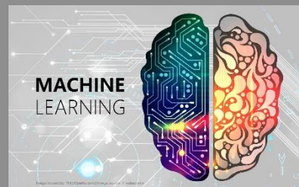
Machine Learning Concepts

Lesson 0: Introduction

Dr.Prasanalakshmi



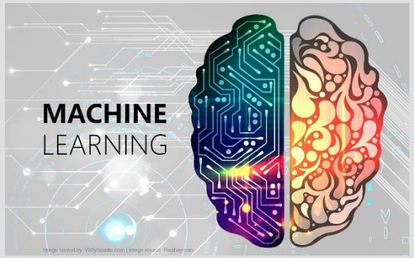
Course Objectives

By the end of this course, you will be able to:

- Programmatically download and analyze data
- Learn data visualization
- Master the art of data analysis by using Ipython notebooks
- Gain insight into the role of a machine learning engineer
- Explain machine learning
- Work with industry-based data
- Learn the tools and techniques for predictive modeling
- Discuss machine learning algorithms and their implementation
- Validate machine learning algorithms
- Demonstrate time series and its related concepts

Outline

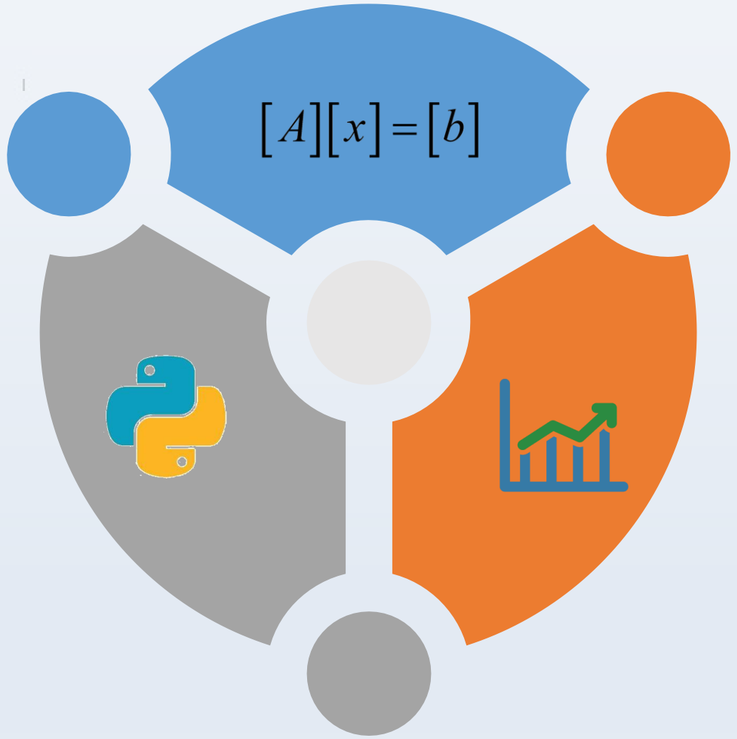
1. Introduction to Artificial Intelligence and Machine Learning
2. Data Wrangling and Manipulation
3. Supervised Learning: Regression
4. Feature Engineering
5. Supervised Learning: Classification
6. Unsupervised learning
7. Time Series Modelling
8. Ensemble Learning
9. Recommender Systems
10. Text Mining



Prerequisites

Prior knowledge of the following domains and technology is helpful:

Linear Algebra



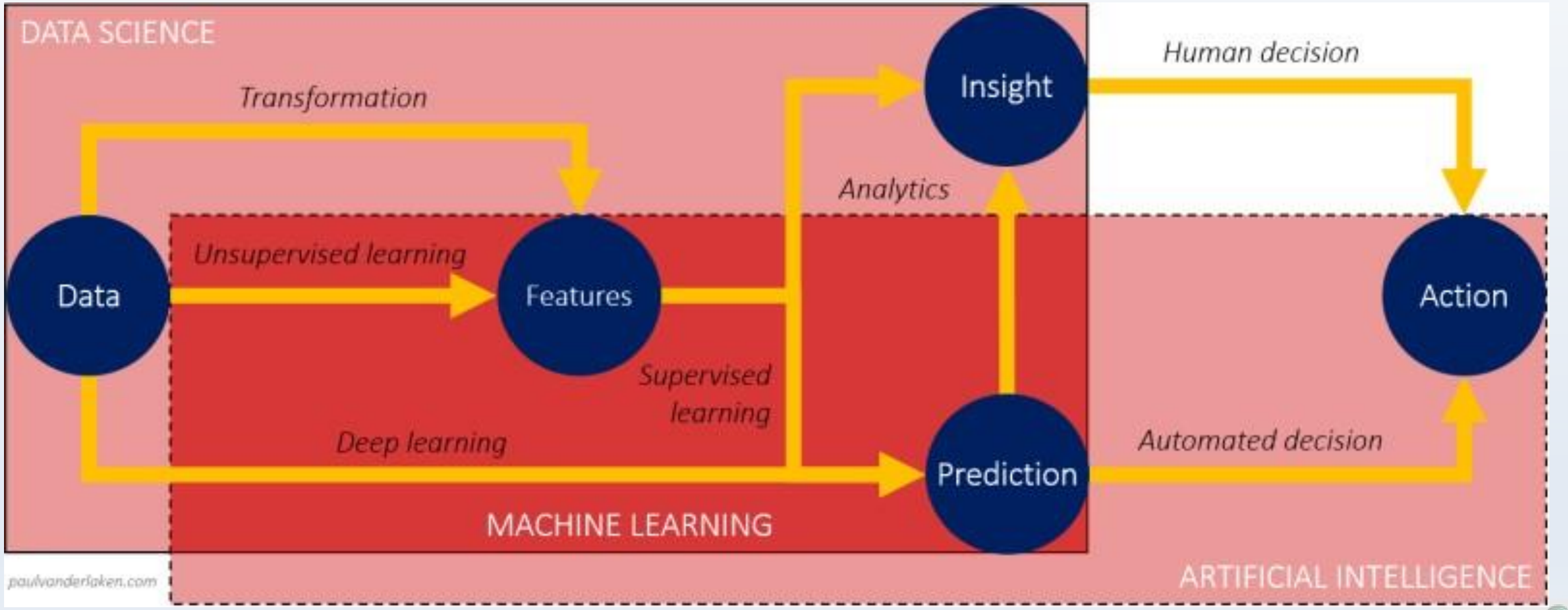
Statistics

Python

MACHINE LEARNING

Machine Learning

Relationship between Artificial Intelligence, Machine Learning, and Data Science



Features of Machine Learning

01

It uses the data to *detect patterns* in a dataset and *adjust program actions accordingly*

02

It *focuses on* the *development of computer programs* that can teach themselves to *grow and change* when *exposed* to *new data*

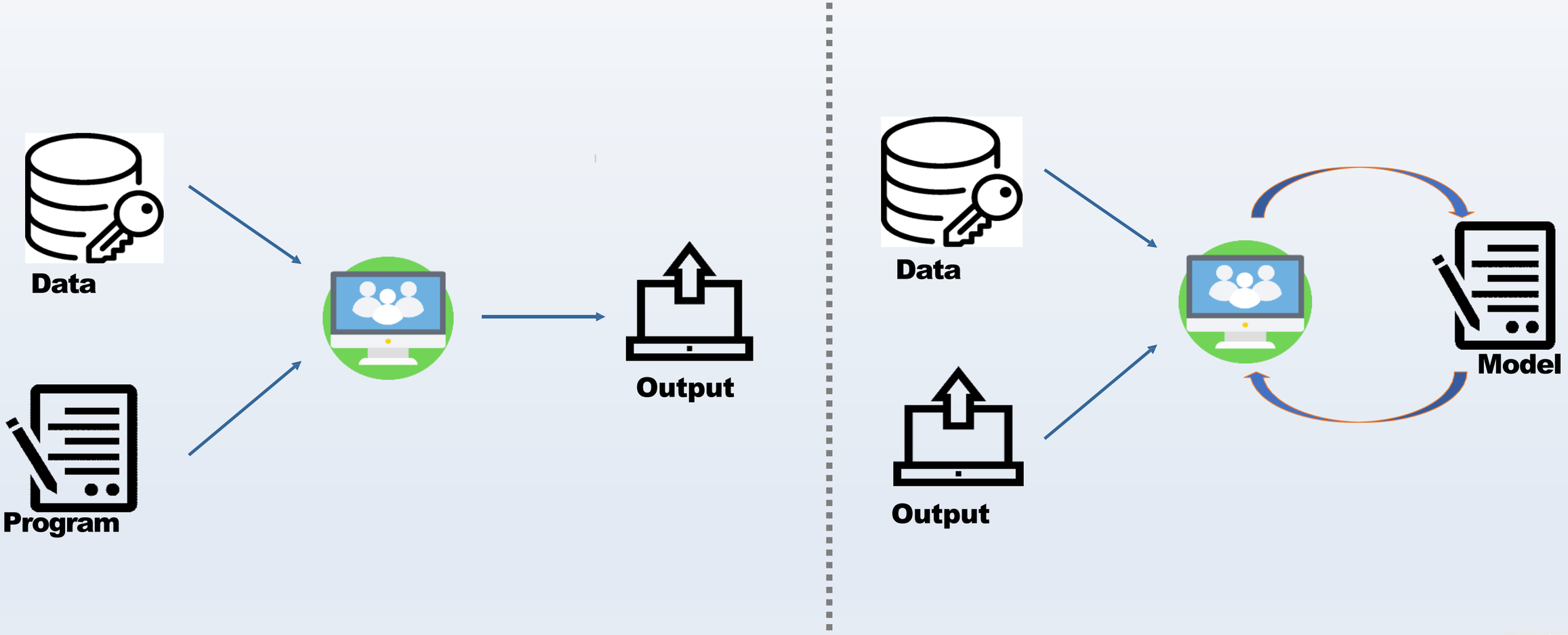
03

It enables computers to *find hidden insights using iterative algorithms without being explicitly programmed*

04

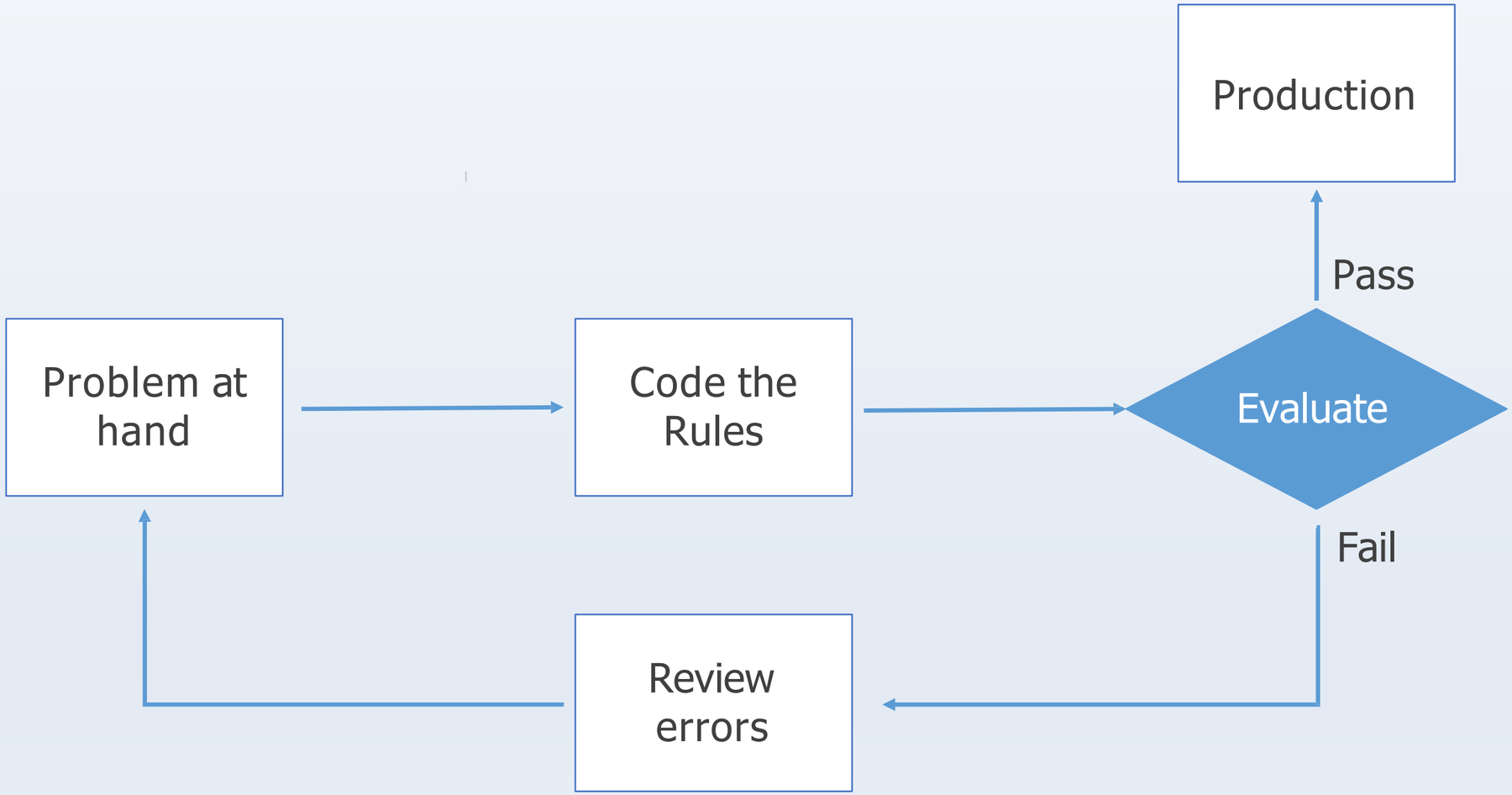
It *automates analytical model building*

Traditional Approach vs. Machine Learning Approach



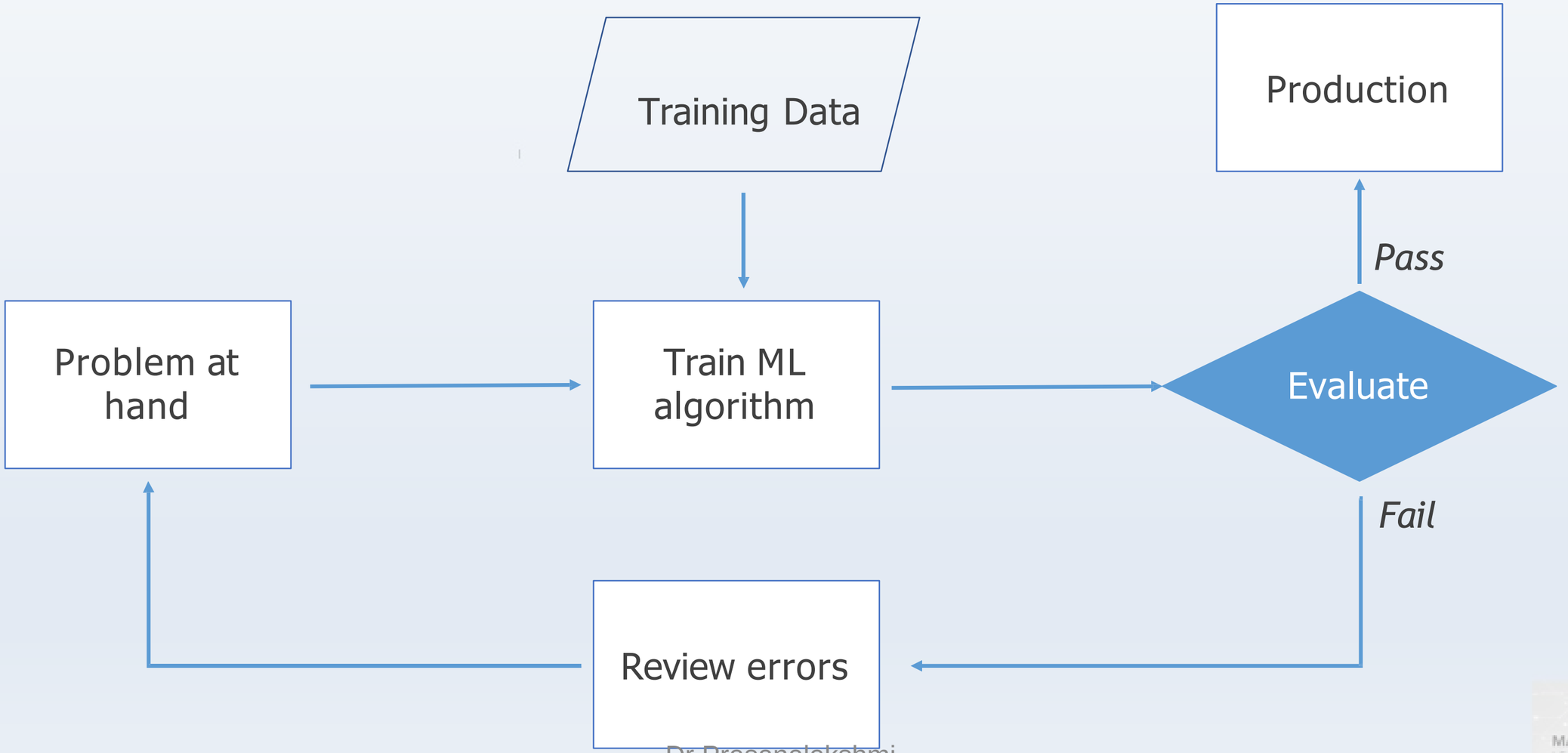
Traditional Approach

Traditional programming relies on hard-coded rules.



Machine Learning Approach

Machine Learning relies on learning patterns based on sample data.



Relationship between Machine Learning and Data Science

Data science is the use of statistical methods to find patterns in the data.

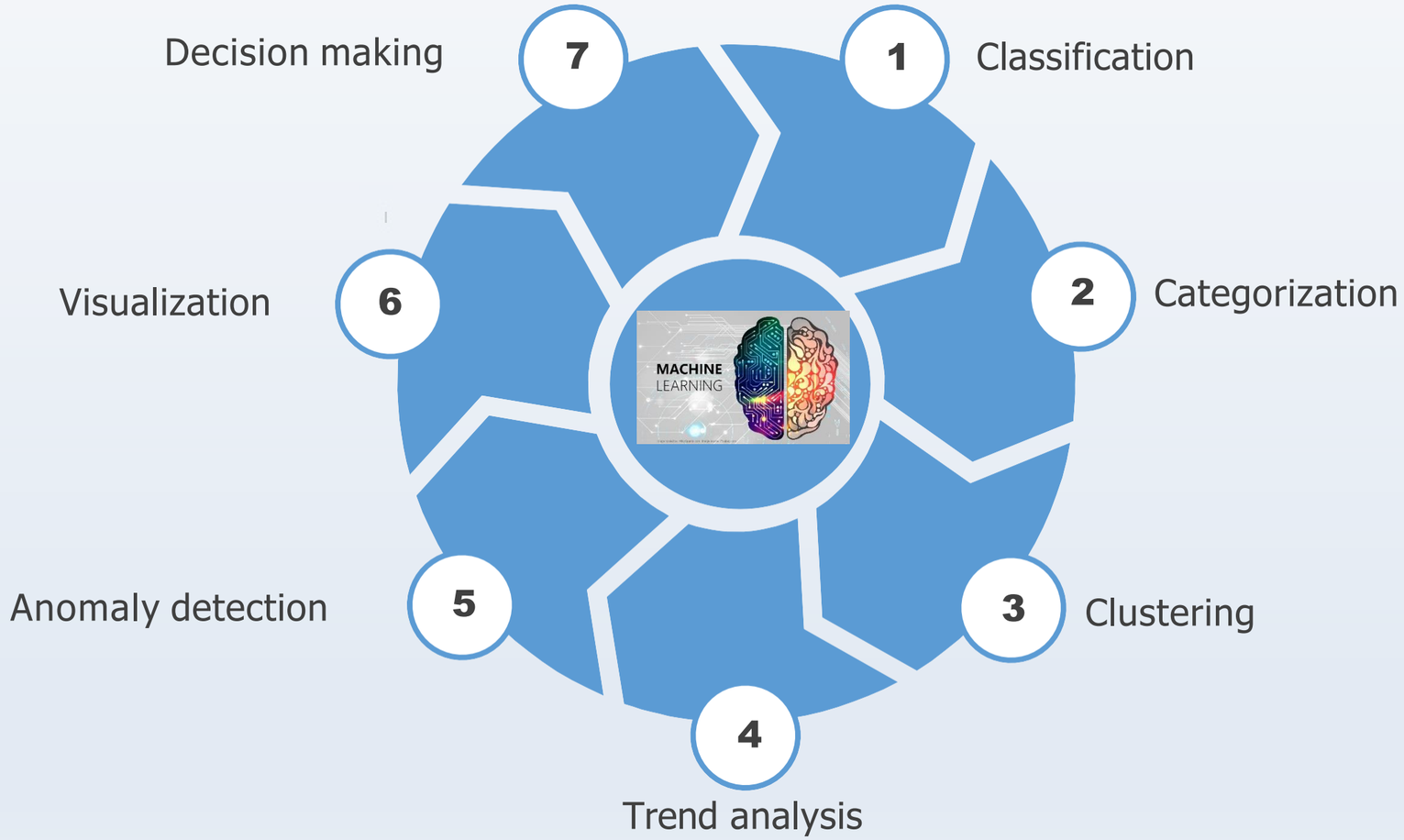
Statistical machine learning uses the same math and techniques as data science.

These techniques are integrated into algorithms that learn and improve on their own.

Machine Learning facilitates Artificial Intelligence as it enables machines to learn from the patterns in data.

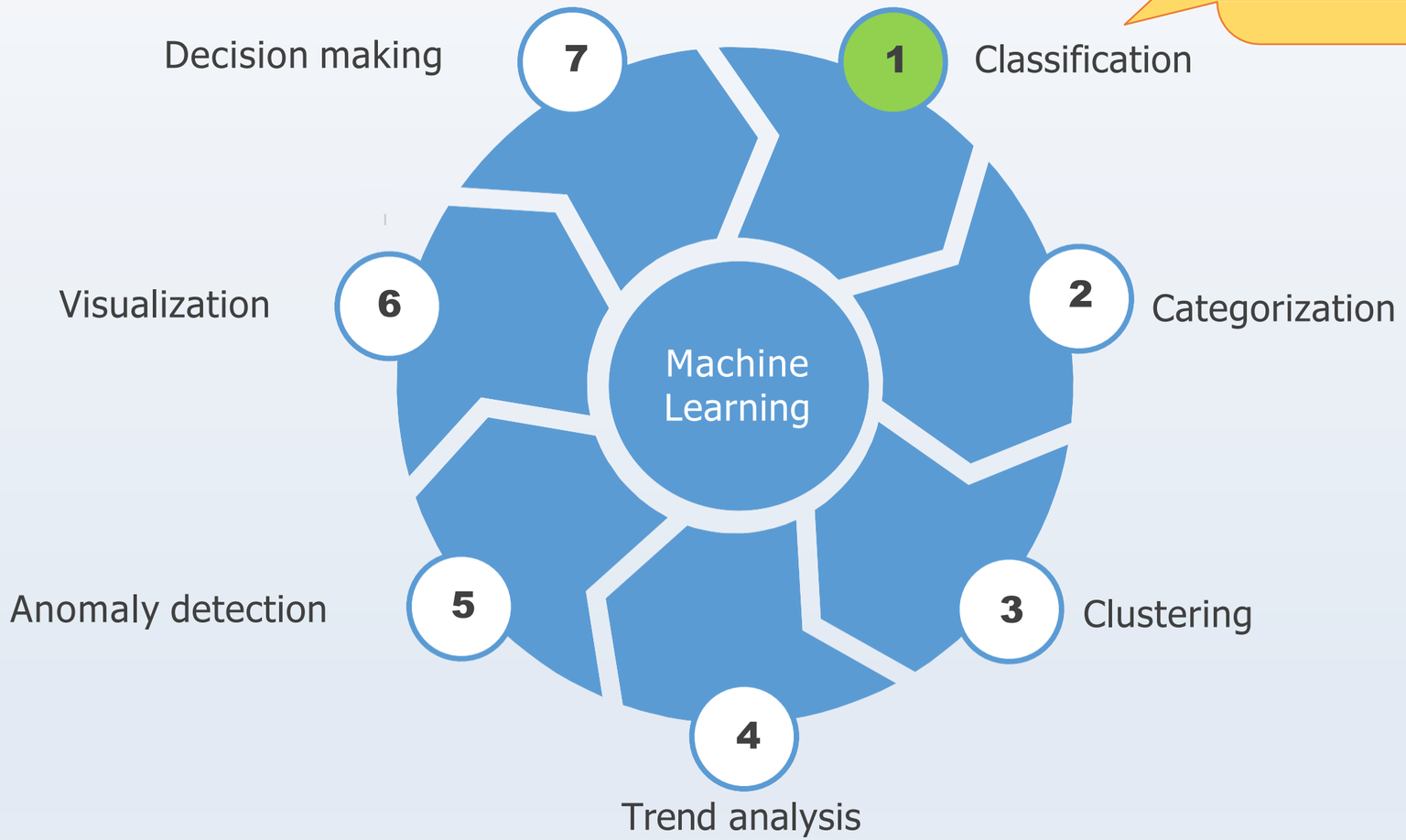
Machine Learning Techniques

Machine Learning uses a number of theories and techniques from Data Science:

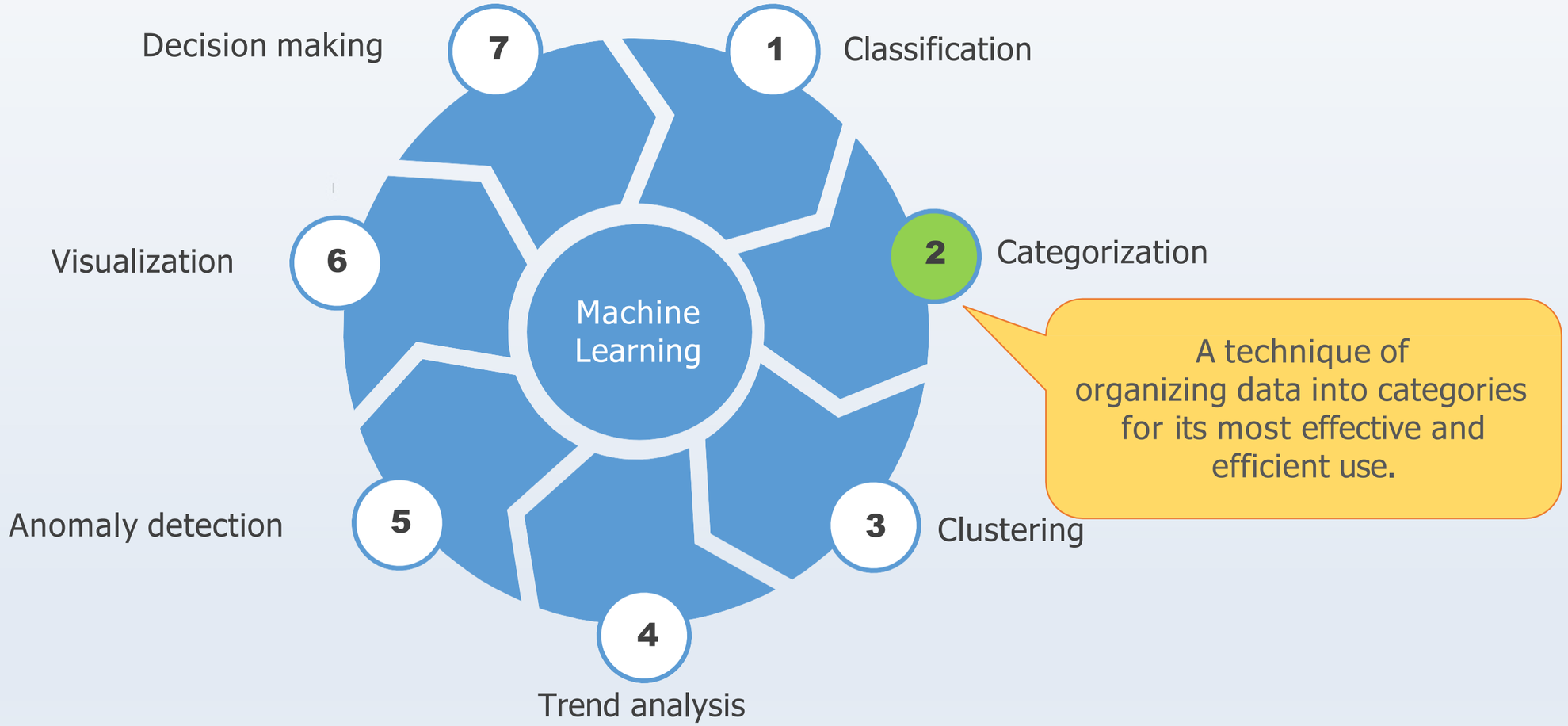


Machine Learning Techniques

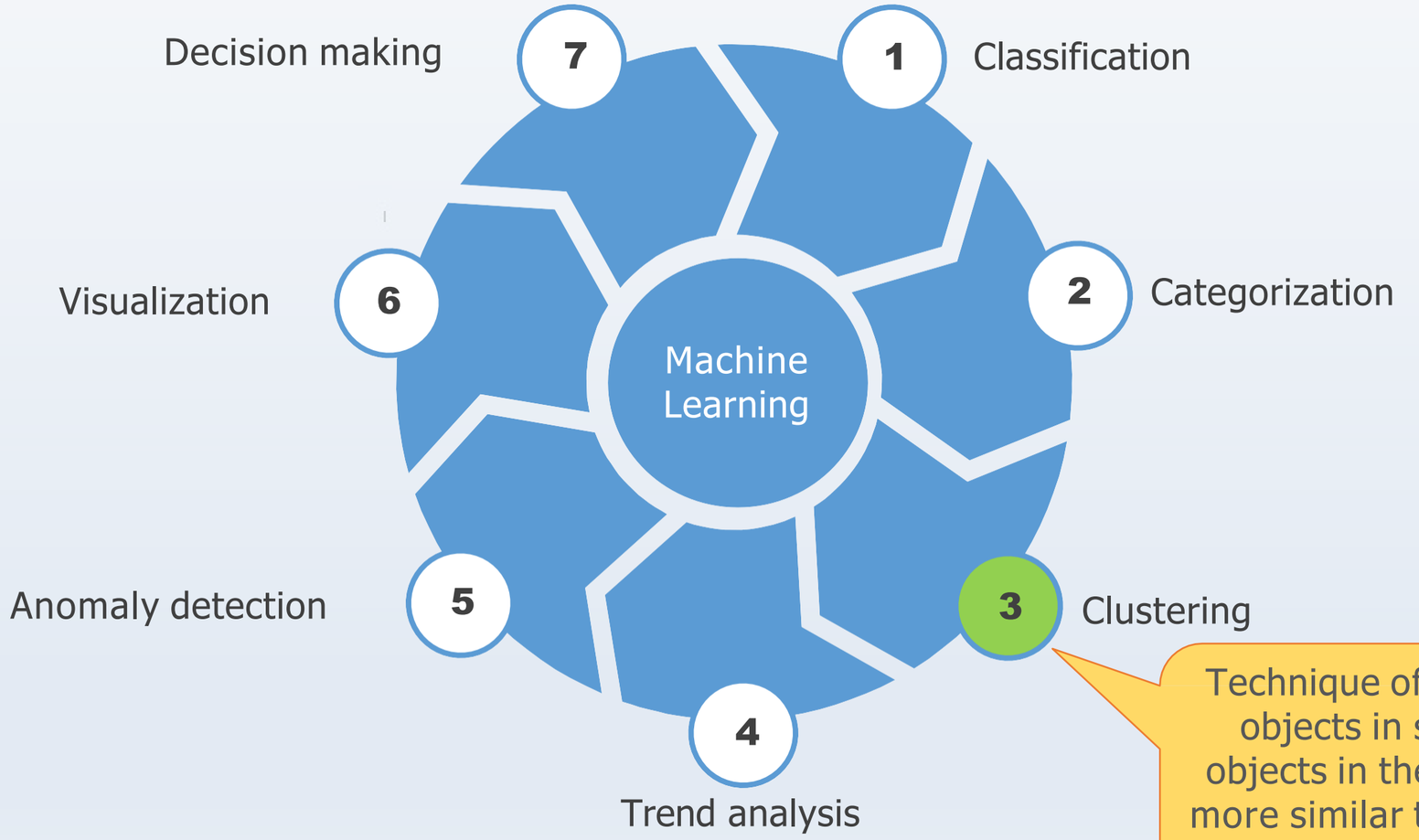
classification is a technique in which the computer program learns from the data input given to it and then uses this **learning** to **classify** new observation



Machine Learning Techniques

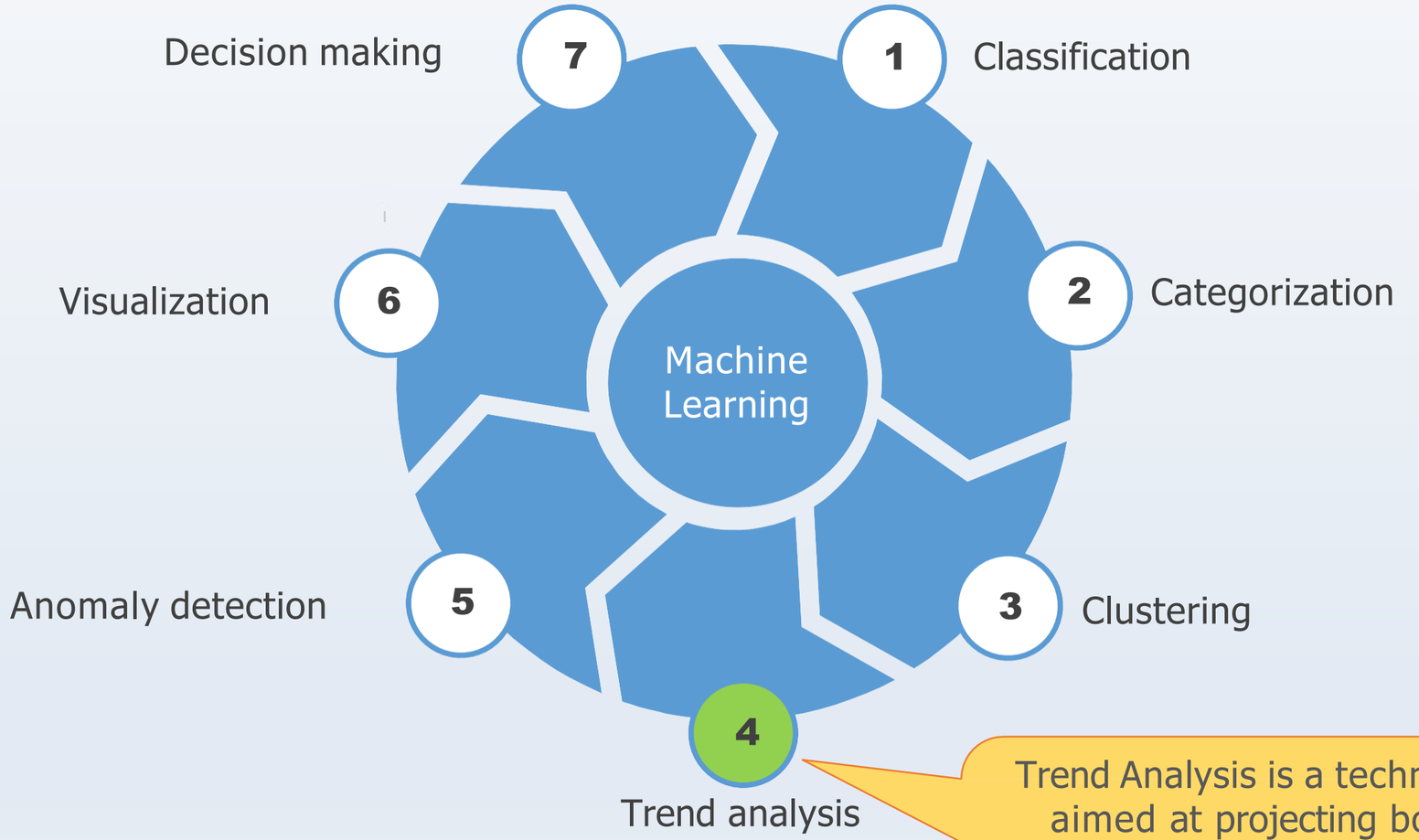


Machine Learning Techniques



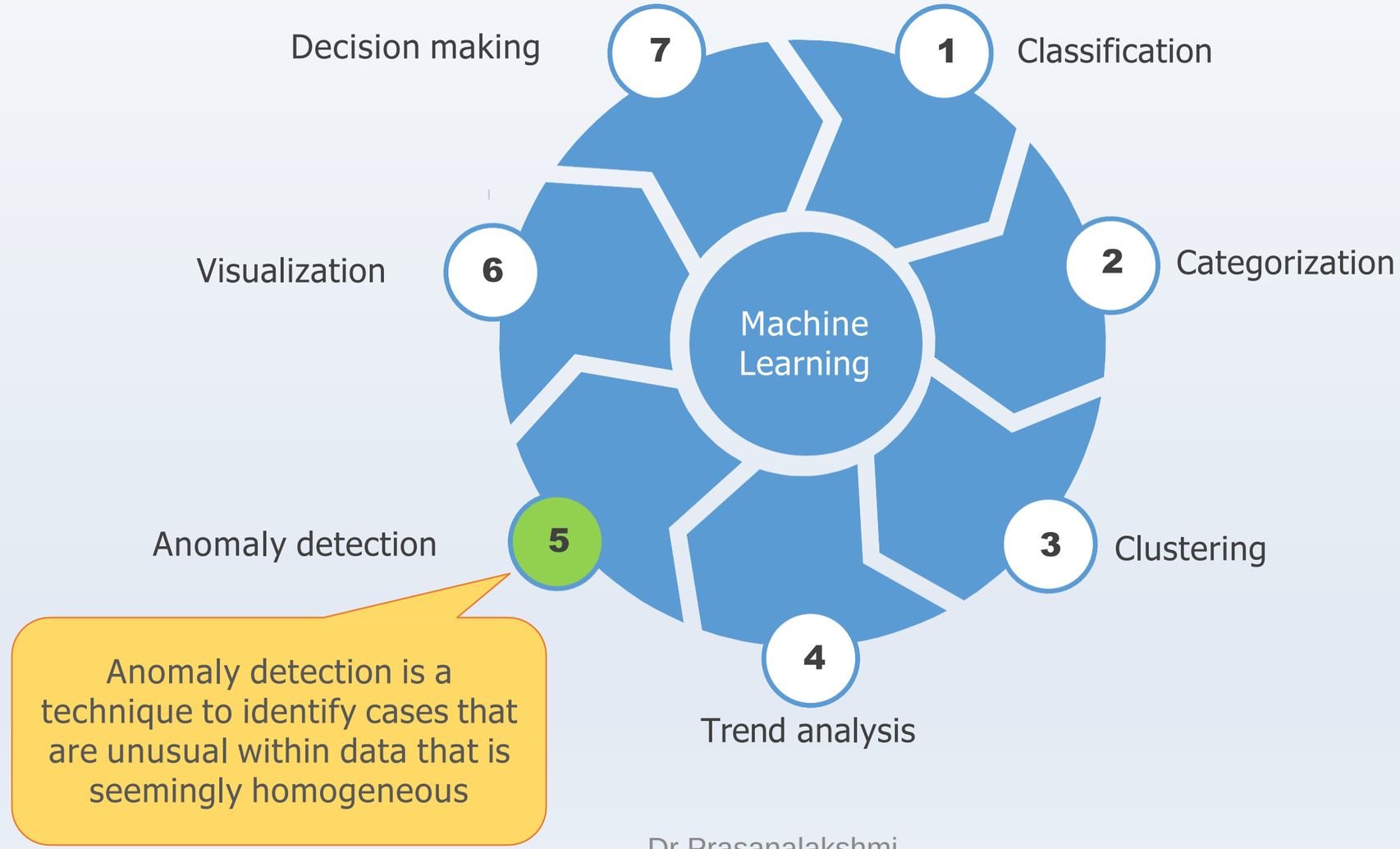
Technique of grouping a set of objects in such a way that objects in the same group are more similar to each other than to those in other groups

Machine Learning Techniques



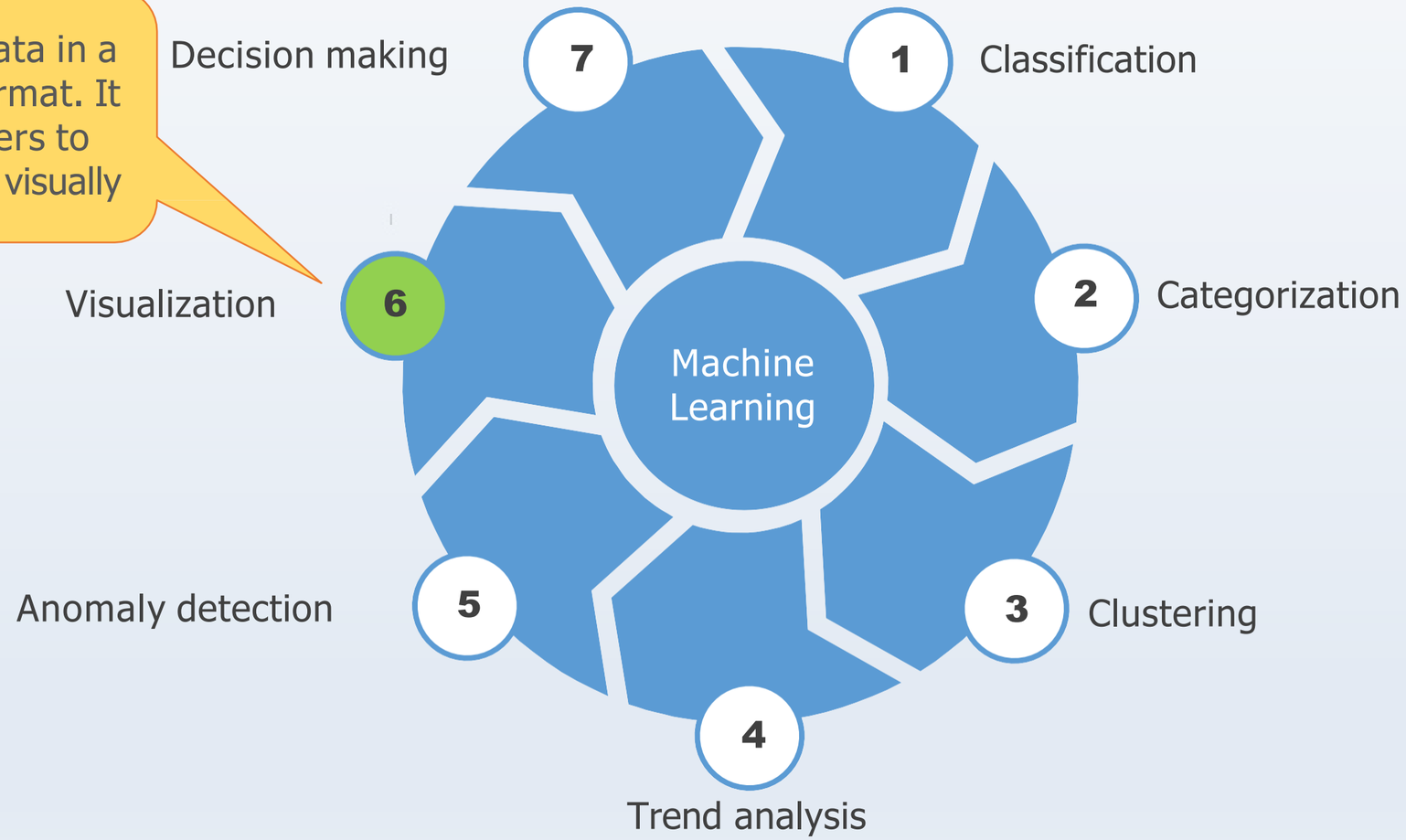
Trend Analysis is a technique aimed at projecting both current and future movement of events through use of time series data analysis

