# Supervised Learning

Unit #2 & 3

Marwadi University

Department of
Computer Engineering

**Machine Learning**
Sem 7
01CE0715
4 Credits

Prof. Urvi Bhatt

# Course Outcomes

After completion of this course, students will be able to

- Understand machine-learning concepts.
- Understand and implement Classification concepts.
- Understand and analyse the different Regression algorithms.
- Apply the concept of Unsupervised Learning.
- Apply the concepts of Artificial Neural Networks.

# Topics - Supervised Learning

**Classification Techniques:**

- Naive Bayes Classification
- Fitting Multivariate Bernoulli Distribution
- Gaussian Distribution and Multinomial Distribution
- K- Nearest Neighbours
- Decision tree
- Random Forest
- Ensemble Learning
- Support Vector Machines
- Evaluation metrics for Classification Techniques: Confusion Matrix, Accuracy, Precision, Recall, F1 Score, Threshold, AUC-ROC

**Regression Techniques:**

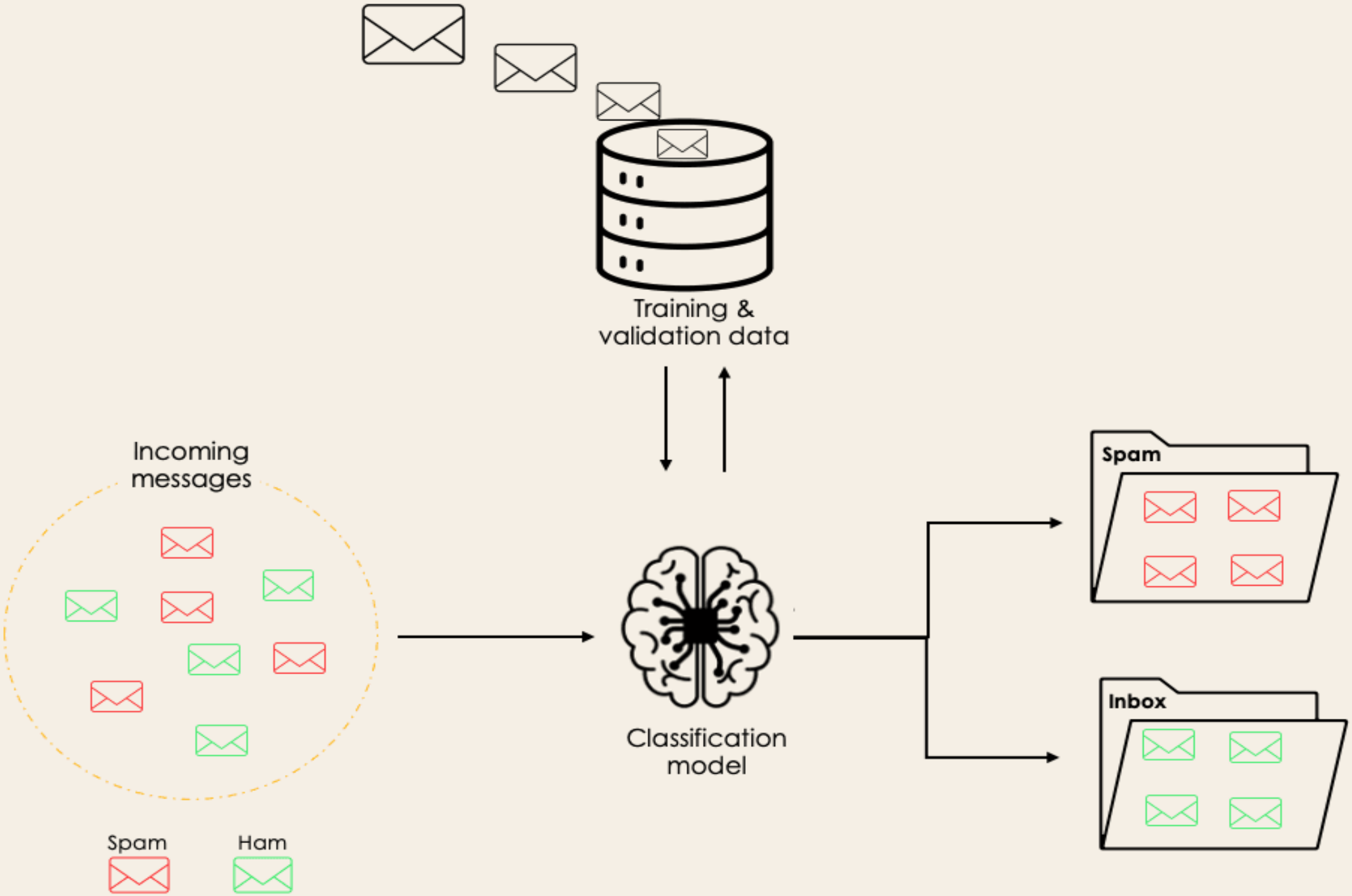
- Basic concepts and applications of Regression
- Simple Linear Regression - Gradient Descent and Normal Equation Method
- Multiple Linear Regression
- Non-Linear Regression
- Linear Regression with Regularization
- Overfitting and Underfitting
- Hyperparameter tuning
- Evaluation Measures for Regression Techniques: MSE, RMSE, MAE, R2

