x^- + \lambda w)+b=1$
   - $w^Tx^- +\lambda w.w+b=1$
   - $-1+\lambda w \cdot w=1$
   - $\lambda=2/w \cdot w$
5. Margin $m = |x^+ - x^-|$
   - $m=|\lambda w|$
   - $m=(2/w \cdot w)*|w|$
   - $m=2/||w||$
6. Objective is to maximize $2/||w||$
   - i.e minimize $||w||$
7. A good decision boundary should be
   - $w^Tx^+ +b>=1$ for all $y=1$
   - $w^Tx^- +b<=-1$ for all $y=-1$
   - i.e $y*(w^Tx+b)>=1$ for all points
8. Now we have the optimization problem with objective and constraints
   - minimize $||w||$ or $(½)*||w||^2$
   - With constant $y(w^Tx+b)>=1$
9. We can solve the above optimization problem to obtain w & b

SVM Result

SVM Result

- SVM doesn't output probability. It directly gives which class the new data point belongs to
- For a new point x_k calculate $w^T x_k + b$. If this value is positive then the prediction is +1 else -1

LAB: First SVM Learning Problem

- Dataset: Fraud Transaction/Transactions_sample.csv
- Draw a classification graph that shows all the classes
- Build a SVM classifier
- Draw the classifier on the data plots
- Predict the (Fraud vs not-Fraud) class for the data points Total_Amount=11000, Tr_Count_week=15 & Total_Amount=2000, Tr_Count_week=4
- Download the complete Dataset: Fraud Transaction/Transaction.csv
- Draw a classification graph that shows all the classes
- Build a SVM classifier
- Draw the classifier on the data plots

Steps - First SVM Learning Problem

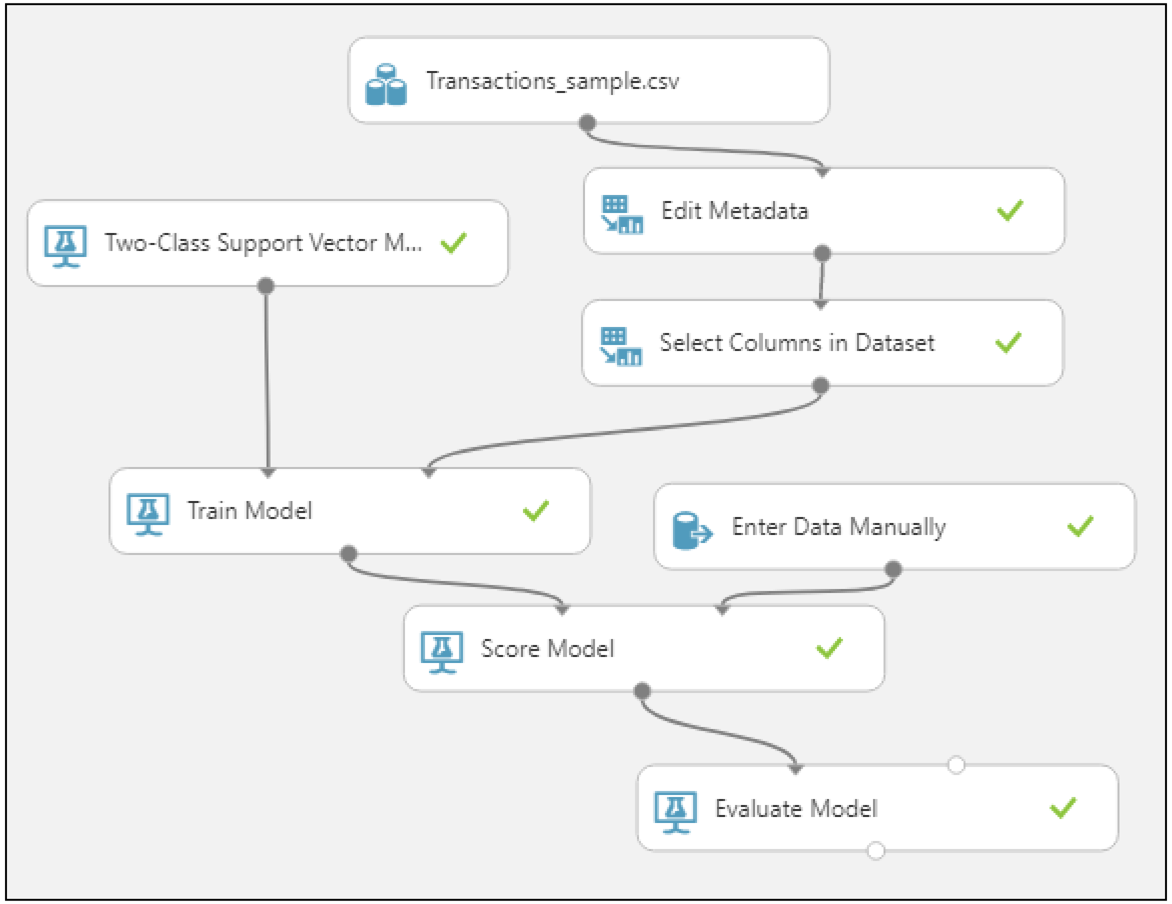
- Drag and drop the **Dataset** into the canvas
- Drag and drop the **Edit Metadata** and connect it to the dataset
- Drag and drop the **Select Columns from the Dataset** and select the columns, connect it to the **Edit Metadata**
- Drag and drop **Two-Class Support Vector Machine, Train Model, Score Model, Enter data Manually** and **Evaluate Model**
- Connect **Two-Class Support Vector Machine** to the first input of **Train Model** and **Select Columns from the Dataset** to the Second input of **Train Model**

Steps - First SVM Learning Problem

- Connect the output of **Train Model** first input of **Score Model** and **Enter Data Manually** to the Second input of **Score Model**
- Connect the output of **Score Model** to the input of **Evaluate Model**
- Click on Train Model and select the column for which the prediction is done(Fraud_id)
- Click run and visualize the output of **Evaluate Model** and **Score Model**

Steps - First SVM Learning Problem

Fig8: Support Vector Machine (Transaction_sample)



Steps - First SVM Learning Problem

Fig9: Properties - Edit Metadata

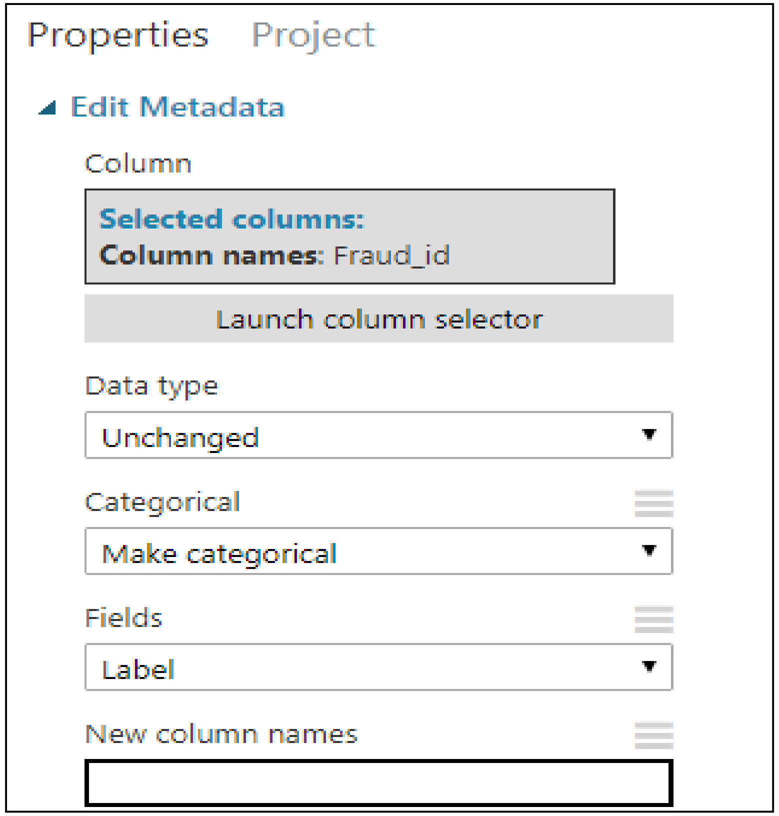
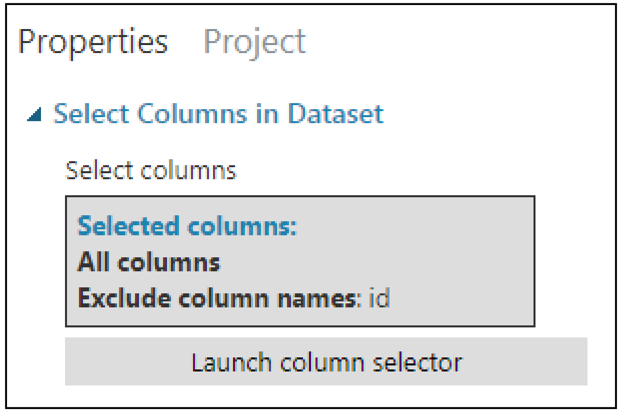


Fig10: Properties - Select Columns from Dataset



Steps - First SVM Learning Problem

Fig11: Properties - Support Vector Machine

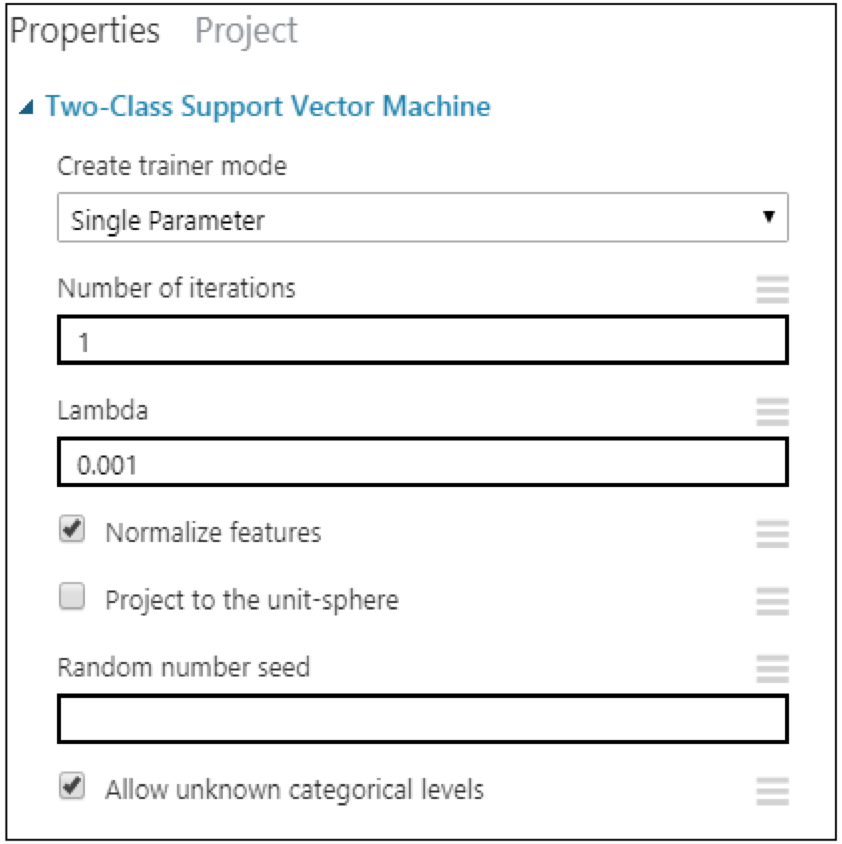
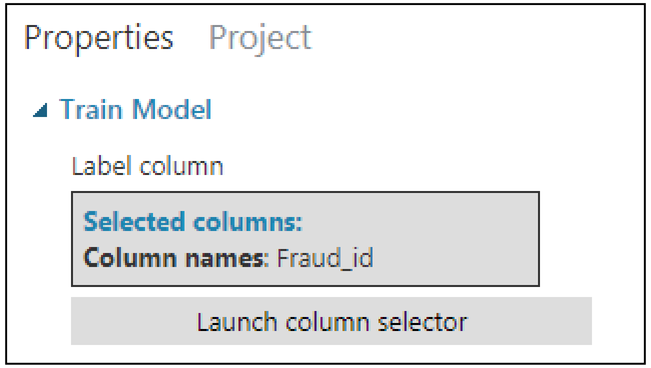


Fig12: Properties - Train Model



Steps - First SVM Learning Problem

Fig13: Data For Prediction

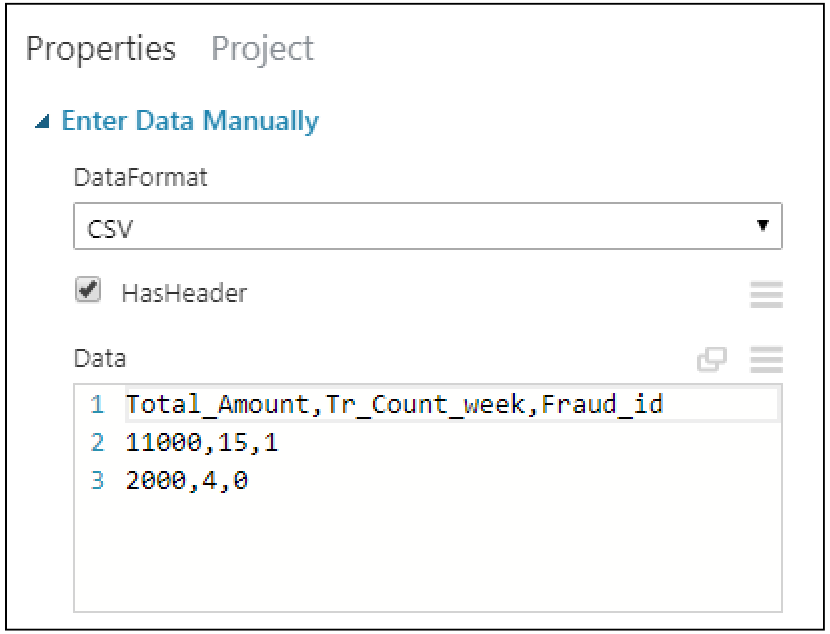
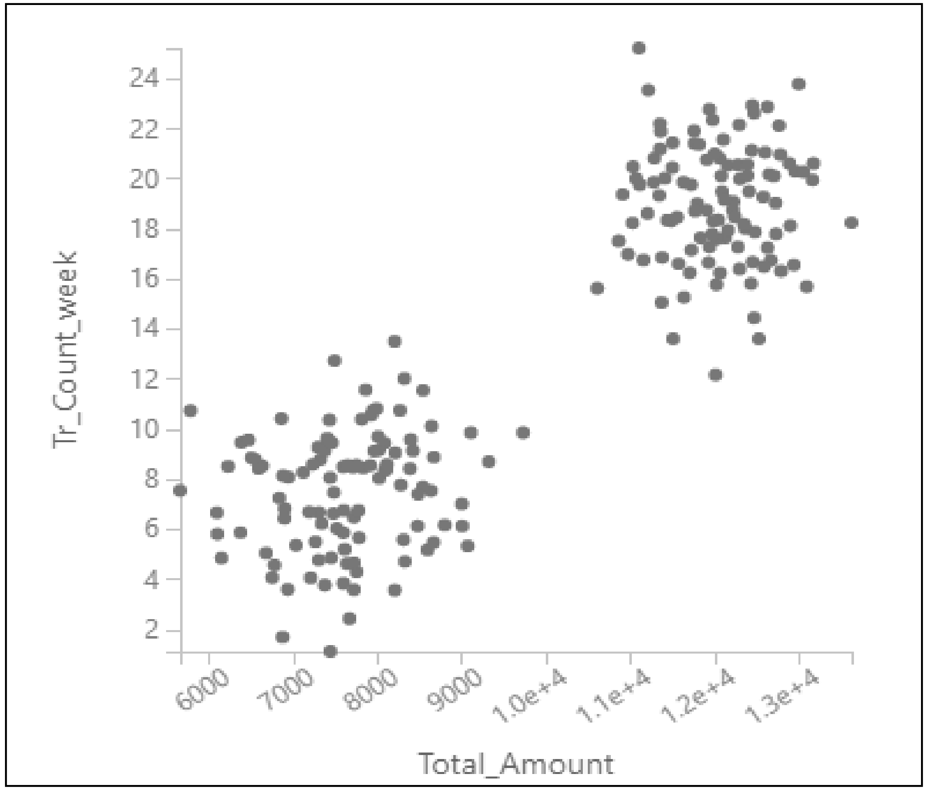
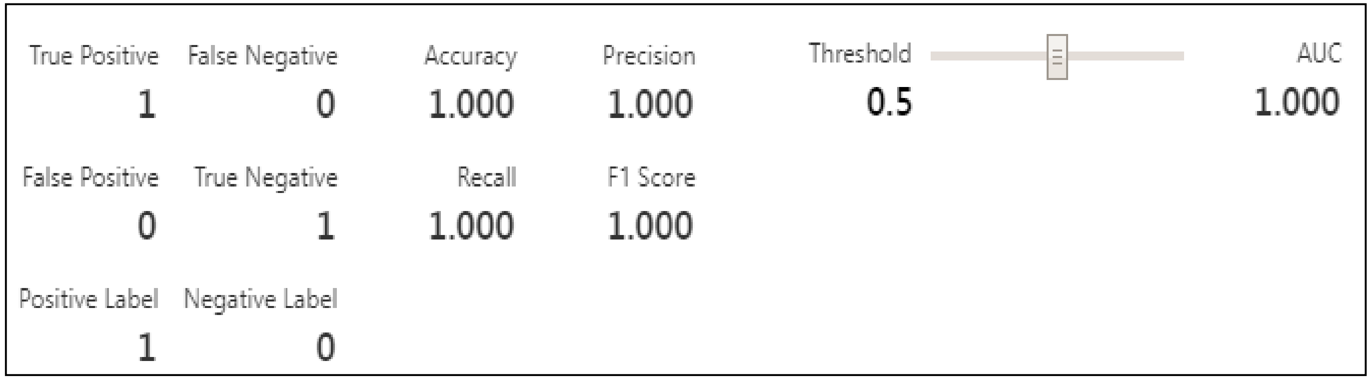


Fig14: Classifier Data Plots



Steps - First SVM Learning Problem

Fig15: Accuracy - Test Data



Steps - First SVM Learning Problem

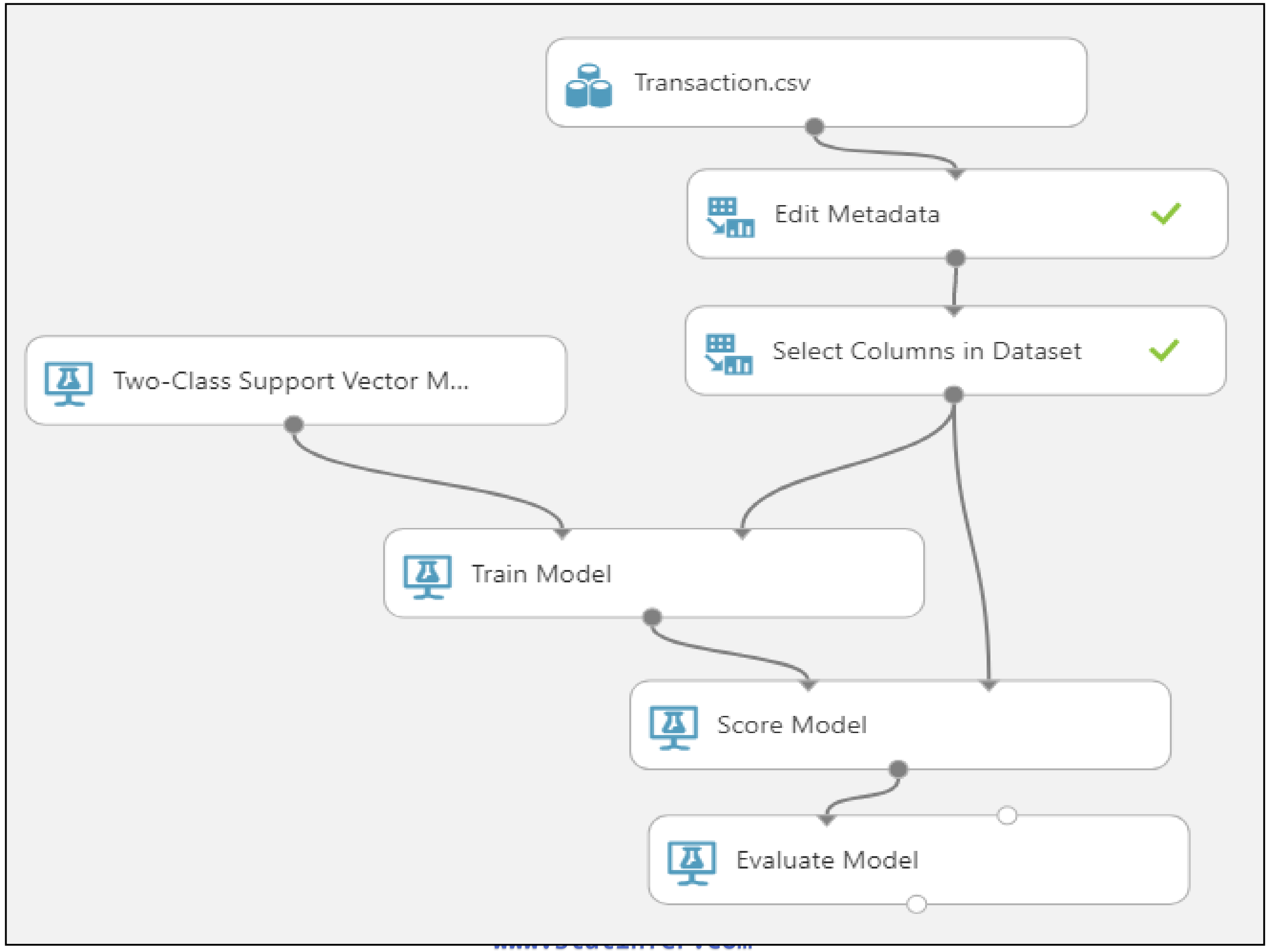
- Drag and drop the **Dataset** into the canvas
- Drag and drop the **Edit Metadata** and connect it to the dataset
- Drag and drop the **Select Columns from the Dataset** and select the columns, connect it to the **Edit Metadata**
- Drag and drop **Two-Class Support Vector Machine, Train Model, Score Model** and **Evaluate Model**
- Connect **Two-Class Support Vector Machine** to the first input of **Train Model** and **Select Columns from the Dataset** to the Second input of **Train Model**

Steps - First SVM Learning Problem

- Connect the output of **Train Model** first input of **Score Model** and **Select Columns from the Dataset** to the Second input of **Score Model**
- Connect the output of **Score Model** to the input of **Evaluate Model**
- Click on Train Model and select the column for which the prediction is done(Fraud_id)
- Click run and visualize the output of **Evaluate Model**

Steps - First SVM Learning Problem

Fig16: Support Vector Machine(Transaction)



Steps - First SVM Learning Problem

Fig17: Properties - Edit Metadata

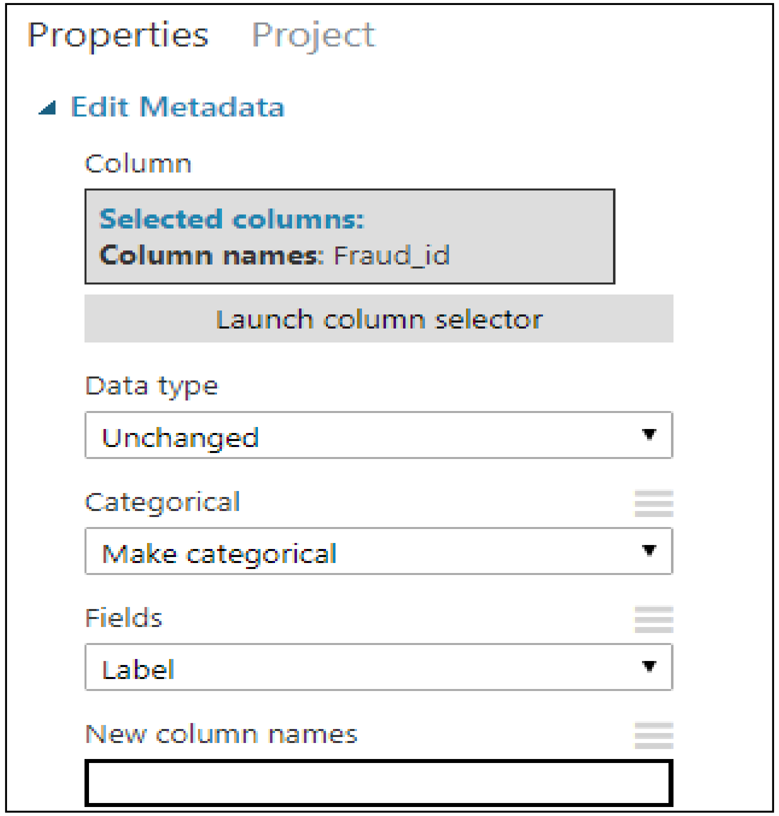
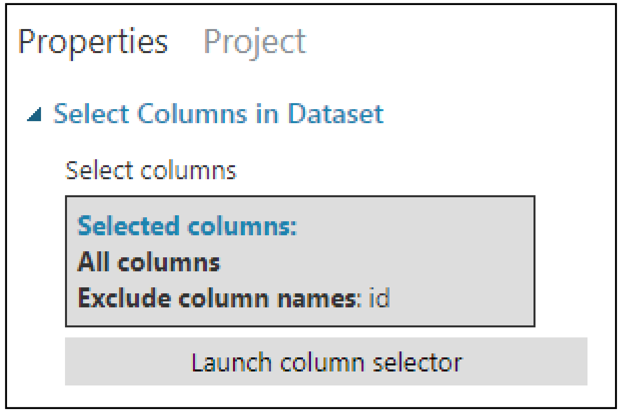


Fig18: Properties - Select Columns from Dataset



Steps - First SVM Learning Problem

Fig19: Properties - Support Vector Machine

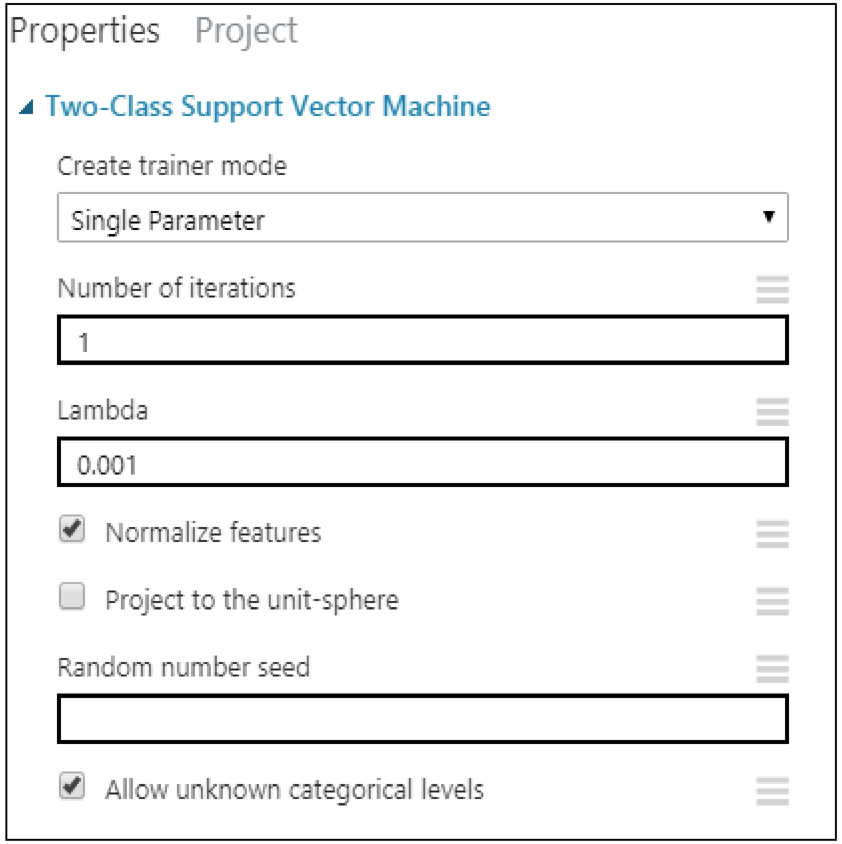
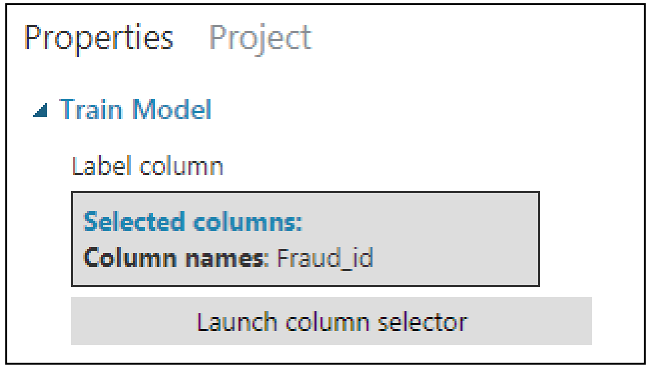
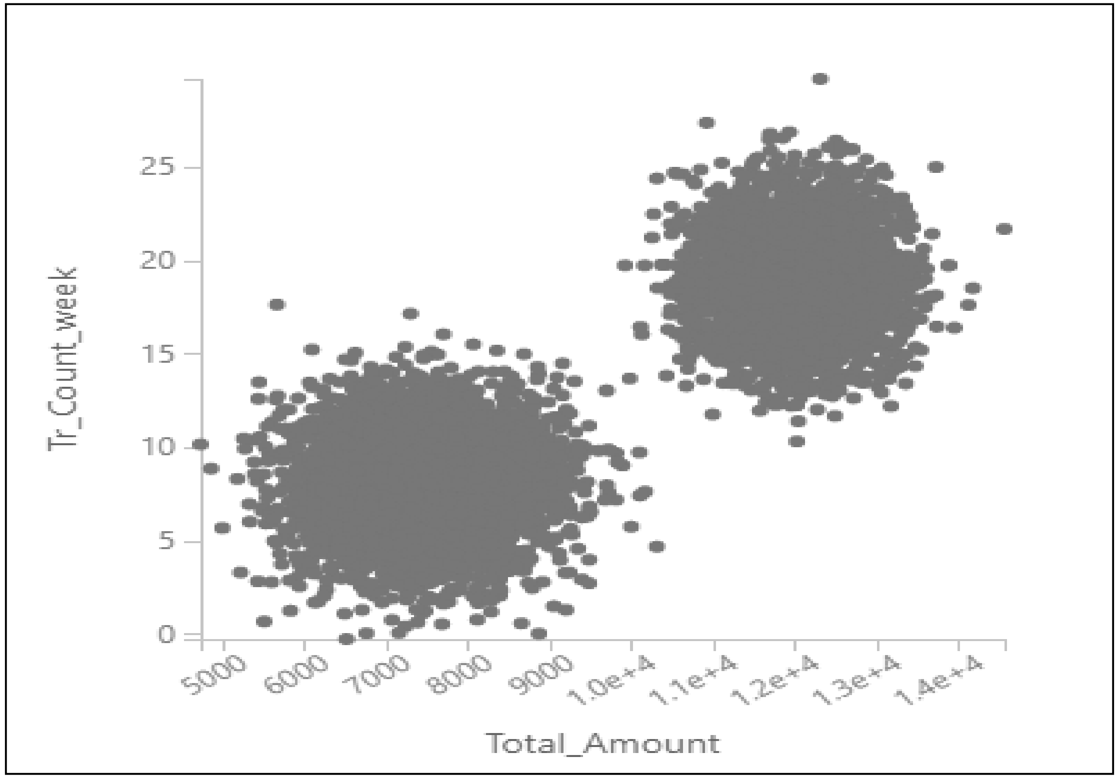


Fig20: Properties - Train Model



Steps - First SVM Learning Problem

Fig21: Classifier Data Plot



Steps - First SVM Learning Problem

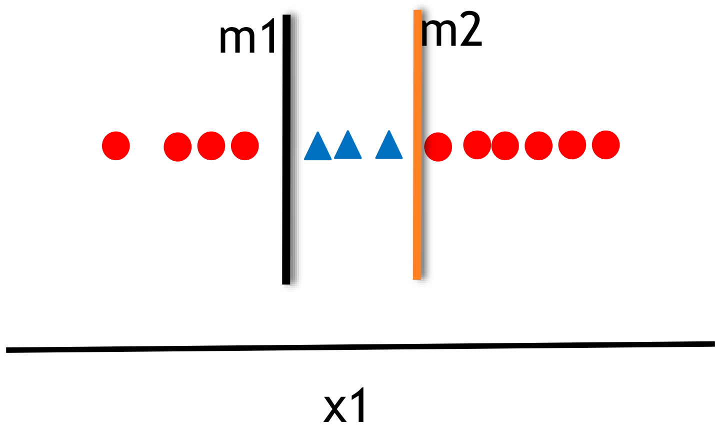
Fig22: Accuracy

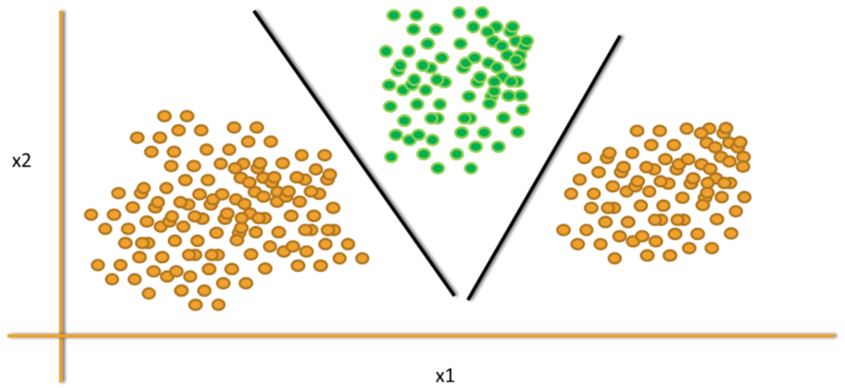
True Positive	False Negative	Accuracy	Precision	Threshold	AUC
22499	1	1.000	1.000	0.5	1.000
False Positive	True Negative	Recall	F1 Score		
5	22495	1.000	1.000		
Positive Label	Negative Label				
1	0				

The Non-Linear Decision boundary

The Non-Linear Decision boundary

- In the above examples we can clearly see the decision boundary is linear
- SVM works well when the data points are linearly separable
- If the decision boundary is non-liner then SVM may struggle to classify
- Observe the below examples, the classes are not linearly separable
- SVM has no direct theory to set the non-liner decision boundary models.





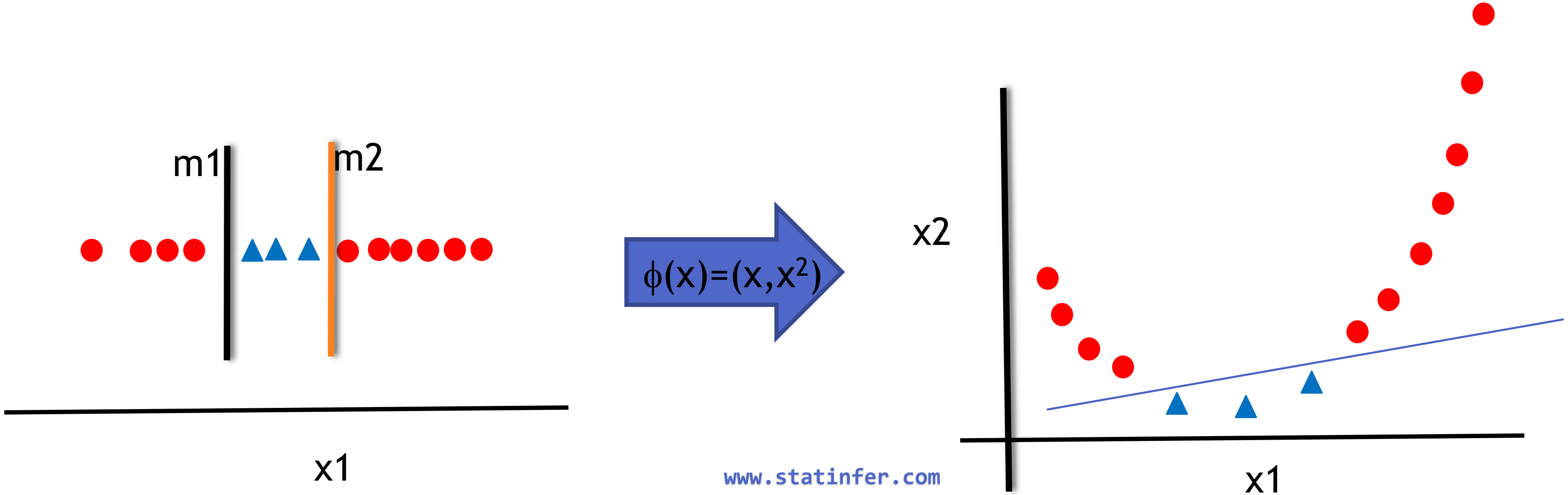
statinfer

Mapping to higher dimensional space

- The original maximum-margin hyperplane algorithm proposed by Vapnik in 1963 constructed a linear classifier.
- To fit a non liner boundary classier, we can create new variables(dimensions) in the data and see whether the decision boundary is linear.
- in 1992, Bernhard E. Boser, Isabelle M. Guyon and Vladimir N. Vapnik suggested a way to create nonlinear classifiers by applying the kernel trick

Mapping to higher dimensional space

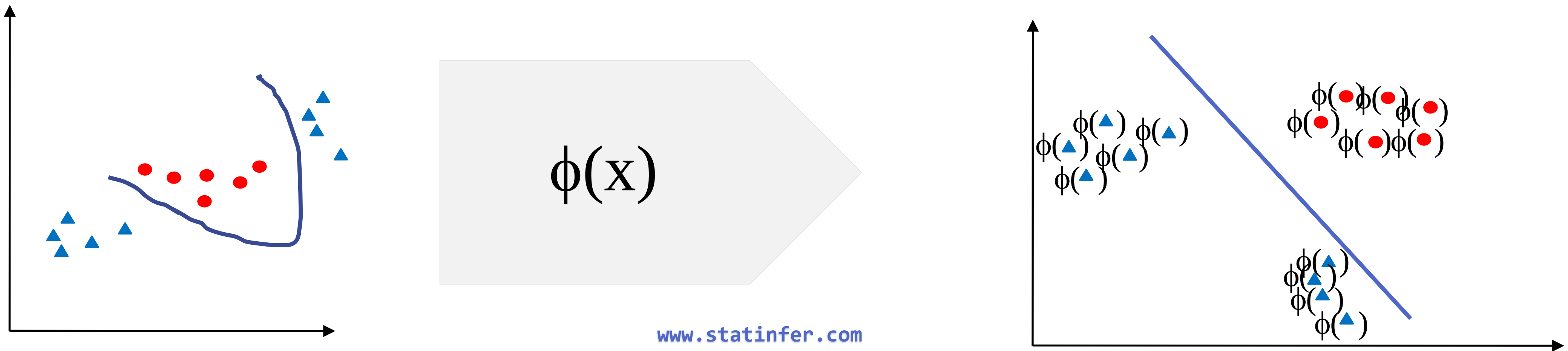
- In the below example, A single linear classifier is not sufficient
- lets create a new variable x2=(x1)^2. In the higher dimensional space
- We can clearly see a possibility of single linear decision boundary
- This is called kernel trick



Kernel Trick

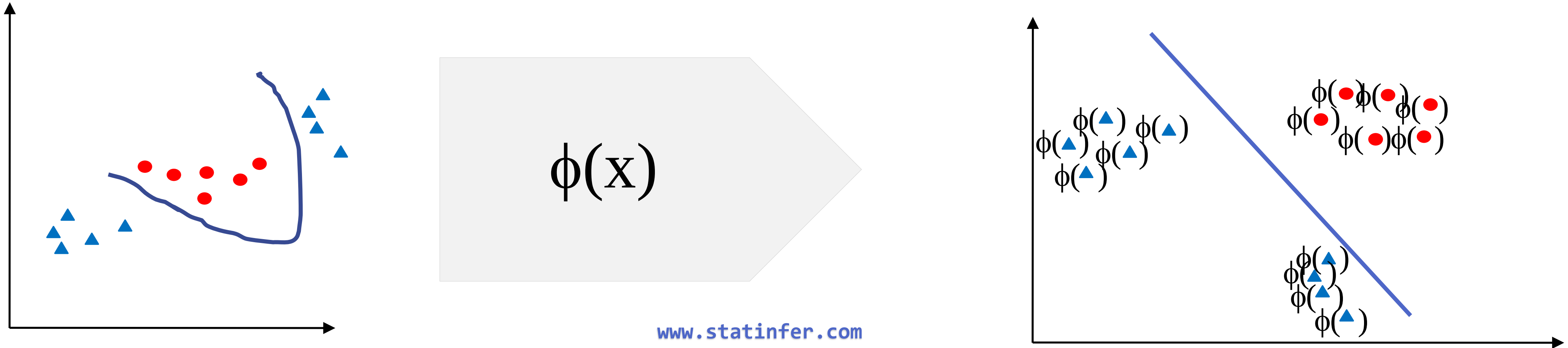
Kernel Trick

- We used a function $\phi(x)=(x,x^2)$ to transform the data x into a higher dimensional space.
- In the higher dimensional space, we could easily fit a liner decision boundary.
- This function $\phi(x)$ is known as kernel function and this process is known as kernel trick in SVM

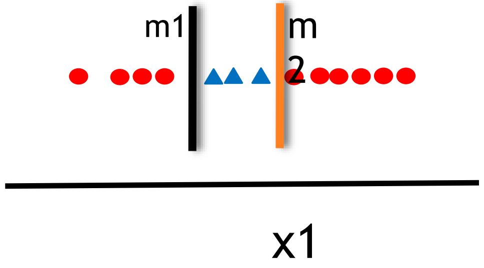


Kernel Trick

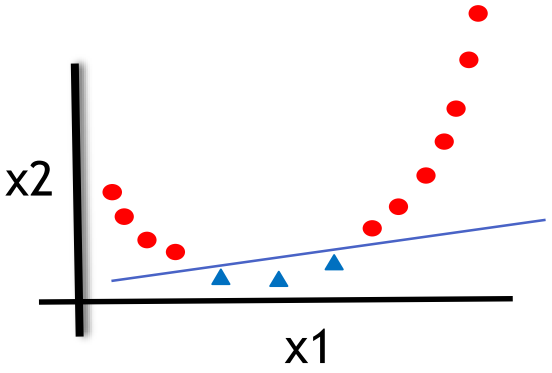
- Kernel trick solves the non-linear decision boundary problem much like the hidden layers in neural networks.
- Kernel trick is simply increasing the number of dimensions. It is to make the non-linear decision boundary in lower dimensional space as a linear decision boundary, in higher dimensional space.
- In simple words, Kernel trick makes the non-linear decision boundary to linear (in higher dimensional space)

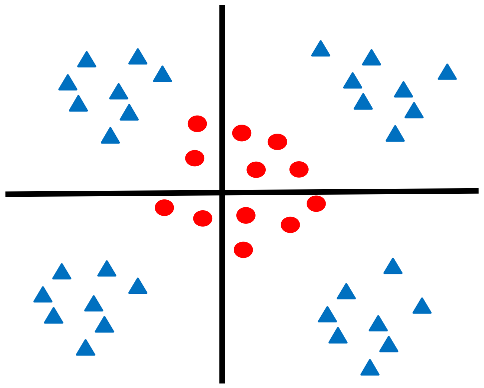


Kernel Trick

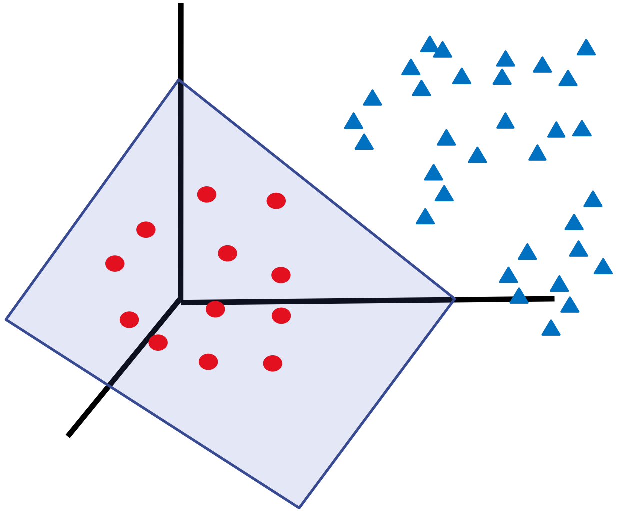


A non-linear decision boundary in single dimensional space is mapped on to a two dimensional space using kernel function $\phi(x)=(x,x^2)$





A non-linear decision boundary in two dimensional space is mapped on to a three dimensional space using kernel function $\phi(x1,x2)=(x_1^2,x_2^2,\sqrt{2}x_1x_2)$



Kernel Function Examples

Name	Function	Type problem
Polynomial Kernel	$(x_i^t x_j + 1)^q$ q is degree of polynomial	Best for Image processing
Sigmoid Kernel	$tanh(a x_i^t x_j + k)$ k is offset value	Very similar to neural network
Gaussian Kernel	$e^{\|\|x_i - x_j\|\|^2 / 2\sigma^2}$	No prior knowledge on data
Linear Kernel	$1 + x_i x_j min(x_i, x_j) - \frac{(x_i + x_j)}{2} min(x_i, x_j)^2 + \frac{min(x_i, x_j)^3}{3}$	Text Classification
Laplace Radial Basis Function (RBF)	$e^{-\gamma \|\|x_i - x_j\|\|}, \gamma \geq 0$	No prior knowledge on data

- There are many more kernel functions.