Machine Learning Techniques



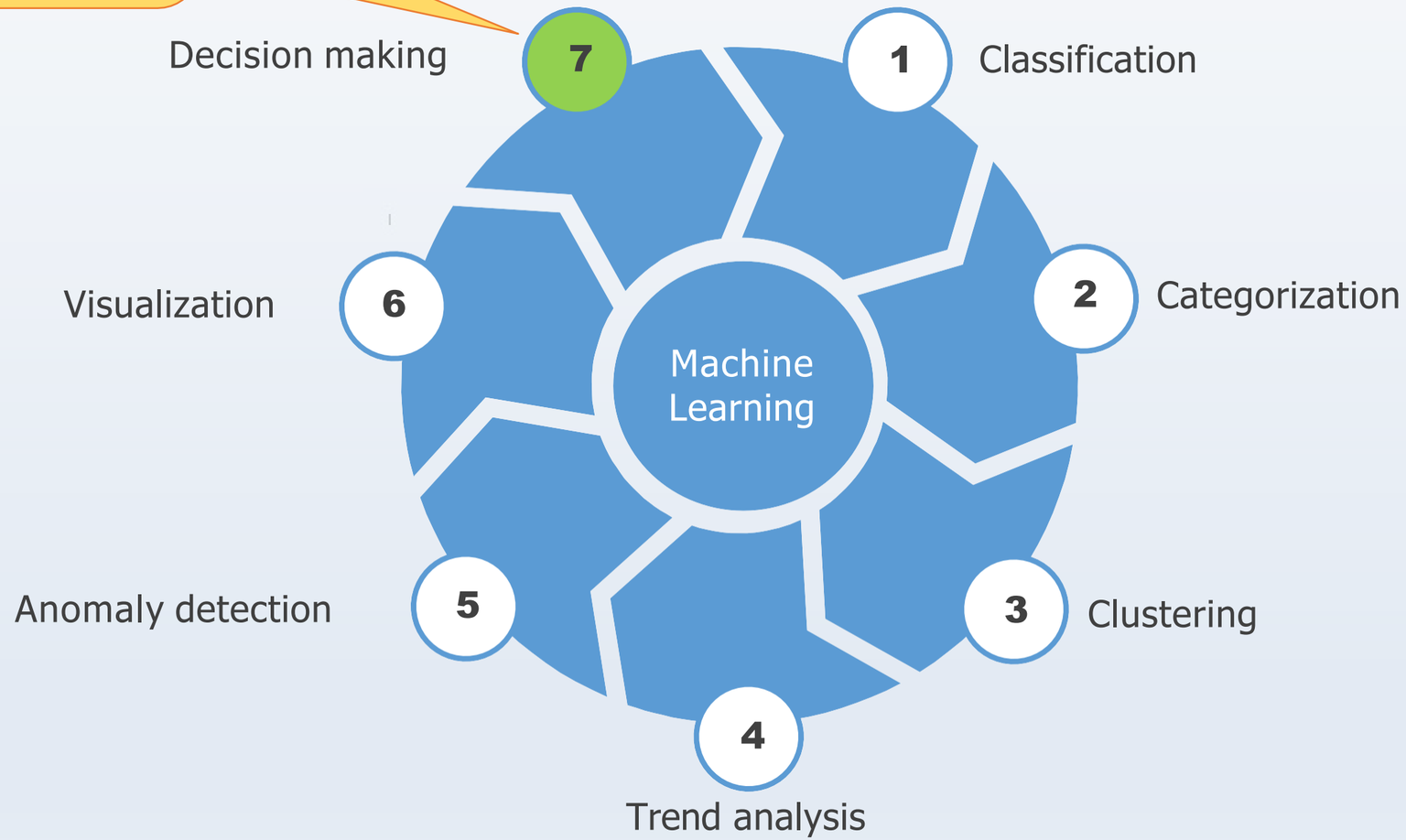
Machine Learning Techniques

Technique to present data in a pictorial or graphical format. It enables decision makers to see analytics presented visually



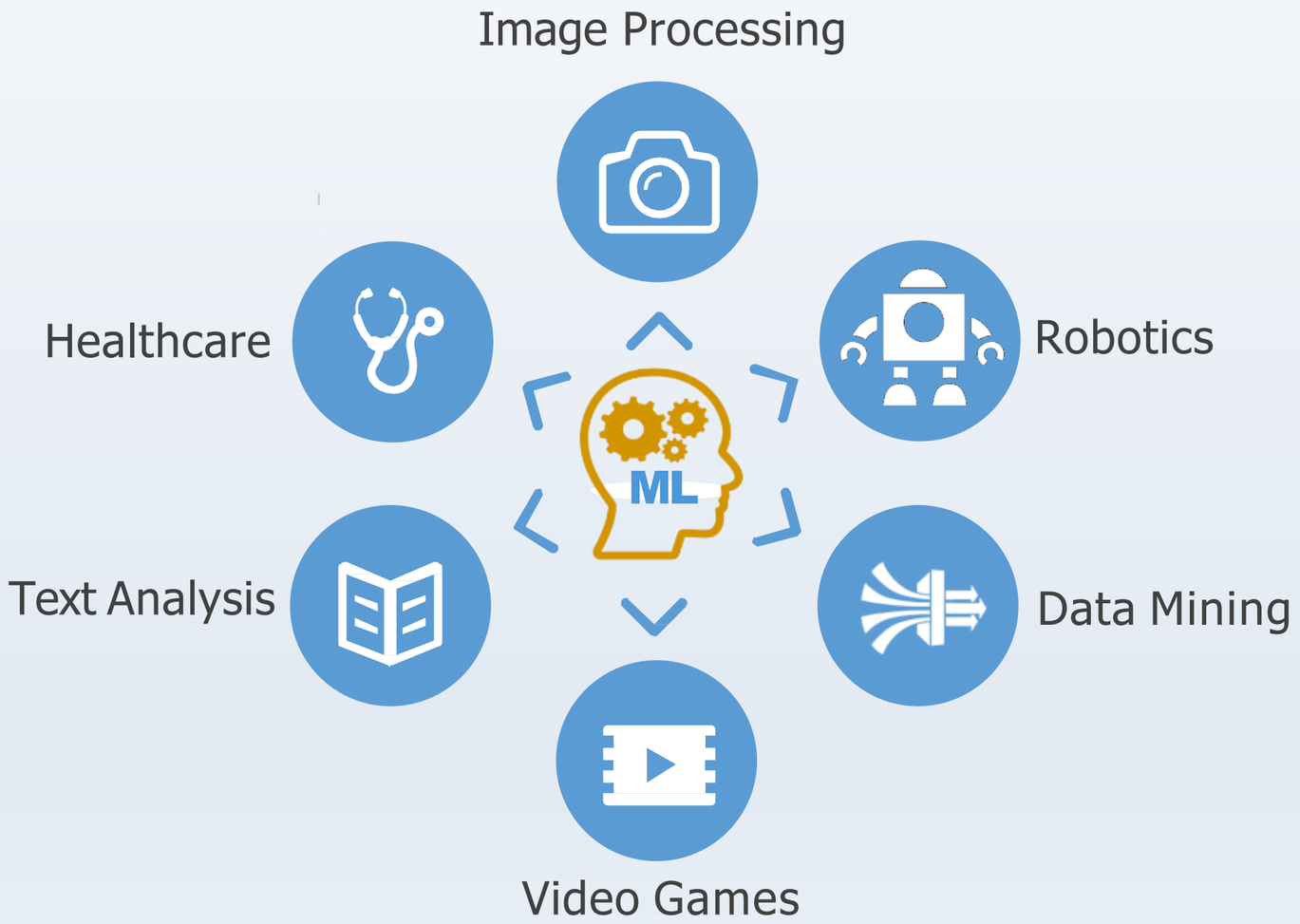
Machine Learning Techniques

A technique/skill which provides you with the ability to influence managerial decisions with data as evidence for those possibilities



Applications of Machine Learning

Artificial intelligence and Machine learning are being increasingly used in various functions such as:



Machine Learning

Lesson 2: Data Wrangling and Manipulation

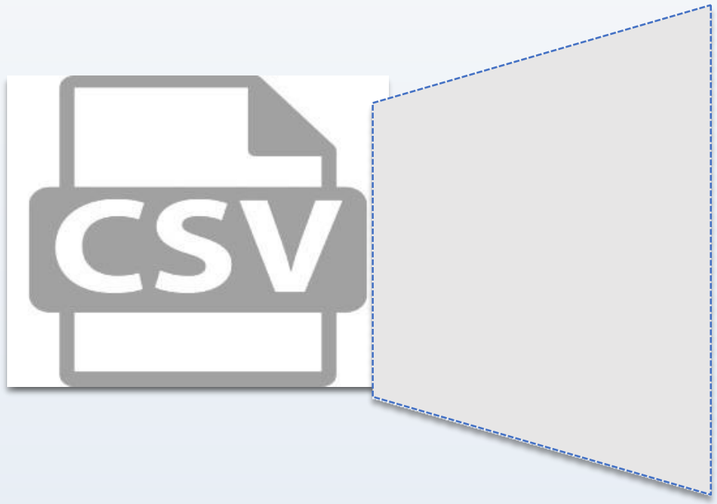
Dr.Prasanalakshmi

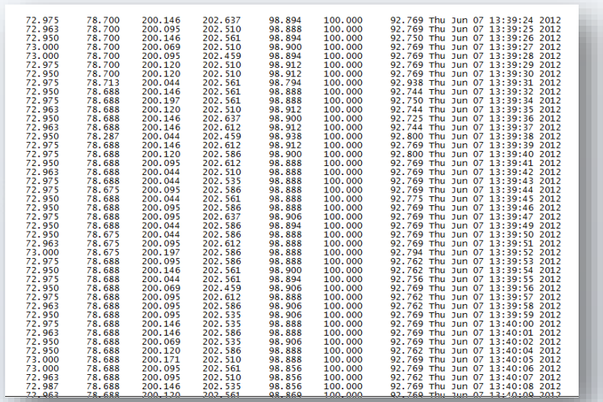
Data Preprocessing

Topic 1: Data Exploration

Loading .csv File in Python

Before starting with a dataset, the first step is to load the dataset. Below is the code for the same:





CSV File

Program Data

Program

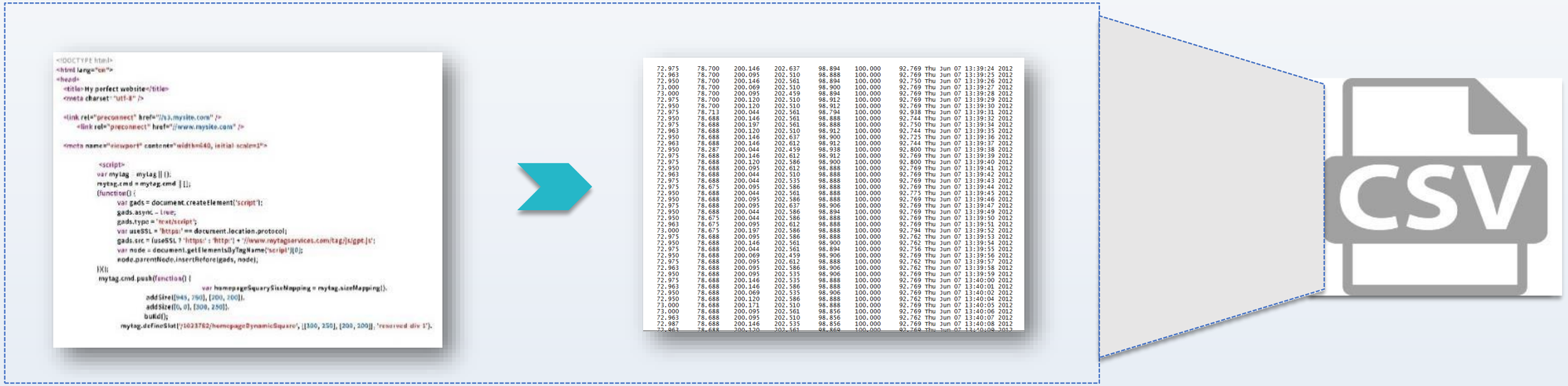
Code

```
df = pandas.read_csv("/home/simpy/Datasets/BostonHousing.csv")
```

Path to file

Loading Data to .csv File

Below is the code for loading the data within an existing csv file:



Program | **Program Data** | **CSV File**

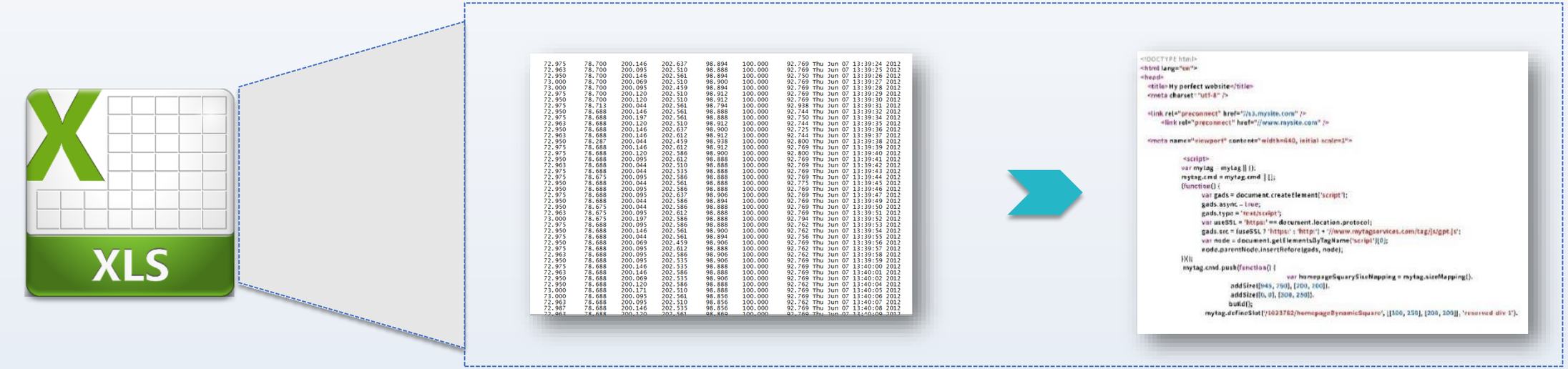
Code

```
df.to_csv("/home/simpy/Datasets/BostonHousing.csv")
```

Path to file

Loading .xlsx File in Python

Below is the code for loading an xlsx file within python:



XLS File

Program Data

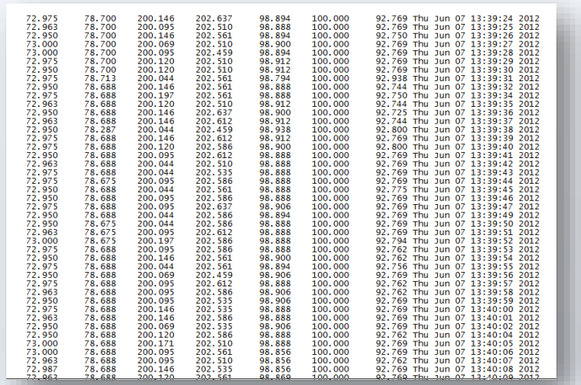
Program

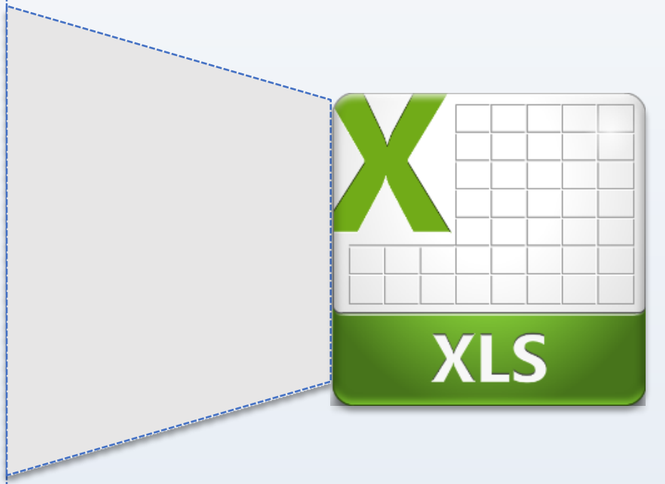
Code

```
df = pandas.read_excel("/home/simpy/Datasets/BostonHousing.xlsx")
```

Loading Data to .xlsx File

Below is the code for loading program data into an existing xlsx file:





Program

Program Data

XLS File

Code

```
df.to_excel("/home/simpy/Datasets/BostonHousing.xlsx")
```

Data Exploration Techniques

- Dimensionality Check
- Type of Dataset
- Slicing and Indexing
- Identifying Unique Elements
- Value Extraction
- Feature Mean
- Feature Median
- Feature Mode

The shape attribute returns a two-item tuple (number of rows and the number of columns) for the data frame. For a Series, it returns a one-item tuple.

Code

```
df.shape
```

```
Out[12]: (506, 14)
```


Data Exploration Techniques (Contd.)

- Dimensionality Check
- Type of Dataset
- Slicing and Indexing
- Identifying Unique Elements
- Value Extraction
- Feature Mean
- Feature Median
- Feature Mode

You can use the type ( ) in python to return the type of object.

Checking the type of data frame:

Code

```
type(df)
```

```
Out[13]:  pandas.core.frame.DataFrame
```

Checking the type of a column (chas) within a data frame:

Code

```
df['chas'].dtype
```

```
Out[21]:  dtype('int64')
```


Data Exploration Techniques (Contd.)

- Dimensionality Check
- Type of Dataset
- Slicing and Indexing
- Identifying Unique Elements
- Value Extraction
- Feature Mean
- Feature Median
- Feature Mode

Using mean( )on the data frame will return mean of the data frame across all the columns.

Code

```
df.mean()
```

```
Out[35]: crim        3.613524
         zn         11.363636
         indus      11.136779
         chas        0.069170
         nox         0.554695
         rm          6.284634
         age        68.574901
         dis         3.795043
         rad         9.549407
         tax       408.237154
         ptratio    18.455534
         b         356.674032
         lstat      12.653063
         medv       22.532806
         dtype: float64
```


Data Exploration Techniques (Contd.)

- Dimensionality Check
- Type of Dataset
- Slicing and Indexing
- Identifying Unique Elements
- Value Extraction
- Feature Mean
- Feature Median
- Feature Mode

Using median( )on the data frame will return median values of the data frame across all the columns.

Code

```
df.median()
```

```
Out[36]: crim          0.25651
         zn            0.00000
         indus         9.69000
         chas          0.00000
         nox           0.53800
         rm            6.20850
         age          77.50000
         dis           3.20745
         rad           5.00000
         tax         330.00000
         ptratio      19.05000
         b           391.44000
         lstat        11.36000
         medv         21.20000
         dtype: float64
```


Data Exploration Techniques (Contd.)

- Dimensionality Check
- Type of Dataset
- Slicing and Indexing
- Identifying Unique Elements
- Value Extraction
- Feature Mean
- Feature Median
- Feature Mode

Using mode( )on the data frame will return mode values of the data frame across all the columns, rows with axis=0 and axis = 1, respectively.

Code

```
df.mode(axis=0)
```

Out[40]:

	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	b	lstat	medv
0	0.01501	0.0	18.1	0.0	0.538	5.713	100.0	3.4952	24.0	666.0	20.2	396.9	6.36	50.0
1	14.33370	NaN	NaN	NaN	NaN	6.127	NaN	NaN	NaN	NaN	NaN	NaN	7.79	NaN
2	NaN	NaN	NaN	NaN	NaN	6.167	NaN	NaN	NaN	NaN	NaN	NaN	8.05	NaN
3	NaN	NaN	NaN	NaN	NaN	6.229	NaN	NaN	NaN	NaN	NaN	NaN	14.10	NaN
4	NaN	NaN	NaN	NaN	NaN	6.405	NaN	NaN	NaN	NaN	NaN	NaN	18.13	NaN
5	NaN	NaN	NaN	NaN	NaN	6.417	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Plotting a Heatmap with Seaborn

Below is the code for plotting a heatmap within Python:

Code

```
import matplotlib.pyplot as plt
import seaborn as sns
correlations = df.corr()
sns.heatmap(data = correlations,square = True, cmap = "bwr")


plt.yticks(rotation=0)
plt.xticks(rotation=90)
```

Rectangular dataset (2D dataset that can be coerced into an ndarray)

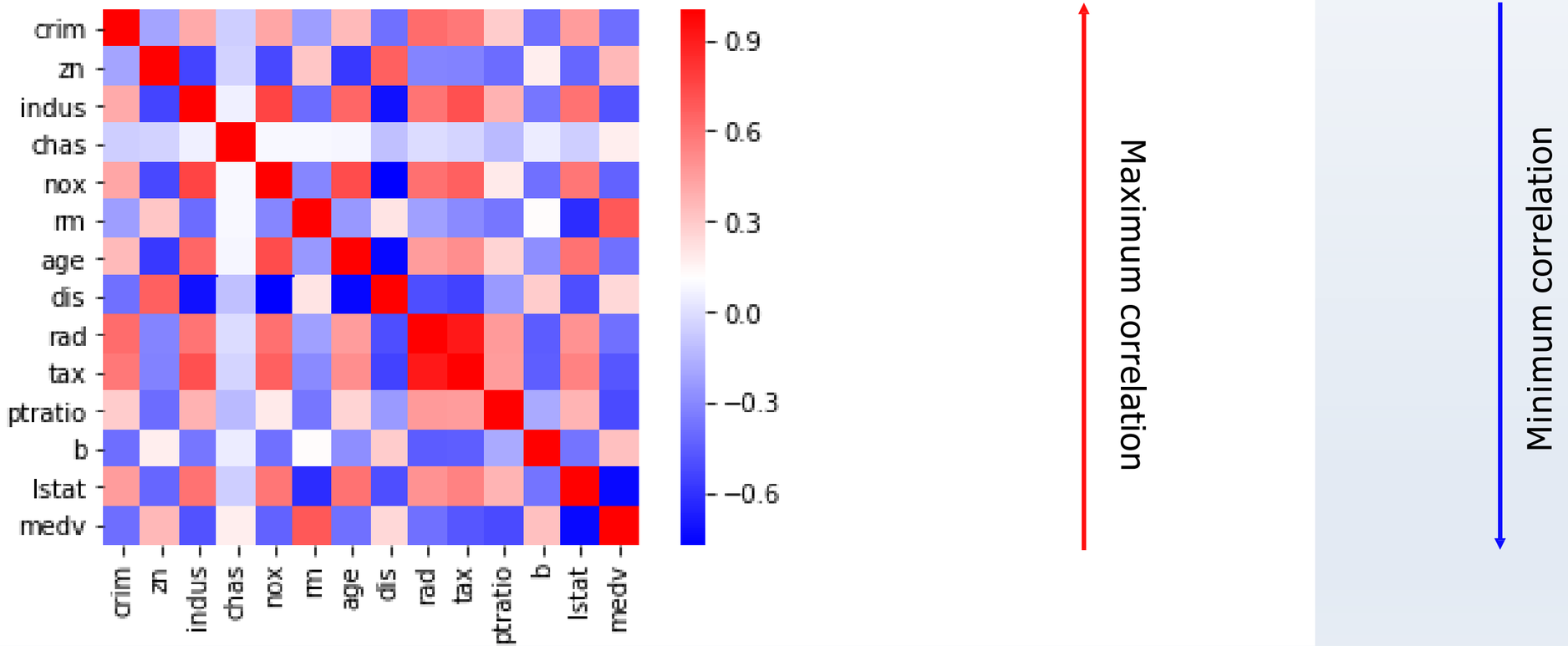
If True, set the Axes aspect to “equal” so each cell will be square-shaped

Matplotlib colormap name or object, or list of colors

Plotting a Heatmap with Seaborn (Contd.)

Below is the heatmap obtained, where, approaching red colour means maximum correlation and approaching blue means minimal correlation.

```
Out[33]: (array([ 0.5,  1.5,  2.5,  3.5,  4.5,  5.5,  6.5,  7.5,  8.5,  9.5, 10.5,
                 11.5, 12.5, 13.5]), <a list of 14 Text xticklabel objects>)
```



Data Import

The first step is to import the data as a part of exploration.

Code

```
df1 = pandas.read_csv("mtcars.csv")
```

Out[35]:

	model	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
0	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
1	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
2	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
3	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
4	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
5	Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
6	Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
7	Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
8	Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
9	Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4

Dr.Prasanalakshmi

Data Exploration

The shape property is usually used to get the current shape of an array/df.

- Dimensionality Check
- Type of Dataset
- Identifying mean value

Code

```
df1.shape
```

```
Out[36]: (32, 12)
```


Data Exploration

type(), returns type of the given object.

- Dimensionality Check
- Type of Dataset
- Identifying mean value

Code

```
type(df1)
```

```
Out[37]: pandas.core.frame.DataFrame
```


Data Exploration

mean( ) function can be used to calculate mean/average of a given list of numbers.

- Dimensionality Check
- Type of Dataset
- Identifying mean value

Code

```
df1['hp'].mean()
```

```
Out[44]: 146.6875
```


Identifying Correlation Using a Heatmap

Heatmap function in seaborn is used to plot the correlation matrix.

Code

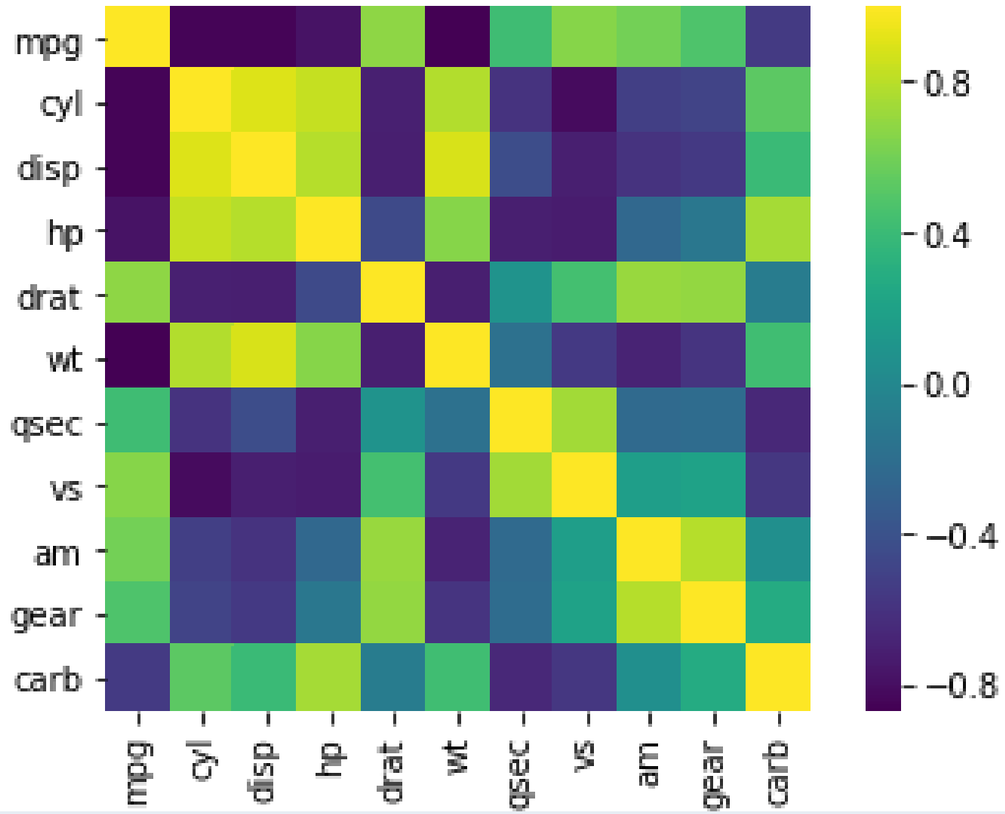
```
import matplotlib.pyplot as plt
import seaborn as sns
correlations = df1.corr()
sns.heatmap(data = correlations,square = True, cmap = "viridis")


plt.yticks(rotation=0)
plt.xticks(rotation=90)
```


Identifying Correlation Using a Heatmap

Graphical representation of data where the individual values contained in a matrix are represented in colors.

```
Out[45]: (array([ 0.5,  1.5,  2.5,  3.5,  4.5,  5.5,  6.5,  7.5,  8.5,  9.5, 10.5]),
          <a list of 11 Text xticklabel objects>)
```



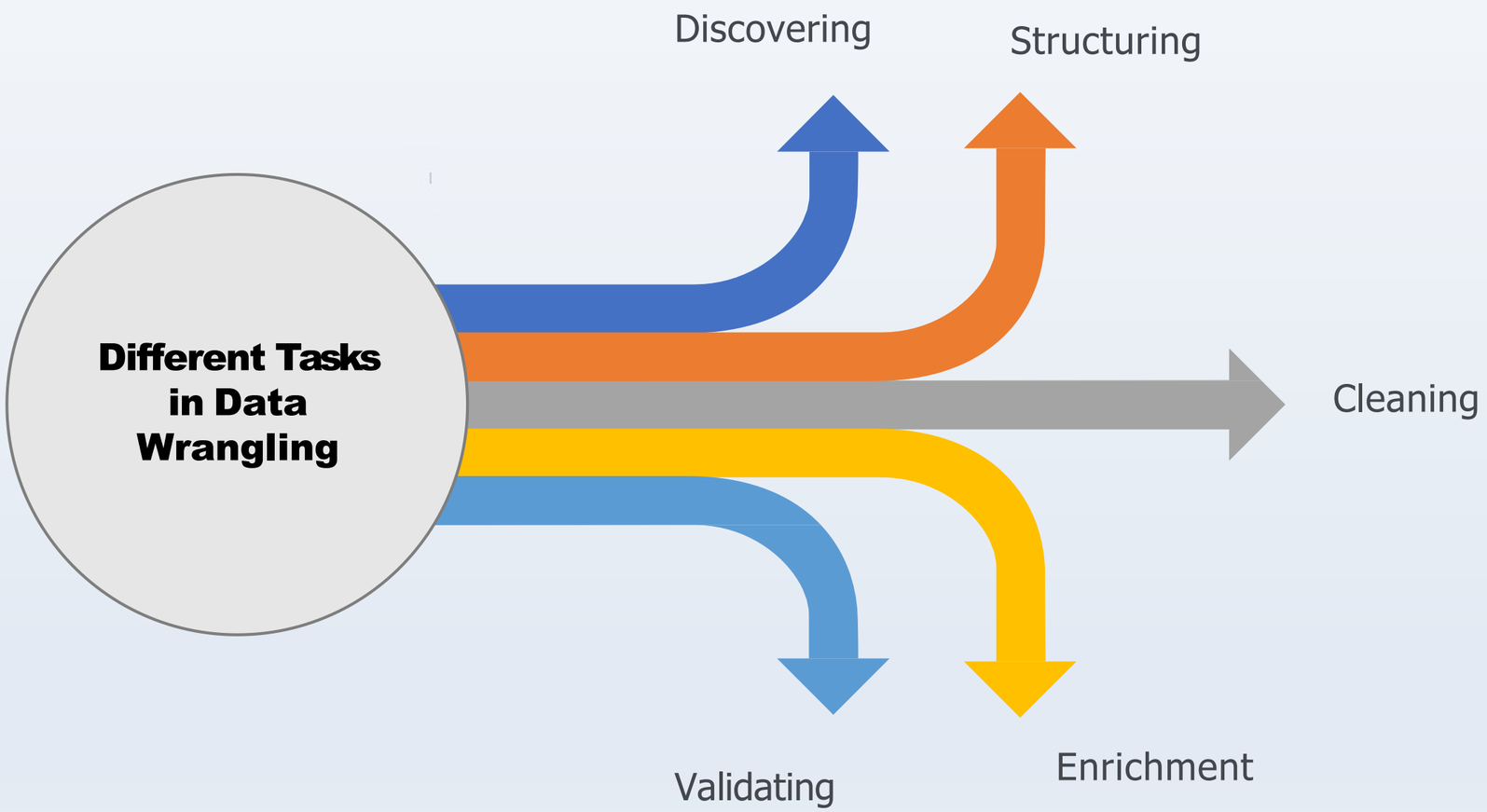
From the adjacent map, you can clearly see that cylinder (cyl) and displacement (disp) are the most correlated features.

Data Preprocessing

Topic 2: Data Wrangling

Data Wrangling

The process of manually converting or mapping data from one raw format into another format is called data wrangling. This includes munging and data visualization.



Need of Data Wrangling

Following are the problems that can be avoided with wrangled data:

- Missing data, a very common problem
- Presence of noisy data (erroneous data and outliers)
- Inconsistent data
- Develop a more accurate model
- Prevent data leakage

Missing Values in a Dataset

Consider a random dataset given below, illustrating missing values.

PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	0	3	male	22	1	0	A/5 21171	7.25		S
2	1	1	female	38	1	0	PC 17599	71.2833	C85	C
3	1	3	female	26	0	0	STON/O2. 3101282	7.925		S
4	1	1	female	35	1	0	113803	53.1	C123	S
5	0	3	male	35	0	0	373450	8.05		S
6	0	3	male		0	0	330877	8.4583		Q

MACHINE LEARNING

Missing Value Detection

Consider a dataset below, imported as df1 within Python, having some missing values.

	Prefix	Assignment	Tutorial	Midterm	TakeHome	Final
0	5	57.14	34.09	64.38	51.48	52.50
1	8	95.05	105.49	67.50	99.07	68.33
2	8	83.70	83.17	30.00	63.15	48.89
3	7	81.22	96.06	49.38	105.93	80.56
4	8	91.32	93.64	95.00	107.41	73.89
5	7	95.00	92.58	93.12	97.78	68.06
6	8	95.05	102.99	56.25	99.07	50.00
7	7	72.85	86.85	60.00	NaN	56.11
8	8	84.26	93.10	47.50	18.52	50.83

Detecting missing values

Code

```
df1.isna().any()
```

```
Out[16]: Prefix          False
         Assignment      False
         Tutorial        False
         Midterm         False
         TakeHome         True
         Final           False
         dtype: bool
```


Missing Value Treatment

Mean Imputation: Replace the missing value with variable's mean

Code

```
from sklearn.preprocessing import Imputer
mean_imputer =
Imputer(missing_values=np.nan,strategy='mean',axis=1)
mean_imputer = mean_imputer.fit(df1)
imputed_df = mean_imputer.transform(df1.values)
df1 = pd.DataFrame(data=imputed_df,columns=cols)
df1
```

Out[75]:

	Prefix	Assignment	Tutorial	Midterm	TakeHome	Final
0	5.0	57.14	34.09	64.38	51.480	52.50
1	8.0	95.05	105.49	67.50	99.070	68.33

Missing Value Treatment (Contd.)

Mean Imputation: Replace the missing value with variable's mean

Median Imputation: Replace the missing value with variable's median

Code

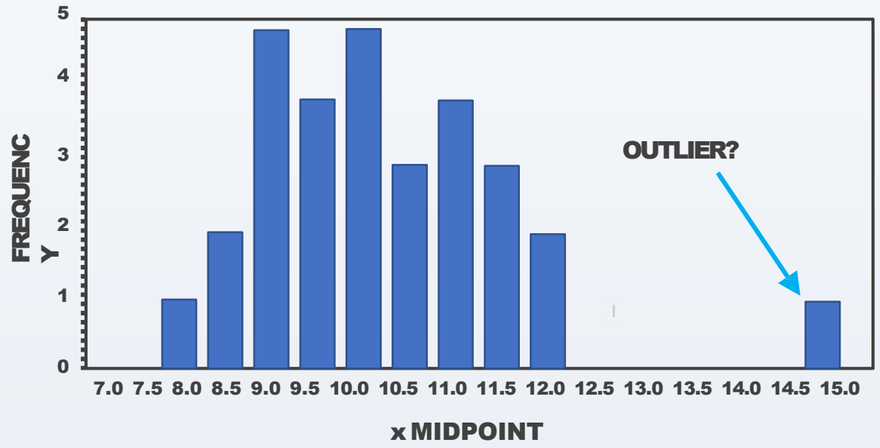
```
from sklearn.preprocessing import Imputer
median_imputer=Imputer(missing_values=np.nan,strategy
='median',axis=1)
median_imputer = median_imputer.fit(df1)
imputed_df = median_imputer.transform(df1.values)
df1 = pd.DataFrame(data=imputed_df,columns=cols)
df1
```

Out[84]:

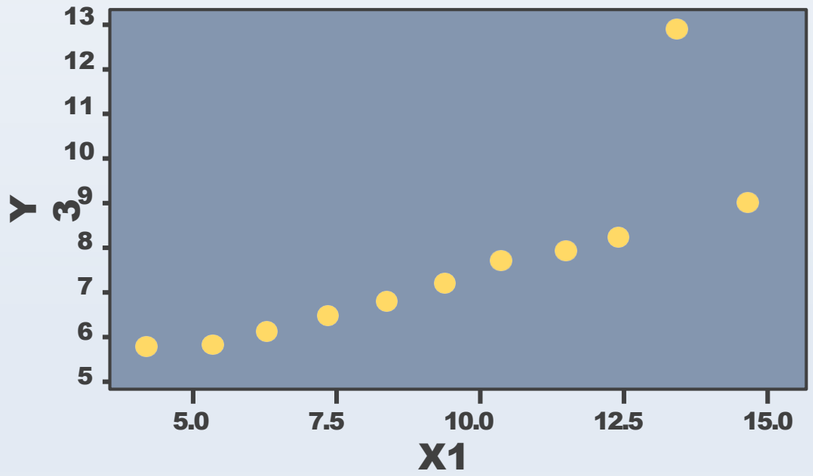
	Prefix	Assignment	Tutorial	Midterm	TakeHome	Final
0	5.0	57.14	34.09	64.38	51.480	52.50
1	8.0	95.05	105.49	67.50	99.070	68.33

Note: *Mean imputation/Median imputation is again model dependent and is valid only on numerical data.*

Outlier Values in a Dataset



An outlier is a value that lies outside the usual observation of values.



Note: *Outliers skew the data when you are trying to do any type of average.*

Dealing with an Outlier

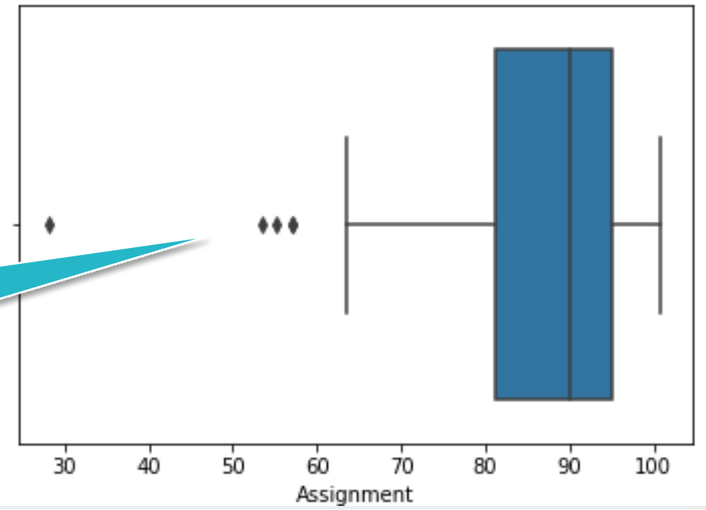
- Outlier Detection
- Outlier Treatment

Detect any outlier in the first column of df1

Code

```
import seaborn as sns
sns.boxplot(x=df1['Assignment'])
```

```
Out[88]: <matplotlib.axes._subplots.AxesSubplot at 0x232d5273da0>
```



Outliers:
Values < 60

Dealing with an Outlier

Outlier Detection

Outlier Treatment

Create a filter based on the boxplot obtained and apply the filter to the data frame

Code

```
filter=df1['Assignment'].values>60
df1_outlier_rem=df1[filter]
df1_outlier_rem
```

Out[106]:

	Prefix	Assignment	Tutorial	Midterm	TakeHome	Final
1	8.0	95.05	105.49	67.50	99.070	68.33
2	8.0	83.70	83.17	30.00	63.150	48.89
3	7.0	81.22	96.06	49.38	105.930	80.56
4	8.0	91.32	93.64	95.00	107.410	73.89
5	7.0	95.00	92.58	93.12	97.780	68.06
6	8.0	95.05	102.99	56.25	99.070	50.00
7	7.0	72.85	86.85	60.00	56.562	56.11

Check for Irregularities

Check for missing values

Code

```
df1['hp'].isna().any()
```

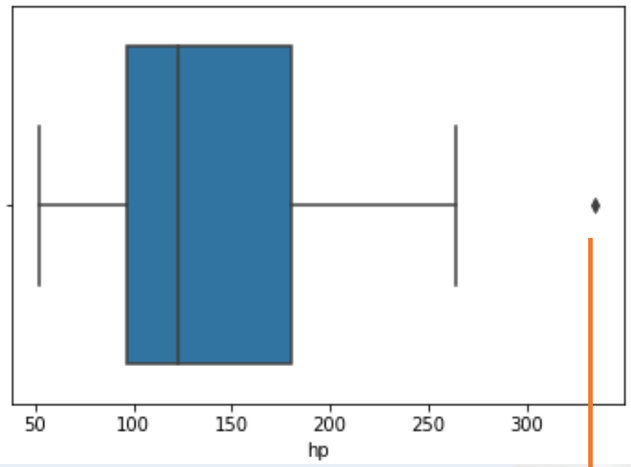
```
Out[114]: False
```

Check for Outliers

Code

```
sns.boxplot(x=df1['hp'])
```

```
Out[113]: <matplotlib.axes._subplots.AxesSubplot at 0x232d55e6f98>
```



Outlier

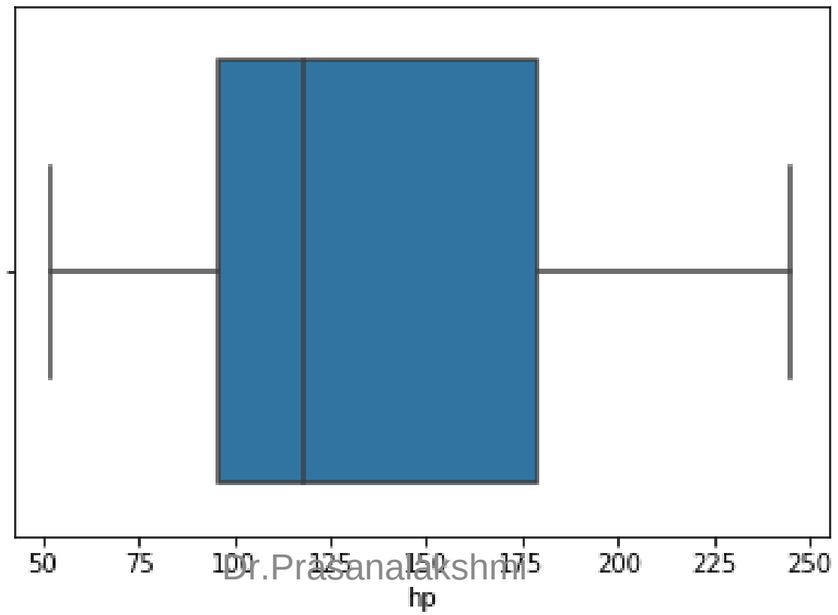
Outlier Treatment

Data with hp>250 is the outlier data. Therefore, you can filter it accordingly.

Code

```
filter = df1['hp']<250
df1_out_rem = df1[filter]
sns.boxplot(x=df2_out_rem['hp'])
```

```
Out[120]: <matplotlib.axes._subplots.AxesSubplot at 0x232d52d6470>
```



Outlier filtered data

Dr.Prasanalakshmi

MACHINE LEARNING

Data Preprocessing

Topic 3: Data Manipulation

Functionalities of Data Object in Python

A data object is a two-dimensional data structure, i.e., data is aligned in a tabular fashion in rows and columns.

- head( )
- tail( ) values( )
- groupby( )
- Concatenation
- Merging

Machine Learning

Lesson 3: Supervised Learning

Supervised Learning

Topic 1: Overview

Supervised Learning

Supervised Learning is a type of machine learning used to train models from labeled training data. It allows you to predict output for future or unseen data.