# Classification Techniques

Introduction, Applications

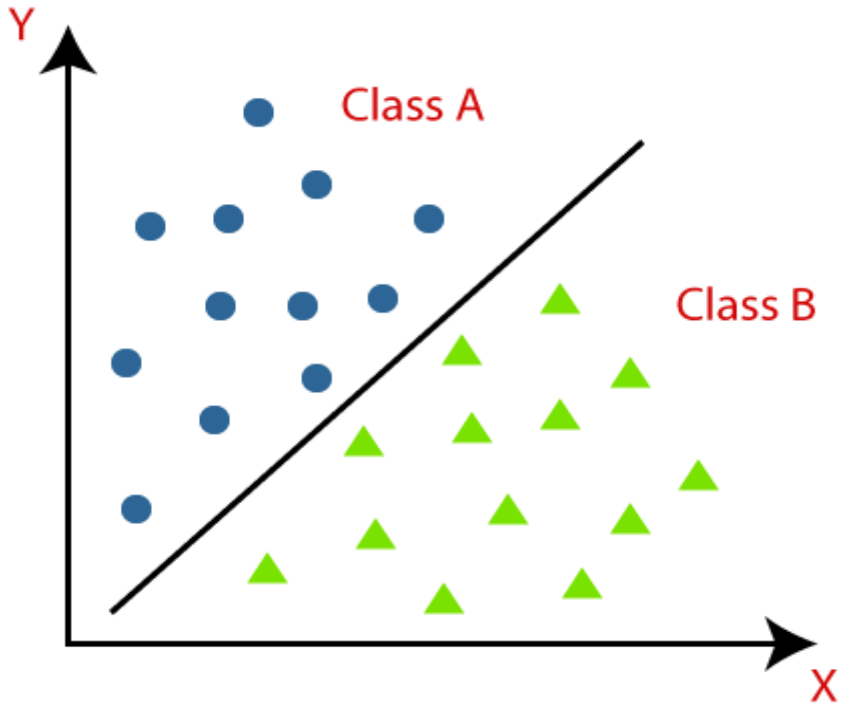
# What is Classification

- Classification is a supervised machine learning process that predicts the class of input data based on the algorithms training data
- Classification is the process of predicting the class of given data points.
- Classification is a supervised machine learning method where the model tries to predict the correct label of a given input data.
- In classification, the model is fully trained using the training data, and then it is evaluated on test data before being used to perform prediction on new unseen data.
- For example, a spam detection machine learning algorithm would aim to classify emails as either “spam” or “not spam.”


Training &
validation data
Incoming
messages
Spam
Ham
Classification
model
Spam
Inbox

# Contd.

- Classes are sometimes called targets, labels or categories.
- Classification predictive modeling is the task of approximating a mapping function (f) from input variables (X) to discrete output variables (y.)