Choosing the Kernel Function

- Probably the most tricky part of using SVM.
- The kernel function is important because it creates the kernel matrix, which summarizes all the data
- There is no proven theory for choosing a kernel function for any given problem. Still there is lot of research going on.
- In practice, a low degree polynomial kernel or RBF kernel with a reasonable width is a good initial try
- Choosing Kernel function is similar to choosing number of hidden layers in neural networks. Both of them have no proven theory to arrive at a standard value.
- As a first step, we can choose low degree polynomial or radial basis function or one of those from the list

LAB: Kernel – Non linear classifier

LAB: Kernel – Non linear classifier

- Dataset : Software users/sw_user_profile.csv
- How many variables are there in software user profile data?
- Plot the active users against age and check weather the relation between age and “Active” status is linear or non-linear
- Build an SVM model(model-1)
- For model-1, create the confusion matrix and find out the accuracy
- Create a new variable. By using the polynomial kernel
- Build an SVM model(model-2), with the new data mapped on to higher dimensions.
- For model-2, create the confusion matrix and find out the accuracy
- Plot the SVM with results.

Steps - Kernel – Non linear classifier

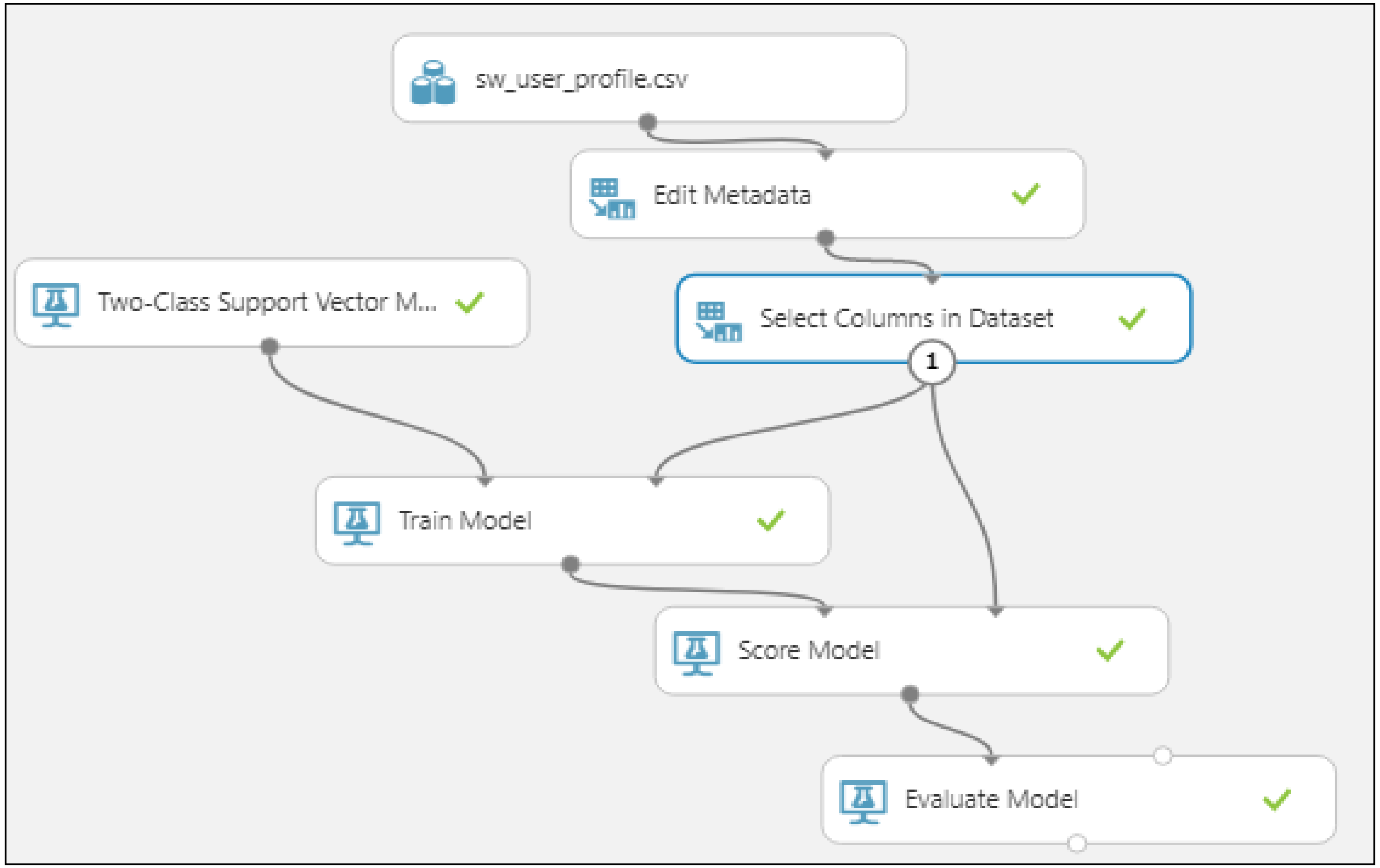
- Drag and drop the **Dataset** into the canvas
- Drag and drop the **Edit Metadata** and connect it to the dataset
- Drag and drop the **Select Columns from the Dataset** and select the columns, connect it to the **Edit Metadata**
- Drag and drop **Two-Class Support Vector Machine, Train Model, Score Model, Enter data Manually** and **Evaluate Model**
- Connect **Two-Class Support Vector Machine** to the first input of **Train Model** and **Select Columns from the Dataset** to the Second input of **Train Model**

Steps - Kernel – Non linear classifier

- Connect the output of **Train Model** first input of **Score Model** and **Enter Data Manually** to the Second input of **Score Model**
- Connect the output of **Score Model** to the input of **Evaluate Model**
- Click on Train Model and select the column for which the prediction is done(Active)
- Click run and visualize the output of **Evaluate Model** and **Score Model**

Steps - Kernel – Non linear classifier

Fig23: Model - 1(without New Column)



Steps - Kernel – Non linear classifier

Fig24: Properties - Edit Metadata

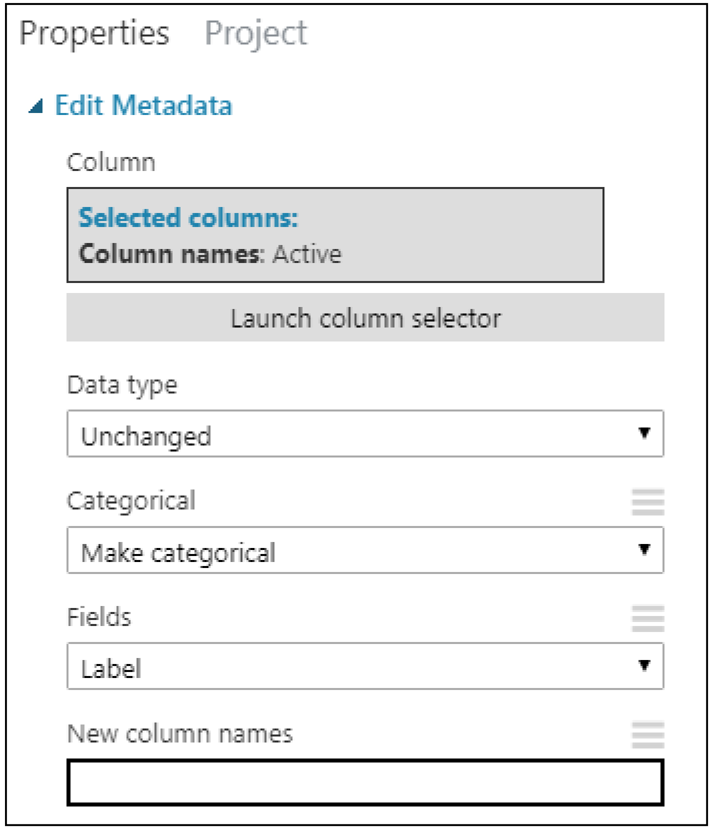
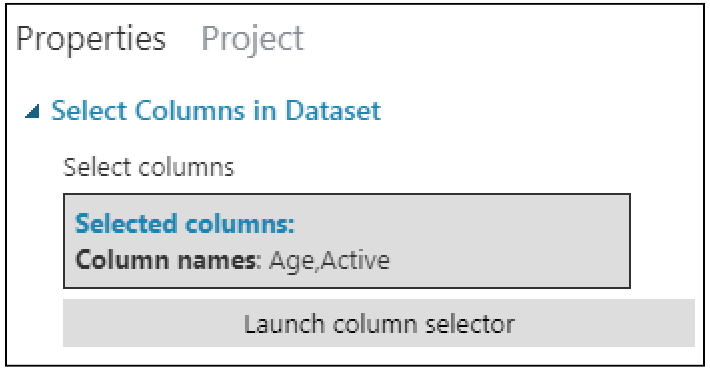


Fig25: Properties - Select Columns from the Dataset



Steps - Kernel – Non linear classifier

Fig26: Properties - Two Class Support Vector Machine

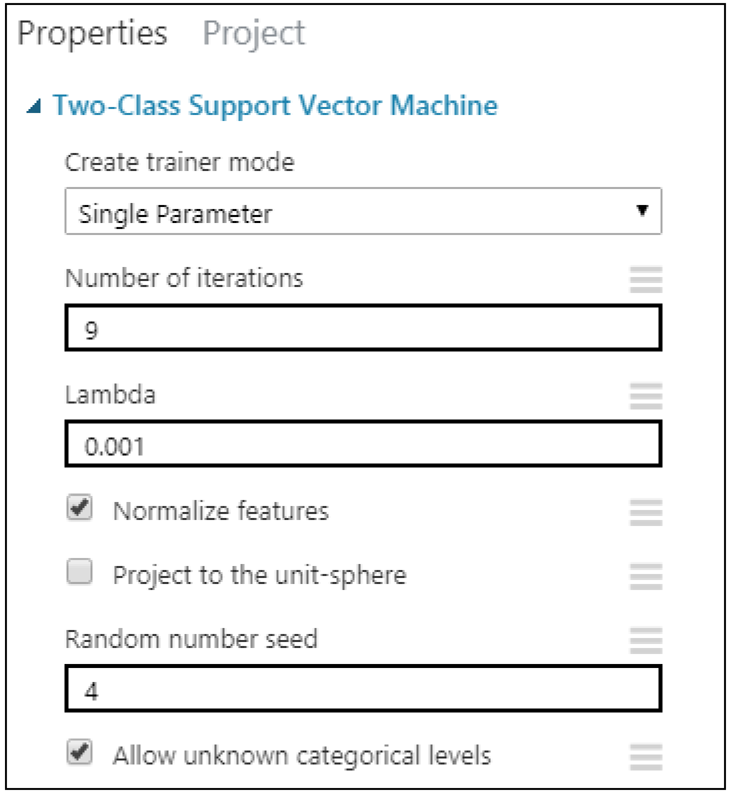
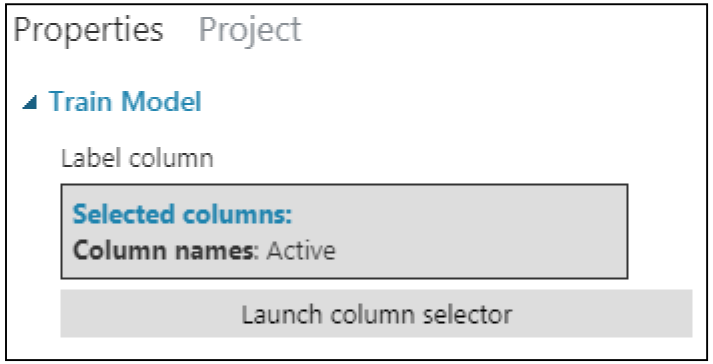


Fig27: Properties - Train



Steps - Kernel – Non linear classifier

Fig28: Scatter Plot – Age vs Active

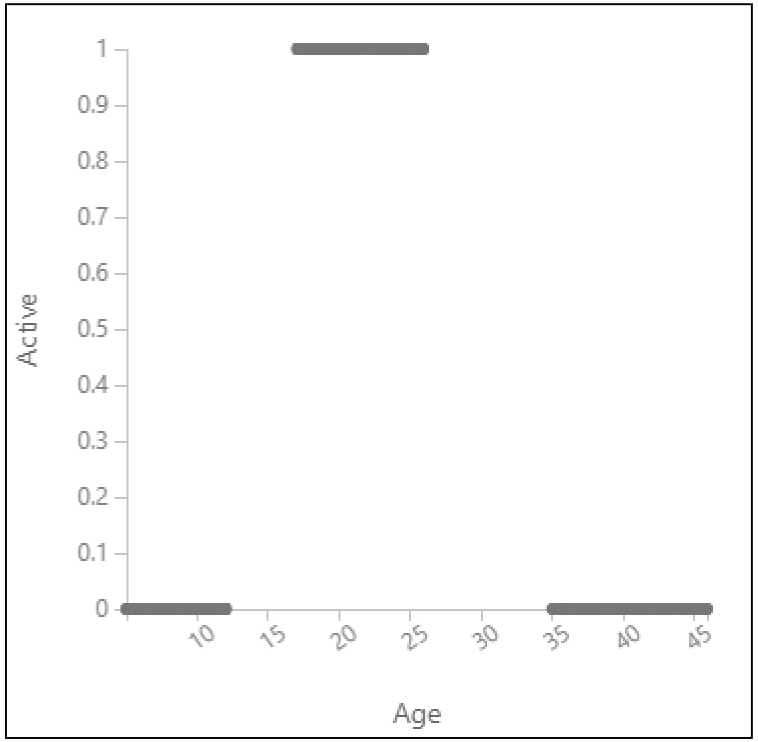


Fig29: Accuracy and Confusion Matrix

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
0	173	0.647	1.000	0.5	0.625
False Positive	True Negative	Recall	F1 Score		
0	317	0.000	0.000		
Positive Label	Negative Label				
1	0				

Steps - Kernel – Non linear classifier

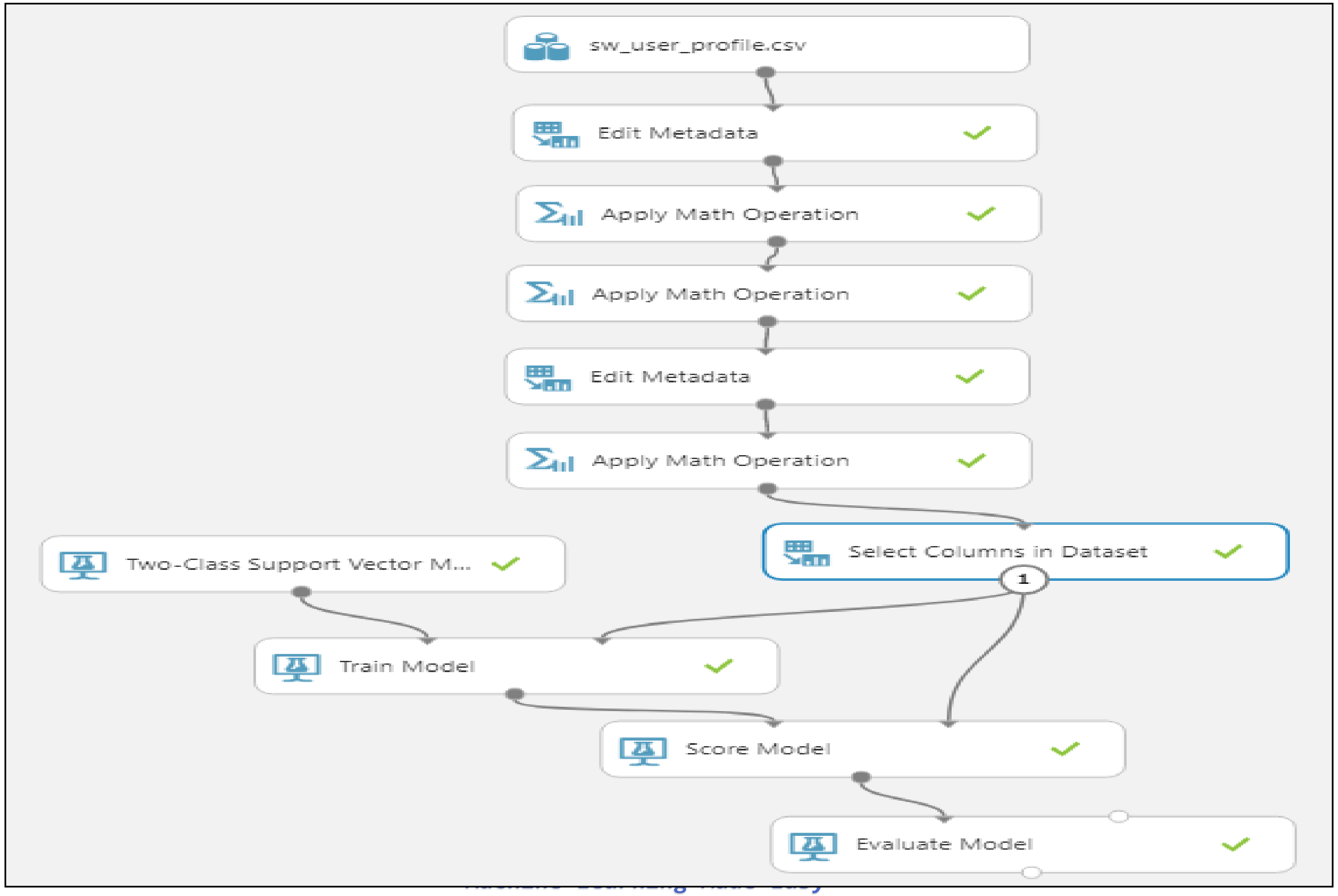
- Drag and drop the **Dataset** into the canvas
- Drag and drop **Edit Metadata**(1) and connect it to the dataset
- Drag and drop **Apply Math Operation**(1), connect it to the **Edit Metadata**(1)
- Drag and drop another **Apply Math Operation**(2), connect it to the previous **Apply Math Operation**(1)
- Drag and drop **Edit Metadata**(2) and connect it to the **Apply Math Operation**(2)
- Drag and drop **Apply Math Operation**(3), connect it to the **Edit Metadata**(2)
- Drag and drop the **Select Columns from the Dataset** and select the columns, connect it to the **Edit Metadata**(2)

Steps - Kernel – Non linear classifier

- Drag and drop **Two-Class Support Vector Machine, Train Model, Score Model, Enter data Manually** and **Evaluate Model**
- Connect **Two-Class Support Vector Machine** to the first input of **Train Model** and **Select Columns from the Dataset** to the Second input of **Train Model**
- Connect the output of **Train Model** first input of **Score Model** and **Enter Data Manually** to the Second input of **Score Model**
- Connect the output of **Score Model** to the input of **Evaluate Model**
- Click on Train Model and select the column for which the prediction is done(Active)
- Click run and visualize the output of **Evaluate Model**

Steps - Kernel – Non linear classifier

Fig30: Model-2(with New Column)



Steps - Kernel – Non linear classifier

Fig31: Properties - Metadata

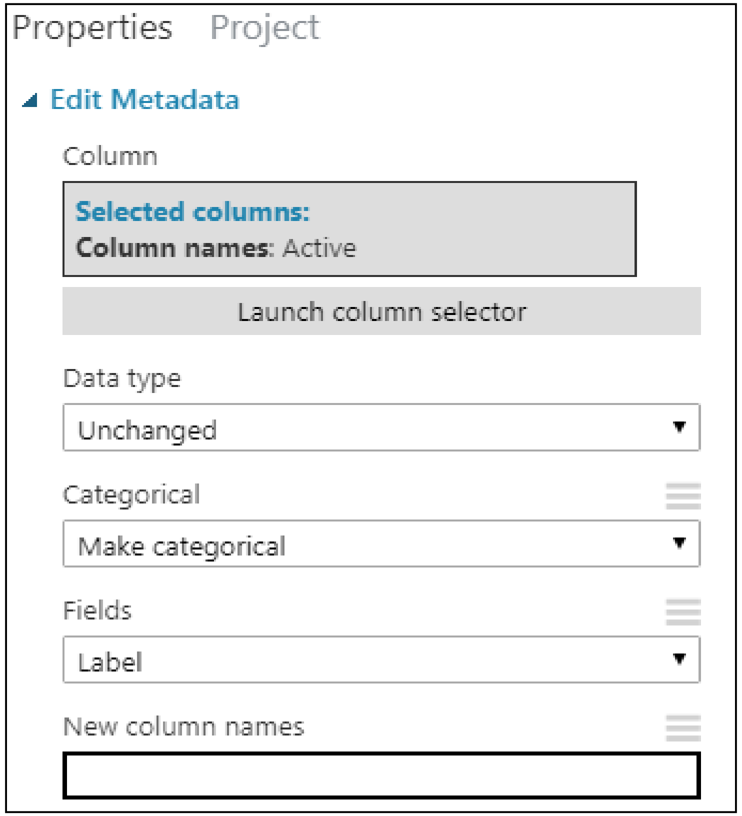
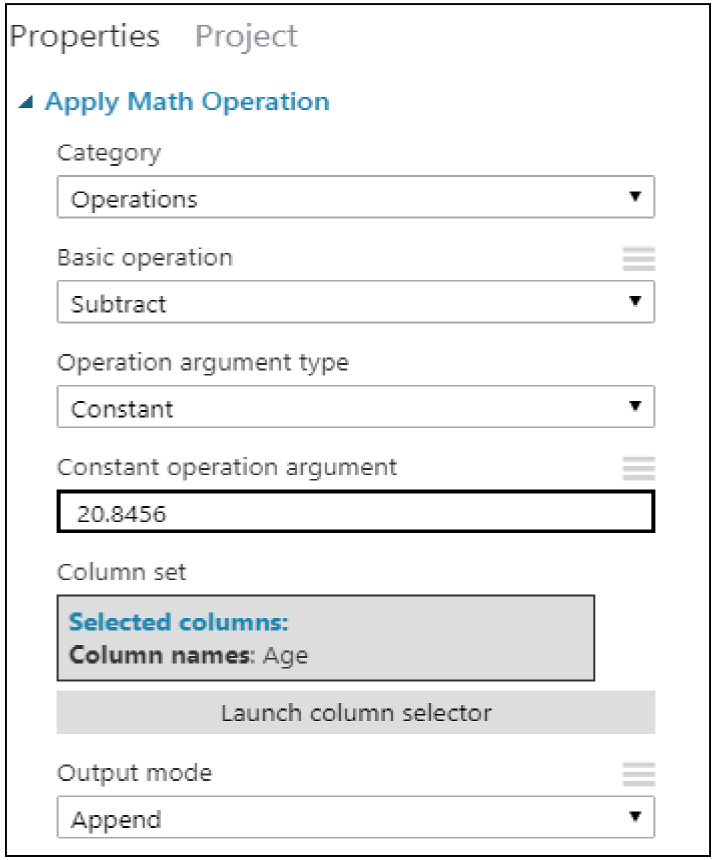


Fig32: Properties - Apply Math Operation1



Steps - Kernel – Non linear classifier

Fig33: Properties - Apply Math Operation2

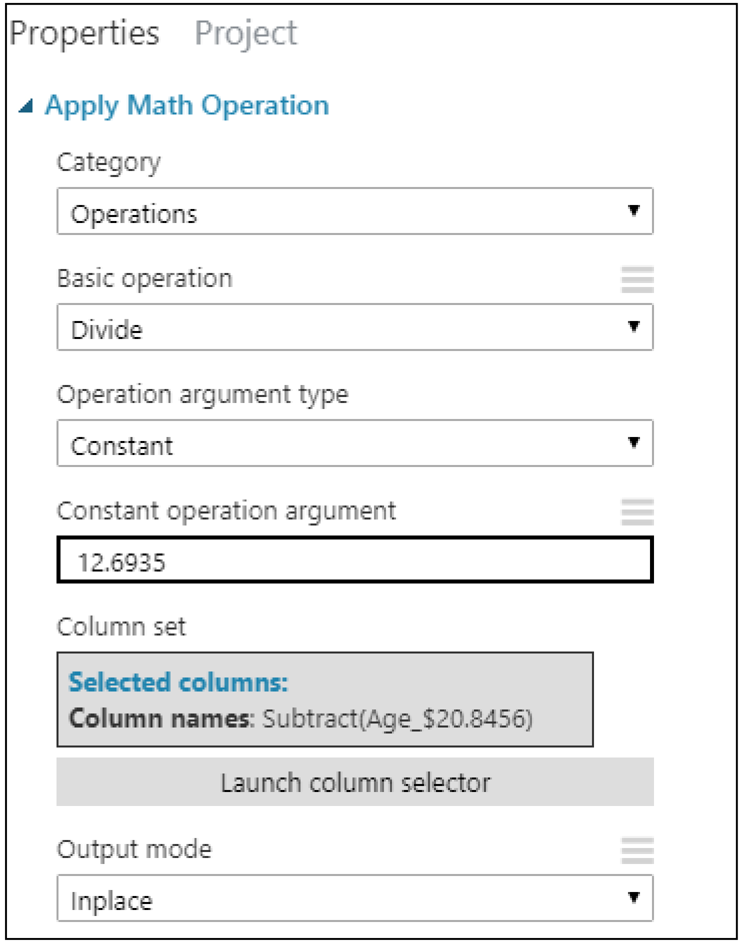
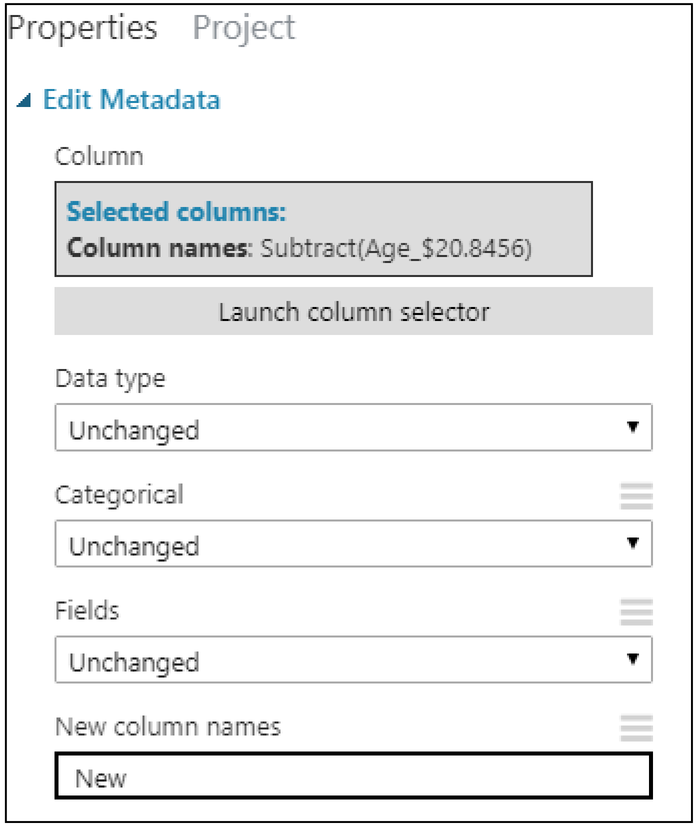


Fig34: Properties - Metadata



Steps - Kernel – Non linear classifier

Fig35: Properties - Apply Math Operation3

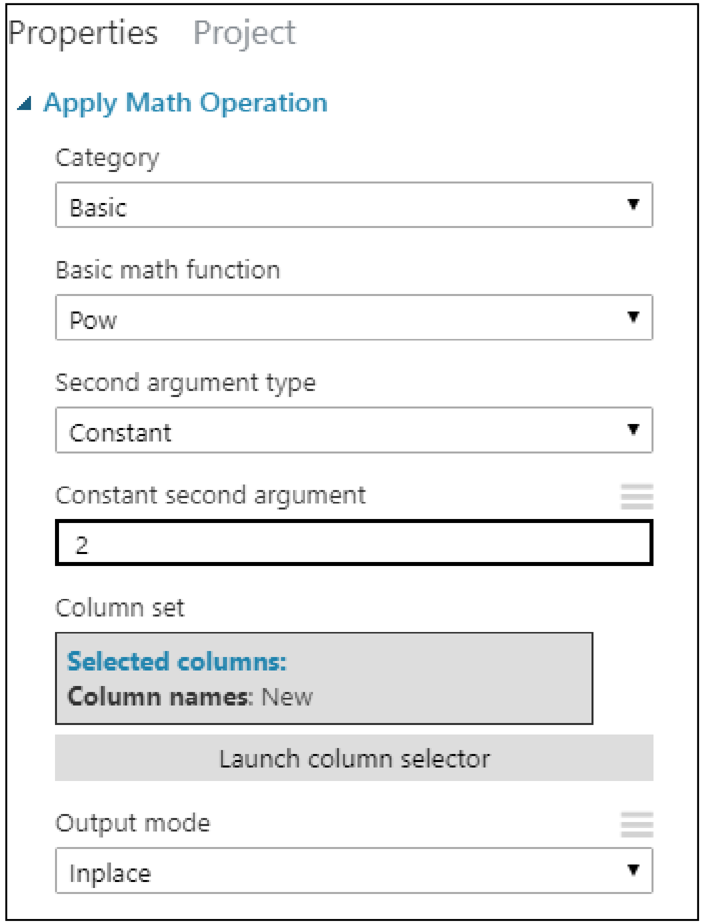
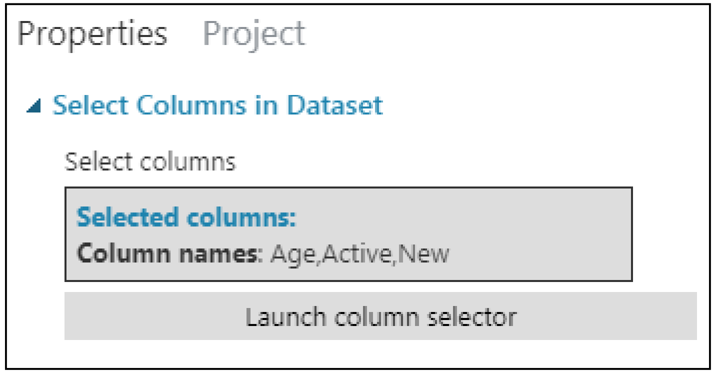


Fig36: Properties - Select Columns



Steps - Kernel – Non linear classifier

Fig37: Properties - Two Class Support Vector Machine

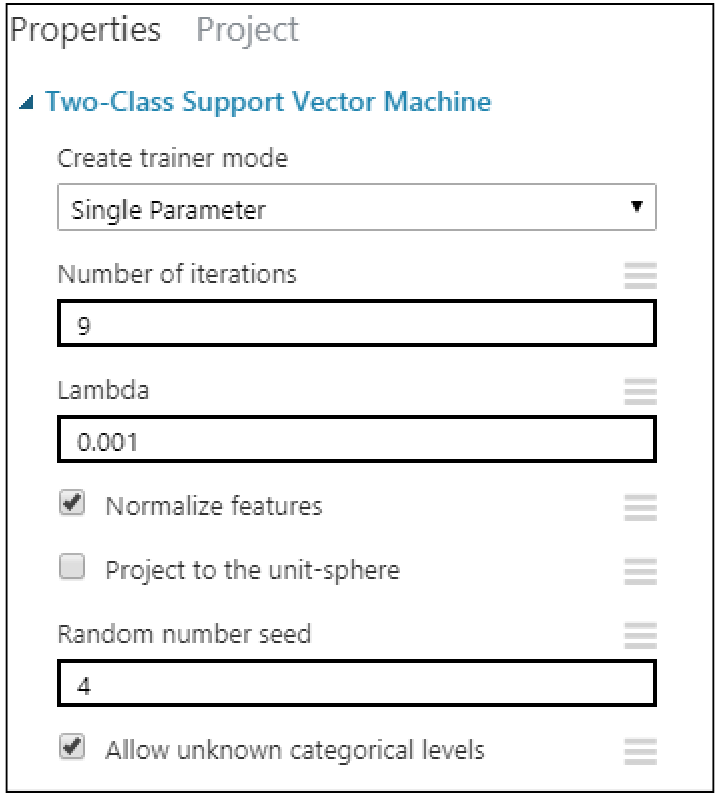
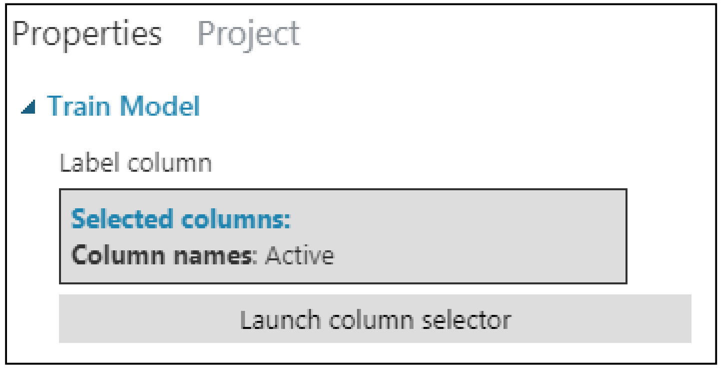


Fig38: Properties - Train Model



Steps - Kernel – Non linear classifier

Fig39: Classifier Data Plot

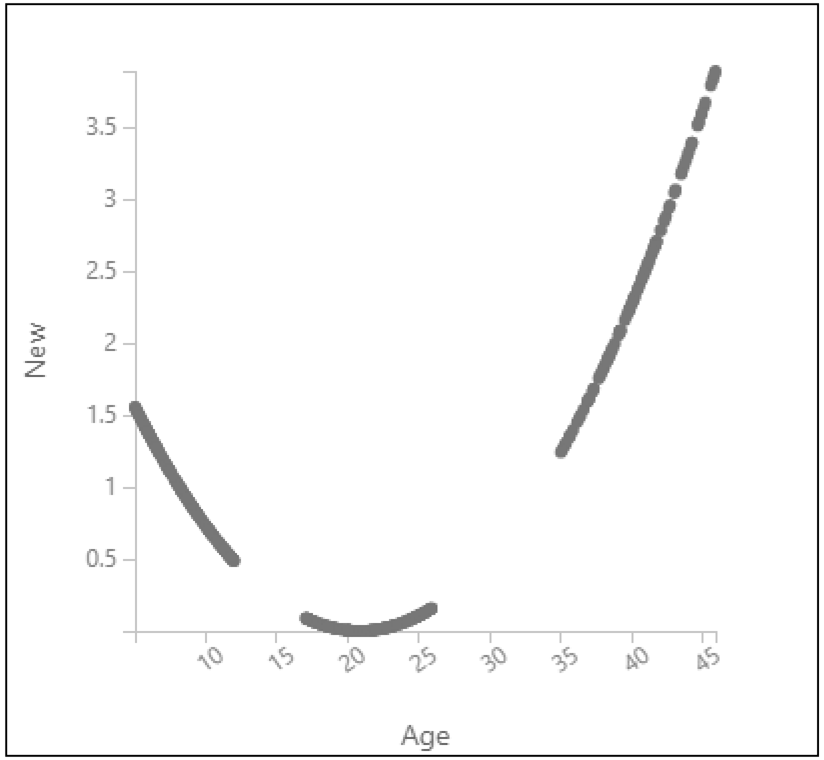


Fig40: Accuracy and Confusion Matrix

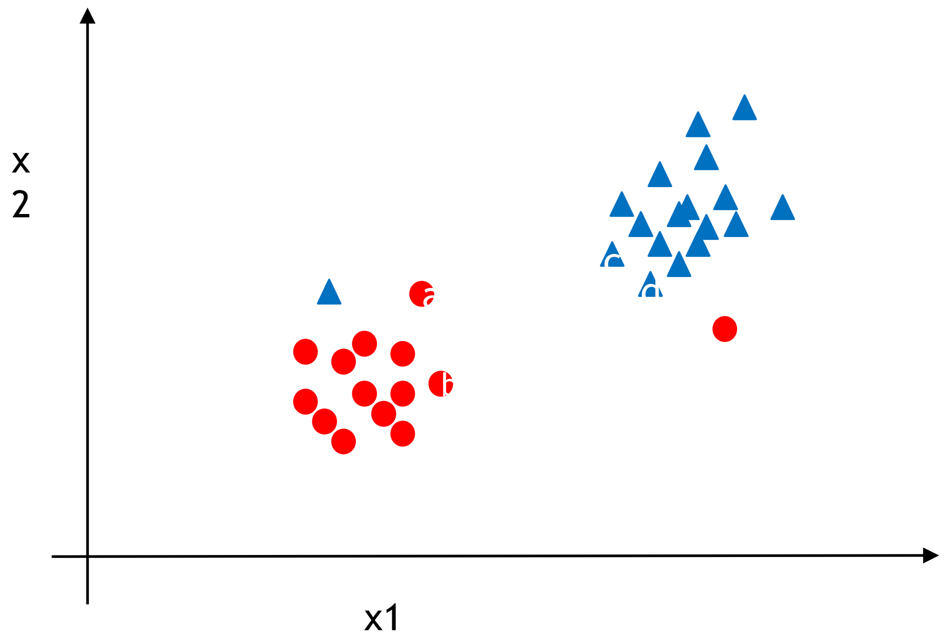
True Positive	False Negative	Accuracy	Precision	Threshold	AUC
173	0	1.000	1.000	0.5	1.000
False Positive	True Negative	Recall	F1 Score		
0	317	1.000	1.000		
Positive Label	Negative Label				
1	0				

statinfer

Soft Margin Classification – Noisy data

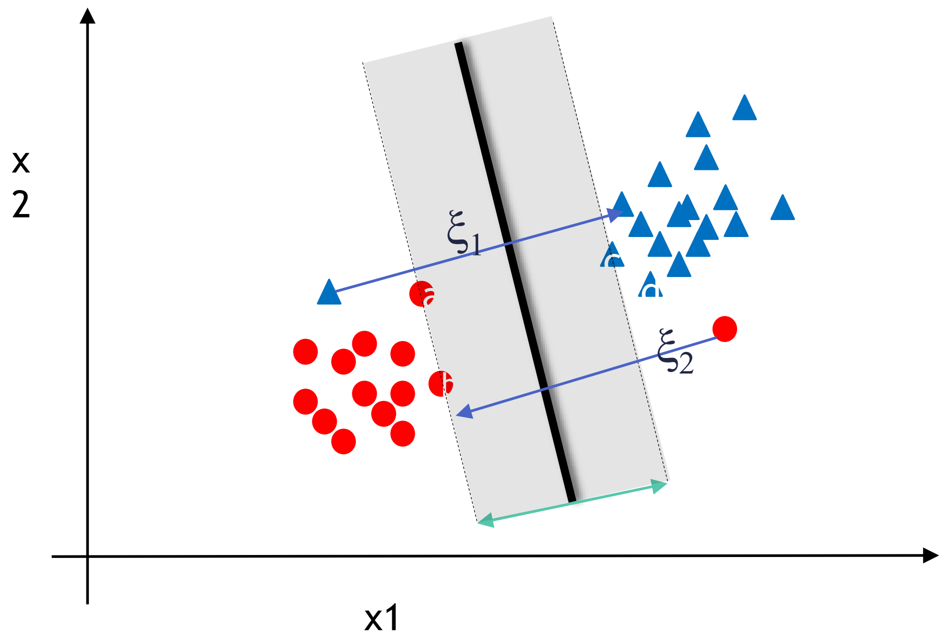
Noisy data

- What if there is some noise in the data.
- What id the overall data can be classified perfectly except few points.
- How to find the hyperplane when few points are on the wrong side.



Soft Margin Classification – Noisy data

- The non-separable cases can be solved by allowing a slack variable(ξ) for the point on the wrong side.
- We are allowing some errors while building the classifier
- In SVM optimization problem we are initially adding some error and then finding the hyperplane
- SVM will find the maximum margin classifier allowing some minimum error due to noise.
- Hard Margin -Classifying all data points correctly,
- Soft margin - Allowing some error



SVM Validation

SVM Validation

- SVM doesn't give us the probability, it directly gives us the resultant classes
- Usual methods of validation like sensitivity, specificity, cross validation, ROC and AUC will are the validation methods

statinfer

SVM Advantages & Disadvantages

SVM Advantages

- SVM's are very good when we have no idea on the data
- Works well with even unstructured and semi structured data like text, Images and trees.
- The kernel trick is real strength of SVM. With an appropriate kernel function, we can solve any complex problem
- Unlike in neural networks, SVM is not solved for local optima.
- It scales relatively well to high dimensional data
- SVM models have generalization in practice, the risk of overfitting is less in SVM.

SVM Disadvantages

- Choosing a "good" kernel function is not easy.
- long training time o large datasets
- Difficult to understand and interpret the final model, variable weights and individual impact
- Since the final model is not so easy to see, we can not do small calibrations to the model hence its tough to incorporate our business logic

SVM Application

SVM Application

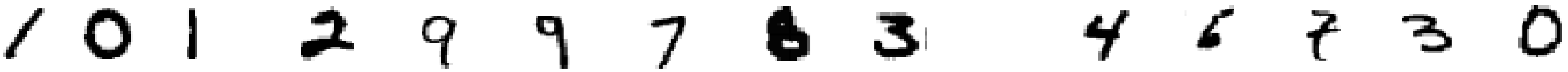
- Protein Structure Prediction
- Intrusion Detection
- Handwriting Recognition
- Detecting Steganography in digital images
- Breast Cancer Diagnosis

statinfer

LAB: Digit Recognition using SVM

LAB: Digit Recognition using SVM

- Take an image of a handwritten single digit, and determine what that digit is.
- Normalized handwritten digits, automatically scanned from envelopes by the U.S. Postal Service. The original scanned digits are binary and of different sizes and orientations; the images here have been de slanted and size normalized, resultingin 16 x 16 grayscale images (Le Cun et al., 1990).
- The data are in two gzipped files, and each line consists of the digitid (0-9) followed by the 256 grayscale values.
- Build an SVM model that can be used as the digit recognizer
- Use the test dataset to validate the true classification power of the model
- What is the final accuracy of the model?

