# BITS Pilani presentation


BITS Pilani
Pilani Campus

Dr. Vivek V. Jog
Dept. of Computer Engineering

innovate achieve lead

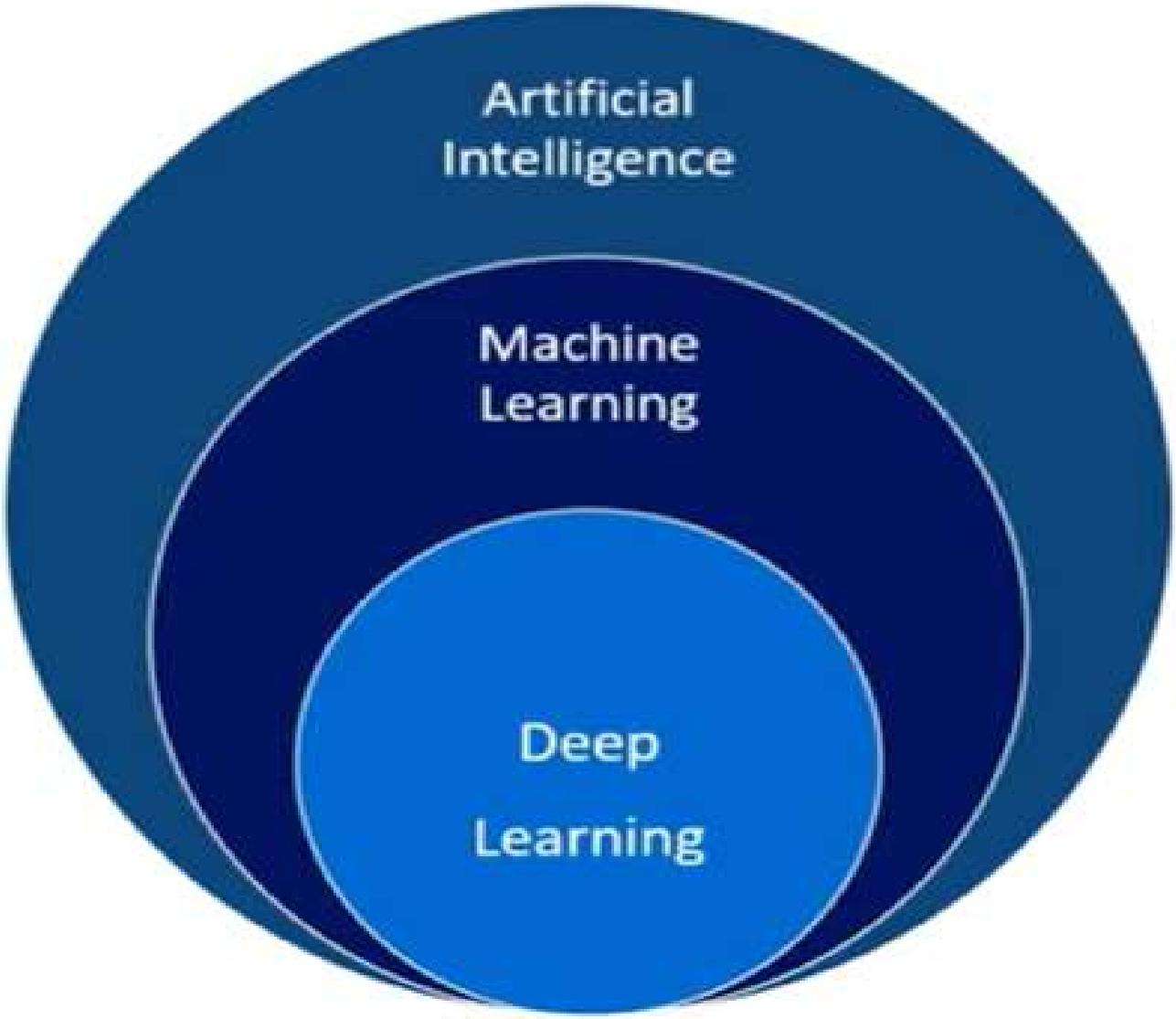
BITS Pilani

Pilani Campus

# Big Data Systems (S1-24_CCZG522)

# **Lecture No.14**

- Intuition of AI vs ML vs DL
- Artificial Intelligence
- Machine Learning
- Deep Learning
- ML vs DL
- Data Science


Artificial
Intelligence
Machine
Learning
Deep
Learning

1.Applied AI(weak AI)- perform some specific tasks.

Alexa

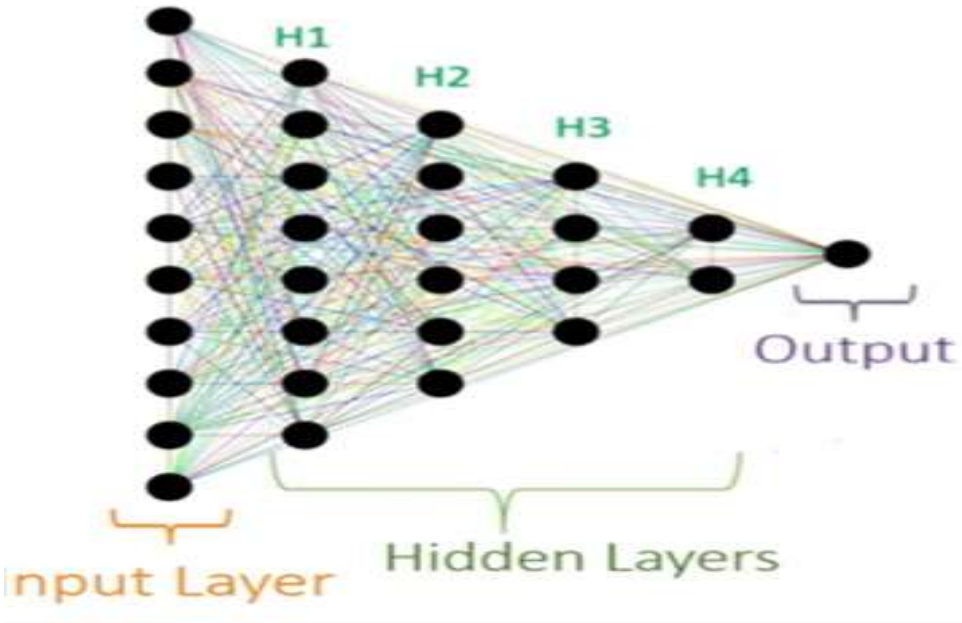
Google Assistant

2.Generalized AI(strong AI)- acts like humans.

- Machine Learning is a subset of AI.
- Machine Learning is a set of algorithms that train on a data set to make predictions or take actions in order to optimize some systems.

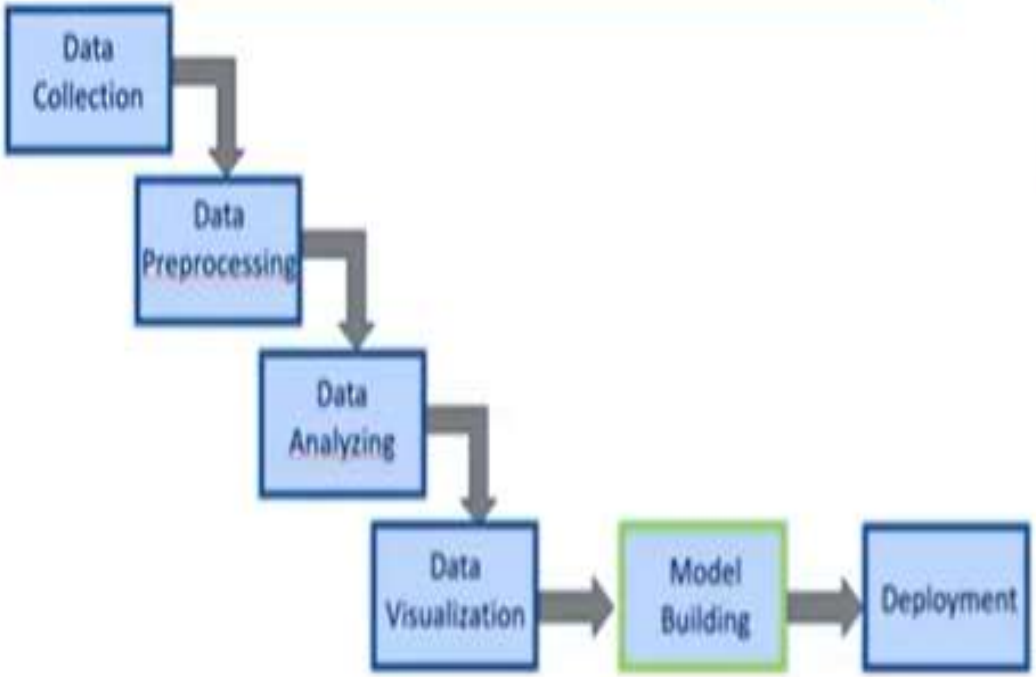
- Deep Learning is a subset of Machine Learning Where learning method is based on data representation or feature learning.
- "Deep" refers to 1 or more hidden layers in this case.

- In Deep Learning data goes through multiple numbers of non-linear transformation obtain an output.

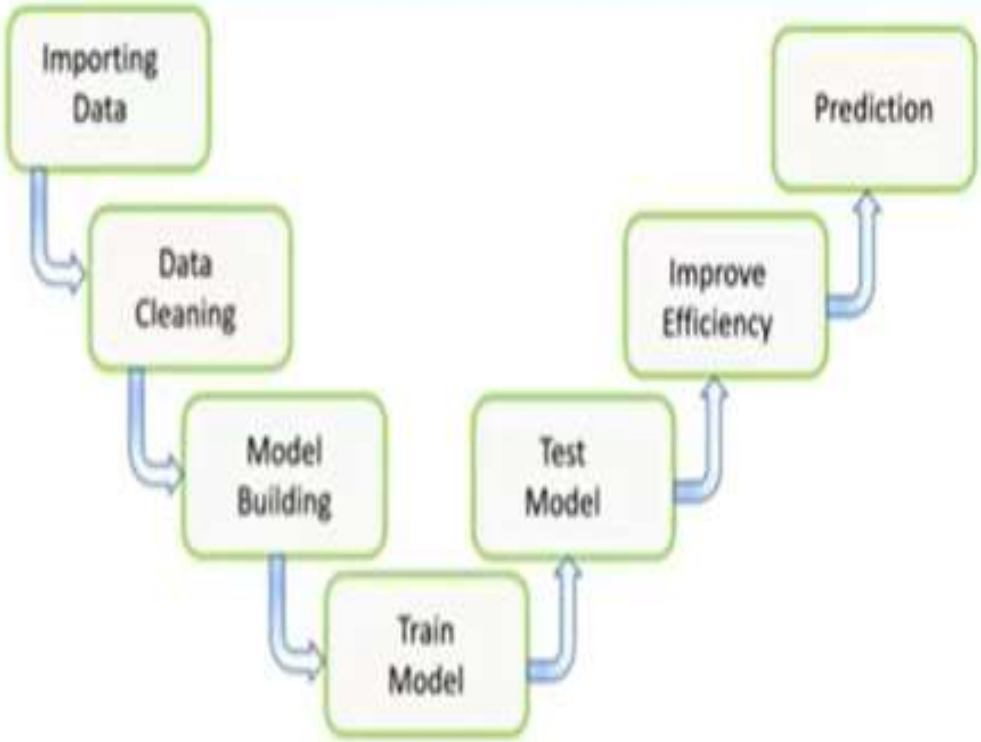


| | Machine | Deep |
|---|---|---|
| 1. Data dependencies | Small Size data | Large Size data |
| 2. Hardware dependencies | Regular Machine | Server Class |
| 3. Feature Engineering | Work with Data | Work with feature |
| 4. Execution Time | Normal Time | Long Time |

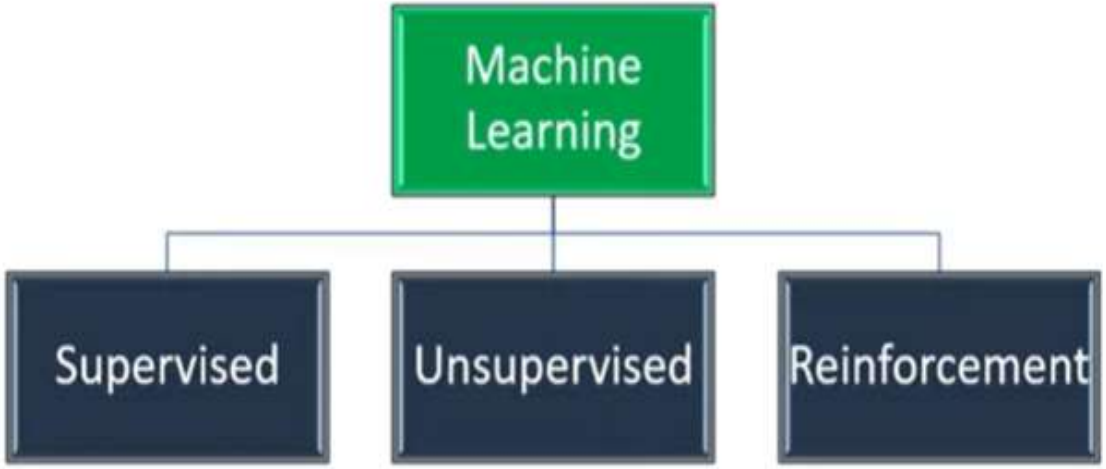
## Data Science Flow Chart



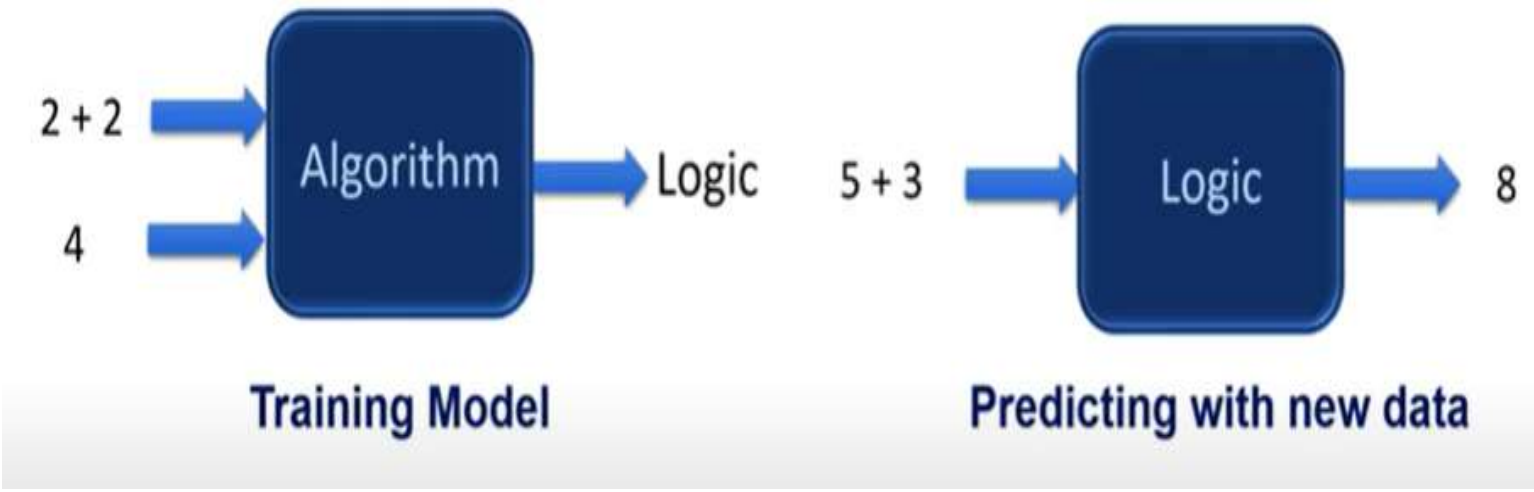

## Model Building



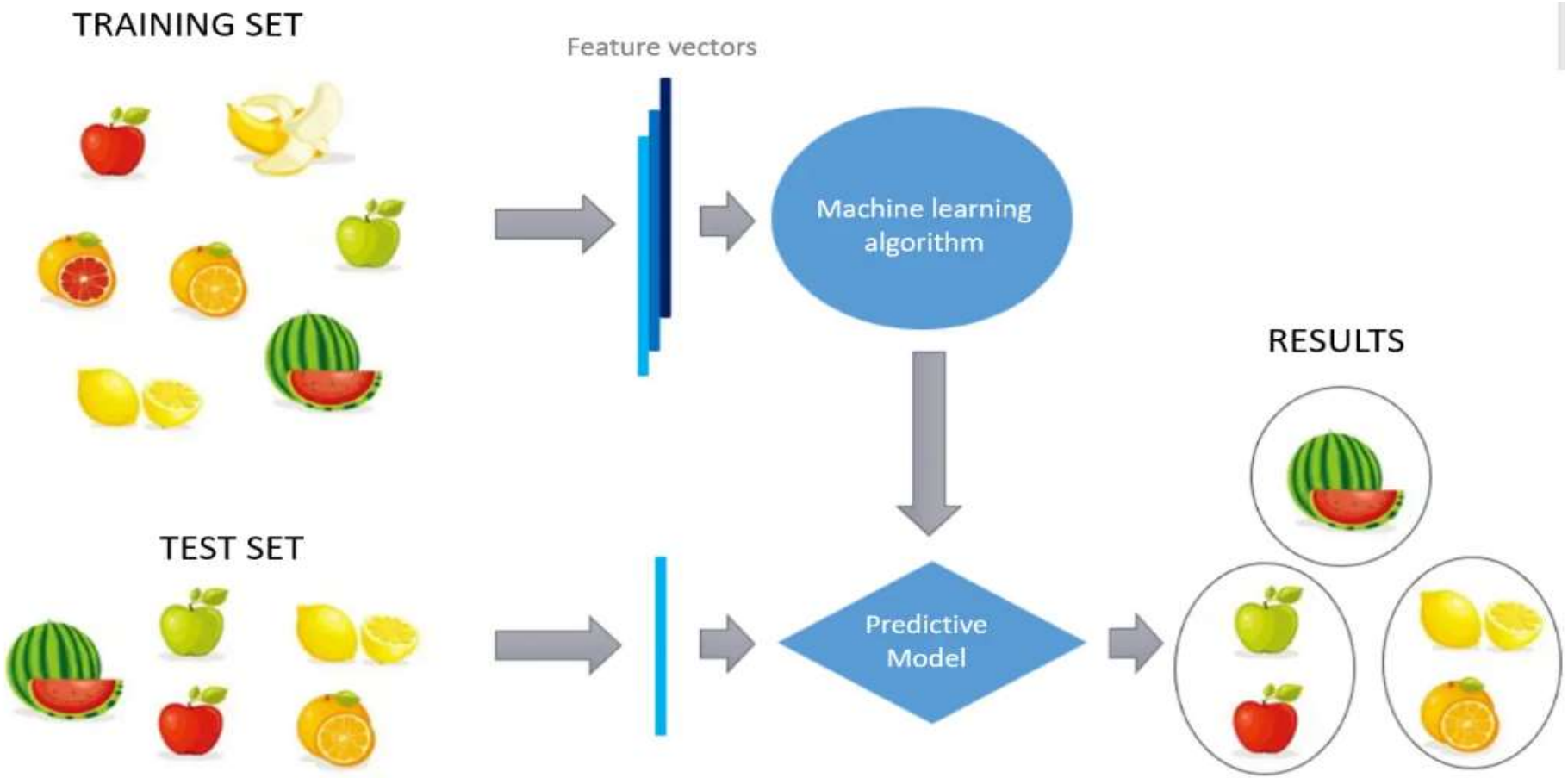
# Types of Machine Learning in Artificial Intelligence