Examples of Supervised Learning

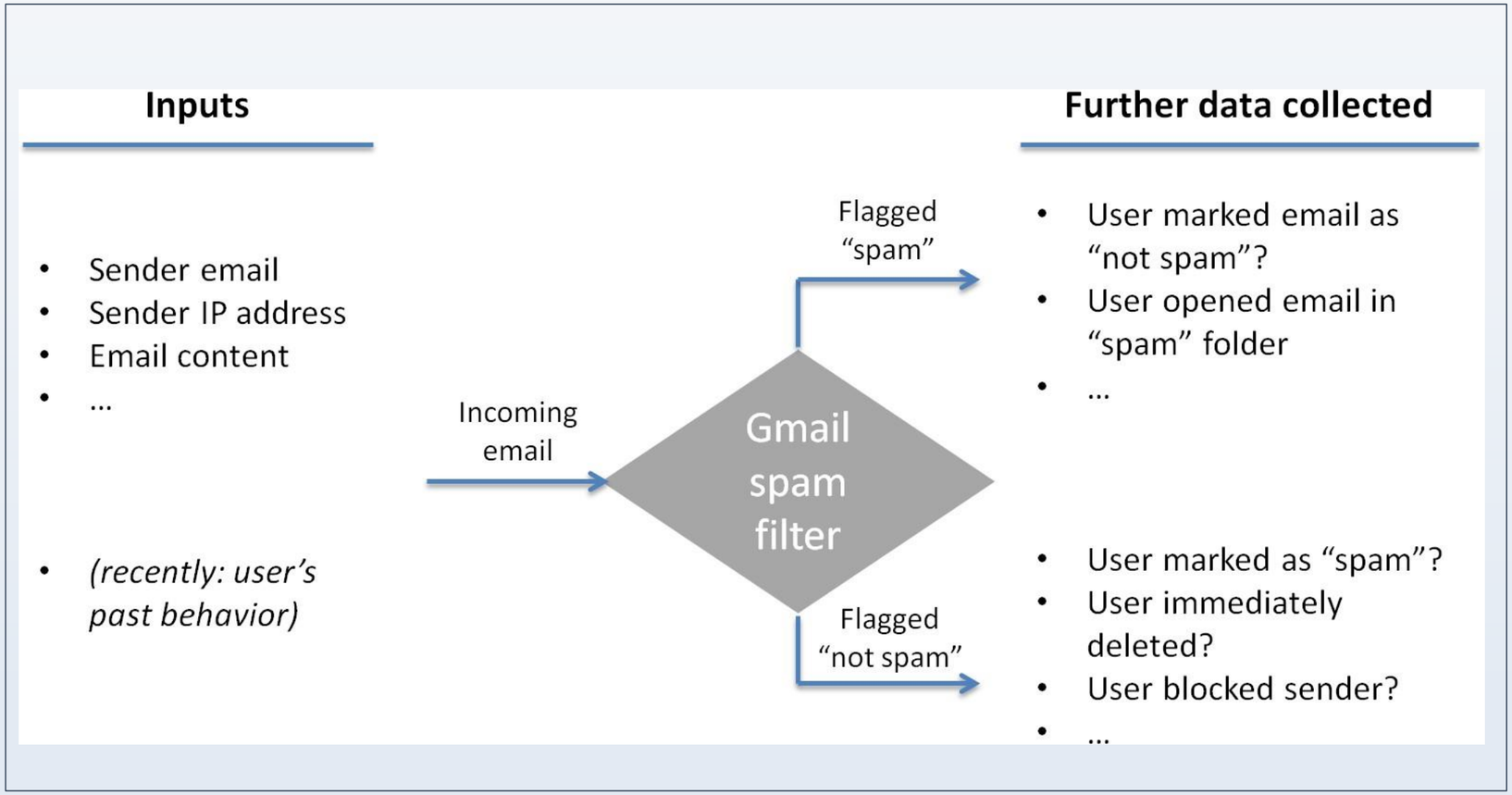
Example 1: Weather Apps

The predictions made by weather apps at a given time are based on prior knowledge and analysis of weather over a period of time for a particular place.

Examples of Supervised Learning (Contd.)

Example 2: Gmail Filters

Gmail filters, a new email into Inbox (normal) or Junk folder (Spam) based on past information of spam.



Supervised Learning: Case Study

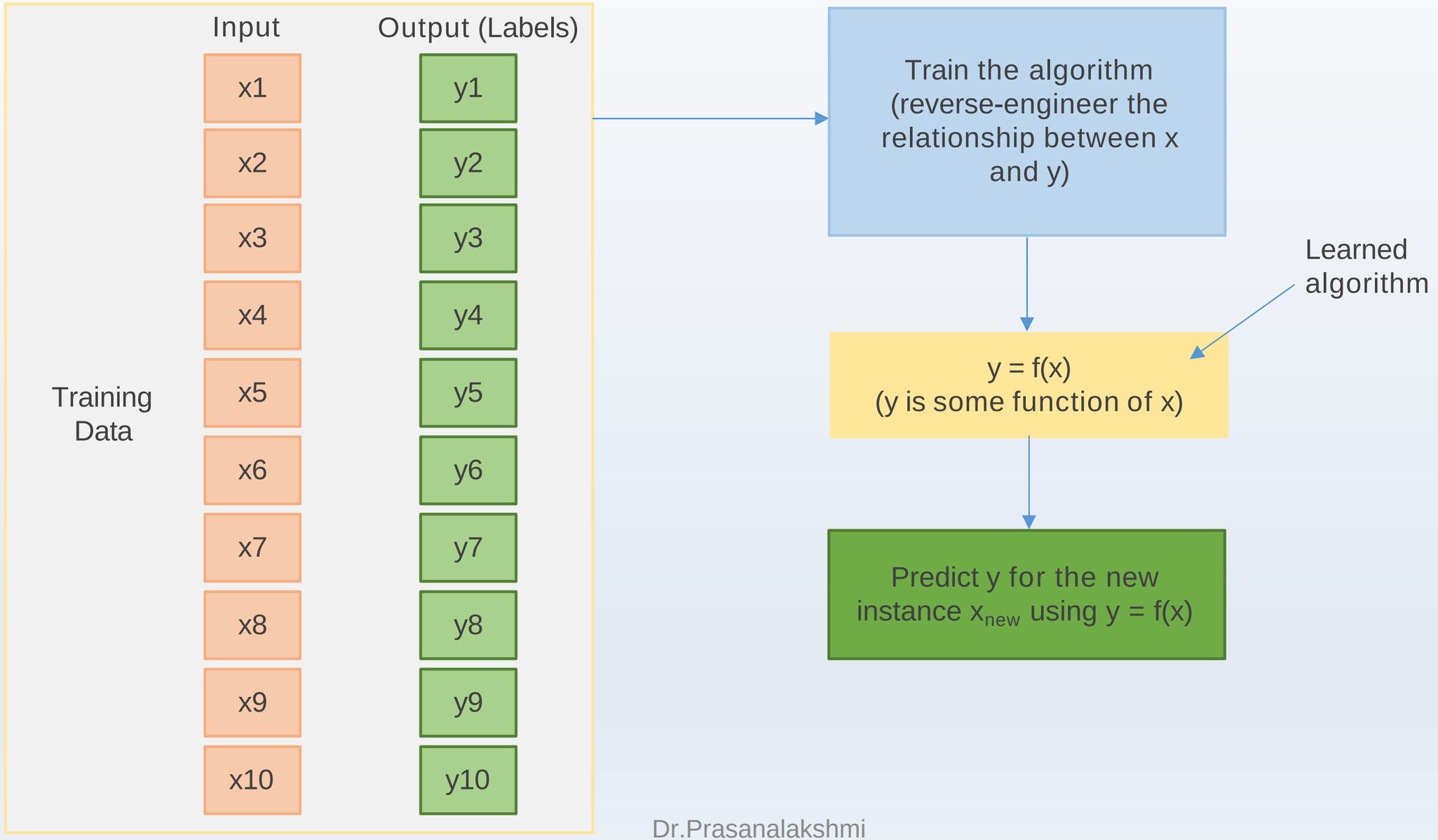
Netflix uses **supervised learning** algorithms to recommend users the shows they may watch based on the viewing history and ratings by similar classes of users

New Input

Algorithm Trained on Historical Data

Predicted Outcome

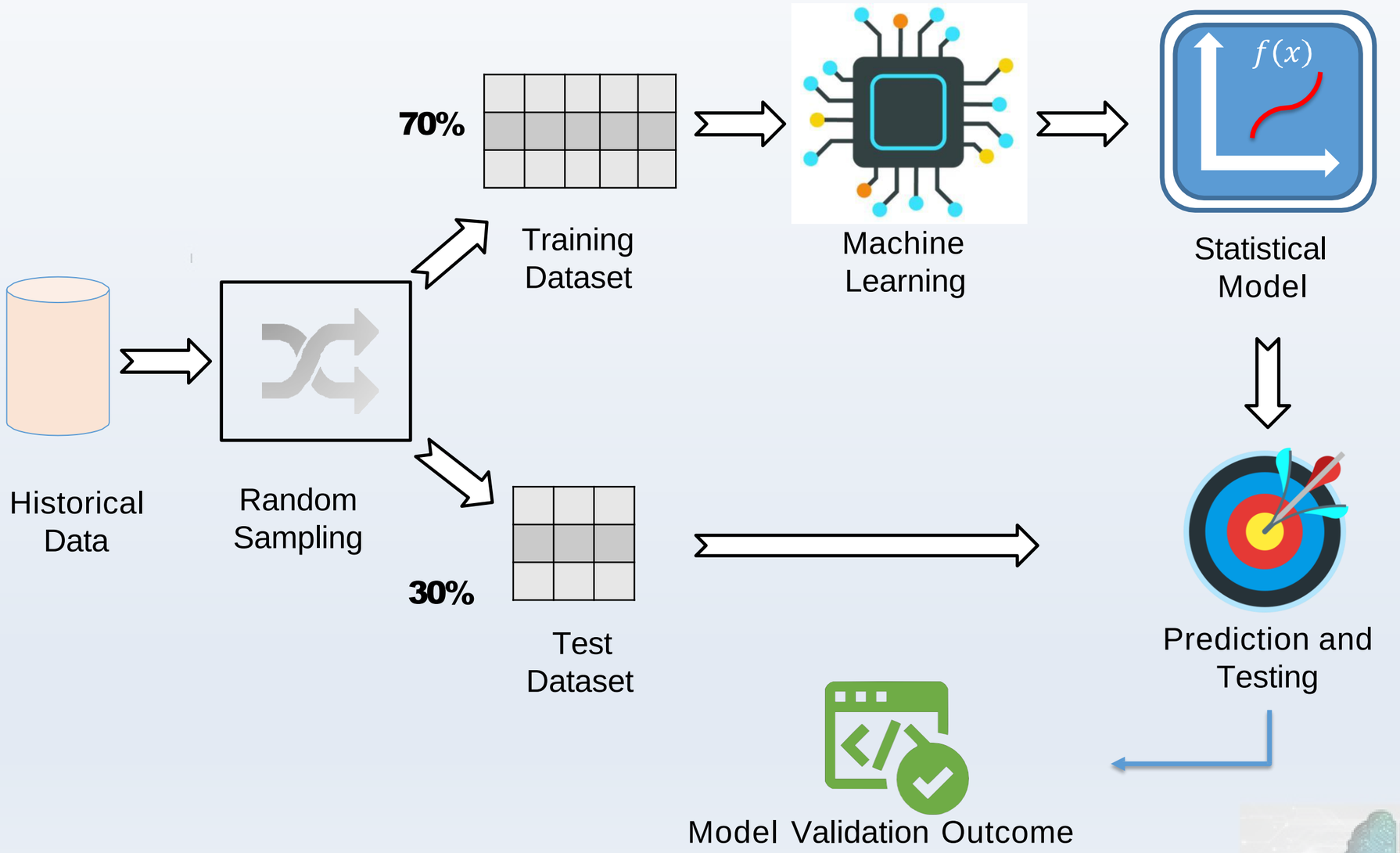
Understanding the Algorithm



Supervised Learning Flow

Training and Testing

Prediction

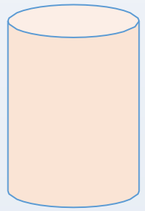


Supervised Learning Flow

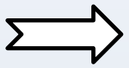
Training and Testing

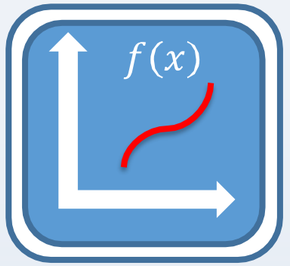
Prediction

Use the learned algorithm y = f(x) to predict production data

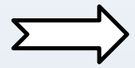


New Data





Model

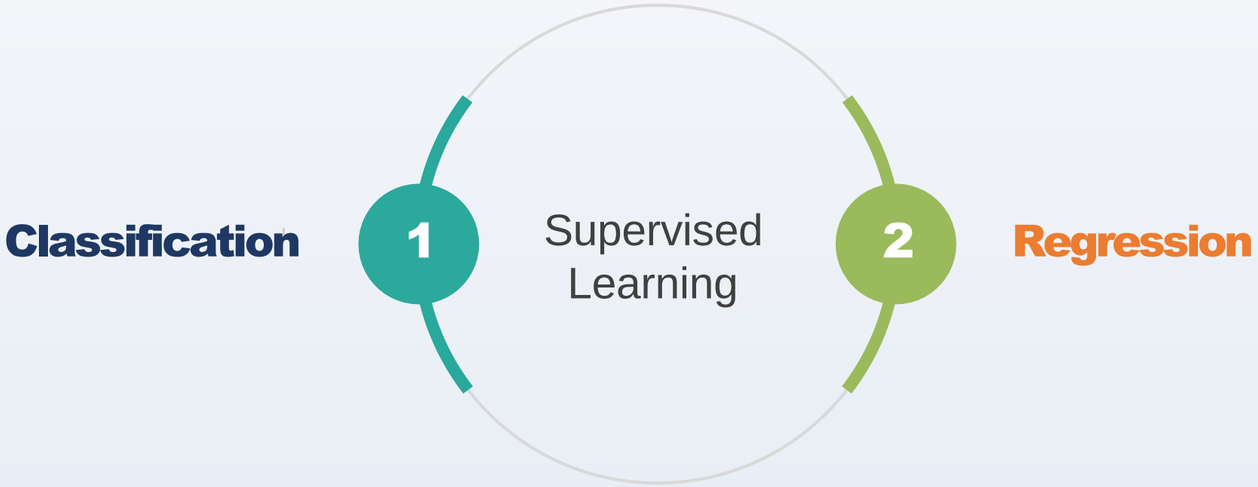


Prediction Outcome

Algorithm prediction can be improved by more training data, capacity, or algorithm redesign.

Topic 2: Types of Supervised Learning

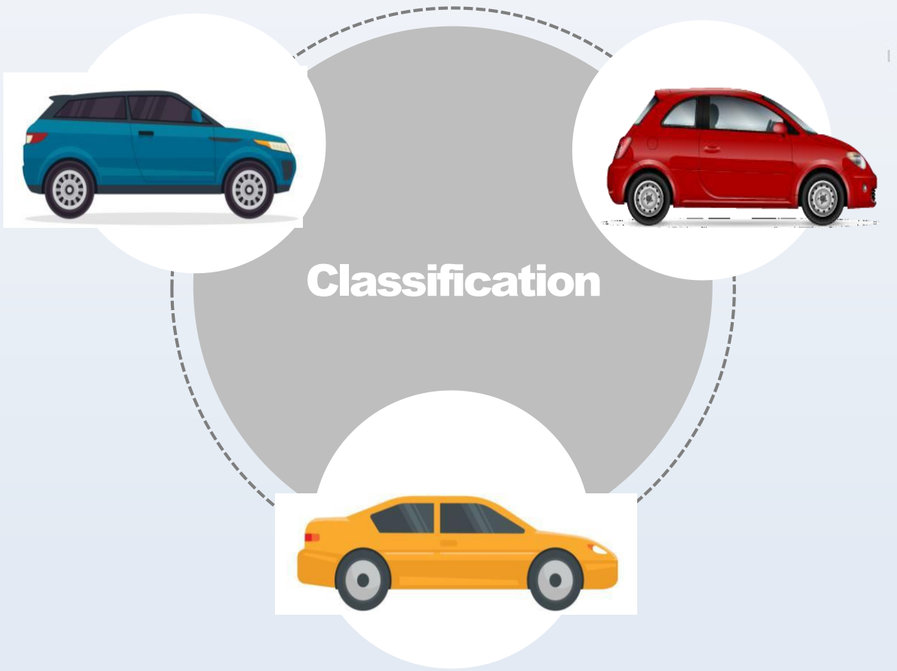
Types of Supervised Learning



In supervised learning, algorithm is selected based on target variable.

Types of Supervised Learning (Contd.)

If target variable is categorical (classes), then use classification algorithm.



In other words, classification is applied when the output has finite and discreet values.

Example: Predict the class of car given its features like horsepower, mileage, weight, colour, etc.

The classifier will build its attributes based on these features.
Analysis has three potential outcomes - Sedan, SUV, or Hatchback

Types of Supervised Learning (Contd.)

If target variable is a continuous numeric variable (100–2000), then use a regression algorithm.

Example: Predict the price of a house given its sq. area, location, no of bedrooms, etc.

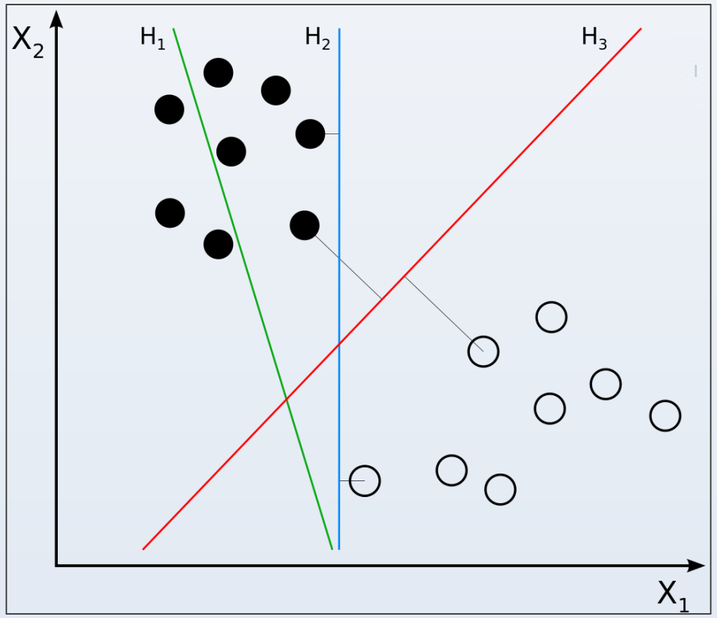
A simple regression algorithm is given below

$$y = w * x + b$$

This shows relationship between price (y) and sq. area (x) where price is a number from a defined range.

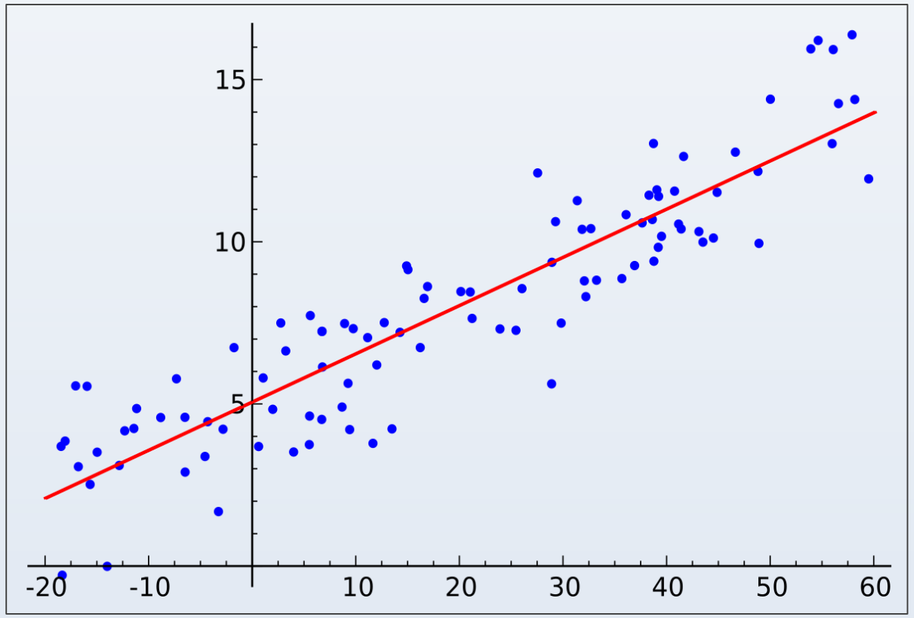
Types of Supervised Learning (Contd.)

Classification



What Class ?

Regression



How much ?

MACHINE LEARNING

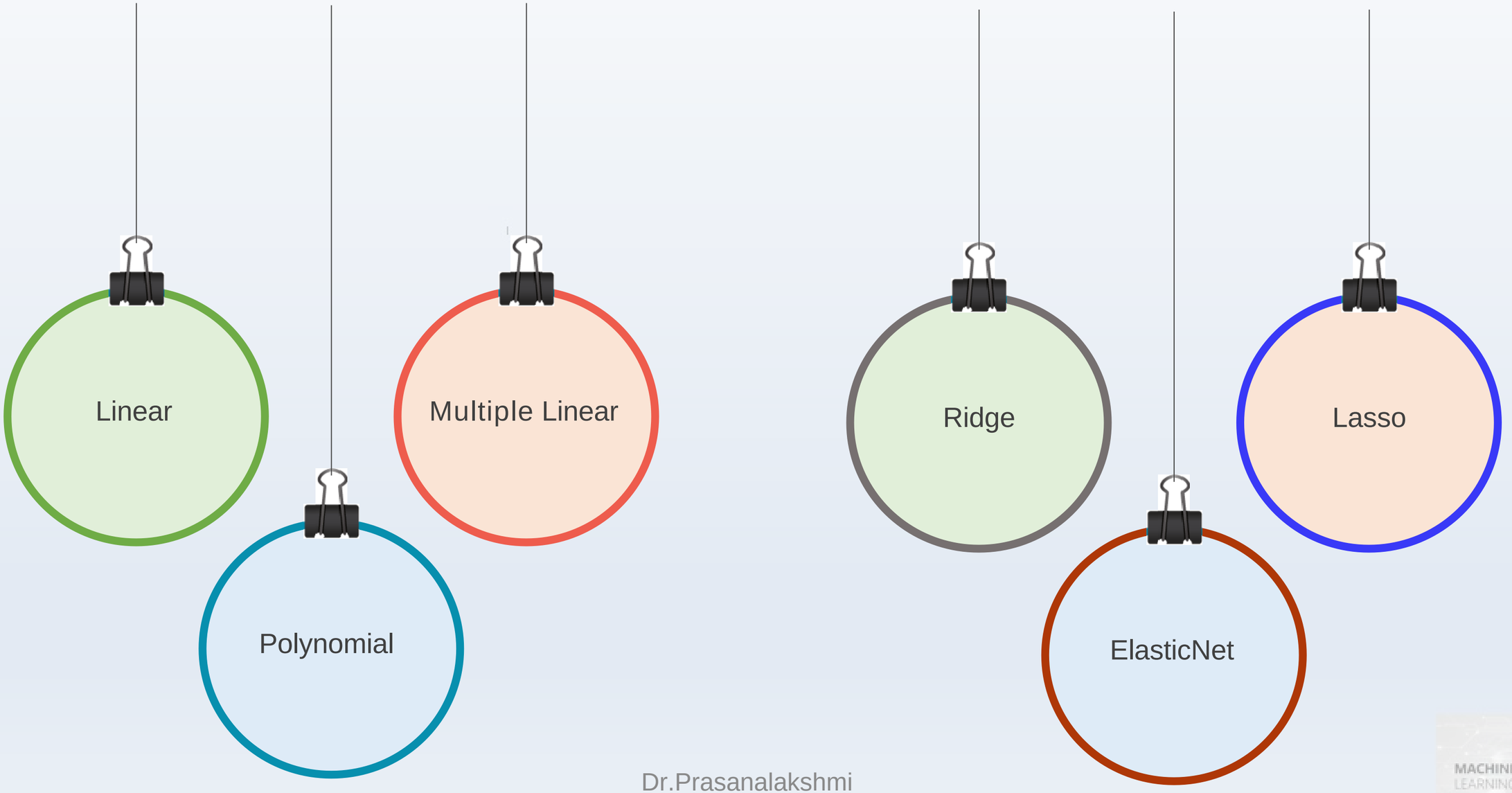
Types of Classification Algorithms

Logistic Regression	Used to estimate discrete values (binary values like 0/1, yes/no, true/false) based on given set of independent variable(s)
Decision Trees	Decision Trees make sequential, hierarchical decisions about the outcome variable based on the predictor data
Random Forest	Random Forest is an ensemble of decision trees. It gives better prediction and accuracy than decision tree
Naive Bayes Classifier	Based on Bayes theorem and works with an assumption that features are independent
Support Vector Machines	SVM draws hyperplane in a feature space that separates instances into different categories with margins in between as far apart as possible

Supervised Learning

Topic 3: Types of Regression Algorithms

Types of regression algorithms



Types of Regression Algorithms

- Linear Regression
- Multiple Linear Regression
- Polynomial Regression
- Ridge Regression
- Lasso Regression
- ElasticNet Regression

Linear Regression is a statistical model used to predict the relationship between independent and dependent variables denoted by x and y respectively

Examine 2 factors

1

How closely are x and y related ?

Linear regression gives a number between -1 and 1 indicating the strength of correlation between the two variables

1 : no correlation
2 : positively correlated
-1 : negatively correlated

2

Prediction

When the relationship between x and y is known, use this to predict future values of y for a value of x

This is done by fitting a regression line and represented by a linear equation:

$$y = a * x + b$$

Types of Regression Algorithms (Contd.)

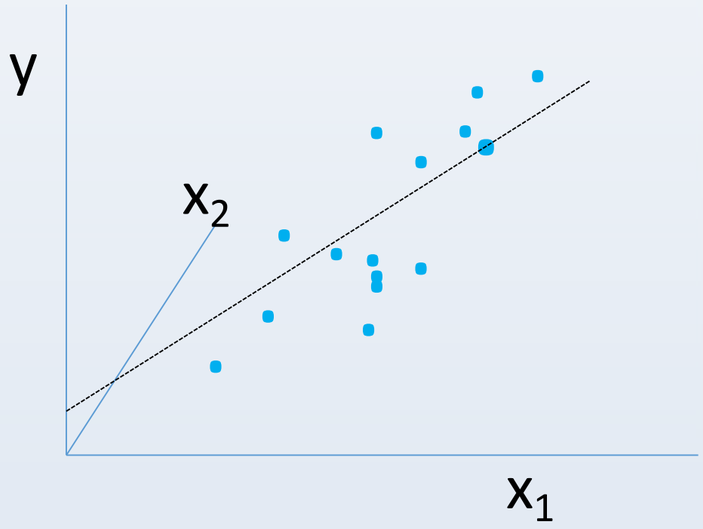
- Linear Regression
- **Multiple Linear Regression**
- Polynomial Regression
- Ridge Regression
- Lasso Regression
- ElasticNet Regression

Multiple linear regression is a statistical technique used to predict the outcome of a response variable through several explanatory variables and model the relationships between them.

Equation for MLR

$$Y = m_1 * x_1 + m_2 * x_2 + m_3 * x_3 + \ldots\ldots + m_n * x_n + c$$

- Y: Dependent Variable
- $m_1, m_2, m_3 \ldots m_n$: Slopes
- c: Coefficient



Types of Regression Algorithms (Contd.)

- Linear Regression
- Multiple Linear Regression
- Polynomial Regression
- Ridge Regression
- Lasso Regression
- ElasticNet Regression

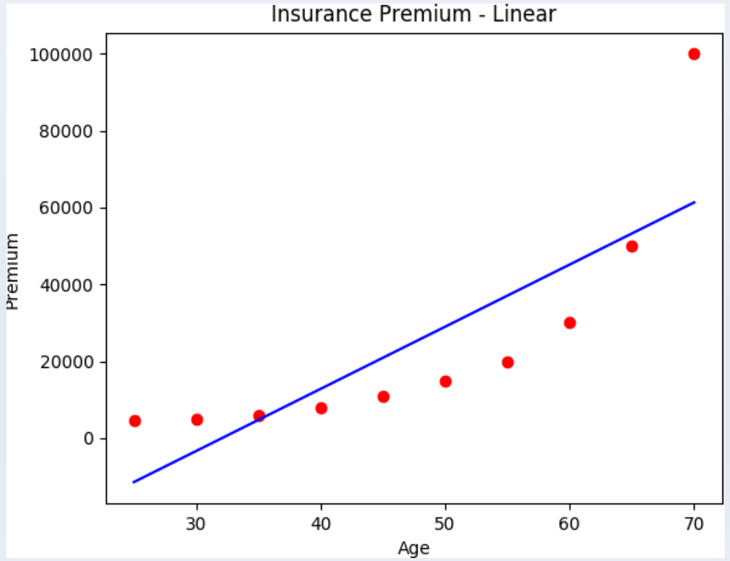
Polynomial regression is applied when data is not formed in a straight line.
It is used to fit a linear model to non-linear data by creating new features from powers of non-linear features.

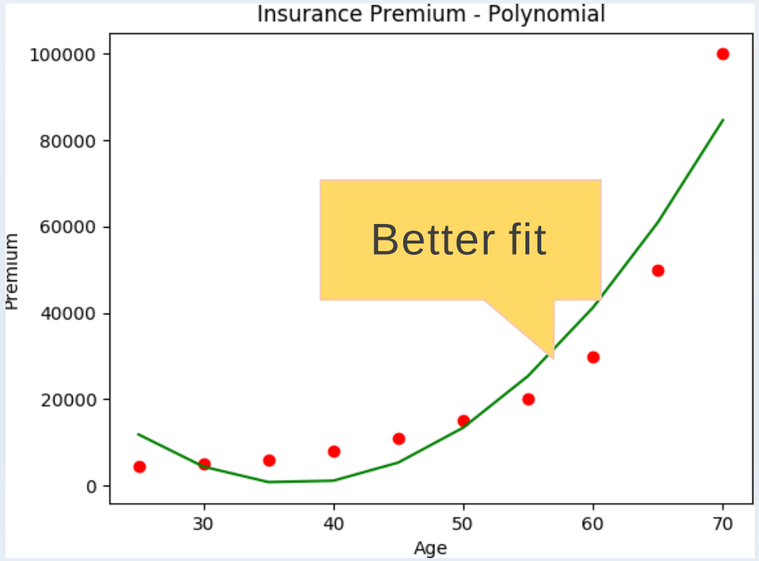
Example: Quadratic features

$$x_2' = x_2^2$$

$$y = w_1x_1 + w_2x_2^2 + 6$$

$$= w_1x_1 + w_2x_2' + 6$$



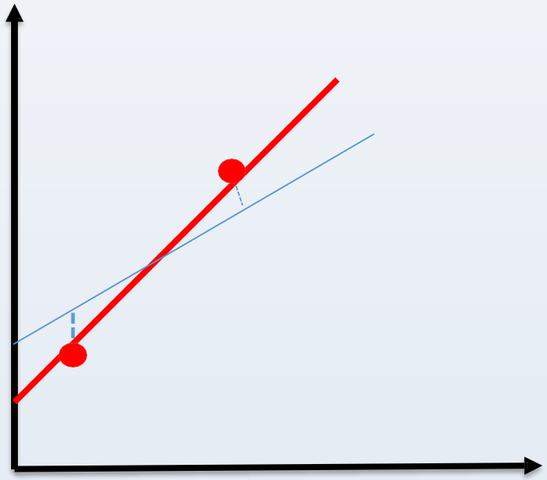


Types of Regression (Contd.)

- Linear Regression
- Multiple Linear Regression
- Polynomial Regression
- **Ridge Regression**
- Lasso Regression
- ElasticNet Regression

Ridge Regression (L2) is used when there is a problem of multicollinearity.

By adding a degree of bias to the regression estimates, ridge regression reduces the standard errors.



The main idea is to find a new line that has some bias with respect to the training data

In return for that small amount of bias, a significant drop in variance is achieved

Minimization objective = LS Obj + λ * (sum of the square of coefficients)

LS Obj refers to least squares objective

λ controls the strength of the penalty term

MACHINE LEARNING

Types of Regression (Contd.)

- Linear Regression
- Multiple Linear Regression
- Polynomial Regression
- Ridge Regression
- **Lasso Regression**
- ElasticNet Regression

Lasso Regression (L1) is similar to ridge, but it also performs feature selection.

It will set the coefficient value for features that do not help in decision making very low, potentially zero.

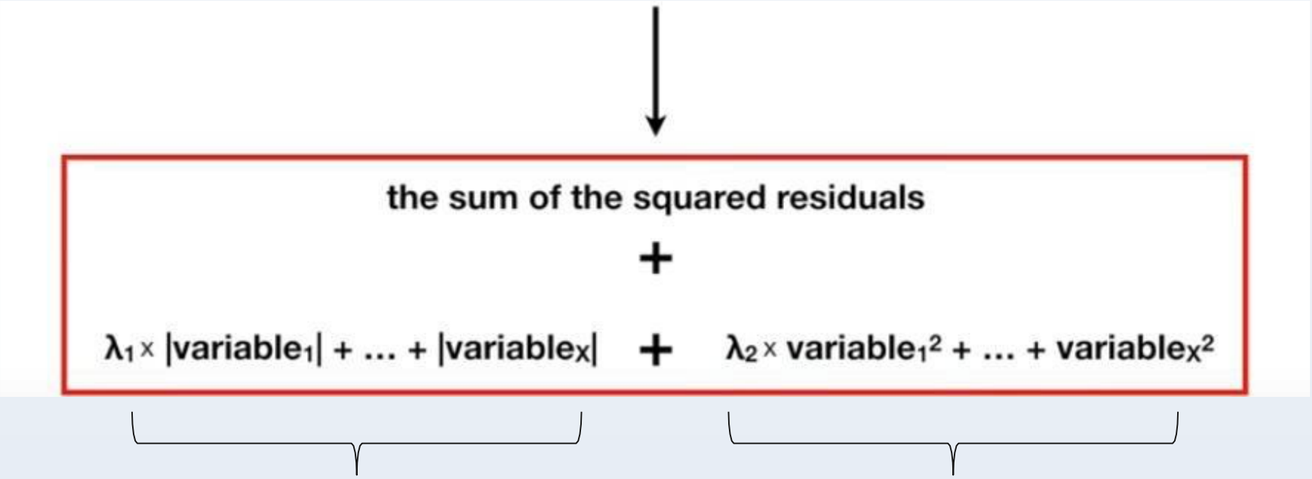
Minimization objective = LS Obj + λ * (sum of absolute coefficient values)

Lasso regression tends to exclude variables that are not required from the equation, whereas ridge tends to do better when all variables are present.

Types of Regression (Contd.)

- Linear Regression
- Multiple Linear Regression
- Polynomial Regression
- Ridge Regression
- Lasso Regression
- ElasticNet Regression

ElasticNet regression combines
the strength of **lasso and ridge regression**



If you are not sure whether to use lasso or ridge, use ElasticNet

MACHINE LEARNING

Regression Use Case

Predicting profit based on expenditures of the company

1 R&D Spend

2 Administration

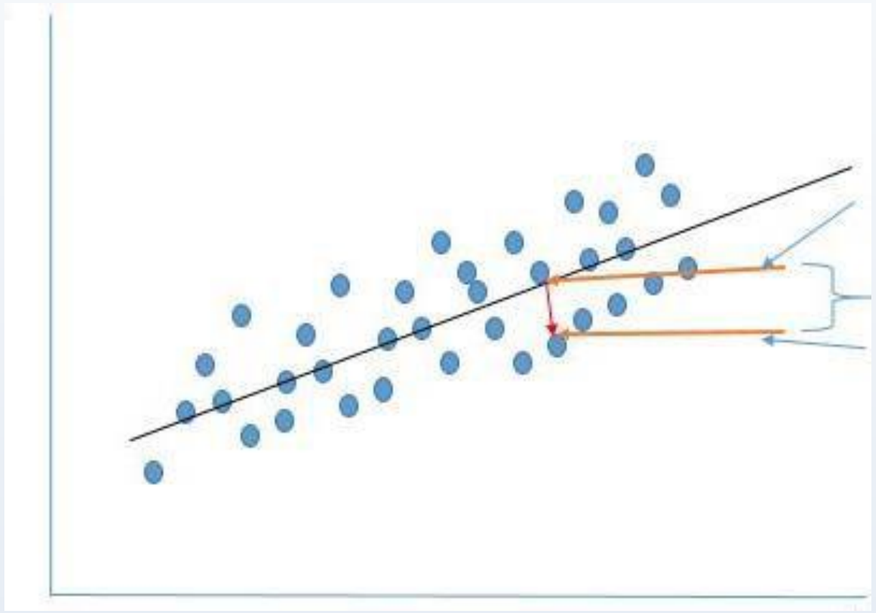
3 Marketing Spend

4 State

5 **Profit**

Profit Estimation

Accuracy Metrics



Predicted Value

Error

Actual Value

R-square = $1 - \frac{\sum(Y\_actual - Y\_predicted)^2}{\sum(Y\_actual - Y\_mean)^2}$

R-square is the most common metric to judge the performance of regression models

R^2 lies between 0 -100 %

Example: Performing linear regression on sq. Area (x) and Price (y) returns **R-square** value as 16
This means you have 16% information to make an accurate prediction about the price.

MACHINE LEARNING

Adjusted R-Squared

The disadvantage with R-squared is that it assumes every independent variable in the model explains variations in the dependent variable.

Use adjusted R-squared when working on a multiple linear regression problem.

$$Adjusted\ R^2 = 1 - \frac{(1 - R^2)(N - 1)}{N - p - 1}$$

where R^2 is R-squared value

P is number of predictor variables

N is number data points

Gradient Descent

Gradient descent is another algorithm used to reduce the loss function.

It is an optimization algorithm that tweaks it's parameters (coefficients) iteratively to minimize a given cost function to its minimum.

Model stops learning when the gradient (slope) is zero

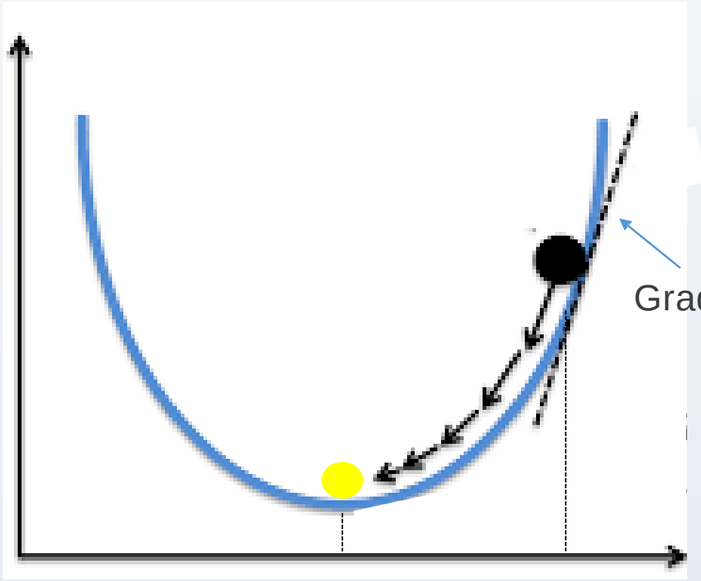
Algorithm :

1) Initialize parameter by some value

2) For each iteration calculate the derivative of the cost function and simultaneously update the parameters until a global minimum

$$\theta := \theta - \alpha \frac{\delta}{\delta\theta} J(\theta)$$

where ά is the learning rate



Evaluating Coefficients

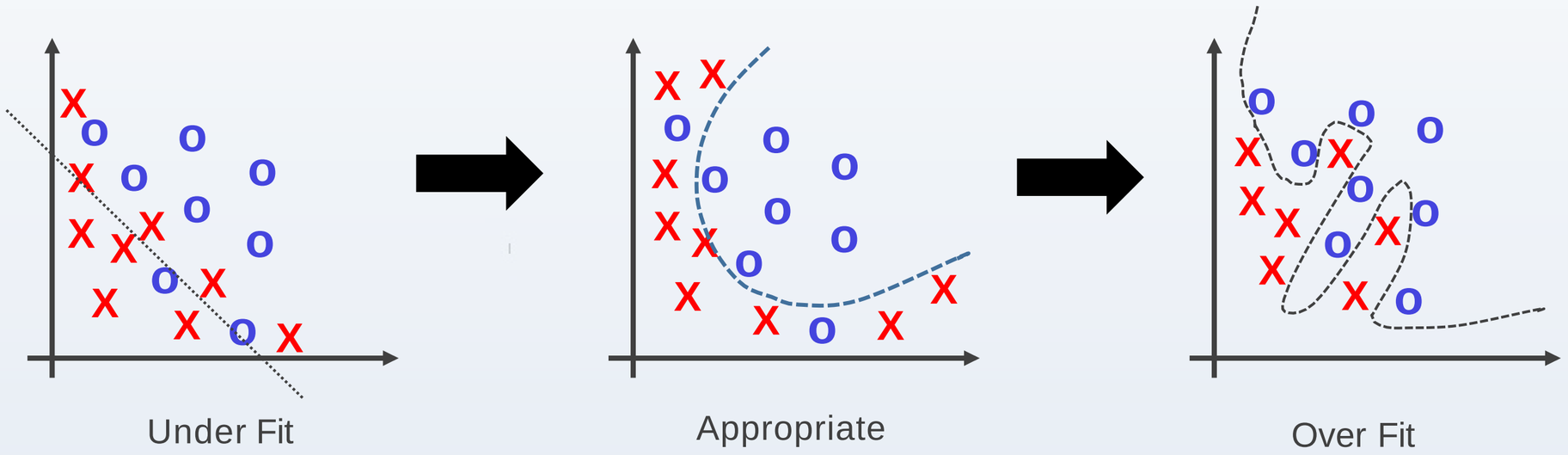
In regression analysis, p-values and coefficients together indicate which relationships in the model are statistically significant and the nature of those relationships.

Coefficients describe the mathematical relationship between each independent variable and the dependent variable.

p-values for the coefficients indicate whether these relationships are statistically significant.

$p < 0.05$	**REJECT** the Null hypothesis, meaning variables have some effect and need to be retained
$p > 0.05$	**ACCEPT** the Null hypothesis, meaning variables have no effect and can be removed

Challenges in Prediction



If the model learning is poor, you have an **underfitted** situation

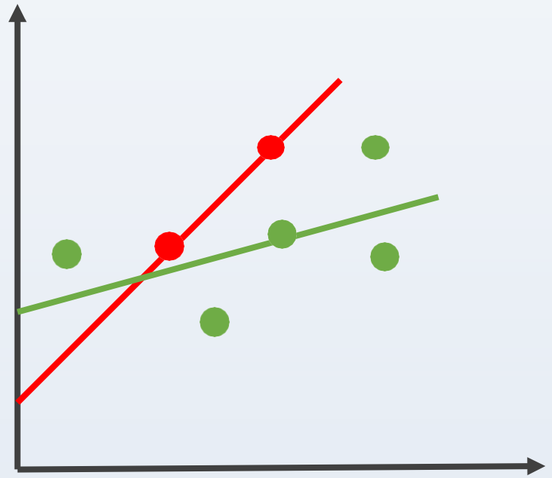
The algorithm will not work well on test data
Retraining may be needed to find a better fit

Overfitting happens when model accuracy for training data is good, but model does not generalize well to the overall population

Algorithm is not able to give good predictions for the new data

Regularization

Regularization solves overfitting to the training data.



Used to restrict the parameters values that are estimated in the model

$$L = \sum(\hat{Y}_i - Y_i)^2 + \lambda \sum \beta^2$$

This loss function includes 2 elements.