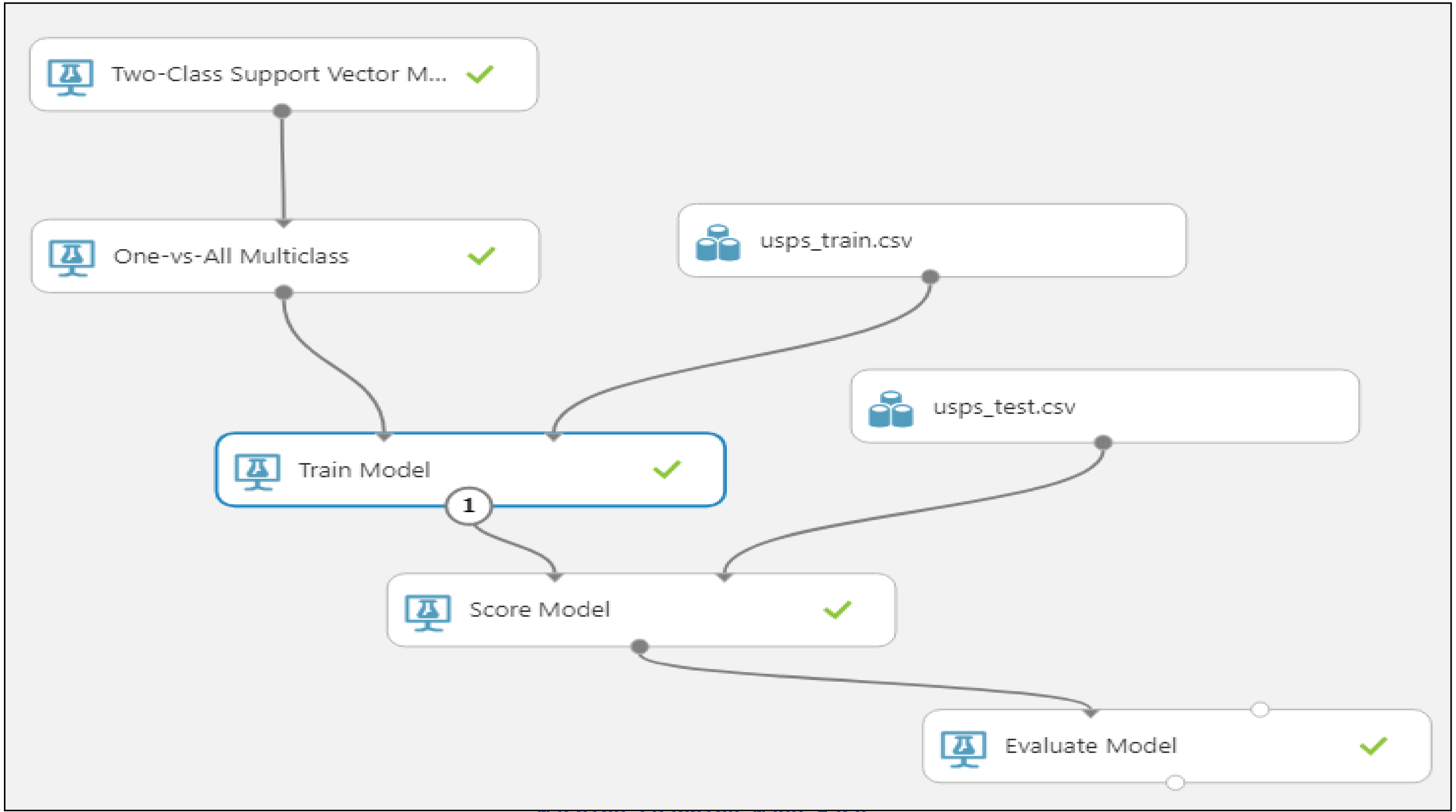


Steps - Digit Recognition using SVM

- Drag and drop the **Training and Test Dataset** into the canvas
- Drag and drop **Two-Class Support Vector Machine, One-vs-All Multiclass, Train Model, Score Model, Enter data Manually** and **Evaluate Model**
- **Connect Two-Class Support Vector Machine** to **One-vs-All Multiclass**
- Connect **One-vs-All Multiclass** to the first input of **Train Model** and **Select Columns from the Dataset** to the Second input of **Train Model**
- Connect the output of **Train Model** first input of **Score Model** and **Enter Data Manually** to the Second input of **Score Model**
- Connect the output of **Score Model** to the input of **Evaluate Model**
- Click on Train Model and select the column for which the prediction is done(Active)
- Click run and visualize the output of **Evaluate Model** and **Score Model**

Steps - Digit Recognition using SVM

Fig41: Digit Recognition – Support Vector Machine



Steps - Digit Recognition using SVM

Fig42: Properties - Support Vector Machine

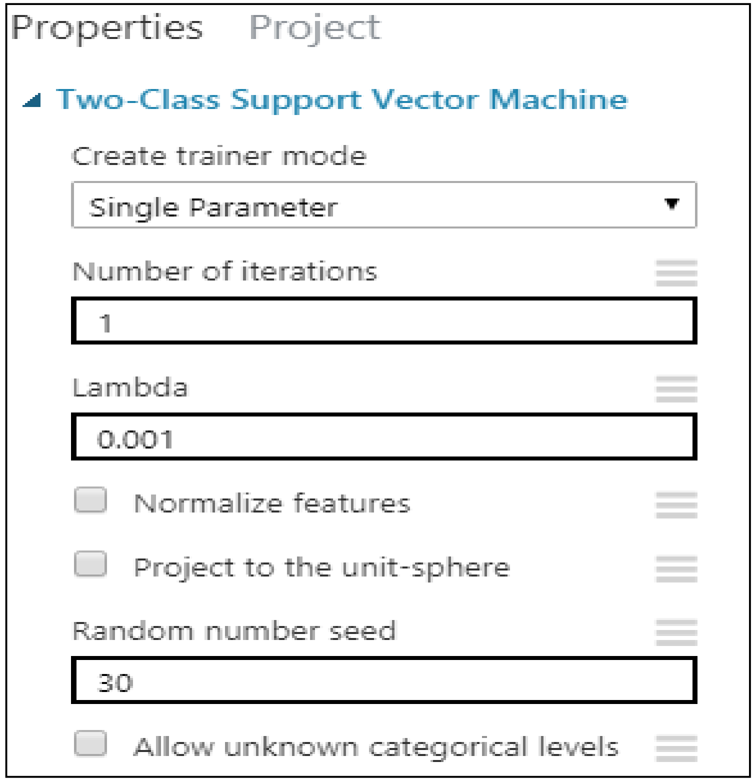
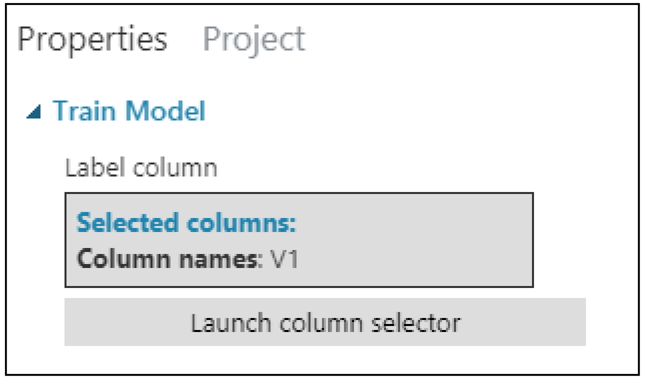


Fig43: Properties - Train Model



Steps - Digit Recognition using SVM

Fig44: Accuracy

Metrics

Overall accuracy	0.897359
Average accuracy	0.979472
Micro-averaged precision	0.897359
Macro-averaged precision	0.888338
Micro-averaged recall	0.897359
Macro-averaged recall	0.886581

Steps - Digit Recognition using SVM

Fig45: Confusion Matrix

Actual Class \ Predicted Class	0	1	2	3	4	5	6	7	8	9
0	96.4%			0.8%	0.8%	0.3%	1.1%		0.3%	0.3%
1		96.6%		1.1%	0.8%		0.8%		0.4%	0.4%
2	2.0%		82.3%	4.5%	3.5%	1.0%	2.0%	1.5%	3.0%	
3	1.8%		1.8%	80.7%	0.6%	10.8%		1.2%	1.2%	1.8%
4	1.0%	1.0%	2.5%		89.0%	0.5%	1.0%	1.0%	1.0%	3.0%
5	2.5%			5.0%	1.3%	85.0%			3.1%	3.1%
6	1.2%		1.8%		1.2%	1.2%	92.9%		1.8%	
7	0.7%		0.7%	1.4%	3.4%			89.1%	0.7%	4.1%
8	3.6%		1.2%	5.4%	2.4%	3.6%		1.2%	80.7%	1.8%
9		0.6%	1.1%		1.1%	0.6%		1.7%	1.1%	93.8%

Conclusion

Conclusion

- Many software tools are available for SVM implementation
- SVMs are really good for text classification
- SVMs are good at finding the best linear separator. The kernel trick makes SVMs non-linear learning algorithms
- Choosing an appropriate kernel is the key for good SVM and choosing the right kernel function is not easy
- We need to be patient while building SVMs on large datasets. They take a lot of time for training.

Thank you

Our e-Learning Modules

www.statinfer.com

statinfer

Part 11/12 - Random Forests & Boosting

Venkat Reddy

Contents

Contents

- Introduction
- Ensemble Learning
- How ensemble learning works
- Bagging
- Building models using Bagging
- Random Forest algorithm
- Random Forest model building

The Wisdom of Crowds

The wisdom of crowds

Problem Statement: What is the estimated monthly expense of a family in our city.

<u>An Eminent Professor built a model</u> Vs. <u>100 Assistant Professors built 100 models</u>

M1

One Single Prediction

$6500

M1 M2 M3 ...... M100

Average of all 100 predictions

$7200

The wisdom of crowds

- One should not expend energy trying to identify an expert within a group but instead rely on the group's collective wisdom, however make sure that Opinions must be independent and some knowledge of the truth must reside with some group members – Surowiecki

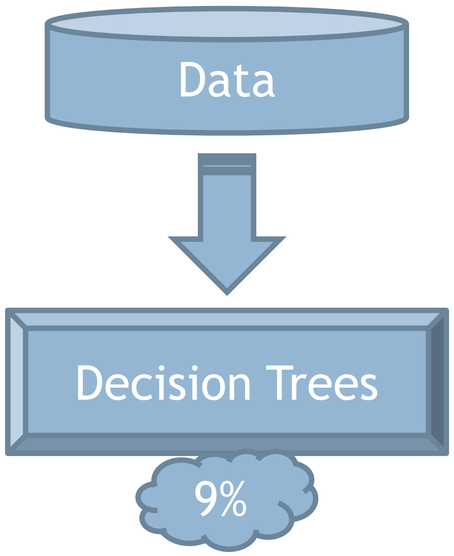
- So instead of trying to build one great model, its better to build some independent moderate models and take their average as final prediction

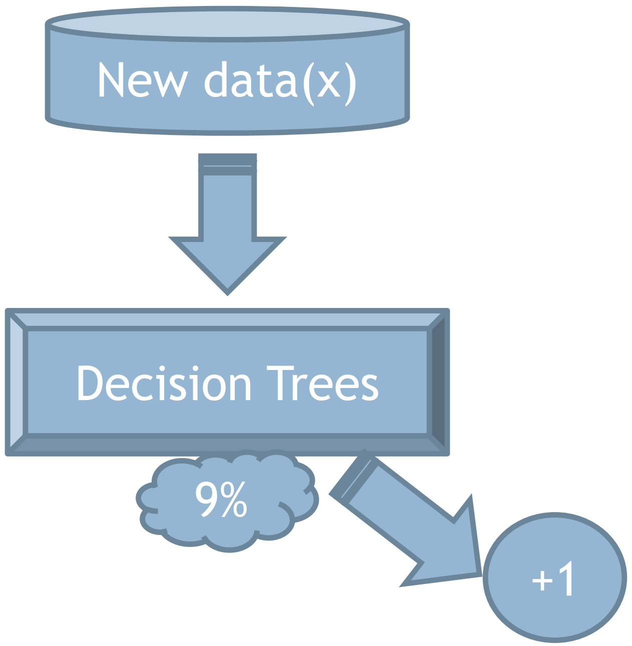
statinfer

What is Ensemble Learning

What is Ensemble Learning

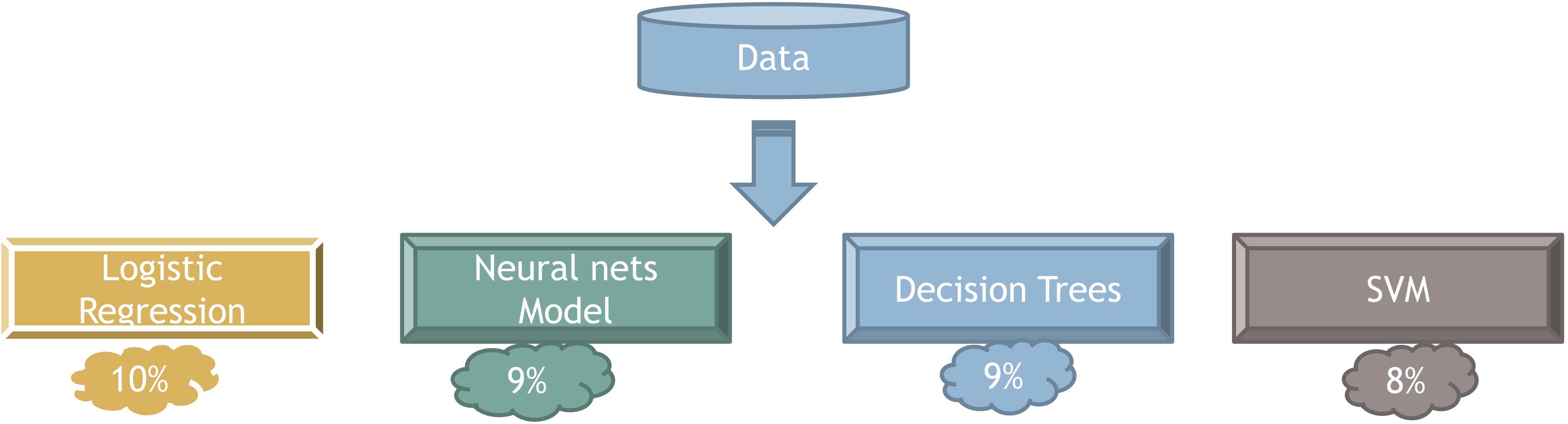
- Imagine a classifier problem, there are two classes +1 & -1 in the target
- Imagine that we built a best possible decision tree, it has 91% accuracy
- Let x be the new data point and our decision tree predicts it to be +1. Is there a way we can do better than 91% by using the same data
- Lets build 3 more models on the same data. And see we can improve the performance





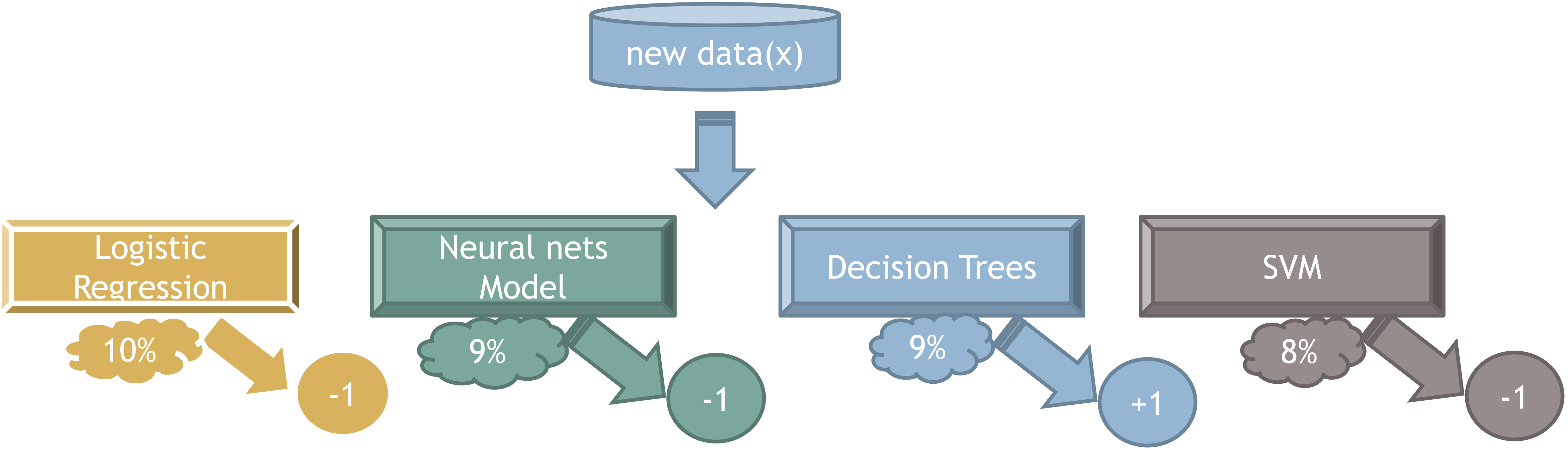
What is Ensemble Learning

- We have four models on the same dataset, Each of them have different accuracy. But unfortunately there seem to be no real improvement in the accuracy.



What is Ensemble Learning

- What about prediction of the data point x?
- Except the decision tree, the rest all algorithms are predicting the class of x as -1
- Intuitively we would like to believe that the class of x is -1
- The combined voting model seem to be having less error than each of the individual models.
- This is the actual philosophy of ensemble learning

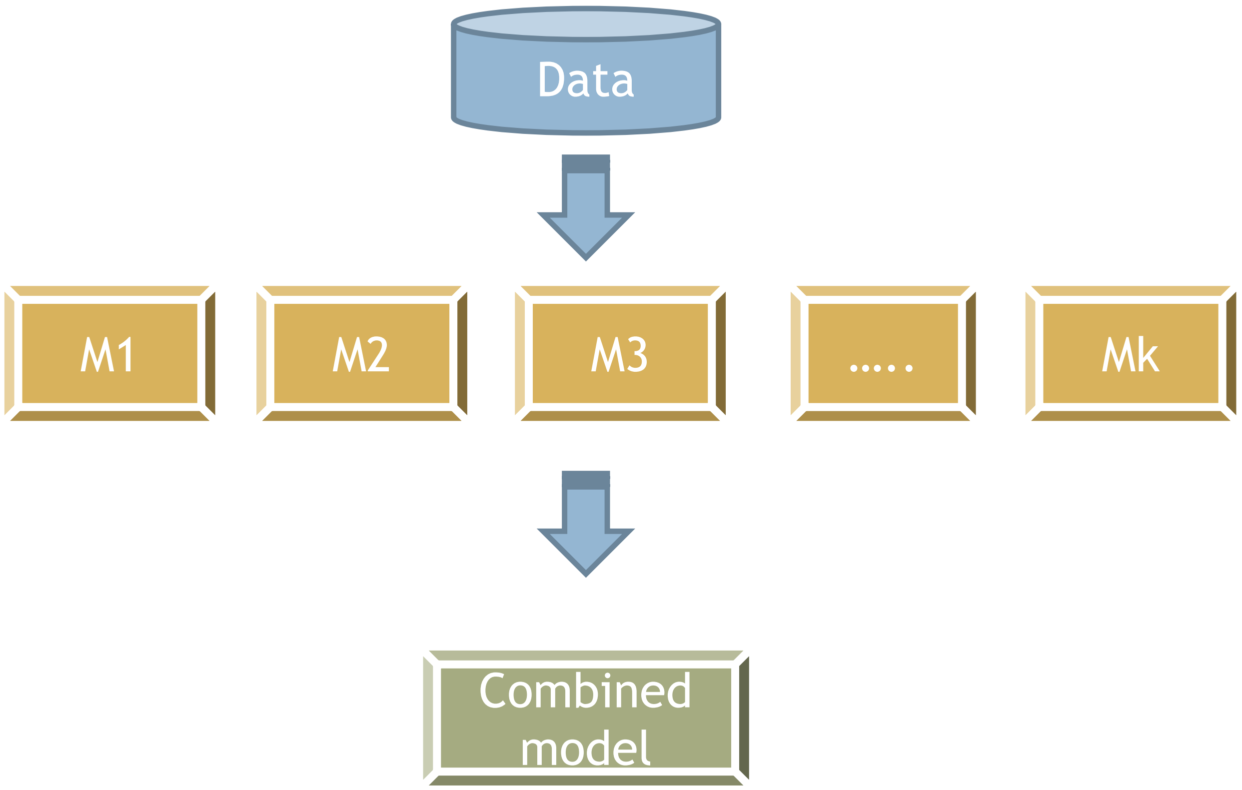


statinfer

Ensemble Models

Ensemble Models

- Obtaining a better predictions using multiple models on the same dataset
- Not every time it is possible to find single best fit model for our data, ensemble model combines multiple models to come up with one consolidated model
- Ensemble models work on the principle that multiple moderately accurate models can give us a highly accurate model
- Understandably, the Building and Evaluating the ensemble models is computationally expensive
- Build one really good model is the usual statistical approach. Build many models and average the results is the philosophy of Ensemble learning



Why Ensemble technique works?

- Imagine three models
  - M1 with an error rate of 10%
  - M2 with an error rate of 10%
  - M3 with an error rate of 10%
- The three models have to be independent, we can't build the same model three times and expect the error to reduce. Any changes to the modeling technique in model -1 should not impact model-2
- In this scenario, the worst ensemble model will have 10% error rate
- The best ensemble model will have an error rate of 2.8%
  - 2 out of 3 models predicted wrong + all models predicted wrong
  - (3C2)*(0.1)(0.1)(0.9) + (0.1)(0.1)(0.1)
  - 2.8%
- The best ensemble model will have an error rate of 2.8%

Types of Ensemble Models

- The above example is a very primitive type of ensemble model. There are better and statistically stronger ensemble methods that will yield better results
- Two most popular ensemble methodologies are
  - Bagging
  - Boosting

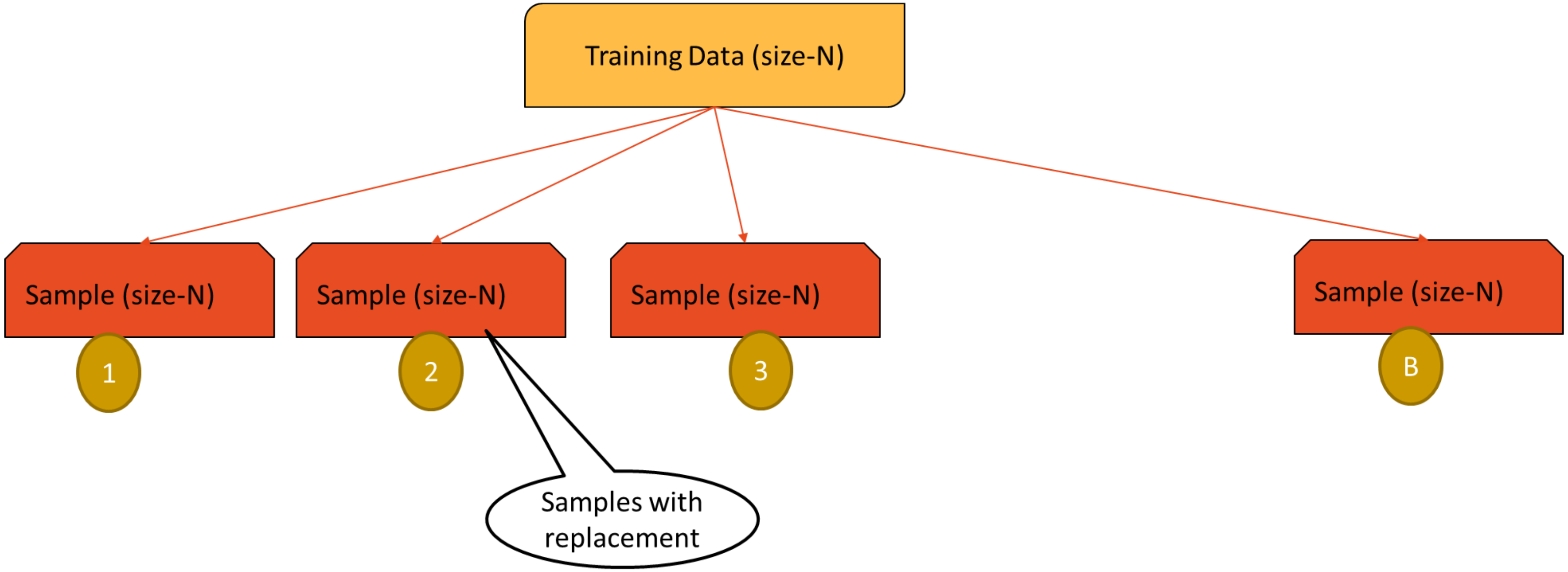
Bagging

Bagging

- Take multiple boot strap samples from the population and build classifiers on each of the samples. For prediction take mean or mode of all the individual model predictions.
- Bagging has two major parts 1) Boot strap sampling 2) Aggregation of learners
- **Bagging** = Bootstrap **Agg**regat**ing**
- In Bagging we combine many unstable models to produce a stable model. Hence the predictors will be very reliable(less variance in the final model).

Boot strapping

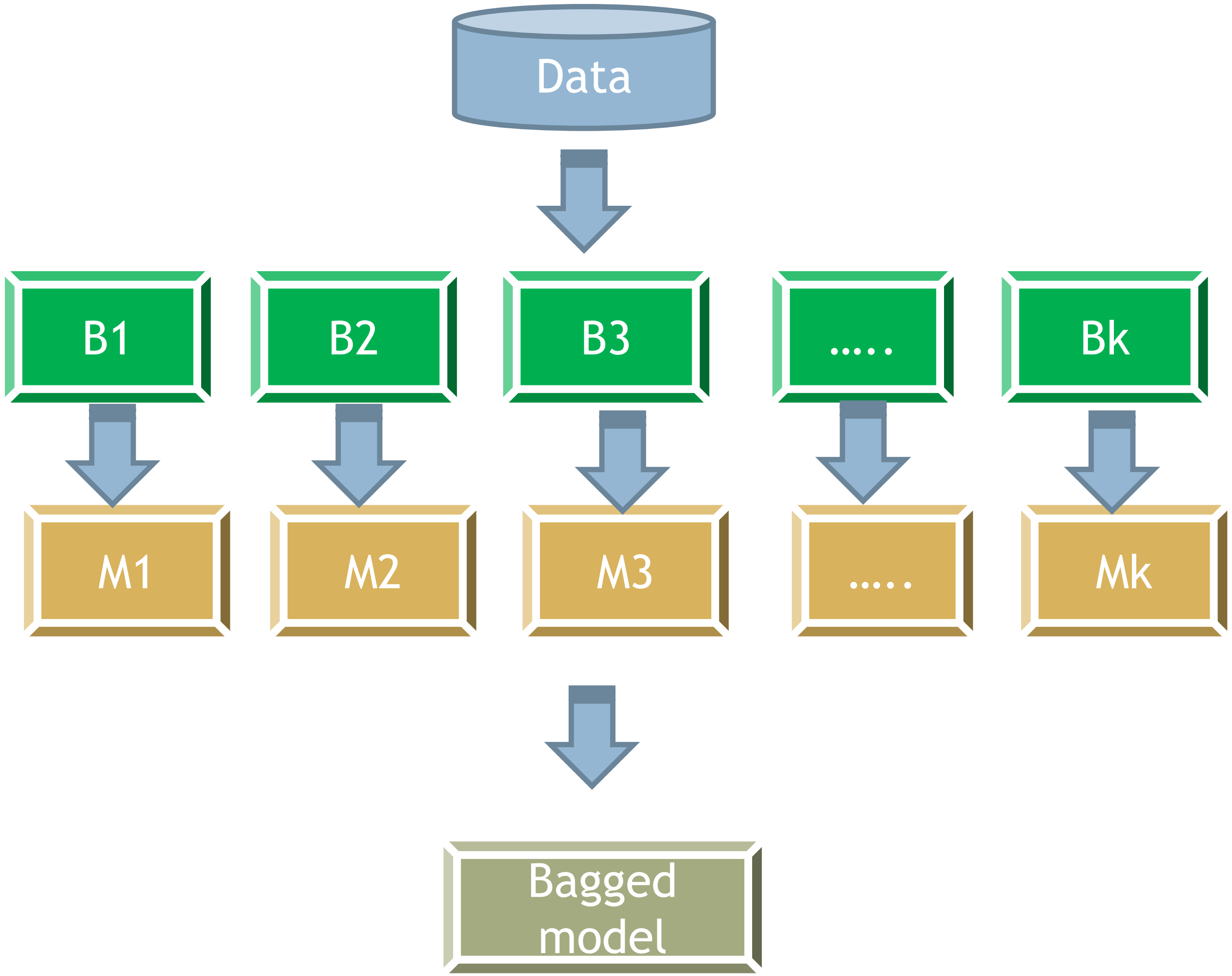
- We have a training data is of size N
- Draw random sample with replacement of size N – This gives a new dataset, it might have repeated observations, some observations might not have even appeared once.
- We are selecting records one-at-a-time, returning each selected record back in the population, giving it a chance to be selected again
- Create B such new datasets. These are called boot strap datasets



The Bagging Algorithm

The Bagging Algorithm

- The training dataset D
- Draw k boot strap sample sets from dataset D
- For each boot strap sample i
  - Build a classifier model M_i
  - We will have total of k classifiers M_1 , M_2 ,….. M_k
  - Vote over for the final classifier output and take the average for regression output



Why Bagging works

- We are selecting records one-at-a-time, returning each selected record back in the population, giving it a chance to be selected again
- Note that the variance in the consolidated prediction is reduced, if we have independent samples. That way we can reduce the unavoidable errors made by the single model.
- In a given boot strap sample, some observations have chance to select multiple times and some observations might not have selected at all.
- There a proven theory that boot strap samples have only 63% of overall population and rest 37% is not present.
- So the data used in each of these models is not exactly same, This makes our learning models independent. This helps our predictors have the uncorrelated errors.
- Finally the errors from the individual models cancel out and give us a better ensemble model with higher accuracy
- Bagging is really useful when there is lot of variance in our data

statinfer

LAB: Bagging Models

LAB: Bagging Models

- Import Boston house price data. It is part of MASS package
- Get some basic meta details of the data
- Take 80% data use it for training and take rest 20% as holdout data
- Build a single linear regression model on the training data. Build the model for medv vs rest of the variables
- On the hold out data, calculate the error (squared deviation) for the regression model.
- Build the regression model using bagging technique. Build at least 25 models
- On the hold out data, calculate the error (squared deviation) for the consolidated bagged regression model.
- What is the improvement of the bagged model when compared with the single model?

Steps - Bagging Models

- Drag and drop the Dataset into the canvas
- Drag and drop the Split Data and connect it to the dataset
- In Split Data properties, select
  - Splitting mode → Split Rows
  - Fraction of rows in the first output dataset → 0.8
  - Check the Randomized split option
- Drag and drop **Linear Regression, Train Model, Score Model** and **Evaluate Model**
- Connect **Linear Regression** to the first input of **Train Model** and first output of **Split Data** to the Second input of **Train Model**
- Connect the output of **Train Model** first input of **Score Model** and second output of **Split Data** to the Second input of **Score Model**
- Connect the output of **Score Model** to the input of **Evaluate Model**

Steps - Bagging Models cont..

- Click on **Train Model** and select the column for which the prediction is done
- Click run and visualize the output of **Evaluate Model**
- Drag and drop **Apply Math Operation(LR-first),** connect the output of **Score Model** to it.
- Select **Apply Math Operation(LR-first)**, click on **Run Selected**
- Drag and drop **Apply Math Operation(LR-second),** connect the output of **Apply Math Operation(LR-first)** to it.
- Select **Apply Math Operation(LR-second)**, click on **Run Selected**
- Drag and drop **Compute Elementary Statistics(LR),** connect the output of **Apply Math Operation(LR-second)** to it.
- Select **Compute Elementary Statistics(LR)**, click on **Run Selected**
- **Note:** Select the properties of **Apply Math Operation(LR-first)**, **Apply Math Operation(LR-second)** and **Compute Elementary Statistics(LR)** before run

Steps - Bagging Models cont..

- Similarly do the same for the **Random Forest Regression** as follows:
- Drag and drop **Decision Forest Regression, Train Model, Score Model** and **Evaluate Model**
- Connect **Decision Forest Regression** to the first input of **Train Model** and first output of **Split Data** to the Second input of **Train Model**
- Connect the output of **Train Model** first input of **Score Model** and second output of **Split Data** to the Second input of **Score Model**
- Connect the output of **Score Model** to the input of **Evaluate Model**
- Click on **Train Model** and select the column for which the prediction is done
- Click run and visualize the output of **Evaluate Model**

Steps - Bagging Models cont..

- Drag and drop **Apply Math Operation(DF-first),** connect the output of **Score Model** to it.
- Select **Apply Math Operation(DF-first),** click on **Run Selected**
- Drag and drop **Apply Math Operation(DF-second),** connect the output of **Apply Math Operation(DF-first)** to it.
- Select **Apply Math Operation(DF-second),** click on **Run Selected**
- Drag and drop **Compute Elementary Statistics(DF),** connect the output of **Apply Math Operation(DF-second)** to it.
- Select **Compute Elementary Statistics(DF),** click on **Run Selected**
- **Note:** Select the properties of **Apply Math Operation(DF-first)**, **Apply Math Operation(DF-second)** and **Compute Elementary Statistics(DF)** before run

Steps - Bagging Models cont..

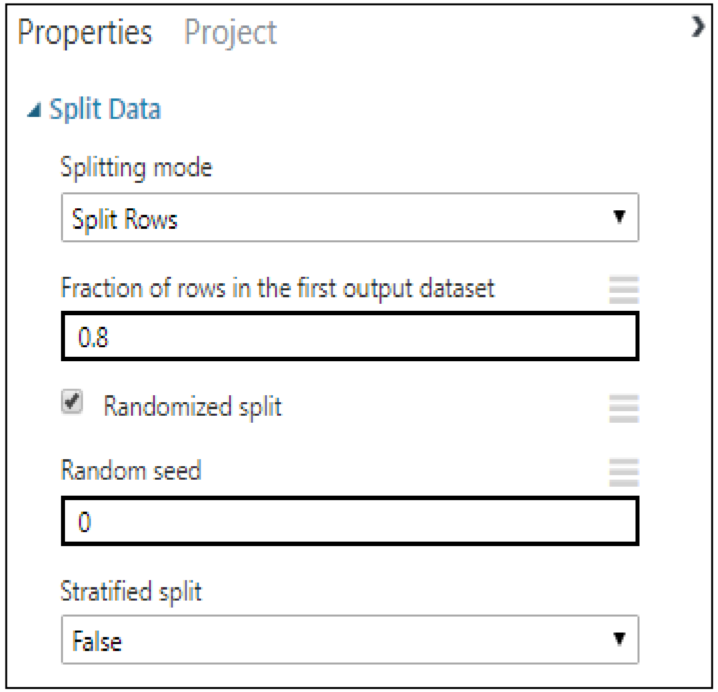
- To compute **Improvement Percentage:**
- Drag and drop **Add Columns** into the canvas
- Connect **Compute Elementary Statistics(LR)** to the first input of the **Add Columns** and **Compute Elementary Statistics(DF)** to the second input of the **Add Columns**
- Drag and drop **Apply Math Operation(IP-first),** connect the output of **Add Columns** to it.
- Select **Apply Math Operation(IP-first)**, click on **Run Selected**
- Drag and drop **Apply Math Operation(IP-second),** connect the output of **Apply Math Operation(IP-first)** to it.
- Select **Apply Math Operation(IP-second)**, click on **Run Selected**

Steps - Bagging Models cont..

- Drag and drop **Apply Math Operation(IP-third),** connect the output of **Apply Math Operation(IP-second)** to it.
- Select **Apply Math Operation(IP-third),** click on **Run Selected**
- Drag and drop **Edit Metadata,** connect it to the **Apply Math Operation(IP-third)**
- Select **Edit Metadata,** click on **Run Selected**
- **Note:** Select the properties of **Apply Math Operation(IP-first), Apply Math Operation(IP-second), Apply Math Operation(IP-third)** and **Edit Metadata** before run

Steps - Bagging Models cont..

Fig1: Split Data



Steps - Bagging Models cont..

Fig2:Properties - Linear Regression

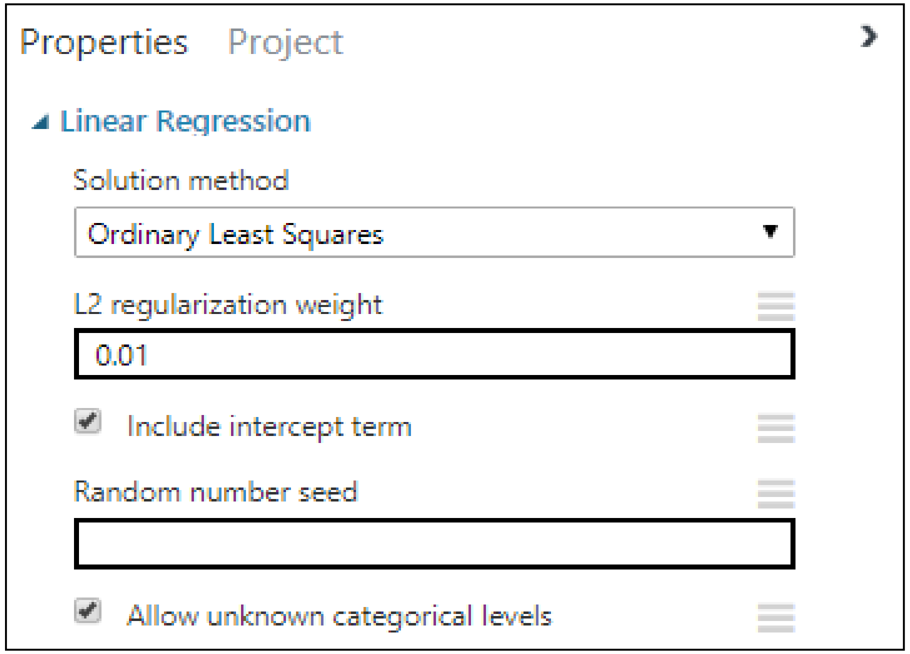
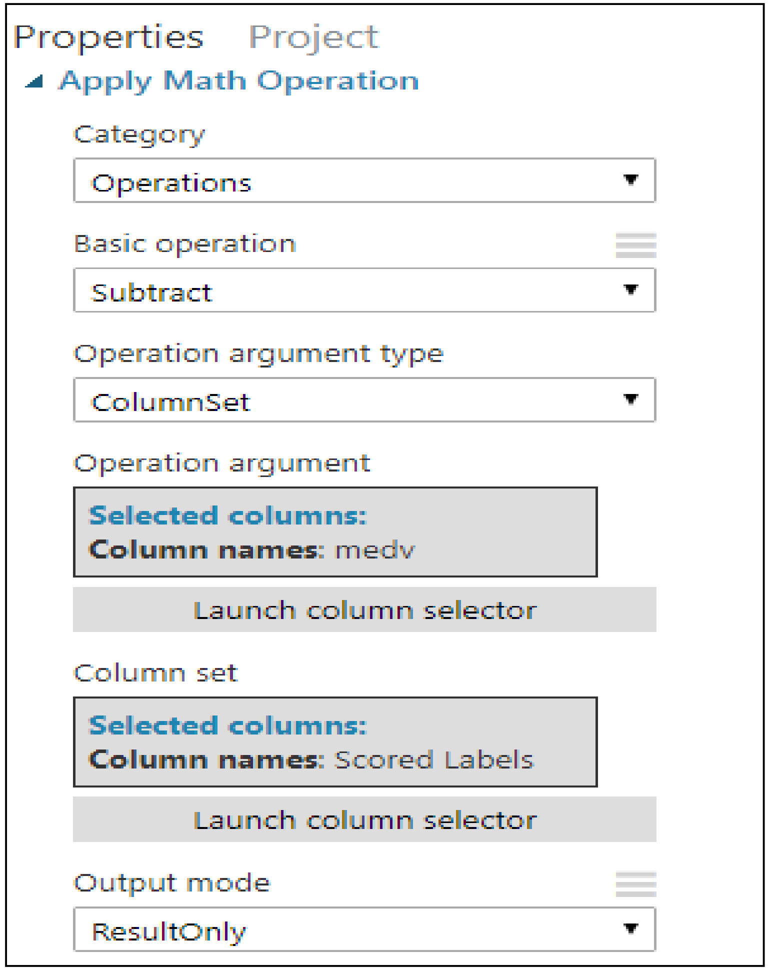


Fig3: Properties - Apply Math Operation(LR-first)



statinfer

Steps - Bagging Models cont..

Fig4:Properties - Apply Math Operation(LR-second)

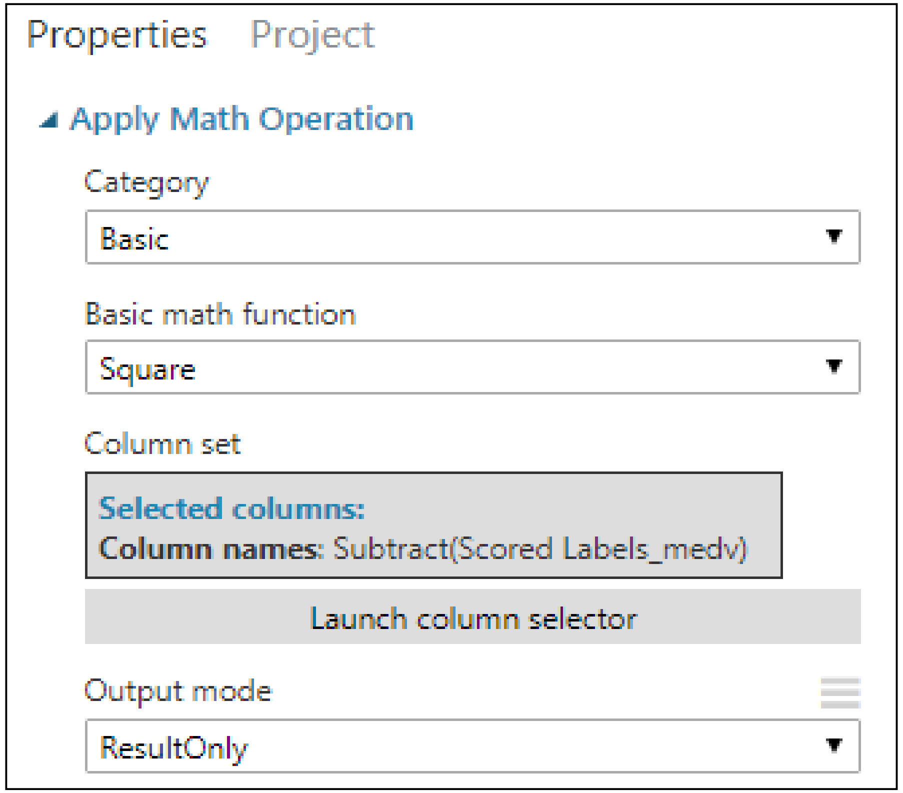
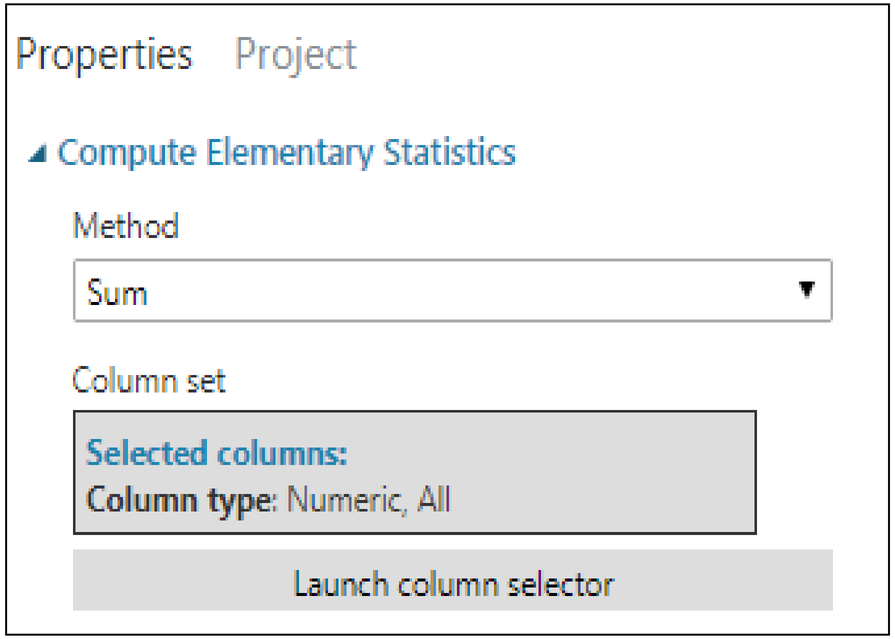


Fig5: Properties - Compute Elementary Statistics(LR)



Steps - Bagging Models cont..