# Supervised Learning



# Unsupervised Learning



# Supervised Learning

1. Regression
   -Continuous Value


2. Classification
   -Categorical Value



1. Regression
   - -Linear Regression
   - -Multiple Linear Regression
   - -Polynomial Regressoin
   - -Support Vector Regression
   - -Decision Tree Regression
   - -Random Forest Regression

2. Classification
   - -Logistic Regression
   - -K-Nearest Neighbors (KNN)
   - -Support Vector Machine (SVM)
   - -Naïve Bayes
   - -Decision Tree Classification
   - -Random Forest Classification

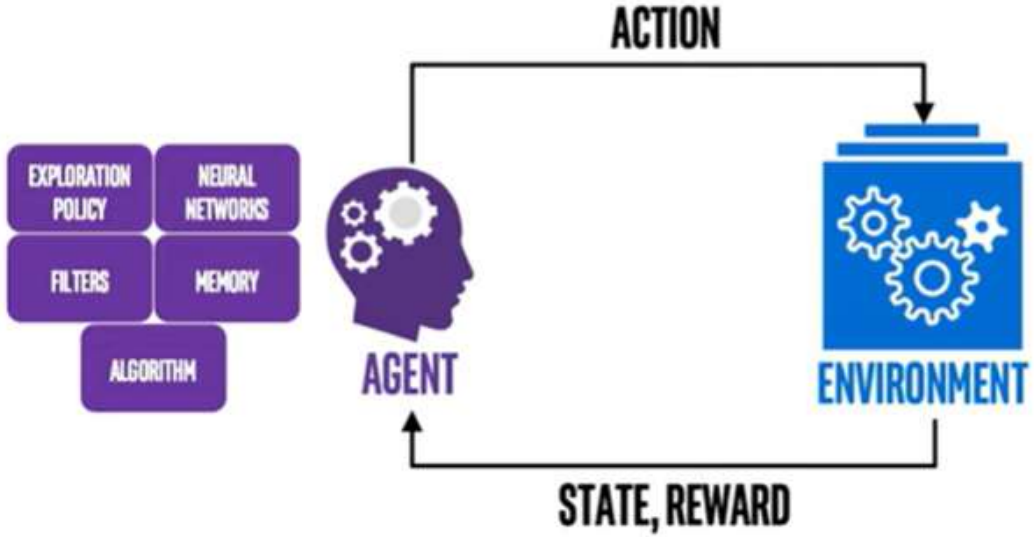
# Unsupervised Learning

1. Clustering
   - -K-Means Clustering
   - -Hierarchical Clustering

2. Association
   - -Apriori
   - -Eclat

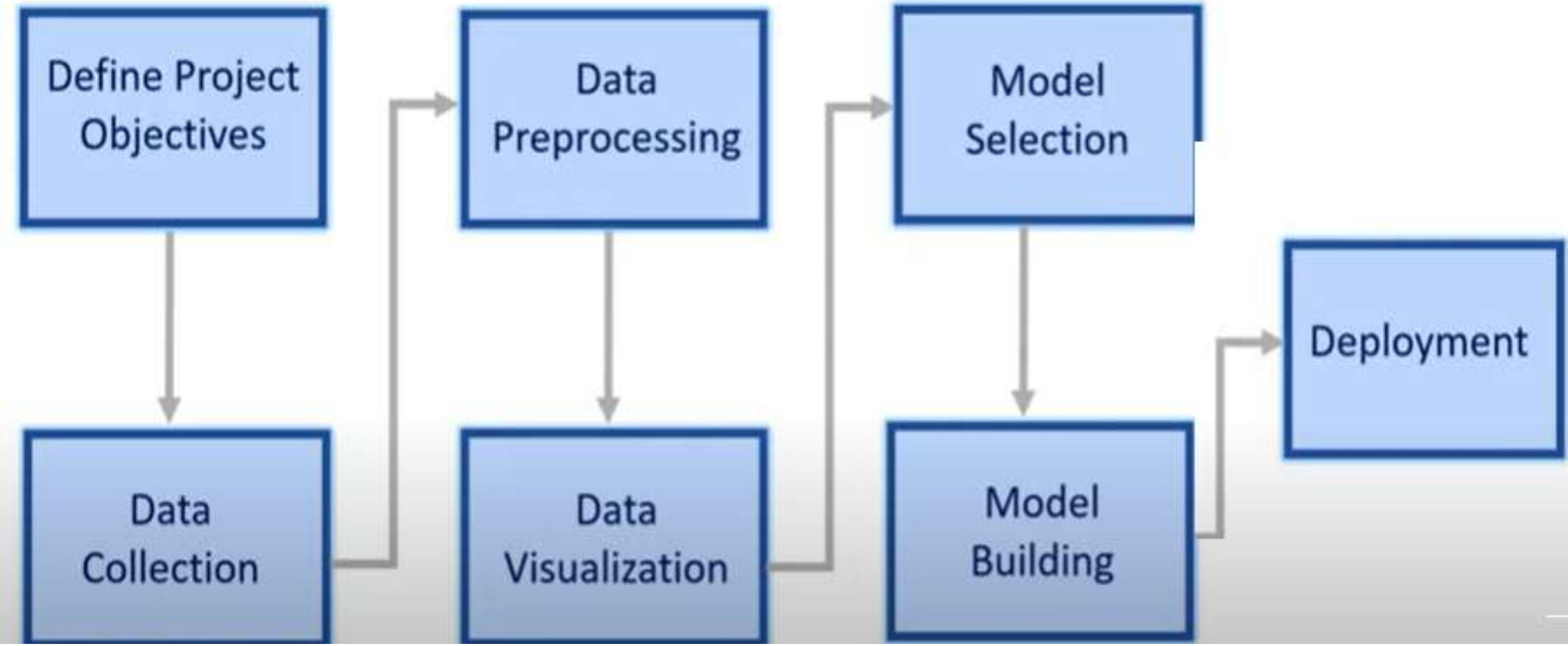
# Reinforcement Learning

Reinforcement learning is a type of machine learning where an agent learns to behave in a environment by performing actions and seeing the results.



# Reinforcement Learning

# Machine Learning Life Cycle



# Data Collection

- Primary Data – Collected by researcher from first-hand source.
- Secondary Data – Collected by someone else and already been passed through the statistical process.

# Data Preprocessing

1. Data Cleaning :
   - -Filling Missing Data
   - -Smoothing Noisy Data
2. Data Transformation
   - -Normalization
3. Dimensionality Reduction

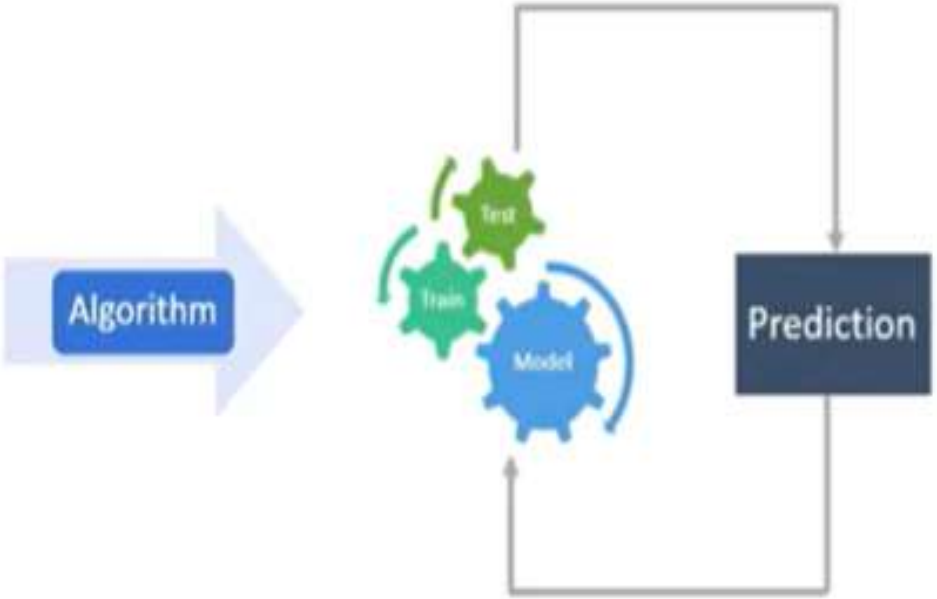
# Data Visualization

Graphical representation of data.

## Model Selection

- Linear Regression
- Logistic Regression
- Random Forest
- K-Means clustering

## Model Building




## Model Deployment

# TOOLS

*Here's a list of the top Python libraries for statistical analysis:*

1. *NumPy*
2. *SciPy*
3. *Pandas*
4. *StatsModels*


NumPy
NumPy
Scientific computations
Multi-dimensional array objects
Data manipulation
Statistics for ML algorithms


Mathematical computations
Provides a collection of sub-packages
Advanced linear algebra functions
Support for signal processing
SciPy
SciPy

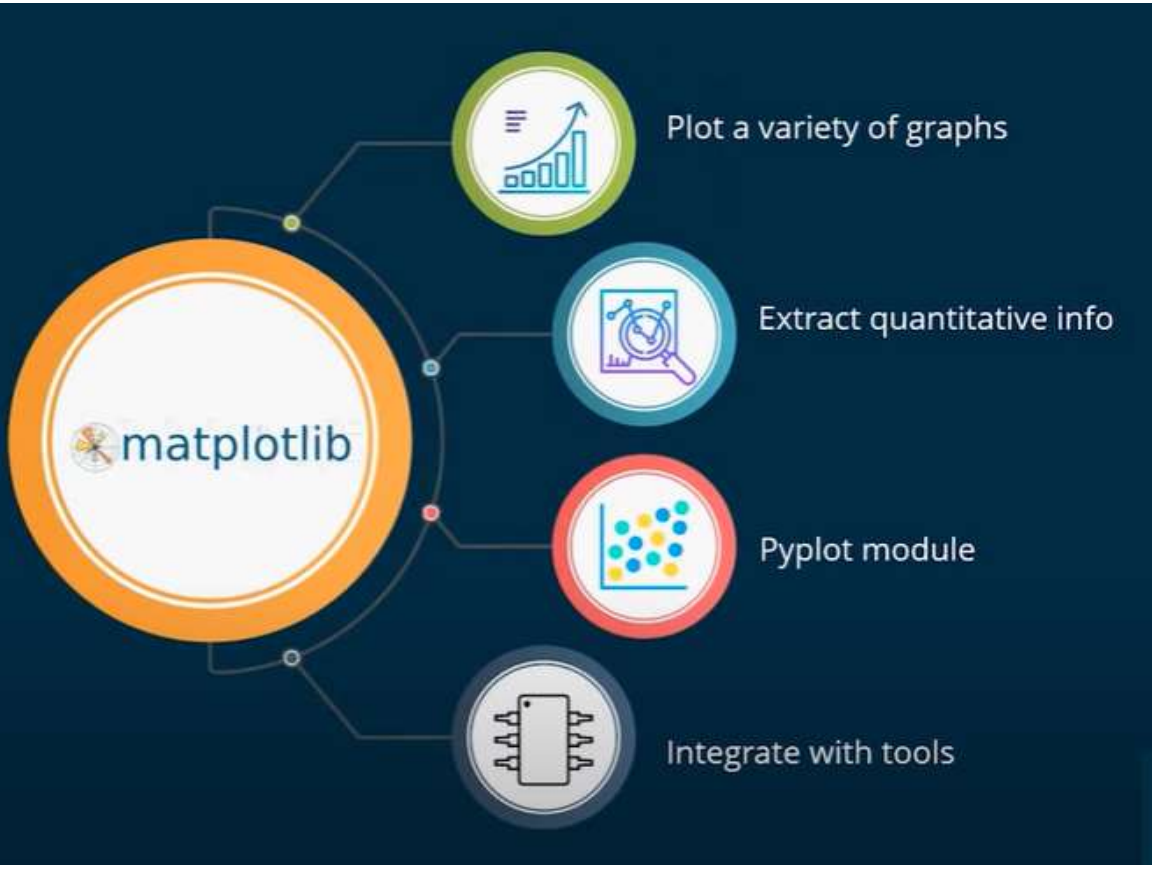

Pandas
Pandas
Data frame objects
Process large data sets
Complex data analysis
Time Series data


Statistical tests
Support for R-style formulas
Implement Generalised Linear Models
Hypothesis testing
StatsModels
StatsModels

*Here's a list of the top Python libraries for data visualization:*

1. *Matplotlib*
2. *Seaborn*
3. *Plotly*
4. *Bokeh*

matplotlib


matplotlib
Plot a variety of graphs
Extract quantitative info
Pyplot module
Integrate with tools


Compatible with various data formats
Support for automated statistical estimation
High-level abstractions
Support for built-in themes
Seaborn
Seaborn


plotly
plotly
Supports a collection of chart types
Build public/ private dashboards
Access graphs on variety of platforms
In-built API for Plotly Grid


Create complex statistical graphs
Supports multiple language bindings
julia
Integration with Flask and django
python
django
Transform visualization for other libraries
Bokeh
Bokeh

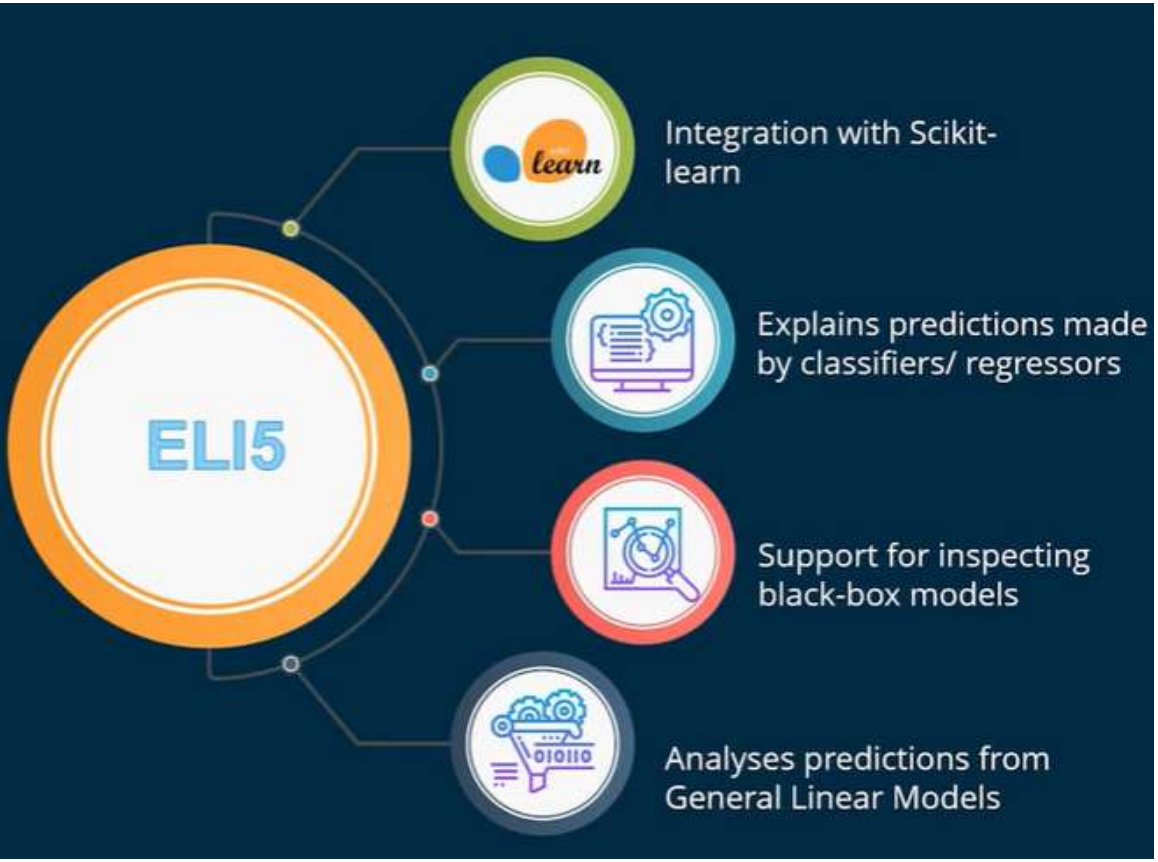
*Here's a list of the top Python libraries for Machine Learning:*

1. *Scikit-learn*
2. *XGBoost*
3. *Eli5*


scikit
learn
scikit
learn
Provides a set of standard datasets
Machine Learning algorithms
In-built functions for feature extraction & selection
Model evaluation

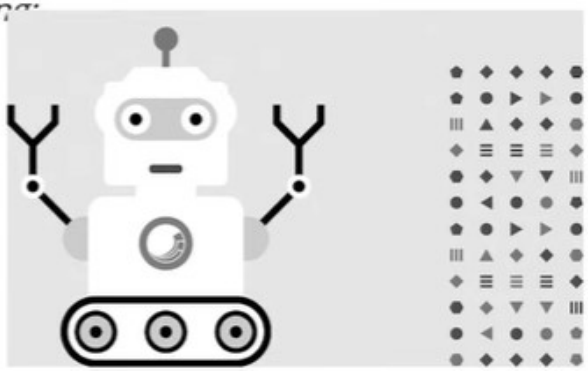

Fast data processing
Supports parallel computation
Provides internal parameters for evaluation
Higher accuracy
XGBoost
XGBoost

ELI5


ELI5
Integration with Scikit-learn
Explains predictions made by classifiers/ regressors
Support for inspecting black-box models
Analyses predictions from General Linear Models