1) the sum of square distances between predicted and actual value

2) the second element is the regularization term

MACHINE LEARNING

Supervised Learning

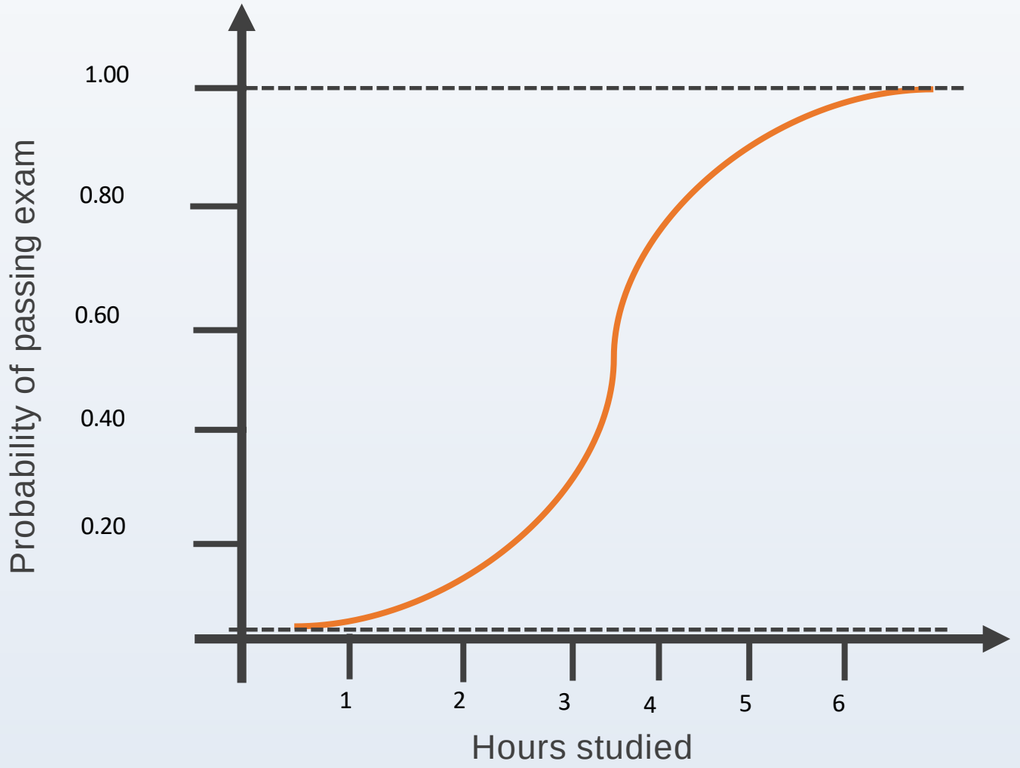
Topic 4: Logistic Regression

Logistic Regression

Logistic Regression is widely used to predict binary outcomes for a given set of independent variables.

The dependent variable's outcome is discrete such as y ϵ {0, 1}

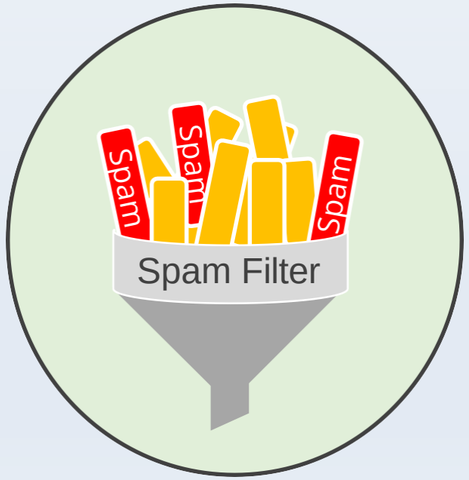
A binary dependent variable can have only two values such as 0 or 1, win or lose, pass or fail, healthy or sick.



Use Cases

Loan sanction

Customer segments



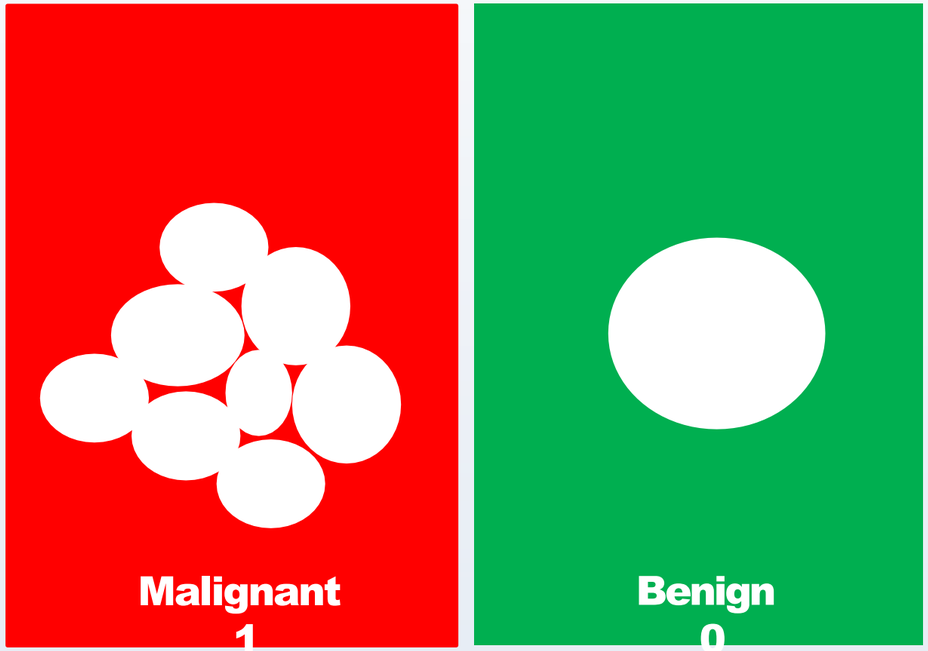
Spam filtering

Exam results – Pass/Fail

Real-Life Scenarios

Weather Forecast

sunny, stormy, cloudy, rainy



Cancer Prediction

Malignant (cancerous) and Benign (non-cancerous)

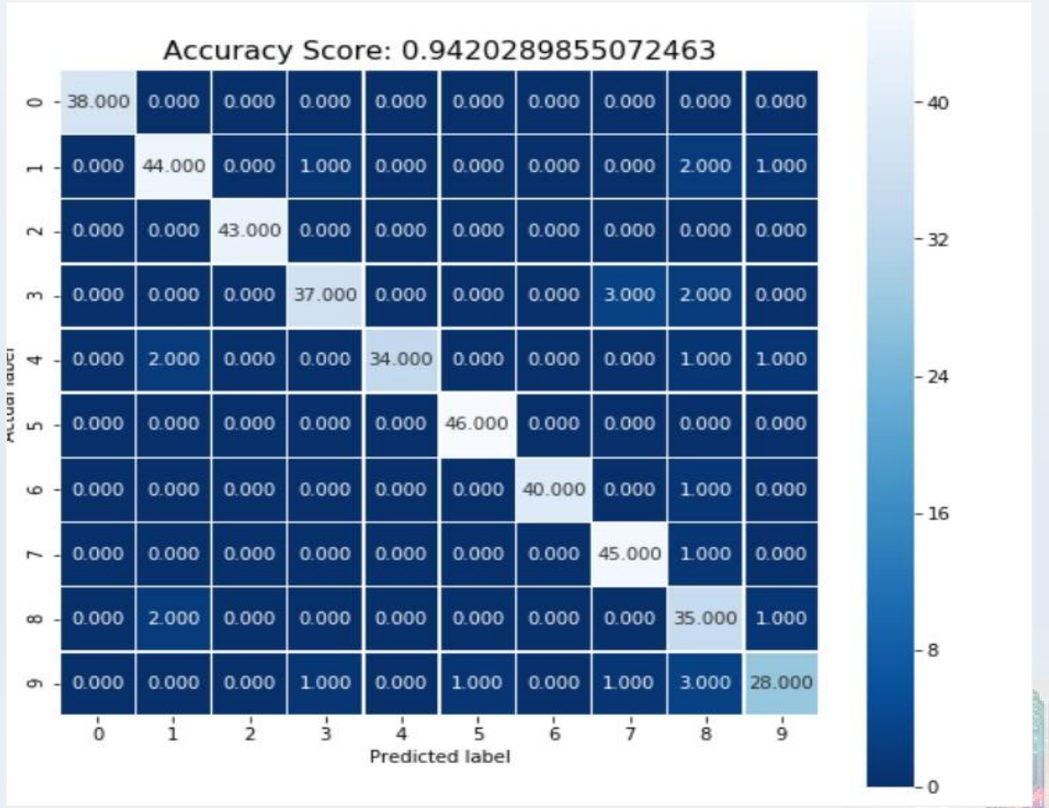
Accuracy Metrics

n=165	Predicted: NO	Predicted: YES	
Actual: NO	TN = 50	FP = 10	60
Actual: YES	FN = 5	TP = 100	105
	55	110	

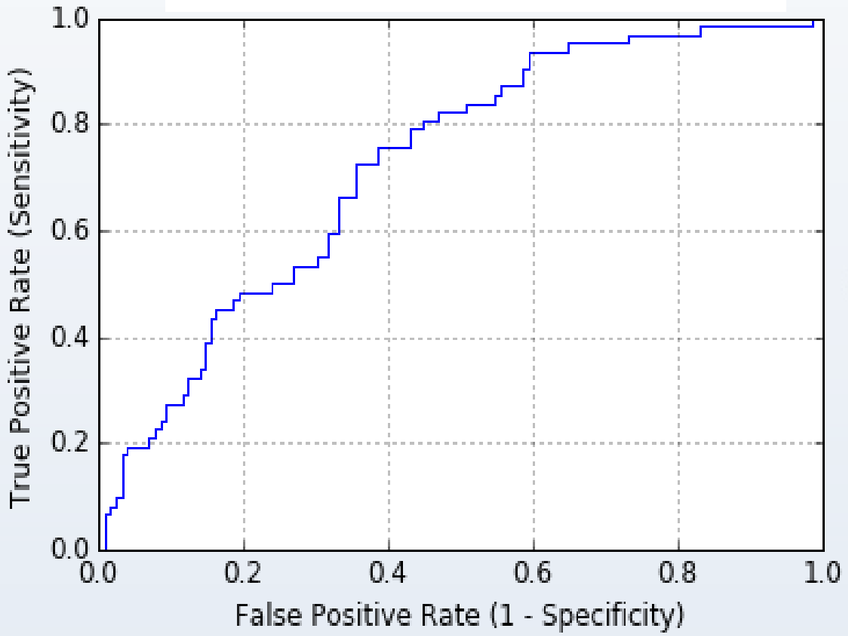
Confusion Matrix

Accuracy = $\frac{\text{True Positive (TP) + True Negative (TN)}}{\text{Total}}$

Confusion Matrix for a multi - class classification

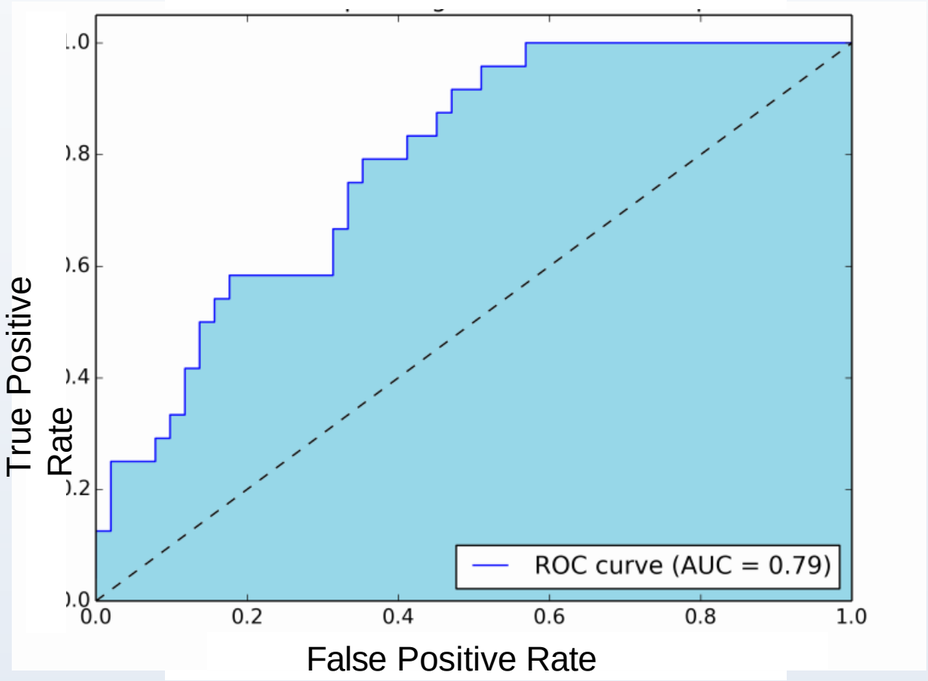


Accuracy Metrics (Contd.)



ROC curve

Compares the model true positive and false positive rates to the ones from a random assignment



AUC (Area under the ROC Curve)

Measures the entire two-dimensional area under the entire ROC curve

Machine Learning

Lesson 4: Feature Engineering

Concepts Covered

- Factor Analysis
- PCA
- LDA
- PCA, LDA in Python

Feature Engineering

Topic 1: Feature Selection

Sample Problem

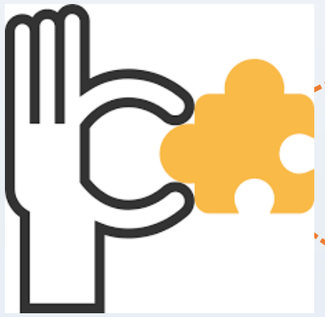
Given a multidimension dataset, extract information based on interrelations among variables.

Temperature	Time	Weight	Location	Weekday	Weekend	Sales
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	..
…	…	…	…	…	…	..
…	…	…	…	…	…	…
…	…	…	…	…	…	…

Probable Solutions

Techniques to extract feature-based information

Temperature	Time	Weight	Location	Weekday	Weekend	Sales
...	...	...	...	...	...	...
...	...	...	...	...	...	...
...	...	...	...	...	...	..
...	...	...	...	...	...	..
...	...	...	...	...	...	...
...	...	...	...	...	...	...



Regression

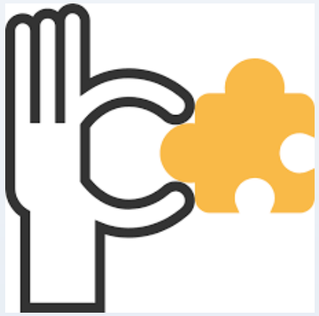
Factor Analysis

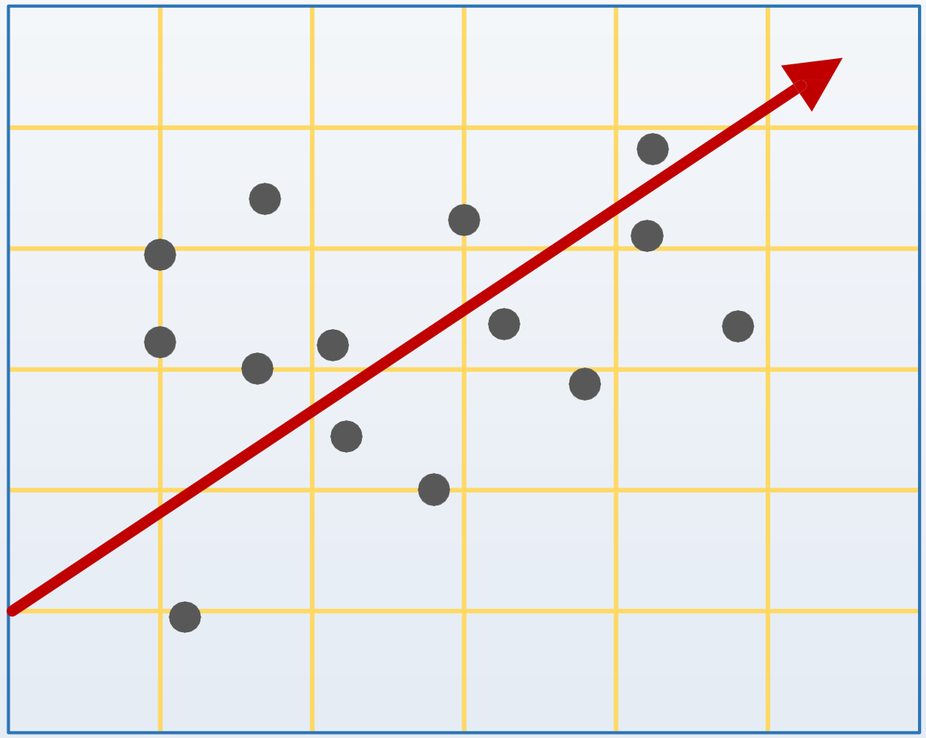
MACHINE LEARNING

Regression

Regression

Factor Analysis



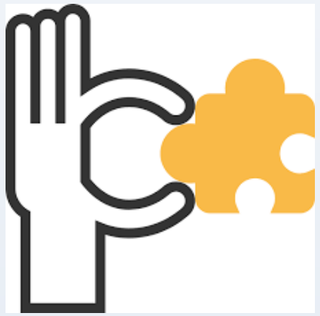


Regression tells the relationship among variables and quantifies the relationship using set of equations

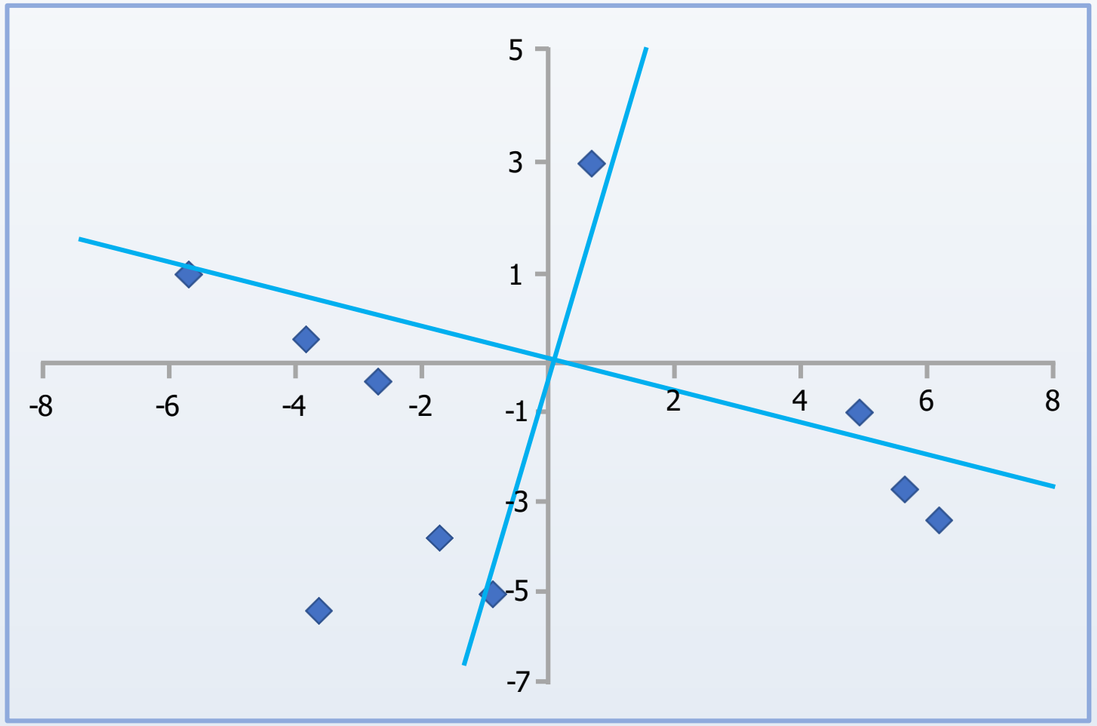
MACHINE LEARNING

Factor Analysis

Regression



Factor Analysis

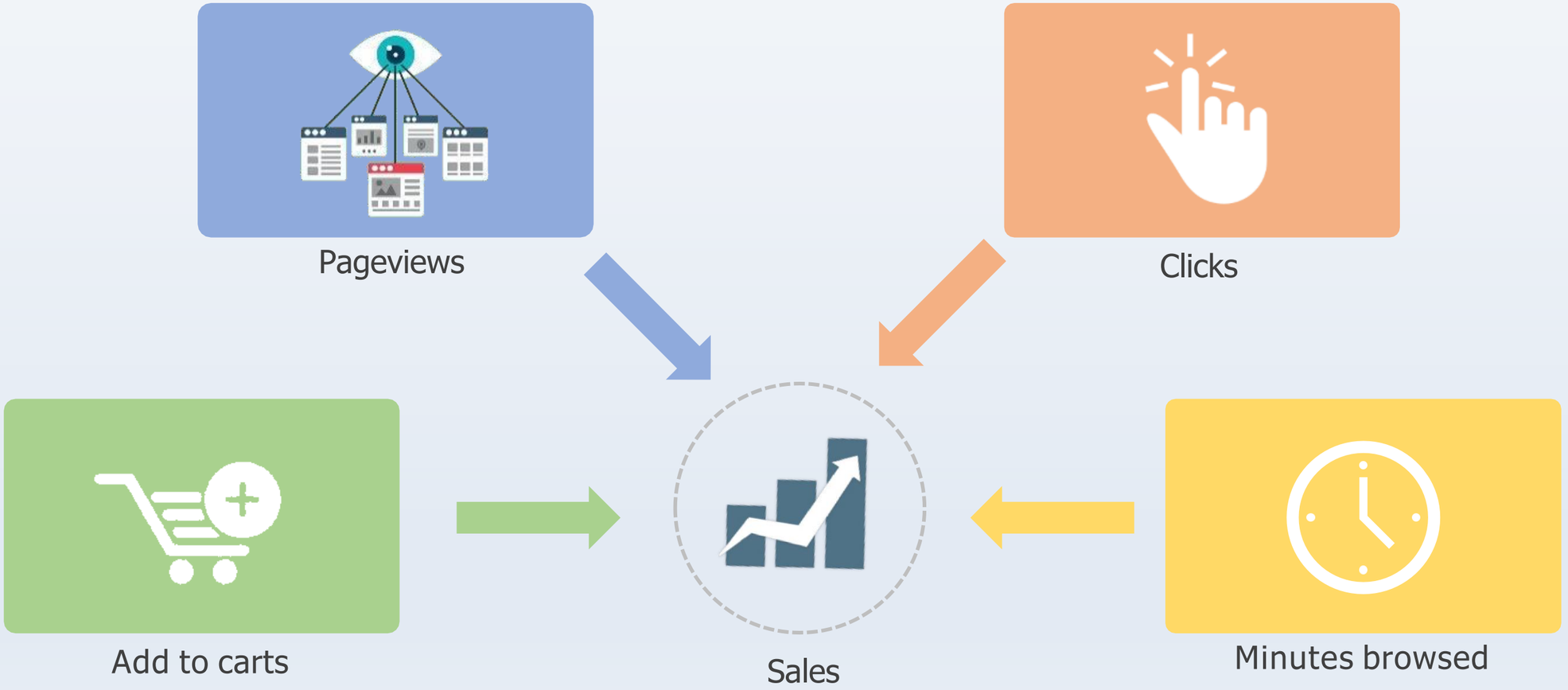


Common factors of the observations explain the variable interdependence

MACHINE LEARNING

Example

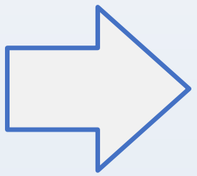
The relationship between observable variables and observable outcome: sales



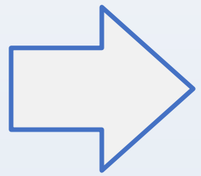
Example (Contd.)

There may be few underlying causes which can affect the output apart from the observed ones:

- Pageviews
- Clicks
- Add to carts
- Minutes Browsed
- Sessions



- Selection
- Marketing spend
- Pricing



- Sales

Observed causes

Underlying causes

Effect

Note: Identifying underlying causes helps in accurate sales prediction

Feature Engineering

Topic 2: Factor Analysis

Factor Analysis Process

There may be few underlying causes which can affect the output, apart from the observed ones:

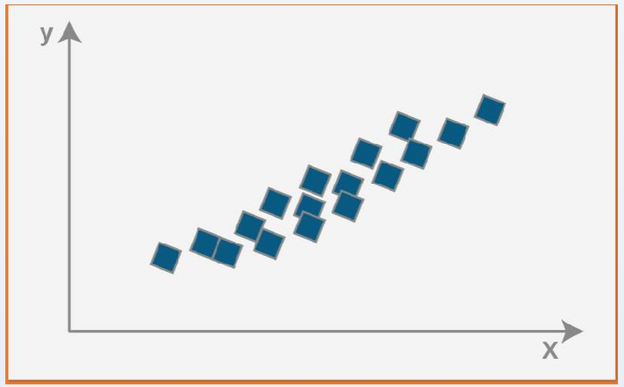
Principal Component Analysis (PCA)

- Extracts hidden factors from the dataset
- Defines your data using less number of components, explaining the variance in your data
- Reduces the computational complexity
- Determines whether a new data point is part of the group of data points from your training set

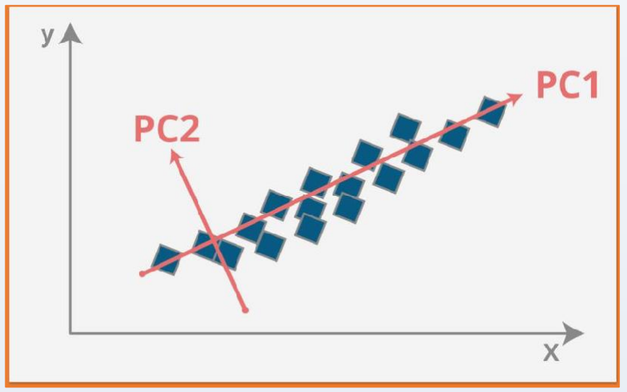
Linear Discriminant Analysis (LDA)

- Reduces Dimensions
- Searches for a linear combination of variables that best separates 2 classes
- Reduces the degree of overfitting
- Determines how to classify a new observation out of a group of classes

Principal Component Analysis



You have data on the x and y axis



Applying PCA: New set of axes are achieved, denoted as PC1 and PC2

Data around PC2 projected along PC1 to ensure no variance is lost

MACHINE LEARNING

Linear Discriminant Analysis (LDA)

Assume a set of D - dimensional samples {X(1, X(2, …, X(N}, N1 of which belong to class ω1 and N2 to class ω2

Obtain a scalar y by projecting the samples x onto a line: $Y = W^T X$

Of all the possible lines, select the one that maximizes the separability of the scalars

Machine Learning

Lesson 5: Supervised Learning–Classification

Dr.Prasanalakshmi

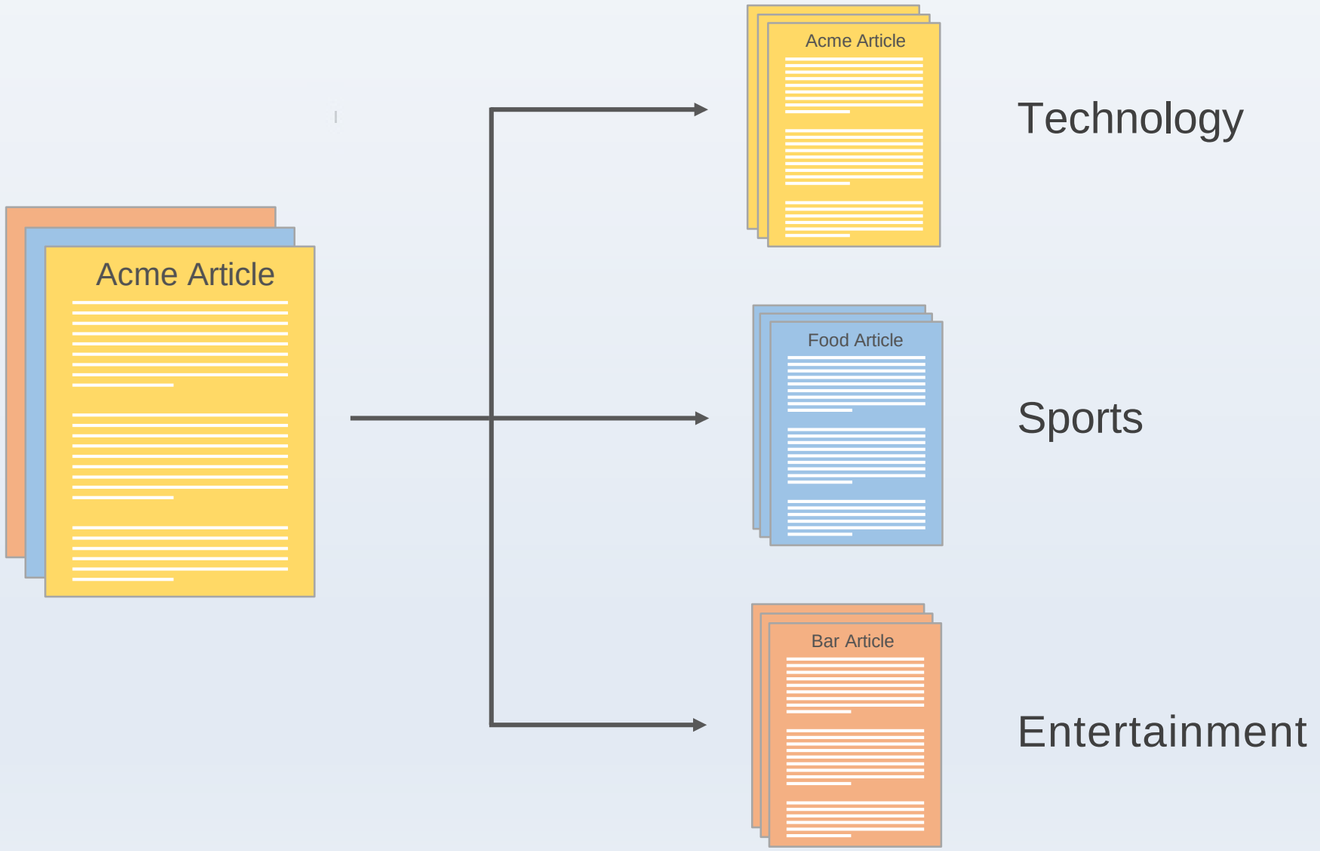
© Simplilearn. All rights reserved.

Classification

Topic 1: Definition of Classification

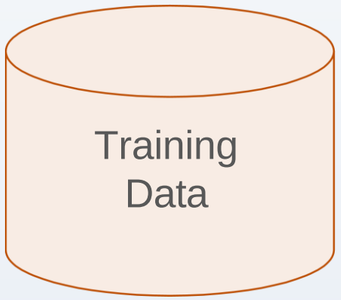
What Is Classification?

A machine learning task that identifies the class to which an instance belongs



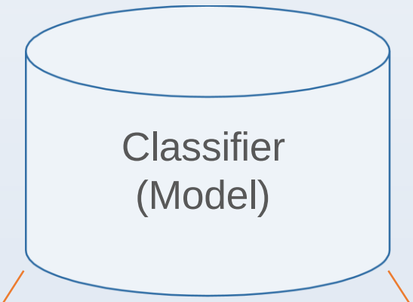
Classification: Example

Training a classifier model with respect to the available data



Classification
Algorithms

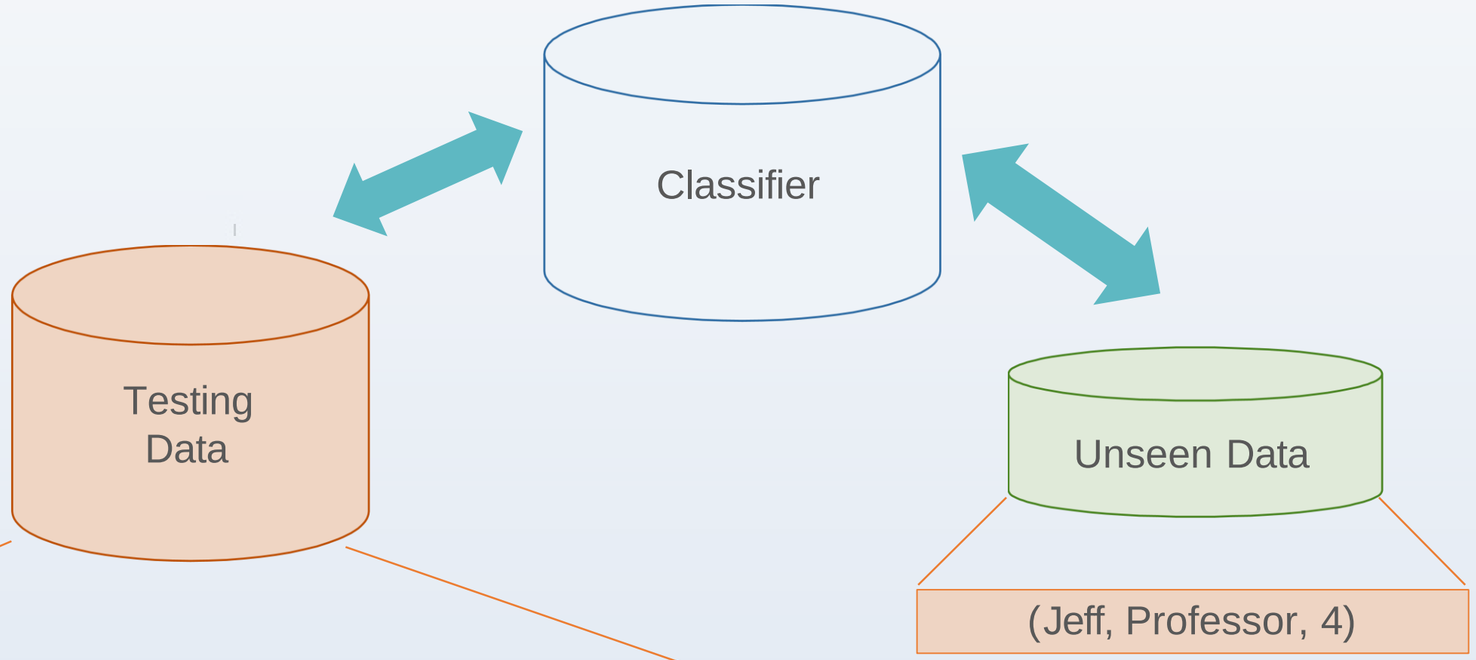
Name	Rank	Years	Tenured
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Associate Prof	6	no
Anne	Associate Prof	3	no



IF rank = “professor”
OR years > 6
THEN tenured = “yes”

Classification: Example

The model will classify if the professors are tenured or not



Name	Rank	Years	Tenured
Tom	Assistant Prof	2	no
Merlisa	Assistant Prof	7	no
George	Professor	5	yes
Joseph	Assistant Prof	7	yes

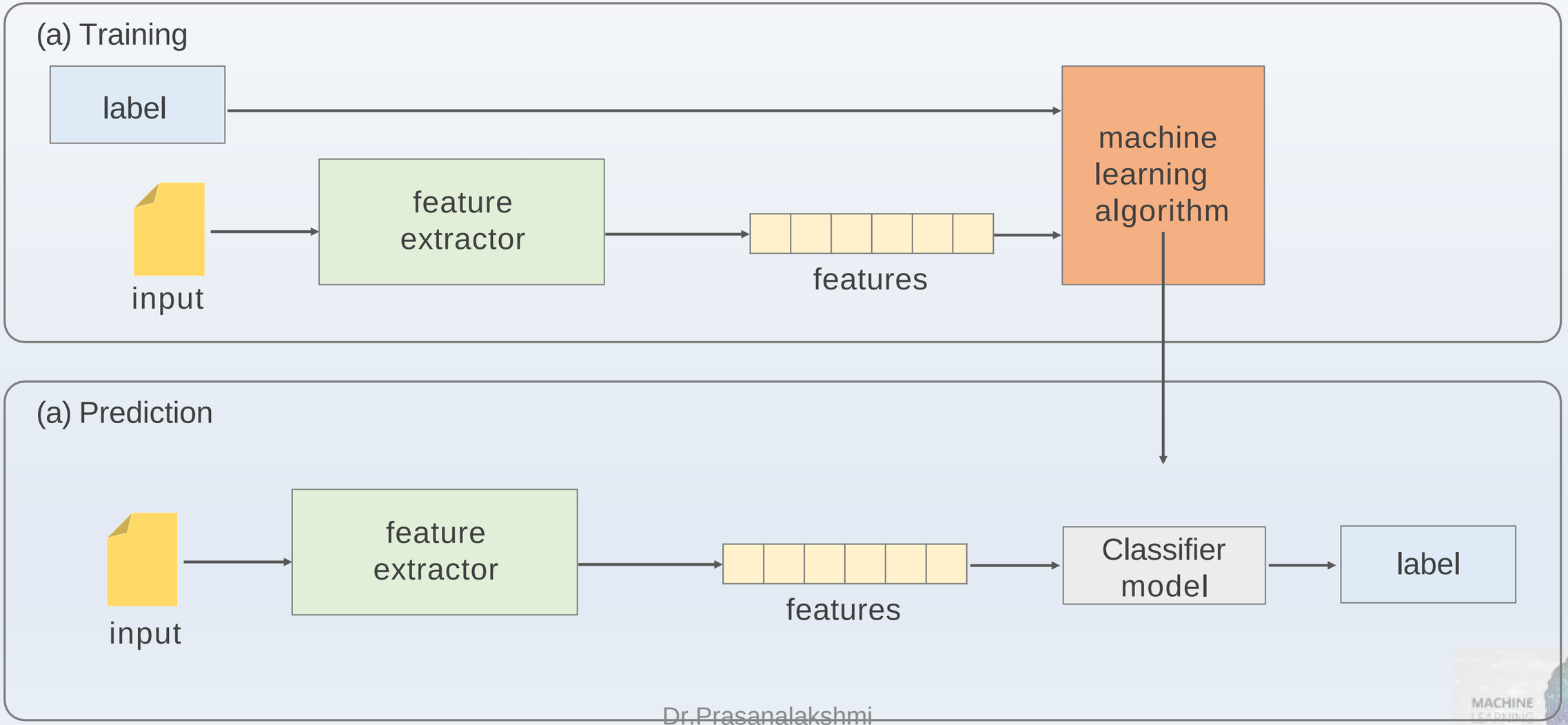
Tenured?

Yes

Dr.Prasanalakshmi

Classification: Work Flow

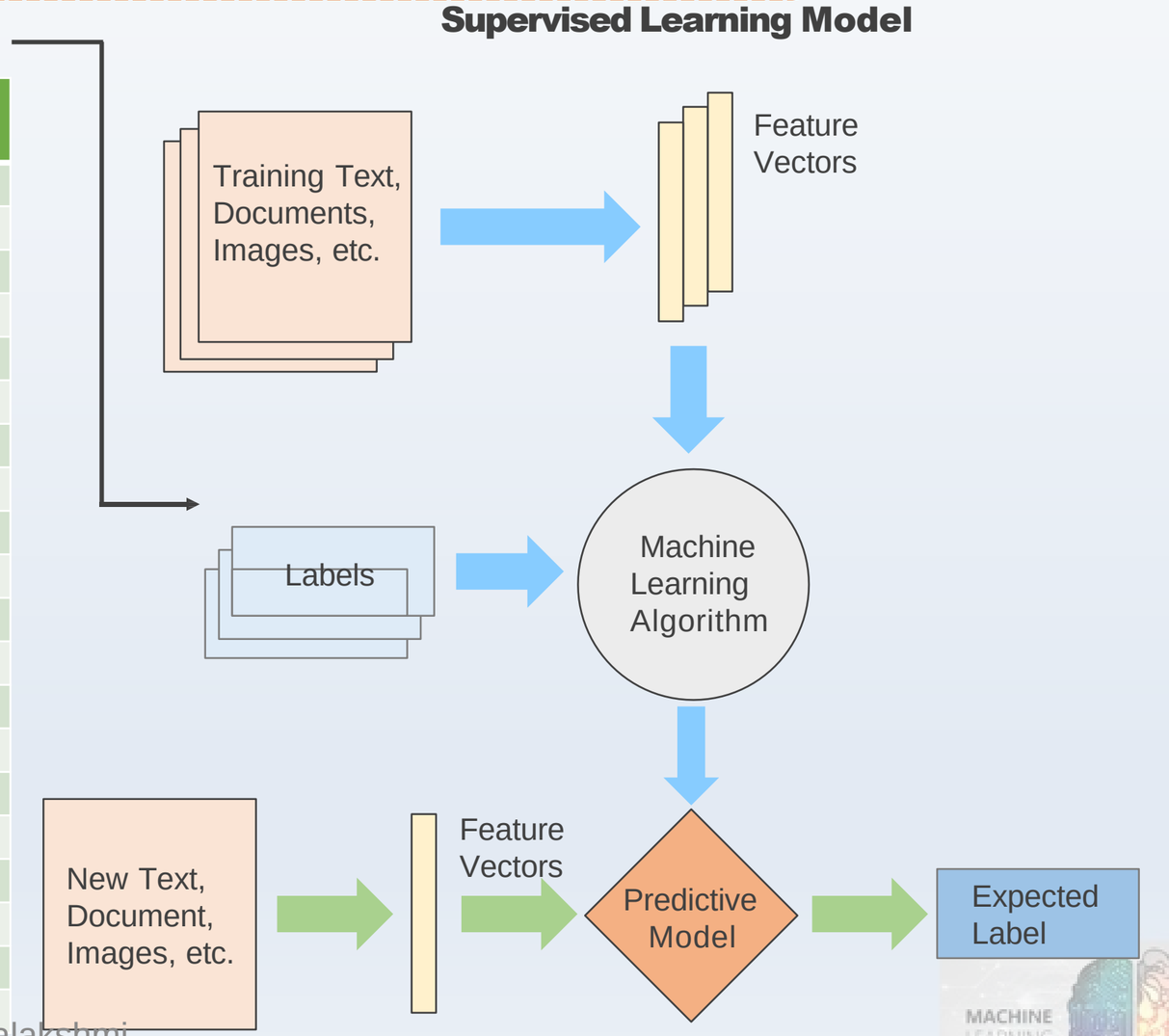
A typical classifier model workflow with input training data and output labels



Classification: A Supervised Learning Algorithm

Classification is a supervised learning algorithm as the training data contains labels

Record ID	Age	Spectacle Prescription	Astigmatic	Tear production Rate	Class Label Lenses
1	Young	Myope	No	Reduced	Noncontact
2	Young	Myope	No	Normal	Soft contact
3	Young	Myope	Yes	Reduced	Noncontact
4	Young	Myope	Yes	Normal	Hard contact
5	Young	Hypermetrope	No	Reduced	Noncontact
6	Young	Hypermetrope	No	Normal	Soft contact
7	Young	Hypermetrope	Yes	Reduced	Noncontact
8	Young	Hypermetrope	Yes	Normal	Hard contact
9	Pre-presbyopic	Myope	No	Reduced	Noncontact
10	Pre-presbyopic	Myope	No	Normal	Soft contact
11	Pre-presbyopic	Myope	Yes	Reduced	Noncontact
12	Pre-presbyopic	Myope	Yes	Normal	Hard contact
13	Pre-presbyopic	Hypermetrope	No	Reduced	Noncontact
14	Pre-presbyopic	Hypermetrope	No	Normal	Soft contact
15	Pre-presbyopic	Hypermetrope	Yes	Reduced	Noncontact
16	Pre-presbyopic	Hypermetrope	Yes	Normal	Noncontact
17	Presbyopic	Myope	No	Reduced	Noncontact
18	Presbyopic	Myope	No	Normal	Noncontact
19	Presbyopic	Myope	Yes	Reduced	Noncontact
20	Presbyopic	Myope	Yes	Normal	Hard contact