Fig6:Properties - Decision Forest Regression

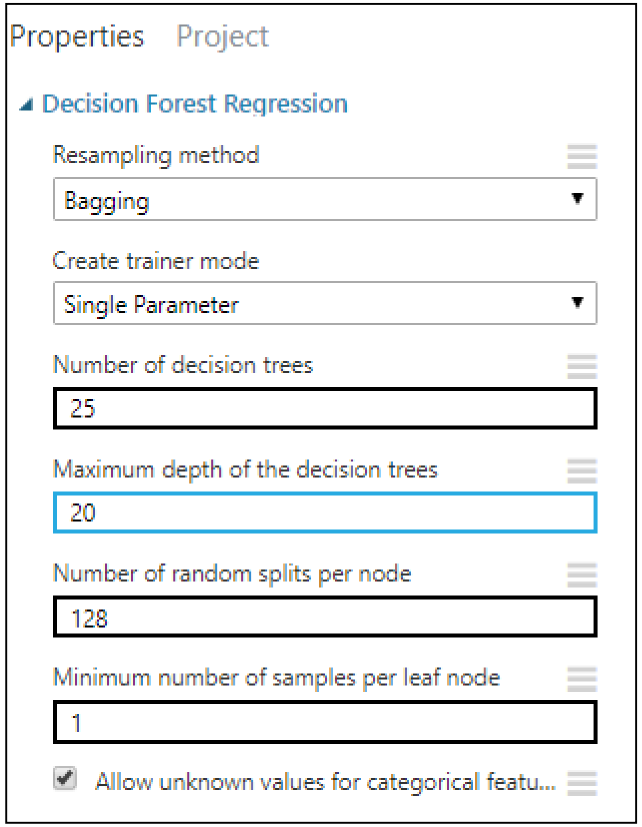
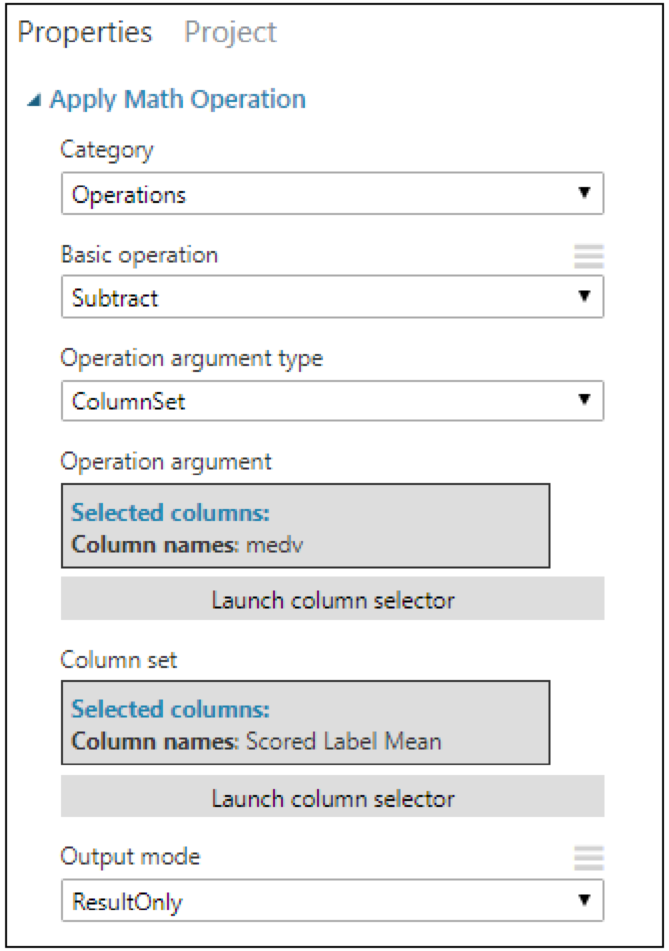


Fig7: Properties - Apply Math Operation(DF-first)



Steps - Bagging Models cont..

Fig8:Properties - Apply Math Operation(DF-second)

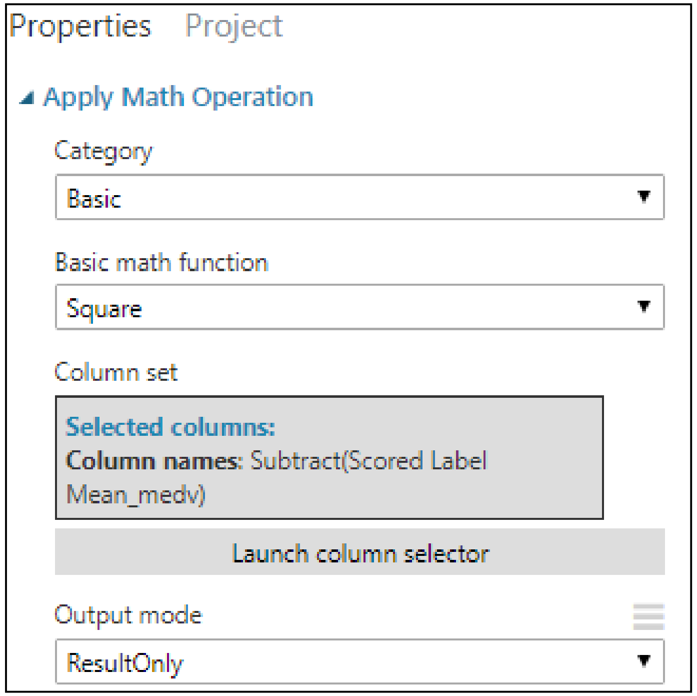
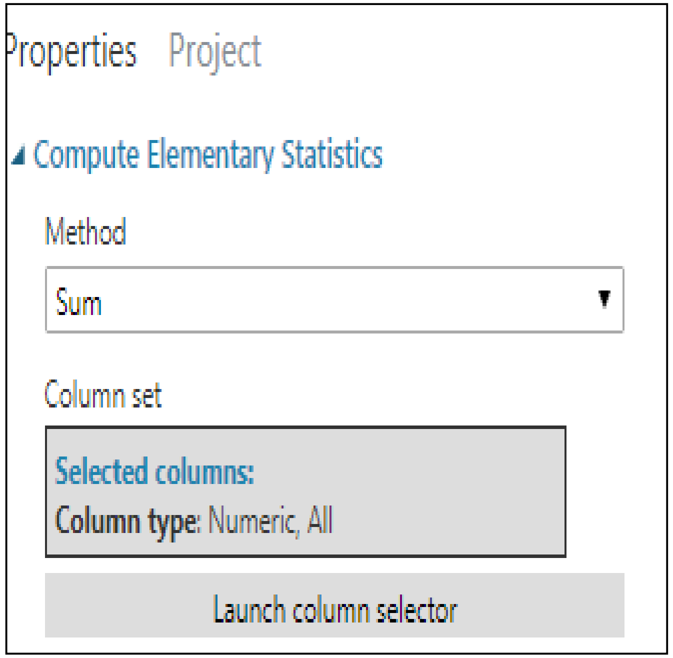


Fig9: Properties - Compute Elementary Statistics(DF)



Steps - Bagging Models cont..

Fig10:Properties - Decision Forest Regression

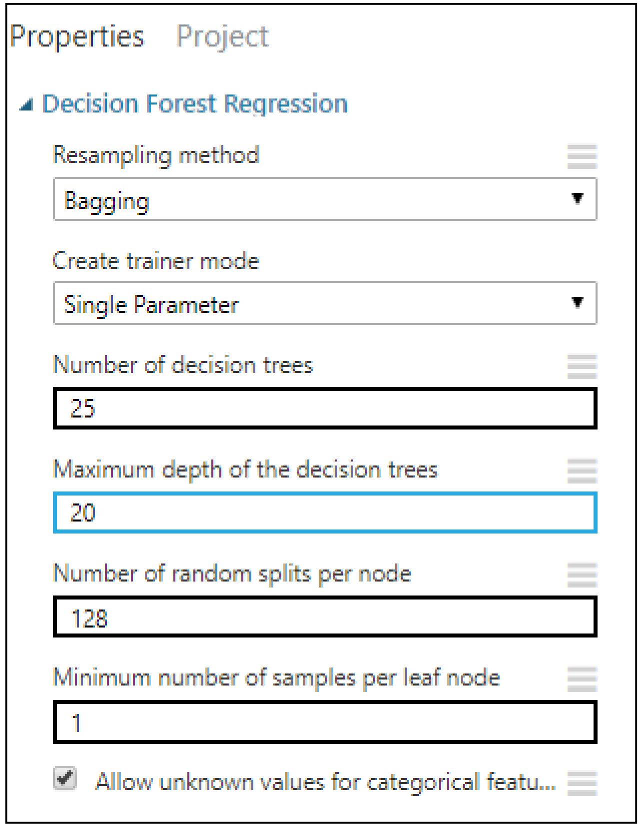
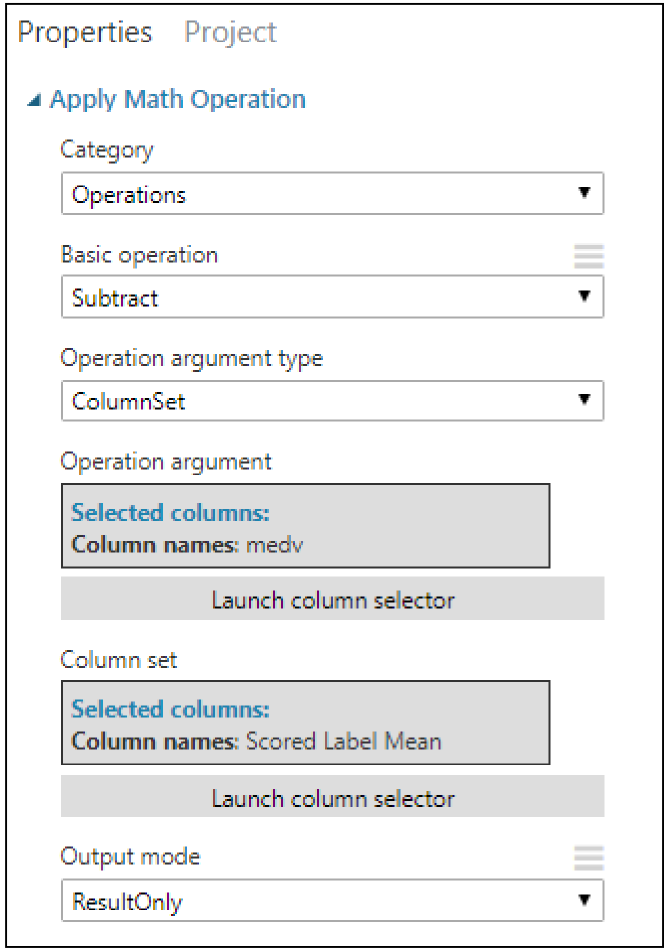


Fig11: Properties - Apply Math Operation(DF-first)



Steps - Bagging Models cont..

Fig12:Properties - Apply Math Operation(DF-second)

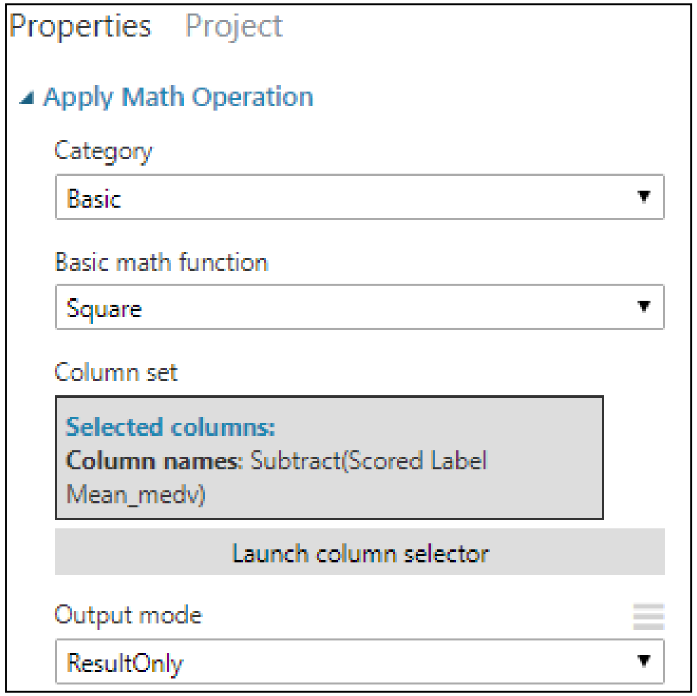
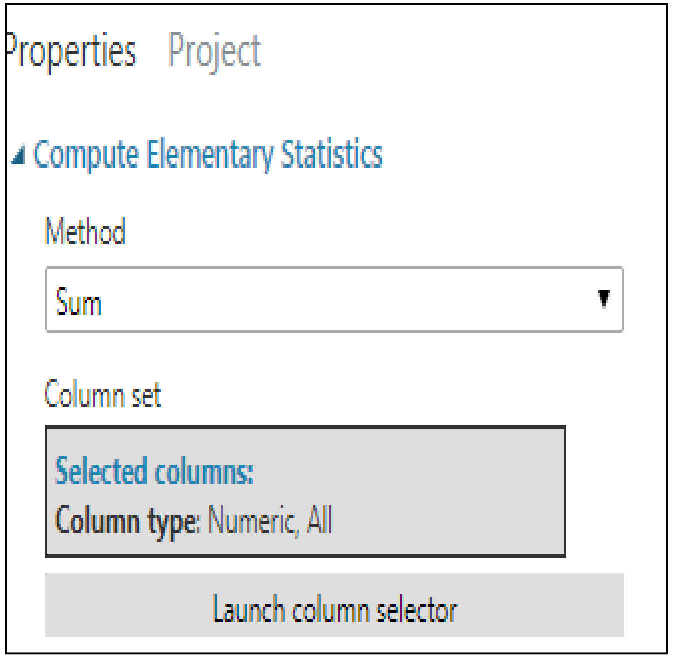


Fig13: Properties - Compute Elementary Statistics(DF)



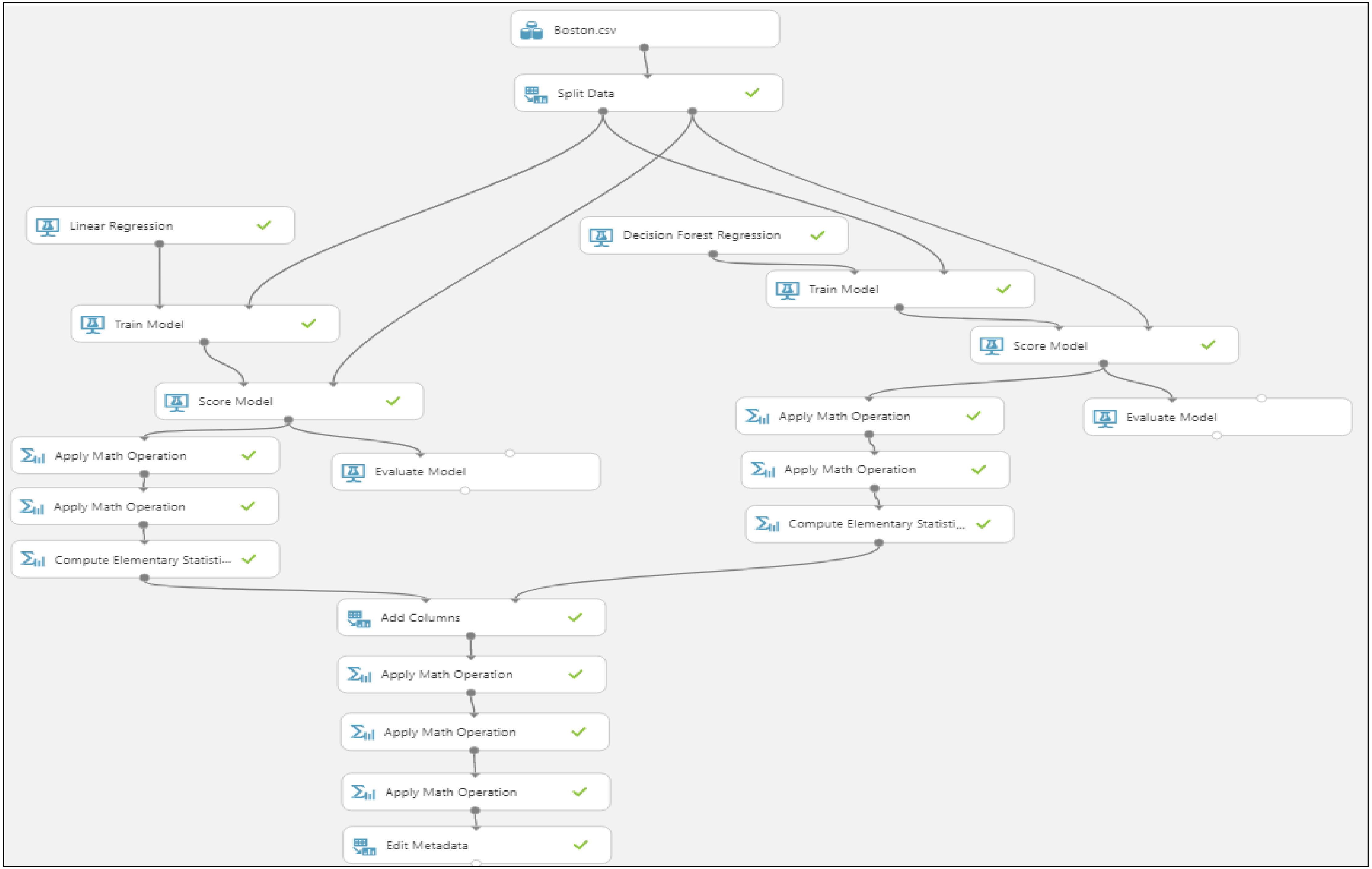


Fig14: Overall Modal

Steps - Bagging Models cont..

Fig15: error(squared deviation) – Linear Regression

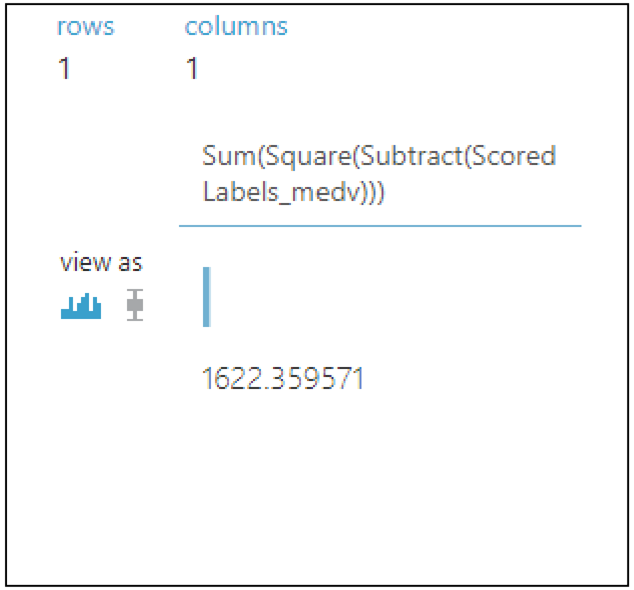
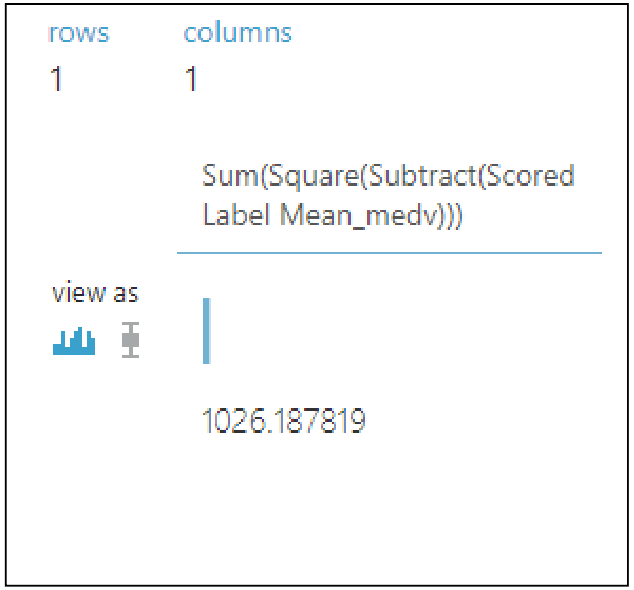


Fig16: error(squared deviation) – Decision Forest Regression



Steps - Bagging Models cont..

Fig17: Improvement Percentage

rows 1

columns 1

view as

Improvement Percentage
36.747202

Random Forest

Random Forest

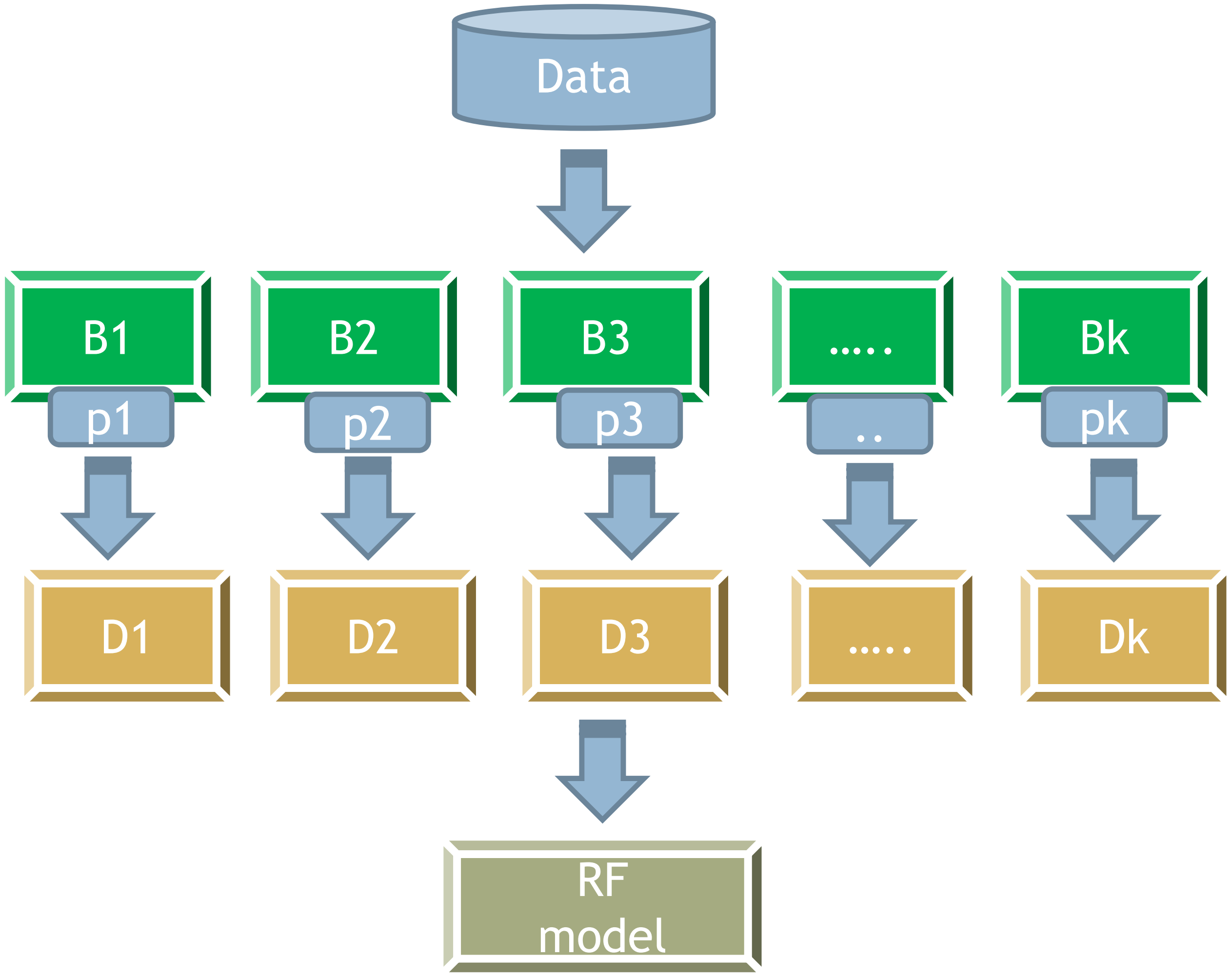
- Random forest is a specific case of bagging methodology. Bagging on decision trees is random forest
- Like many trees form a forest, many decision tree model together form a Random Forest model

Random Forest

- In random forest we induce two types of randomness
  - Firstly, we take the boot strap samples of the population and build decision trees on each of the sample.
  - While building the individual trees on boot strap samples, we take a subset of the features randomly
- Random forests are very stable they are as good as SVMs and sometimes better

Random Forest algorithm

- The training dataset D with t number of features
- Draw k boot strap sample sets from dataset D
- For each boot strap sample i
  - Build a decision tree model M_i using only p number of features (where p<<t)
  - Each tree has maximal strength they are fully grown and not pruned.
  - We will have total of k decision treed M_1 , M_2 ,….. M_k; Each of these trees are built on reactively different training data and different set of features
  - Vote over for the final classifier output and take the average for regression output

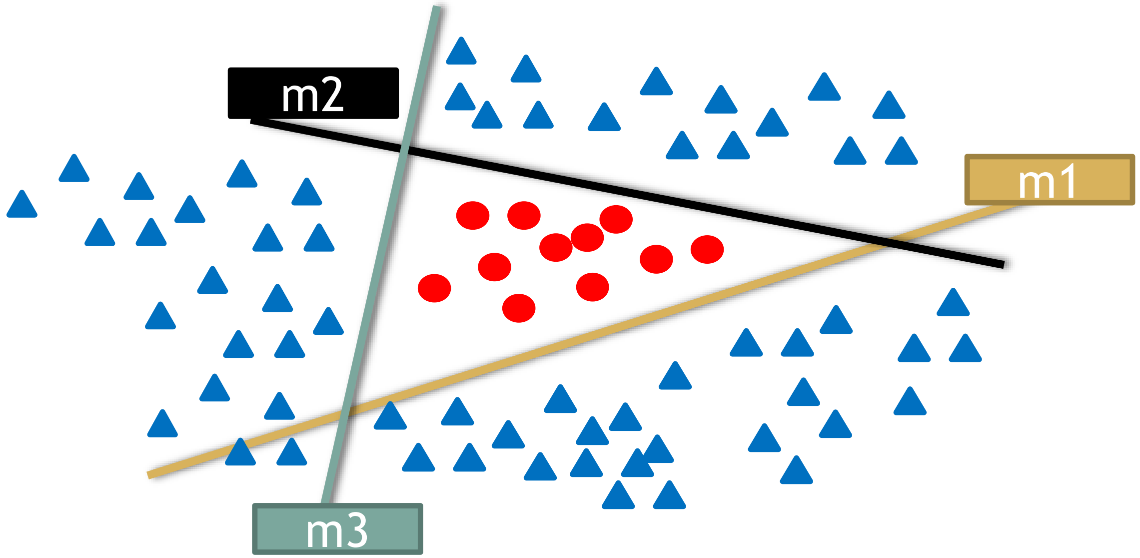


The Random Factors in Random Forest

- We need to note the most important aspect of random forest, i.e inducing randomness into the bagging of trees. There are two major sources of randomness
  - Randomness in data: Boot strapping, this will make sure that any two samples data is somewhat different
  - Randomness in features: While building the decision trees on boot strapped samples we consider only a random subset of features.
- Why to induce the randomness?
  - The major trick of ensemble models is the independence of models.
  - If we take the same data and build same model for 100 times, we will not see any improvement
  - To make all our decision trees independent, we take independent samples set and independent features set
  - As a rule of thumb we can consider square root of the number features, if 't' is very large else p=t/3

Why Random Forest Works

- For a training data with 20 features we are building 100 decision trees with 5 features each, instated of single great decision. The individual trees may be weak classifiers.
- Its like building weak classifiers on subsets of data. The grouping of large sets of random trees generally produces accurate models.



- In this example we have three simple classifiers.
- m1 classifies anything above the line as +1 and below as -1, m2 classifies all the points above the line as -1 and below as +1 and m3 classifies everything on the left as -1 and right as +1
- Each of these models have fair amount of misclassification error.
- All these three weak models together make a strong model.

LAB: Random Forest

LAB: Random Forest

- Dataset: /Car Accidents IOT/Train.csv
- Build a decision tree model to predict the fatality of accident
- Build a decision tree model on the training data.
- On the test data, calculate the classification error and accuracy.
- Build a random forest model on the training data.
- On the test data, calculate the classification error and accuracy.
- What is the improvement of the Random Forest model when compared with the single tree?

Steps - Random Forest

- Decision Trees:
- Drag and drop the **Training Dataset** and **Test Dataset** into the canvas
- Drag and drop **Two-Class Boosted Decision Tree, Train Model, Score Model** and **Evaluate Model**
- Connect **Two-Class Boosted Decision Tree** to the first input of **Train Model** and **Training Dataset** to the Second input of **Train Model**
- Connect the output of **Train Model** first input of **Score Model** and **Test Dataset** to the Second input of **Score Model**
- Connect the output of **Score Model** to the input of **Evaluate Model**
- Click on **Train Model** and select the column for which the prediction is done(Fatal)
- Click run and visualize the output of **Evaluate Model**
- **Note:** Select the properties for **Two-Class Decision Forest, Train Model, Score Model** and **Evaluate Model** before run

Steps - Random Forest

- Random Forest:
- Drag and drop **Two-Class Decision Forest, Train Model, Score Model** and **Evaluate Model**
- Connect **Two-Class Decision Forest** to the first input of **Train Model** and **Training Dataset** to the Second input of **Train Model**
- Connect the output of **Train Model** first input of **Score Model** and **Test Dataset** to the Second input of **Score Model**
- Connect the output of **Score Model** to the input of **Evaluate Model**
- Click on **Train Model** and select the column for which the prediction is done(Fatal)
- Click run and visualize the output of **Evaluate Model**
- **Note:** Select the properties for **Two-Class Decision Forest, Train Model, Score Model** and **Evaluate Model** before run

Steps - Random Forest

Fig18: Properties - Two-Class Boosted Decision Tree

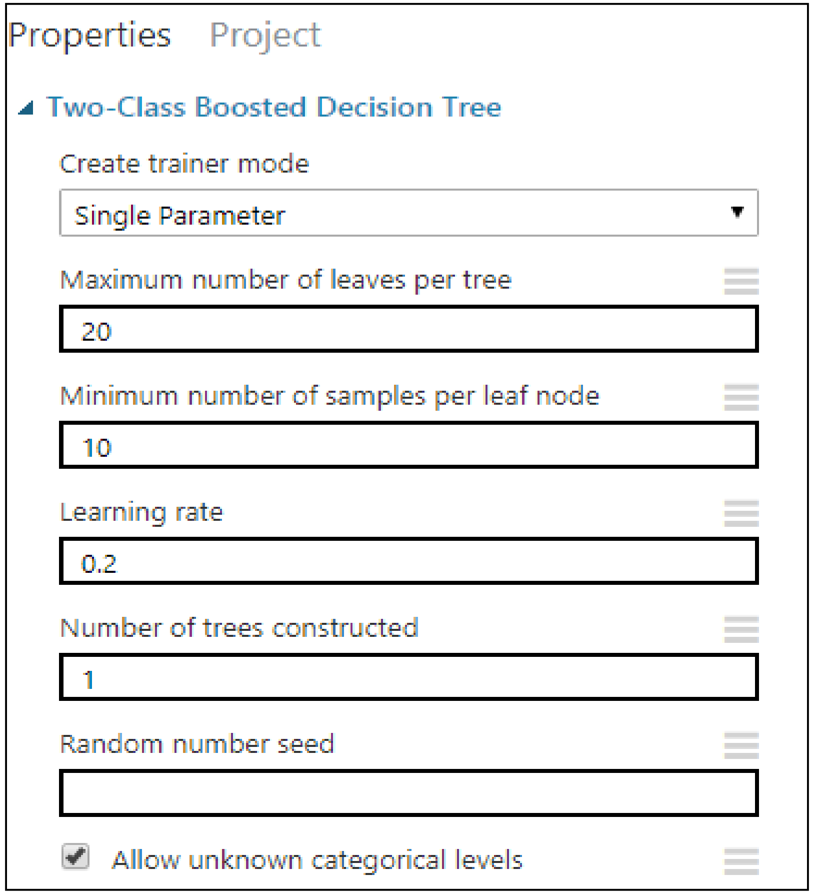
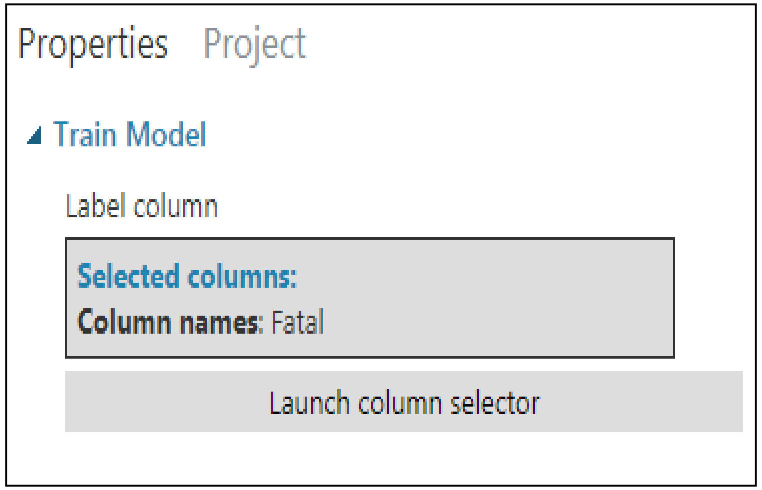


Fig19: Properties - Train Mode(Decision Trees)



Steps - Random Forest

Fig20: Properties - Two-Class Decision Forest

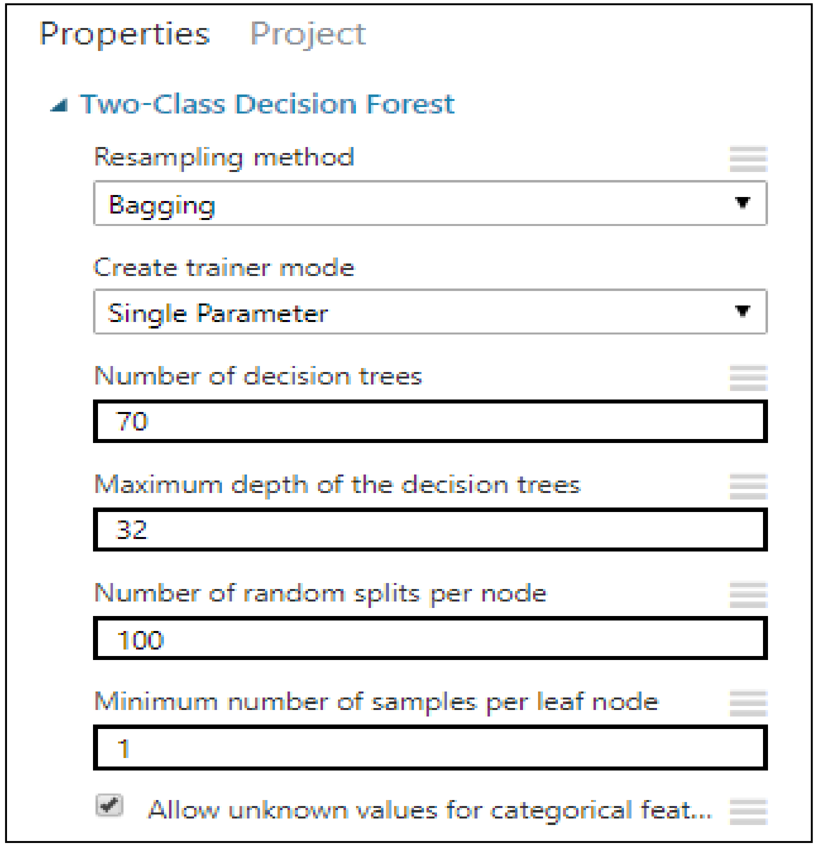
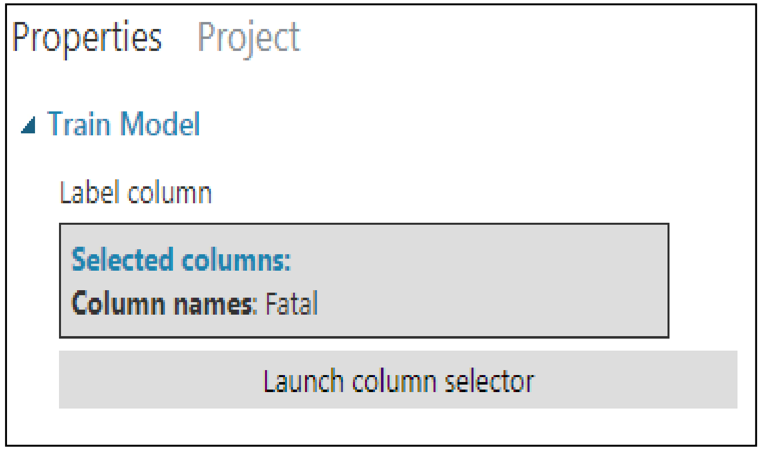
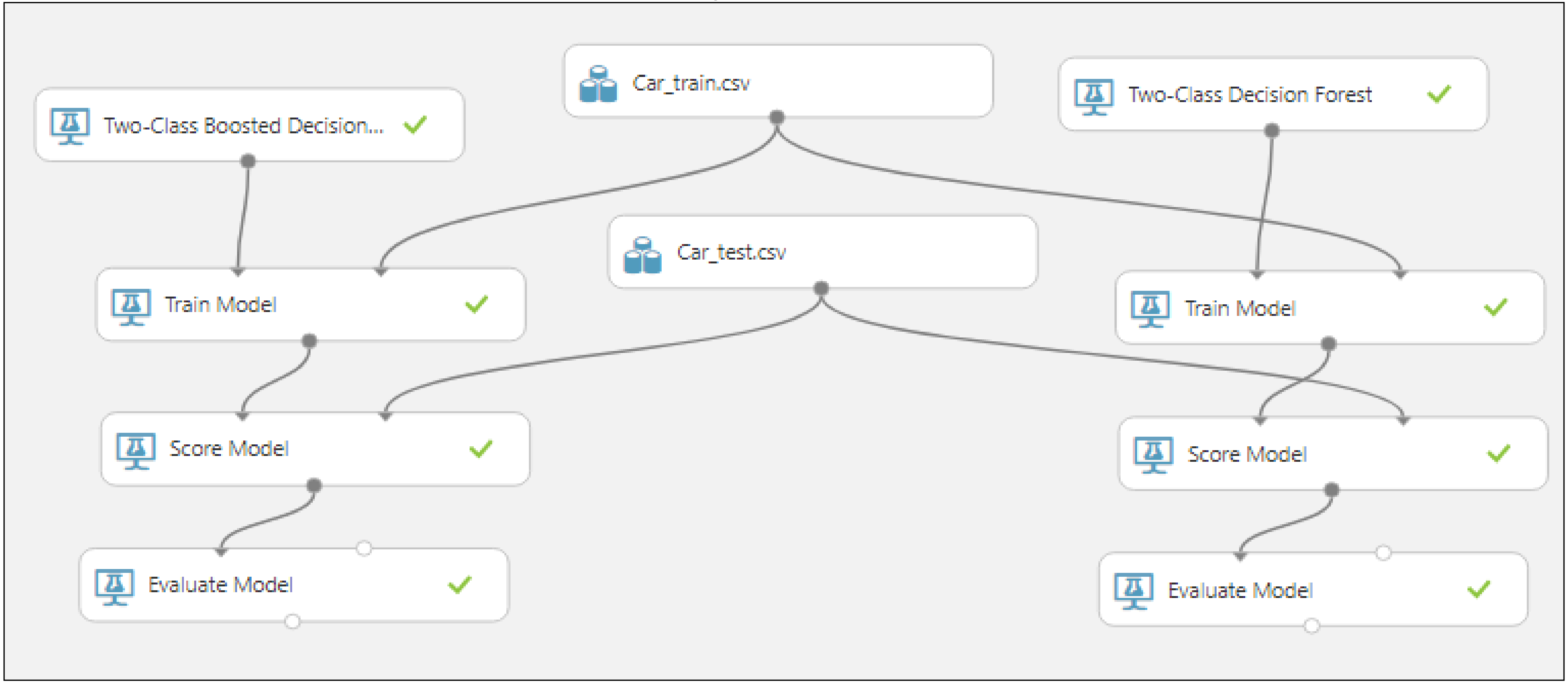


Fig21: Properties - Train Mode(Decision Forest)



Steps - Random Forest

Fig22: Overall Model



Steps - Random Forest

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
4817	356	0.861	0.843	0.5	0.893
False Positive	**True Negative**	**Recall**	**F1 Score**		
900	2992	0.931	0.885		
Positive Label	**Negative Label**				
1	0				

Fig23: Accuracy – Decision Tree Modal

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
4979	194	0.935	0.926	0.5	0.970
False Positive	**True Negative**	**Recall**	**F1 Score**		
399	3493	0.962	0.944		
Positive Label	**Negative Label**				
1	0				

Fig24: Accuracy – Decision Forest Modal

statinfer

Case Study: Direct Mail Marketing Response Model

LAB: Direct Mail Marketing Response Model

- Large Marketing Response Data/train.csv
- How many variables are there in the dataset?
- Take a one third of the data as training data and one third as test data
- Look at the response rate from target variables
- Find out the overall missing values and missing values by variables
- Do the missing value and outlier treatment, prepare data for analysis
- Build a RF model
- Find the training data accuracy
- Find the accuracy on test data

Steps - Direct Mail Marketing Response Model

- Drag and drop the **Dataset** into the canvas
- Drag and drop the **Split Data**(90% training, 10% testing) connect to the dataset
- Drag and drop **Clean Missing Data,** connect it to **Split Data**
- Drag and drop **Convert to dataset**, connect it to **Clean Missing Data**
- Drag and drop **Clean Missing Data,** connect it to **Convert to dataset**
- Drag and drop three **Select Columns from the Dataset**, connect all the three to **Clean Missing Data**
- Drag and drop two **Select Columns from the Dataset**, connect it to **Select Columns from the Dataset**(for which numeric is selected)
- Drag and drop **Clean Missing Data**, connect it to **Select Columns from the Dataset**(for which string is selected)

Steps - Direct Mail Marketing Response Model

- Drag and drop **Select Columns from the Dataset**, connect it to **Clean Missing Data**
- Drag and drop **Clean Missing Data**, connect it to **Select Columns from the Dataset**(for which Boolean is selected)
- Drag and drop two **Edit Metadata**, connect it to **Select Columns from the Dataset**(for which Double is selected)
- Drag and drop **Clean Missing Data**, connect it to **Edit Metadata**
- Drag and drop **Add Columns**, connect **Select Columns from the Dataset**(for which Integer is selected) and **Edit Metadata** to it
- Drag and drop another **Add Columns**, connect **Select Columns from the Dataset** and Previous **Add Columns** to it
- Drag and drop **Add Columns**, connect **Add Columns** and **Clean Missing Data** to it

Steps - Direct Mail Marketing Response Model

- Drag and drop **Split Data**, connect it to the **Add Columns**
- Drag and drop **Two-Class Decision Forest, Train Model, Score Model** and **Evaluate Model**
- Connect **Two-Class Decision Forest** to the first input of **Train Model** and first output of **Split data** to the Second input of **Train Model**
- Connect the output of **Train Model** first input of **Score Model** and second output of **Split data** to the Second input of **Score Model**
- Connect the output of **Score Model** to the input of **Evaluate Model**
- Click on **Train Model** and select the column for which the prediction is done(target)
- Click on run, Visualize the Evaluate Model
- **Note:** Select the properties for all by seeing the figure before running

Steps - Direct Mail Marketing Response Model

Fig25: Split Data(90% Training, 10% Test)