*Here's a list of the top Python libraries for Deep Learning:*

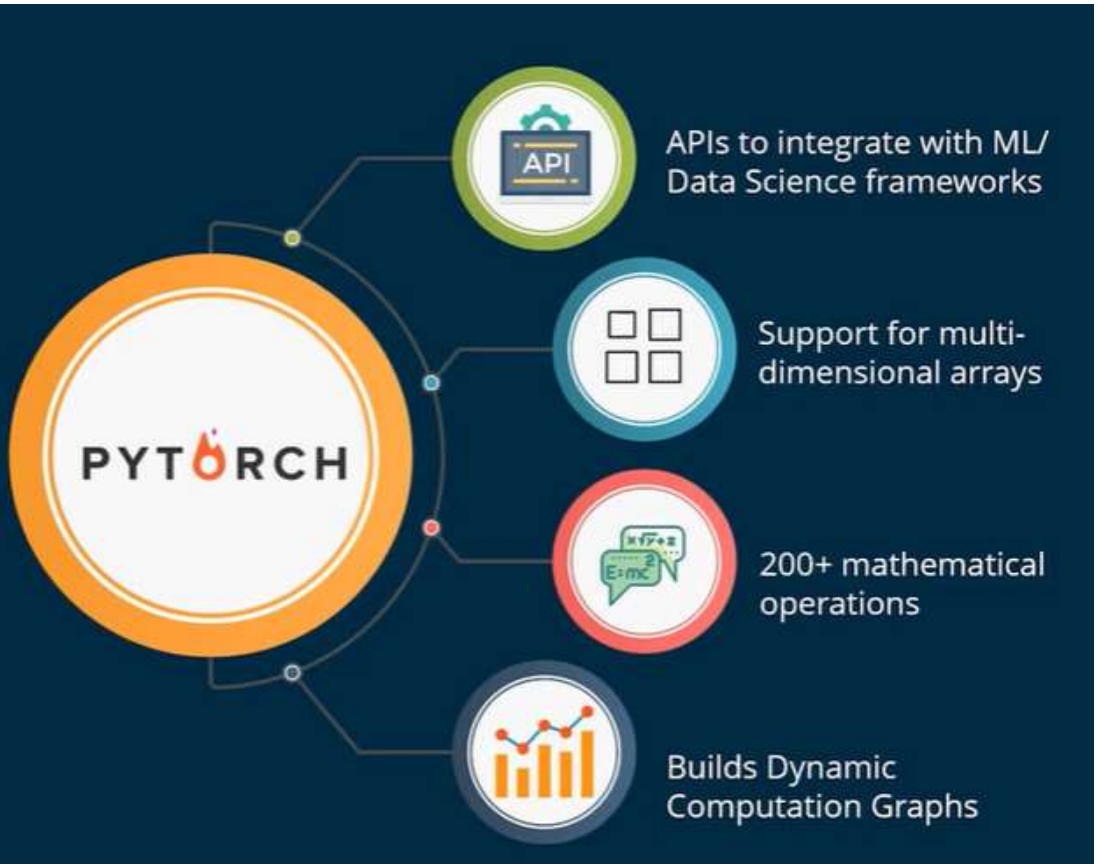
1. *TensorFlow*
2. *Keras*
3. *Pytorch*




TensorFlow
TensorFlow
Build and train multiple neural networks
Statistical analysis
In-built functions to improve the accuracy
Comes with an in-built Visualizer


Support various types of Neural Networks
Advanced neural network computations
Provides pre-processed dataset
Easily extensible
Keras
Keras

PYTORCH


PYTORCH
API
APIs to integrate with ML/ Data Science frameworks
Support for multi-dimensional arrays
200+ mathematical operations
Builds Dynamic Computation Graphs

*Here's a list of the top Python libraries for NLP:*

1. *NLTK*
2. *SpaCy*
3. *Gensim*


Natural Language Analysis with Python NLTK
Natural Language Analysis with Python NLTK
Provides a suite of data and text processing methods
Contains wrappers for industrial-level NLP libraries
Comprehensive doc & API documentation guide
Huge community of users


Linguistic computation
Analyse grammatical structure
Methods for Text processing
Provides support for 51+ languages
spaCy
spaCy

gensim


gensim
Precisely classify documents
Text processing algorithms
Provides I/O wrappers
Intuitive GUI

# Supervised Algo -- K-Nearest Neighbour


| Sepal Length | Sepal Width | Species |
|---|---|---|
| 5.3 | 3.7 | Setosa |
| 5.1 | 3.8 | Setosa |
| 7.2 | 3.0 | Virginica |
| 5.4 | 3.4 | Setosa |
| 5.1 | 3.3 | Setosa |
| 5.4 | 3.9 | Setosa |
| 7.4 | 2.8 | Virginica |
| 6.1 | 2.8 | Verscicolor |
| 7.3 | 2.9 | Virginica |
| 6.0 | 2.7 | Verscicolor |
| 5.8 | 2.8 | Virginica |
| 6.3 | 2.3 | Verscicolor |
| 5.1 | 2.5 | Verscicolor |
| 6.3 | 2.5 | Verscicolor |
| 5.5 | 2.4 | Verscicolor |

| Sepal Length | Sepal Width | Species |
|---|---|---|
| 5.2 | 3.1 | ? |

## Step 1: Find Distance

$$\text{Distance (Sepal Length, Sepal Width)} = \sqrt{(x-a)^2 + (y-b)^2}$$

$$\text{Distance (Sepal Length, Sepal Width)} = \sqrt{(5.2-5.3)^2 + (3.1-3.7)^2}$$

$$\text{Distance (Sepal Length, Sepal Width)} = 0.608$$

| Sepal Length | Sepal Width | Species | Distance |
|---|---|---|---|
| 5.3 | 3.7 | Setosa | 0.608 |

# K-Nearest Neighbour


| Sepal Length | Sepal Width | Species | Distance | Rank |
|---|---|---|---|---|
| 5.3 | 3.7 | Setosa | 0.608 | 3 |
| 5.1 | 3.8 | Setosa | 0.707 | 6 |
| 7.2 | 3.0 | Virginica | 2.002 | 13 |
| 5.4 | 3.4 | Setosa | 0.36 | 2 |
| 5.1 | 3.3 | Setosa | 0.22 | 1 |
| 5.4 | 3.9 | Setosa | 0.82 | 8 |
| 7.4 | 2.8 | Virginica | 2.22 | 15 |
| 6.1 | 2.8 | Verscicolor | 0.94 | 10 |
| 7.3 | 2.9 | Virginica | 2.1 | 14 |
| 6.0 | 2.7 | Verscicolor | 0.89 | 9 |
| 5.8 | 2.8 | Virginica | 0.67 | 5 |
| 6.3 | 2.3 | Verscicolor | 1.36 | 12 |
| 5.1 | 2.5 | Verscicolor | 0.60 | 4 |
| 6.3 | 2.5 | Verscicolor | 1.25 | 11 |
| 5.5 | 2.4 | Verscicolor | 0.75 | 7 |

**Step 2: Find Rank**

# K-Nearest Neighbour


| Sepal Length | Sepal Width | Species | Distance | Rank |
|---|---|---|---|---|
| 5.3 | 3.7 | Setosa | 0.608 | 3 |
| 5.1 | 3.8 | Setosa | 0.707 | 6 |
| 7.2 | 3.0 | Virginica | 2.002 | 13 |
| 5.4 | 3.4 | Setosa | 0.36 | 2 |
| 5.1 | 3.3 | Setosa | 0.22 | 1 |
| 5.4 | 3.9 | Setosa | 0.82 | 8 |
| 7.4 | 2.8 | Virginica | 2.22 | 15 |
| 6.1 | 2.8 | Verscicolor | 0.94 | 10 |
| 7.3 | 2.9 | Virginica | 2.1 | 14 |
| 6.0 | 2.7 | Verscicolor | 0.89 | 9 |
| 5.8 | 2.8 | Virginica | 0.67 | 5 |
| 6.3 | 2.3 | Verscicolor | 1.36 | 12 |
| 5.1 | 2.5 | Verscicolor | 0.60 | 4 |
| 6.3 | 2.5 | Verscicolor | 1.25 | 11 |
| 5.5 | 2.4 | Verscicolor | 0.75 | 7 |

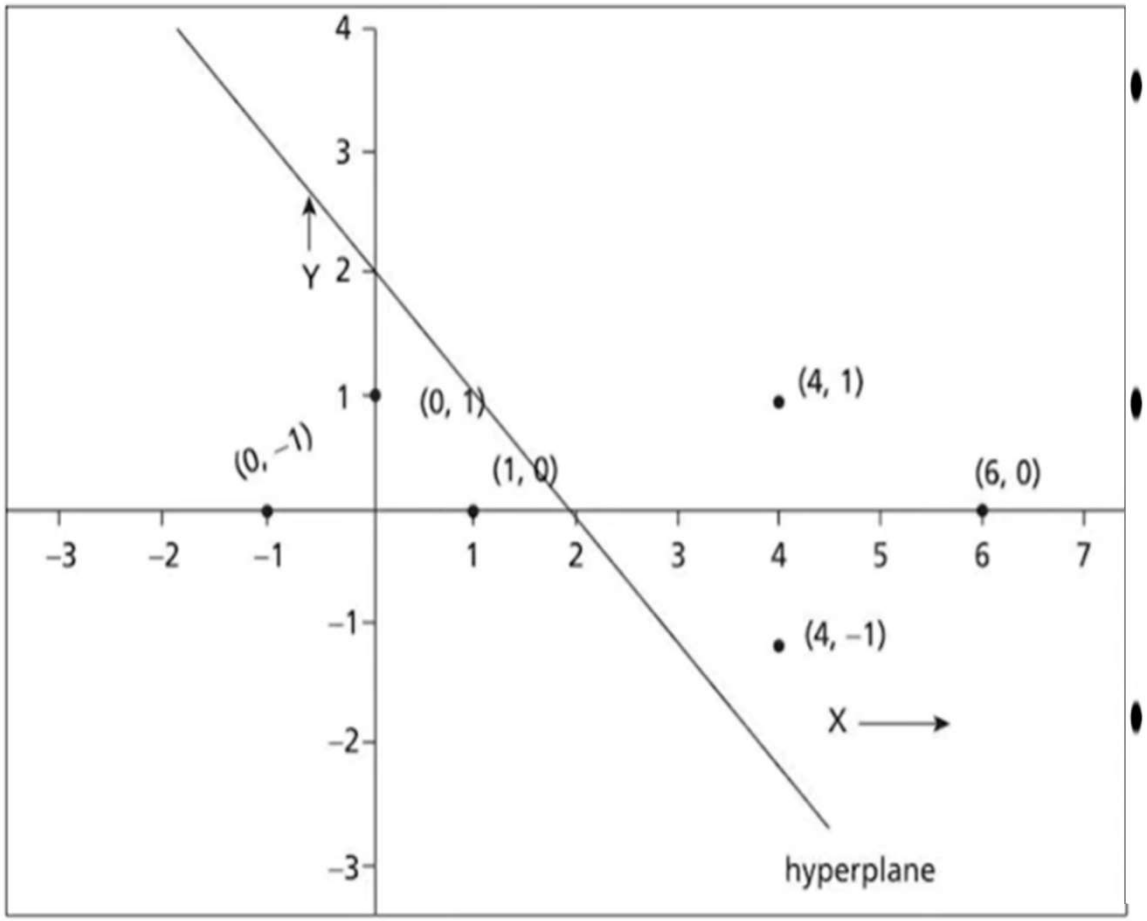
**Step 3: Find the Nearest Neighbor**

**If k = 1 – Setosa**

**If k = 2 – Setosa**

**If k = 5 – Setosa**

# Support Vector Machine

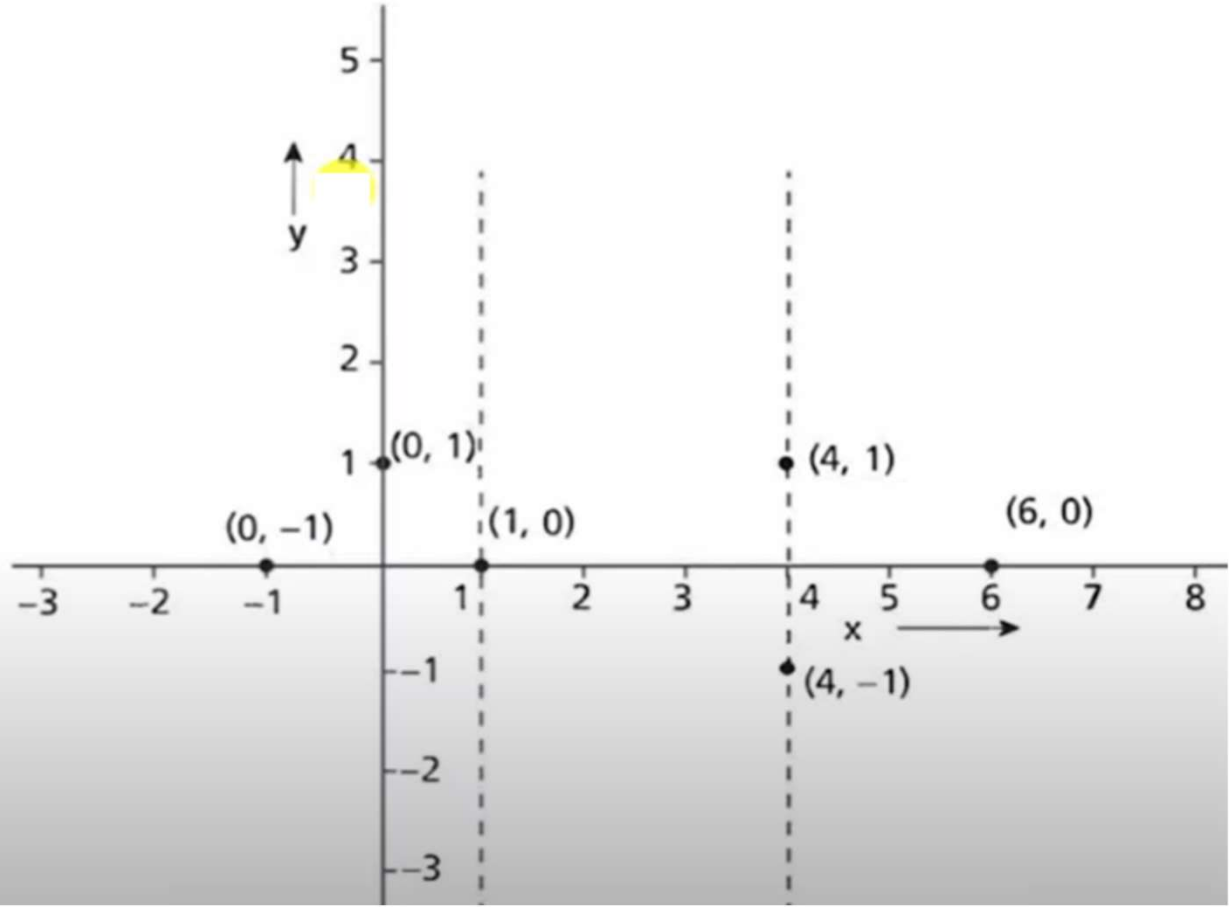



- Points (4, 1), (4, -1) and (6, 0) belong to class positive and
- points (1, 0), (0, 1) and (0, -1) belong to negative class.
- Draw an optimal hyperplane to classify the points.

# Support Vector Machine



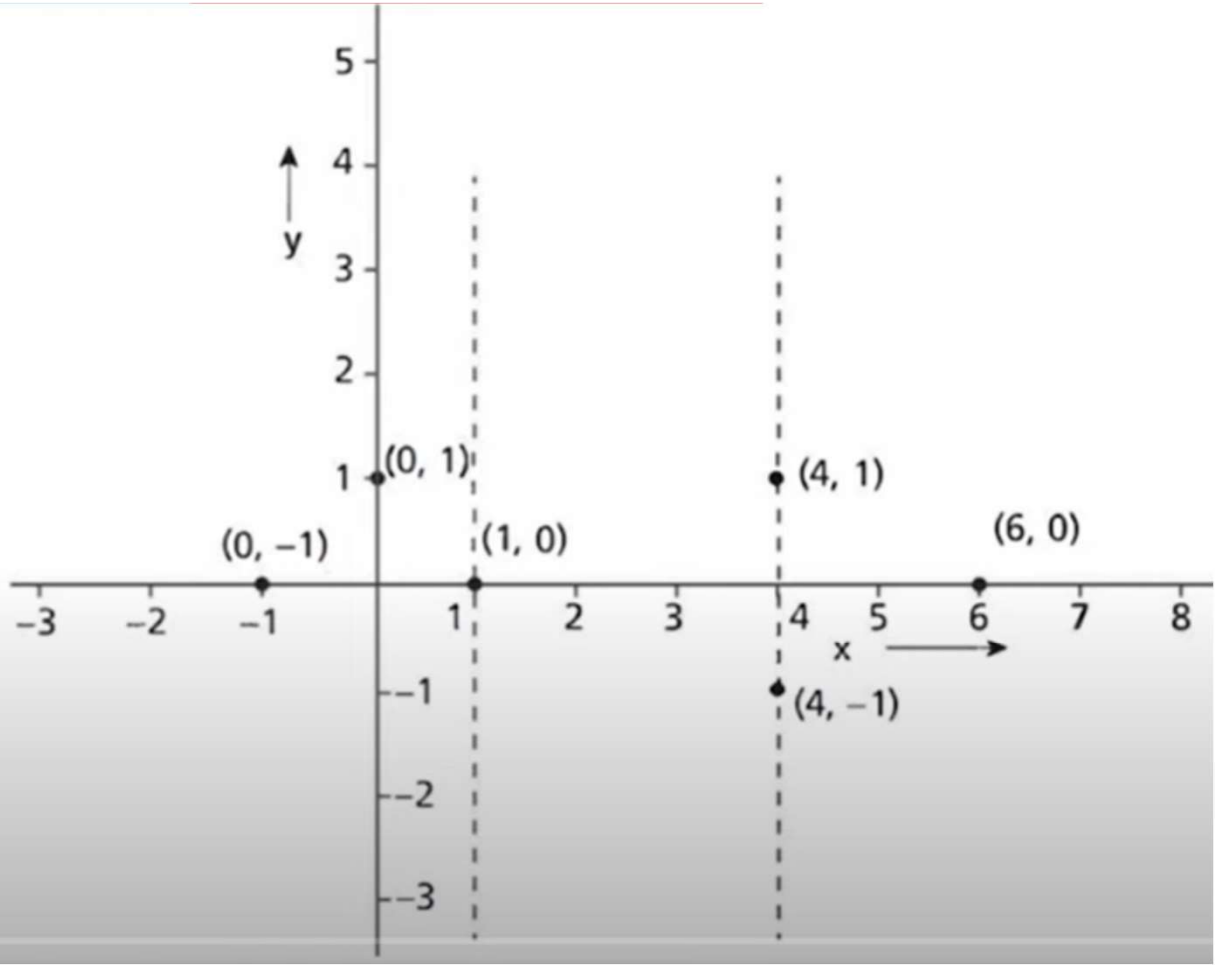

- Points (4, 1), (4, -1) and (6, 0) belong to class positive
- points (1, 0), (0, 1) and (0, -1) belong to negative class.