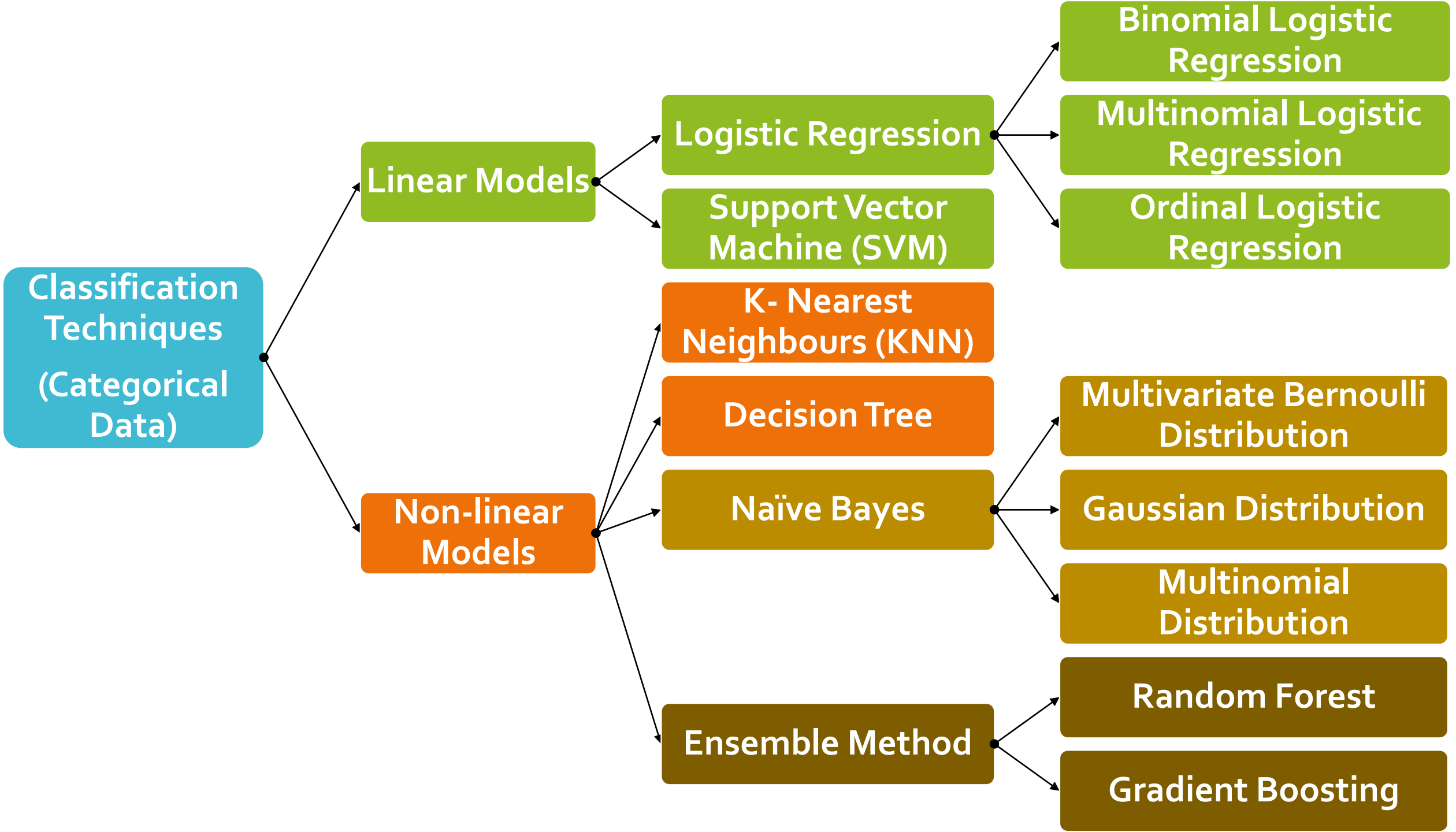


# Applications of Classification

- **Image Recognition:**
  - object detection, facial recognition, and autonomous driving.
- **Object Detection in Autonomous Vehicles:**
  - autonomous vehicles Detection , enabling them to identify and respond to pedestrians, traffic signs, and other vehicles.
- **Face Recognition:**
  - identify and authenticate individuals based on facial features, finding applications in security systems, access control, and surveillance.
- **Disease Diagnosis:**
  - classify diseases or predict the likelihood of certain conditions, assisting in medical diagnosis.
- **Land Cover Classification in Remote Sensing:**
  - classify land cover types (e.g., forests, urban areas, water bodies) in satellite or aerial imagery, aiding in environmental monitoring, urban planning, and natural resource management.
- **Voice Recognition:**
  - classify spoken words or phrases, enabling applications like voice assistants, transcription services, and speaker identification.
- **Language Identification:**
  - classify text data into different languages, aiding in language identification tasks, multilingual analysis, and machine translation.
- **Sentiment Analysis:**
  - classify text data (e.g., customer reviews, social media posts) to determine sentiment (positive, negative, neutral) and understand public opinion and brand perception.
- **Email Spam Filtering:**
  - classify emails as either spam or non-spam, helping in filtering unwanted or malicious emails
- **Toxic Comment Classification:**
  - classify text comments as toxic or non-toxic, helping to identify and moderate harmful or abusive content on online platforms.

# Applications of Classification

- **Handwriting Recognition:** .
  - classify handwritten characters or text, finding applications in optical character recognition (OCR) systems and digitizing handwritten documents
- **Document Classification:**
  - categorize documents, such as news articles, legal documents, or customer support tickets, into relevant categories, facilitating efficient document retrieval and organization
- **Social Media Text Classification:**
  - extract valuable insights from social media data.
- **Predicting Loan Defaults:**
  - loan defaults, assisting financial institutions in managing risk and making informed lending decisions
- **Fraud Detection:**
  - identify fraudulent transactions or activities, playing a critical role in financial institutions, e-commerce platforms, and security systems
- **Stock Market Prediction:**
  - classify stocks as buy, sell, or hold based on historical market data and indicators