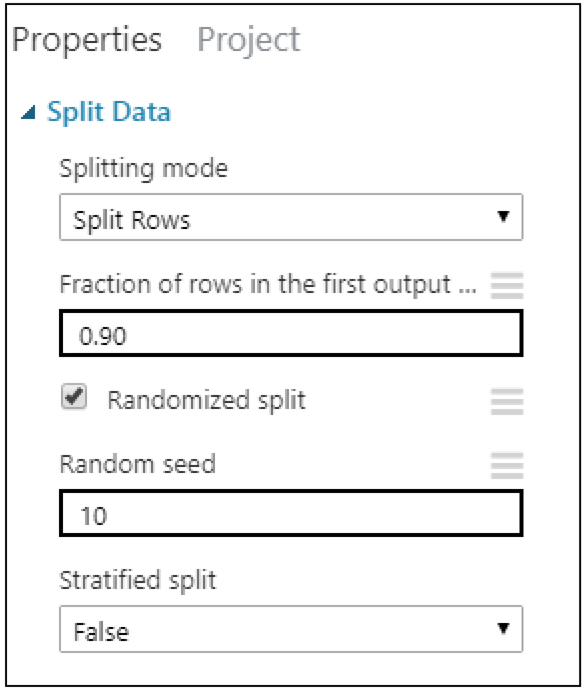
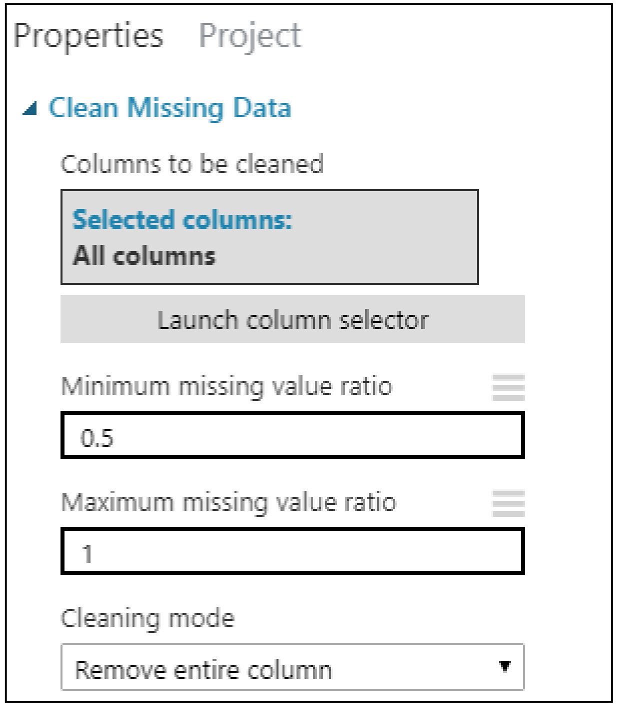


Fig26: Clean Missing Data(>50% Null Value Columns)



Steps - Direct Mail Marketing Response Model

Fig27: Setting NA as Missing Value

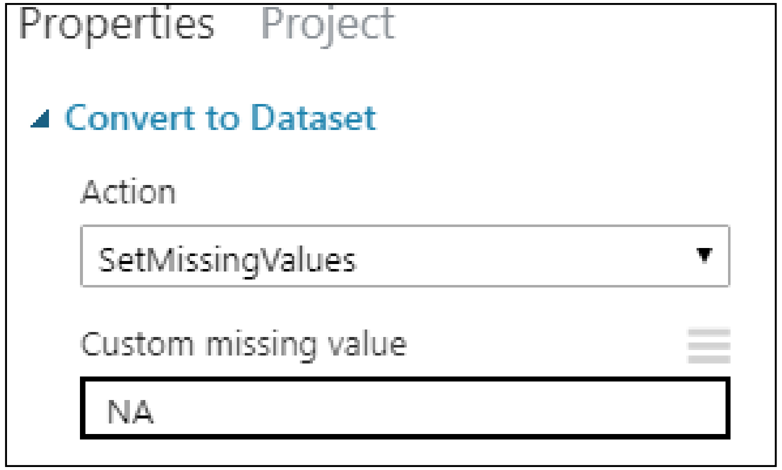
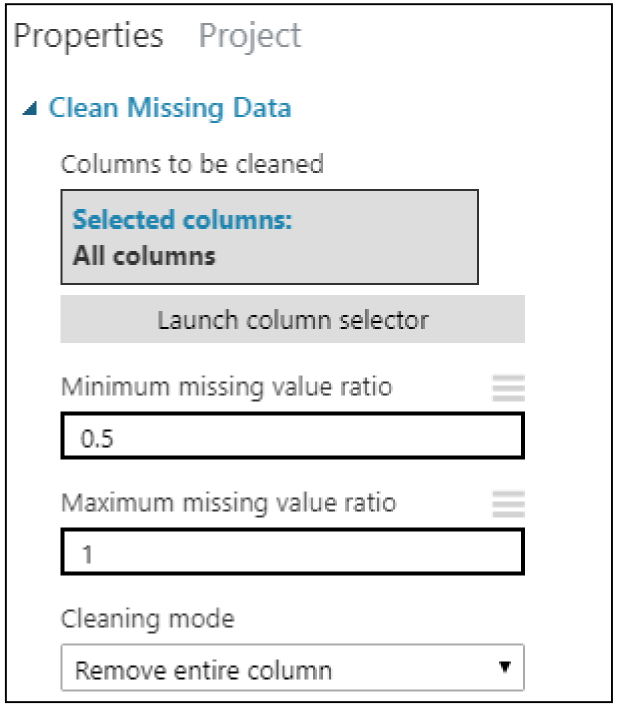


Fig28: Clean Missing Data(>50% Null Value Columns)



Steps - Direct Mail Marketing Response Model

Fig29: Selecting Columns of Type String

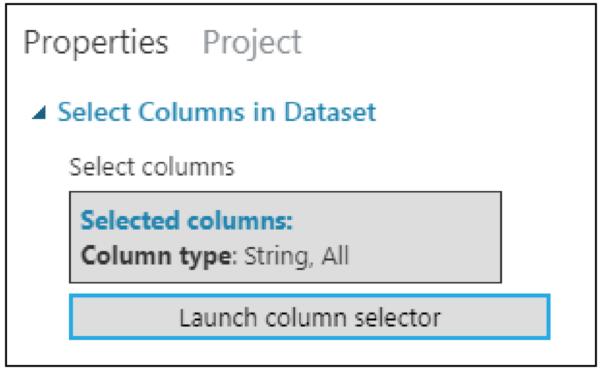


Fig30:Selecting Columns of Type Numeric

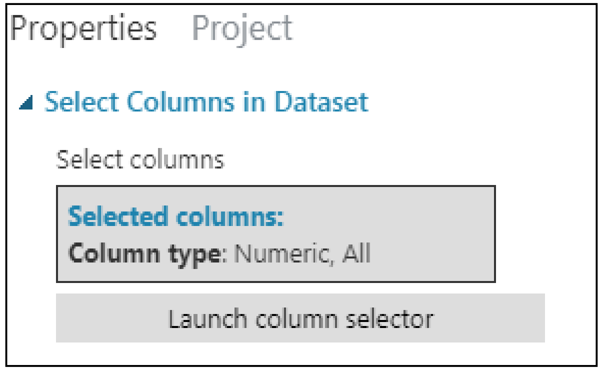
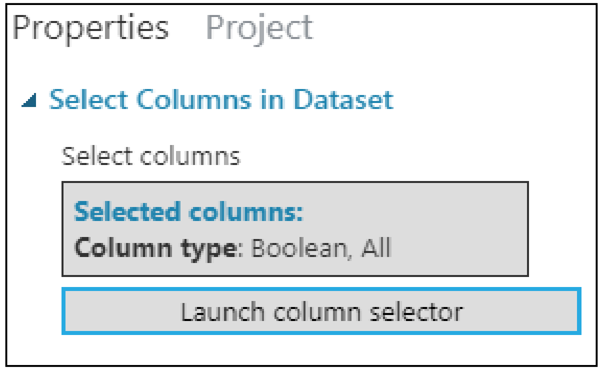


Fig31:Selecting Columns of Type Boolean



Steps - Direct Mail Marketing Response Model

Fig32: Selecting Integer from Numeric

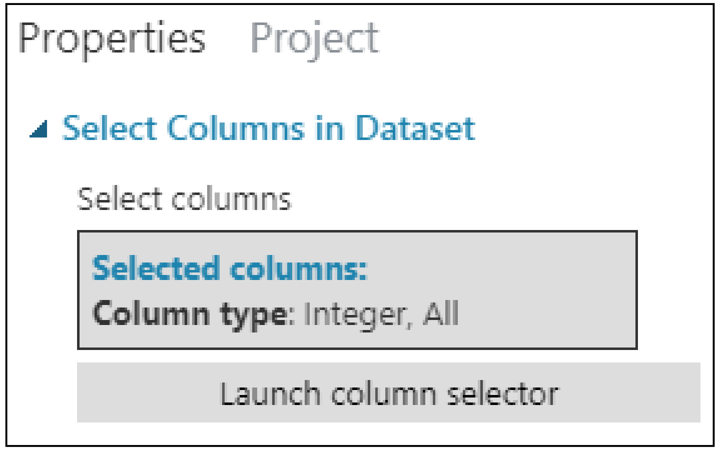
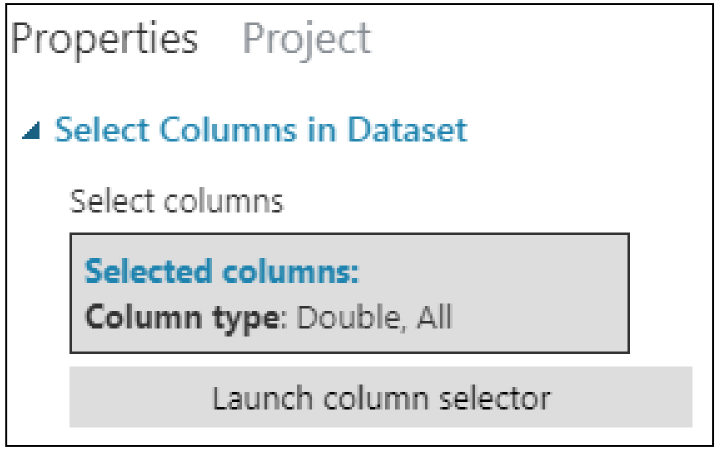


Fig33: Selecting Double from Numeric



Steps - Direct Mail Marketing Response Model

Fig34: Missing value Imputation(String)

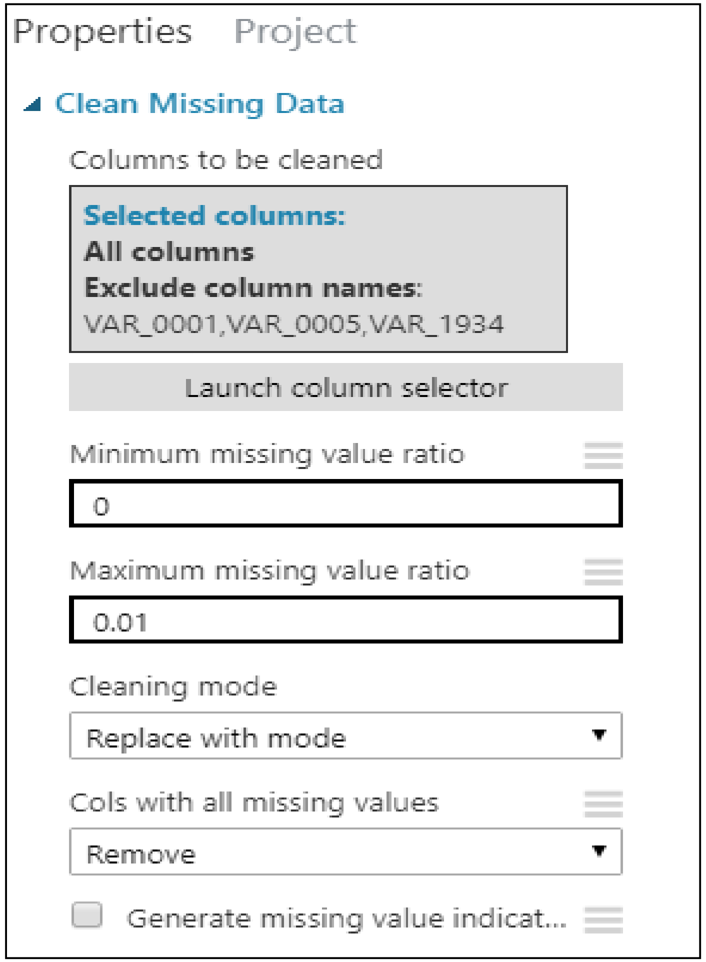
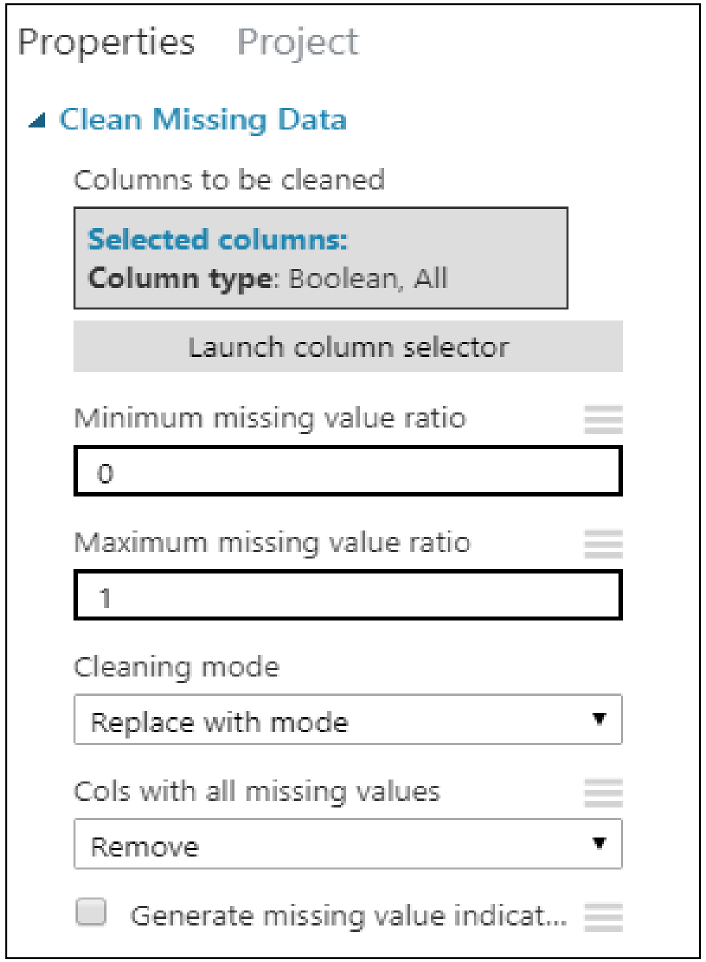


Fig35: Missing Value Imputation(Boolean)



Steps - Direct Mail Marketing Response Model

Fig36: Converting From Double to Integer

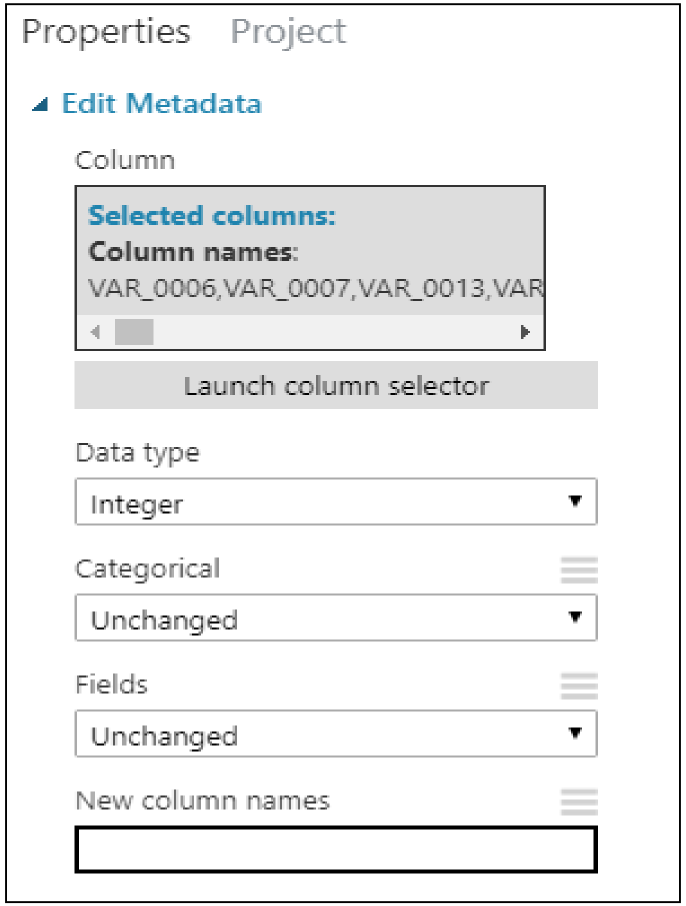
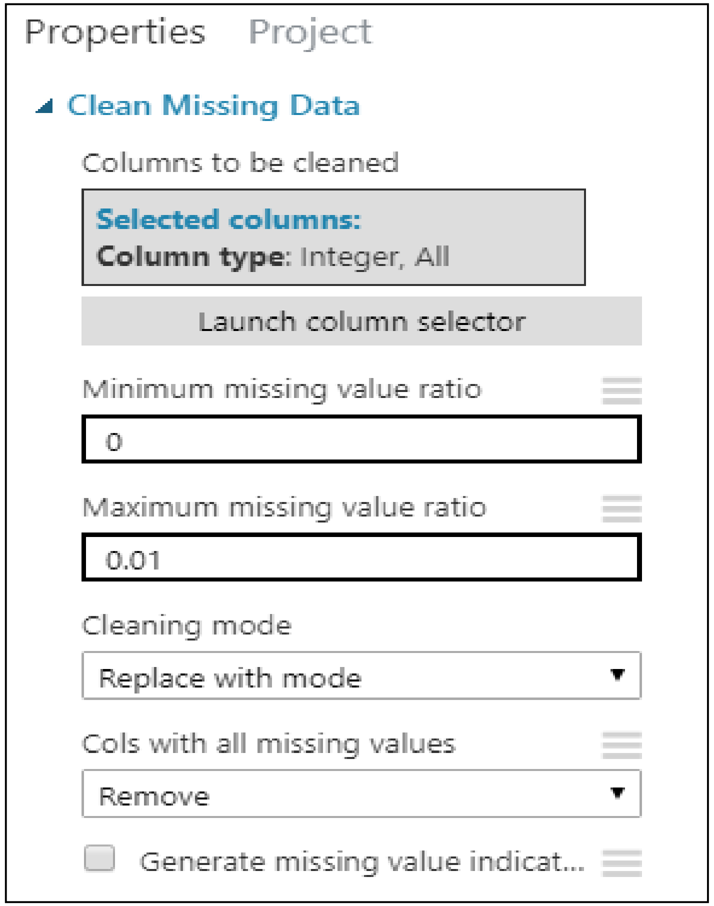


Fig37: Missing Value Imputation(Integers)



Steps - Direct Mail Marketing Response Model

Fig38: Properties - Two-Class Decision Forest

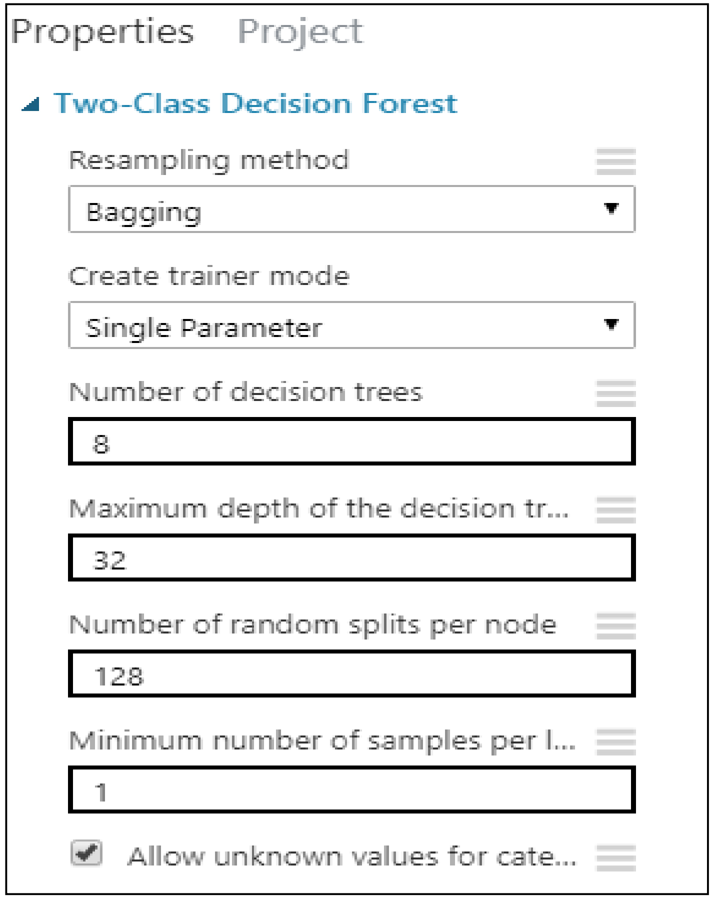
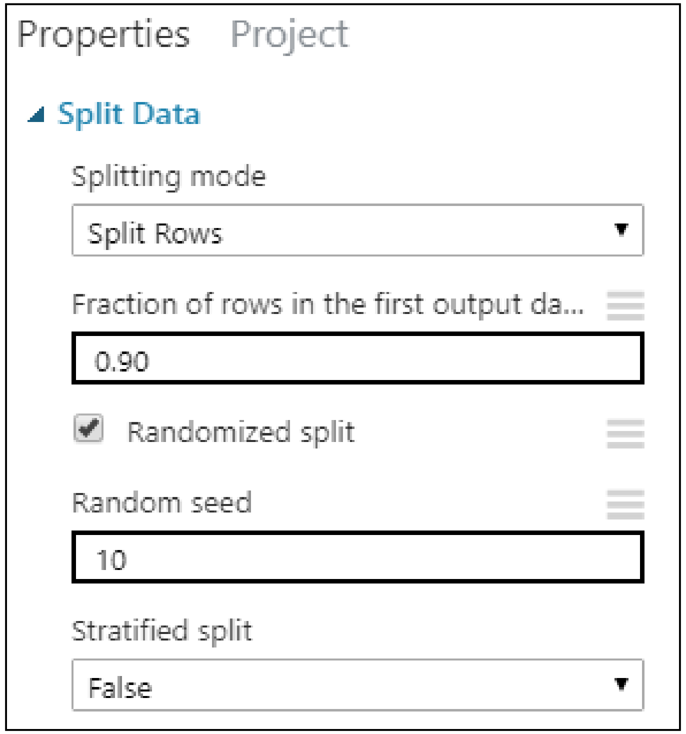


Fig39: Split Data(90% Training, 10% Validation)



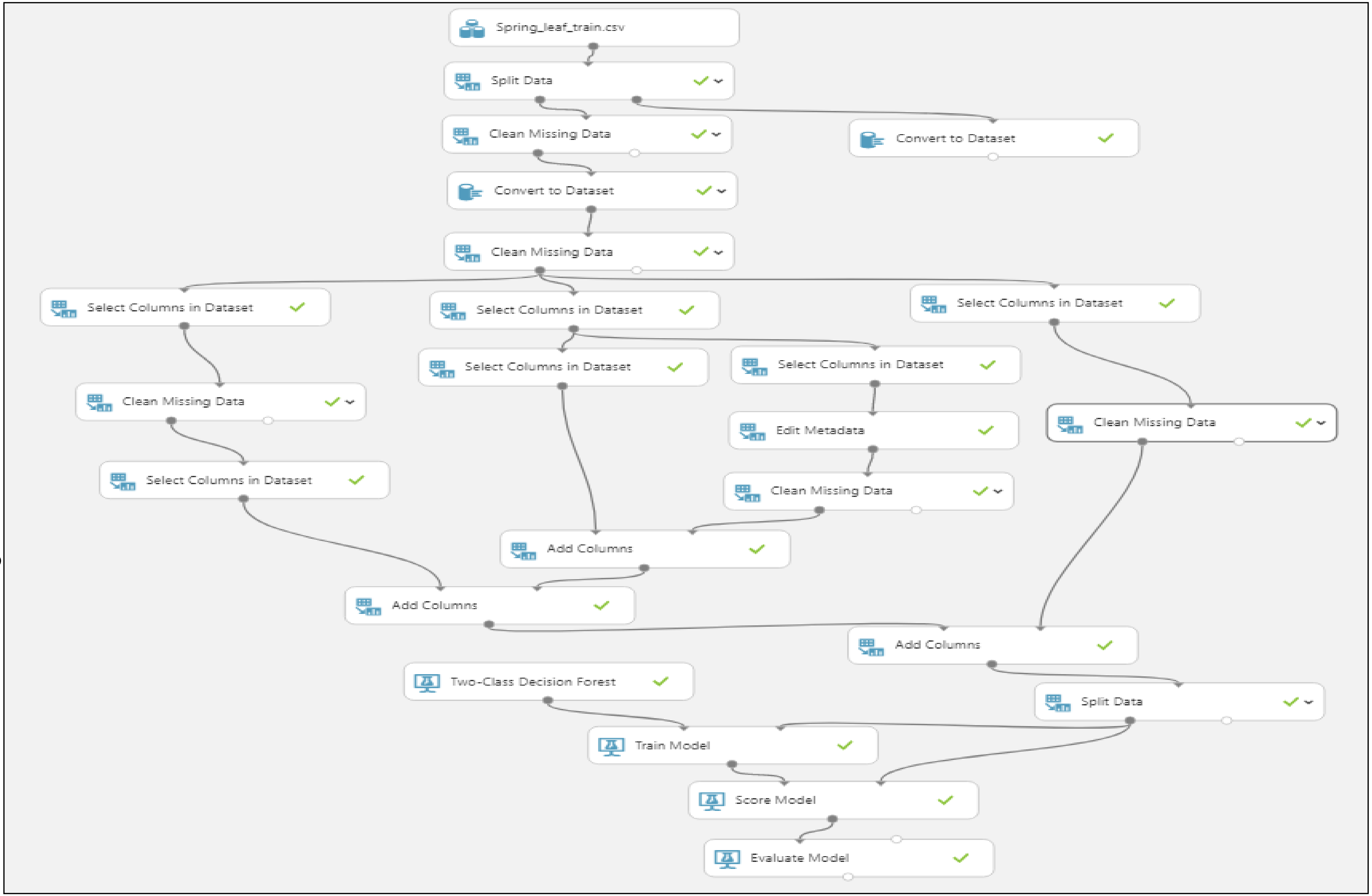


Fig40: Overall Modal

Steps - Direct Mail Marketing Response Model

Fig41: Accuracy - Training

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
23603	3868	0.966	0.994	0.5	0.997
False Positive	**True Negative**	**Recall**	**F1 Score**		
143	90023	0.859	0.922		
Positive Label	**Negative Label**				
1	0				

Fig42: Accuracy - Validation

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
567	2414	0.777	0.529	0.5	0.678
False Positive	**True Negative**	**Recall**	**F1 Score**		
505	9585	0.190	0.280		
Positive Label	**Negative Label**				
1	0				

Fig43: Accuracy - Test

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
620	2701	0.773	0.509	0.5	0.687
False Positive	**True Negative**	**Recall**	**F1 Score**		
599	10603	0.187	0.273		
Positive Label	**Negative Label**				
1	0				

When Ensemble doesn’t work?

When Ensemble doesn't work?

- The models have to be independent
- We can't build the same model multiple times and expect the error to reduce.
- We may have to bring in the independence by choosing subsets of data, or subset of features while building the individual models
- Ensemble may backfire if we use dependent models that are already less accurate. The final ensemble might turn out to be even worse model.

When Ensemble doesn't work?

- Yes, there is a small disclaimer in "Wisdom of Crowd" theory.
- We need moderately good independent individuals. If we collate any dependent individuals with poor knowledge, then we might end up with an even worse ensemble.
- For example, we built three models, model-1 , model-2 are bad, model-3 is good. Most of the times ensemble will result the combined output of model-1 and model-2, based on voting

Conclusion

Conclusion

- Ensemble methods are the most widely used methods these days. With advanced machines, its not really a huge task to build multiple models.
- Both bagging and boosting does a good job of reducing bias and variance
- Random forests are relatively fast, since we are building many small trees, it doesn’t put lot of pressure on the computing machine
- Random forest can also give the variable importance. We need to be careful with categorical features, random forests trend to give higher importance to variables with higher number of levels.
- In Boosted algorithms we may have to restrict the number of iterations to avoid overfitting.
- Ensemble models are the final effort of a data scientist, while building the most suitable predictive model for the data.

statinfer

Boosting Method in Azure

Venkat Reddy

Contents

Contents

- What is boosting
- Boosting algorithm
- Boosting illustration
- Theory behind Boosting Algorithm
- Building models using Boosted Decision Tree

Boosting

statinfer

- Boosting is one more famous ensemble method
- Boosting uses a slightly different techniques to that of bagging.
- Boosting is a well proven theory that works really well on many of the machine learning problems like speech recognition
- If bagging is wisdom of crowds then boosting is wisdom of crowds where each individual is given some weight based on their expertise

Boosting

- Boosting in general decreases the bias error and builds strong predictive models.
- Boosting is an iterative technique. We adjust the weight of the observation based on the previous classification.
- If an observation was classified incorrectly, it tries to increase the weight of this observation and vice versa.

Boosting Main idea

Take a random sample from population of size N

Each record has 1/N Chance of picking

Let 1/N be the weight w

Build a classifier

Note down the accuracy

The Classifier may misclassify some of the records. Note them down

Take a weighted sample

This time give more weight to misclassified records from previous model

Update the weight w accordingly to pick the misclassified records

Build a new classifier on the reweighted sample

Since we picked many previously misclassified records, we expect this model to build a better model for those records

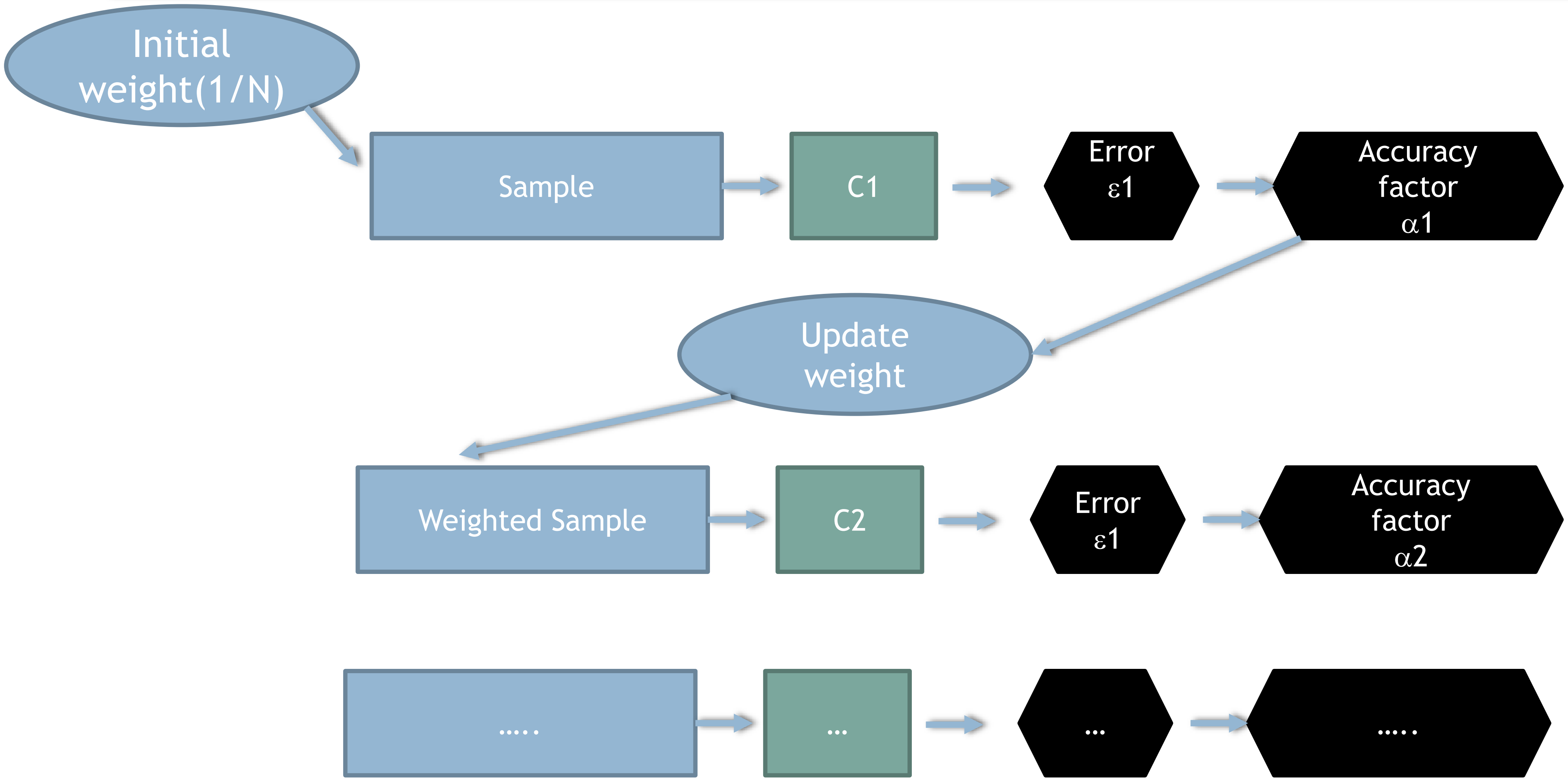
Check the error and resample

Does this classifier still has some misclassifications

If yes, then re-sample

Final Weighted Classifier $C = \sum \alpha_i c_i$

Boosting Main idea



statinfer

How weighted samples are taken

Data	1	2	3	4	5	6	7	8	9	10
Class	-	-	+	+	-	+	-	-	+	+
Predicted Class M1	-	-	-	-	-	-	-	-	+	+
M1 Result	✔	✔	✖	✖	✔	✖	✔	✔	✔	✔

Weighted Sample1	1	2	3	4	5	6	7	4	3	6
Class	-	-	+	+	-	+	-	+	+	+
Predicted Class M2	-	-	+	+	+	+	+	+	+	+
M2 Result	✔	✔	✔	✔	✖	✔	✖	✔	✔	✔

Weighted Sample2	6	5	3	4	5	6	7	7	5	7
Class	+	-	+	+	-	+	-	-	-	-
Predicted Class M3	+	-	+	+	-	+	-	-	-	-
M3 Result	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔

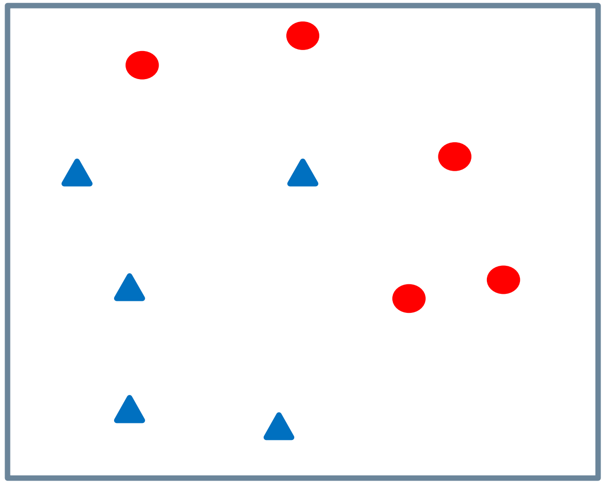
Boosting illustration

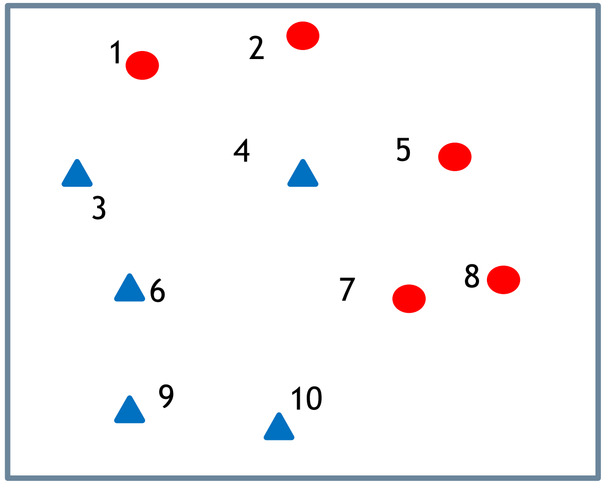
Boosting illustration

Below is the training data and their class

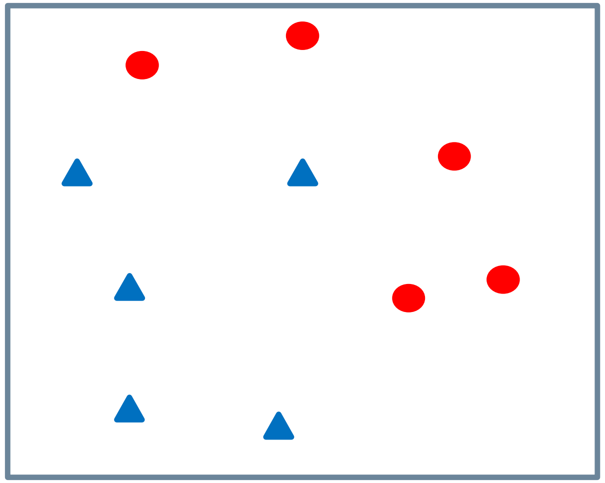
We need to take a note of record numbers, they will help us in weighted sampling later

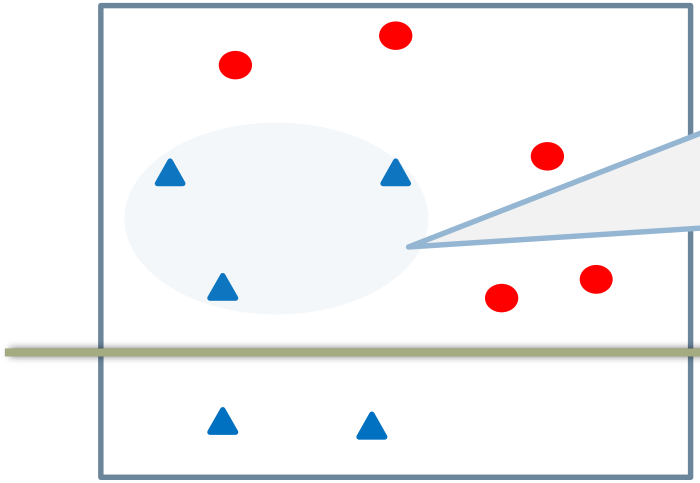
Data Points	1	2	3	4	5	6	7	8	9	10
Class	-	-	+	+	-	+	-	-	+	+





Boosting illustration

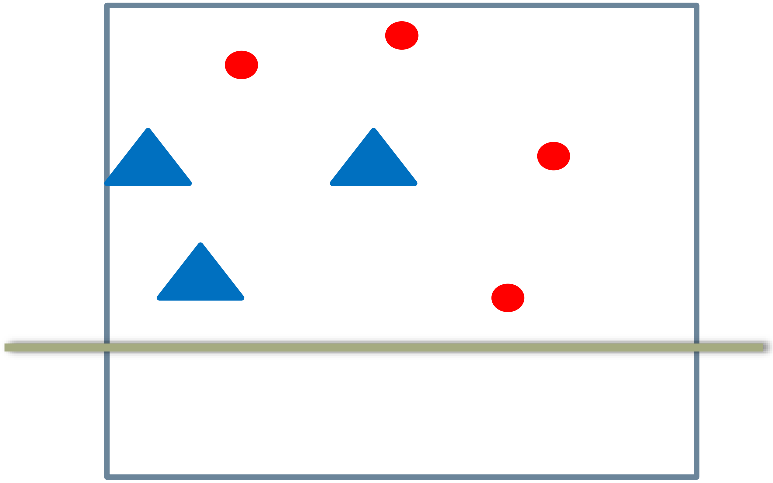


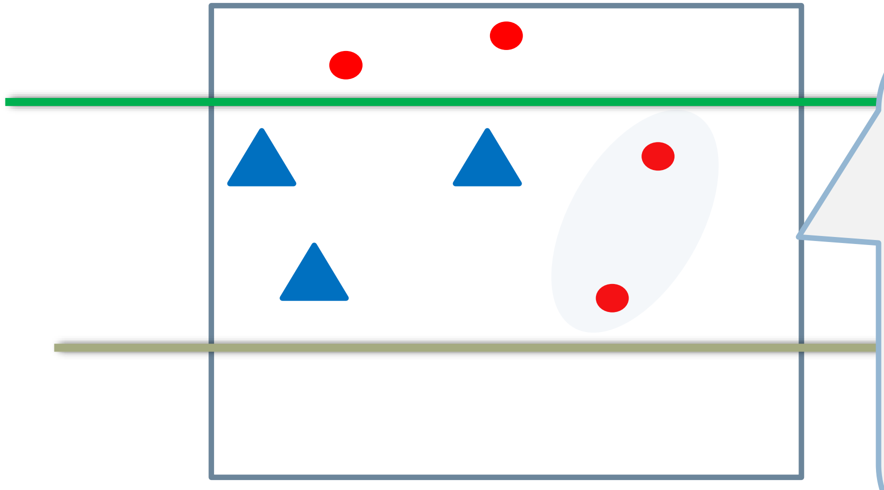


- Model M1 is built, anything above the line is – and below the line is +
- 3 out of 10 are misclassified by the model M1
- These data points will be given more weight in the re-sampling step
- We may miss out on some of the correctly classified records

Data	1	2	3	4	5	6	7	8	9	10
Class	-	-	+	+	-	+	-	-	+	+
Predicted Class M1	-	-	-	-	-	-	-	-	+	+
M1 Result	✔	✔	✗	✗	✔	✗	✔	✔	✔	✔

Boosting illustration

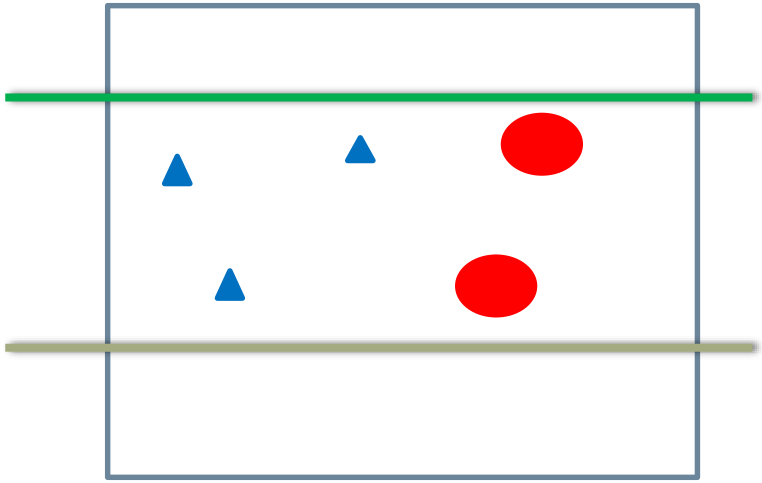


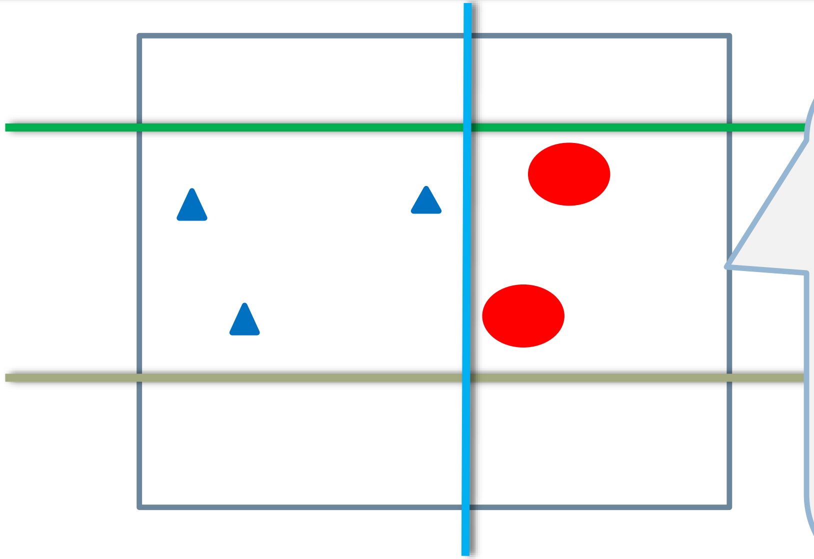


- The misclassified points 3,4,& 6 have appeared more often than others in this weighted sample.
- The sample points 9,10 didn't appear
- M2 is built on this data. Anything above the line is – and below the line is +
- M2 is classifying the points 5 & 7 incorrectly.
- They will be given more weight in the next sample