# Support Vector Machine

- It can be observed that the support vectors are (1, 0), (4, 1) and (4, -1)

$$s_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, s_2 = \begin{pmatrix} 4 \\ 1 \end{pmatrix}, s_3 = \begin{pmatrix} 4 \\ -1 \end{pmatrix}$$



# Support Vector Machine


$$s_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, s_2 = \begin{pmatrix} 4 \\ 1 \end{pmatrix}, s_3 = \begin{pmatrix} 4 \\ -1 \end{pmatrix}$$

- The augmented vector can be obtained by adding the bias given as follows:

$$\tilde{s}_1 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}, \tilde{s}_2 = \begin{pmatrix} 4 \\ 1 \\ 1 \end{pmatrix}, \tilde{s}_3 = \begin{pmatrix} 4 \\ -1 \\ 1 \end{pmatrix}$$

From these, a set of three equations can be obtained based on these three support vectors as follows:

$$\alpha_1 \tilde{s}_1 \tilde{s}_1 + \alpha_2 \tilde{s}_2 \tilde{s}_1 + \alpha_3 \tilde{s}_3 \tilde{s}_1 = -1$$
$$\alpha_1 \tilde{s}_1 \tilde{s}_2 + \alpha_2 \tilde{s}_2 \tilde{s}_2 + \alpha_3 \tilde{s}_3 \tilde{s}_2 = +1$$
$$\alpha_1 \tilde{s}_1 \tilde{s}_3 + \alpha_2 \tilde{s}_2 \tilde{s}_3 + \alpha_3 \tilde{s}_3 \tilde{s}_3 = +1$$

# Support Vector Machine


$$\alpha_1 \begin{pmatrix}1\\0\\1\end{pmatrix}\begin{pmatrix}1\\0\\1\end{pmatrix} + \alpha_2 \begin{pmatrix}4\\1\\1\end{pmatrix}\begin{pmatrix}1\\0\\1\end{pmatrix} + \alpha_3 \begin{pmatrix}4\\-1\\1\end{pmatrix}\begin{pmatrix}1\\0\\1\end{pmatrix}$$
$$= 2\alpha_1 + 5\alpha_2 + 5\alpha_3 = -1$$

$$\tilde{s}_1 = \begin{pmatrix}1\\0\\1\end{pmatrix},\ \tilde{s}_2 = \begin{pmatrix}4\\1\\1\end{pmatrix},\ \tilde{s}_3 = \begin{pmatrix}4\\-1\\1\end{pmatrix}$$

$$\alpha_1 \begin{pmatrix}1\\0\\1\end{pmatrix}\begin{pmatrix}4\\1\\1\end{pmatrix} + \alpha_2 \begin{pmatrix}4\\1\\1\end{pmatrix}\begin{pmatrix}4\\1\\1\end{pmatrix} + \alpha_3 \begin{pmatrix}4\\-1\\1\end{pmatrix}\begin{pmatrix}4\\1\\1\end{pmatrix}$$
$$= 5\alpha_1 + 18\alpha_2 + 16\alpha_3 = +1$$

$$\alpha_1 \begin{pmatrix}1\\0\\1\end{pmatrix}\begin{pmatrix}4\\-1\\1\end{pmatrix} + \alpha_2 \begin{pmatrix}4\\1\\1\end{pmatrix}\begin{pmatrix}4\\-1\\1\end{pmatrix} + \alpha_3 \begin{pmatrix}4\\-1\\1\end{pmatrix}\begin{pmatrix}4\\-1\\1\end{pmatrix}$$
$$= 5\alpha_1 + 16\alpha_2 + 18\alpha_3 = +1$$

$$\alpha_1 \tilde{s}_1 \tilde{s}_1 + \alpha_2 \tilde{s}_2 \tilde{s}_1 + \alpha_3 \tilde{s}_3 \tilde{s}_1 = -1$$
$$\alpha_1 \tilde{s}_1 \tilde{s}_2 + \alpha_2 \tilde{s}_2 \tilde{s}_2 + \alpha_3 \tilde{s}_3 \tilde{s}_2 = +1$$
$$\alpha_1 \tilde{s}_1 \tilde{s}_3 + \alpha_2 \tilde{s}_2 \tilde{s}_3 + \alpha_3 \tilde{s}_3 \tilde{s}_3 = +1$$

# Support Vector Machine


Solving these three simultaneous equations with three unknowns yields the values:

$$\alpha_1 = -3$$

$$\alpha_2 = +1$$

$$\alpha_3 = 0$$

Solving these three simultaneous equations with three unknowns yields the values:

$$\alpha_1 = -3$$

$$\alpha_2 = +1$$

$$\alpha_3 = 0$$

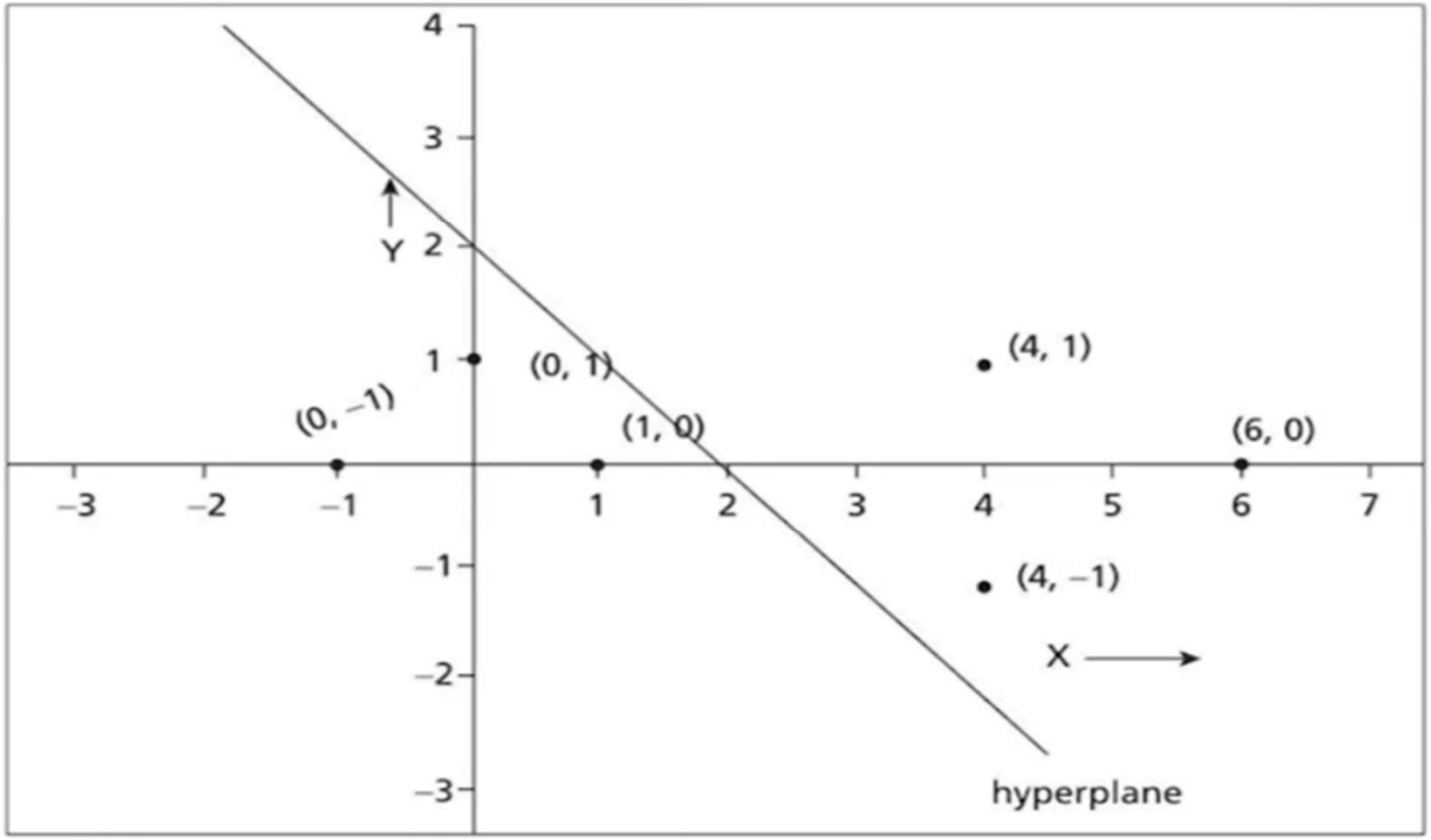
The optimal Hyperplane is given as:

$$w = \sum_{1}^{3} \alpha_i \times \tilde{s}_i$$

$$= -3 \times \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + 1 \times \begin{pmatrix} 4 \\ 1 \\ 1 \end{pmatrix} + 0 \times \begin{pmatrix} 4 \\ -1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ -2 \end{pmatrix}$$

# Support Vector Machine


The hyperplane is (1, 1) with an offset -2.



# Navie Bayes


Naive Bayes is a probabilistic classifier algorithm that uses Bayes' Theorem to calculate the joint probabilities of values and their attributes within a set of cases. It's a simple, popular algorithm that's used in many industrial applications, including: **Spam filtering**:

# Navie Bayes -Example

innovate achieve lead

| No. | Color | Type | Origin | Stolen |
|---|---|---|---|---|
| 1 | Red | Sports | Domestic | Yes |
| 2 | Red | Sports | Domestic | No |
| 3 | Red | Sports | Domestic | Yes |
| 4 | Yellow | Sports | Domestic | No |
| 5 | Yellow | Sports | Imported | Yes |
| 6 | Yellow | SUV | Imported | No |
| 7 | Yellow | SUV | Imported | Yes |
| 8 | Yellow | SUV | Domestic | No |
| 9 | Red | SUV | Imported | No |
| 10 | Red | Sports | Imported | Yes |

$X = \{ \text{Red}, \text{SUV}, \text{Domestic} \}$

$$P(X|Y) = \frac{P(Y|X) \cdot P(X)}{P(Y)}$$

$P(X|\text{Yes}) = ?$

$P(X|\text{No}) = ?$

# Navie Bayes -Example

$$P(\text{Red}|\text{Yes}) = \frac{P(\text{Yes}|\text{Red}) \cdot P(\text{Red})}{P(\text{Yes})} = \frac{3/5 \cdot 5/10}{5/10} = \frac{3}{5}$$

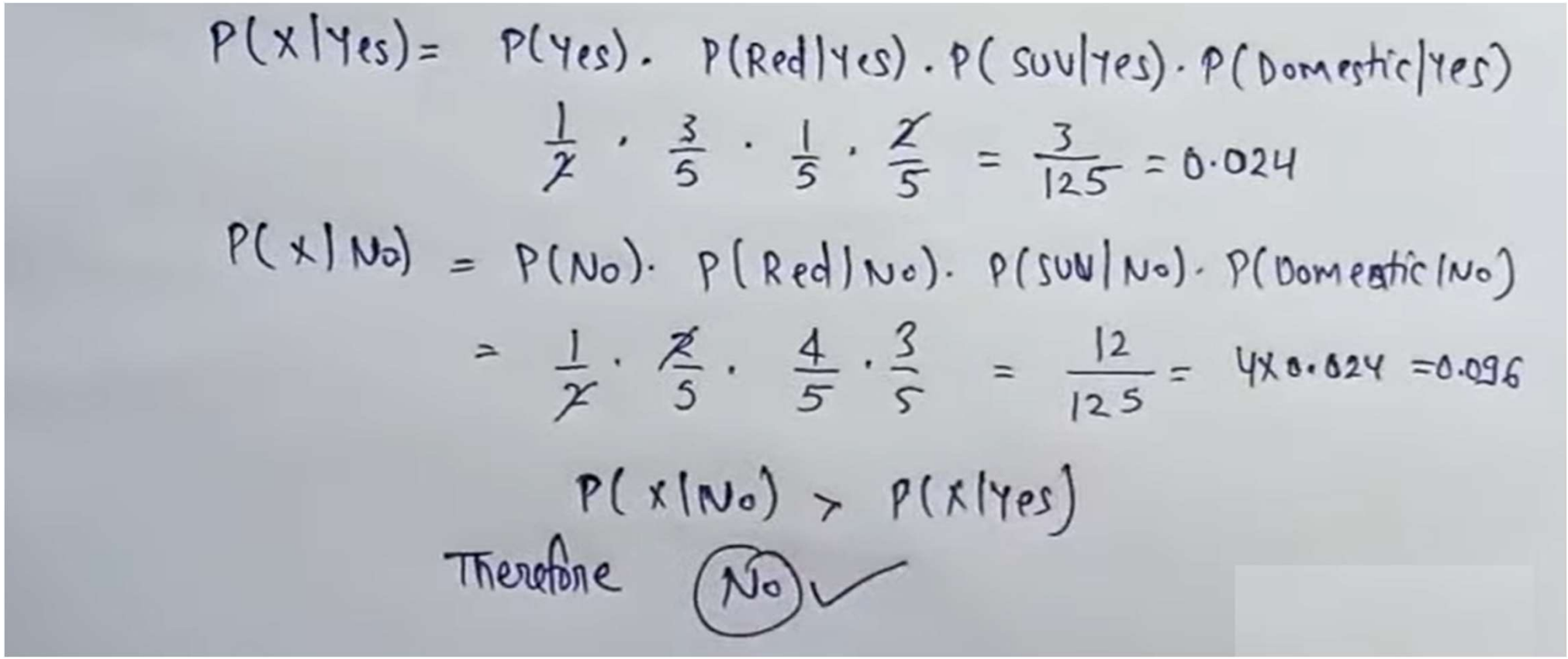
$$P(\text{SUV}|\text{Yes}) = \frac{P(\text{Yes}|\text{SUV}) \cdot P(\text{SUV})}{P(\text{Yes})} = \frac{\frac{1}{4} \cdot \frac{4}{10}}{\frac{5}{10}} = \frac{1}{5}$$

$$P(\text{Domestic}|\text{Yes}) = \frac{P(\text{Yes}|\text{Domestic}) \cdot P(\text{Domestic})}{P(\text{Yes})} = \frac{\frac{2}{5} \cdot \frac{5}{10}}{\frac{5}{10}} = \frac{2}{5}$$

$(\because P + q = 1 \Rightarrow q = 1 - P)$

$$P(\text{Red}|\text{No}) = 1 - \frac{3}{5} = \frac{2}{5}, \quad P(\text{SUV}|\text{No}) = 1 - \frac{1}{5} = \frac{4}{5}$$

# Navie Bayes -Example



| Day | Outlook | Temp | Humidity | Wind | PlayTennis |
|---|---|---|---|---|---|
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| D3 | Overcast | Hot | High | Weak | Yes |
| D4 | Rain | Mild | High | Weak | Yes |
| D5 | Rain | Cool | Normal | Weak | Yes |
| D6 | Rain | Cool | Normal | Strong | No |
| D7 | Overcast | Cool | Normal | Strong | Yes |
| D8 | Sunny | Mild | High | Weak | No |
| D9 | Sunny | Cool | Normal | Weak | Yes |
| D10 | Rain | Mild | Normal | Weak | Yes |
| D11 | Sunny | Mild | Normal | Strong | Yes |
| D12 | Overcast | Mild | High | Strong | Yes |
| D13 | Overcast | Hot | Normal | Weak | Yes |
| D14 | Rain | Mild | High | Strong | No |

## Attribute: Outlook

$Values(Outlook) = Sunny, Overcast, Rain$

$S = [9+, 5-]$ $\quad Entropy(S) = -\frac{9}{14}log_2\frac{9}{14} - \frac{5}{14}log_2\frac{5}{14} = 0.94$

$S_{Sunny} \leftarrow [2+, 3-]$ $\quad Entropy(S_{Sunny}) = -\frac{2}{5}log_2\frac{2}{5} - \frac{3}{5}log_2\frac{3}{5} = 0.971$

$S_{Overcast} \leftarrow [4+, 0-]$ $\quad Entropy(S_{Overcast}) = -\frac{4}{4}log_2\frac{4}{4} - \frac{0}{4}log_2\frac{0}{4} = 0$

$S_{Rain} \leftarrow [3+, 2-]$ $\quad Entropy(S_{Rain}) = -\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5} = 0.971$

$$Gain(S, Outlook) = Entropy(S) - \sum_{v \in \{Sunny, Overcast, Rain\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, Outlook) = Entropy(S) - \frac{5}{14}Entropy(S_{Sunny}) - \frac{4}{14}Entropy(S_{Overcast}) - \frac{5}{14}Entropy(S_{Rain})$$

$$Gain(S, Outlook) = 0.94 - \frac{5}{14}0.971 - \frac{4}{14}0 - \frac{5}{14}0.971 = 0.2464$$

# Decision Tree

| Day | Outlook | Temp | Humidity | Wind | PlayTennis |
|---|---|---|---|---|---|
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| D3 | Overcast | Hot | High | Weak | Yes |
| D4 | Rain | Mild | High | Weak | Yes |
| D5 | Rain | Cool | Normal | Weak | Yes |
| D6 | Rain | Cool | Normal | Strong | No |
| D7 | Overcast | Cool | Normal | Strong | Yes |
| D8 | Sunny | Mild | High | Weak | No |
| D9 | Sunny | Cool | Normal | Weak | Yes |
| D10 | Rain | Mild | Normal | Weak | Yes |
| D11 | Sunny | Mild | Normal | Strong | Yes |
| D12 | Overcast | Mild | High | Strong | Yes |
| D13 | Overcast | Hot | Normal | Weak | Yes |
| D14 | Rain | Mild | High | Strong | No |

## Attribute: Temp

$Values\ (Temp) = Hot, Mild, Cool$

$S = [9+, 5-]$ $\quad Entropy(S) = -\frac{9}{14} log_2 \frac{9}{14} - \frac{5}{14} log_2 \frac{5}{14} = 0.94$

$S_{Hot} \leftarrow [2+, 2-]$ $\quad Entropy(S_{Hot}) = -\frac{2}{4} log_2 \frac{2}{4} - \frac{2}{4} log_2 \frac{2}{4} = 1.0$