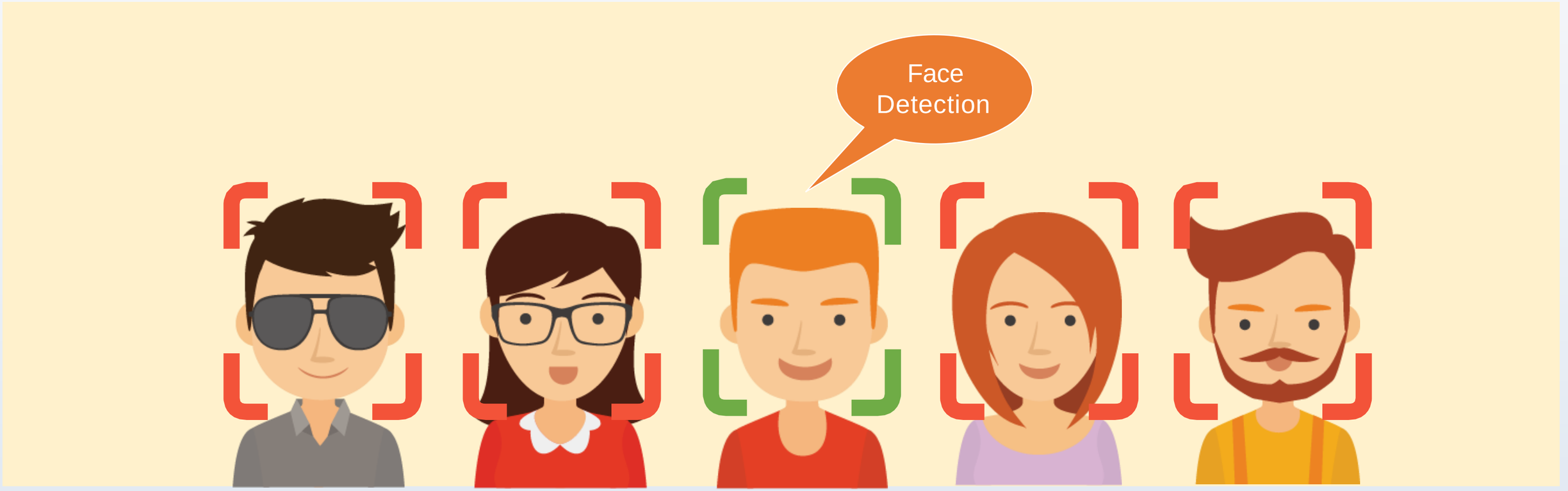


Classification

Topic 2: Use Cases and Algorithms


Sentiment
Analysis


Fraud
Detection

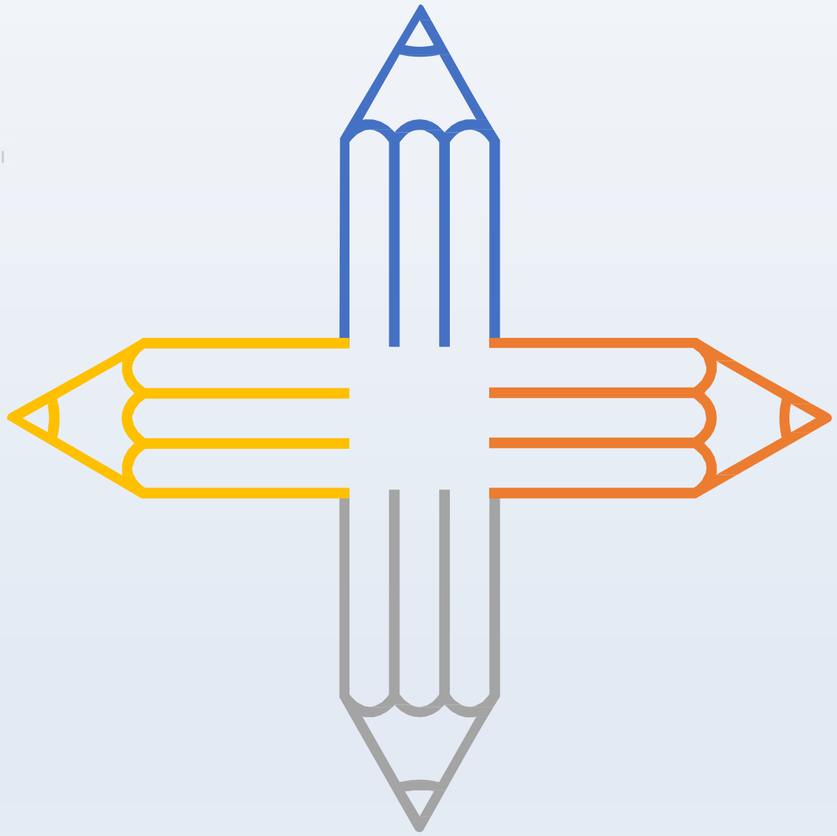

Face
Detection

Classification Algorithms

Few of the most commonly used classification algorithms:

Decision Tree

Support Vector Machines (SVM)



Random Forest

Naîve Bayes Classifier

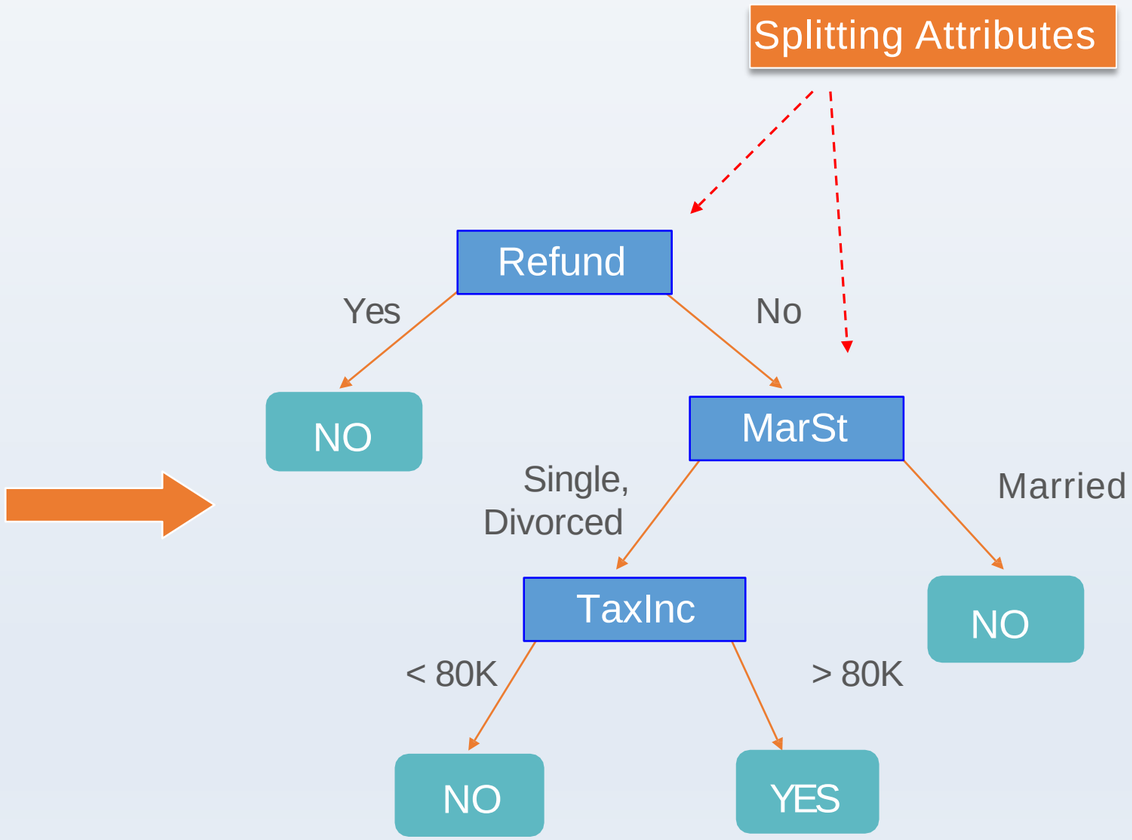
Decision Tree: Example 1

Below example illustrates the splitting attributes with respect to the adjacent training data

categorical categorical continuous class

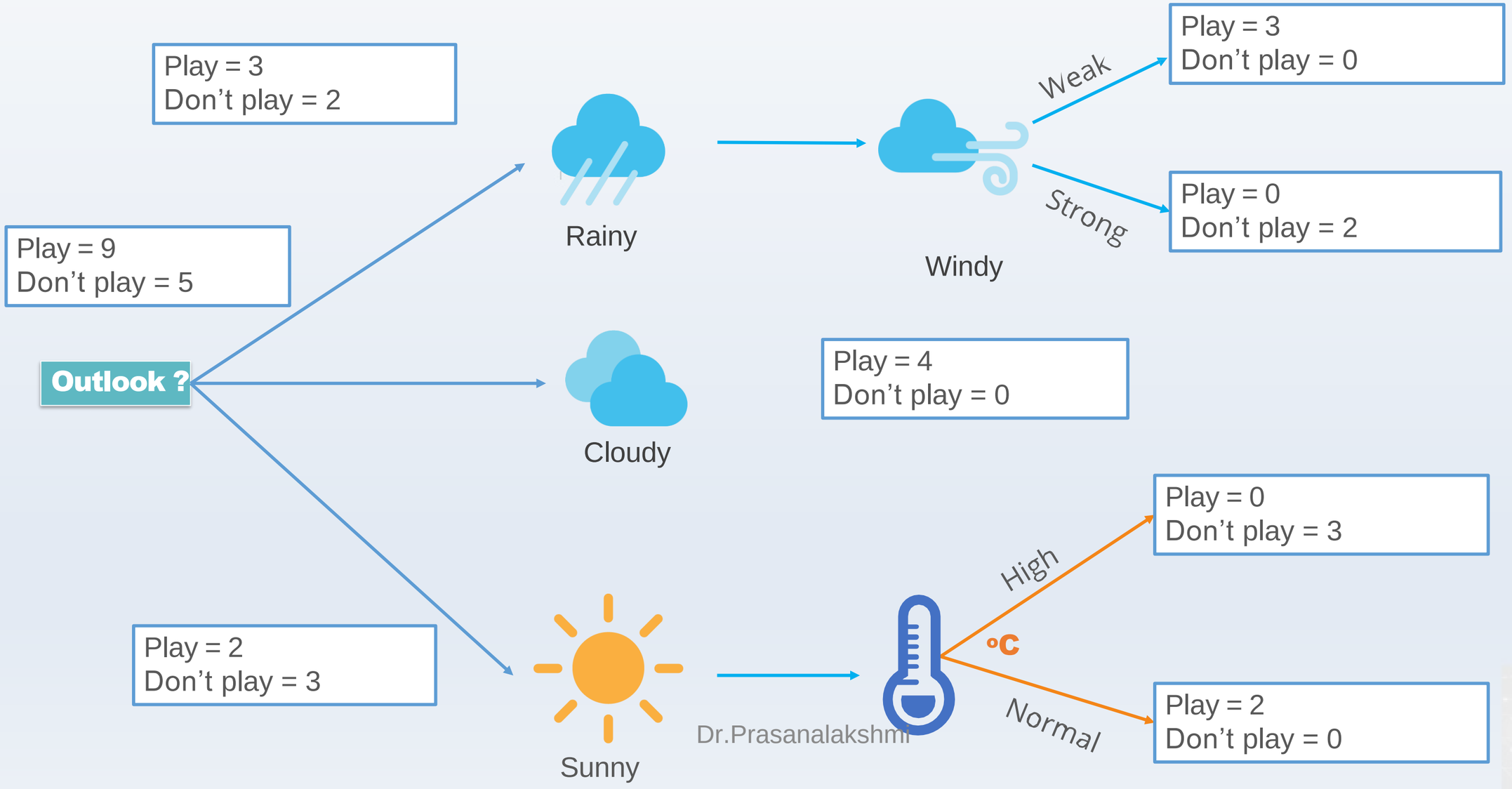
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	**No**
2	No	Married	100K	**No**
3	No	Single	70K	**No**
4	Yes	Married	120K	**No**
5	No	Divorced	95K	**Yes**
6	No	Married	60K	**No**
7	Yes	Divorced	220K	**No**
8	No	Single	85K	**Yes**
9	No	Married	75K	**No**
10	No	Single	90K	**Yes**

Training Data



Decision Tree: Example 2

Forming a decision tree to check if the match will be played or not based on climatic conditions



Dr.Prasanalakshmi

Classification

Topic 4: Random Forest Classifier

Bagging and Bootstrapping

Bagging

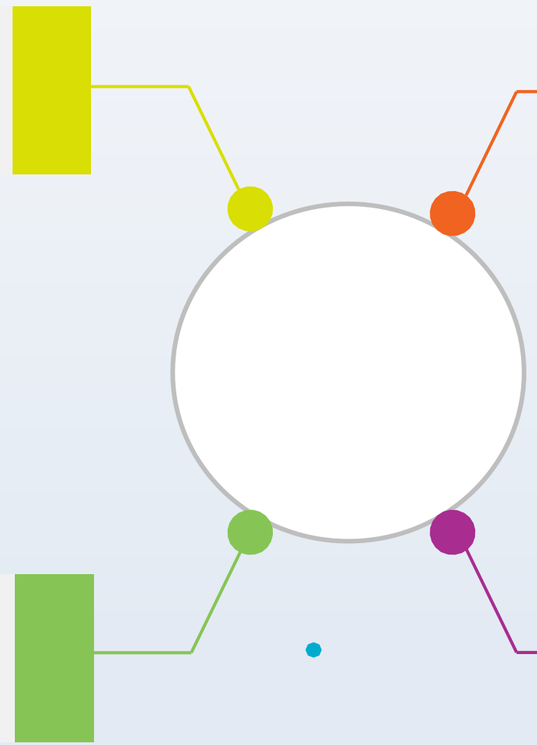
A technique for reducing the variance of an estimated prediction function

For classification, a committee of trees each cast a vote for the predicted class

Bootstrapping

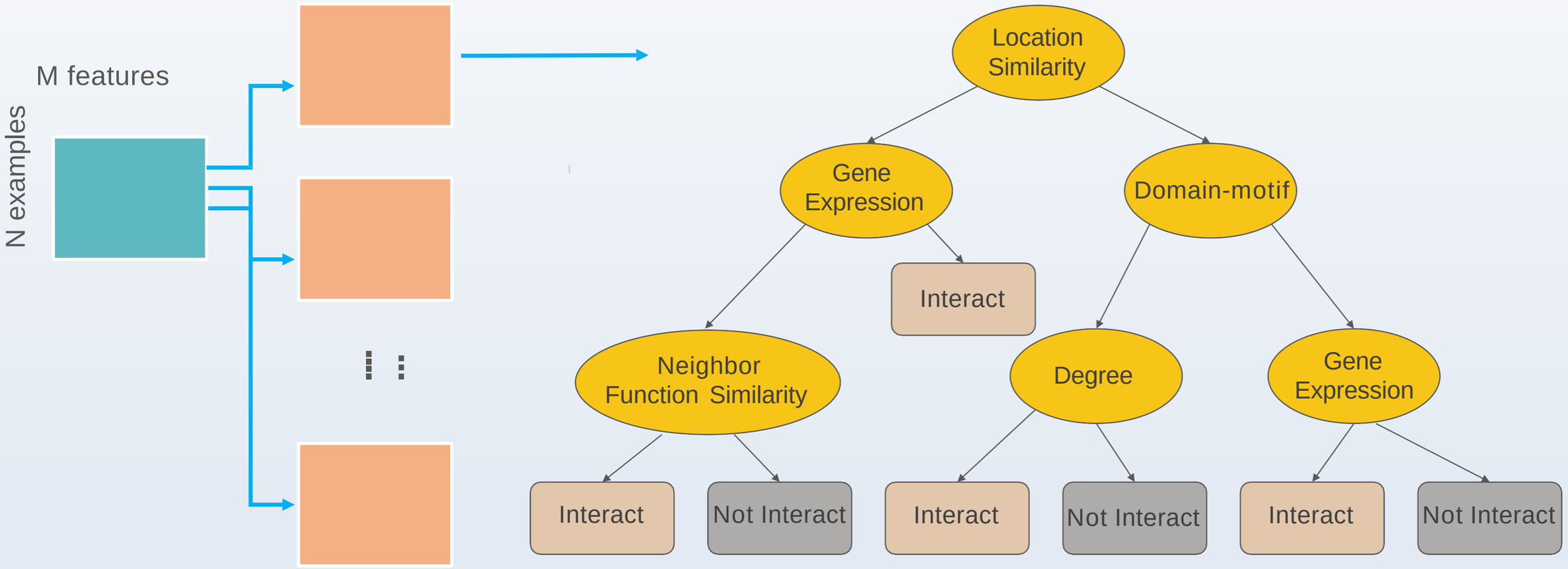
Randomly draws datasets with replacement from the training data

Each sample is of the same size as the original training set

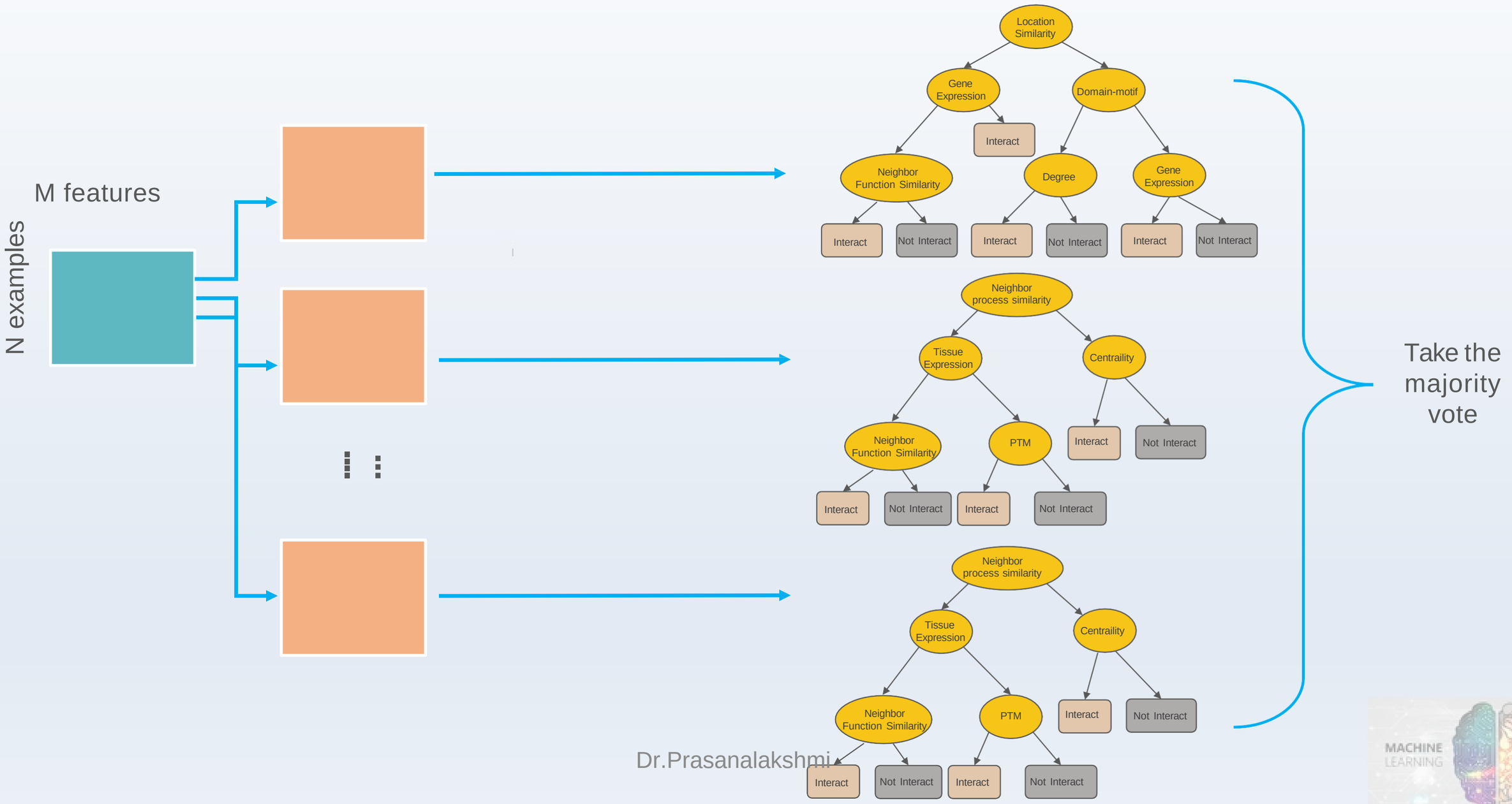


Decision Tree Classifier

Each sample contributes to a decision tree classifier



Random Forest Classifier



Dr.Prasanalakshmi

Classification

Topic 5: Performance Measures

Confusion Matrix

Focus on the predictive capability of a model

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a	b
	Class=No	c	d

a: TP (true positive)

b: FN (false negative)

c: FP (false positive)

d: TN (true negative)

Accuracy Metric

Ratio of true positives and true negatives to the sum of true positives, true negatives, false negatives, and false positives

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

Limitation of Accuracy

Consider a 2-class problem

Number of Class 0 examples = 9990

Number of Class 1 examples = 10

If the model predicts every example to be class 0, accuracy is 9990/10000 = 99.9 %

Hence, accuracy is misleading because the model does not detect any class 1 example

Get the Data

Code

```
loans = pd.read_csv('loan_borowwer_data.csv’)
loans.describe()
```

```
loans.describe()
```

	credit.policy	int.rate	installment	log.annual.inc	dti	fico	days.with.cr.line	revol.bal	revol.util	inq.last.6mths	delinq.2yrs
count	9578.000000	9578.000000	9578.000000	9578.000000	9578.000000	9578.000000	9578.000000	9.578000e+03	9578.000000	9578.000000	9578.00000(
mean	0.804970	0.122640	319.089413	10.932117	12.606679	710.846314	4560.767197	1.691396e+04	46.799236	1.577469	0.16370{
std	0.396245	0.026847	207.071301	0.614813	6.883970	37.970537	2496.930377	3.375619e+04	29.014417	2.200245	0.54621{
min	0.000000	0.060000	15.670000	7.547502	0.000000	612.000000	178.958333	0.000000e+00	0.000000	0.000000	0.00000(
25%	1.000000	0.103900	163.770000	10.558414	7.212500	682.000000	2820.000000	3.187000e+03	22.600000	0.000000	0.00000(
50%	1.000000	0.122100	268.950000	10.928884	12.665000	707.000000	4139.958333	8.596000e+03	46.300000	1.000000	0.00000(
75%	1.000000	0.140700	432.762500	11.291293	17.950000	737.000000	5730.000000	1.824950e+04	70.900000	2.000000	0.00000(
max	1.000000	0.216400	940.140000	14.528354	29.960000	827.000000	17639.958330	1.207359e+06	119.000000	33.000000	13.00000(

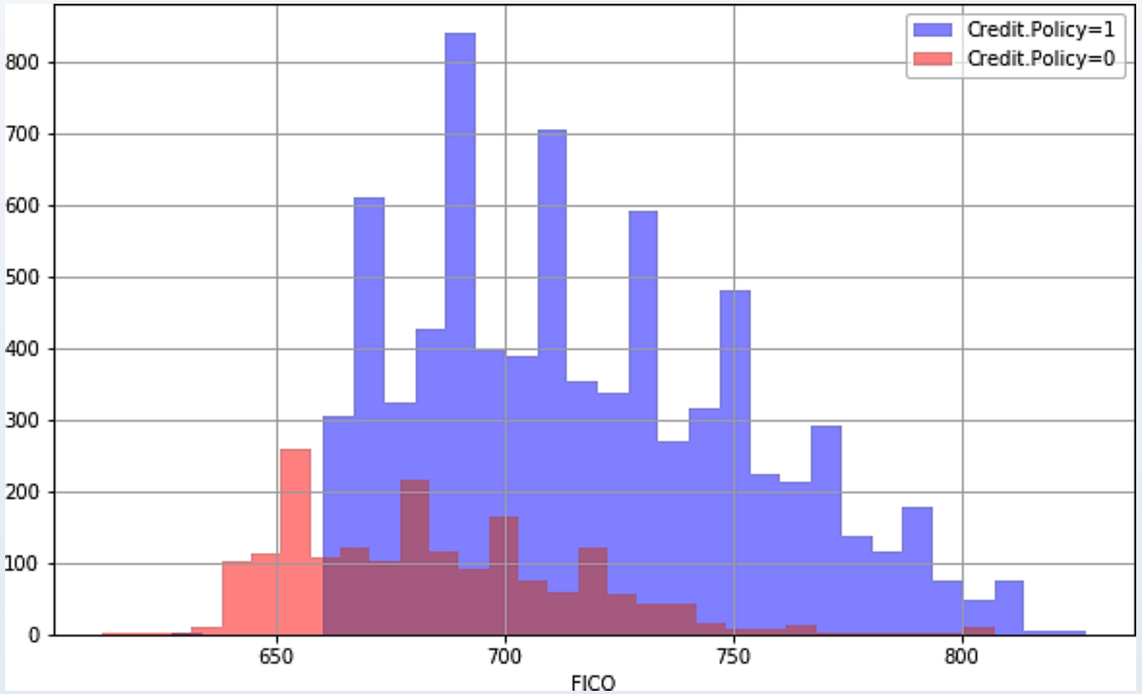
Exploratory Data Analysis

Create a histogram of two FICO distributions on top of each other, one for each credit.policy outcome.

Code

```
plt.figure(figsize=(10,6))
loans[loans['credit.policy']==1]['fico'].hist(alpha=0.5,color='blue',
bins=30,label='Credit.Policy=1')
loans[loans['credit.policy']==0]['fico'].hist(alpha=0.5,color='red',
bins=30,label='Credit.Policy=0')
plt.legend()
plt.xlabel('FICO')
```


Exploratory Data Analysis



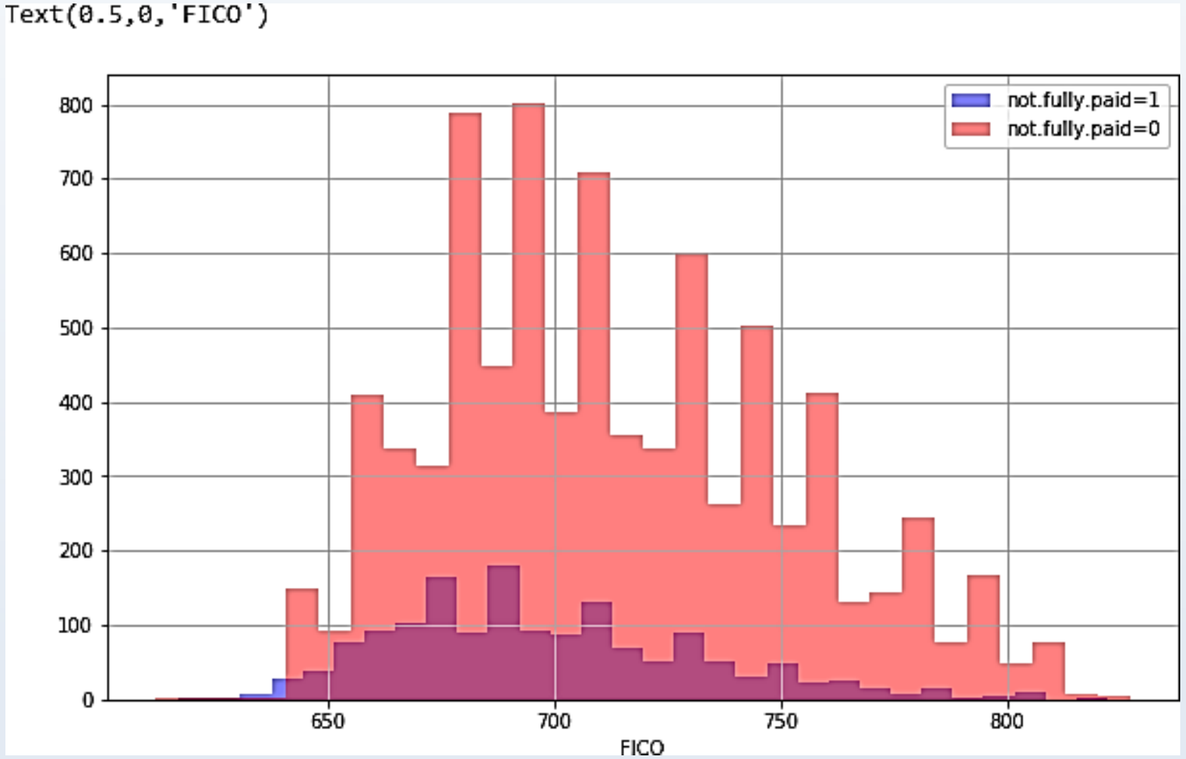
Exploratory Data Analysis

Create a similar figure; select the not.fully.paid column

Code

```
plt.figure(figsize=(10,6))
loans[loans['not.fully.paid']==1]['fico'].hist(alpha=0.5,color='blue',
bins=30,label='not.fully.paid=1')
loans[loans['not.fully.paid']==0]['fico'].hist(alpha=0.5,color='red',
bins=30,label='not.fully.paid=0')
plt.legend()
plt.xlabel('FICO')
```


Exploratory Data Analysis



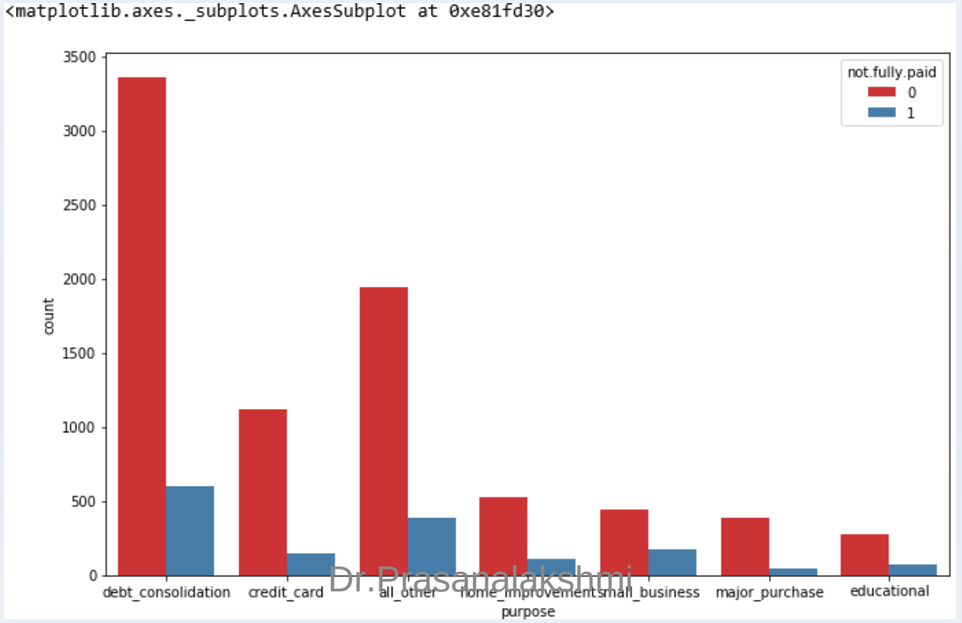
Exploratory Data Analysis

Create a countplot using seaborn showing the counts of loans by purpose, with the hue defined by not.fully.paid.

Code

```
plt.figure(figsize=(11,7))
sns.countplot(x='purpose',hue='not.fully.paid',data=loans,palette='Set1')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0xe81fd30>
```



Dr.Prasanalakshmi

MACHINE LEARNING

Setting Up the Data

Create a list of elements, containing the string “purpose.” Call this list cat_feats.

Code

```
cat_feats = ['purpose’]
```


Setting Up the Data

Now use pd.get_dummies (loans,columns=cat_feats,drop_first=True) to create a fixed larger data frame that has new feature columns with dummy variables. Set this data frame as final_data.

Code

```
final_data = pd.get_dummies(loans,columns=cat_feats,drop_first=True)
final_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9578 entries, 0 to 9577
Data columns (total 19 columns):
credit.policy                 9578 non-null int64
int.rate                      9578 non-null float64
installment                   9578 non-null float64
log.annual.inc                9578 non-null float64
dti                           9578 non-null float64
fico                          9578 non-null int64
days.with.cr.line             9578 non-null float64
revol.bal                     9578 non-null int64
revol.util                    9578 non-null float64
inq.last.6mths                9578 non-null int64
delinq.2yrs                   9578 non-null int64
pub.rec                       9578 non-null int64
not.fully.paid                9578 non-null int64
purpose_credit_card           9578 non-null uint8
purpose_debt_consolidation    9578 non-null uint8
purpose_educational           9578 non-null uint8
purpose_home_improvement      9578 non-null uint8
purpose_major_purchase        9578 non-null uint8
purpose_small_business        9578 non-null uint8
dtypes: float64(6), int64(7), uint8(6)
memory usage: 1.0 MB
```

Dr.Prasanalakshmi

Train-Test Split

Code

```
X = final_data.drop('not.fully.paid',axis=1)
y = final_data['not.fully.paid']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30,
random_state=101)
```


Training Decision Tree Model

Code

```
from sklearn.tree import DecisionTreeClassifier
dtree = DecisionTreeClassifier()
dtree.fit(X_train,y_train)
```

```
Out[24]: DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
                                max_features=None, max_leaf_nodes=None,
                                min_impurity_decrease=0.0, min_impurity_split=None,
                                min_samples_leaf=1, min_samples_split=2,
                                min_weight_fraction_leaf=0.0, presort=False, random_state=None,
                                splitter='best')
```


Evaluating Decision Tree

Create predictions from the test set, and create a classification report and a confusion matrix.

Code

```
predictions = dtree.predict(X_test)
from sklearn.metrics import classification_report,confusion_matrix
print(classification_report(y_test,predictions))
```

	precision	recall	f1-score	support
0	0.85	0.82	0.84	2431
1	0.19	0.23	0.21	443
avg / total	0.75	0.73	0.74	2874

Confusion Matrix

Code

```
print(confusion_matrix(y_test,predictions))
```

```
[[1993  438]
 [ 340  103]]
```


Training Random Forest Model

Code

```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(n_estimators=600)
rfc.fit(X_train,y_train)
```

```
Out[37]: RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                                max_depth=None, max_features='auto', max_leaf_nodes=None,
                                min_impurity_decrease=0.0, min_impurity_split=None,
                                min_samples_leaf=1, min_samples_split=2,
                                min_weight_fraction_leaf=0.0, n_estimators=600, n_jobs=1,
                                oob_score=False, random_state=None, verbose=0,
                                warm_start=False)
```


Evaluating Random Forest Model

Code

```
predictions = rfc.predict(X_test)
from sklearn.metrics import classification_report,confusion_matrix
print(classification_report(y_test,predictions))
```

	precision	recall	f1-score	support
0	0.85	1.00	0.92	2431
1	0.62	0.02	0.04	443
avg / total	0.81	0.85	0.78	2874

Printing the Confusion Matrix

Code

```
print(confusion_matrix(y_test,predictions))
```

```
[[2425    6]
 [ 433   10]]
```


Machine Learning

Lesson 6: Unsupervised Learning

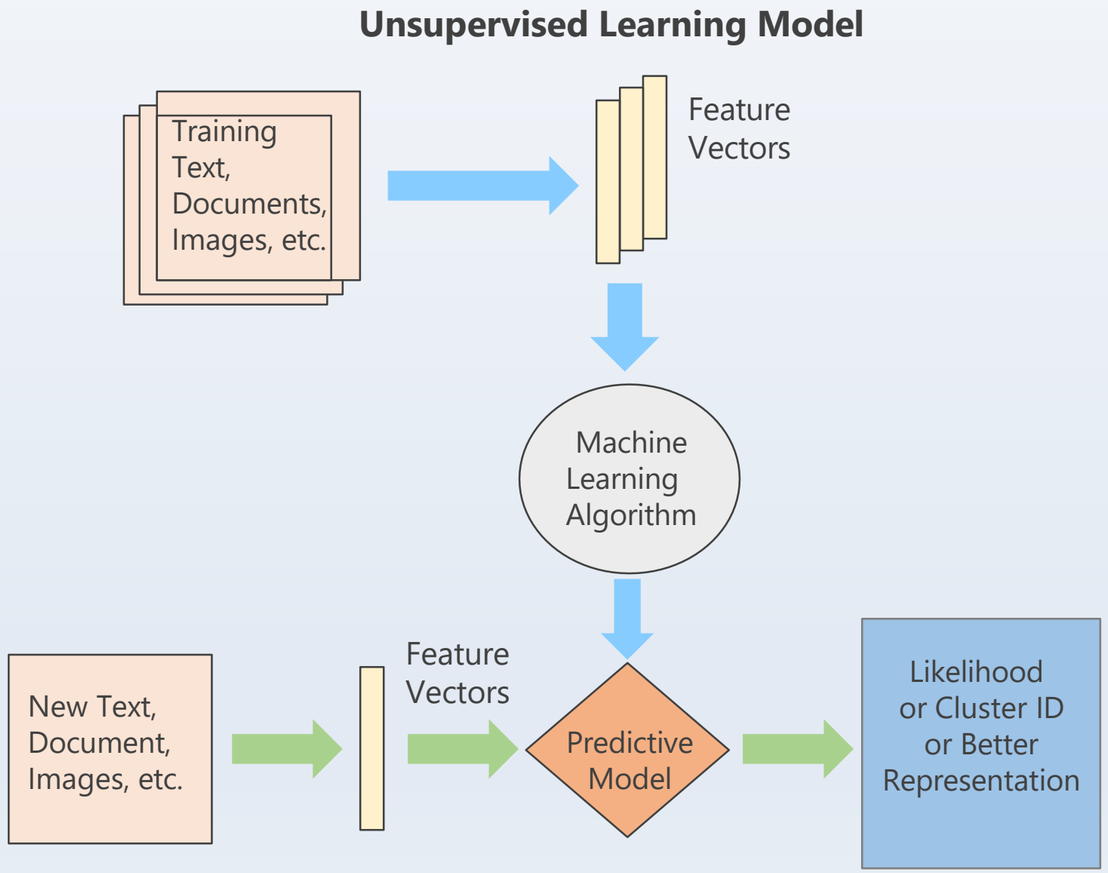
Dr.Prasanalakshmi

Concepts Covered

- Unsupervised Learning
- Hierarchical Clustering
- Dendrogram
- K means clustering

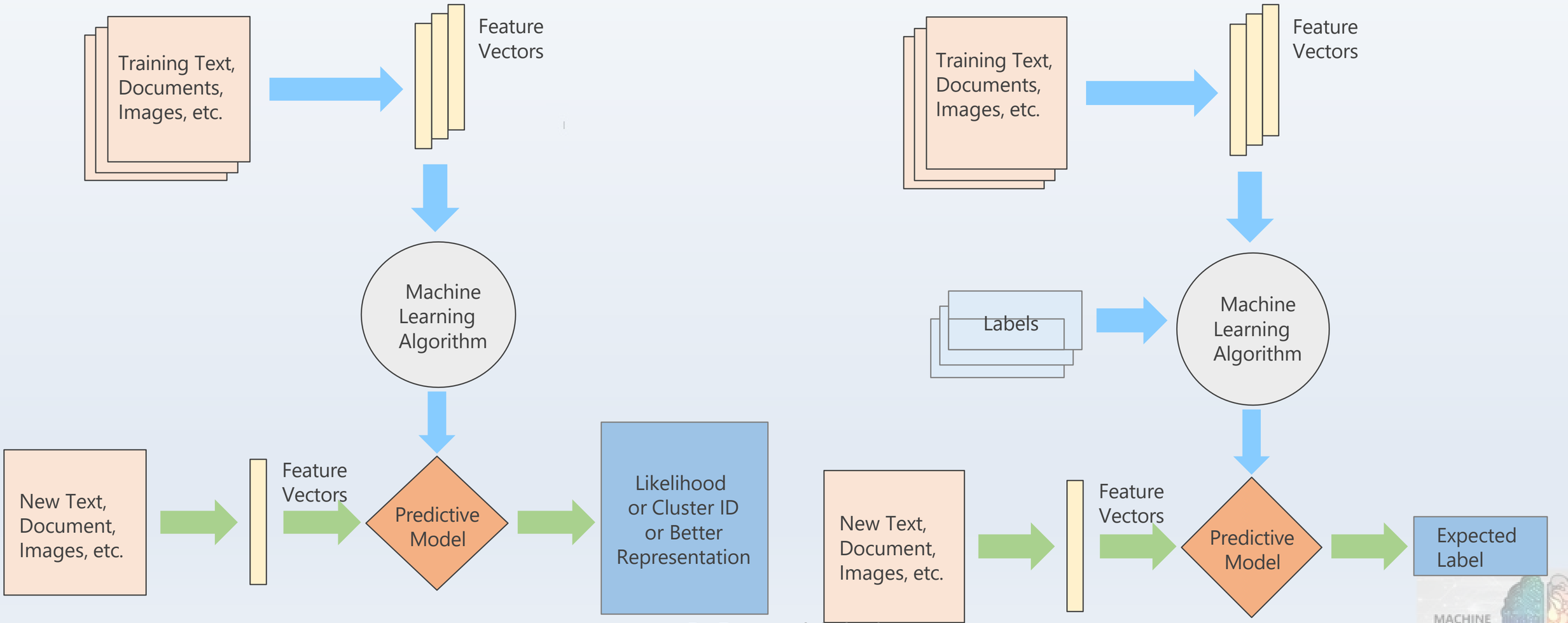
Unsupervised Learning Process Flow

The data has no labels. The machine just looks for whatever patterns it can find.