Weighted Sample1	1	2	3	4	5	6	7	4	3	6
Class	-	-	+	+	-	+	-	+	+	+
Predicted Class M2	-	-	+	+	+	+	+	+	+	+
M2 Result	✓	✓	✓	✓	✗	✓	✗	✓	✓	✓

Boosting illustration

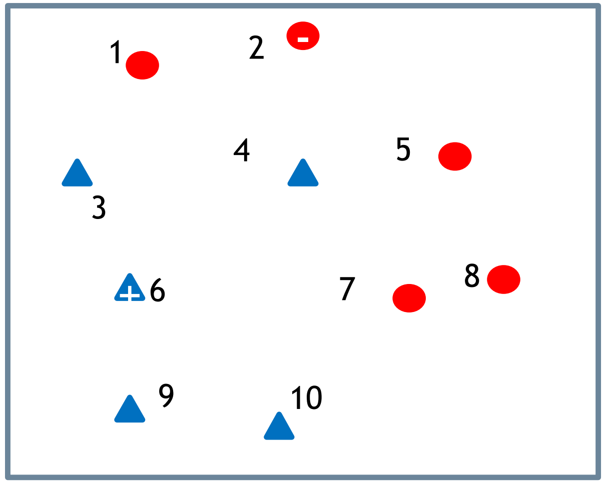


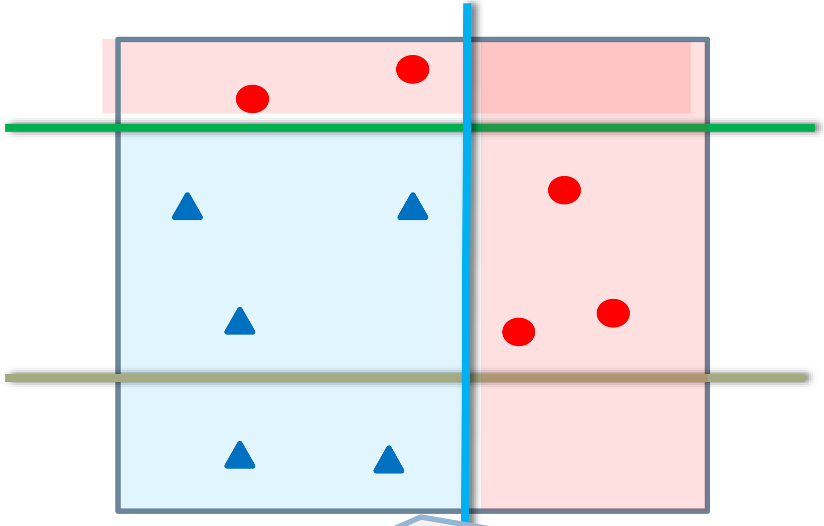


- The misclassified points 5 & 7 have appeared more often than others in this weighted sample.
- M3 is built on this data. Anything above the line is – and below the line is +
- M3 is now classifying everything correctly

Weighted Sample2	6	5	3	4	5	6	7	7	5	7
Class	+	-	+	+	-	+	-	-	-	-
Predicted Class M3	+	-	+	+	-	+	-	-	-	-
M3 Result	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔

Boosting illustration





- The final model now will be picked on weighted Votes.
- For a given data point more than 2 models seam to be indicating the right class.
- For example take point 6, it is classified as – by M1, + by M2 and + by M3, final result will be +
- Similarly take a point 2, it will be classified as –by M1, -by M2 and + by M3, final result will be -
- So the final weighted combination of three models predictions will yield in accurate perdition.

statinfer

Theory behind Boosting Algorithm

Theory behind Boosting Algorithm

- Take the dataset Build a classifier C_m and find the error
- Calculate error rate of the classifier
  - Error rate of ε_m
    - = $\sum w_i I(y_i \neq C_m(x))$ / $\sum w_i$
    - =Sum of misclassification weight / sum of sample weights
- Calculate an intermediate factor called α. It analogous to accuracy rate of the model. It will be later used in weight updating. It is derived from error
  - $\alpha_m = \log((1-\varepsilon_m)/\varepsilon_m)$

Theory behind Boosting Algorithm..contd

- Update weights of each record in the sample using the α factor. The indicator function will make sure that the misclassifications are given more weight
  - For i=1,2….N
    - $w_{i+1} = w_i e^{\alpha_m I(y_i \neq C_m(x))}$
    - Renormalize so that sum of weights is 1
- Repeat this model building and weight update process until we have no misclassification
- Final collation is done by voting from all the models. While taking the votes, each model is weighted by the accuracy factor α
  - $C = sign(\sum \alpha_i C_i(x))$

Two-Class Boosted Decision Tree

- A boosted decision tree is an ensemble learning method in which the second tree corrects for the errors of the first tree, the third tree corrects for the errors of the first and second trees, and so forth
- Predictions are based on the entire ensemble of trees together that makes the prediction
- when properly configured, boosted decision trees are the easiest methods to get top performance on a wide variety of machine learning tasks
- They are more memory-intensive learners, current implementation holds everything in memory
- Boosted decision tree model might not be able to process the very large datasets

Steps - Decision Tree Building

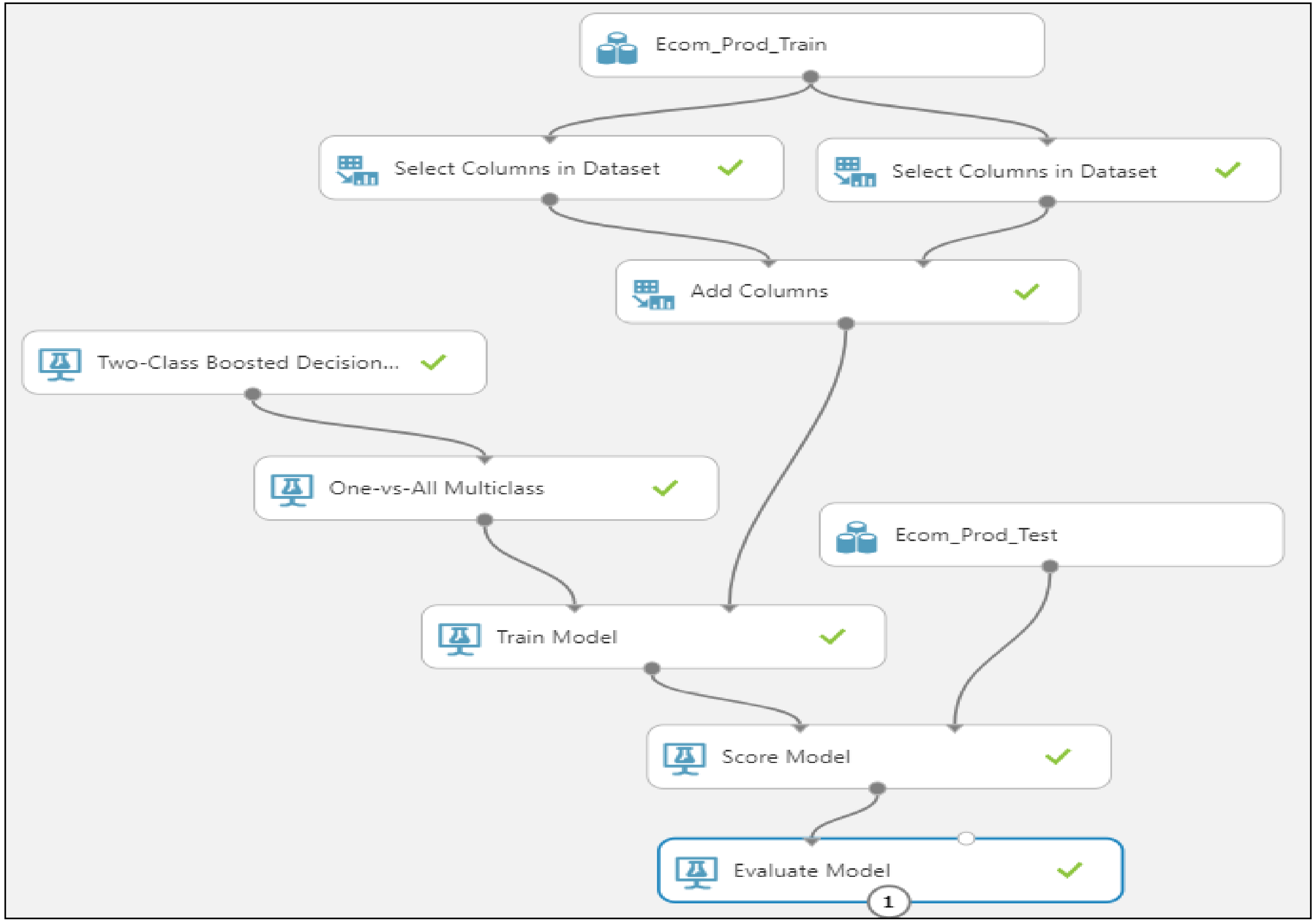
- The Decision Tree has the following parameters:

Parameters	Description
Create trainer mode	If you are sure about what to be specified for the parameters then select **Single Parameter,** else select **Parameter Range** which allows you to select multiple values and fin the optimal parameters using **Tune Model HyperParameters.**
Maximum number of leaves per tree	This indicates the maximum number of the last level nodes. Should be choose optimal, higher values may lead to overfitting and longer training time, lower values may underfit the data.
Minimum number of samples per leaf node	This indicates the minimum number of observations that a terminal node should contain for an applied split condition
Learning rate	This takes a value between 0 to 1 which defines the step size. This determines how fast or slow the decisions are taken to achieve the optimal solution, if small it takes longer time, if large it may overshoot the optimal solution.
Number of trees constructed	This indicates the number of decision trees that should be created. Since we are not using ensemble modal choose 1.
Random number seed	Type an non-negative integer value if you want the same data to be used at each run.
Allow unknown categorical levels	If checked, the test data can contain variables that are not in the training. These variables does not affect the prediction.

Steps - Decision Tree Building

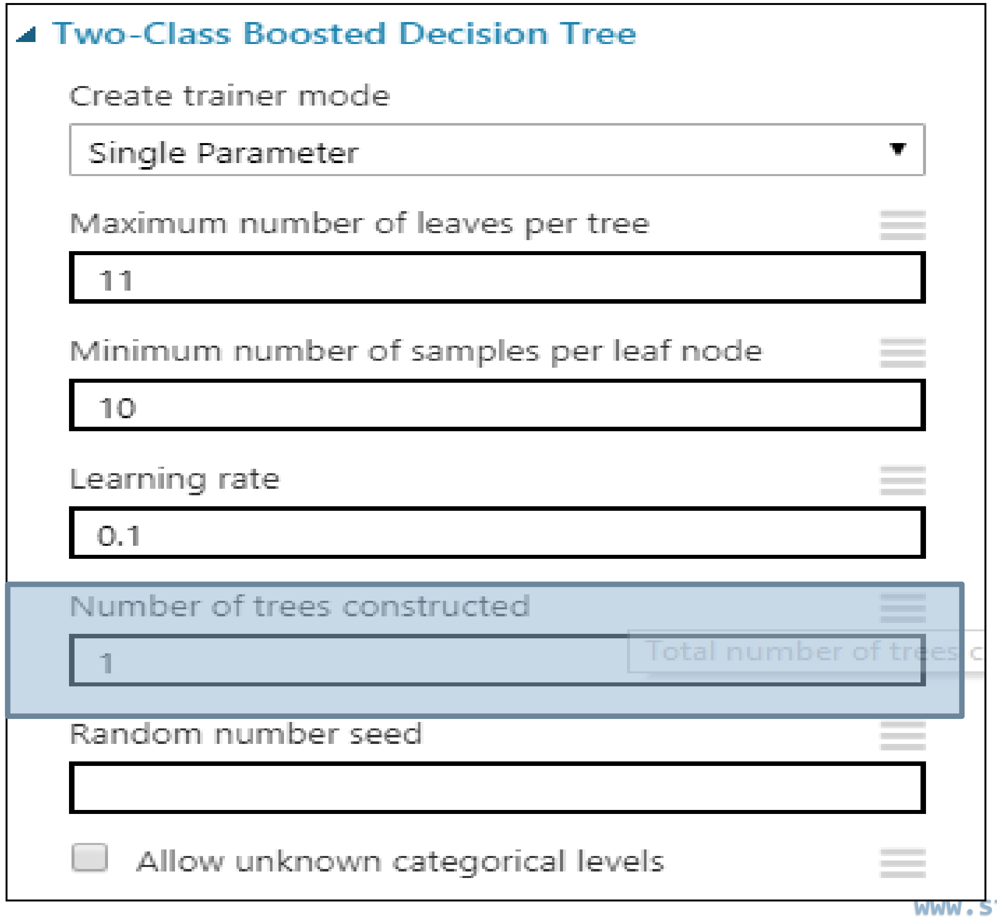
- Drag and drop the Dataset into the canvas
- Drag and drop **Two-Class Boosted Decision Tree, Train Model, Score Model** and **Evaluate Model**
- Connect **Two-Class Boosted Decision Tree** to the first input of **Train Model** and **Dataset** to the Second input of **Train Model**
- Connect the output of **Train Model** first input of **Score Model** and **Dataset** to the Second input of **Score Model**
- Connect the output of **Score Model** to the input of **Evaluate Model**

Steps - Decision Tree Building(Without Boosting)

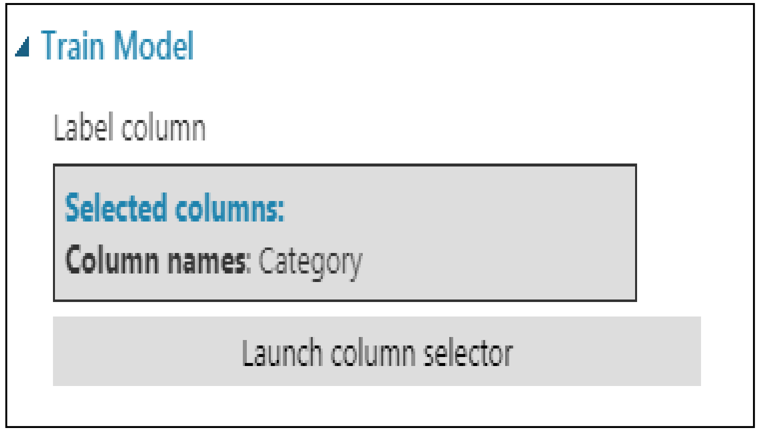


Steps - Decision Tree Building(Without Boosting)

Properties: Decision Tree(Without Boosting)



Properties: Train Model



statinfer

Steps - Decision Tree Building(Without Boosting)

rows 11756 columns 11

view as

Category	Scored Probabilities for Class "Accessories"	Scored Probabilities for Class "Appliances"	Scored Probabilities for Class "Camara"	Scored Probabilities for Class "Ipod"	Scored Probabilities for Class "Laptops"	Scored Probabilities for Class "Mobiles"	Scored Probabilities for Class "Personal_Care"	Scored Probabilities for Class "Tablets"	Scored Probabilities for Class "TV"	Scored Labels
Mobiles	0.035559	0.604452	0.039445	0.005317	0.091811	0.038849	0.046592	0.098291	0.039683	Appliances
Mobiles	0.071532	0.098983	0.07935	0.010695	0.122967	0.063908	0.288875	0.183861	0.079829	Personal_Care
Mobiles	0.043305	0.480286	0.031343	0.004224	0.072952	0.030869	0.074968	0.230522	0.031532	Appliances
Mobiles	0.085562	0.118396	0.094913	0.012793	0.147084	0.093479	0.112111	0.219921	0.115741	Tablets
Mobiles	0.021192	0.36023	0.023508	0.003168	0.232303	0.018933	0.056228	0.260789	0.02365	Appliances
Mobiles	0.030699	0.04248	0.034054	0.00459	0.052773	0.682012	0.040225	0.078907	0.03426	Mobiles
Mobiles	0.077231	0.106869	0.085672	0.011547	0.132764	0.200022	0.101195	0.198509	0.08619	Mobiles
Mobiles	0.033871	0.03058	0.024515	0.003304	0.03799	0.024144	0.76413	0.056803	0.024663	Personal_Care
Mobiles	0.053784	0.469262	0.059663	0.008042	0.092458	0.048052	0.070473	0.138243	0.060023	Appliances
Mobiles	0.088876	0.122982	0.09859	0.013288	0.152782	0.079404	0.116453	0.22844	0.099185	Tablets
Mobiles	0.019828	0.027437	0.021995	0.002965	0.034085	0.017715	0.802883	0.050964	0.022128	Personal_Care
Mobiles	0.087331	0.120844	0.096876	0.013057	0.150125	0.095411	0.114428	0.224468	0.09746	Tablets
Mobiles	0.028065	0.477065	0.031132	0.004196	0.124373	0.072686	0.074465	0.156699	0.03132	Appliances
Mobiles	0.028515	0.039458	0.705112	0.004263	0.049019	0.031154	0.037363	0.073293	0.031823	Camara
Mobiles	0.020696	0.028639	0.652448	0.003094	0.053437	0.022612	0.027118	0.168858	0.023097	Camara
Mobiles	0.021396	0.029607	0.023735	0.003199	0.121677	0.019116	0.028035	0.702286	0.050948	Tablets
Mobiles	0.088876	0.122982	0.09859	0.013288	0.152782	0.079404	0.116453	0.22844	0.099185	Tablets
Mobiles	0.021152	0.029269	0.785468	0.003163	0.036361	0.018898	0.027715	0.054368	0.023606	Camara
Mobiles	0.130061	0.117423	0.094133	0.012688	0.145875	0.075814	0.111189	0.218114	0.094702	Tablets

Steps - Decision Tree Building(Without Boosting)

Accuracy

Metrics

Overall accuracy	0.686713
Average accuracy	0.930381
Micro-averaged precision	0.686713
Macro-averaged precision	0.717103
Micro-averaged recall	0.686713
Macro-averaged recall	0.564385

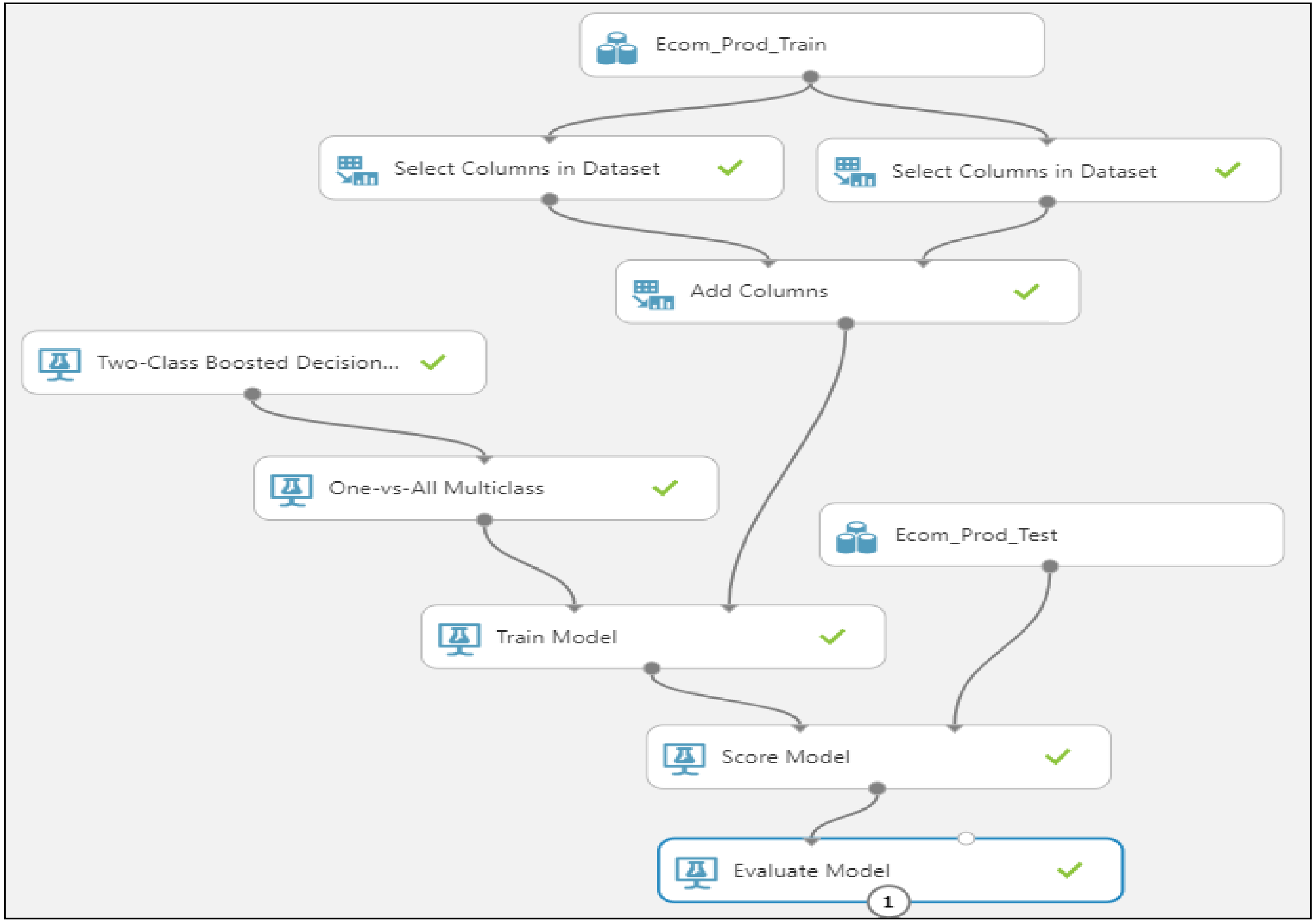
Steps - Decision Tree Building(Without Boosting)

Confusion Matrix

Predicted

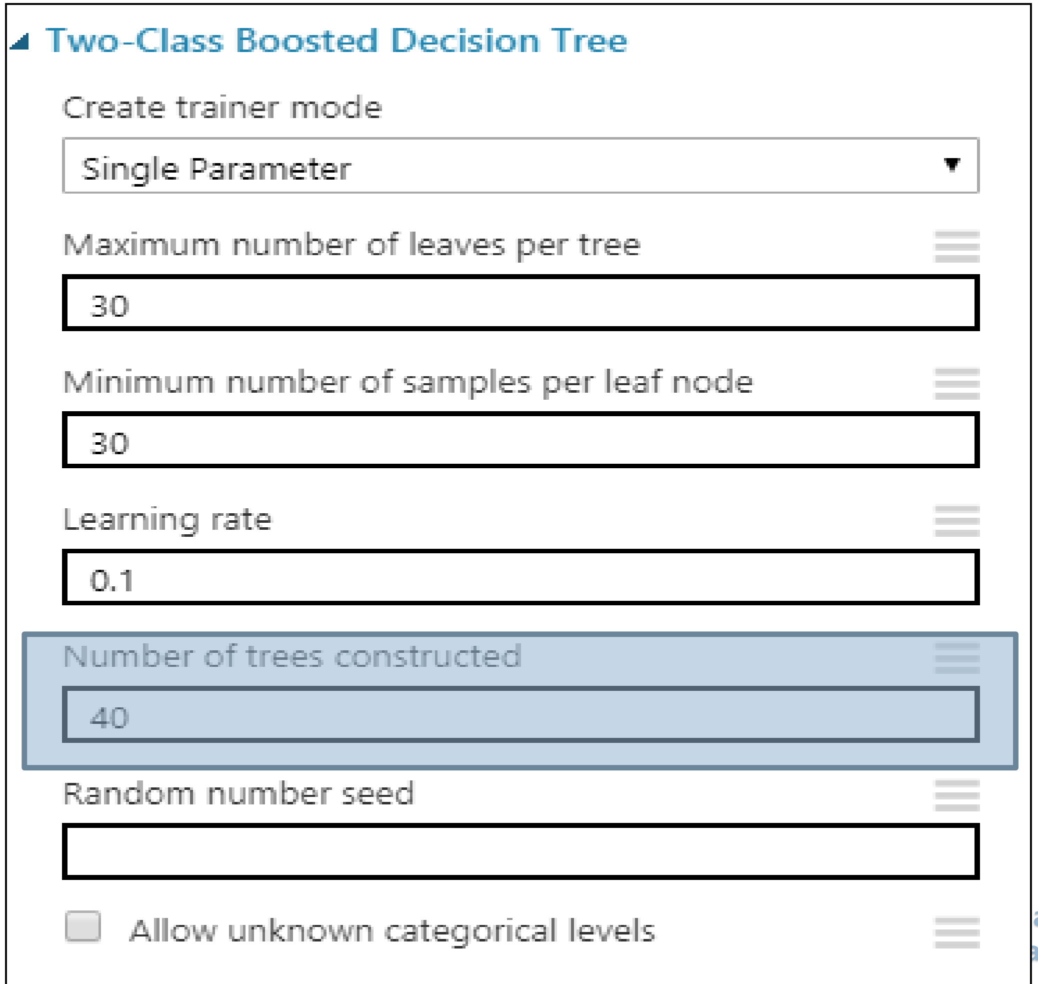
Actual	Accessor...	Appliances	Camara	Ipod	Laptops	Mobiles	Personal...	Tablets	TV
Accessor...	40.2%	6.0%	0.6%	0.2%	4.0%	1.8%	8.2%	38.4%	0.6%
Appliances	0.7%	74.8%	2.2%	0.2%	0.6%	1.7%	5.4%	14.4%	0.1%
Camara	0.1%	4.7%	67.8%	0.3%	0.1%	1.0%	7.5%	18.4%	0.1%
Ipod		0.2%		91.0%	1.3%		1.0%	6.5%	
Laptops	1.0%	0.7%	1.1%		27.2%		0.3%	69.3%	0.5%
Mobiles	0.8%	11.1%	16.7%		0.8%	19.9%	12.4%	38.3%	
Personal...	0.9%	1.7%	1.2%		0.4%	0.3%	89.6%	5.7%	0.2%
Tablets	0.3%	0.6%	0.8%	0.3%	12.7%	0.1%	0.3%	84.3%	0.5%
TV	1.4%	0.2%	0.2%	0.8%	13.9%		5.7%	64.6%	13.2%

Steps - Decision Tree Building(Boosting)

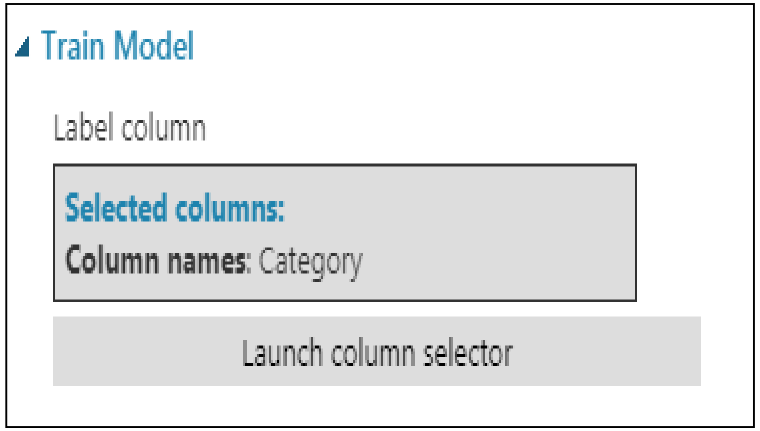


Steps - Decision Tree Building(Boosting)

Properties: Decision Tree(Boosting)



Properties: Train Model



statinfer

Steps - Decision Tree Building(Boosting)

rows	columns
11756	11

view as

Category	Scored Probabilities for Class "Accessories"	Scored Probabilities for Class "Appliances"	Scored Probabilities for Class "Camara"	Scored Probabilities for Class "Ipod"	Scored Probabilities for Class "Laptops"	Scored Probabilities for Class "Mobiles"	Scored Probabilities for Class "Personal_Care"	Scored Probabilities for Class "Tablets"	Scored Probabilities for Class "TV"	Scored Labels
Mobiles	0.008562	0.062554	0.025065	0.000938	0.009566	0.797035	0.081422	0.011814	0.003044	Mobiles
Mobiles	0.009553	0.008258	0.012927	0.001193	0.010559	0.01016	0.900951	0.038533	0.007867	Personal_Care
Mobiles	0.041743	0.19449	0.025257	0.002115	0.02624	0.19159	0.473065	0.037331	0.008169	Personal_Care
Mobiles	0.10227	0.24232	0.051478	0.007129	0.055338	0.085885	0.192996	0.223409	0.039176	Appliances
Mobiles	0.006862	0.64519	0.009023	0.000811	0.055575	0.179118	0.062502	0.034768	0.006152	Appliances
Mobiles	0.005962	0.00982	0.258559	0.0007	0.004804	0.700399	0.010066	0.005762	0.003927	Mobiles
Mobiles	0.008667	0.359717	0.005644	0.000608	0.003844	0.60863	0.006825	0.004152	0.001914	Mobiles
Mobiles	0.242827	0.004313	0.013704	0.000821	0.01364	0.029266	0.680253	0.01119	0.003986	Personal_Care
Mobiles	0.044273	0.880845	0.014476	0.003857	0.011092	0.012618	0.011814	0.016084	0.004941	Appliances
Mobiles	0.008301	0.008031	0.007097	0.000902	0.004974	0.904405	0.053285	0.008136	0.004869	Mobiles
Mobiles	0.009813	0.005065	0.005148	0.001163	0.005079	0.010482	0.953365	0.004242	0.005641	Personal_Care
Mobiles	0.078072	0.609037	0.019888	0.002682	0.012919	0.151208	0.097347	0.02014	0.008706	Appliances
Mobiles	0.011822	0.69039	0.005015	0.001051	0.009977	0.061518	0.208154	0.008365	0.003709	Appliances
Mobiles	0.012287	0.017265	0.84783	0.001	0.007538	0.084492	0.015538	0.008978	0.005072	Camara
Mobiles	0.007419	0.005501	0.684002	0.000711	0.00821	0.253458	0.030693	0.007622	0.002384	Camara
Mobiles	0.007441	0.003749	0.004096	0.000813	0.206038	0.002723	0.005194	0.751268	0.018677	Tablets

Steps - Decision Tree Building(Boosting)

statinfer

Accuracy

Metrics

Overall accuracy	0.794403
Average accuracy	0.954312
Micro-averaged precision	0.794403
Macro-averaged precision	0.783649
Micro-averaged recall	0.794403
Macro-averaged recall	0.726485

Steps - Decision Tree Building(Boosting)

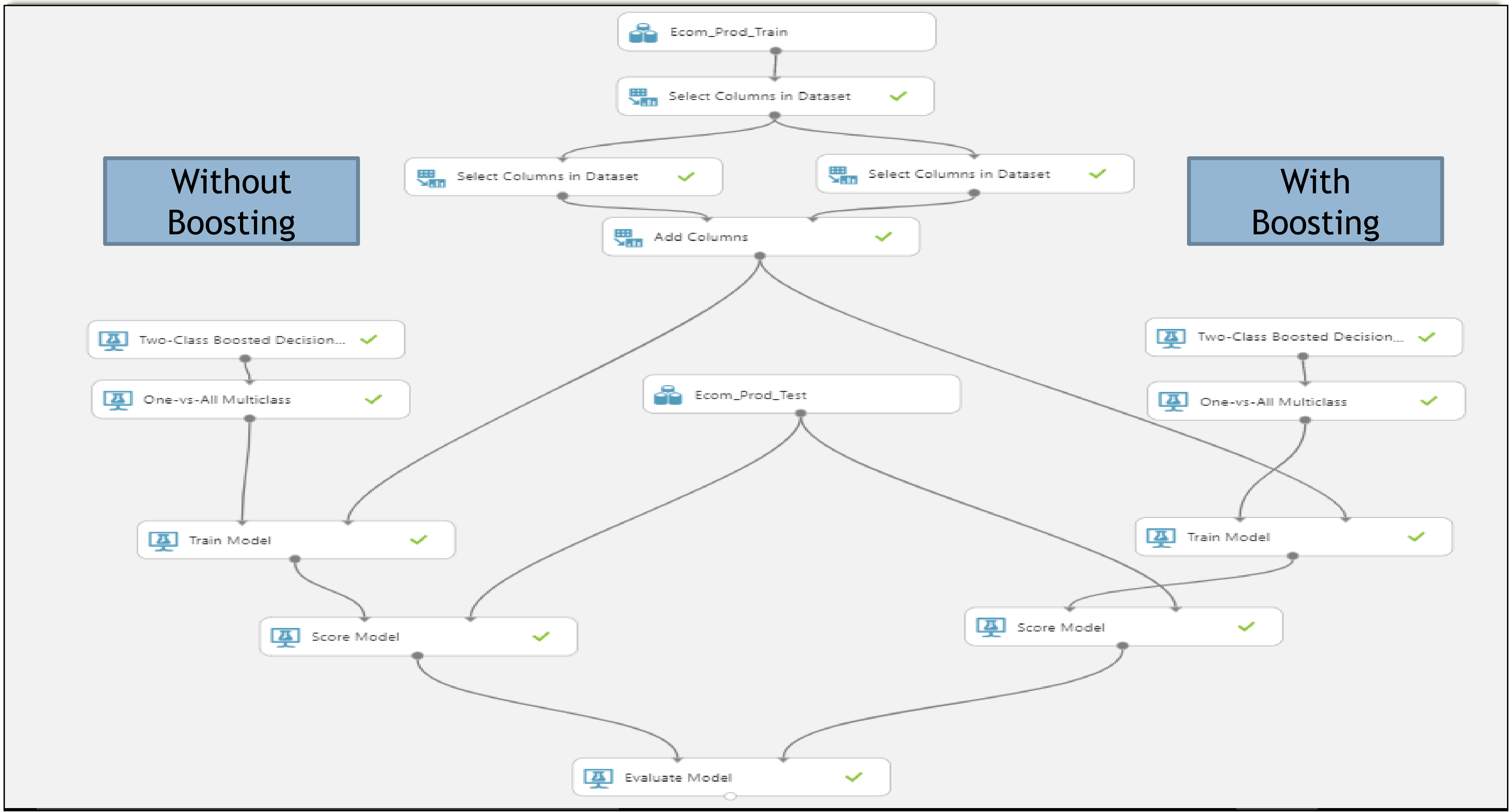
- <u>Confusion Matrix</u>

Predicted

Actual

	Accessor...	Appliances	Camara	Ipod	Laptops	Mobiles	Personal...	Tablets	TV
Accessor...	65.3%	6.0%	0.4%	0.2%	7.4%	2.4%	7.6%	9.2%	1.4%
Appliances	1.0%	91.2%	1.7%	0.1%	0.2%	1.3%	3.2%	1.4%	
Camara	0.3%	4.3%	85.4%	0.2%	0.1%	2.6%	4.1%	2.8%	0.1%
Ipod	0.8%	0.2%		95.4%		0.2%	0.4%	3.1%	
Laptops	2.0%	0.3%	0.2%		48.2%	0.1%	0.2%	46.5%	2.6%
Mobiles	0.8%	17.3%	19.7%		1.6%	46.6%	8.4%	5.7%	
Personal...	1.4%	1.4%	1.3%		0.2%	0.5%	94.0%	1.2%	0.1%
Tablets	0.8%	0.4%	0.2%	0.3%	12.6%		0.4%	83.8%	1.4%
TV	2.0%			0.6%	10.8%		4.7%	37.9%	44.0%

Steps - Decision Tree Building(Combined)



statinfer

Thank you

Part 12/12: Cluster Analysis

Cluster Analysis

Statinfer.com

Contents

- Introduction to Segmentation & Cluster analysis
- Applications of Cluster Analysis
- Types of Clusters
- Similarity measure
- K-Means clustering
- The Algorithm
- Building clusters
- Deciding the cluster numbers
- Working with non-numerical data

What is the need of segmentation?

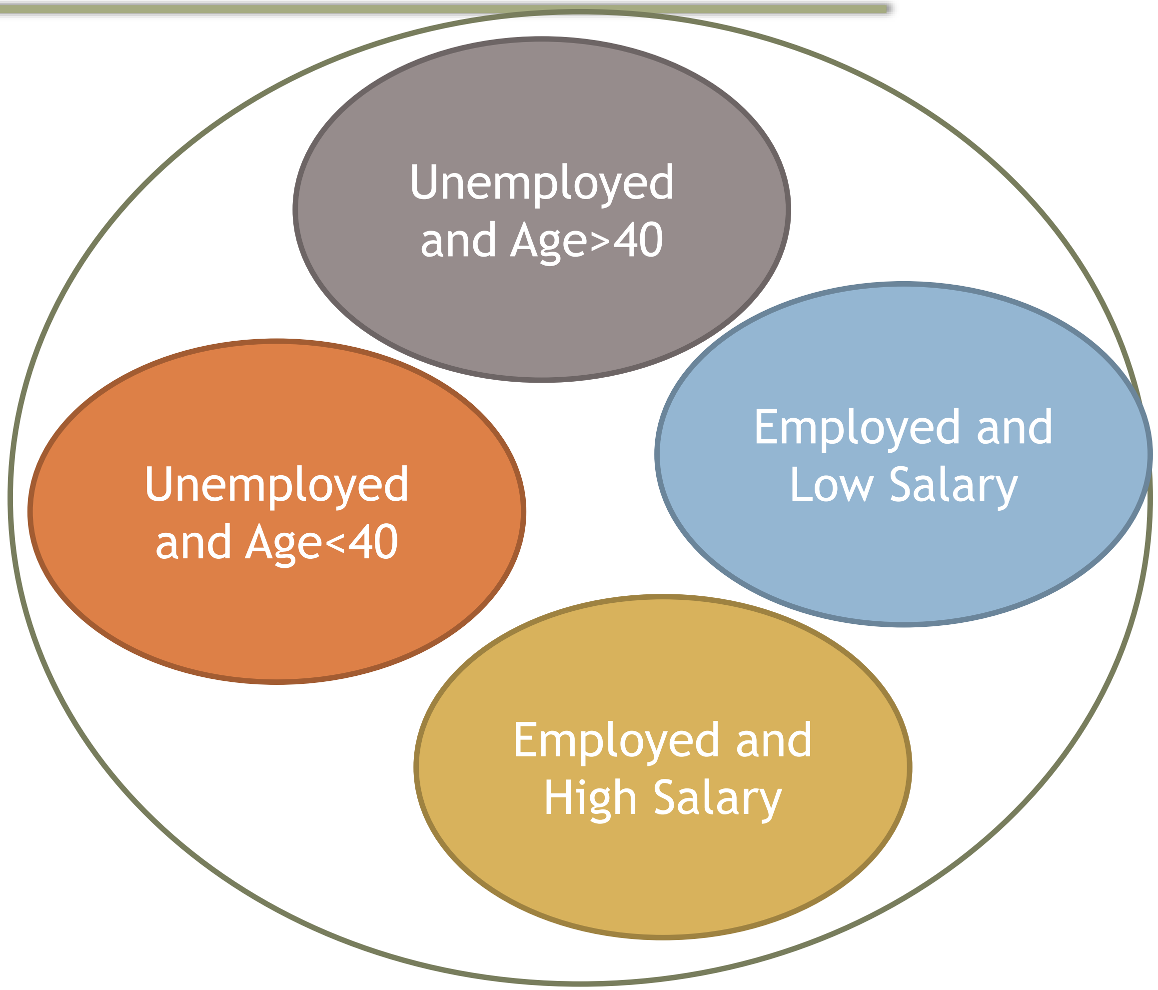
Problem:

- 10,000 Customers - we know their age, city name, income, employment status, designation
- You have to sell 100 smart phones(each costs $1000) to the people in this group. You have maximum of 7 days
- If you start giving demos to each individual, 10,000 demos will take more than one year. How will you sell maximum number of phones by giving minimum number of demos?

What is the need of segmentation?

Solution

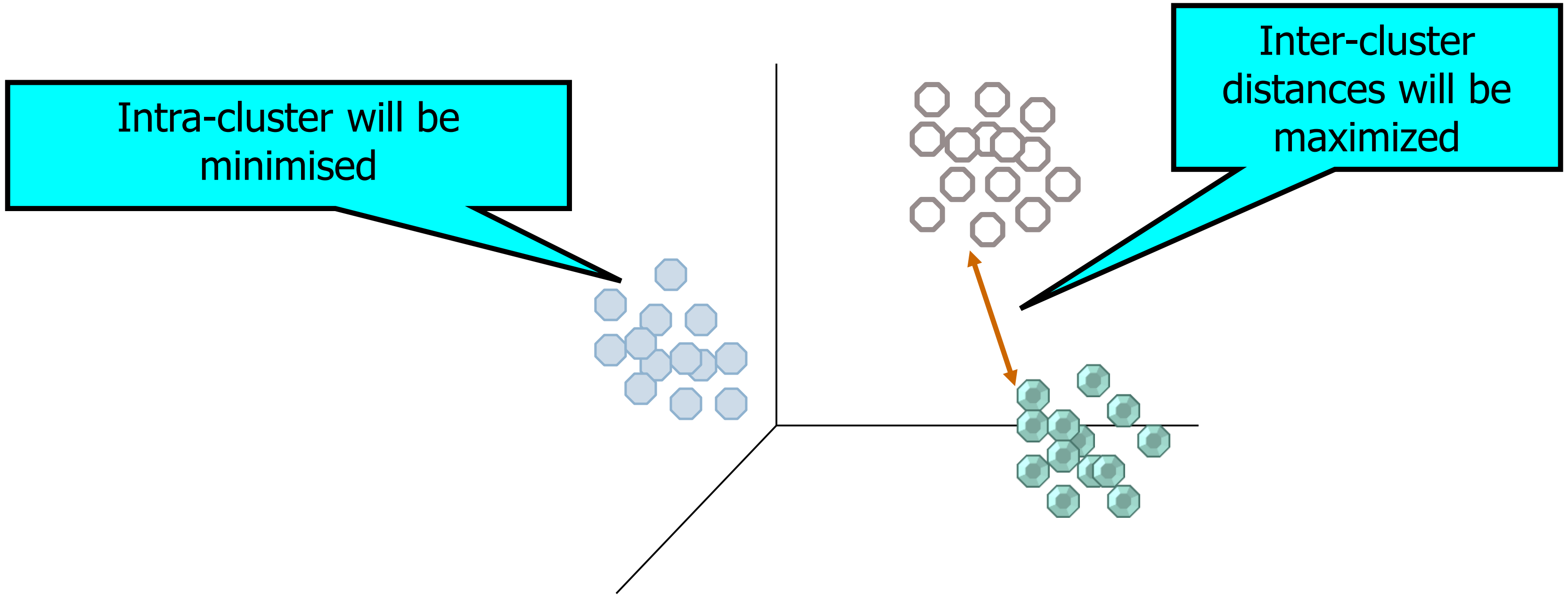
- Partition the whole population into groups
- Same type of customers should be clubbed together
- Dis-similar customers should not be in the same group



Segmentation and Cluster Analysis

- Cluster is a group of similar objects (cases, points, observations, examples, members, customers, patients, locations, etc)
- Finding the groups of cases/observations/ objects in the population such that the objects are
- Homogeneous within the group (high intra-class similarity)
- Heterogeneous between the groups(low inter-class similarity)

Segmentation and Cluster Analysis



Applications of Cluster Analysis

- **Market Segmentation:** Grouping people (with the willingness, purchasing power, and the authority to buy) according to their similarity
- **Sales Segmentation:** Clustering can tell you what types of customers buy what products
- **Credit Risk:** Segmentation of customers based on their credit history
- **Operations:** High performer segmentation & promotions based on person’s performance
- **Insurance:** Identifying groups of motor insurance policy holders with a high average claim cost.