$S_{Mild} \leftarrow [4+, 2-]$ $\quad Entropy(S_{Mild}) = -\frac{4}{6} log_2 \frac{4}{6} - \frac{2}{6} log_2 \frac{2}{6} = 0.9183$

$S_{Cool} \leftarrow [3+, 1-]$ $\quad Entropy(S_{Cool}) = -\frac{3}{4} log_2 \frac{3}{4} - \frac{1}{4} log_2 \frac{1}{4} = 0.8113$

$$Gain\ (S, Temp) = Entropy(S) - \sum_{v \in \{Hot, Mild, Cool\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$Gain(S, Temp)$

$$= Entropy(S) - \frac{4}{14} Entropy(S_{Hot}) - \frac{6}{14} Entropy(S_{Mild})$$

$$- \frac{4}{14} Entropy(S_{Cool})$$

$$Gain(S, Temp) = 0.94 - \frac{4}{14} 1.0 - \frac{6}{14}\ 0.9183 - \frac{4}{14} 0.8113 = 0.0289$$

# Decision Tree


| Day | Outlook | Temp | Humidity | Wind | PlayTennis |
|---|---|---|---|---|---|
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| D3 | Overcast | Hot | High | Weak | Yes |
| D4 | Rain | Mild | High | Weak | Yes |
| D5 | Rain | Cool | Normal | Weak | Yes |
| D6 | Rain | Cool | Normal | Strong | No |
| D7 | Overcast | Cool | Normal | Strong | Yes |
| D8 | Sunny | Mild | High | Weak | No |
| D9 | Sunny | Cool | Normal | Weak | Yes |
| D10 | Rain | Mild | Normal | Weak | Yes |
| D11 | Sunny | Mild | Normal | Strong | Yes |
| D12 | Overcast | Mild | High | Strong | Yes |
| D13 | Overcast | Hot | Normal | Weak | Yes |
| D14 | Rain | Mild | High | Strong | No |

## Attribute: Humidity

$Values\,(Humidity) = High, Normal$

$S = [9+, 5-]$ $\qquad Entropy(S) = -\frac{9}{14}log_2\frac{9}{14} - \frac{5}{14}log_2\frac{5}{14} = 0.94$

$S_{High} \leftarrow [3+, 4-]$ $\qquad Entropy(S_{High}) = -\frac{3}{7}log_2\frac{3}{7} - \frac{4}{7}log_2\frac{4}{7} = 0.9852$

$S_{Normal} \leftarrow [6+, 1-]$ $\qquad Entropy(S_{Normal}) = -\frac{6}{7}log_2\frac{6}{7} - \frac{1}{7}log_2\frac{1}{7} = 0.5916$

$$Gain\,(S, Humidity) = Entropy(S) - \sum_{v \in \{High, Normal\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, Humidity)$$

$$= Entropy(S) - \frac{7}{14}Entropy(S_{High}) - \frac{7}{14}Entropy(S_{Normal})$$

$$Gain(S, Humidity) = 0.94 - \frac{7}{14}0.9852 - \frac{7}{14}0.5916 = 0.1516$$

# Decision Tree


| Day | Outlook | Temp | Humidity | Wind | PlayTennis |
|---|---|---|---|---|---|
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| D3 | Overcast | Hot | High | Weak | Yes |
| D4 | Rain | Mild | High | Weak | Yes |
| D5 | Rain | Cool | Normal | Weak | Yes |
| D6 | Rain | Cool | Normal | Strong | No |
| D7 | Overcast | Cool | Normal | Strong | Yes |
| D8 | Sunny | Mild | High | Weak | No |
| D9 | Sunny | Cool | Normal | Weak | Yes |
| D10 | Rain | Mild | Normal | Weak | Yes |
| D11 | Sunny | Mild | Normal | Strong | Yes |
| D12 | Overcast | Mild | High | Strong | Yes |
| D13 | Overcast | Hot | Normal | Weak | Yes |
| D14 | Rain | Mild | High | Strong | No |

## Attribute: Wind

$Values\,(Wind) = Strong, Weak$

$S = [9+, 5-]$ $\qquad Entropy(S) = -\frac{9}{14} log_2 \frac{9}{14} - \frac{5}{14} log_2 \frac{5}{14} = 0.94$

$S_{Strong} \leftarrow [3+, 3-]$ $\qquad Entropy(S_{Strong}) = 1.0$

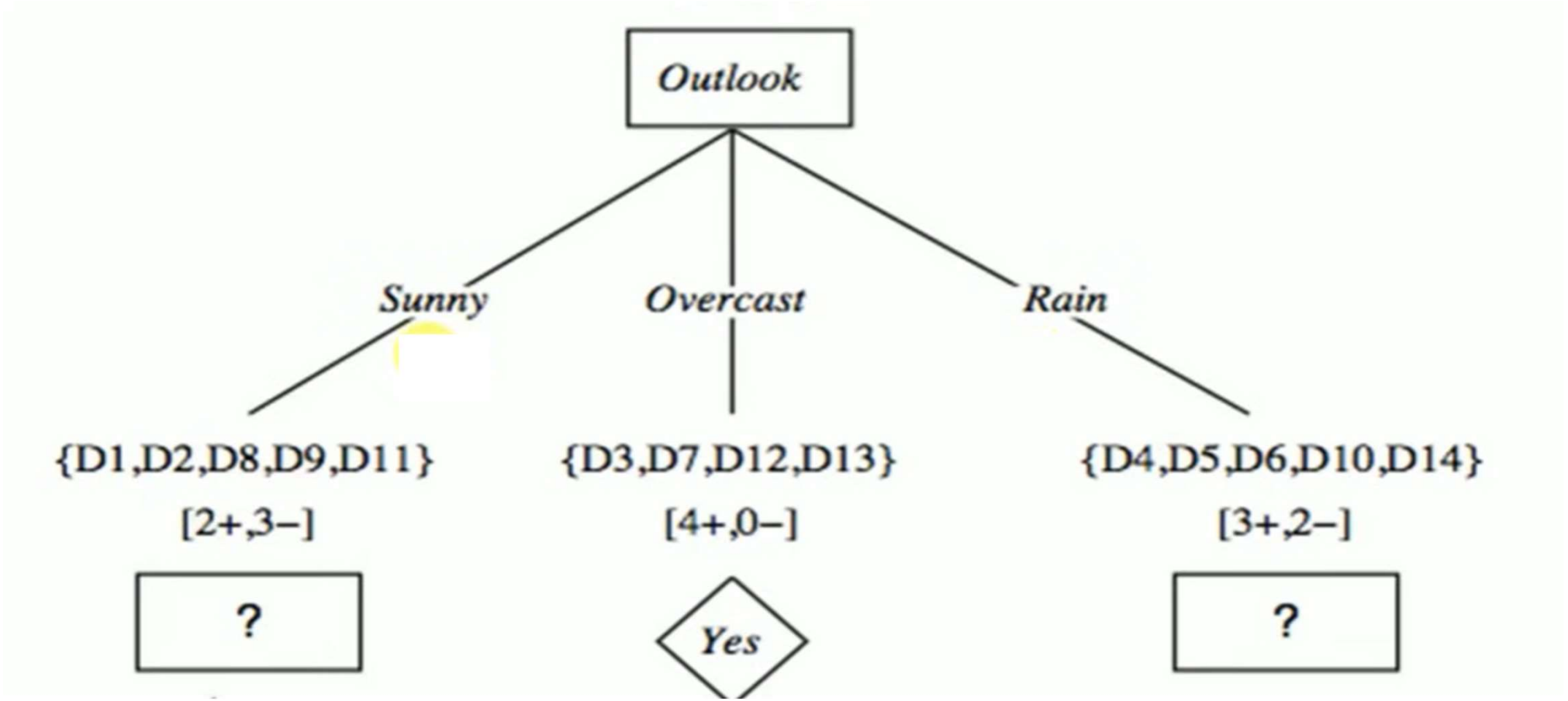
$S_{Weak} \leftarrow [6+, 2-]$ $\qquad Entropy(S_{Weak}) = -\frac{6}{8} log_2 \frac{6}{8} - \frac{2}{8} log_2 \frac{2}{8} = 0.8113$

$$Gain\,(S, Wind) = Entropy(S) - \sum_{v \in \{Strong, Weak\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, Wind) = Entropy(S) - \frac{6}{14} Entropy(S_{Strong}) - \frac{8}{14} Entropy(S_{Weak})$$

$$Gain(S, Wind) = 0.94 - \frac{6}{14}\, 1.0 - \frac{8}{14}\, 0.8113 = 0.0478$$

# Decision Tree



# Decision Tree


| Day | Temp | Humidity | Wind | Play Tennis |
|---|---|---|---|---|
| D1 | Hot | High | Weak | No |
| D2 | Hot | High | Strong | No |
| D8 | Mild | High | Weak | No |
| D9 | Cool | Normal | Weak | Yes |
| D11 | Mild | Normal | Strong | Yes |

**Attribute: Temp**

$Values\ (Temp) = Hot, Mild, Cool$

$S_{Sunny} = [2+, 3-]$ $\quad Entropy(S_{Sunny}) = -\frac{2}{5}log_2\frac{2}{5} - \frac{3}{5}log_2\frac{3}{5} = 0.97$

$S_{Hot} \leftarrow [0+, 2-]$ $\quad Entropy(S_{Hot}) = 0.0$

$S_{Mild} \leftarrow [1+, 1-]$ $\quad Entropy(S_{Mild}) = 1.0$

$S_{Cool} \leftarrow [1+, 0-]$ $\quad Entropy(S_{Cool}) = 0.0$

$$Gain\ (S_{Sunny}, Temp) = Entropy(S) - \sum_{v \in \{Hot, Mild, Cool\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S_{Sunny}, Temp)$$

$$= Entropy(S) - \frac{2}{5}Entropy(S_{Hot}) - \frac{2}{5}Entropy(S_{Mild})$$

$$- \frac{1}{5}Entropy(S_{Cool})$$

$$Gain(S_{sunny}, Temp) = 0.97 - \frac{2}{5}0.0 - \frac{2}{5}1 - \frac{1}{5}0.0 = 0.570$$

# Decision Tree



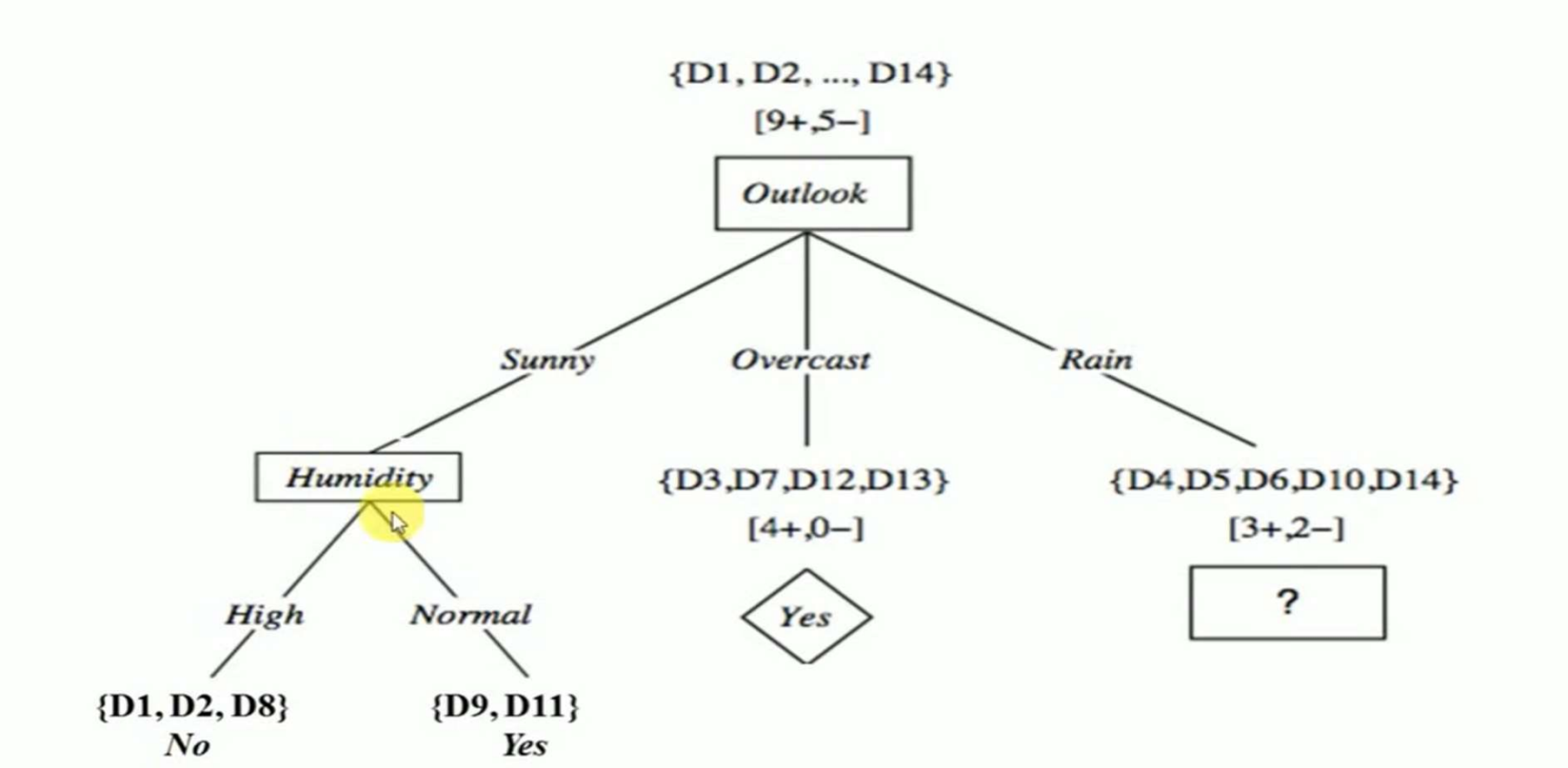
| Day | Temp | Humidity | Wind | Play Tennis |
|---|---|---|---|---|
| D1 | Hot | High | Weak | No |
| D2 | Hot | High | Strong | No |
| D8 | Mild | High | Weak | No |
| D9 | Cool | Normal | Weak | Yes |
| D11 | Mild | Normal | Strong | Yes |

$$Gain(S_{sunny}, Temp) = 0.570$$

$$Gain(S_{sunny}, Humidity) = 0.97$$

$$Gain(S_{sunny}, Wind) = 0.0192$$

# Decision Tree



# Regression


Regression analysis is a set of statistical methods used for the estimation of relationships between a dependent variable and one or more independent variables. It can be utilized to assess the strength of the relationship between variables and for modeling the future relationship between them.

# Logistic Regression


- The dataset of pass or fail in an exam of 5 students is given in the table.
- Use logistic regression as classifier to answer the following questions.

1. Calculate the probability of pass for the student who studied 33 hours.
2. At least how many hours student should study that makes he will pass the course with the probability of more than 95%.

| Hours Study | Pass (1) / Fail (0) |
|---|---|
| 29 | 0 |
| 15 | 0 |
| 33 | 1 |
| 28 | 1 |
| 39 | 1 |

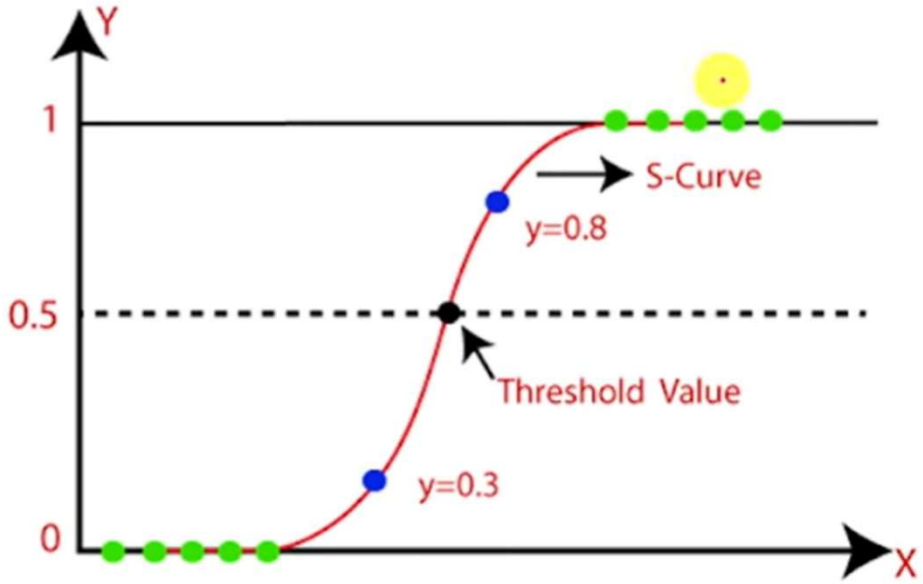
Assume the model suggested by the optimizer for odds of passing the course is,

$$\log(odds) = -64 + 2 * hours$$

# Logistic Regression


- We use Sigmoid Function in logistic regression

- $s(x) = \frac{1}{1+e^{-x}}$



# Logistic Regression


1. Calculate the probability of pass for the student who studied 33 hours.

- $p = \frac{1}{1+e^{-z}}$ $\qquad s(x) = \frac{1}{1+e^{-x}}$
- $z = -64 + 2 * 33 = -64 + 66 = 2$
- $p = \frac{1}{1+e^{-2}} = 0.88$
- That is, if student studies 33 hours, then there is **88% chance** that the student will pass the exam

| Hours Study | Pass (1) / Fail (0) |
|---|---|
| 29 | 0 |
| 15 | 0 |
| 33 | 1 |
| 28 | 1 |
| 39 | 1 |

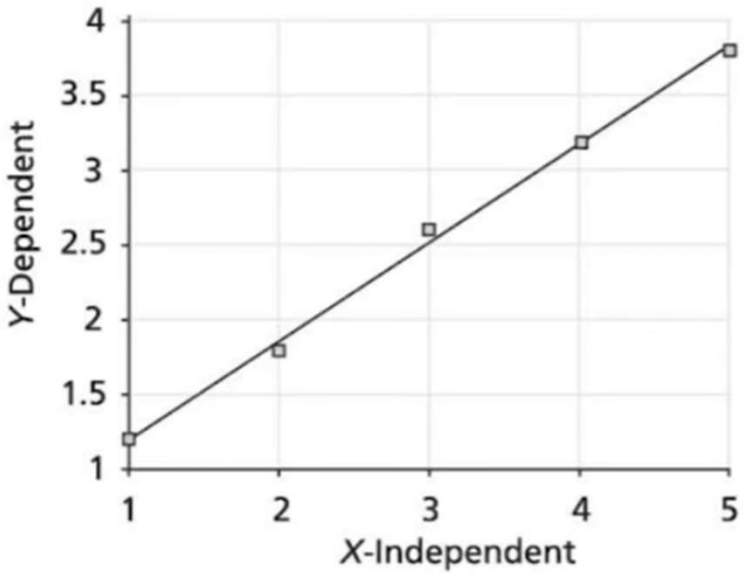
$$\log(odds) = z = -64 + 2 * hours$$

# Linear Regression


- Let us consider an example where the five weeks' sales data (in Thousands) is given as shown in Table.
- Apply linear regression technique to predict the $7^{th}$ and $12^{th}$ week sales.