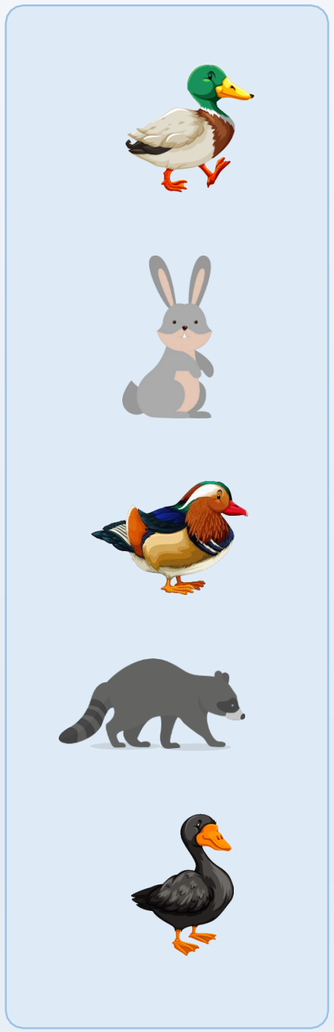
Unsupervised Learning vs. Supervised Learning

The only difference is the labels in the training data

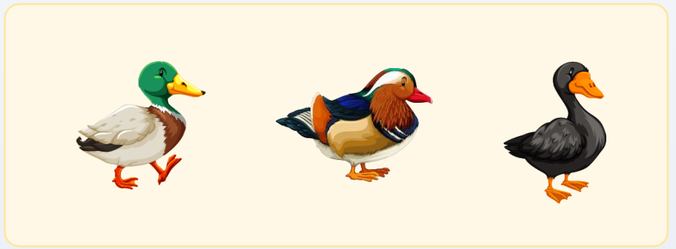


Unsupervised Learning: Example

Clustering like-looking birds/animals based on their features

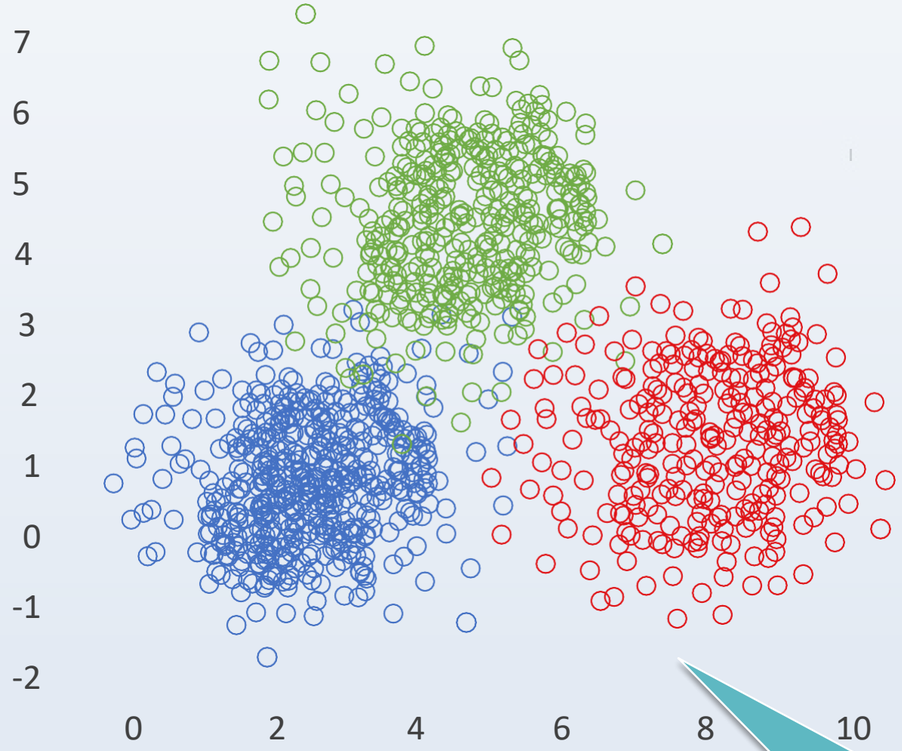


Unsupervised Learning

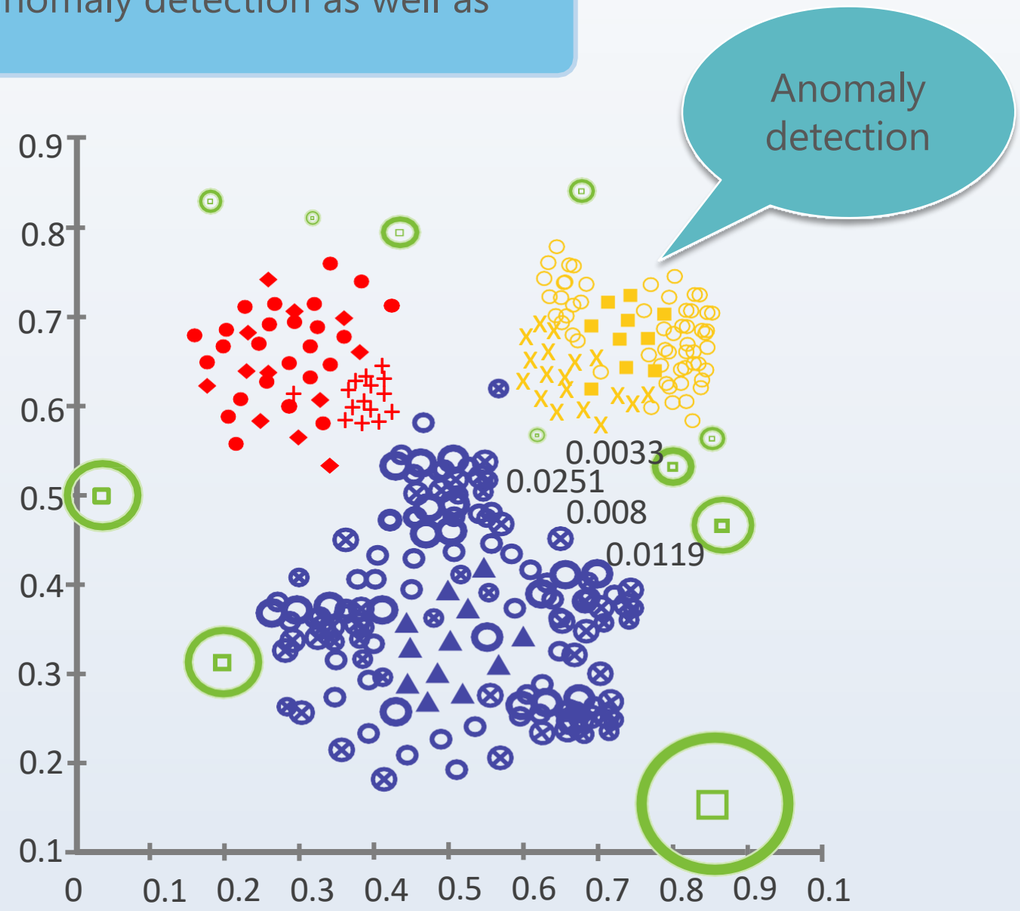


Application of Unsupervised Learning

Unsupervised learning can be used for anomaly detection as well as clustering



Identifying similarities in groups (Clustering)

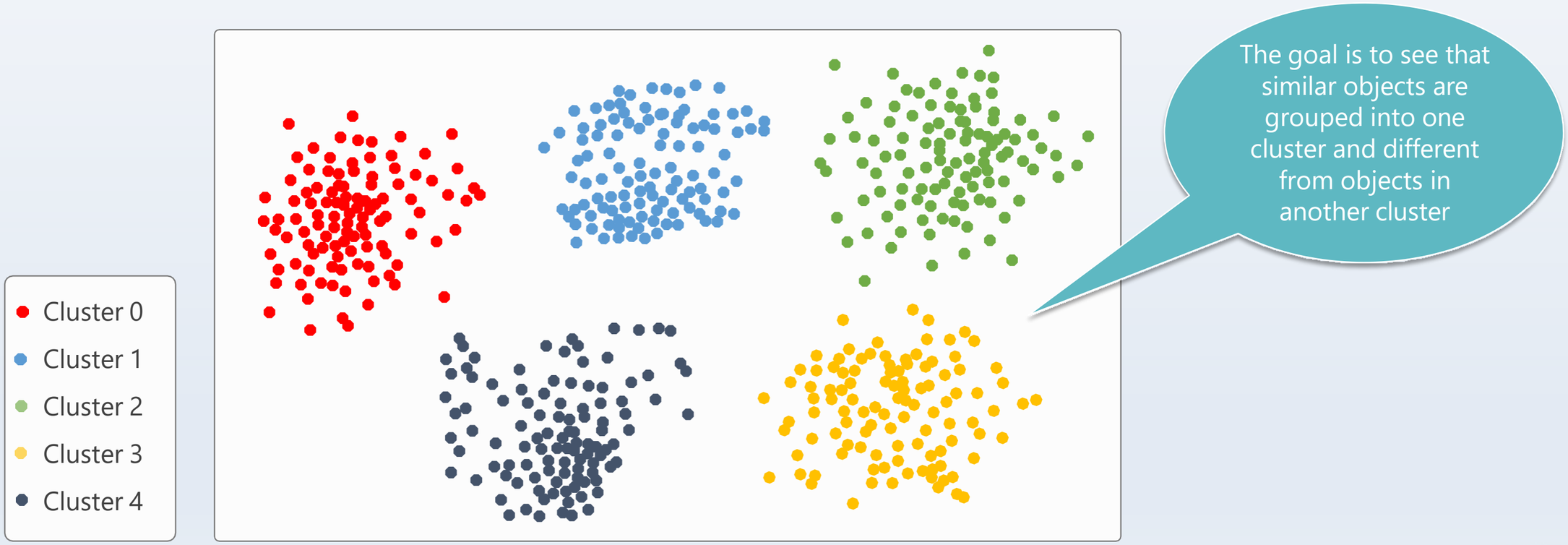


Unsupervised Learning

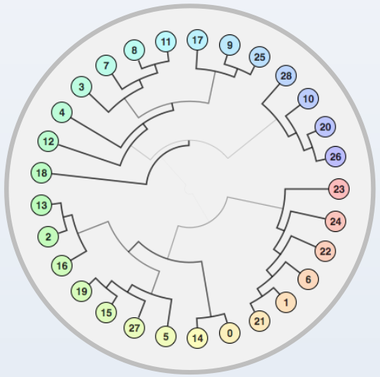
Topic 2: Clustering

Clustering

Grouping objects based on the information found in data that describes the objects or their relationship



Need of Clustering



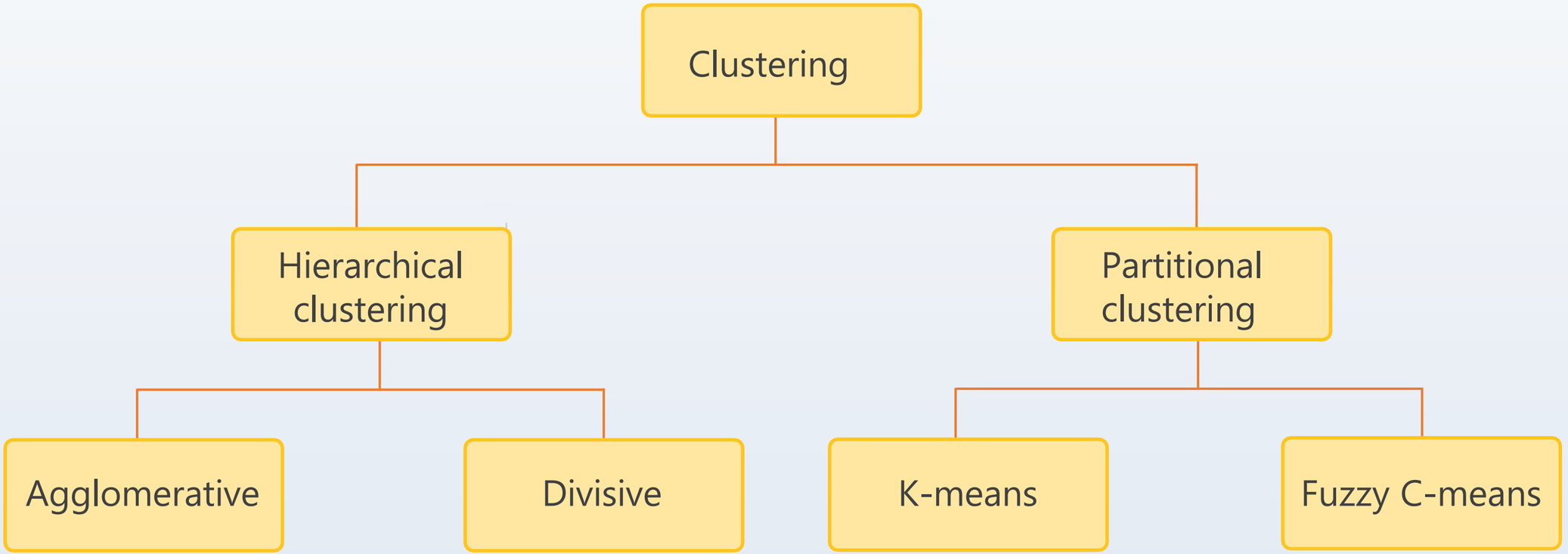
To determine the intrinsic grouping in a set of unlabeled data

To organize data into clusters showing internal structure of the data

To partition the data points

To understand and extract value from large sets of structured and unstructured data

Types of Clustering

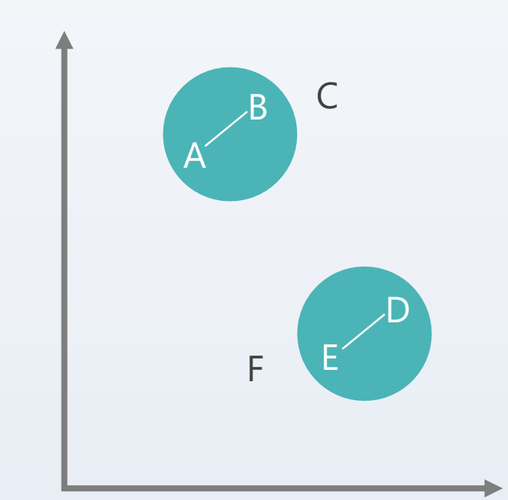


Unsupervised Learning

Topic 3: Hierarchical Clustering

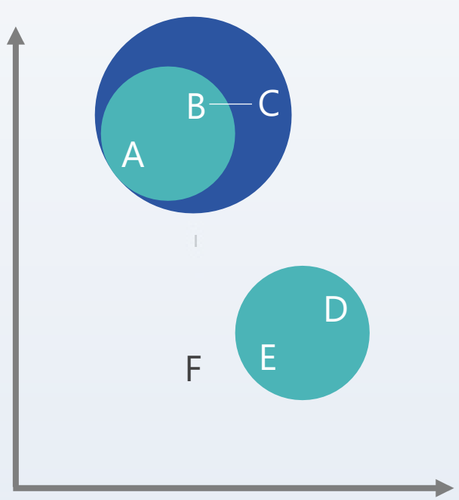
Hierarchical Clustering

Outputs a hierarchy, a structure that is more informative than the unstructured set of clusters returned by flat clustering



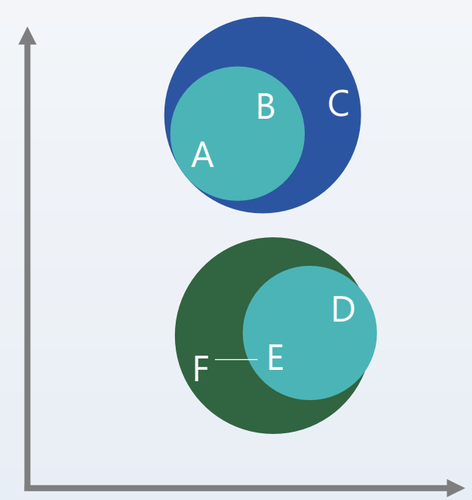
Combine A and B based on similarity
Combine D and E based on similarity

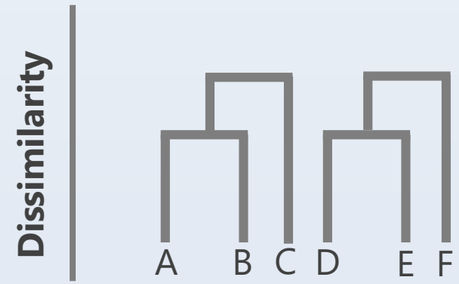
1



Combination of A and B is combined with C

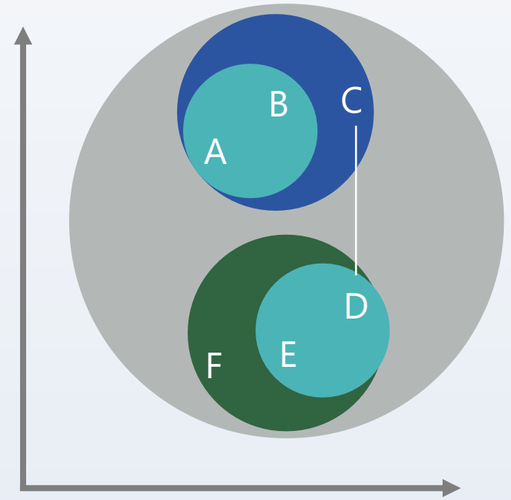
2

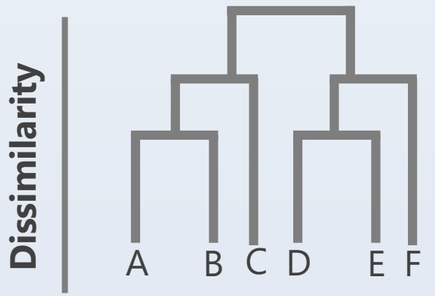




Combination of D and E is combined with F

3





Final tree contains all clusters
Combined into a single cluster

4

Working: Hierarchical Clustering

Assign each item to its own cluster, such that if you have N number of items, you now have N number of clusters

Find the closest (most similar) pair of clusters and merge them into a single cluster. Now you have one less cluster

Compute distances (similarities) between the new cluster and every old cluster

Repeat steps 2 and 3 until all items are clustered into a single cluster of size N

Distance Measures

Complete - Linkage clustering

- Find the maximum possible distance between points belonging to two different clusters

Single - Linkage Clustering

- Find the minimum possible distance between points belonging to two different clusters

Mean - Linkage Clustering

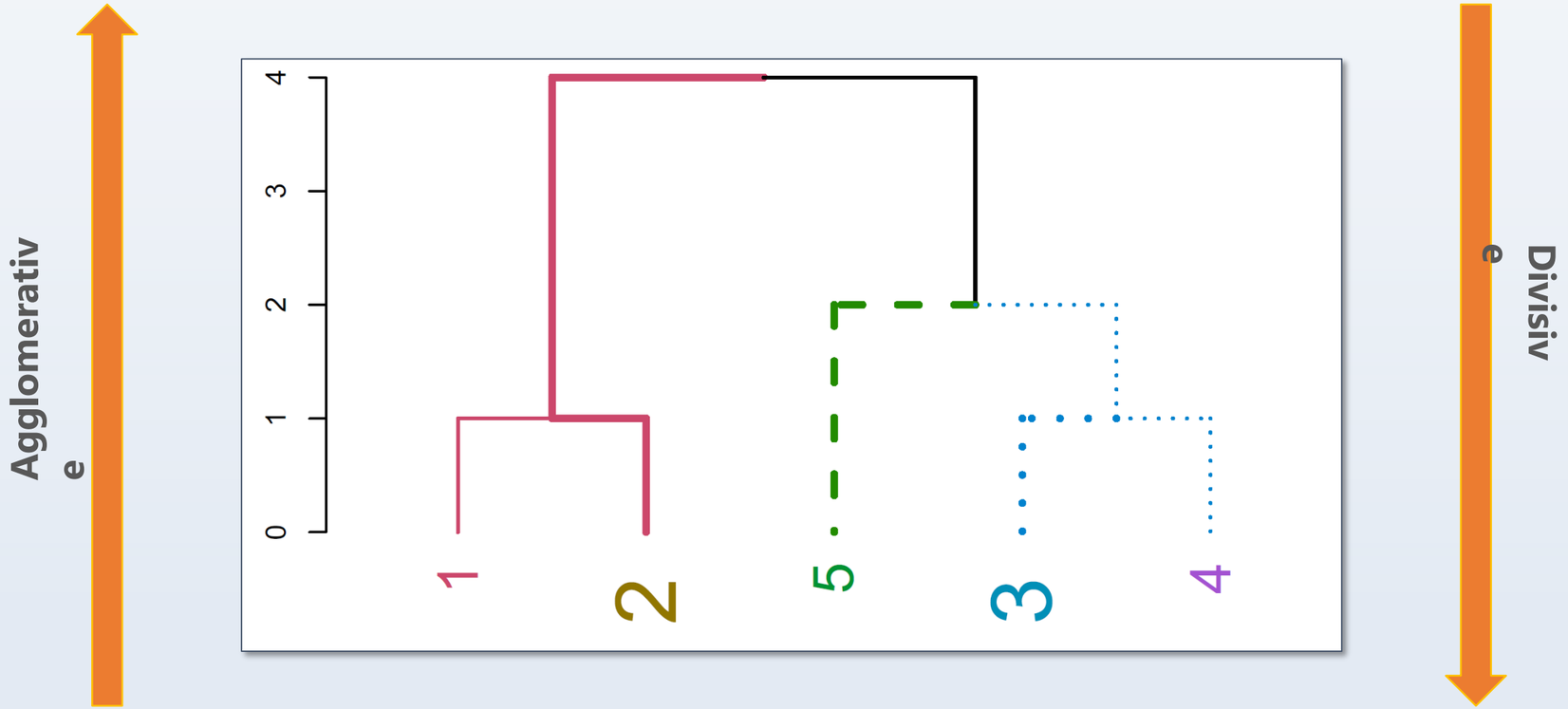
- Find all possible pair-wise distances for points belonging to two different clusters and then calculate the average

Centroid - Linkage Clustering

- Find the centroids of each cluster and calculate the distance between them

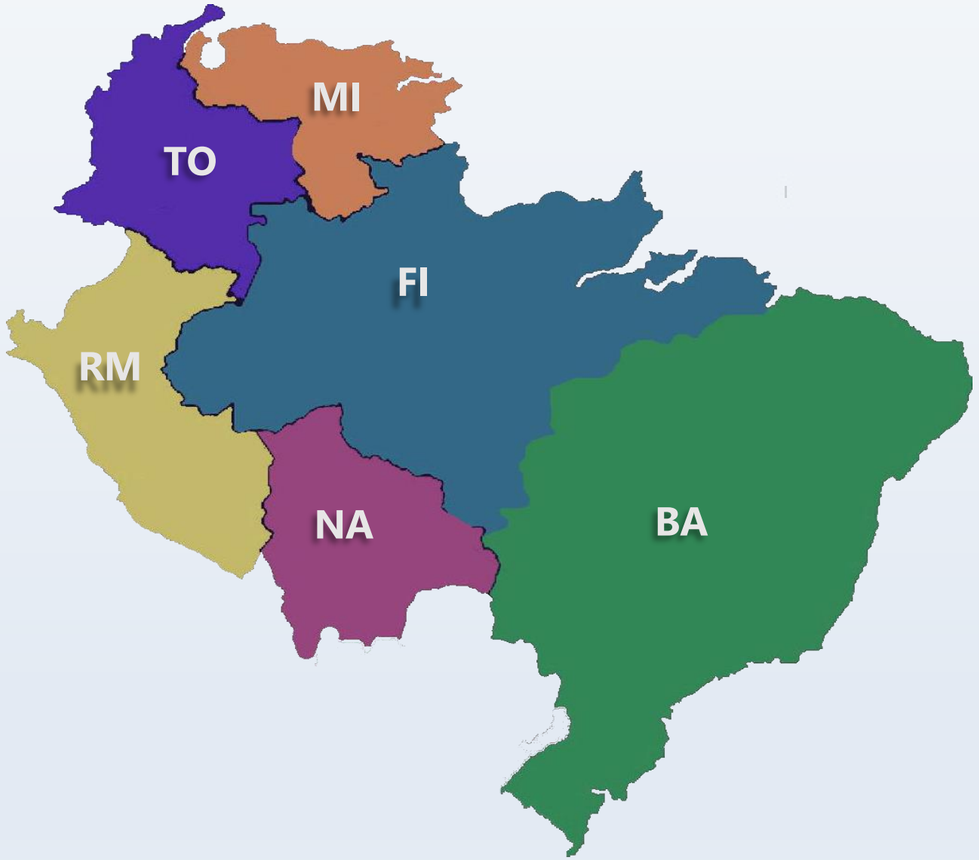
The Dendrogram

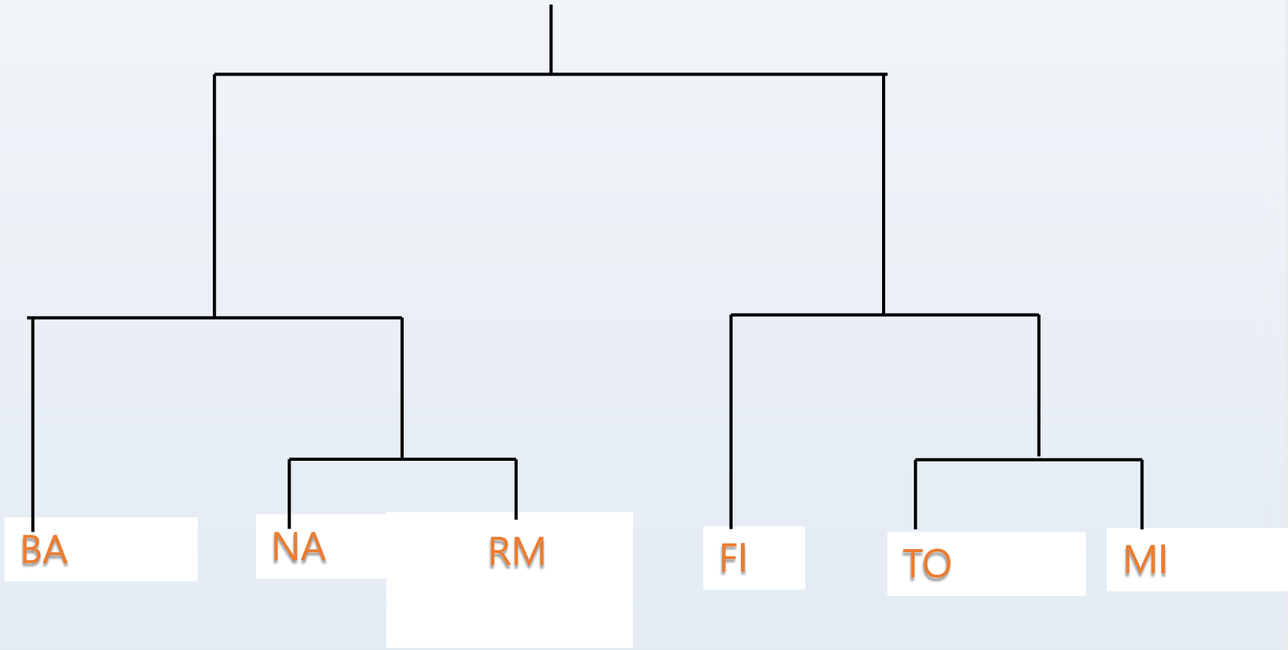
Dendrogram ((in Greek, *dendro* means tree and *gramma* means drawing) is a tree diagram frequently used to illustrate the arrangement of the clusters produced by hierarchical clustering.



Hierarchical Clustering: Example

A hierarchical clustering of distances between cities in kilometers





Step1: Data Import

Code

```
import pandas as pd
import numpy as np
customer_data = pd.read_csv('shopping_data.csv’)
customer_data
```

Out[6]:

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40
5	6	Female	22	17	76
6	7	Female	35	18	6
7	8	Female	23	18	94
8	9	Male	64	19	3
9	10	Female	30	19	72
10	11	Male	67	19	14
11	12	Female	35	19	99
12	13	Female	58	20	15
13	14	Female	24	20	77
14	15	Male	37	20	13

Dr.Prasanalakshmi

MACHINE LEARNING

Step 2: Filter Columns

Discard all the data, except annual income (in thousands of dollars) and spending score (1-100)

Code

```
data = customer_data.iloc[:,3:5].values
data
```

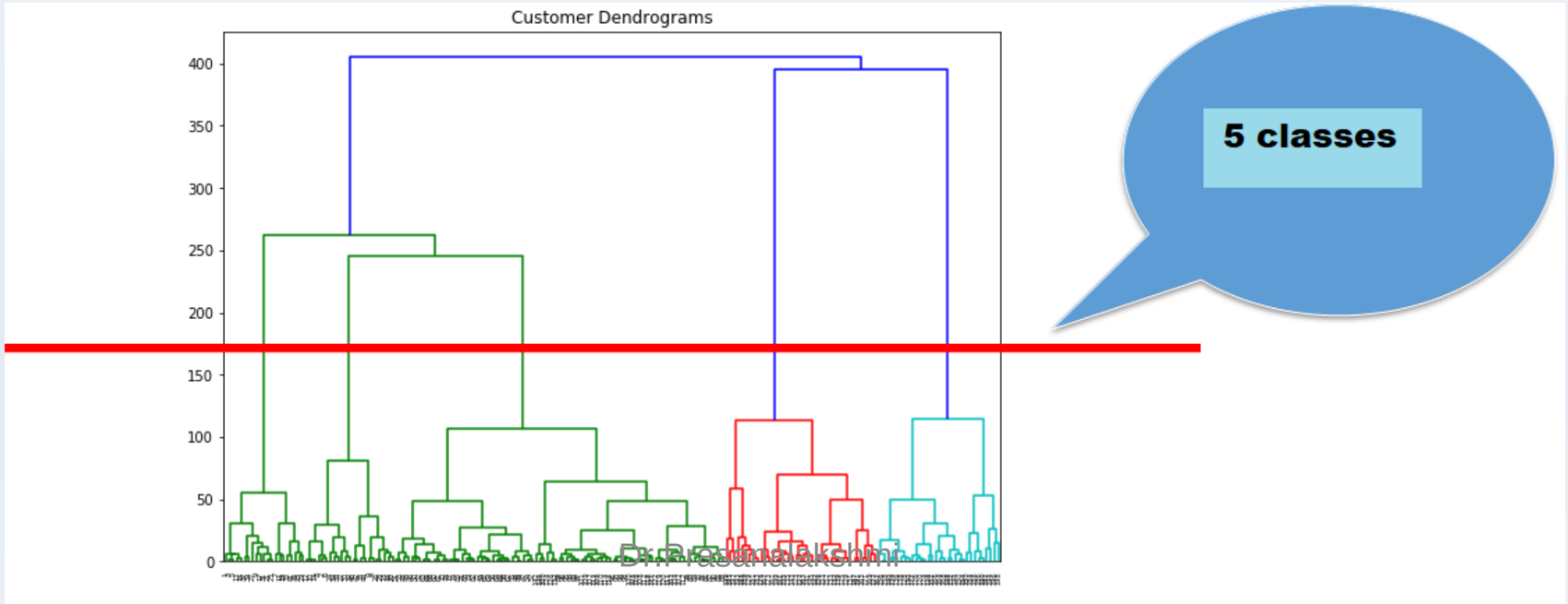
```
Out[8]: array([[ 15,  39],
               [ 15,  81],
               [ 16,   6],
               [ 16,  77],
               [ 17,  40],
               [ 17,  76],
               [ 18,   6],
               [ 18,  94],
               [ 19,   3],
               [ 19,  72],
               [ 19,  14],
               [ 19,  99],
               [ 20,  15],
               [ 20,  77],
               [ 20,  13],
               [ 20,  79],
               [ 21,  35],
               [ 21,  66],
               [ 23,  29],
```


MACHINE LEARNING

Step 3: Create Dendrograms

Code

```
import matplotlib.pyplot as plt
%matplotlib inline
import scipy.cluster.hierarchy as shc
plt.figure(figsize=(10,7))
plt.title('Customer Dendrograms')
dend = shc.dendrogram(shc.linkage(data,method='ward'))
```



Step 4: Agglomerative Clustering

Since there are five clusters, group the data points into these five clusters

Code

```
from sklearn.cluster import AgglomerativeClustering
cluster = AgglomerativeClustering(n_clusters=5, affinity='euclidean',
linkage='ward')
cluster.fit_predict(data)
```

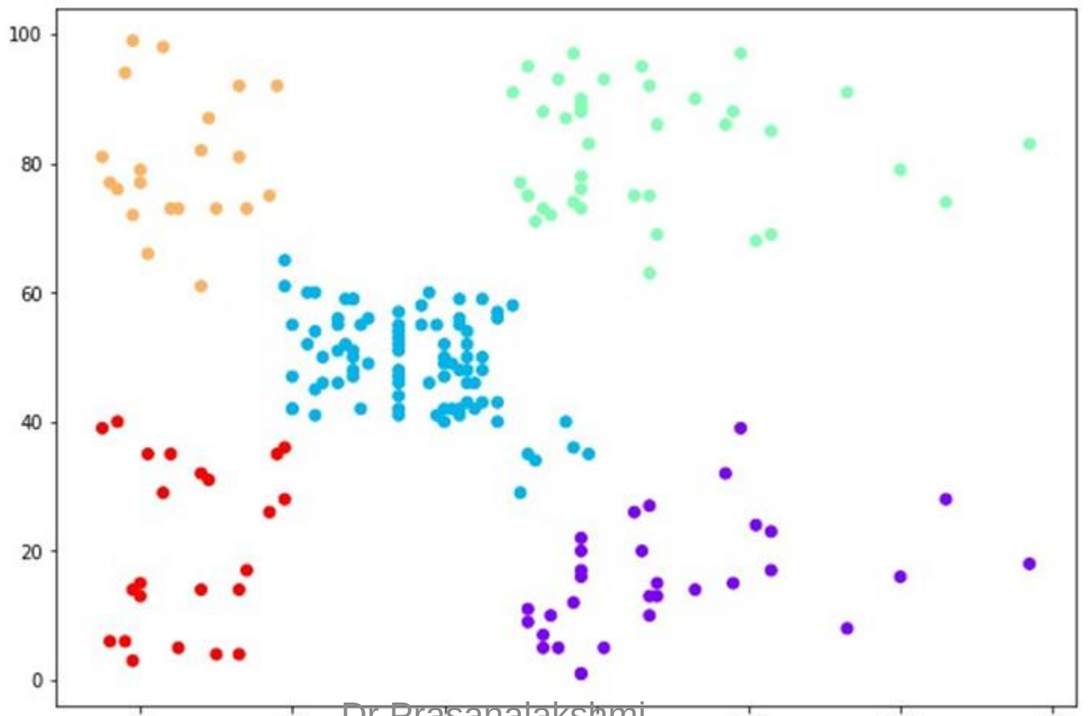
```
Out[13]: array([4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3,
                4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 3, 4, 1,
                4, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 1, 2, 0, 2, 0, 2,
                1, 2, 0, 2, 0, 2, 0, 2, 0, 2, 1, 2, 0, 2, 1, 2, 0, 2, 0, 2, 0, 2,
                0, 2, 0, 2, 0, 2, 1, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2,
                0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2,
                0, 2], dtype=int64)
```


Step 4: Plotting the Clusters

Code

```
plt.figure(figsize=(10, 7))
plt.scatter(data[:,0], data[:,1], c=cluster.labels_, cmap='rainbow')
```

```
Out[14]: <matplotlib.collections.PathCollection at 0x18268c22cc0>
```



Dr.Prasanalakshmi

MACHINE LEARNING

Unsupervised Learning

Topic 4: K-means Clustering

K-means Algorithm: Steps

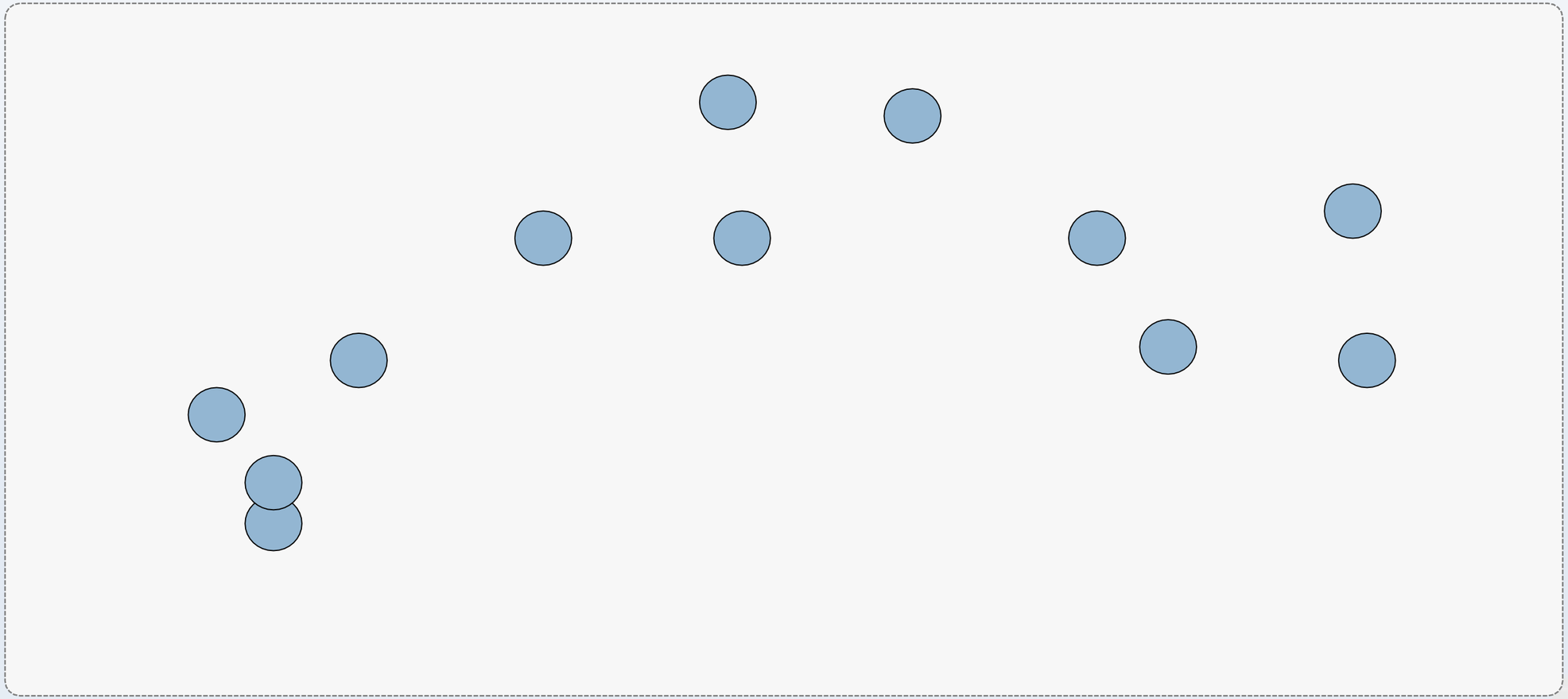
1. Randomly chooses k datapoints as initial centroids
2. Assigns each datapoint closest to the centroid
3. Calculates new cluster centroids
4. Checks if the convergence criterion is met

©Simplilearn. All rights reserved.

MACHINE LEARNING simplilearn

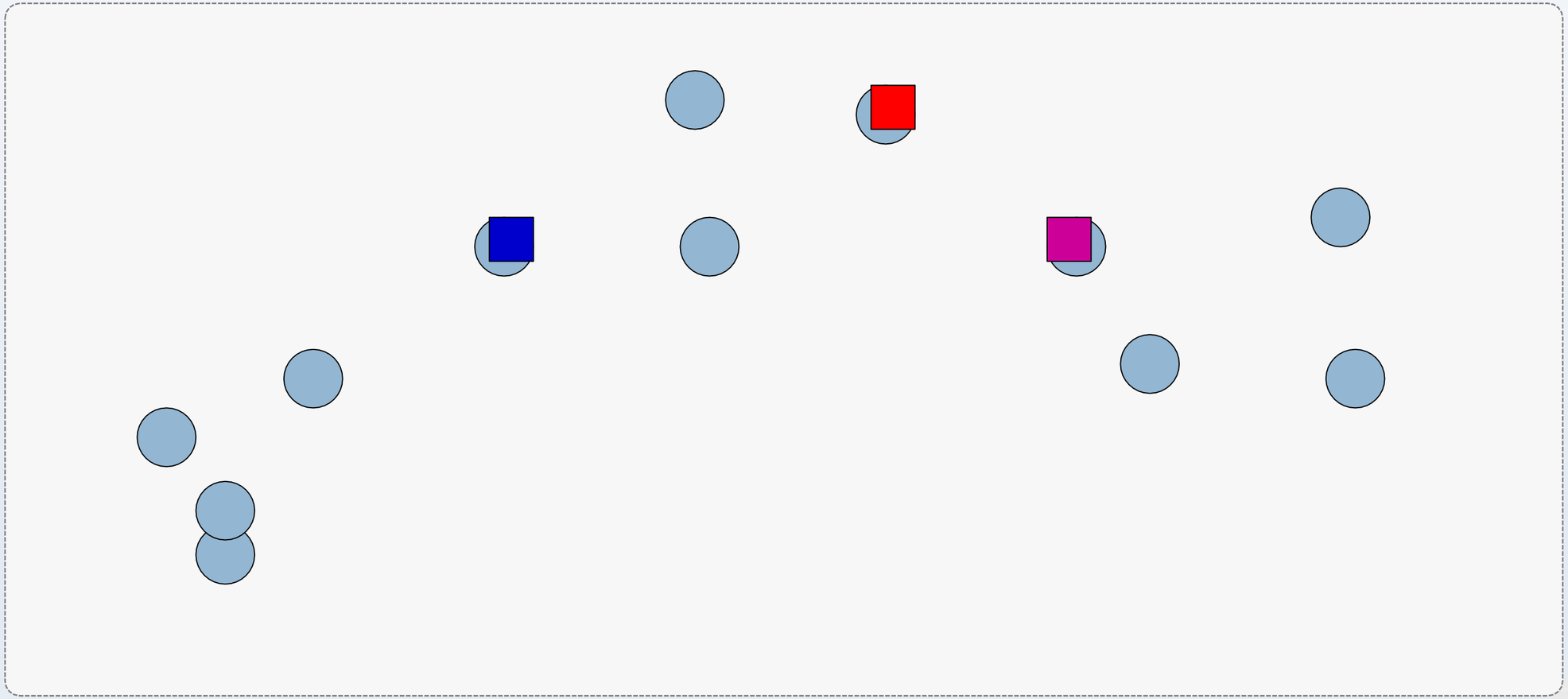
K-means: Example

Consider the below datapoints



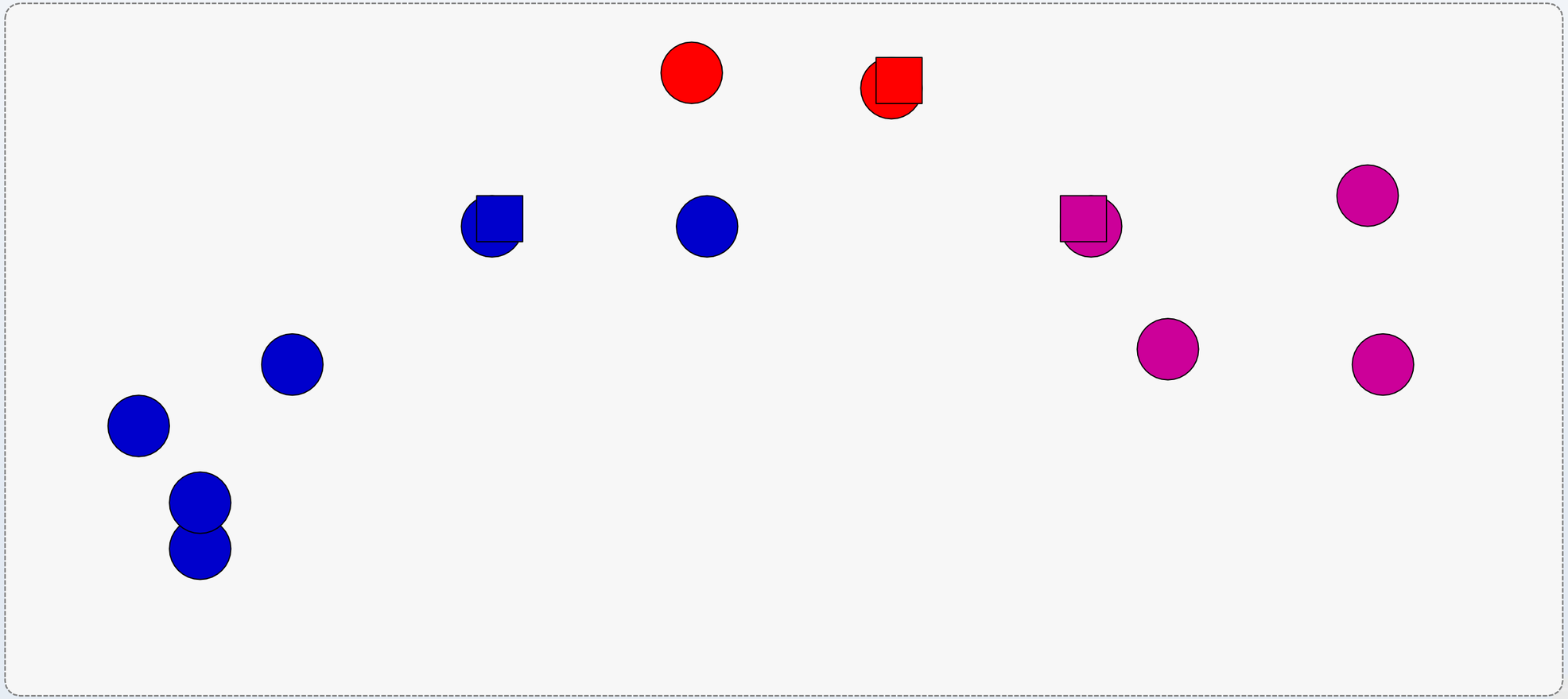
K-means: Example (Contd.)

Initialize centers randomly



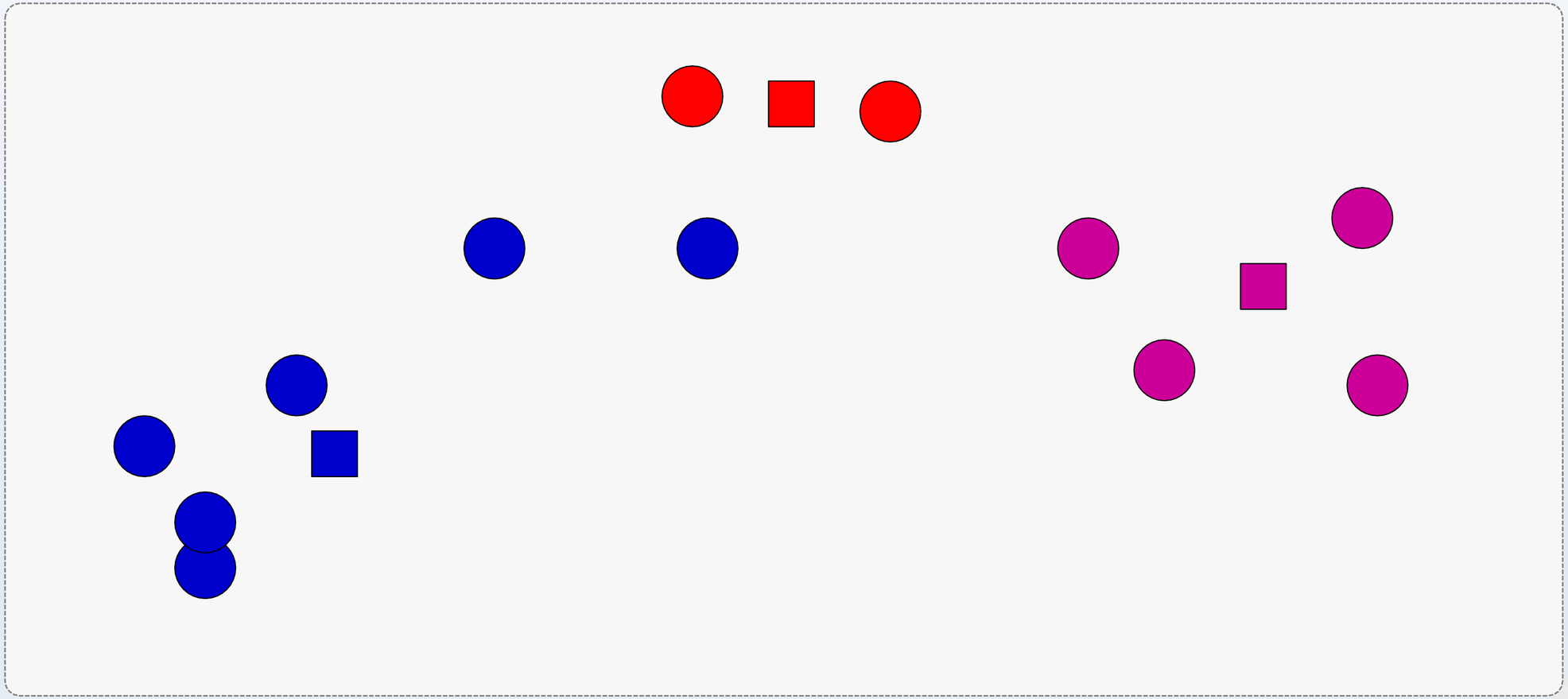
K-means: Example (Contd.)