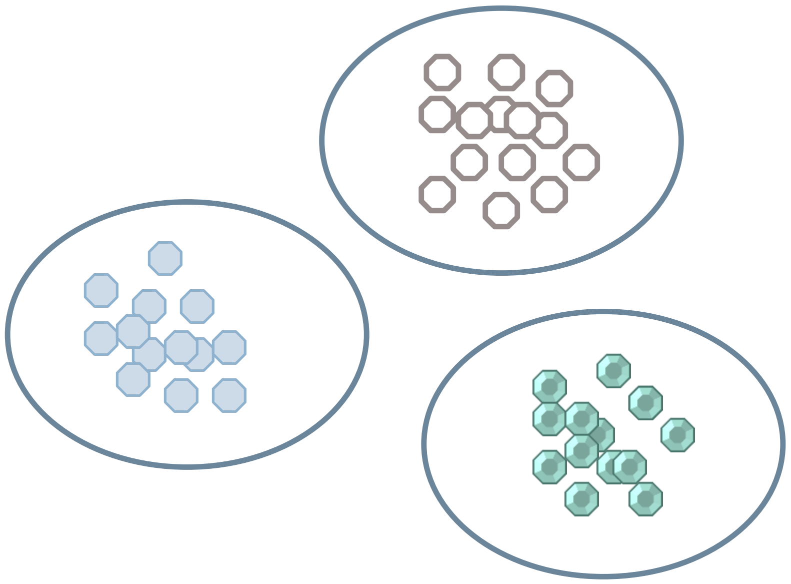
Types of Clusters

Two most widely used methods

- Partitional clustering or non-hierarchical
- Hierarchical clustering

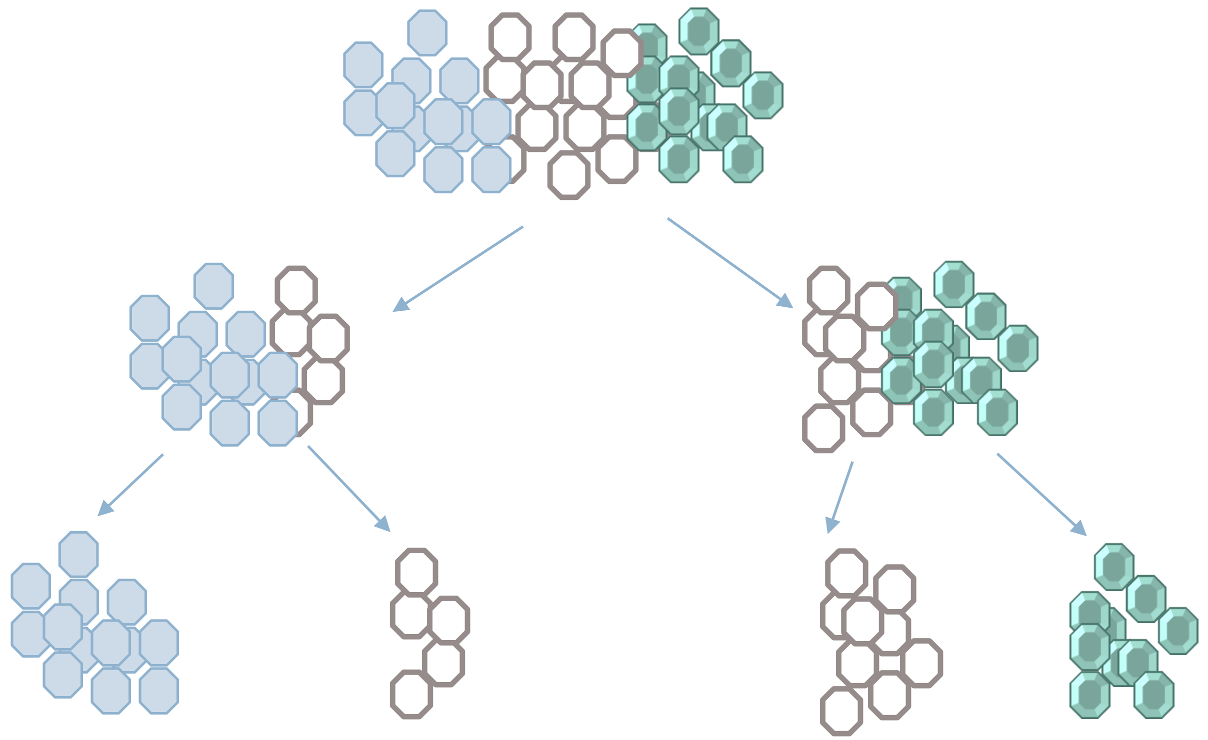
Partitional clustering or non-hierarchical

- A division of dataset into non-overlapping subsets (clusters)
- The non-hierarchical methods divide a dataset of N objects into M clusters.
- K-means clustering, a non-hierarchical technique, is the most commonly used one in business analytics



Hierarchical clustering

- Nested clusters
- A set of nested clusters organized as a hierarchical tree



statinfer

Dissimilarity & Similarity

Dissimilarity & Similarity

	Weight
Cust1	68
Cust2	72
Cust3	100

Which two customers are similar?

	Weight	Age
Cust1	68	25
Cust2	72	70
Cust3	100	28

Which two customers are similar now?

	Weight	Age	Income
Cust1	68	25	60,000
Cust2	72	70	9,000
Cust3	100	28	62,000

Which two customers are similar in this case?

Quantify dissimilarity -Distance measures

- To measure similarity between two observations a distance measure is needed. With a single variable, similarity is straightforward
- Example: income – two individuals are similar if their income level is similar and the level of dissimilarity increases as the income gap increases
- Multiple variables require an aggregate distance measure
- Many characteristics (e.g. income, age, consumption habits, family composition, owning a car, education level, job…), it becomes more difficult to define similarity with a single value
- The most known measure of distance is the Euclidean distance, which is the concept we use in everyday life for spatial coordinates.

Examples of distances

$$\sqrt{\sum_{k=1}^{n}\left(x_{ik}-x_{jk}\right)^2}$$ Euclidian Distance

$$\sum_{k=1}^{n}\left|x_{ik}-x_{jk}\right|$$ Manhattan distance

$$r\left(x_{ik}, x_{jk}\right)$$ Correlation -Similarity measure

$$\max_{k}\left|x_{ik}-x_{jk}\right|$$ Chebyshev distance

Other distance measures:

- Minkowski
- Mahalanobis
- maximum distance
- Cosine similarity
- Jacob's distance

Generalised distance measure

- Minkowski distance

$$D = \sqrt[q]{(|x_{i1} - x_{j1}|^q + |x_{i2} - x_{j2}|^q + \ldots + |x_{ik} - x_{jk}|^q)}$$

$(x_{i1}, x_{i2}, \ldots, x_{ik})$ $(x_{j1}, x_{j2}, \ldots, x_{jk})$ are two *k*-dimensional data points

- Substitute q=1 in the above formula, D is Manhattan distance
- Substitute q=2 in the above formula, D is Euclidian distance

Distance Matrix

Data matrix

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix}$$

Dissimilarity matrix

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ : & : & : & & \\ d(n,1) & d(n,2) & \ldots & \ldots & 0 \end{bmatrix}$$

Calculating the distance

	Weight
Cust1	68
Cust2	72
Cust3	100

- Cust1 vs Cust2 :- (68-72)= **4**
- Cust2 vs Cust3 :- (72-100) = **28**
- Cust3 vs Cust1 :- (100-68) =**32**

	Weight	Age
Cust1	68	25
Cust2	72	70
Cust3	100	28

- Cust1 vs Cust2 :- sqrt((68-72)^2 + (25-70)^2)=**44.9**
- Cust2 vs Cust3 :- **50.54**
- Cust3 vs Cust1 :- **32.14**

LAB: Calculation of distance

- Import the data `Data:` `“./Credit_Score_Expenses/Credit_Score_Expenses.csv”`
- Calculate the pairwise distances
- Which two customers are close to each other?
- Which two customers are very dis-similar?

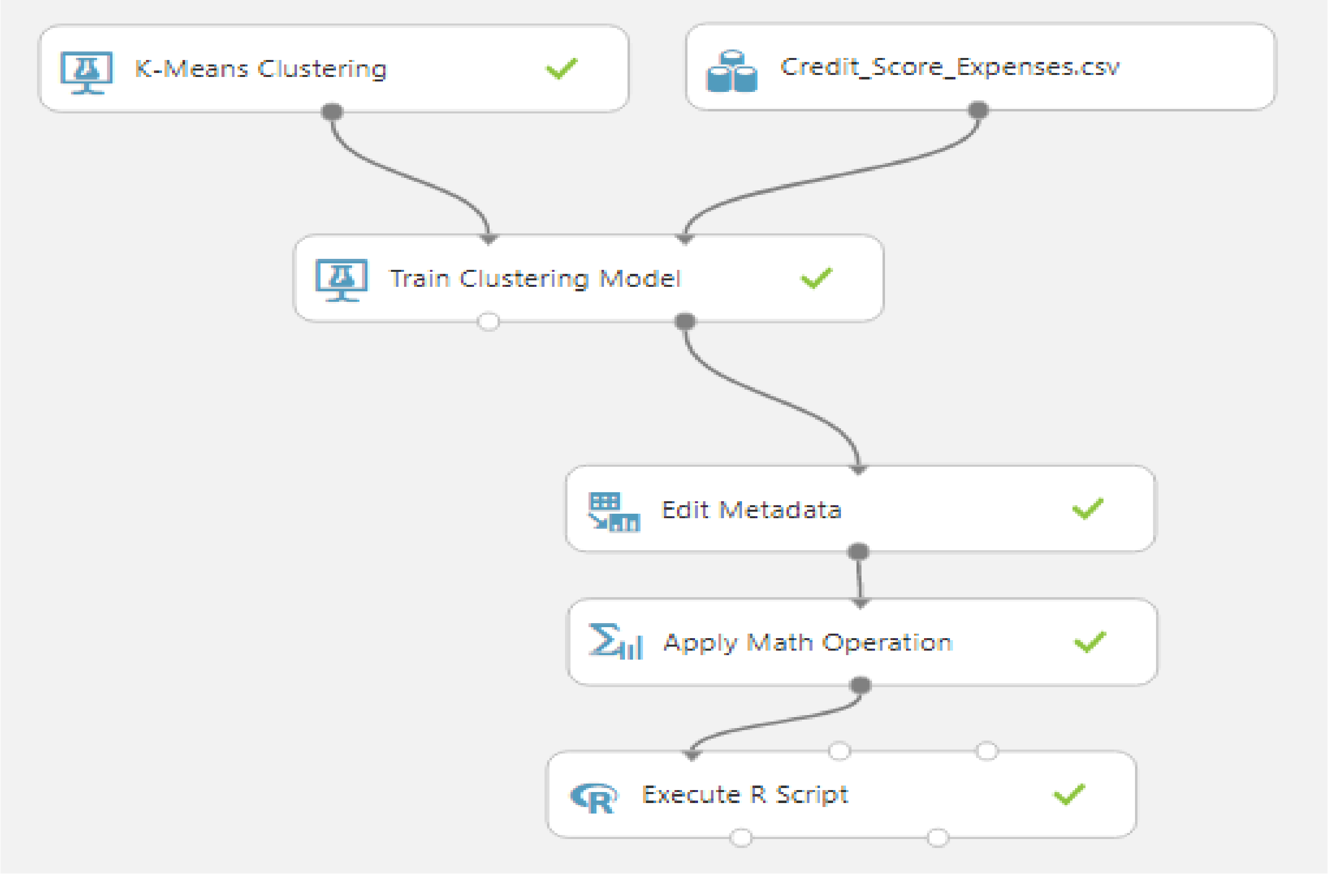
Steps - Calculation of distance

- Drag and drop the **Dataset** into the canvas
- Drag and drop **K - Means Clustering** into the canvas
- Drag and drop **Train Clustering Model**, connect **K - Means Clustering** to the first input and **Dataset** to the second input
- Drag and drop **Edit Metadata**, connect the second output of **Train Clustering Model** to the input of it
- Drag and drop **Apply Math Operation**, connect **Edit Metadata** to it
- Drag and drop **Execute R Script**, connect **Apply Math Operation** to it
- Click on run visualize the first output of **Execute R Script**

Note: Select the properties for **K - Means Clustering, Train Clustering Model, Edit Metadata, Apply Math Operation, Execute R Script** before run

Steps - Calculation of distance

Fig1: Clustering – Credit Score Expenses



Steps - Calculation of distance

Fig2: Properties - K-Means Clustering

Properties   Project

K-Means Clustering

Create trainer mode

Single Parameter

Number of Centroids

5

Initialization

K-Means++

Random number seed

Metric

Euclidean

Iterations

100

Assign Label Mode

Ignore label column

Fig3: Properties - Train Clustering Model

Properties   Project

Train Clustering Model

Column Set

Selected columns:
All columns
Exclude column names: Cust_id

Launch column selector

Check for Append or Uncheck for Result O...

Steps - Calculation of distance

Fig4: Properties - Edit Metadata

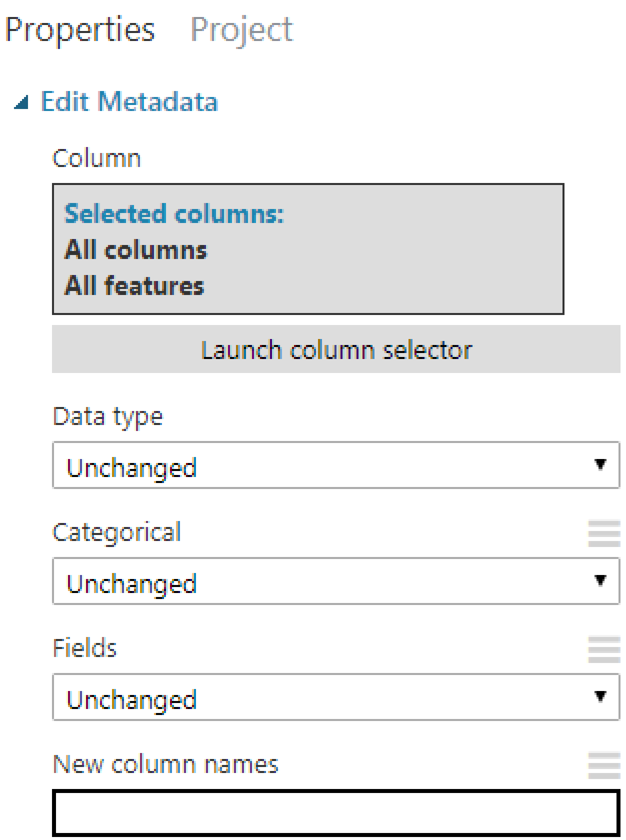
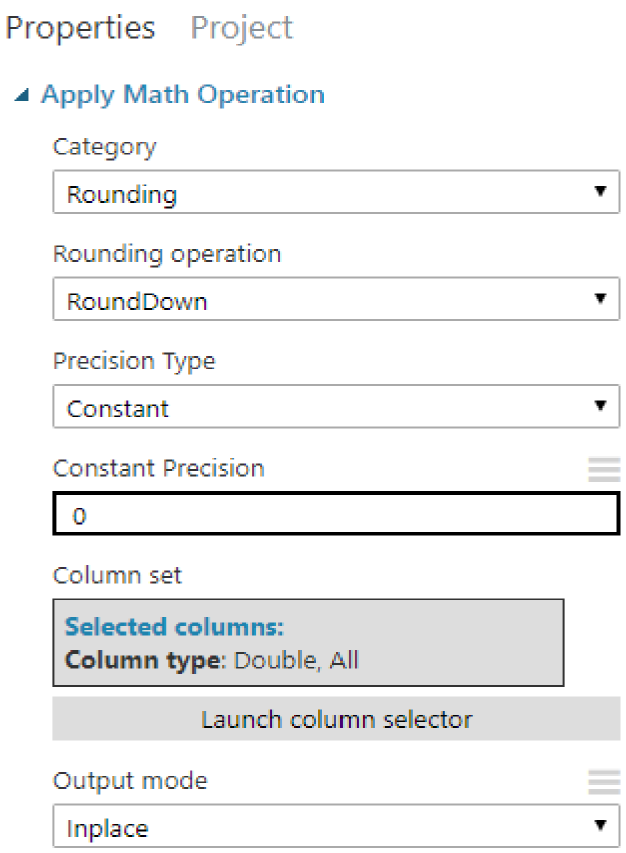


Fig5: Properties - Apply Math Operation



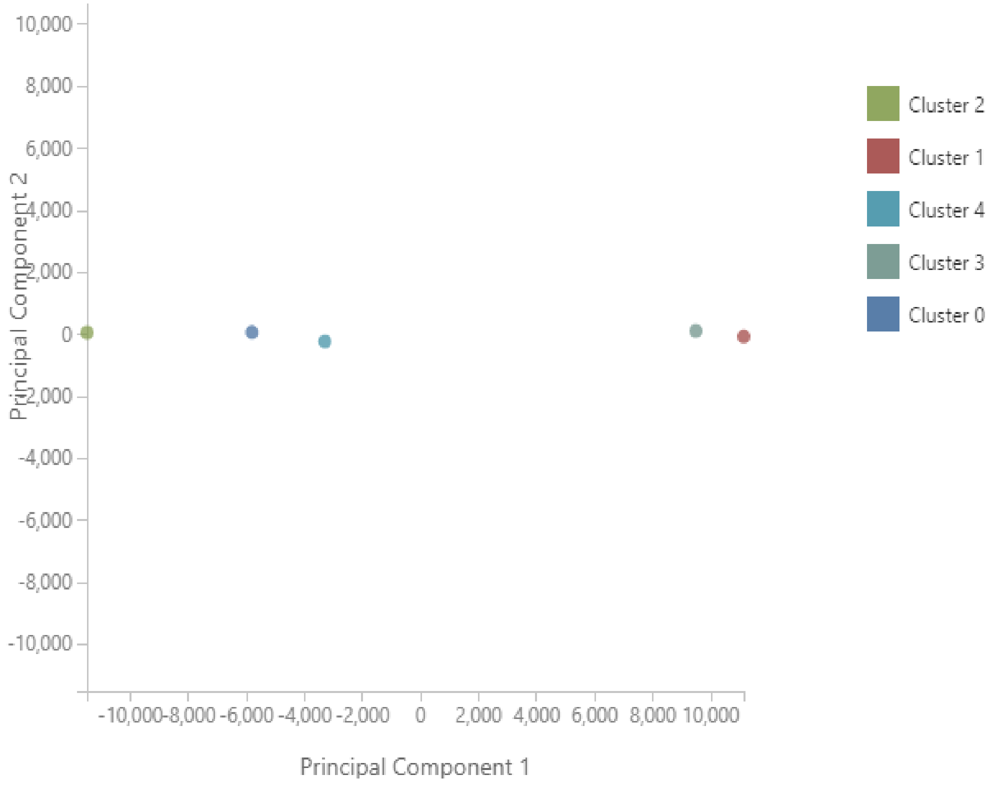
Steps - Calculation of distance

Fig6: R-Script - Sorting

R Script

```
dataset1 <- maml.mapInputPort(1) # class: data.frame

Order<-dataset1[order(dataset1$Assignments),]

maml.mapOutputPort("Order");
```

Fig7: Clusters



Steps - Calculation of distance

Fig8: Clustering – Distance Matrix

Assignments	DistancesToClusterCenter no.0	DistancesToClusterCenter no.1	DistancesToClusterCenter no.2
0	0	16950	5678
1	16950	0	22628
2	5678	22628	0
3	15310	1648	20988
4	2528	14439	8193

statinfer

Clustering algorithms

Clustering algorithms

- k-means clustering algorithm
- Fuzzy c-means clustering algorithm
- Hierarchical clustering algorithm
- Gaussian(EM) clustering algorithm
- Quality Threshold (QT) clustering algorithm
- MST based clustering algorithm
- Density based clustering algorithm
- kernel k-means clustering algorithm

- **NOTE**: As of now only K-Means clustering is available in Azure, If you want to use other types of clustering write an R-Script using Execute R-Script module

K -Means Clustering – Algorithm

1. The number *k* of clusters is fixed
2. An initial set of *k* *“seeds” (aggregation centres)* is provided
    1. First *k* elements
    2. Other seeds (randomly selected or explicitly defined)
3. Given a certain fixed threshold, all units are assigned to the nearest cluster seed
4. New seeds are computed
5. Go back to step 3 until no reclassification is necessary

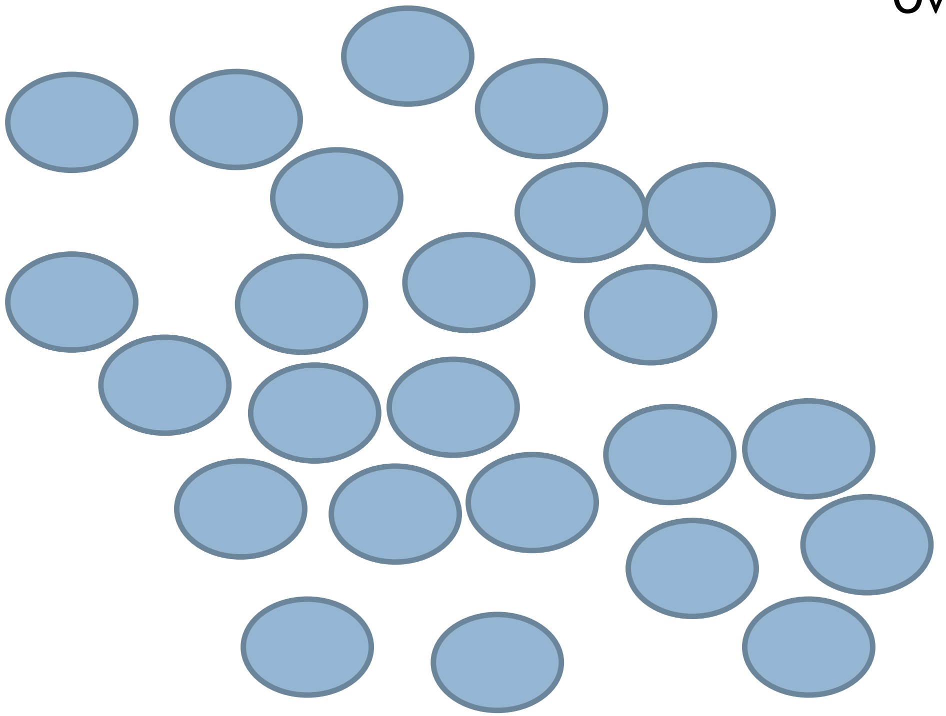
K -Means Clustering – Algorithm

In simple terms

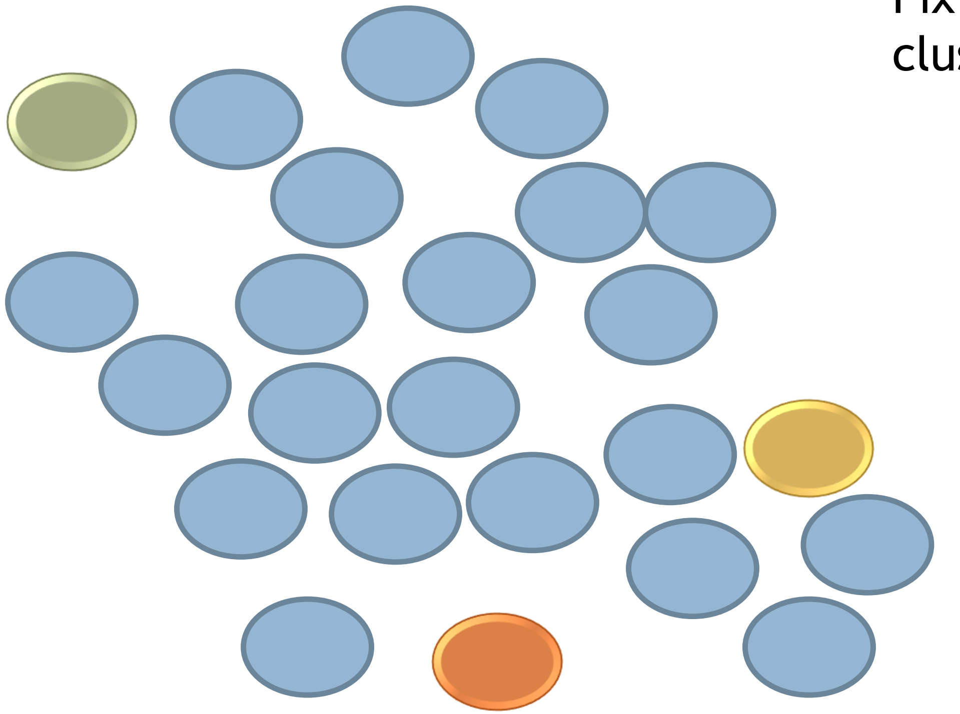
- Initialize k cluster centres
  - Do
    - Assignment step: Assign each data point to its closest cluster center
    - Re-estimation step: Re-compute cluster centers
    - While (there are still changes in the cluster centers)

K-Means clustering

Overall population

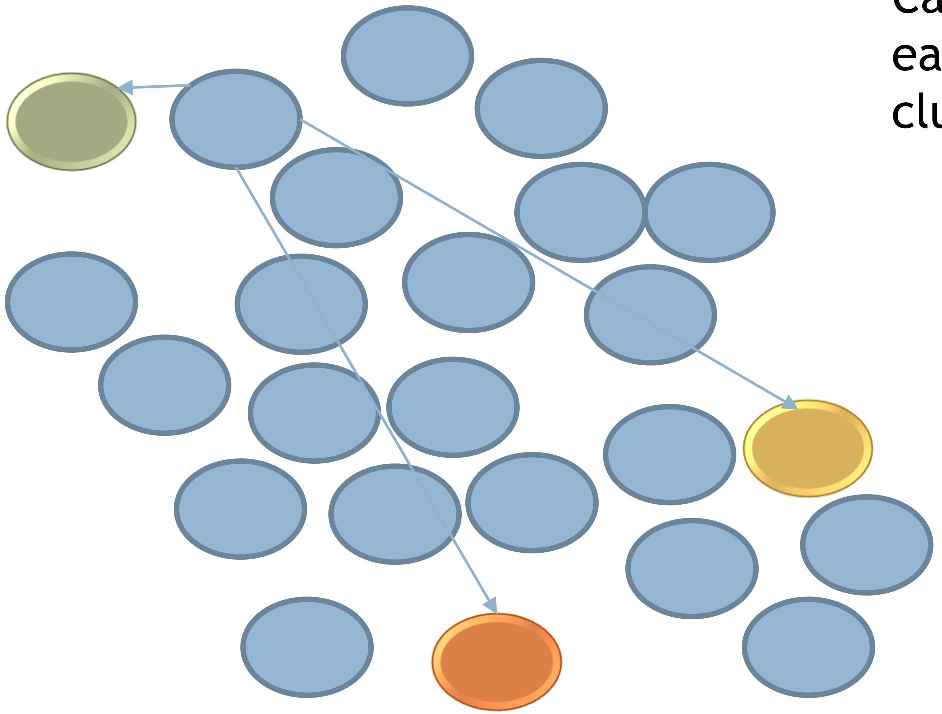


K-Means clustering



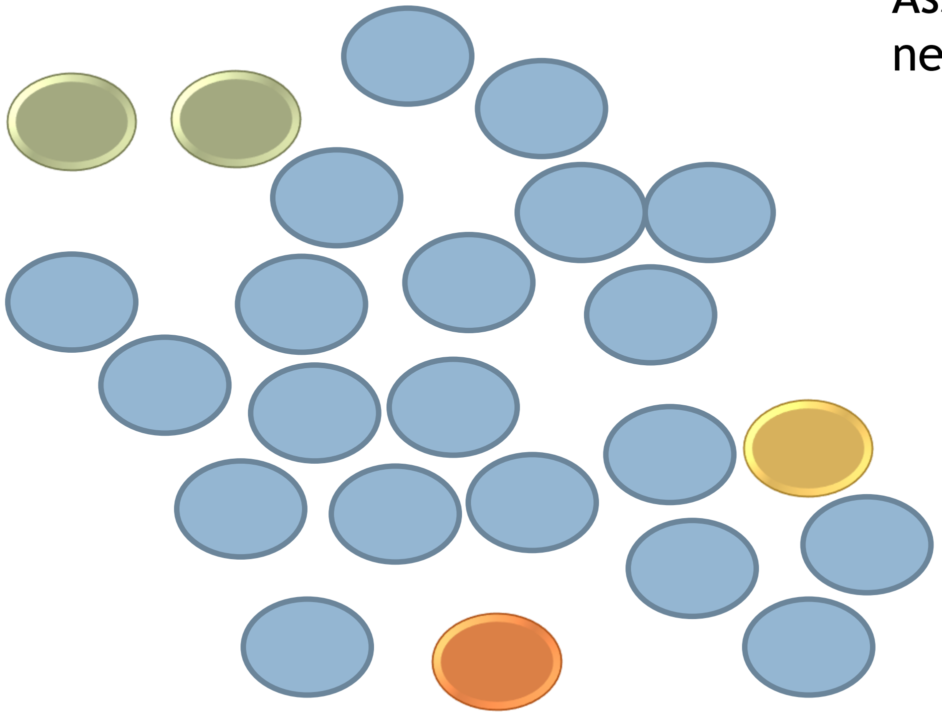
Fix the number of clusters

K-Means clustering



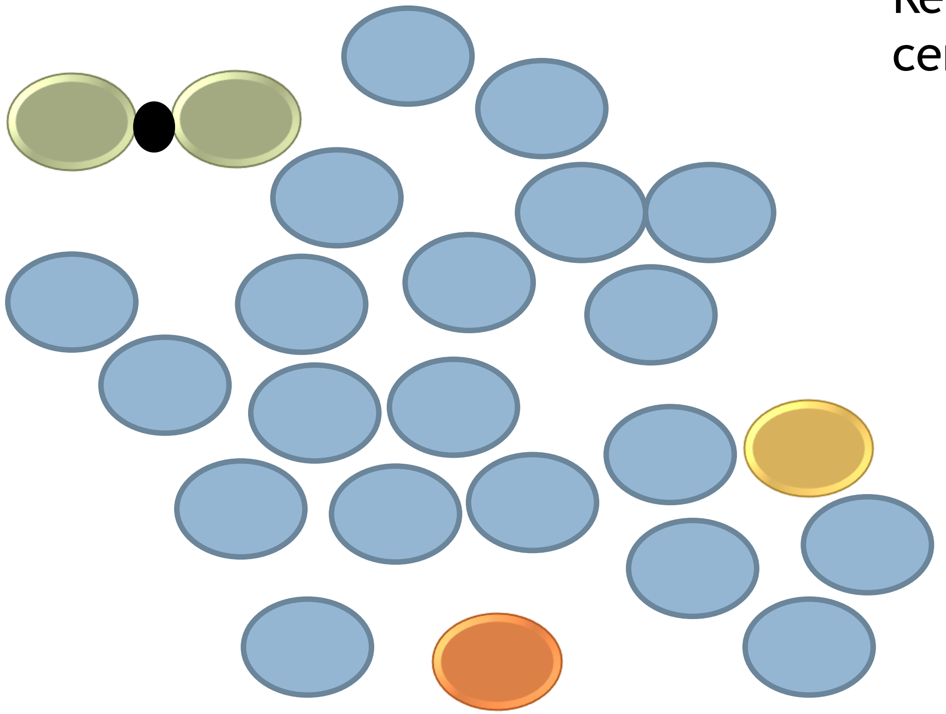
Calculate the distance of each case from all clusters