| $x_i$ $(Week)$ | $y_j$ $(Sales\ in\ Thousands)$ |
|---|---|
| 1 | 1.2 |
| 2 | 1.8 |
| 3 | 2.6 |
| 4 | 3.2 |
| 5 | 3.8 |

# Linear Regression




| $x_i$ (Week) | $y_j$ (Sales in Thousands) |
|---|---|
| 1 | 1.2 |
| 2 | 1.8 |
| 3 | 2.6 |
| 4 | 3.2 |
| 5 | 3.8 |

# Linear Regression


- Linear regression equation is given by
- $y = a_0 + a_1 * x + e$
- *where*
- $a_1 = \frac{(\overline{xy}) - (\bar{x})(\bar{y})}{\overline{x^2} - \bar{x}^2}$
- $a_0 = \bar{y} - a_1 * \bar{x}$

| $x_i$ (*Week*) | $y_j$ (*Sales in Thousands*) |
|---|---|
| 1 | 1.2 |
| 2 | 1.8 |
| 3 | 2.6 |
| 4 | 3.2 |
| 5 | 3.8 |

# Linear Regression

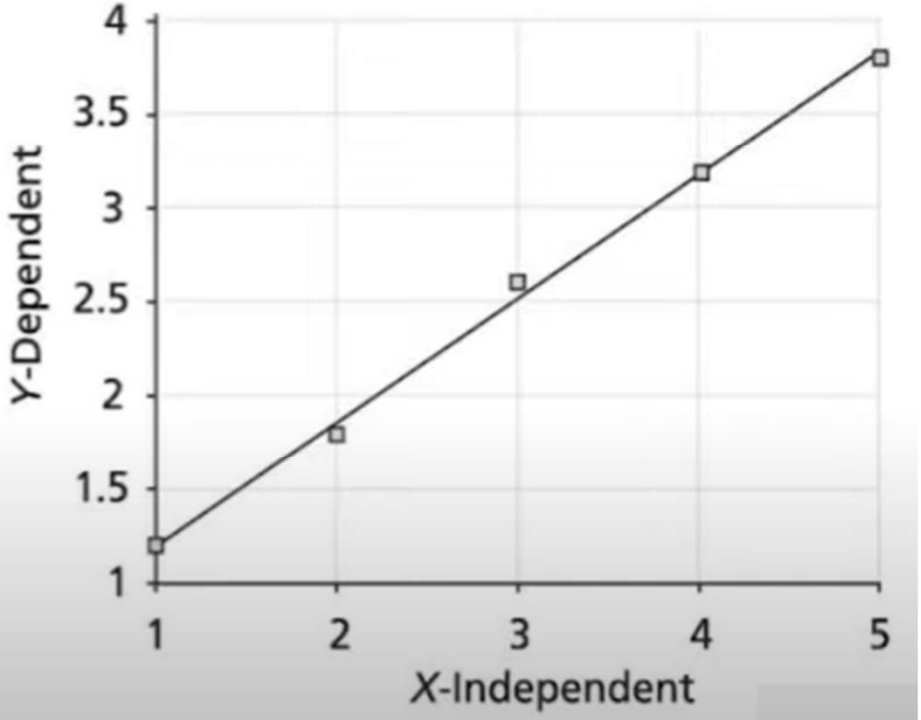

- Here, there are 5 items, i.e., i = 1, 2, 3, 4, 5.

| | $x_i$ (Week) | $y_j$ (Sales in Thousands) | $x_i^2$ | $x_i * y_j$ |
|---|---|---|---|---|
| | 1 | 1.2 | 1 | 1.2 |
| | 2 | 1.8 | 4 | 3.6 |
| | 3 | 2.6 | 9 | 7.8 |
| | 4 | 3.2 | 16 | 12.8 |
| | 5 | 3.8 | 25 | 19 |
| Sum | 15 | 12.6 | 55 | 44.4 |
| Average | $\bar{x} = 3$ | $\bar{y} = 2.52$ | $\overline{x^2} = 11$ | $\overline{xy} = 8.88$ |

# Linear Regression


- $\bar{x} = 3 \qquad \bar{y} = 2.52 \qquad x^2 = 11 \qquad \overline{xy} = 8.88$

- $a_1 = \frac{(\overline{xy}) - (\bar{x})(\bar{y})}{\overline{x^2} - \bar{x}^2} = \frac{8.88 - 3 * 2.52}{11 - 3^2} = 0.66$
- $a_0 = \bar{y} - a_1 * \bar{x} = 2.52 - 0.66 * 3 = 0.54$
- **Regression equation is**
- $y = a_0 + a_1 * x$
- $y = 0.54 + 0.66 * x$



# Linear Regression


- **Regression equation is**
- $y = a_0 + a_1 * x$
- $y = 0.54 + 0.66 * x$
- The predicted 7th week sale (when x = 7) is,
- **y = 0.54 + 0.66 x 7 = 5.16**
- the predicted 12th week sale (when x = 12) is,
- **y = 0.54 + 0.66 x 12 = 8.46**

# Multi Linear Regression


- In linear regression model we have one dependent and one independent variable.
- Multiple regression model involves multiple predictors or independent variables and one dependent variable.
- This is an extension of the linear regression problem.

# Multi Linear Regression


- The multiple regression of two variables x1 and x2 is given as follows:

$$y = f(x_1, x_2)$$

$$y = a_0 + a_1 x_1 + a_2 x_2$$

- In general, this is given for 'n' independent variables as:

$$y = f(x_1, x_2, \ldots . x_n)$$

$$y = a_0 + a_1 x_1 + a_2 x_2 + \ldots \ldots + a_n x_n + \varepsilon$$

- Here, $x_1, x_2, \ldots . x_n$ are predictor variables, $y$ is the dependent variable, $(a_0, a_1, a_2, \ldots . a_n)$ are the coefficients of the regression equation and $\varepsilon$ is the error term.

# Multi Linear Regression


- Here, the matrices for Y and X are given as follows:

$$X = \begin{pmatrix} 1 & 1 & 4 \\ 1 & 2 & 5 \\ 1 & 3 & 8 \\ 1 & 4 & 2 \end{pmatrix} \text{ and } Y = \begin{pmatrix} 1 \\ 6 \\ 8 \\ 12 \end{pmatrix}.$$

- The coefficient of the multiple regression equation is given as

$$a = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix}.$$

| x1 Product 1 Sales | x2 Product 2 Sales | Y Weekly Sales |
|---|---|---|
| 1 | 4 | 1 |
| 2 | 5 | 6 |
| 3 | 8 | 8 |
| 4 | 2 | 12 |

# Multi Linear Regression



- The regression coefficient for multiple regression is calculated the same way as linear regression:

$$\hat{a} = ((X^T X)^{-1} X^T) Y$$

| x1 Product 1 Sales | x2 Product 2 Sales | Y Weekly Sales |
|---|---|---|
| 1 | 4 | 1 |
| 2 | 5 | 6 |
| 3 | 8 | 8 |
| 4 | 2 | 12 |

# Multi Linear Regression


- The regression coefficient for multiple regression is calculated the same way as linear regression:

$$\hat{a} = ((X^T X)^{-1} X^T) Y$$

$$X^T X = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \\ 4 & 5 & 8 & 2 \end{pmatrix} \begin{pmatrix} 1 & 1 & 4 \\ 1 & 2 & 5 \\ 1 & 3 & 8 \\ 1 & 4 & 2 \end{pmatrix} = \begin{pmatrix} 4 & 10 & 19 \\ 10 & 30 & 46 \\ 19 & 46 & 109 \end{pmatrix}$$

| x1 Product 1 Sales | x2 Product 2 Sales | Y Weekly Sales |
|---|---|---|
| 1 | 4 | 1 |
| 2 | 5 | 6 |
| 3 | 8 | 8 |
| 4 | 2 | 12 |

# Multi Linear Regression


$$X^T X)^{-1} X^T = \begin{pmatrix} 3.15 & -0.59 & -0.30 \\ -0.59 & 0.20 & 0.016 \\ -0.30 & 0.016 & 0.054 \end{pmatrix} \times \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \\ 4 & 5 & 8 & 2 \end{pmatrix} = \begin{pmatrix} 0.05 & 0.47 & -1.02 & 0.19 \\ -0.32 & -0.098 & 0.155 & 0.26 \\ -0.065 & 0.005 & 0.185 & -0.125 \end{pmatrix}$$

$$\hat{a} = ((X^T X)^{-1} X^T)Y = \begin{pmatrix} 0.05 & 0.47 & -1.02 & 0.19 \\ -0.32 & -0.098 & 0.155 & 0.26 \\ -0.065 & 0.005 & 0.185 & -0.125 \end{pmatrix} \times \begin{pmatrix} 1 \\ 6 \\ 8 \\ 12 \end{pmatrix} = \begin{pmatrix} -1.69 \\ 3.48 \\ -0.05 \end{pmatrix}$$

# Multi Linear Regression


$$a_0 = -1.69$$
$$a_1 = 3.48$$
$$a_2 = -0.05$$

| x1 Product 1 Sales | x2 Product 2 Sales | Y Weekly Sales |
|---|---|---|
| 1 | 4 | 1 |
| 2 | 5 | 6 |
| 3 | 8 | 8 |
| 4 | 2 | 12 |

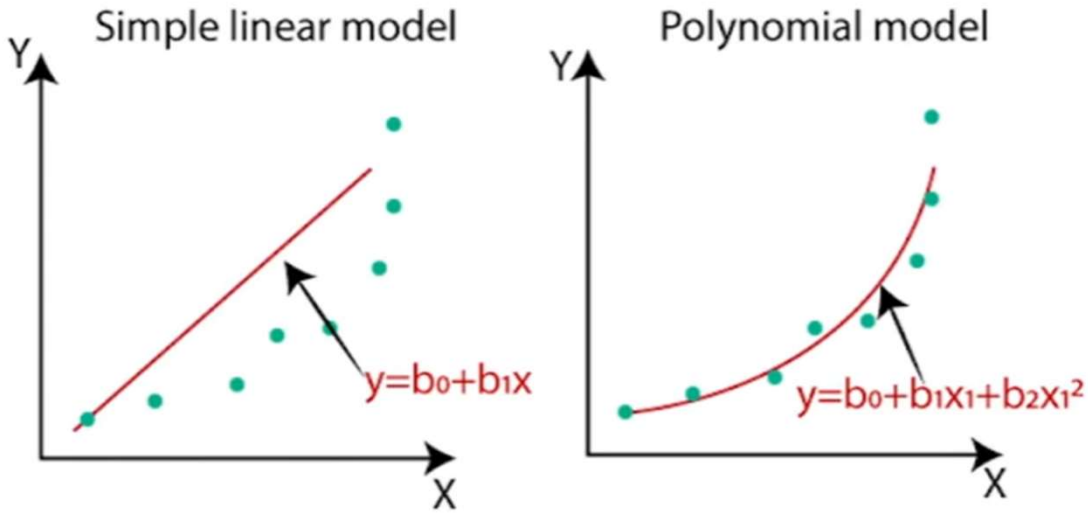
- $y = a_0 + a_1x_1 + a_2x_2$
- Hence, the constructed model is:
- $y = -1.69 + 3.48x_1 - 0.05x_2$

X1 & X2 ----Independent
Y -- Dependent

# Polynomial Regression



- If the relationship between the independent and dependent variables is not linear, then linear regression cannot be used as it will result in large errors.



- The problem of non-linear regression can be solved by two methods:

  1. Transformation of non-linear data to linear data, so that the linea regression can handle the data
  2. Using polynomial regression

**Transformations**

- The trick is to convert non-linear data to linear data that can be handled using the linear regression method.
- Let us consider an exponential function $y = ae^{bx}$.
- The transformation can be done by applying log function to both sides to get:

$$ln(y) = ln(a) + ln(e^{bx})$$

$$ln(y) = ln(a) + bx * ln(e)$$

$$ln(y) = ln(a) + bx$$

## Polynomial Regression

- It can handle non-linear relationships among variables by using $n^{th}$ degree of a polynomial.
- Instead of applying transformations, polynomial regression can be directly used to deal with different levels of curvilinearity.
- For example, the second-degree polynomial ( called quadratic transformation) is given as: $y = a_0 + a_1 x + a_2 x^2$ and third degree polynomial is called cubic transformation given as: $y = a_0 + a_1 x + a_2 x^2 + a_3 x^3$
- Generally, polynomials of maximum degree 4 are used, as higher order polynomials take some strange shapes and make the curve more flexible.
- It leads to a situation of overfitting and hence is avoided.

- Consider the polynomial of 2nd degree.
- The polynomial equation in given by $\boldsymbol{y = a_0 + a_1 x + a_2 x^2}$
- The coefficients $\boldsymbol{a_0, a_1}$ $\boldsymbol{and}$ $\boldsymbol{a_2}$ are calculated using the formula,

$$\boldsymbol{a = X^{-1} B}$$

- Where,

$$X = \begin{bmatrix} n & \sum x_i & \sum x_i^2 \\ \sum x_i & \sum x_i^2 & \sum x_i^3 \\ \sum x_i^2 & \sum x_i^3 & \sum x_i^4 \end{bmatrix} \quad B = \begin{bmatrix} \sum y_i \\ \sum (x_i, y_i) \\ \sum (x_i^2, y_i) \end{bmatrix}$$

| $x_i$ | $y_i$ | $x_i y_i$ | $x_i^2$ | $x_i^2 y$ | $x_i^3$ | $x_i^4$ |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 2 | 4 | 8 | 4 | 16 | 8 | 16 |
| 3 | 9 | 27 | 9 | 81 | 27 | 81 |
| 4 | 15 | 60 | 16 | 240 | 64 | 256 |
| $\Sigma x_i = 10$ | $\Sigma y_i = 29$ | $\Sigma x_i y_i = 96$ | $\Sigma x_i^2 = 30$ | $\Sigma x_i^2 y_i = 338$ | $\Sigma x_i^3 = 100$ | $\Sigma x_i^4 = 354$ |

$$\boldsymbol{a = X^{-1}B} \quad X = \begin{bmatrix} n & \Sigma x_i & \Sigma x_i^2 \\ \Sigma x_i & \Sigma x_i^2 & \Sigma x_i^3 \\ \Sigma x_i^2 & \Sigma x_i^3 & \Sigma x_i^4 \end{bmatrix} \quad B = \begin{bmatrix} \Sigma y_i \\ \Sigma(x_i, y_i) \\ \Sigma(x_i^2, y_i) \end{bmatrix} \quad \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 4 & 10 & 30 \\ 10 & 30 & 100 \\ 30 & 100 & 354 \end{bmatrix}^{-1} \times \begin{bmatrix} 29 \\ 96 \\ 338 \end{bmatrix} = \begin{pmatrix} -0.75 \\ 0.95 \\ 0.75 \end{pmatrix}$$

$$\boldsymbol{a = X^{-1}B} \quad \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 4 & 10 & 30 \\ 10 & 30 & 100 \\ 30 & 100 & 354 \end{bmatrix}^{-1} \times \begin{bmatrix} 29 \\ 96 \\ 338 \end{bmatrix} = \begin{pmatrix} -0.75 \\ 0.95 \\ 0.75 \end{pmatrix} \quad y = -0.75 + 0.95\,x + 0.75\,x^2$$

# Unsupervised algo -- Clustering


**Clustering** is an unsupervised machine learning technique designed to group **unlabeled examples** based on their similarity to each other. (If the examples are labeled, this kind of grouping is called **classification**.) Consider a hypothetical patient study designed to evaluate a new treatment protocol.

# K-Means


- Suppose that the data mining task is to cluster points into three clusters,
- where the points are
- $A1(2, 10)$, $A2(2, 5)$, $A3(8, 4)$, $B1(5, 8)$, $B2(7, 5)$, $B3(6, 4)$, $C1(1, 2)$, $C2(4, 9)$.
- The distance function is Euclidean distance.
- Suppose initially we assign $A1$, $B1$, and $C1$ as the center of each cluster, respectively.