Assign points to the nearest center



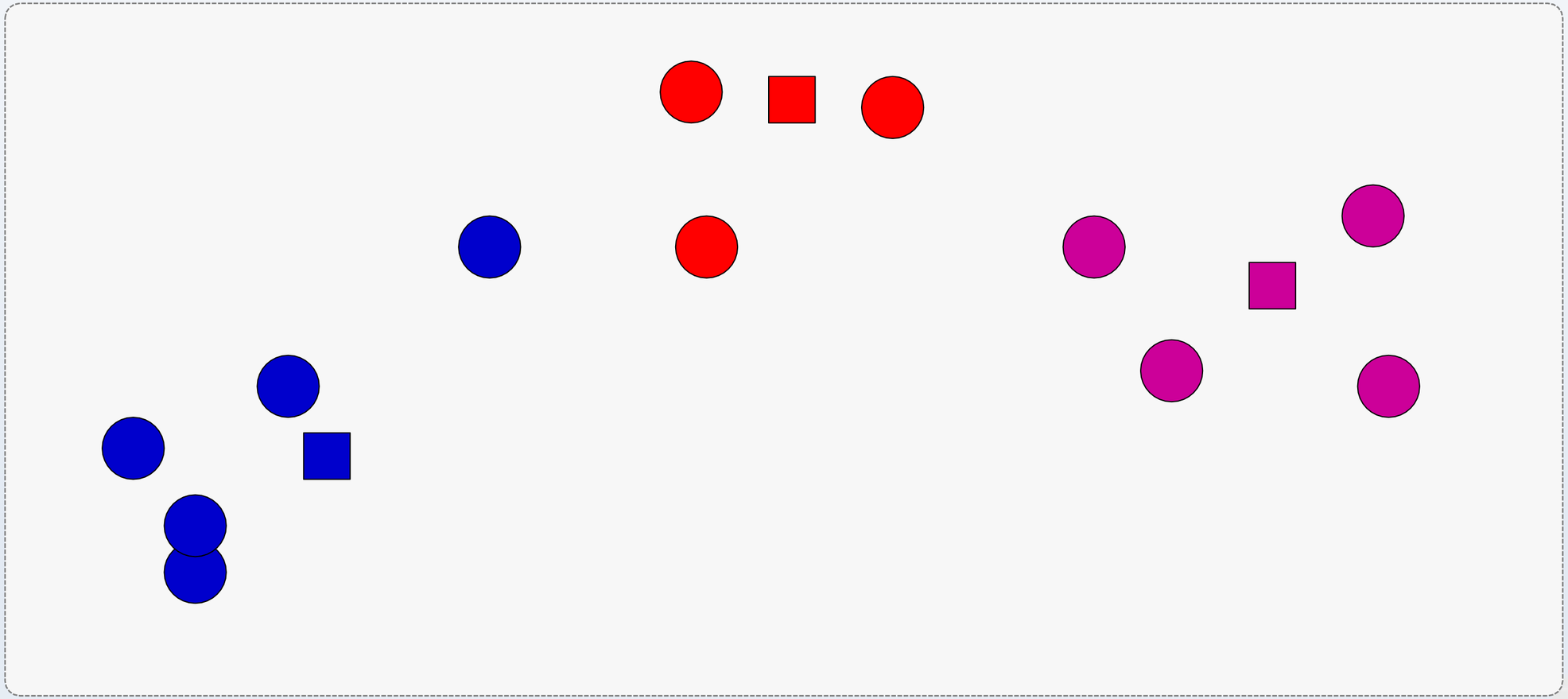
K-means: Example (Contd.)

Readjust centers



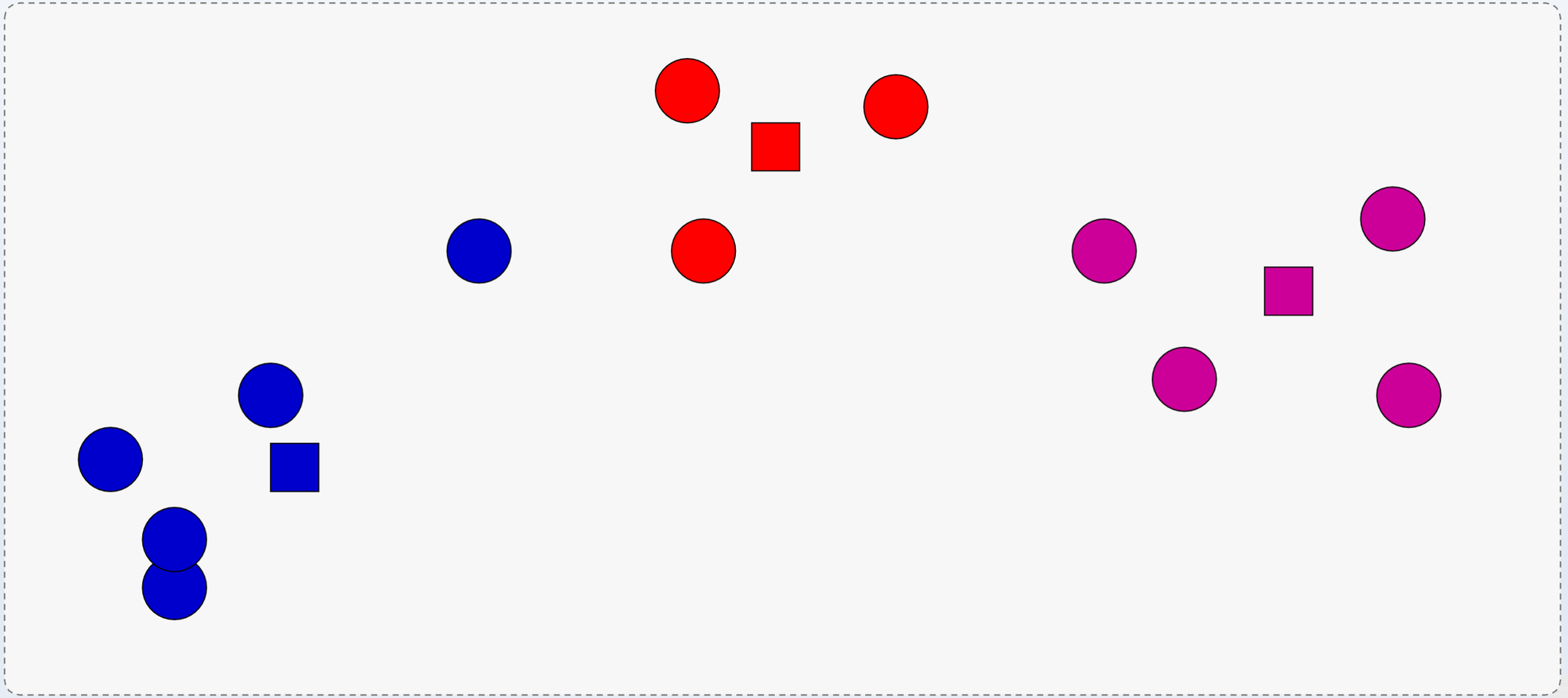
K-means: Example (Contd.)

Assign points to the nearest center



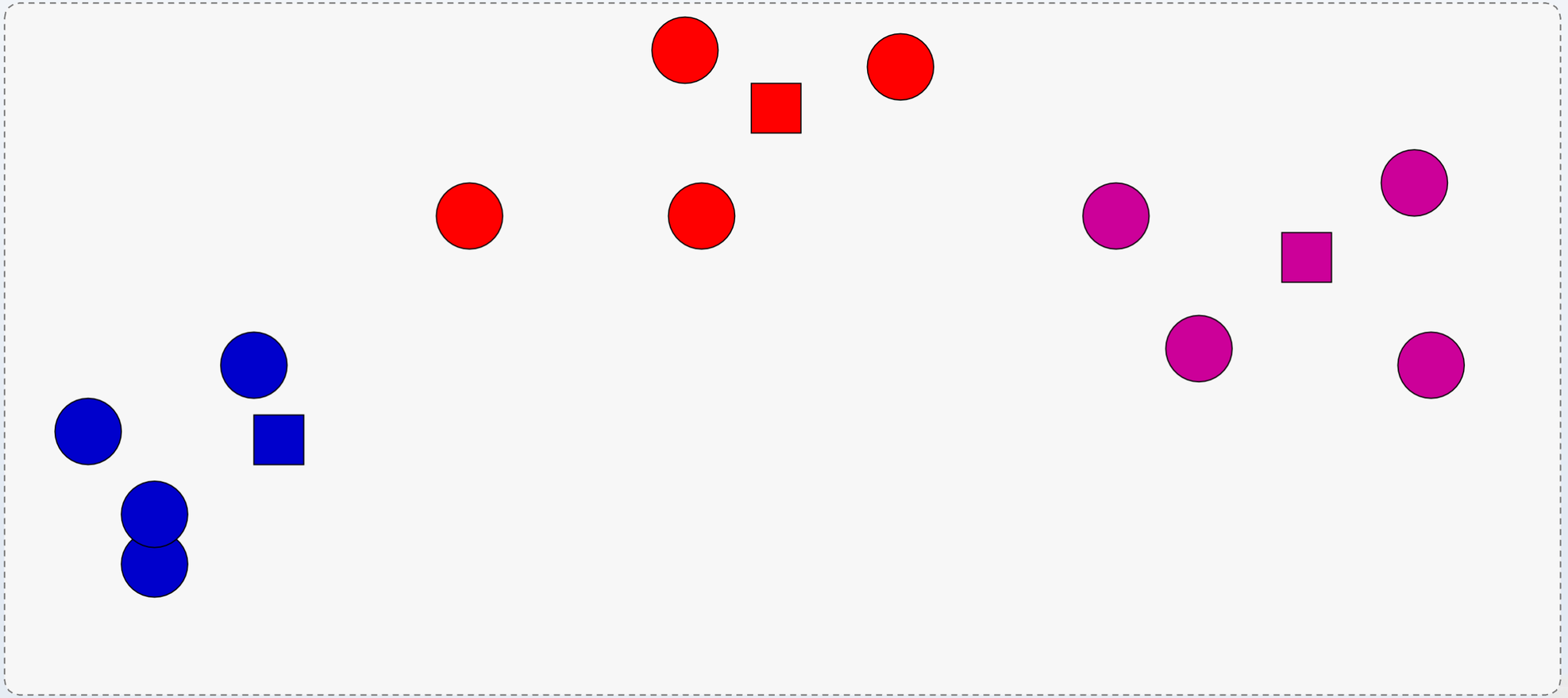
K-means: Example (Contd.)

Re-adjust centres



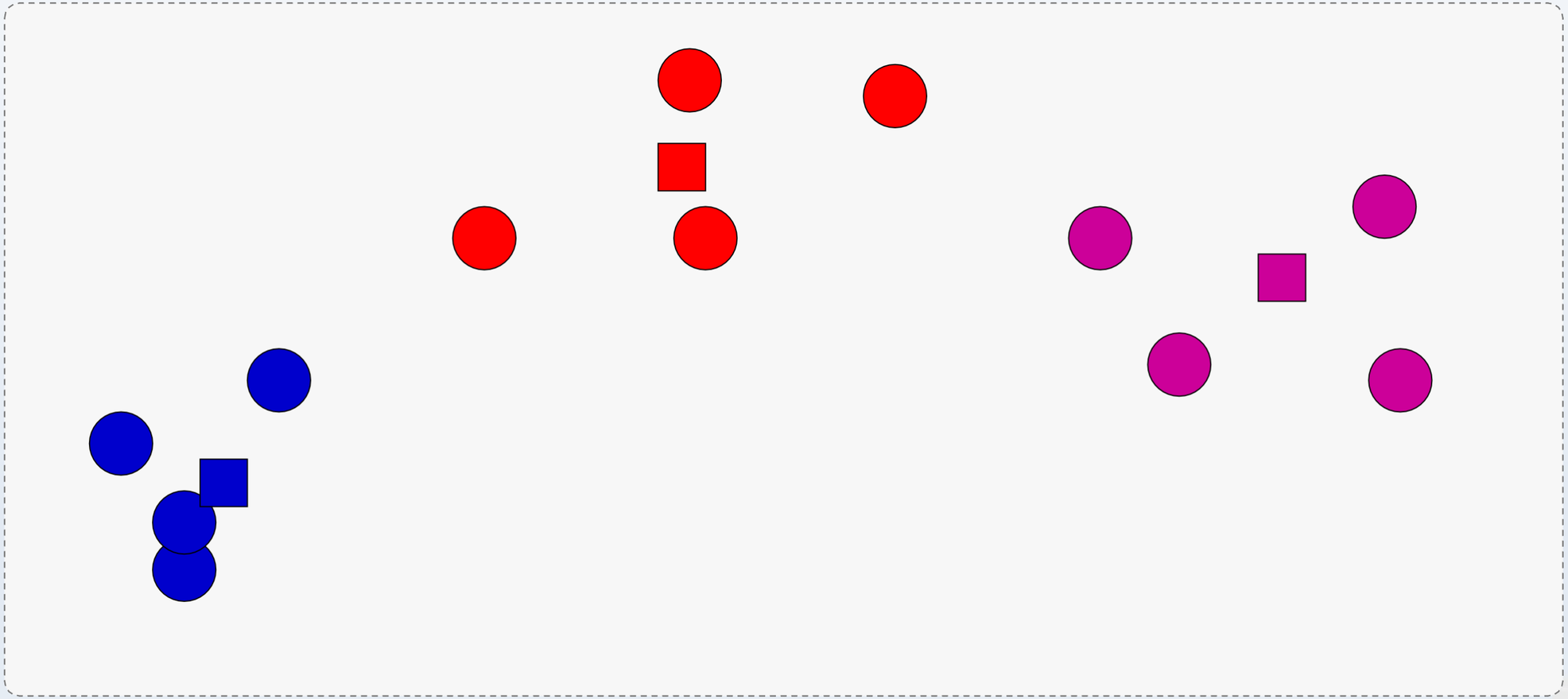
K-means: Example (Contd.)

Assign points to the nearest center



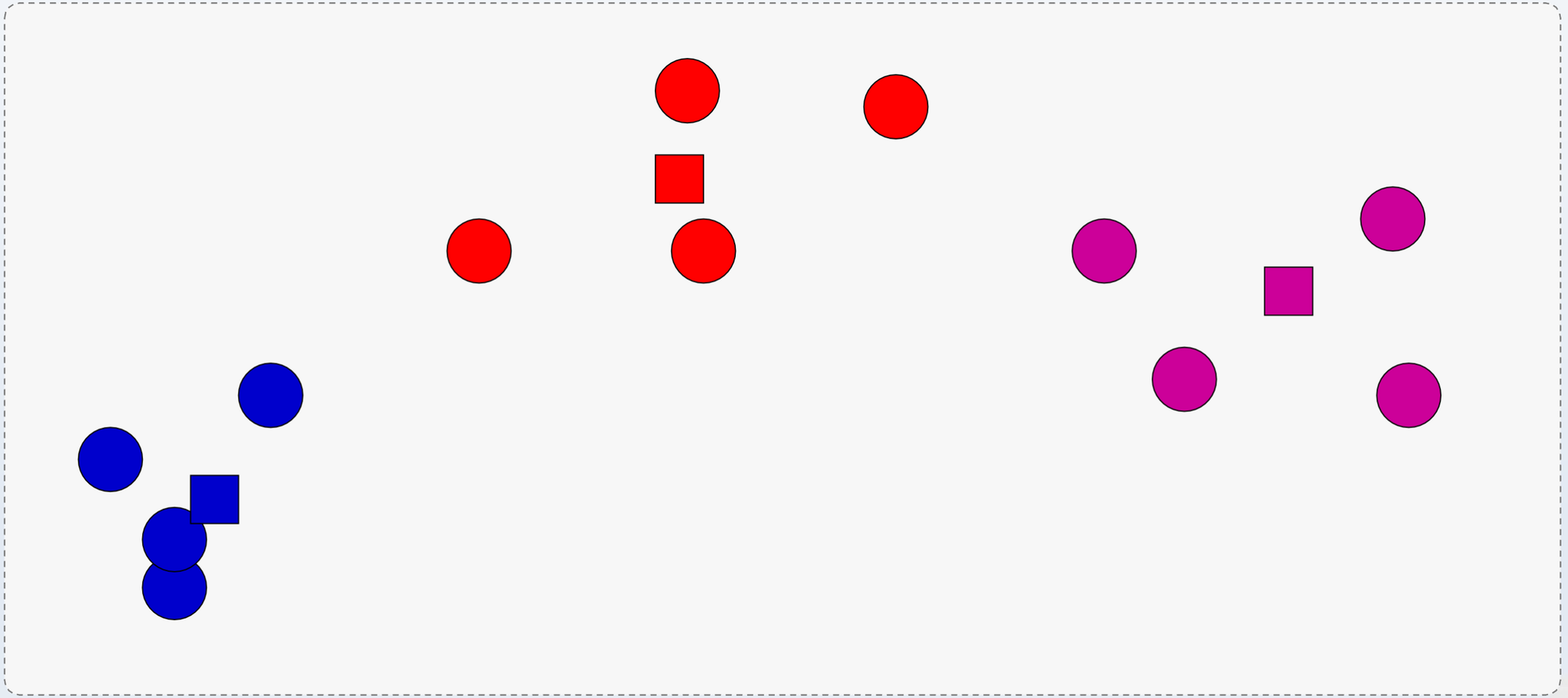
K-means: Example (Contd.)

Readjust centers



K-means: Example (Contd.)

Assign points to the nearest center

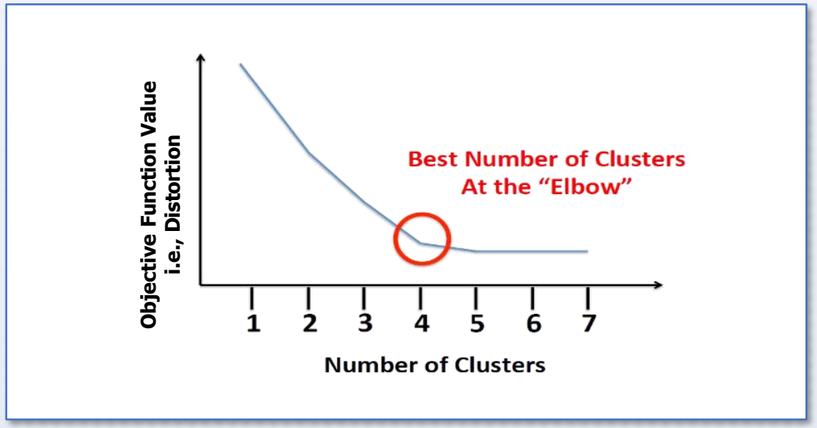


Optimal Number of Clusters

If you plot k against the SSE, you will see that the error decreases as k increases

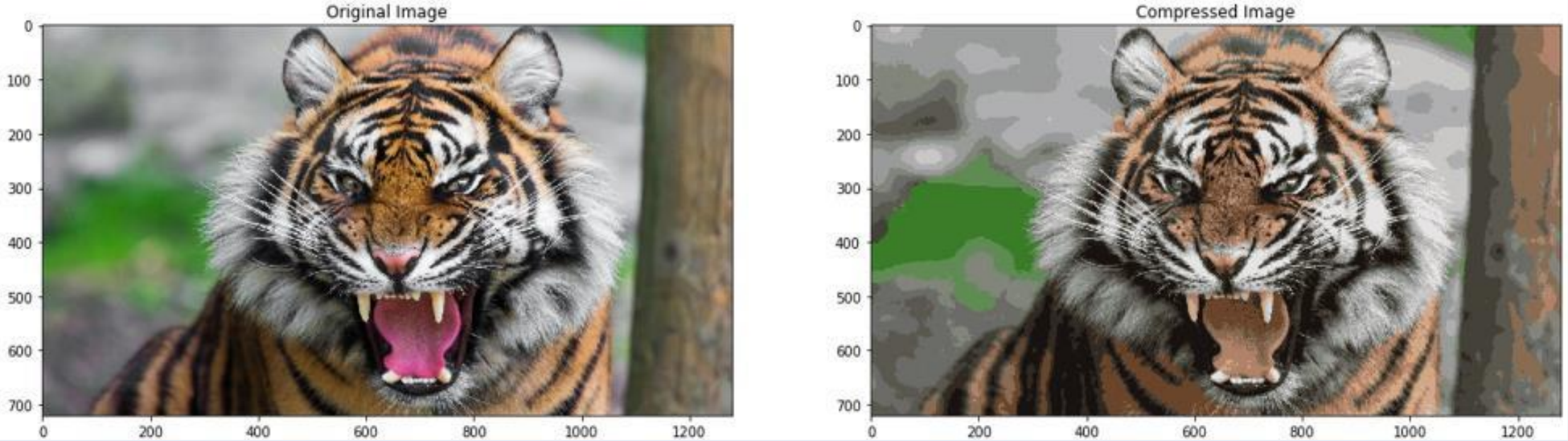
This is because their size decreases and hence distortion is also smaller"

The goal of elbow method is to choose k where SSE decreases abruptly



Elbow Plot

Example :Original Plot vs. Compressed Image

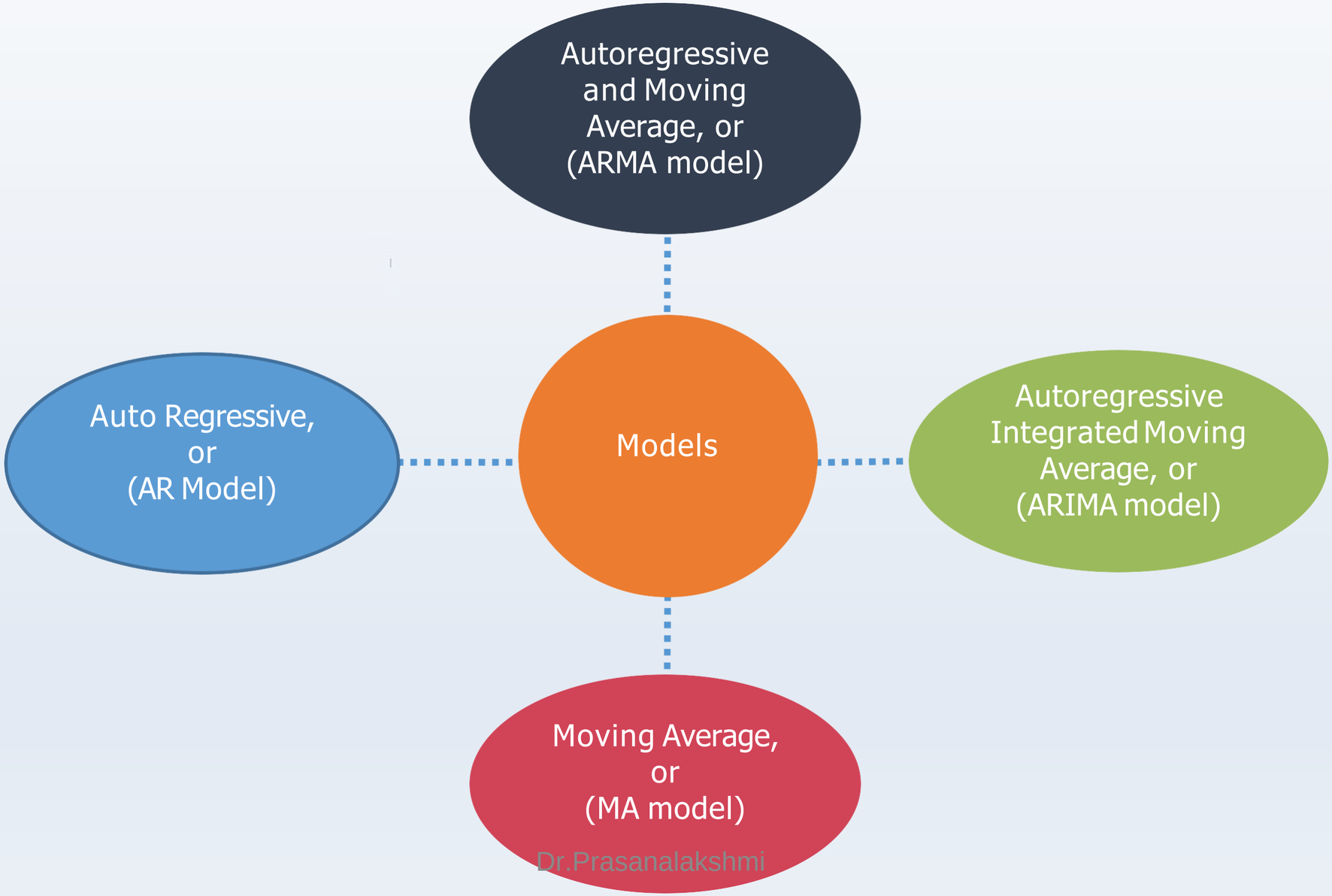


MACHINE LEARNING

Time Series Modeling

Topic 3: Various Time Series Models

Time Series Models



Dr.Prasanalakshmi

Steps in Time Series Forecasting

Step 01	Visualize the time series – check for trend, seasonality, or random patterns
Step 02	Stationarize the series using decomposition or differencing techniques
Step 03	Plot ACF / PACF and find (p, d, q) parameters
Step 04	Build ARIMA model
Step 05	Make predictions using final ARIMA model

Machine Learning

Lesson 8: Ensemble Learning

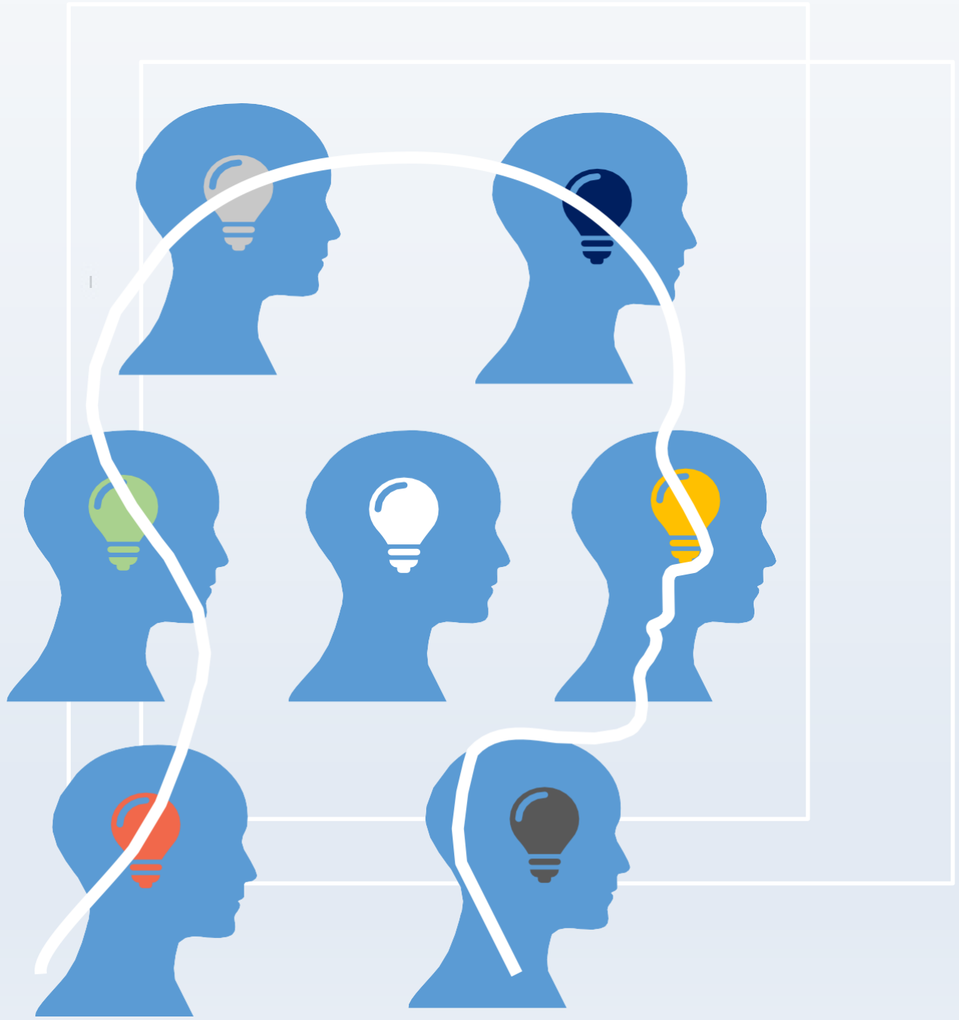
Dr.Prasanalakshmi

Concepts Covered

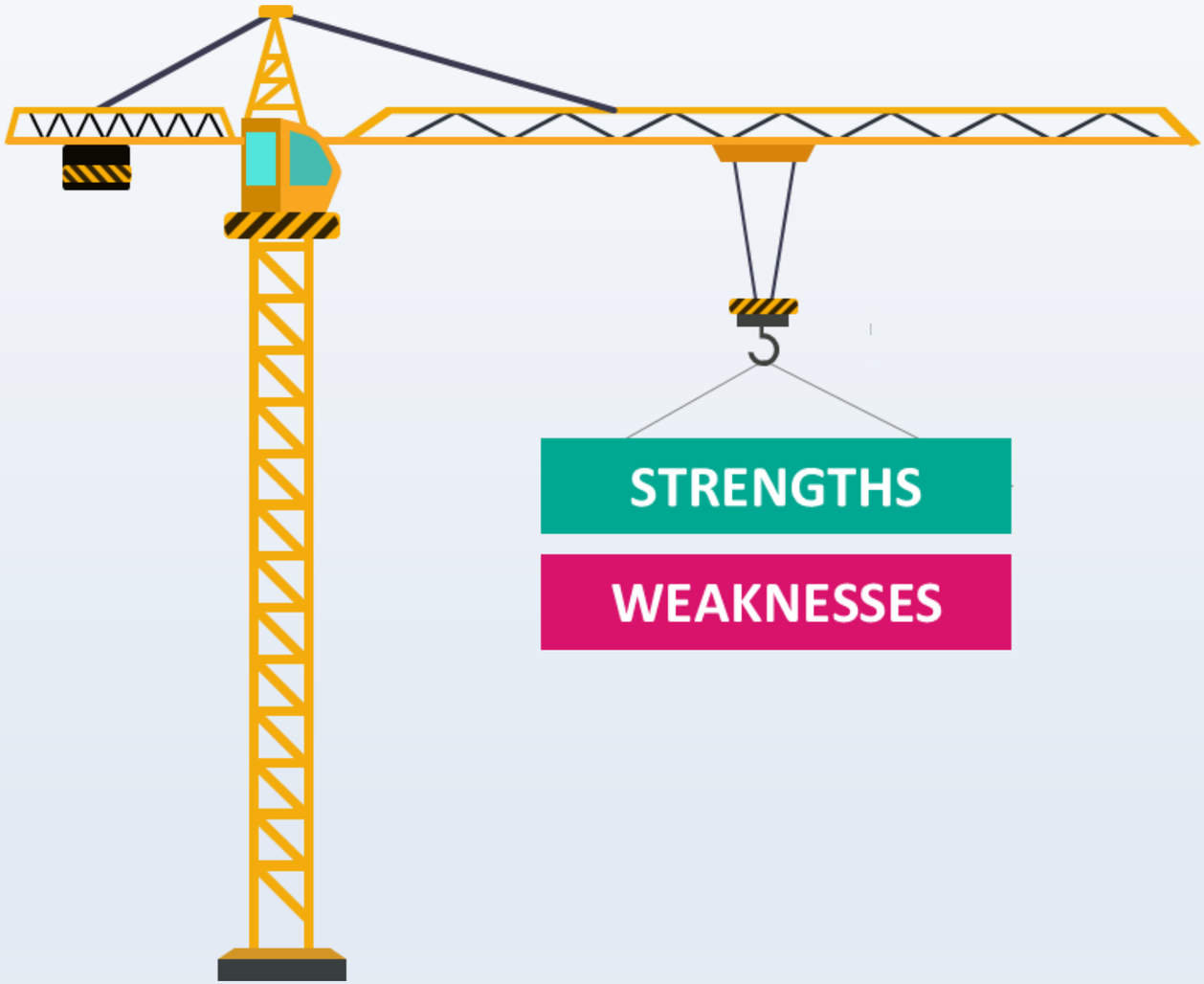
- Ensemble Learning
- Bagging and Boosting Algorithms
- Model Selection
- Cross-validation

Definition

Ensemble techniques combine individual models together to improve the stability and predictive power of the model



Ideology



Certain models do well in modeling one aspect of the data, while others do well in modeling another

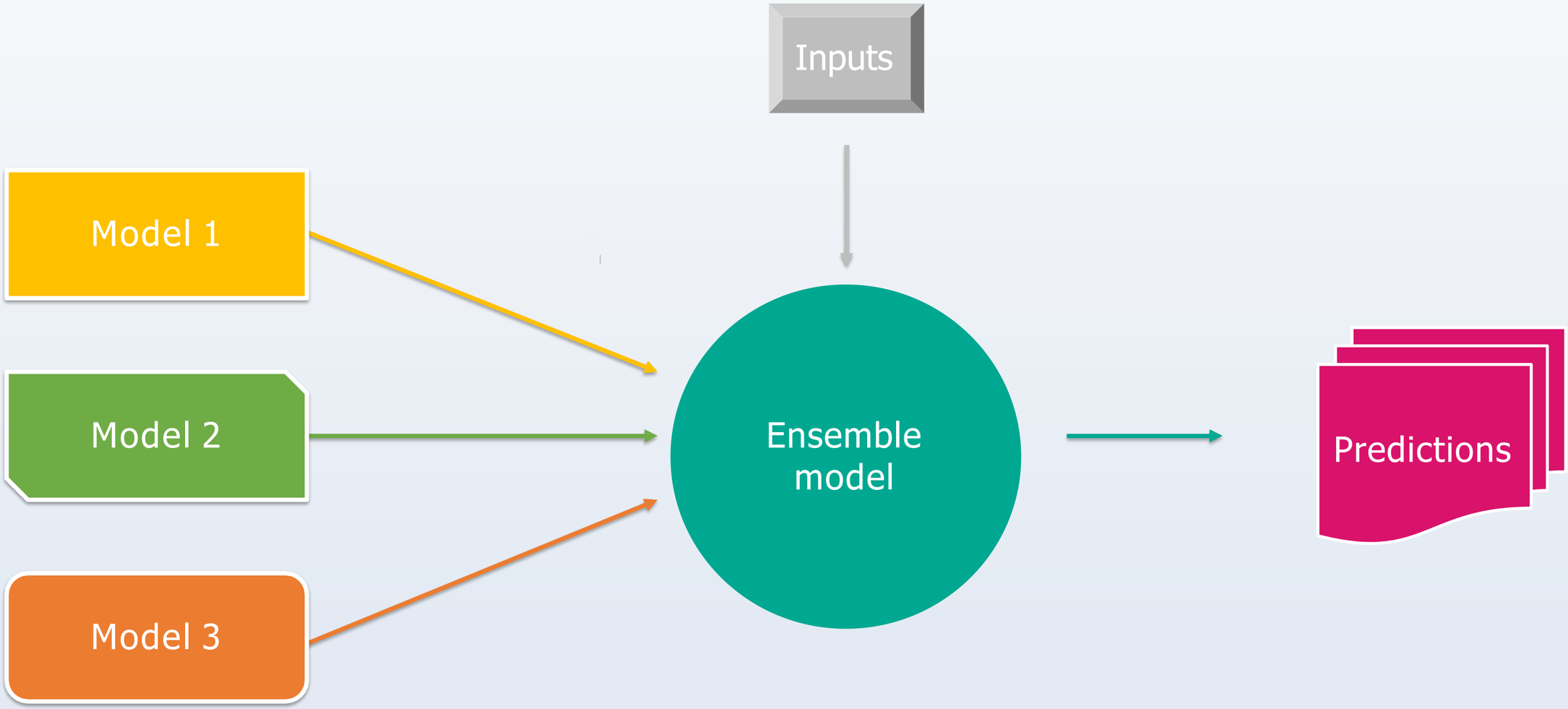
Instead of learning a single complex model, learn several simple models and combine their output to produce the final decision

In ensemble learning, other models strength performs offset on individual model variances and biases

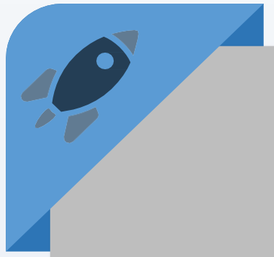
Ensemble learning will provide a composite prediction where the final accuracy is better than the accuracy of individual models

MACHINE LEARNING

Working



Significance



Robustness

Ensemble models incorporate the predictions from all the base learners

01

Accuracy

Ensemble models deliver accurate predictions and have improved performances

02

Ensemble Learning Methods

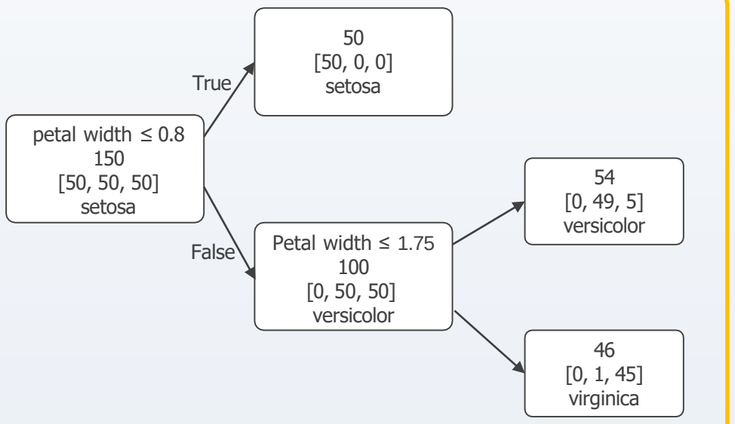
Techniques to create an Ensemble model

Combine all “weak” learners to form an ensemble

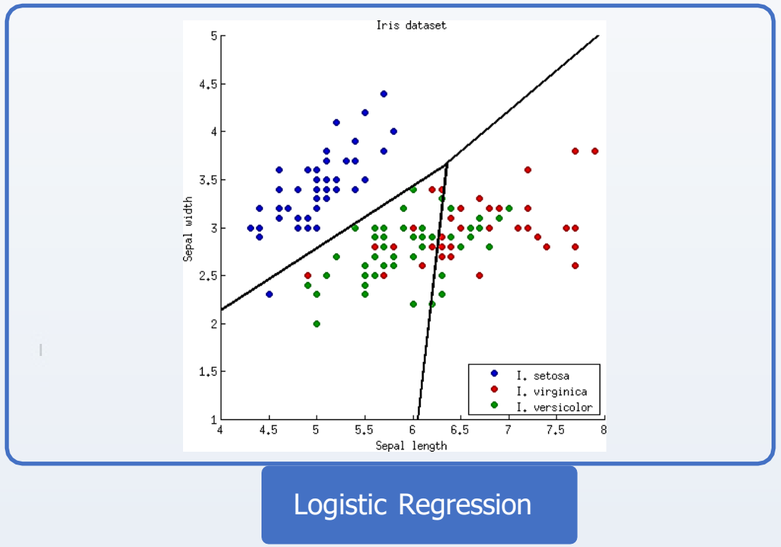
OR

Create an ensemble of well-chosen strong and diverse models

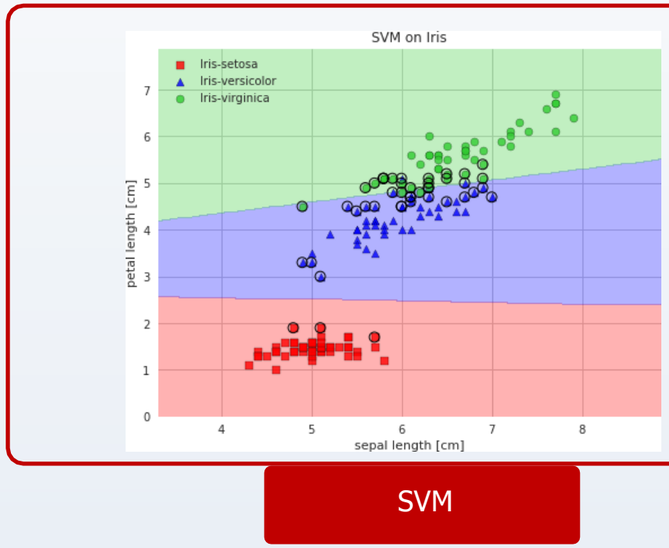
Averaging



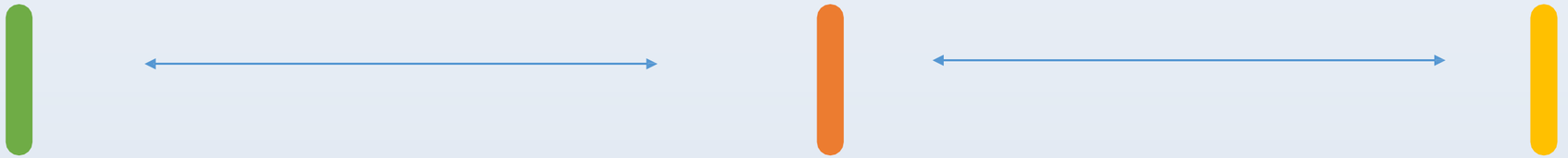
Decision Tree



Logistic Regression



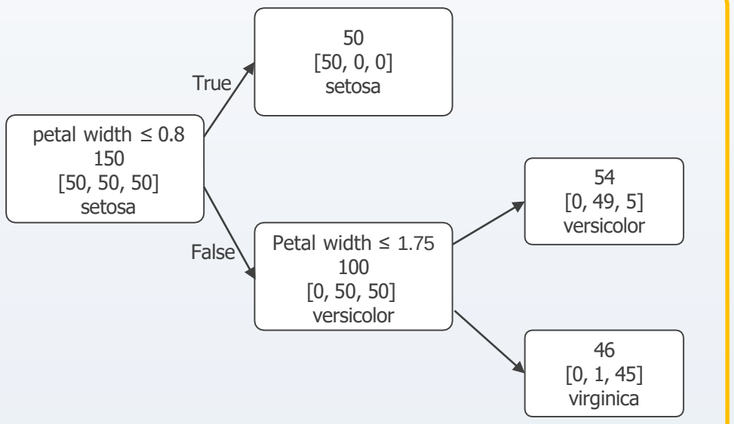
SVM



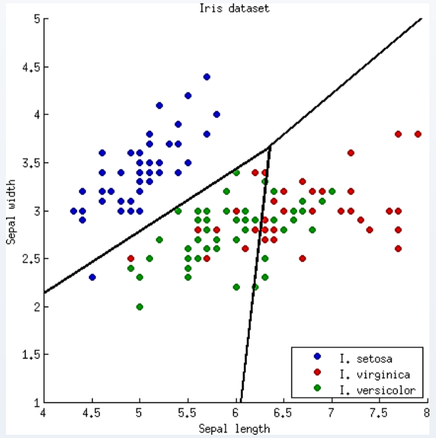
Equal weights are assigned to different models

$$P = \frac{p1 + p2 + p3}{3}$$

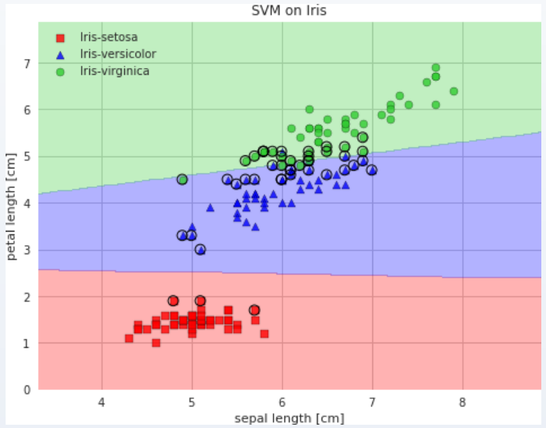
Weighted Averaging



Decision Tree



Logistic Regression



SVM

50

35

15

$w1 + w2 + w3 = 1$

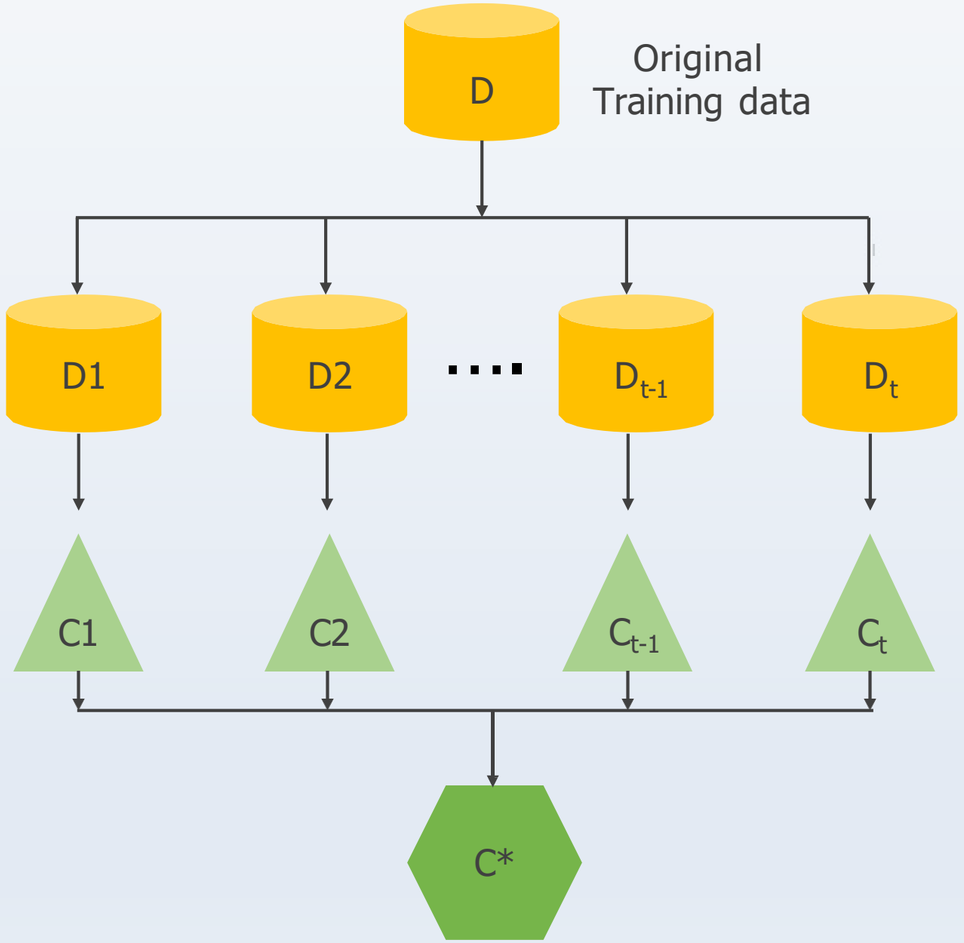
Each model is assigned a different weight