K-Means clustering



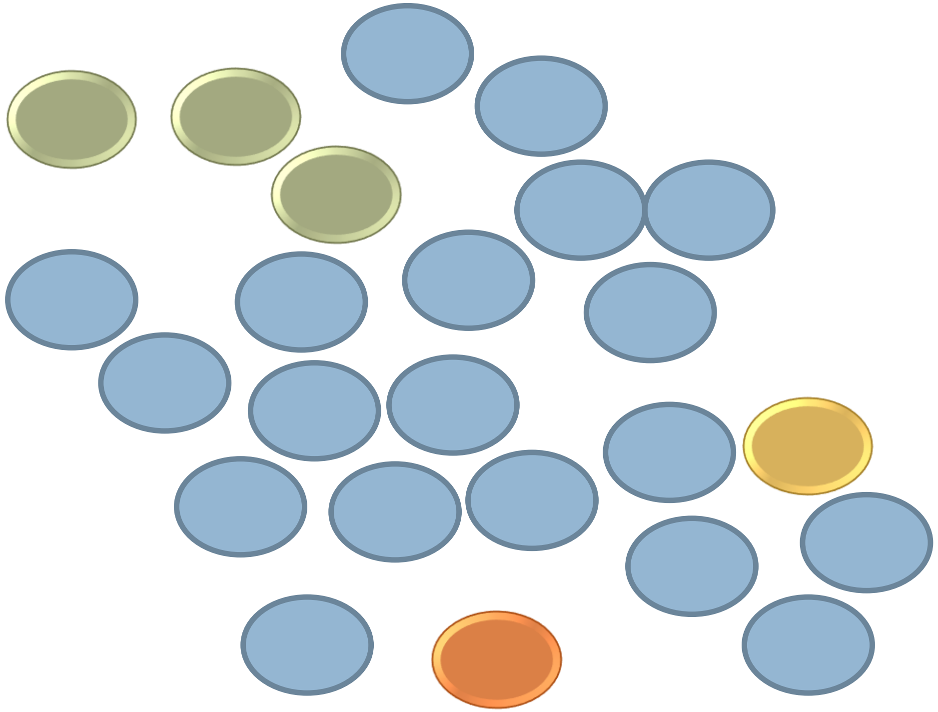
Assign each case to nearest cluster

K-Means clustering

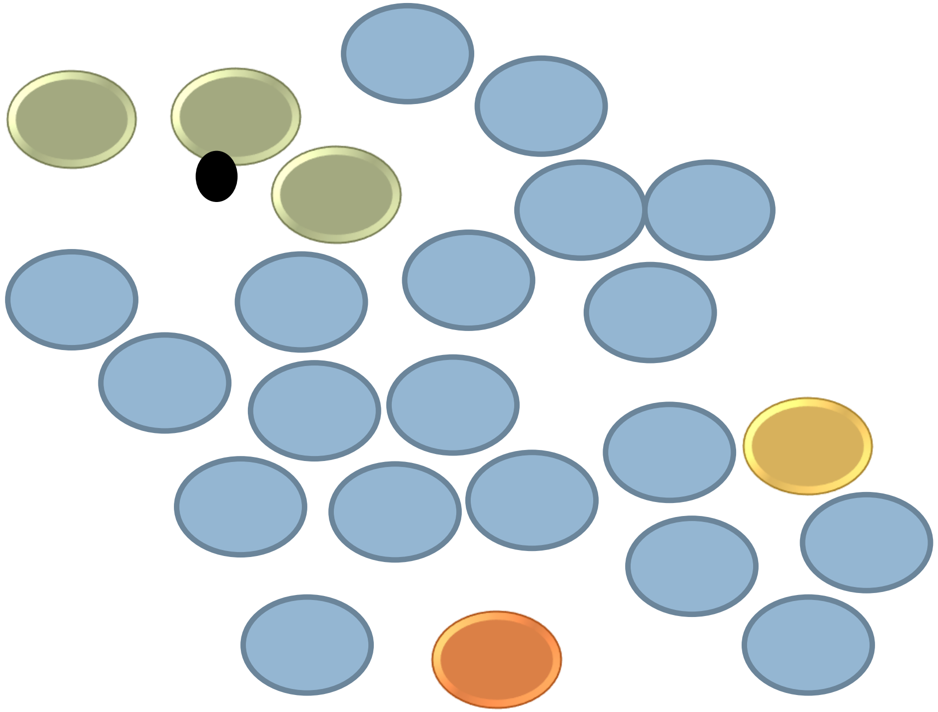


Re calculate the cluster centers

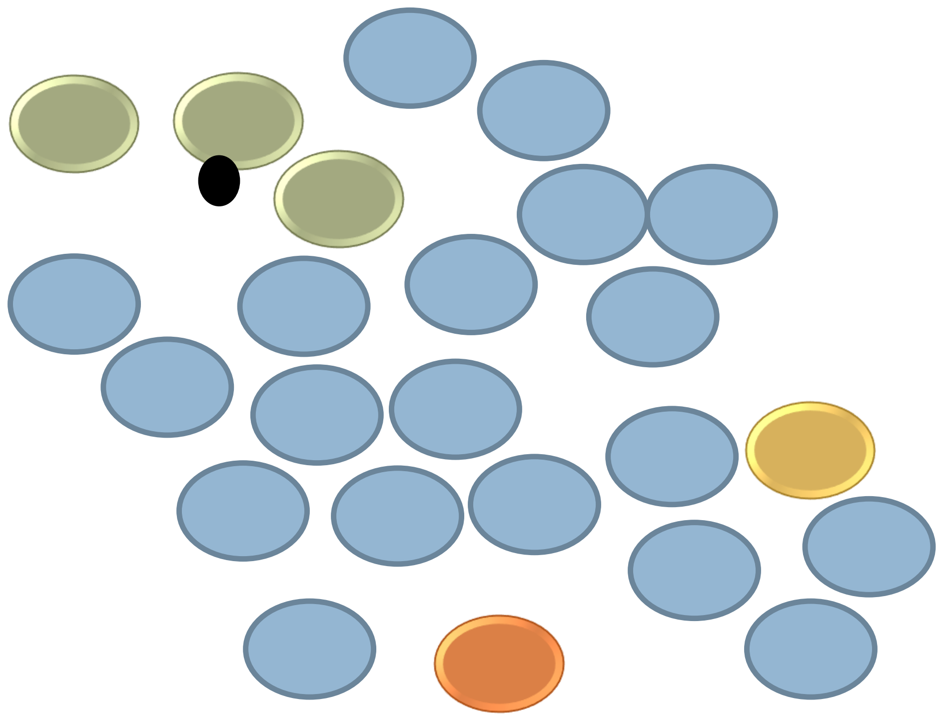
K-Means clustering



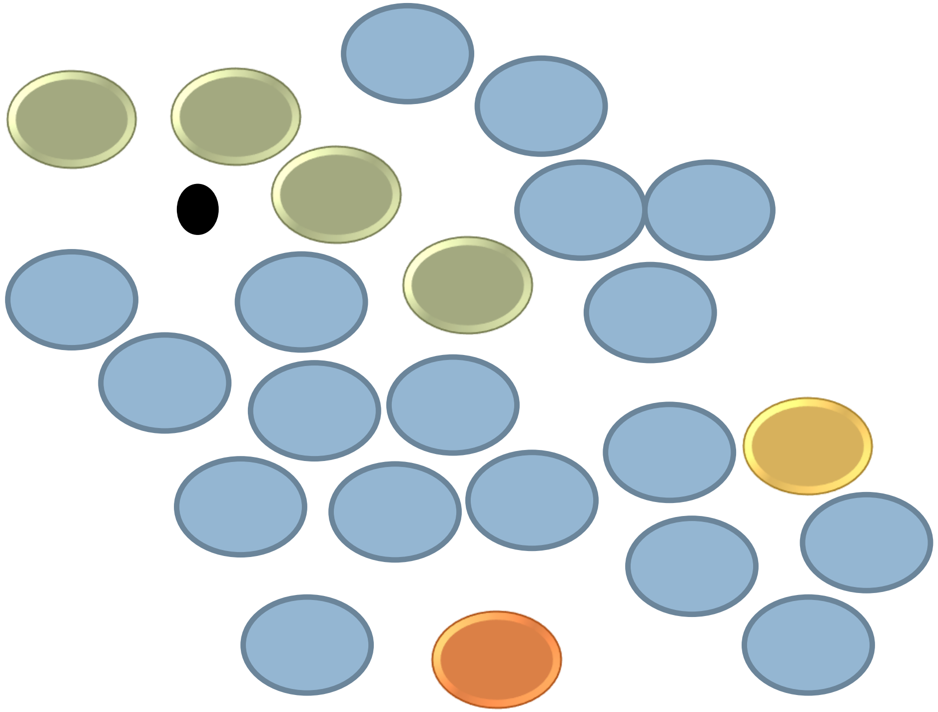
K-Means clustering



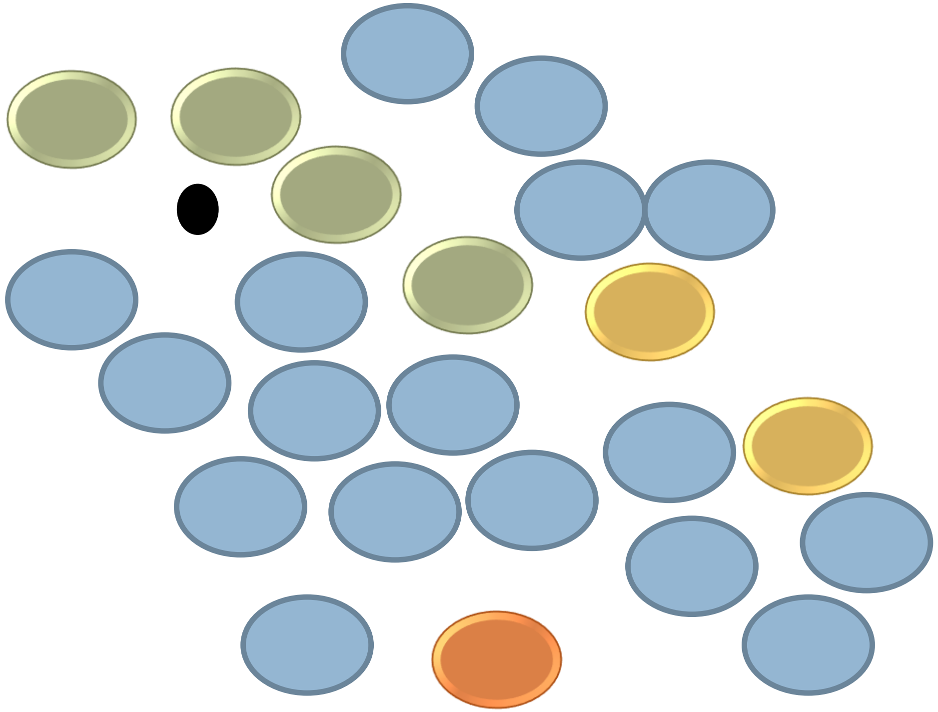
K-Means clustering



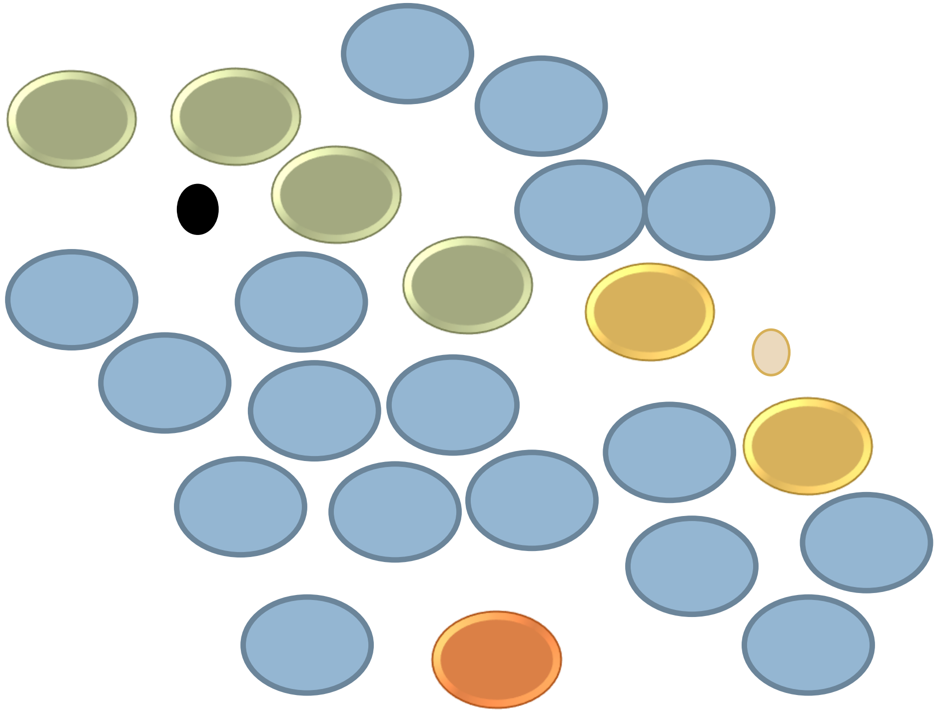
K-Means clustering



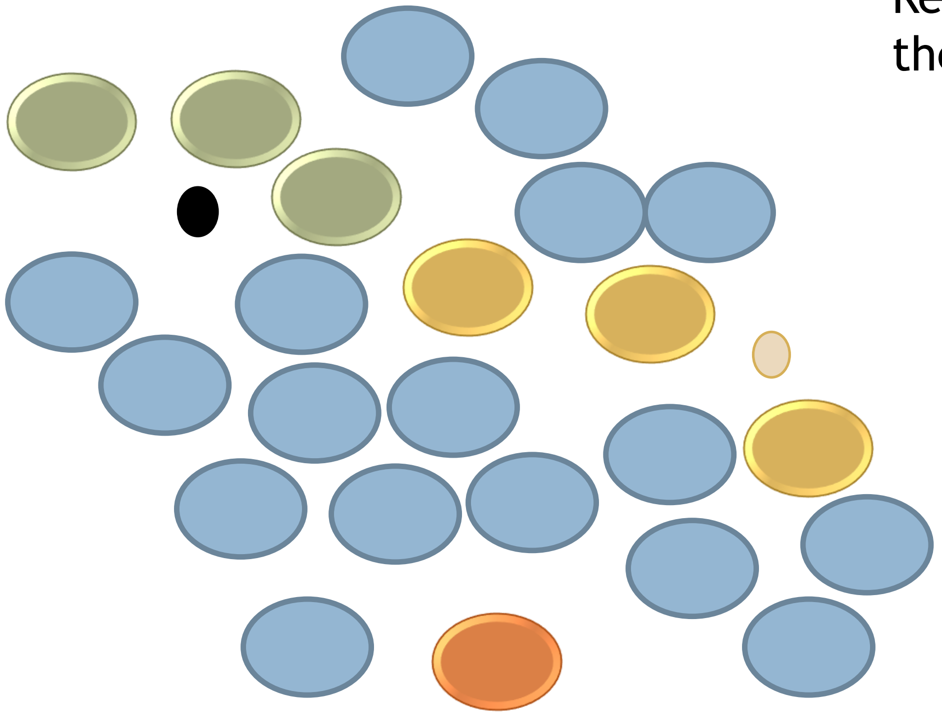
K-Means clustering



K-Means clustering

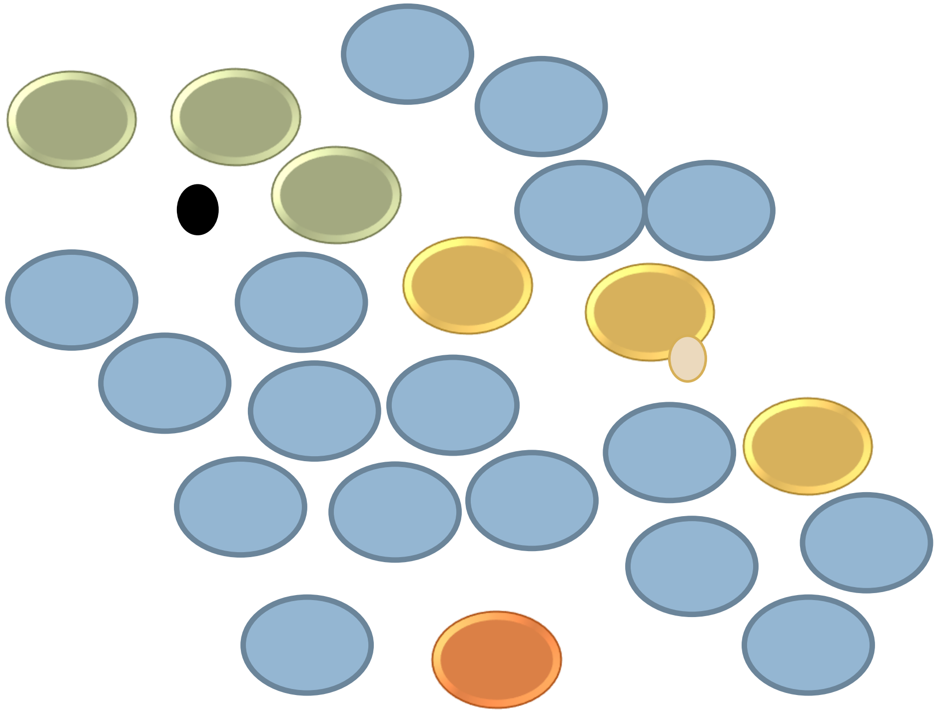


K-Means clustering

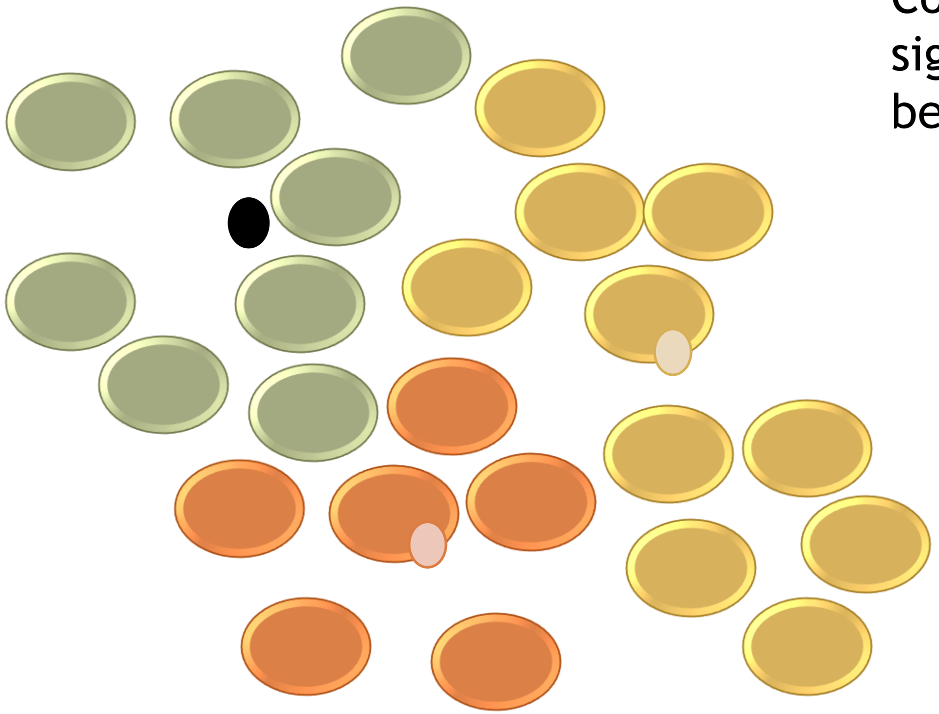


Reassign after changing
the cluster centers

K-Means clustering



K-Means clustering



Continue till there is no significant change between two iterations

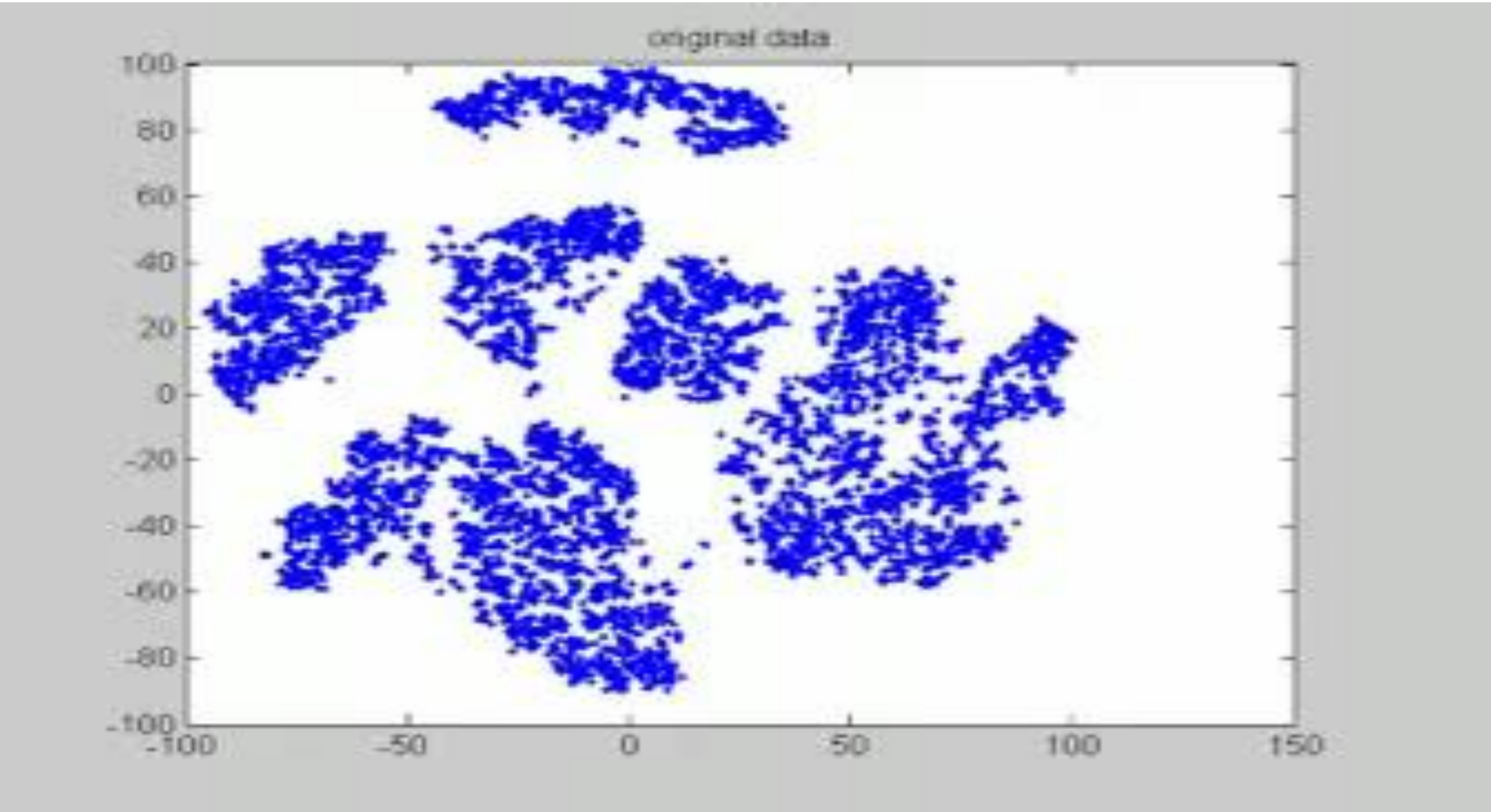
K -Means Clustering – Algorithm

In simple terms

- Initialize k cluster centres
  - Do
    - Assignment step: Assign each data point to its closest cluster center
    - Re-estimation step: Re-compute cluster centers
    - While (there are still changes in the cluster centers)

K Means clustering in action

Dividing the data into 10 clusters using K-Means



K -Means Clustering – Properties

- The K-Means Clustering has the following parameters:

Parameters		Description
Create trainer mode		If you know the exact parameters to choose then choose Single Parameter, else choose Parameter Range and go for Sweep Clustering
Single Parameter	**Parameter Range**	
Number of Centroid	Range for Number of Centroids	Number of clusters the algorithm should begin with, the algorithm starts with this number of clusters and iterates to find the best fit
Initialization	Initialization for Sweep	Specify the algorithm that is used define the initial configuration of cluster
Random number seed	Random number seed	Provide some value so that, whenever we choose this value we get the same data for analysis, when we use Parameter Sweep we can also mention the **Number of seeds to sweep** ie., number of different seeds to start with
Metric	Metric	Choose a metric for measuring the distance between cluster vectors, or between new data points and the randomly chosen centroid
Iterations	Iterations	Number of times the algorithm must be iterated before finalizing the centroid
Assign Label Mode	Assign Label Mode	If we use label column in the dataset then choose one of the options How it should be handled

K -Means Clustering – Properties

- Initialization or Initialization for Sweep Parameter of the K-Means clustering has the following options:

Options	Description
First N	Some initial number of data points are chosen from the dataset as the initial means, also called Forgy method
Random	It randomly places a data points in the cluster and computes the initial mean which will be the centroids od the randomly assigned points in the cluster, also called Random Partition method
K-Means++	It specifies a procedure to initialize the cluster centers before proceeding with the standard *k*-means optimization iterations
K-Means++Fast.	Optimized K-Means ++ algorithm for fast clustering
Evenly	Centroids are selected in such a way that they are equidistant from each other in d-dimensional space of n data points
Use label column	Centroids are selected based on the label column values

K -Means Clustering – Properties

- Metric Parameter of the K-Means clustering has the following options:

Metric	Description
Euclidean	❑ This metric is used to calculate the distance between the numeric variables ❑ It squares the difference in value at each point and sums it up ❑ Formula: $\sqrt{\sum_{k=1}^{n}(x_{ik} - x_{jk})^2}$
Cosine	❑ This is most commonly used similarity metric in text analytics ❑ This metric calculates the angle between two vectors ❑ If the value is equal to 1(cos 0) two vectors are similar, else if the value is equal to 0(cos 90) two vectors are dissimilar

K -Means Clustering – Properties

- Assign Label Mode Parameter of the K-Means clustering has the following options:

Options	Description
Ignore label column	The label column in the dataset is ignored while building the model
Fill missing values	The label column is used for building the model. If there are missing values in the label column then it is imputed using the other features
Overwrite from closest to center	The label column values are replaced with the values of predicted labels of the point that is closest to the centroid

LAB: Building Clusters using K-Means

- A Supermarket wanted to send some promotional coupons to 100 families
- The idea is to identify 100 customers with medium income and low recent spends

Steps - Building Clusters using K-Means

- Drag and drop the **Dataset** into the canvas
- Drag and drop **K - Means Clustering** into the canvas
- Drag and drop **Train Clustering Model**, connect **K - Means Clustering** to the first input and **Dataset** to the second input
- Drag and drop **Edit Metadata,** connect the second output of **Train Clustering Model** to the input of it
- Drag and drop four **Split Data,** connect one below the other, connect the first one with **Edit Metadata**
- Drag and drop five **Compute Elementary Statistics**, connect them to first output of each **Split Data,** in the last one connect in both the outputs

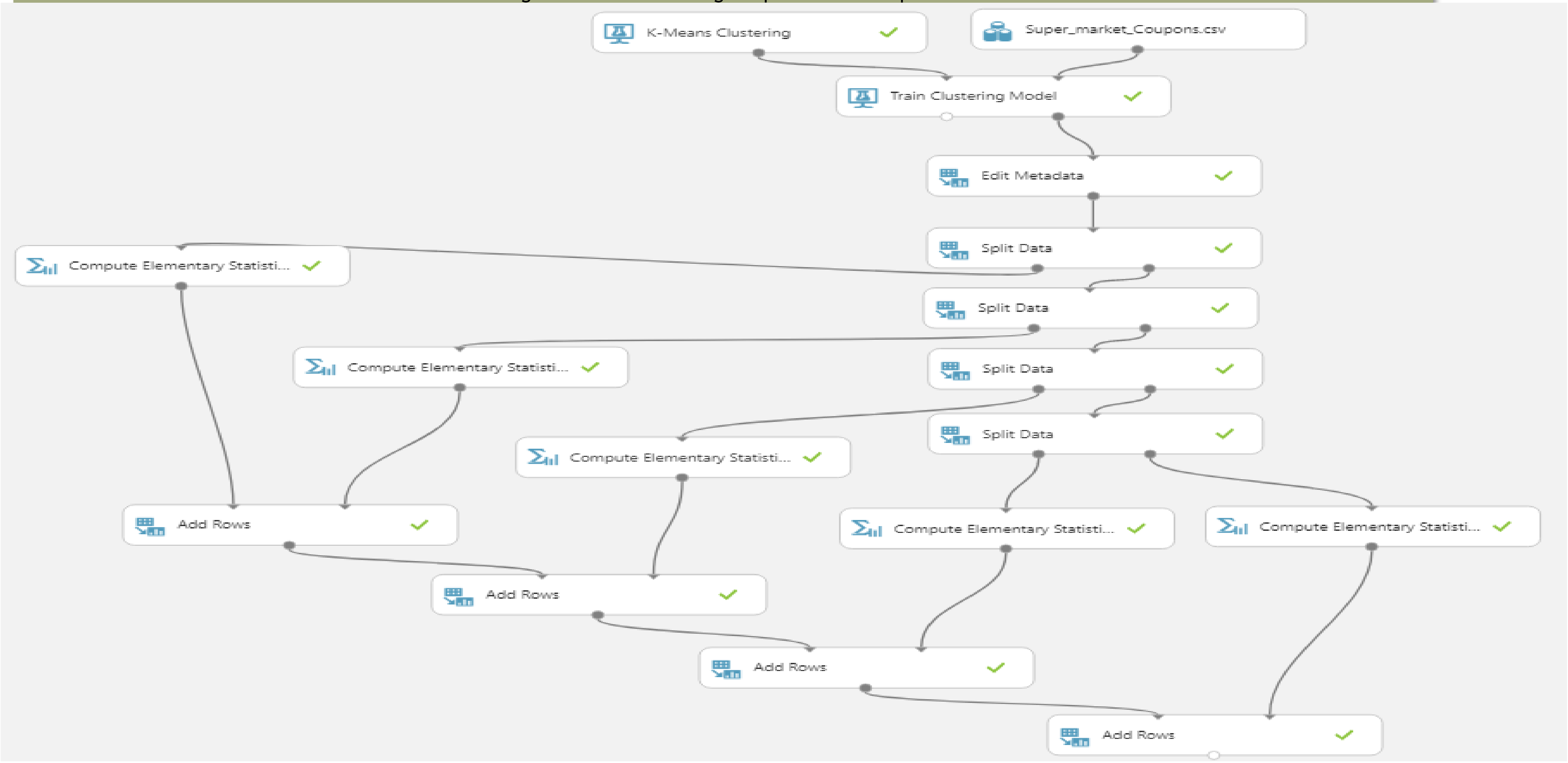
Steps - Building Clusters using K-Means

- Drag and drop four **Add Rows**, Connect **Compute Elementary Statistics** to it as shown in the fig-9
- Click run and visualize the output of Last **Add Rows, Edit Metadata** and second output of **Train Clustering Model**

- Note: Select the properties for **K - Means Clustering**, **Train Clustering Model**, **Edit Metadata, Split Data** and **Compute Elementary Statistics** before run

Steps - Building Clusters using K-Means

Fig9: K-Means Clustering – Super Market Coupons



Steps - Building Clusters using K-Means

Fig10: Properties - K-Means Clustering

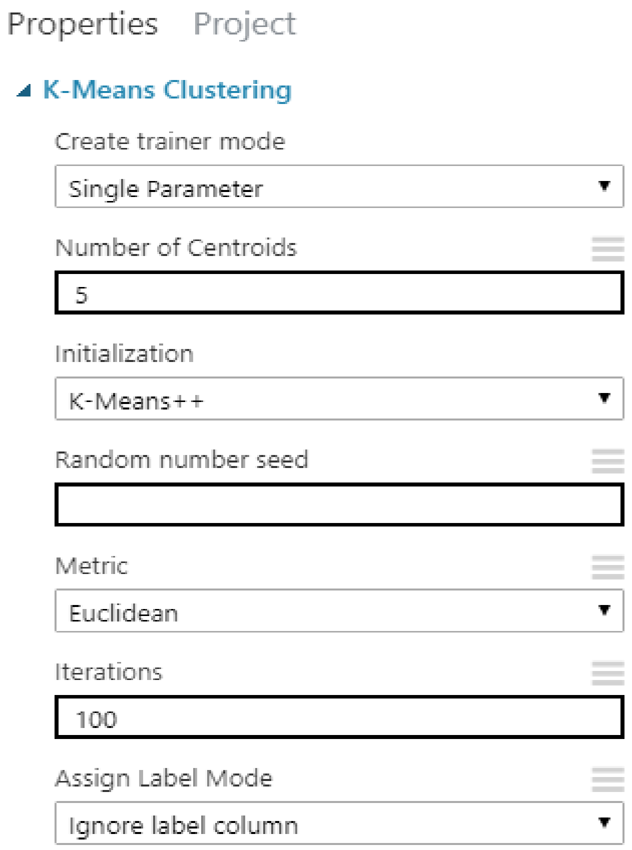
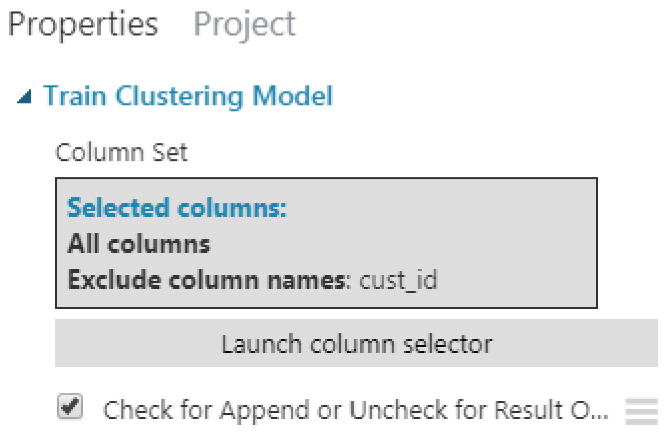


Fig11: Properties -



Steps - Building Clusters using K-Means

Fig12: Properties - Edit Metadata

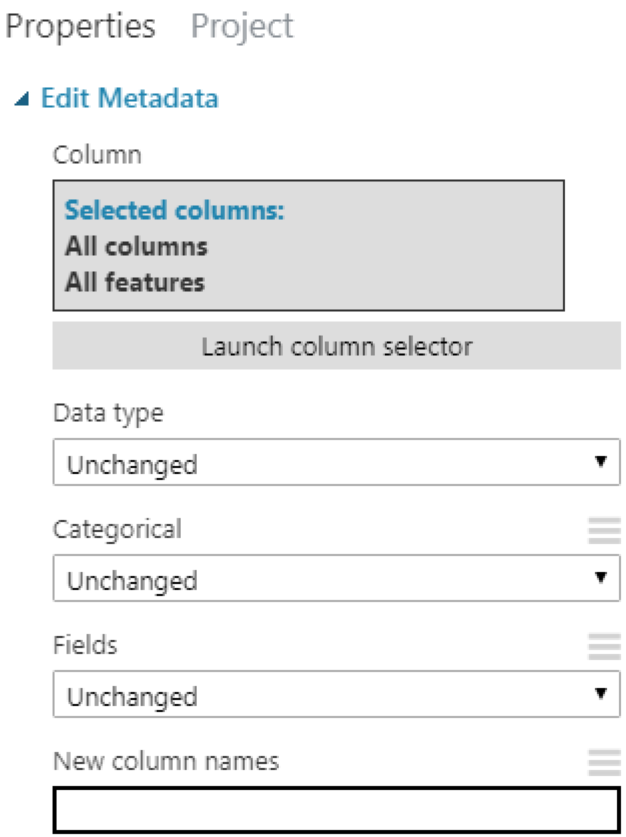
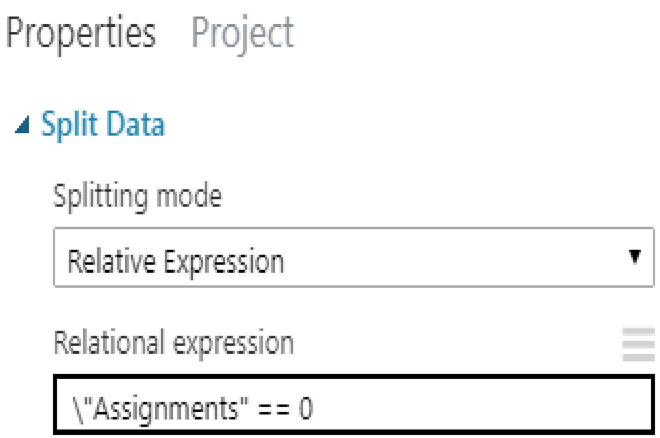


Fig13: Properties - Split Data1



Steps - Building Clusters using K-Means

Fig14: Properties - Split Data2

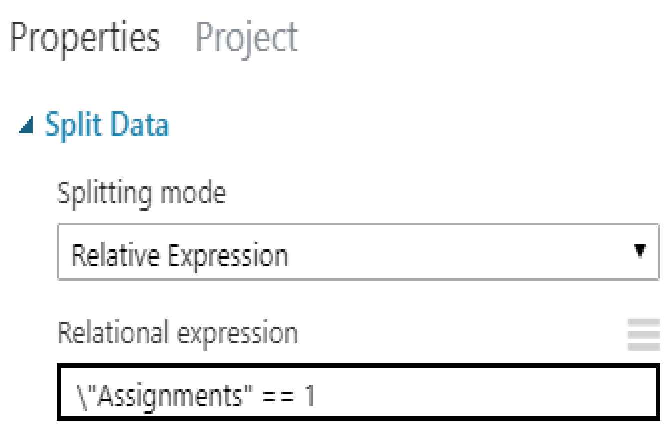
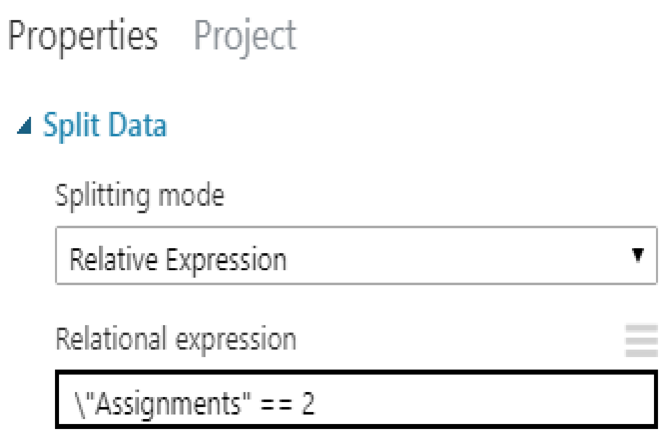


Fig15: Properties - Split Data3



Steps - Building Clusters using K-Means

Fig16: Properties - Split Data4

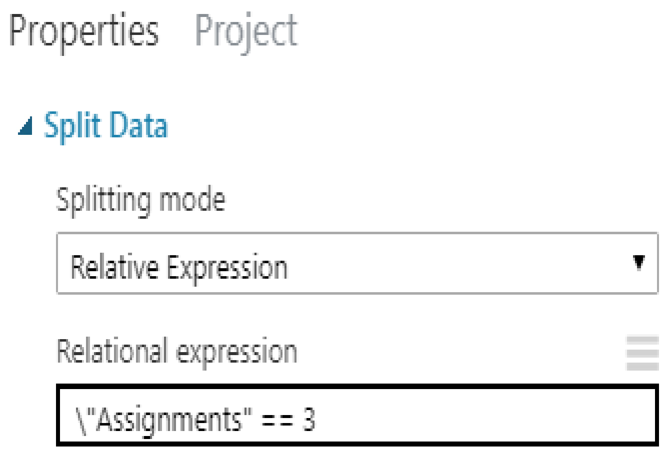
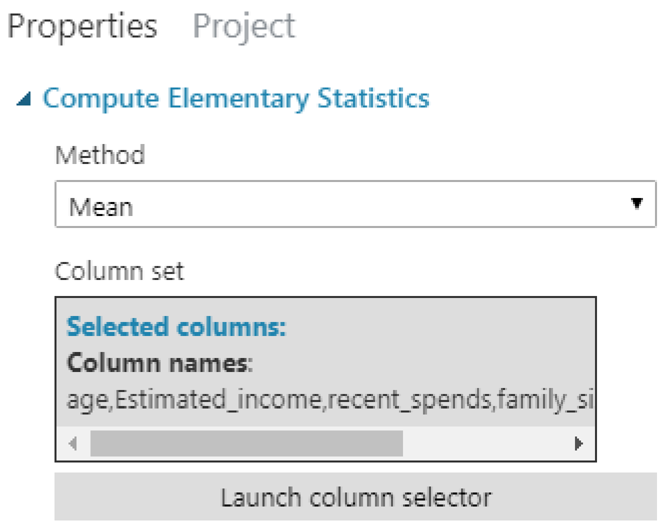


Fig17: Properties - Compute Linear Correlation(same for all)



Steps - Building Clusters using K-Means

Fig18: Centres

Mean(age)	Mean(Estimated_income)	Mean(recent_spends)	Mean(family_size)	Mean(Avg_visits_p
51.547812	7011.721232	1919.986916	1.903566	5.466775
51	142000	25181.23303	5	10
52.700735	1572.062792	455.668536	1.484302	5.603874
54.857143	57507	17310.147427	2.285714	7.857143
53.62069	15255.613027	4974.679002	2.249042	5.417625

Advantages

- Very less **computation time**. This is a huge advantage if you are dealing with large datasets.
- Scaling up is easy and interpretation is simple
- Easy to understand and interpret

Disadvantages of K-Means

- We need to choose the **number of clusters k**, in advance. At times choosing K is not an easy job
- Effective for **numerical data.** Calculating centroid and Euclidian distance requires all the values to be numerical
- Not suitable for data with **outliers and noise.** This type of input data results into clusters with non-homogenous cases in one cluster.
  - Either clean the data for outliers before applying algorithm

Choosing Number of Clusters - K

- If you are not sure with choosing the number of clusters, then go for **Sweep Clustering**
- In **Sweep Clustering** we have the following parameters

Parameters	Description
Metric for measuring clustering result	Algorithm for choosing the best fit for clusters
Specify parameter sweeping mode	The values which should be used while training(Entire Grid or Random Sweep)
Maximum number of runs on random sweep	If Random Sweep is selected, enter a value to limit the number of iterations
Random seed	If Random Sweep is selected, specify a initial seed value so that the values does not change at each run
Column Set	Select the columns based on which the cluster should be built
Check for Append or Uncheck for Result Only	If checked, it appends the Assignments and Distance matrix to the dataset, else it returns only the Assignments and Distance matrix

Choosing Number of Clusters - K

- The Metric for measuring clustering result has four options:

Options	Description
Simplified Silhouette	❑ It is a measure of how similar an object is to its own cluster compared to other clusters ❑ Ranges between -1 to 1,where high value indicates that the object is well matched to its own cluster ❑ If most objects have high values then the configuration of cluster is good
Davies-Bouldin	❑ Davies–Bouldin index(DBI) is the ratio of scatter within the cluster and separation between cluster ❑ It minimize the intra cluster variance and maximize the distance between clusters
Dunn	❑ Dunn Index(DI) is calculated based on minimum inter-cluster distance divided by the maximum cluster size ❑ A higher DI implies better clustering ❑ Whenever we need larger inter-cluster distance and smaller clusters Dann Index can be used
Average deviation	❑ This is calculated by taking the average distance from each data point to its cluster center ❑ This is not useful if we go for Random Sweep for finding centroids ❑ If you want to use this select sweeping mode as Entire Grid

statinfer

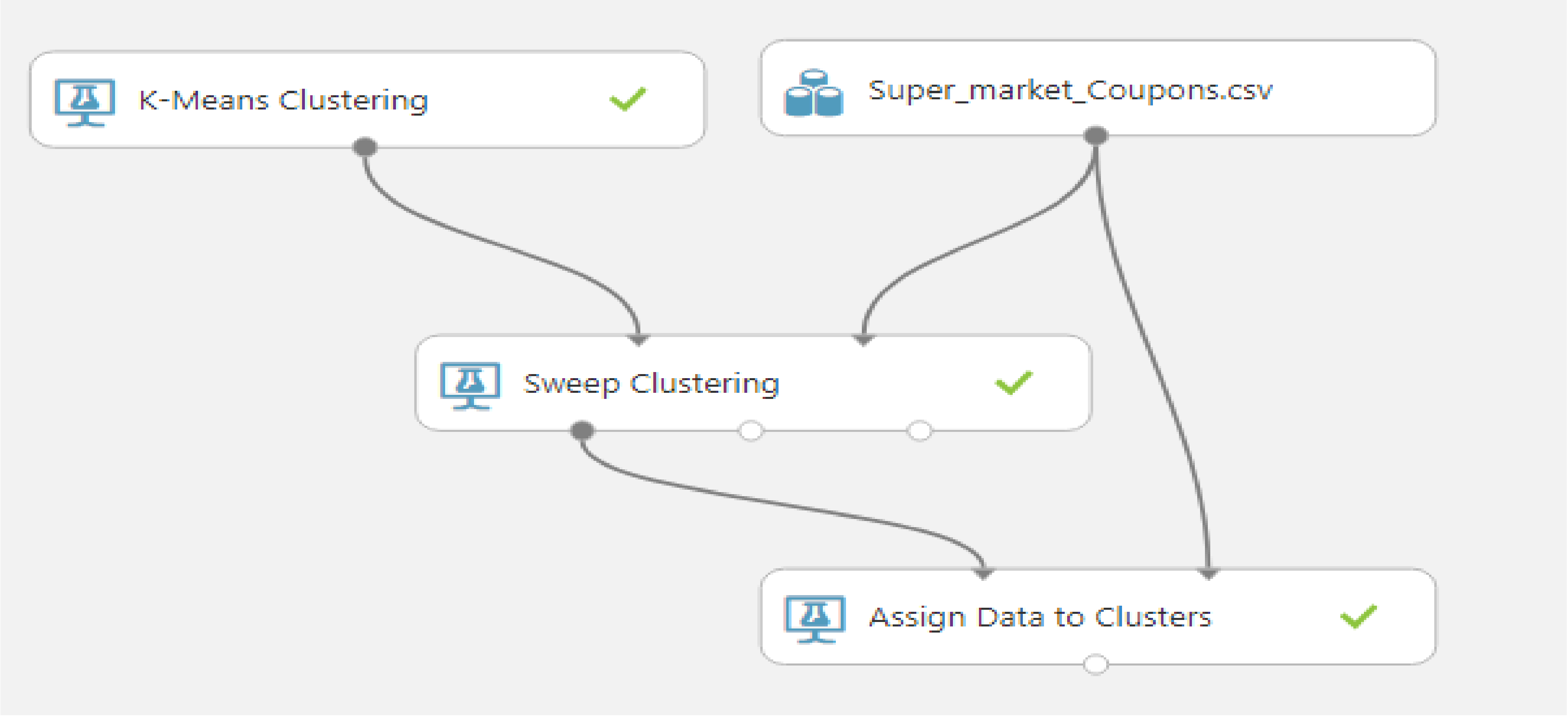
Steps - Choosing Number of Clusters - K

- Drag and drop the **Dataset** into the canvas
- Drag and drop the **K-Means Clustering** module in to the canvas
- Drag and drop the **Sweep Clustering**, connect **Dataset** to the second input and **K-Means Clustering** to the first input
- Drag and drop **Assign to Clusters**, connect the first output of the **Sweep Clustering** to the first input and **Dataset** to the second input
- Click on run, visualize the second and third output of **Sweep Clustering** and the output of **Assign to Clusters**

- Note: select the properties of **K-Means Clustering** and **Sweep Clustering** before run

Steps - Choosing Number of Clusters - K

Fig19: K-Means Clustering with Parameter Sweep



Steps - Choosing Number of Clusters - K

statinfer

Fig20: Properties - K-Means Clustering

Properties   Project

K-Means Clustering

Create trainer mode

Parameter Range ▼

Range for Number of Centroids

☐ Use Range Builder

2, 3, 4, 5

Initialization for sweep

K-Means++ ▼

Random number seed

20

Number of seeds to sweep

10

Metric

Euclidean ▼

Iterations

100

Assign Label Mode

Ignore label column ▼

Fig21: Properties - Sweep Clustering

Properties   Project

Sweep Clustering

Metric for measuring clustering result

Average Deviation ▼

Specify parameter sweeping mode

Entire grid ▼

Column Set

Selected columns:
All columns
Exclude column names: cust_id

Launch column selector

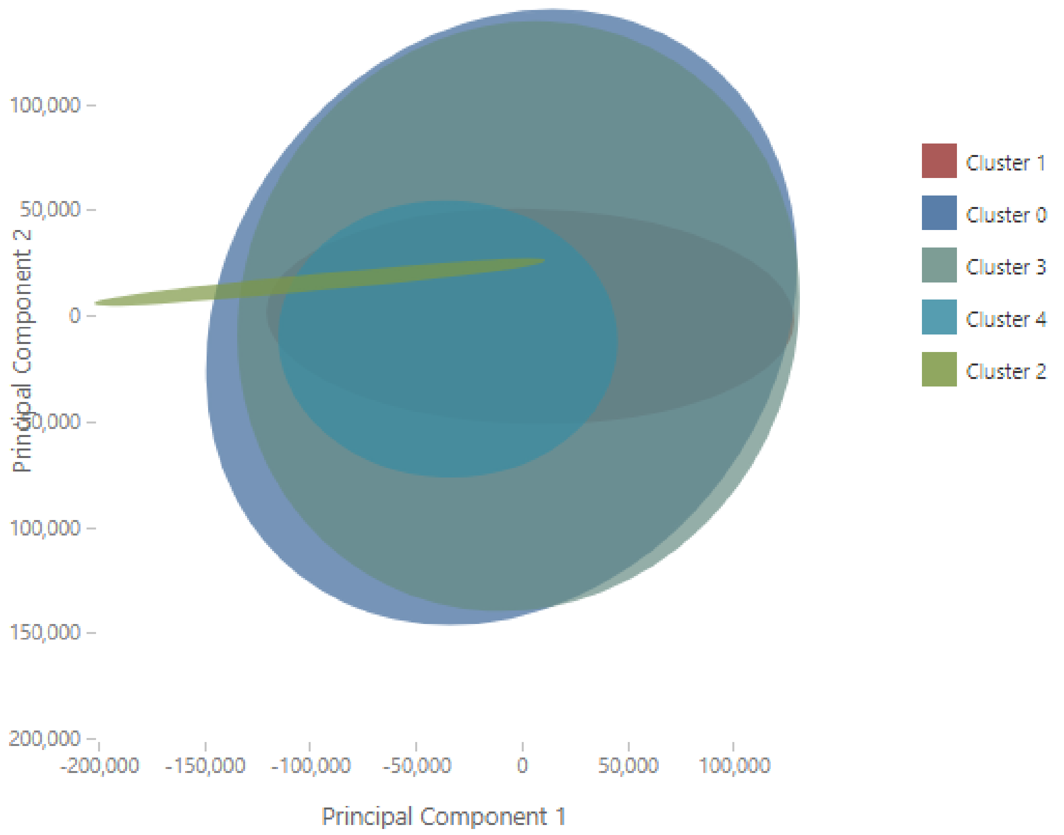
☑ Check for Append or Uncheck for Result Only

Steps - Choosing Number of Clusters - K

Fig22: Sweep Results at each Iteration

Cluster Metric	Number of Centroids	Index of Run
2392.890504	5	35
2423.485656	5	30
2423.485656	5	32
2423.485656	5	33
2423.485656	5	34
2423.485656	5	36
2423.485656	5	37
2423.485656	5	38
2423.485656	5	39
2714.374167	5	31
2738.666107	4	22
2777.756029	4	25

Fig23: Clusters



Steps - Choosing Number of Clusters - K

Fig24: Result Dataset with Assignments and Distance Matrix

cust_id	age	Estimated_income	recent_spends	family_size	Avg_visits_permonth	Assignments	DistancesToClusterCenter no.0
1	30	3300	771.572261	1	4	1	11890.440886
2	46	12454	128.922027	3	3	0	4784.471412
3	76	0	0	1	8	1	15271.056529
4	38	3000	76.967031	3	3	1	12400.8005
5	39	2500	2499.99975	1	1	1	12272.552098
6	24	750	749.999925	1	5	1	14345.86128
7	68	1	0.368876	1	3	1	15269.983057
8	38	13000	10842.62435	3	9	0	6652.3202
9	62	0	0	1	3	1	15271.042452
10	29	2231	2213.187625	1	6	1	12584.89376
11	46	3326	32.557755	3	2	1	12111.612104
12	27	764	5.194507	1	8	1	14539.437012
13	31	2000	8.528158	3	6	1	13365.14394
14	67	0	0	1	4	1	15271.045869

Data Standardisation

statinfer

Standardised Data

Actual Data

Custid	Debt Ratio	Credit Limit
C1	0.4	5000
C2	0.39	5100
C3	0.8	5000

Distance Matrix

Assignments	DistancesToClusterCenter no.0	DistancesToClusterCenter no.1	DistancesToClusterCenter no.2
0	0	100.00084	0.4
1	100.00084	0	100.000001
2	0.4	100.000001	0

$$Standardised\ value = \frac{x - mean(x)}{sd(x)}$$

Standardised data

Custid	Debt Ratio	Credit Limit
C1	-0.555793	-0.57735
C2	-0.598546	1.154702
C3	1.154339	-0.57735

Distance Matrix

Assignments	DistancesToClusterCenter no.0	DistancesToClusterCenter no.1	DistancesToClusterCenter no.2
0	0	2.464267	1.710132
1	2.464267	0	1.732579
2	1.710132	1.732579	0

Conclusion

Conclusion

- K – means is a partitional clustering algorithm.
- K-Means is an unsupervised learning method
- There are other methods too. Some algorithms work well on a certain type of problems.
  - Hierarchical Clustering, Density-based ,Grid-based Clustering,Model-based Clustering, Frequent pattern-based Clustering
- Try multiple times to decide the right K-value
- Clustering is also used in text mining
  - Document clustering
  - News articles clustering

Appendix

statinfer

Non- Numerical Data

Distance Measure for Non- Numeric data

- Distance measure for Binary Variables/Flag Variable/Indicator variable / Boolean Variable

Point X_j

Point X_i

	1	0	
1	A	B	A+B
0	C	D	C+D
	A+C	B+D	A+B+C+D

$$d = \frac{B+C}{A+B+C+D}$$

Distance Measure For binary Variables

Customer ID	House Loan	Existing Customer	Gender	Marital Status	Premier Customer
C001	Yes	Yes	M	No	No
C002	Yes	No	M	Yes	No

C002

		1-Yes	0-No	
C001	1-Yes	2	1	
	0-No	1	1	
				5

$$d = \frac{B+C}{A+B+C+D}$$

Distance Measure For binary Variables

Customer ID	House Loan	Existing Customer	Gender	Marital Status	Premier Customer
C001	Yes	Yes	M	No	No
C002	Yes	No	M	Yes	No

C002

	1-Yes	0-No	
C001 1-Yes	2	1	
C001 0-No	1	1	
			5

$$d = \frac{B+C}{A+B+C+D}$$

Distance (Dis-similarity) =2/5

Distance Measure for Categorical Variables

- Categorical variables are a generalization of the binary variables that can take more than two values
- We can create multiple binary variables(dummy variables) from one categorical variable. If there are ten classes in a categorical variable then we can create ten dummy variables (Nine are sufficient)

Region
East
West
North
South
West

East	West	North	South
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1
0	1	0	0

Distance Measure for Categorical Variables

- Categorical values have lot of classes we can simply calculate the distance by considering Matching vs Non-Matching Cases
- K - Number of variables
- S - Number matching Cases

$$d = \frac{N - S}{N}$$

Distance Measure for Categorical Variables

Customer ID	Region	Card Type	Status Code	Marital Status	Account type
1	EAST	C	A	No	Premier
2	NORTH	B	D	Yes	Premier
3	NORTH	B	H	Yes	Basic

d(1,2)= (5-1)/5 = 4/5
d(1,3)= (5-0)/5 = 5/5
d(2,3)= (5-3)/5 = 2/5

Centroid for Non-Numerical data

- Cluster mean is not possible for categorical data
- We can use two metrics as central tendencies
- Mode
  - Most occurring class is one more measure of central tendency like mean
- Medoids
  - Medoids are similar in concept to means or centroids, but medoids are always members of the data set. Medoids are most commonly used on data when a mean or centroid cannot be defined
  - Medoid one chosen, centrally located object in the cluster.
  - Most centrally located observation in a cluster.

K-Means for Non-Numerical Data: K-modes

- Follow the same algorithm but consider below options
  - Choose a distance matrix that can handle categorical values
  - Choose a centroid that can handle categorical values

Thank you

Credits

- Document prepared by Rangesh

statinfer

www.statinfer.com