# K-Means


Initial Centroids:
A1: (2, 10)
B1: (5, 8)
C1: (1, 2)

| Data Points | | | Distance to | | | | | | Cluster | New Cluster |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | 2 | 10 | 5 | 8 | 1 | 2 | | |
| A1 | 2 | 10 | 0.00 | | 3.61 | | 8.06 | | 1 | |
| A2 | 2 | 5 | 5.00 | | 4.24 | | 3.16 | | 3 | |
| A3 | 8 | 4 | 8.49 | | 5.00 | | 7.28 | | 2 | |
| B1 | 5 | 8 | 3.61 | | 0.00 | | 7.21 | | 2 | |
| B2 | 7 | 5 | 7.07 | | 3.61 | | 6.71 | | 2 | |
| B3 | 6 | 4 | 7.21 | | 4.12 | | 5.39 | | 2 | |
| C1 | 1 | 2 | 8.06 | | 7.21 | | 0.00 | | 3 | |
| C2 | 4 | 9 | 2.24 | | 1.41 | | 7.62 | | 2 | |

$$d(p_1, p_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

# K-Means


Current Centroids:
A1: (2, 10)
B1: (6, 6)
C1: (1.5, 3.5)

New Centroids:
A1: (3, 9.5)
B1: (6.5, 5.25)
C1: (1.5, 3.5)

| Data Points | | | Distance to | | | | | | Cluster | New Cluster |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | 2 | 10 | 6 | 6 | 1.5 | 1.5 | | |
| A1 | 2 | 10 | 0.00 | | 5.66 | | 6.52 | | 1 | 1 |
| A2 | 2 | 5 | 5.00 | | 4.12 | | 1.58 | | 3 | 3 |
| A3 | 8 | 4 | 8.49 | | 2.83 | | 6.52 | | 2 | 2 |
| B1 | 5 | 8 | 3.61 | | 2.24 | | 5.70 | | 2 | 2 |
| B2 | 7 | 5 | 7.07 | | 1.41 | | 5.70 | | 2 | 2 |
| B3 | 6 | 4 | 7.21 | | 2.00 | | 4.53 | | 2 | 2 |
| C1 | 1 | 2 | 8.06 | | 6.40 | | 1.58 | | 3 | 3 |
| C2 | 4 | 9 | 2.24 | | 3.61 | | 6.04 | | 2 | 1 |

$$d(p_1, p_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

# K-Means


Current Centroids:
A1: (3, 9.5)
B1: (6.5, 5.25)
C1: (1.5, 3.5)

New Centroids:
A1: (3.67, 9).
B1: (7, 4.33)
C1: (1.5, 3.5)

| Data Points | | | Distance to | | | | | | Cluster | New Cluster |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | 3 | 9.5 | 6.5 | 5.25 | 1.5 | 3.5 | | |
| A1 | 2 | 10 | 1.12 | | 6.54 | | 6.52 | | 1 | 1 |
| A2 | 2 | 5 | 4.61 | | 4.51 | | 1.58 | | 3 | 3 |
| A3 | 8 | 4 | 7.43 | | 1.95 | | 6.52 | | 2 | 2 |
| B1 | 5 | 8 | 2.50 | | 3.13 | | 5.70 | | 2 | 1 |
| B2 | 7 | 5 | 6.02 | | 0.56 | | 5.70 | | 2 | 2 |
| B3 | 6 | 4 | 6.26 | | 1.35 | | 4.53 | | 2 | 2 |
| C1 | 1 | 2 | 7.76 | | 6.39 | | 1.58 | | 3 | 3 |
| C2 | 4 | 9 | 1.12 | | 4.51 | | 6.04 | | 1 | 1 |

$$d(p_1, p_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

# K-Means


Current Centroids:
A1: (3.67, 9)
B1: (7, 4.33)
C1: (1.5, 3.5)

| Data Points | | | Distance to | | | | | | Cluster | New Cluster |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | 3.67 | 9 | 7 | 4.33 | 1.5 | 3.5 | | |
| A1 | 2 | 10 | 1.94 | | 7.56 | | 6.52 | | 1 | 1 |
| A2 | 2 | 5 | 4.33 | | 5.04 | | 1.58 | | 3 | 3 |
| A3 | 8 | 4 | 6.62 | | 1.05 | | 6.52 | | 2 | 2 |
| B1 | 5 | 8 | 1.67 | | 4.18 | | 5.70 | | 1 | 1 |
| B2 | 7 | 5 | 5.21 | | 0.67 | | 5.70 | | 2 | 2 |
| B3 | 6 | 4 | 5.52 | | 1.05 | | 4.53 | | 2 | 2 |
| C1 | 1 | 2 | 7.49 | | 6.44 | | 1.58 | | 3 | 3 |
| C2 | 4 | 9 | 0.33 | | 5.55 | | 6.04 | | 1 | 1 |

$$d(p_1, p_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

# Agglomerative Hierarchical Clustering


## Agglomerative Hierarchical Clustering Solved Example

- Consider the following set of 6 one dimensional data points:
- 18, 22, 25, 42, 27, 43
- Apply the **agglomerative hierarchical clustering** algorithm to build the hierarchical clustering **dendogram**.
- Merge the clusters using **Min distance** and update the proximity matrix accordingly.
- Clearly show the **proximity matrix** corresponding to each iteration of the algorithm.

- Step – 1

| | 18 | 22 | 25 | 27 | 42 | 43 |
|---|---|---|---|---|---|---|
| 18 | 0 | 4 | 7 | 9 | 24 | 25 |
| 22 | 4 | 0 | 3 | 5 | 20 | 21 |
| 25 | 7 | 3 | 0 | 2 | 17 | 18 |
| 27 | 9 | 5 | 2 | 0 | 15 | 16 |
| 42 | 24 | 20 | 17 | 15 | 0 | 1 |
| 43 | 25 | 21 | 18 | 16 | 1 | 0 |

(42, 43)

- Step – 2

| | 18 | 22 | 25 | 27 | 42, 43 |
|---|---|---|---|---|---|
| 18 | 0 | 4 | 7 | 9 | 24 |
| 22 | 4 | 0 | 3 | 5 | 20 |
| 25 | 7 | 3 | 0 | 2 | 17 |
| 27 | 9 | 5 | 2 | 0 | 15 |
| 42, 43 | 24 | 20 | 17 | 15 | 0 |

- Step – 2

| | 18 | 22 | 25 | 27 | 42, 43 |
|---|---|---|---|---|---|
| 18 | 0 | 4 | 7 | 9 | 24 |
| 22 | 4 | 0 | 3 | 5 | 20 |
| 25 | 7 | 3 | 0 | 2 | 17 |
| 27 | 9 | 5 | 2 | 0 | 15 |
| 42, 43 | 24 | 20 | 17 | 15 | 0 |

(42, 43), (25, 27)

- Step – 3

| | 18 | 22 | 25, 27 | 42, 43 |
|---|---|---|---|---|
| 18 | 0 | 4 | 7 | 24 |
| 22 | 4 | 0 | 3 | 20 |
| 25, 27 | 7 | 3 | 0 | 15 |
| 42, 43 | 24 | 20 | 15 | 0 |

- Step – 3

| | 18 | 22 | 25, 27 | 42, 43 |
|---|---|---|---|---|
| 18 | 0 | 4 | 7 | 24 |
| 22 | 4 | 0 | 3 | 20 |
| 25, 27 | 7 | 3 | 0 | 15 |
| 42, 43 | 24 | 20 | 15 | 0 |

(42, 43), ( (25, 27), 22)

- Step – 4

| | 18 | 22, 25, 27 | 42, 43 |
|---|---|---|---|
| 18 | 0 | 4 | 24 |
| 22, 25, 27 | 4 | 0 | 15 |
| 42, 43 | 24 | 15 | 0 |

- Step – 4

| | 18 | 22, 25, 27 | 42, 43 |
|---|---|---|---|
| 18 | 0 | 4 | 24 |
| 22, 25, 27 | 4 | 0 | 15 |
| 42, 43 | 24 | 15 | 0 |

(42, 43), ( ( (25, 27), 22), 18)

- Step – 5

| | 18, 22, 25, 27 | 42, 43 |
|---|---|---|
| 18, 22, 25, 27 | 0 | 15 |
| 42, 43 | 15 | 0 |

# Final step

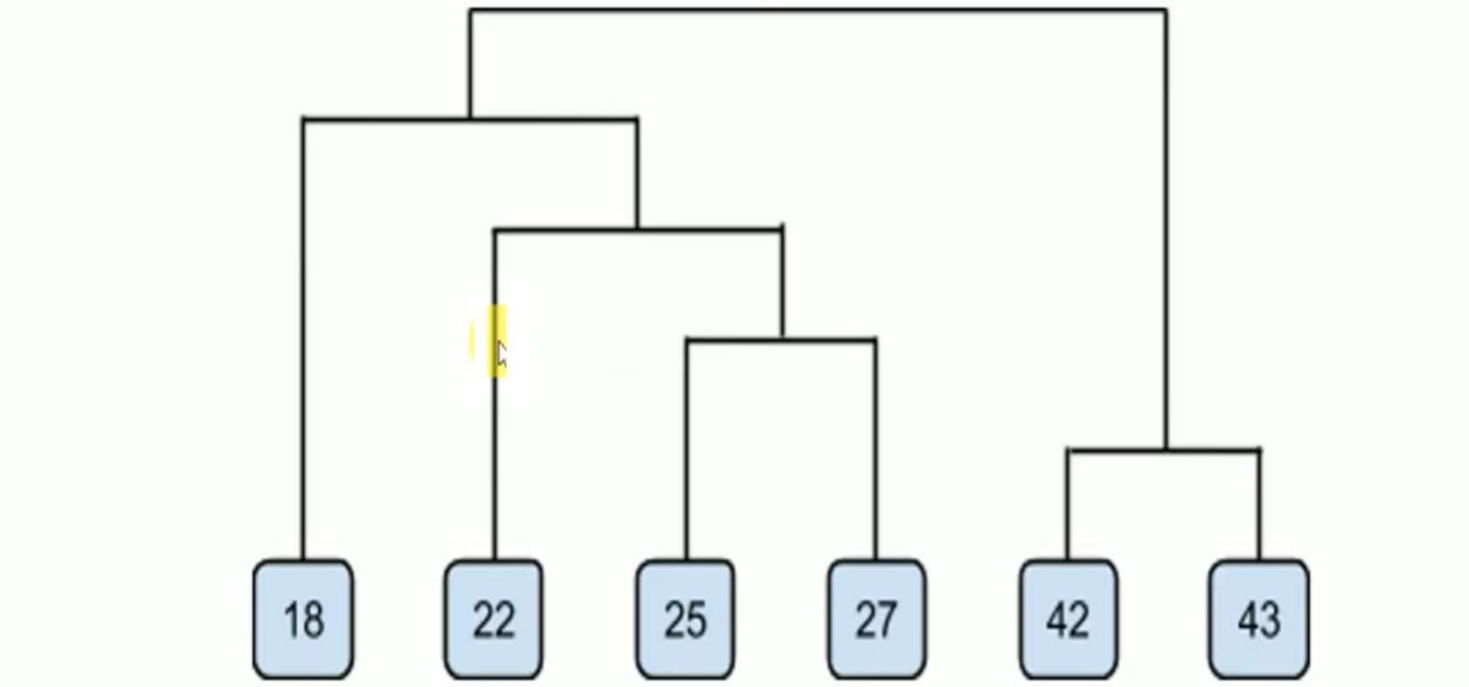
- Step – 6

| | |
|---|---|
| | 18, 22, 25, 27, 42, 43 |
| 18, 22, 25, 27, 42, 43 | 0 |

# Dendrogram

- Dendrogram

((42, 43), ( ( (25, 27), 22), 18) )




A dendrogram illustrates how similar objects are related to each other. The branches of the tree represent categories, or classes, of objects that share similar

# Association


**Association rule learning** is a type of unsupervised learning technique that checks for the dependency of one data item on another data item and maps accordingly

# Apriory

| Transaction ID | Items Bought |
|---|---|
| 1 | {Bread, Butter, Milk} |
| 2 | {Bread, Butter} |
| 3 | {Beer, Cookies, Diapers} |
| 4 | {Milk, Diapers, Bread, Butter} |
| 5 | {Beer, Diapers} |

Generate candidate itemsets (Ck) and qualified frequent itemsets (Lk) step by step until the largest frequent itemset is generated.

| 1-Itemset | Support_count |
|---|---|
| Bread | 3 |
| Butter | 3 |
| Milk | 2 |
| Beer | 2 |
| Cookies | 1 |
| Diapers | 3 |

Bread, Butter, Milk, Diapers, Beer

# Minimum Support Count


min_support = 40%,

min_support_count

= min_support x itemset_count

= 40% x 5

= 2

| 1-Frequent Itemset | Support_count |
|---|---|
| Bread | 3 |
| Butter | 3 |
| Diapers | 3 |
| Milk | 2 |
| Beer | 2 |

| T ID | Items Bought |
|---|---|
| 1 | {Bread, Butter, Milk} |
| 2 | {Bread, Butter} |
| 3 | {Beer, Cookies, Diapers} |
| 4 | {Milk, Diapers, Bread, Butter} |
| 5 | {Beer, Diapers} |

| 2-Itemset | Support_count |
|---|---|
| Bread, Butter | 3 |
| Bread, Diapers | 1 |
| Bread, Milk | 2 |
| Bread, Beer | 0 |
| Butter, Diapers | 1 |
| Butter, Milk | 2 |
| Butter, Beer | 0 |
| Diapers, Milk | 1 |
| Diapers, Beer | 2 |
| Milk, Beer | 0 |



| 2-Frequent Itemset | Support_count |
|---|---|
| Bread, Butter | 3 |
| Bread, Milk | 2 |
| Butter, Milk | 2 |
| Diapers, Beer | 2 |

| T ID | Items Bought |
|---|---|
| 1 | {Bread, Butter, Milk} |
| 2 | {Bread, Butter} |
| 3 | {Beer, Cookies, Diapers} |
| 4 | {Milk, Diapers, Bread, Butter} |
| 5 | {Beer, Diapers} |

Bread, Butter, Milk, Diapers, Beer

| 3-Itemset | Support_count |
|---|---|
| Bread, Butter, Milk | 2 |
| Bread, Butter, Diapers | 1 |
| Bread, Butter, Beer | 0 |
| Bread, Milk, Diapers | 1 |
| Bread, Milk, Beer | 0 |
| Bread, Diapers, Beer | 0 |
| Butter, Milk, Diapers | 1 |
| Butter, Milk, Beer | 0 |
| Butter, Diapers, Beer | 0 |
| Milk, Diapers, Beer | 0 |


| 3-Frequent Itemset | Support_count |
|---|---|
| Bread, Butter, Milk | 2 |

| 1-Frequent Itemset | Support_count |
|---|---|
| Bread | 3 |
| Butter | 3 |
| Diapers | 3 |
| Milk | 2 |
| Beer | 2 |

| 2-Frequent Itemset | Support_count |
|---|---|
| Bread, Butter | 3 |
| Bread, Milk | 2 |
| Butter, Milk | 2 |
| Diapers, Beer | 2 |

| 3-Frequent Itemset | Support_count |
|---|---|
| Bread, Butter, Milk | 2 |

- **min_confidence = 70%**
- Confidence (X→Y) = P(Y | X) = P(X ∪ Y) / P(X)
- We have 5 frequent itemsets:
- {Bread, Butter}, {Bread, Milk}, {Butter, Milk}, {Diapers, Beer} and {Bread, Butter, Milk}.
- Therefore, candidate rules are:
- For {Bread, Butter},
  - bread>butter = 3/3 = 100% (Strong)
  - butter>bread = 3/3 = 100% (Strong)
- For {Bread, Milk}
  - bread>milk = 2/3 = 67%
  - milk>bread = 2/2 =100% (Strong)

- For {Bread, Butter, Milk}
  - bread,butter>milk = 2/3 = 67% ✗
  - bread,milk>butter = 2/2 = 100% (Strong)
  - milk,butter>bread = 2/2 = 100% (Strong)
  - bread>butter,milk = 2/3 = 67% ✗
  - butter>bread,milk = 2/3 = 67% ✗
  - milk>bread,butter = 2/2=100% (Strong)

- Therefore, candidate rules are:
- For {Butter, Milk}
  - butter>milk = 2/3 = 67% ✗
  - milk>butter = 2/2 = 100% (Strong)

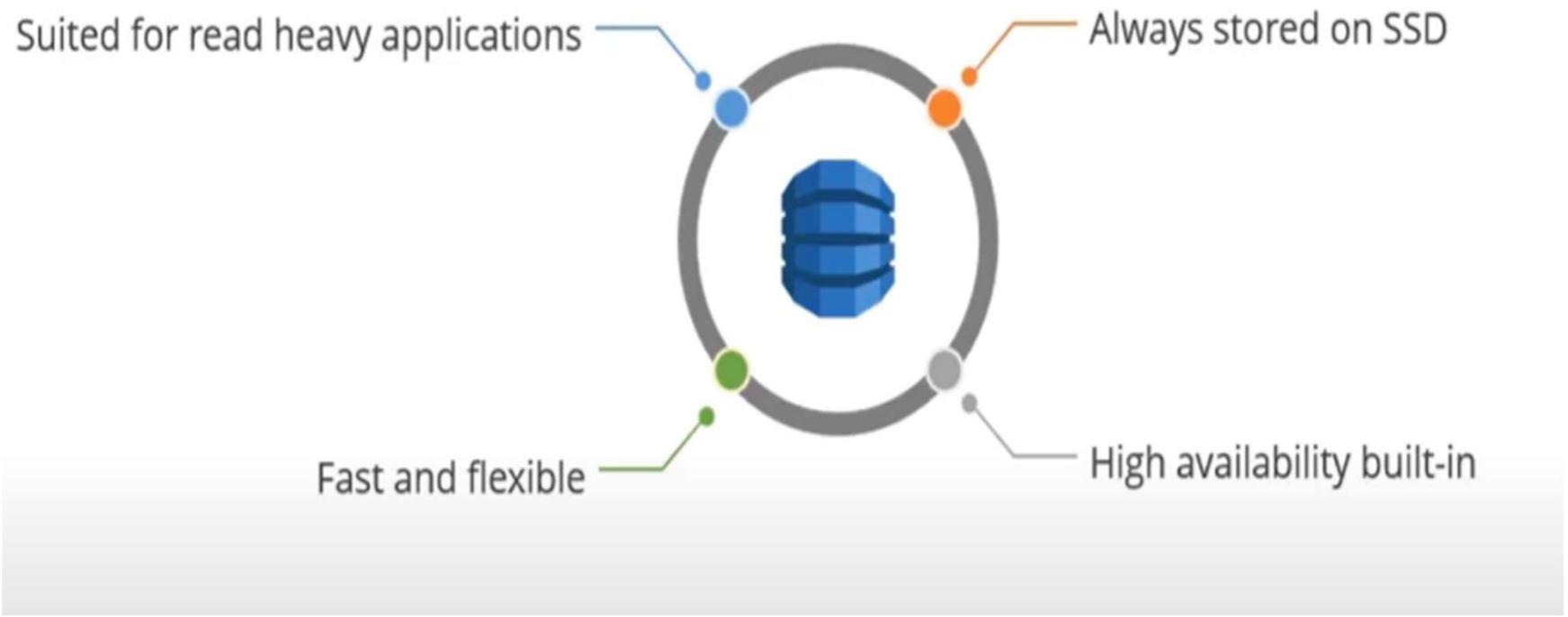
# Why DynamoDB  Why?


- ✓ DynamoDB is a key:value store of the NoSQL family developed and offered by Amazon as part of AWS
- ✓ High Scale, High Performance & Fully Managed DB Service
- ✓ Accessible via Web Service APIs
- ✓ Provides speed, Scalability & Ease of use
- ✓ Takes care of hardware or software provisioning, setup and configuration, software patching, operating a reliable, distributed database cluster, or partitioning data over multiple instances as you scale.

# Features


- ✓ Scalable
- ✓ Fast, Predictable Performance
- ✓ Easy Administration
- ✓ Built-in Fault Tolerance
- ✓ Flexible
- ✓ Strong Consistency, Atomic Counters
- ✓ Integrated Monitoring
- ✓ Elastic MapReduce Integration
- ✓ Secure

# Benefits


Suited for read heavy applications
Always stored on SSD
Fast and flexible
High availability built-in

# Concept Of DynamoDB


- ✓ **The Amazon DynamoDB data model concepts include tables, items and attributes.**
- ✓ **In Amazon DynamoDB, a database is a collection of tables. A table is a collection of items and each item is a collection of attribute**
- ✓ **DynamoDB allows several "tables", where a record ("Item"), is identified by one or two "Attributes", named:**
  - ✓ A "Hash Key", which works as a Primary Key
  - ✓ Optionally, a "Range Key", which lets you build Composite Keys
- ✓ Beside the Key Attributes, everything else is **unstructured**.
- ✓ **For the Keys, there are three data types:** String, Binary, Number (anything with up to 38 significant digits and between 10^-128 and 10^126).
- ✓ You can also have a "**Set" datatype** (for String, Binary, and Numbers), though they are not indexed.