$P = w1 * p1 + w2 * p2 + w3 * p3$

Dr.Prasanalakshmi

Bagging

Bagging or bootstrap aggregation **reduces variance** of an estimate by taking mean of multiple estimates



STEP 1 Create randomly sampled datasets of the original training data (bootstrapping)

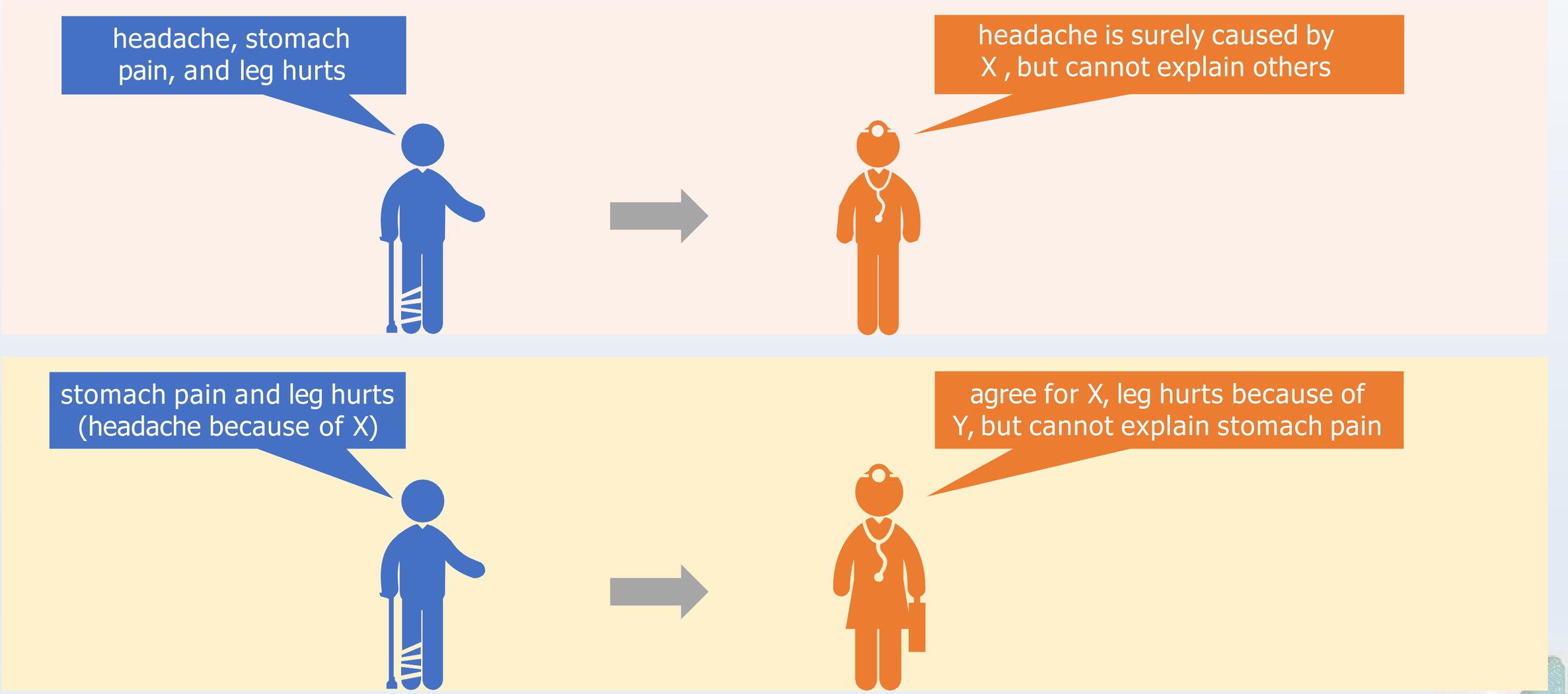
STEP 2 Build and fit several classifiers to each of these diverse copies

STEP 3 Take the average of all the predictions to make final overall prediction

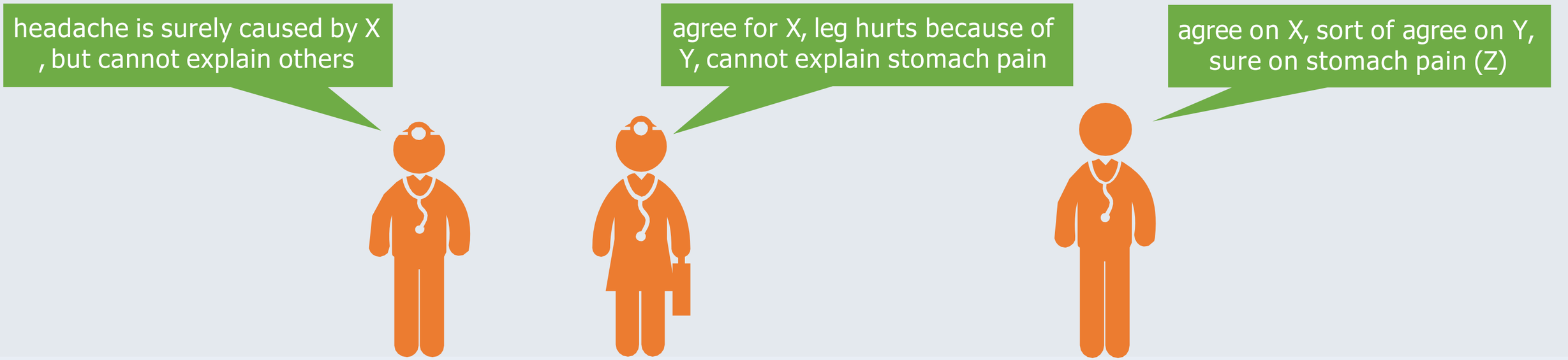
$$f(x) = 1/M \sum_{m=1}^{M} f_m(x)$$

Boosting

Boosting **reduces bias** by training weak learners sequentially, each trying to correct its predecessor



Boosting (Contd.)



Final Diagnosis

Result of the weighted opinions that the doctors (sequentially)

MACHINE LEARNING

Boosting Algorithm

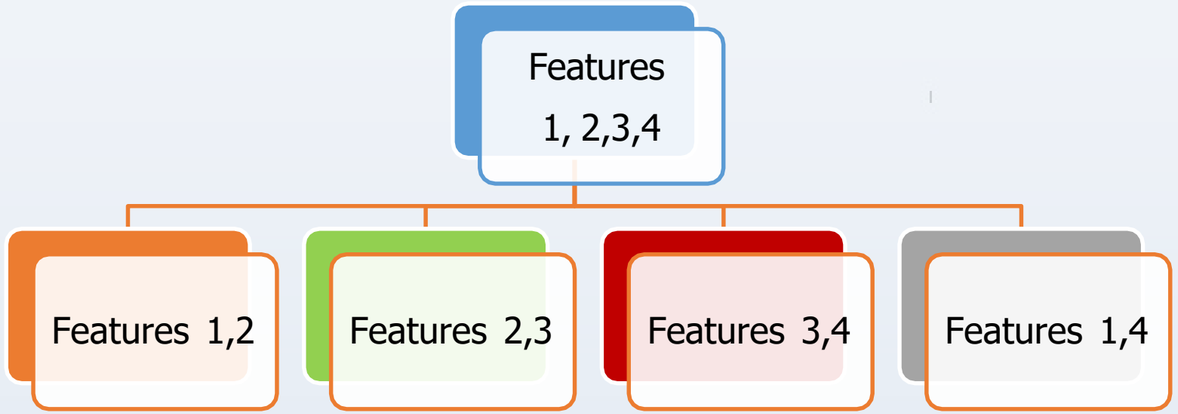
STEP: 01	STEP: 02	STEP: 03
Train a classifier H1 that best classifies the data with respect to accuracy	Identify the region where H1 produces errors, add weights to it and produce a H2 classifier	Exaggerate those samples for which H1 gives a different result from H2 and produces H3 classifier Repeat step 02 for a new classifier

Ensemble Learning

Topic 2: Algorithms

Random Forests

Random forests are utilized to produce decorrelated decision trees



RF's create random subsets of the *features*

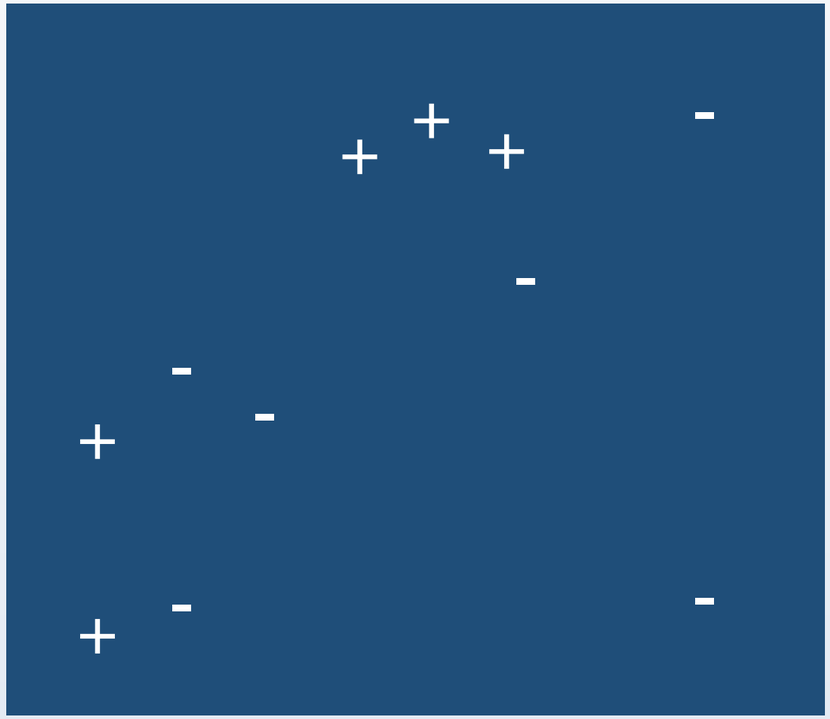
Smaller trees are built using these subsets creating tree diversity

To overcome overfitting, diverse sets of decision trees are required

Adaboost

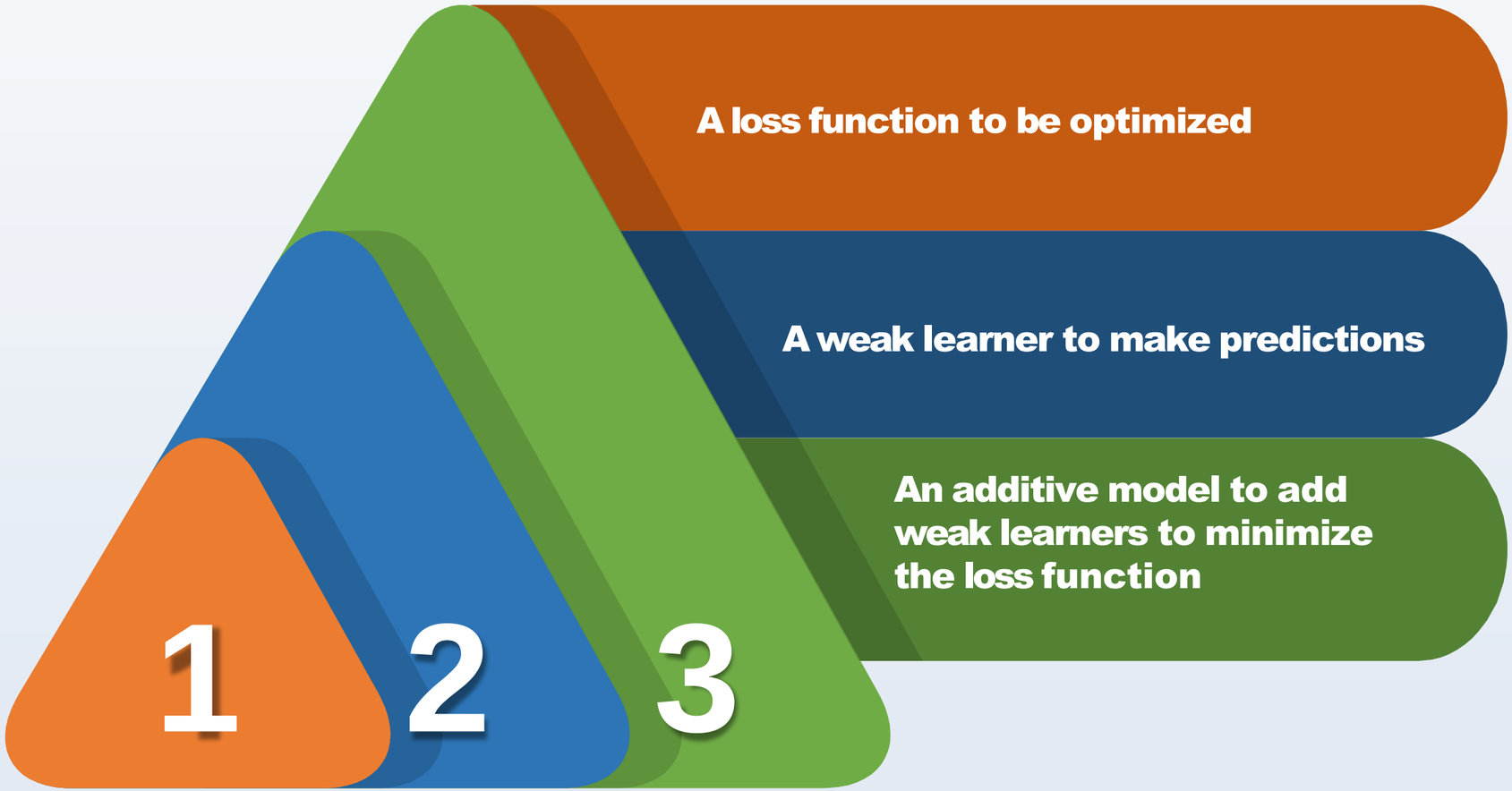
Consider a scenario, where there are '+' and '–'

Objective : Classify '+' and '–'



Gradient Boosting (GBM)

Gradient boosting involves three elements:



GBM minimizes the loss function (MSE) of a model by adding weak learners using a gradient descent procedure.

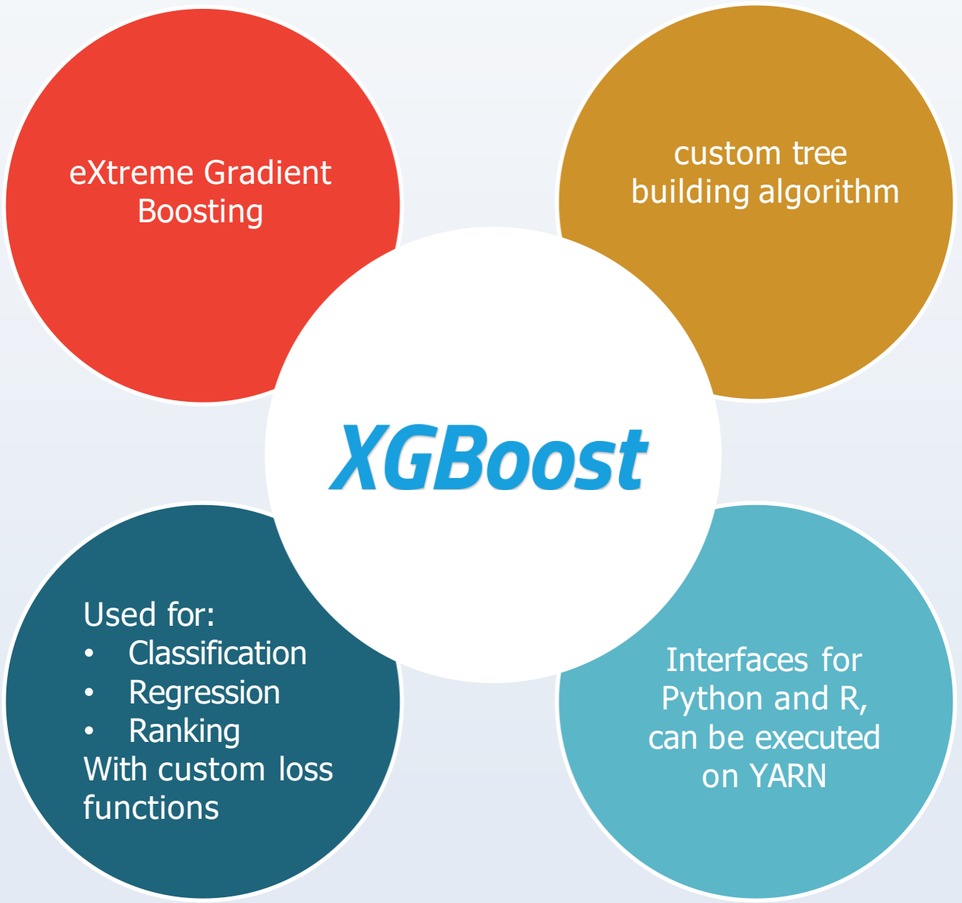
MACHINE LEARNING

GBM Algorithm

Step 01	Fit a simple regression or classification model
Step 02	Calculate error residuals (actual value - predicted value)
Step 03	Fit a new model on error residuals as target variable with same input variables
Step 04	Add the predicted residuals to the previous predictions
Step 05	Fit another model on residuals that are remaining and repeat steps 2 and 5 until model is overfit or the sum of residuals becomes constant

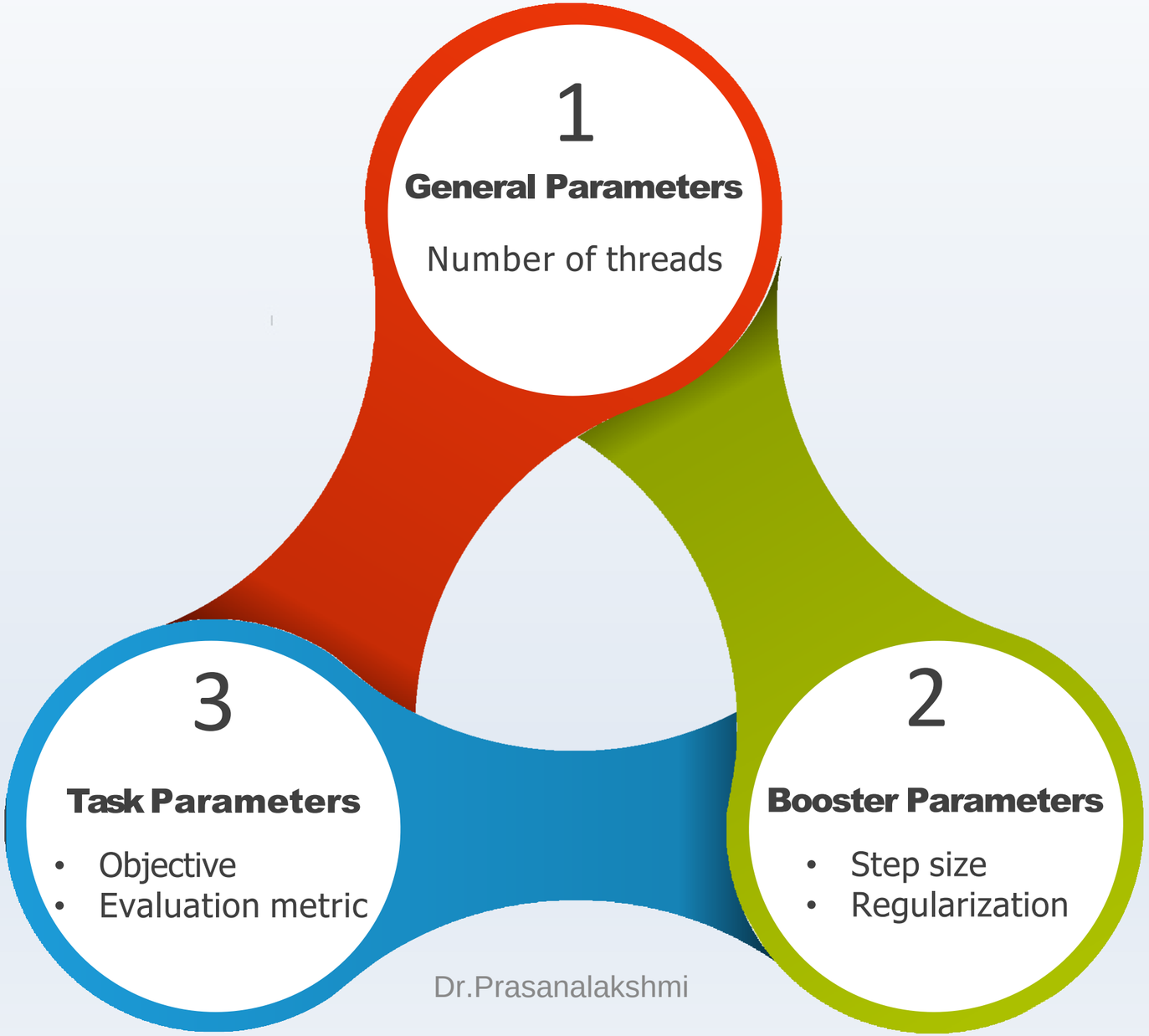
XGBoost

e**X**treme **G**radient **B**oosting is a library for developing fast and high-performance gradient boosting tree models.



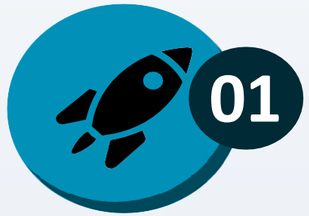
XGBoost is extensively used in ML competitions as it is almost 10 times faster than other gradient boosting techniques

XGBoost Parameters



XGBoost Library Features

XGBoost library features tools are built for the sole purpose of model performance and computational speed.



SYSTEM

Parallelization
Tree construction using all CPU cores while training

Distributed Computing
Training very large models using a cluster of machines

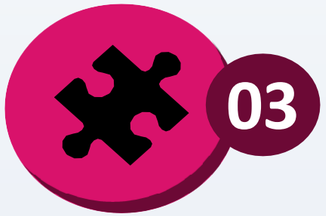
Cache Optimization
Data structures make best use of hardware

ALGORITHM

Sparse Aware
Automatic handling of missing data values

Block Structure
Supports the parallelization of tree construction

Continued Training
To boost an already fitted model on new data



MODELS

Gradient Boosting
Gradient boosting machine algorithm including learning rate

Stochastic Gradient Boosting
Sub-sampling at the row, column and column per split levels

Regularized Gradient Boosting
With both L1 and L2 regularization

Dr.Prasanalakshmi

Ensemble Learning

Topic 3: Model Selection

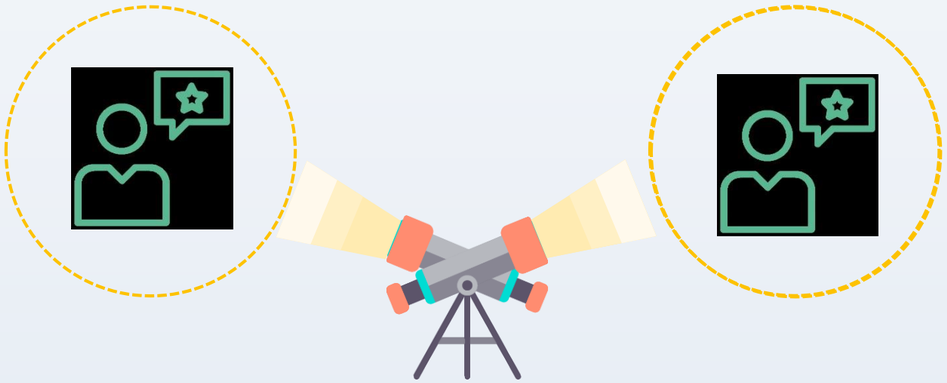
Model Evaluation

Models can be evaluated based on their measure of performance

Assessing Model Performance

Train/Test Split

- Divide the training dataset
- Train on the first training set
- Test on the second set

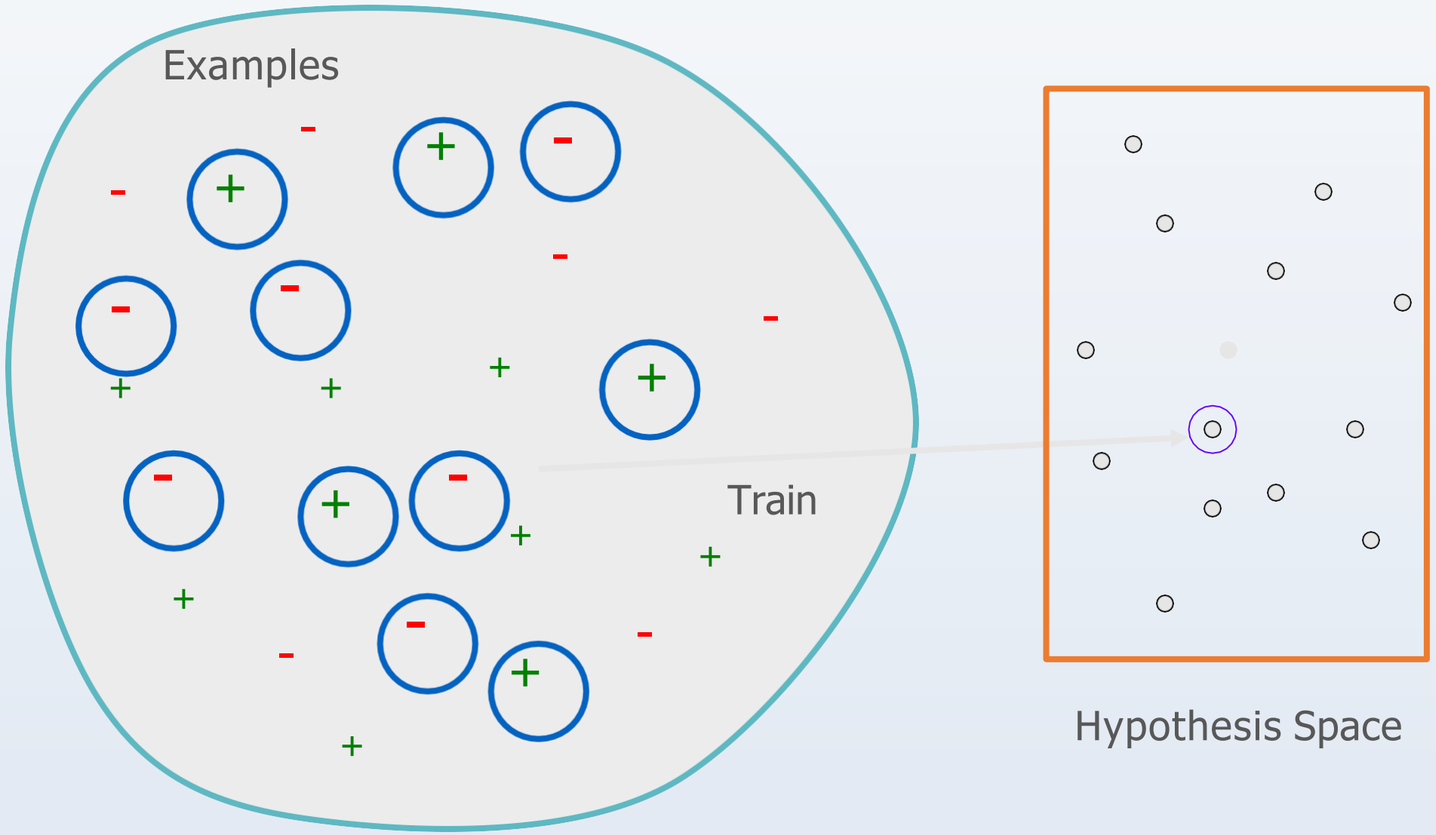


Cross Validation Split

- Sets of train/test splits created
- Accuracy for each set is checked
- Results are averaged

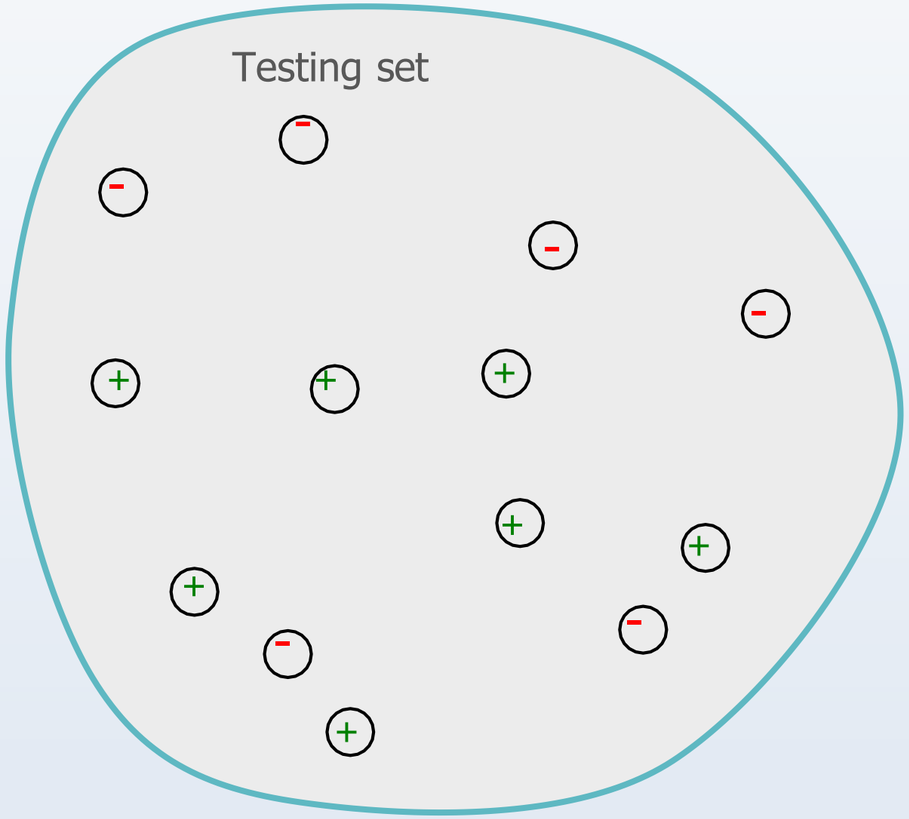
Techniques to assess model performance

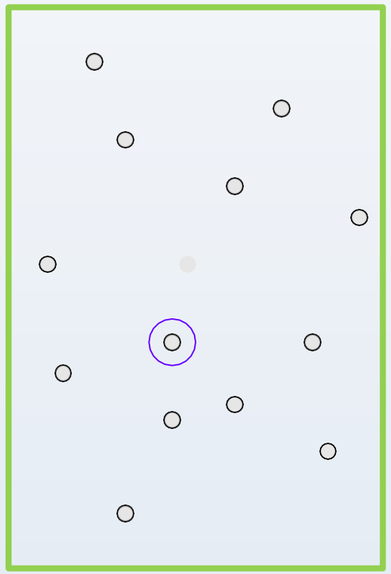
Train/Test Split



Creating the training dataset

Train/Test Split (Contd.)

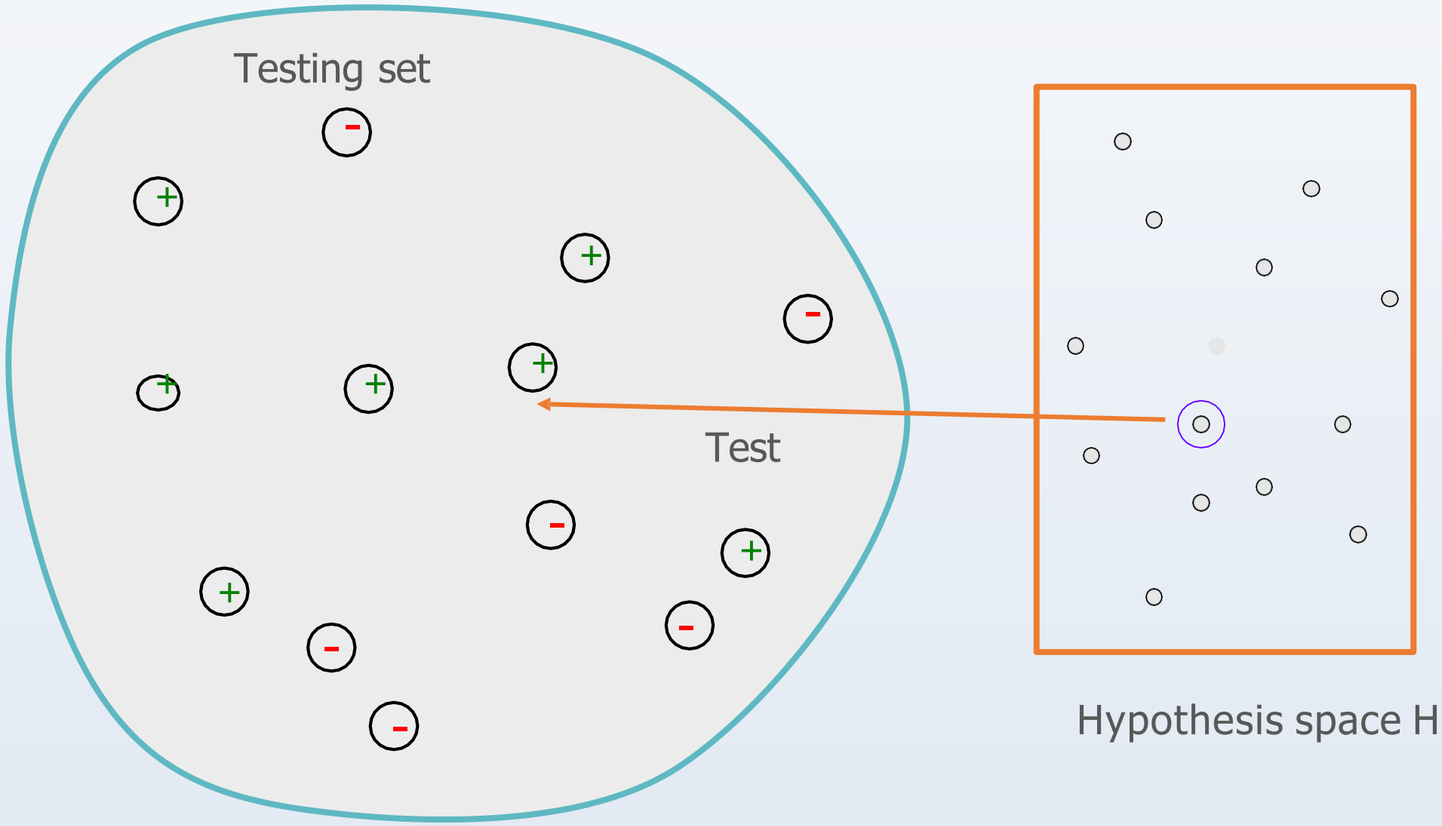




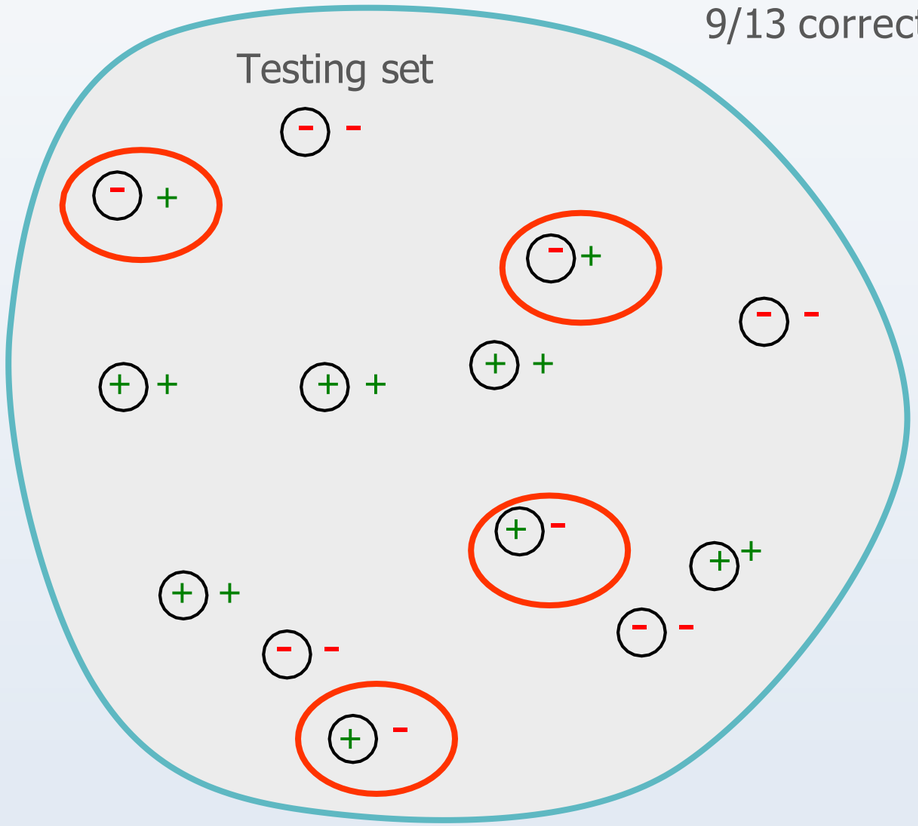
Hypothesis space H

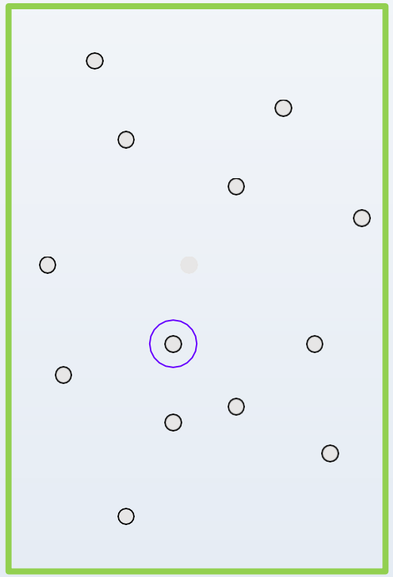
Creating the testing dataset

Train/Test Split (Contd.)



Train/Test Split (Contd.)





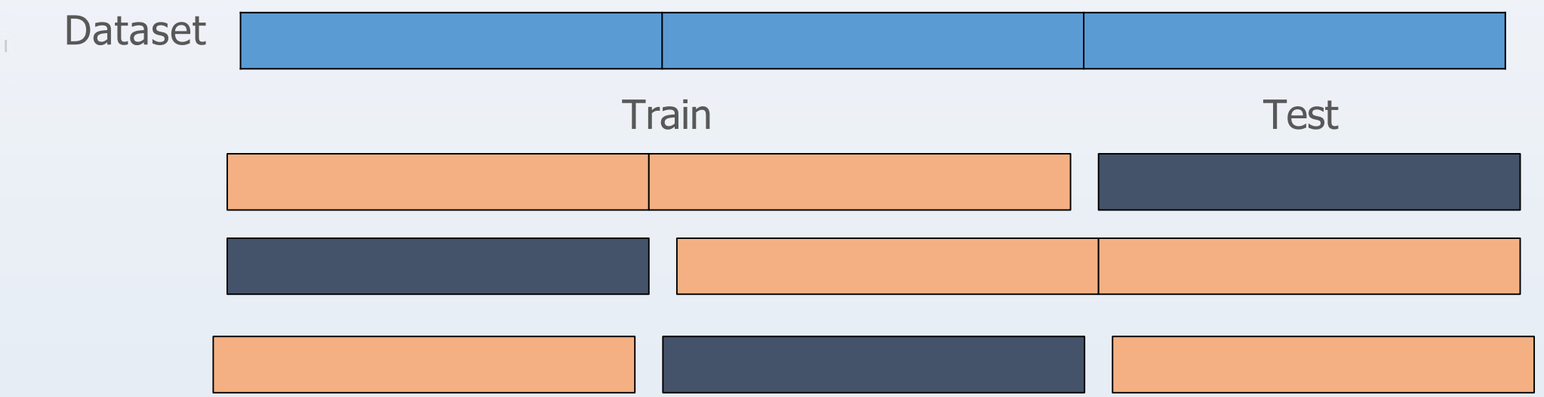
Hypothesis space H

Verifying the results

Common Splitting Strategies

K-Fold Cross-Validation

Leave-one-out



Identifying the Train/Test split ratio

MACHINE LEARNING

Common Splitting Strategies (Contd.)

K-Fold Cross-Validation

Leave-one-out

Train/Test Split vs. Cross-Validation

Cross-validation

- More accurate estimate of out-of-sample accuracy
- More efficient use of data(every observation is used for both training and testing)

Train/Test Split

- Runs K-times faster than K-fold cross-validation
- Simpler to examine the detailed results of testing process

Thank You