# Tables & Forums with Unique Subject


| Table Name | Primary Key Type | Hash Attribute Name and Type | Range Attribute Name and Type |
|---|---|---|---|
| ProductCatalog (Id, ... ) | Hash | Attribute Name: Id<br>Type: Number | - |
| Forum ( Name, ... ) | Hash | Attribute Name: Name<br>Type: String | - |
| Thread (ForumName, Subject, ... ) | Hash and Range | Attribute Name: ForumName<br>Type: String | Attribute Name: Subject<br>Type: String |
| Reply ( Id, ReplyDateTime, ... ) | Hash and Range | Attribute Name: Id<br>Type: String | Attribute Name: ReplyDateTime<br>Type: String |

- ✓ The ProductCatalog table represents a table in which each product item is uniquely identified by an Id
- ✓ The Forum, Thread, and Reply tables are modeled after the AWS forums. Each AWS service maintains one or more forums. Customers start a thread by posting a message that has a unique subject. Each thread might receive one or more replies at different times. These replies are stored in the Reply table

# DynamoDB Concept

- ✓ **The following operations are possible:**
  - ✓ PutItem / GetItem / DeleteItem, where you DynamoDB Record (or "Item") directly, provided you have the HashKey (and Range Key). Except the latter, there are "Batch" variants available
- ✓ **Query**, where you can look up Items via Conditions based on the Primary Key Attributes.
- ✓ **Scan**, where a whole table is scanned.

Explore Table: testEdureka

List Tables | Browse Items | Item Details

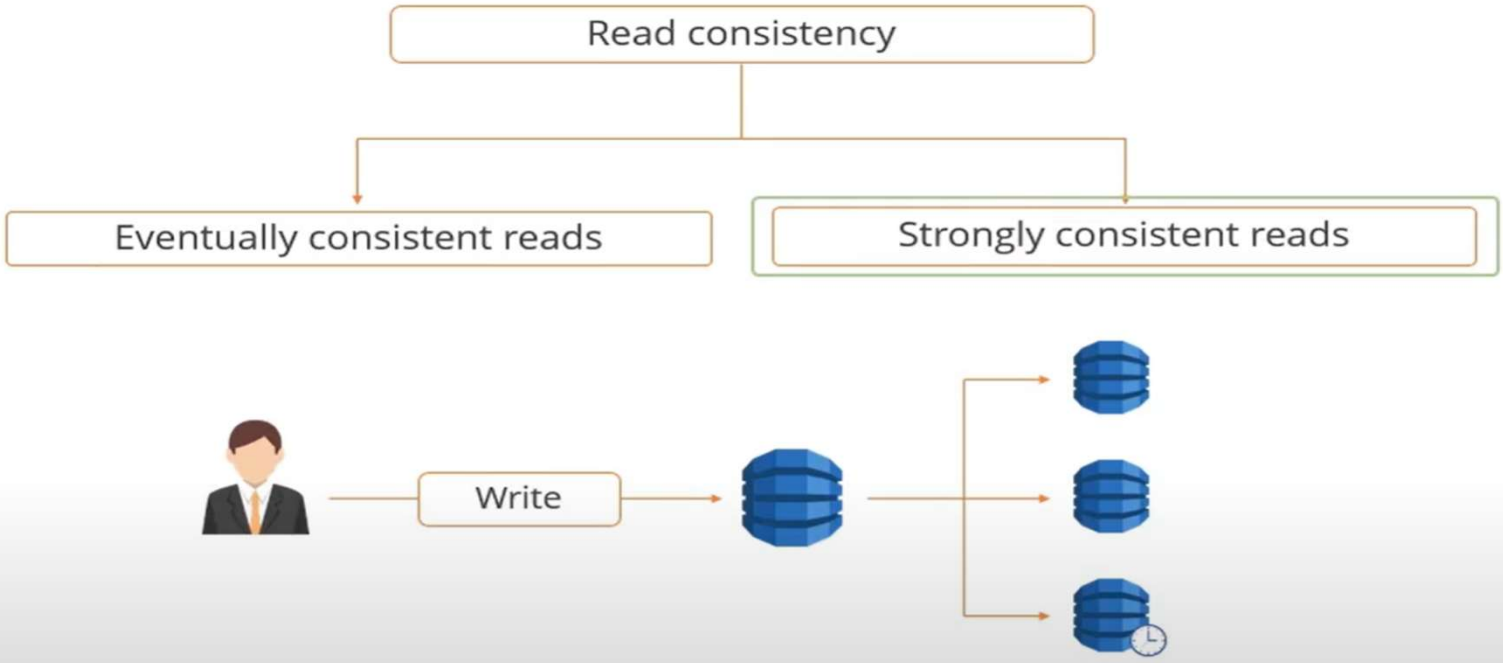
Scan / Get | Go | New Item | 1 to 3 of 3 loaded items

| trainingID | Course | Date | Duration | Faculty | Modules | Name | Organization | Trainer Name |
|---|---|---|---|---|---|---|---|---|
| "2" | ".Net" | | | | | "Amazo" | | |
| "1" | "AWS" | "5th August" | | | | | | "Tarun" |
| "5" | "Hadoop" | | "5 Hours" | "Gaurav" | "10" | | "Edureka" | |

# DynamoDB Capacity to Manage…


- ✓ **Hardware provisioning**
- ✓ **Cross-availability zone replication**
- ✓ **Monitoring and handling of hardware failures**
  - ✓ Replicas automatically regenerated whenever necessary
- ✓ **Changing the level of provisioned throughput**
  - ✓ Data might need to be redistributed around the cluster
  - ✓ No service disruption or performance impact

# Read Consistency




Eventually consistent reads can throw "dirty reads" results at times, which means you request a read but do not receive the up-to-date version.

# DynamoDB Modes


## DynamoDB Provisioned Mode

You define some capacity, and DynamoDB provisions that capacity for you. This is pretty similar to provisioning an Auto Scaling Group of EC2 instances, but imagine the size of the instance is fixed, and it's one group for reads and another one for writes. Here's how that capacity translates into actual read and write operations.

### Capacity in Provisioned Mode

Capacity is provisioned separately for reads and writes, and it's measured in Capacity Units.

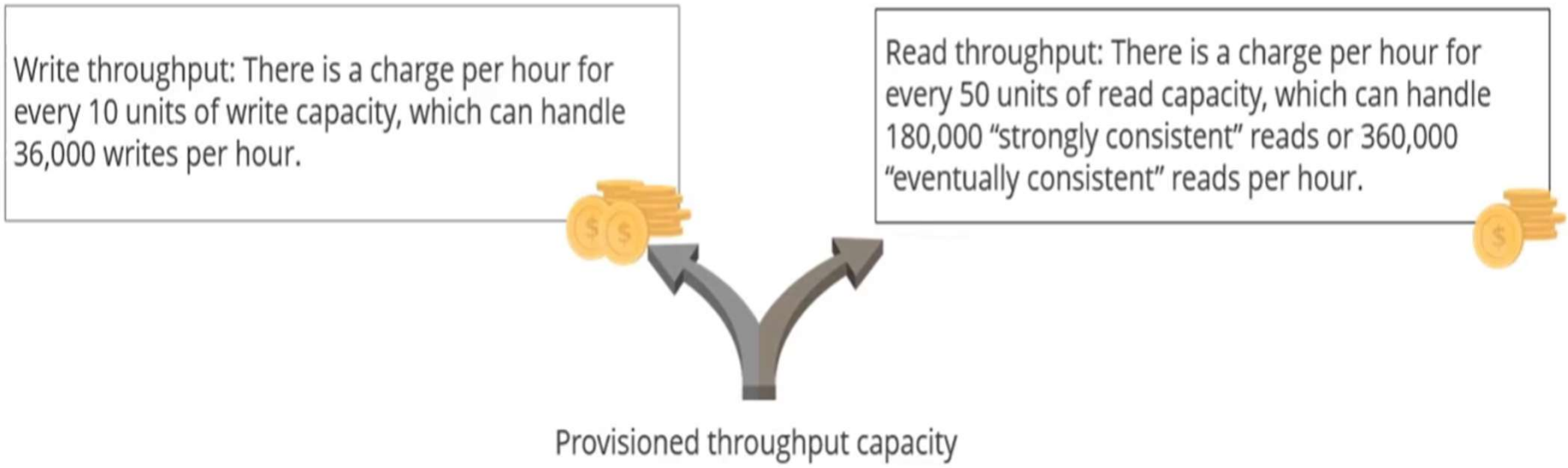
**1 Read Capacity Unit (RCU) is equivalent to 1 strongly consistent read of up to 4 KB, per second**. Eventually consistent reads consume half that capacity. Reads over 4 KB consume 1 RCU (1/2 for eventually consistent) per 4 KB, rounded up. That means if you have 5 RCUs, you can perform 10 eventually consistent reads every second, or 2 strongly consistent reads for 7 KB of data each (remember it's rounded up) plus 1 strongly consistent read for 1 KB of data (again, it's rounded up).

Write Capacity Units (WCU) work the same, but for writes. **1 WCU = 1 write per second, of up to 1 KB**. So, with 5 WCUs, you can perform 1 write operation per second of 4.5 KB, or 5 writes of less than 1 KB.

# DynamoDB Read & Write


- ✓ A unit of Write Capacity enables you to perform one write per second for items of up to 1Kb in size
- ✓ a unit of Read Capacity enables you to perform one strongly consistent read per second (or two eventually consistent reads per second) of items of up to 1Kb in size. Larger items will require more capacity.
  - Units of Capacity required for writes = Number of item writes per second x item size (rounded up to the nearest KB)
  - Units of Capacity required for reads* = Number of item reads per second x item size (rounded up to the nearest KB)
- ✓ If your items are less than 1KB in size, then each unit of Read Capacity will give you 1 read/second of capacity and each unit of Write Capacity will give you 1 write/second of capacity.
  - ✓ For example, if your items are 512 bytes and you need to read 100 items per second from your table, then you need to provision 100 units of Read Capacity.
- ✓ If your items are larger than 1KB in size, then you should calculate the number of units of Read Capacity and Write Capacity that you need.

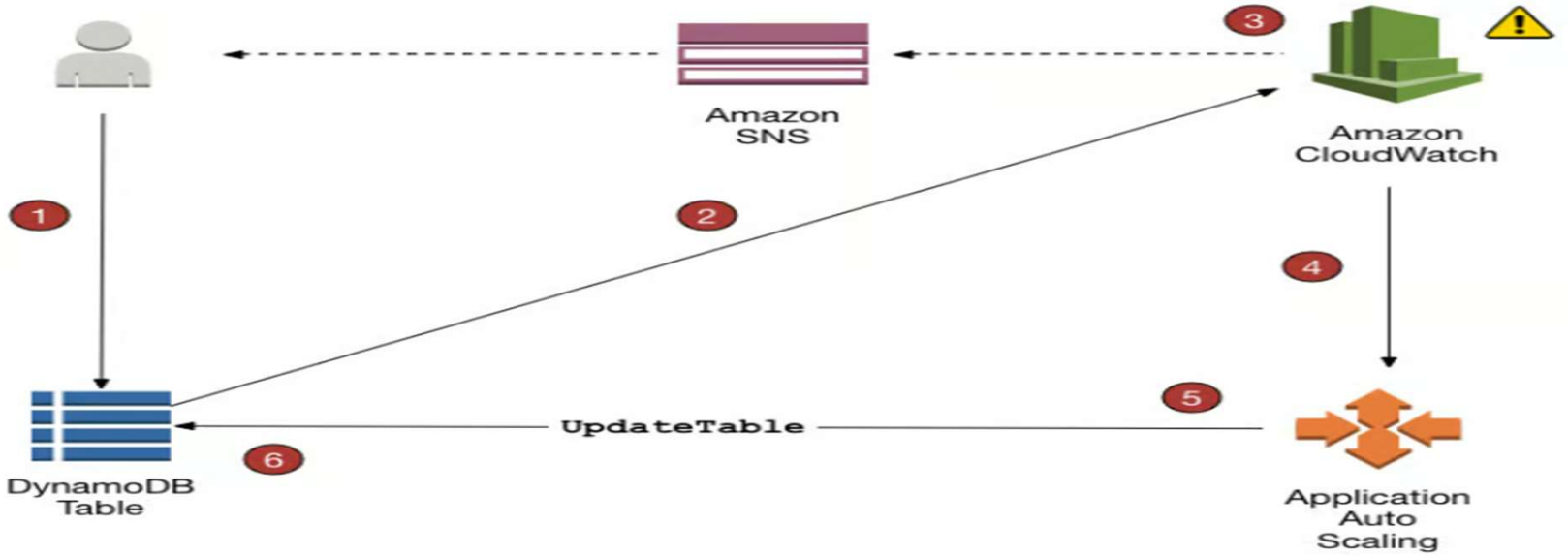
# Provisioned Throughput Capacity



# Scaling Provision Mode

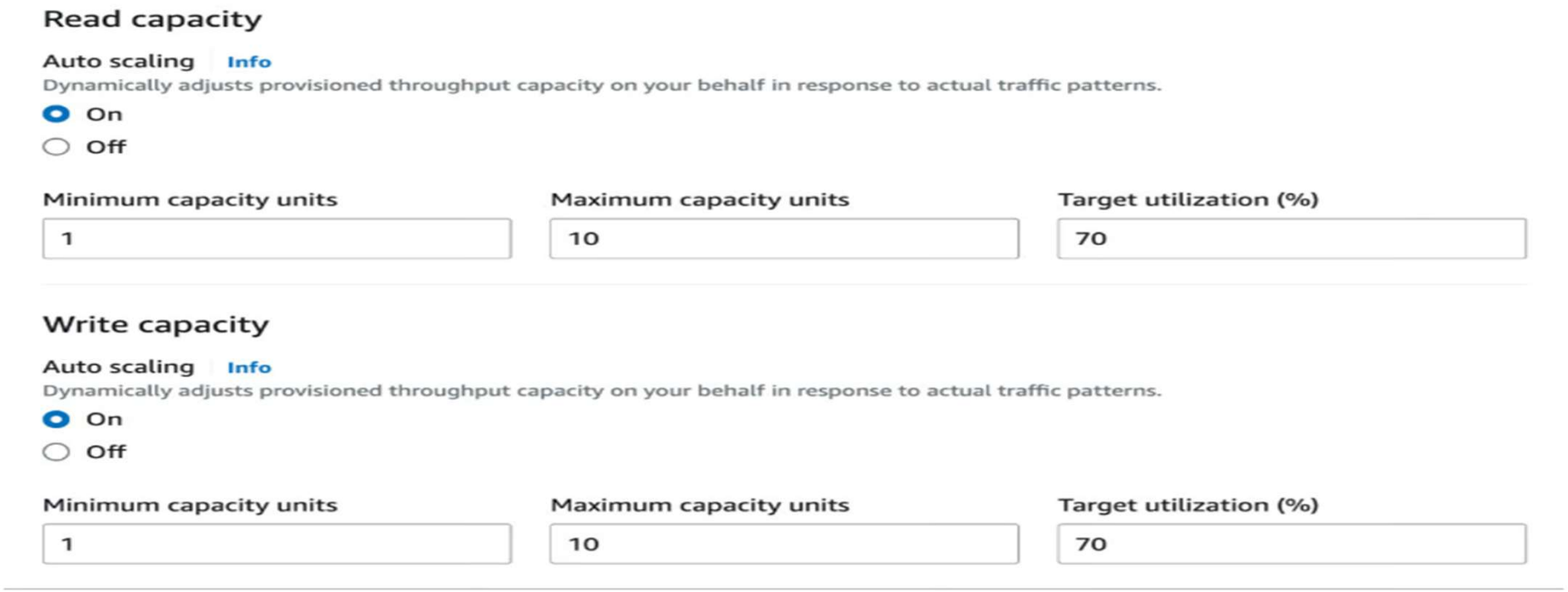



This is the real scaling. DynamoDB tables continuously send metrics to CloudWatch, CloudWatch triggers alarms when those metrics cross a certain threshold, DynamoDB gets notified about that and modifies Capacity Units accordingly.




On DynamoDB you enable Auto Scaling, set a minimum and maximum capacity units, and set a target utilization (%). You can enable scaling separately for Reads and Writes.

# Table Capacity


In the table metrics (handled by CloudWatch) you can view provisioned and consumed capacity, and throttled request count.

# Capacity in On-Demand Mode


The cost of reads and writes stays the same: a read operation consumes 1 Read Request Unit (RRU) for every 4 KB read (half if it's eventually consistent), and a write operation consumes 1 WRU (Write Request Unit) for every 1 KB written.

Here's the difference: There is no capacity you can set. You're billed for every actual operation, and DynamoDB manages capacity automatically and transparently. However, it does have a set capacity, it does scale, and understanding how it does is important.

# Scaling in On-Demand Mode


Every newly-created table in On-Demand mode starts with 4.000 WCUs and 12.000 RCUs (yeah, that's a lot). You're not billed for those capacity units though; you'll only be billed for actual operations.

Every time your peak usage goes over 50% of the current assigned capacity, DynamoDB increases the capacity of that table to double your peak. So, suppose you used 5.000 WRUs, now your table's WCUs are 10.000. This growth has a cooldown period of 30 minutes, meaning it won't happen again until 30 minutes after the last increase.

**WCU stands for** Write Capacity Unit, which is a unit of measurement for write requests to a table in Amazon DynamoDB. One WCU allows for one write request per second for items up to 1 KB in size. For items larger than 1 KB, additional WCUs are required

RCU stands for "Read Capacity Unit". It represents the number of times per second that data can be read from a table

# Switching from Provisioned Mode to On-Demand Mode


You can switch modes in either direction, but you can only do so once every 24 hours. If you switch from Provisioned Mode to On-Demand mode, the table's initial RCUs are the maximum of 12.000, your current RCUs, or double the units of the highest peak. Same for WCUs, the maximum between 4.000, your current WCUs, or double the units of the highest peak.

If you switch from On-Demand mode to Provisioned Mode, you need to set up your capacity or auto scaling manually.

In either case the switch takes up to 30 minutes, during which the table continues to function like before the switch.

# That is all


Please choose

https://www.tutorialspoint.com/dynamodb/index.htm

To learn Practical operations