- **Recommendation Systems:**
  - predict user preferences and classify items or content to provide personalized recommendations, improving user experience in e-commerce, streaming platforms, and content curation
- **Intrusion Detection in Cybersecurity:**
  - classify network intrusions and cyber threats, distinguishing between normal network traffic and malicious activities. classify emails as either spam or non-spam, helping in filtering unwanted or malicious emails


Classification Techniques
(Categorical Data)
Linear Models
Logistic Regression
Binomial Logistic Regression
Multinomial Logistic Regression
Ordinal Logistic Regression
Support Vector Machine (SVM)
Non-linear Models
K- Nearest Neighbours (KNN)
Decision Tree
Naïve Bayes
Multivariate Bernoulli Distribution
Gaussian Distribution
Multinomial Distribution
Ensemble Method
Random Forest
Gradient Boosting

# Learners in Classification Problems:

In the classification problems, there are two types of learners:

- **Lazy Learners:**
  - Lazy Learner firstly stores the training dataset and wait until it receives the test dataset.
  - In Lazy learner case, classification is done on the basis of the most related data stored in the training dataset.
  - It takes less time in training but more time for predictions.
  - **Example:** K-NN algorithm, Case-based reasoning
- **Eager Learners:**
  - Eager Learners develop a classification model based on a training dataset before receiving a test dataset.
  - Opposite to Lazy learners, Eager Learner takes more time in learning, and less time in prediction.
  - **Example:** Logistic Regression, Support Vector Machine, Decision Trees, Naïve Bayes, ANN.

# Lazy learners VS Eager Learners

| Feature | Lazy Learners | Eager Learners |
|---|---|---|
| When they learn | Learn at prediction time | Learn at training time |
| Storage | Store all training data | Store a summary/model of data |
| Training speed | Fast | Slow |
| Prediction speed | Slow (must process all data) | Fast (uses pre-built model) |
| Memory use | High (stores all data) | Lower (stores only model) |
| Flexibility | High (can adapt easily to new data) | Lower (model is fixed once built) |
| Examples | k-Nearest Neighbors (k-NN) | Decision Trees, Naive Bayes |
| Advantages | Simple, efficient training | Fast prediction, efficient for large datasets |
| | Flexible with model complexity | Less memory required during prediction |
| Disadvantages | Slow, memory-intensive predictions | Slow training, less flexible after training |

# Logistic Regression

Introduction, Type of Logistic Regression, Sigmoid Function, Example, Advantage & Disadvantage

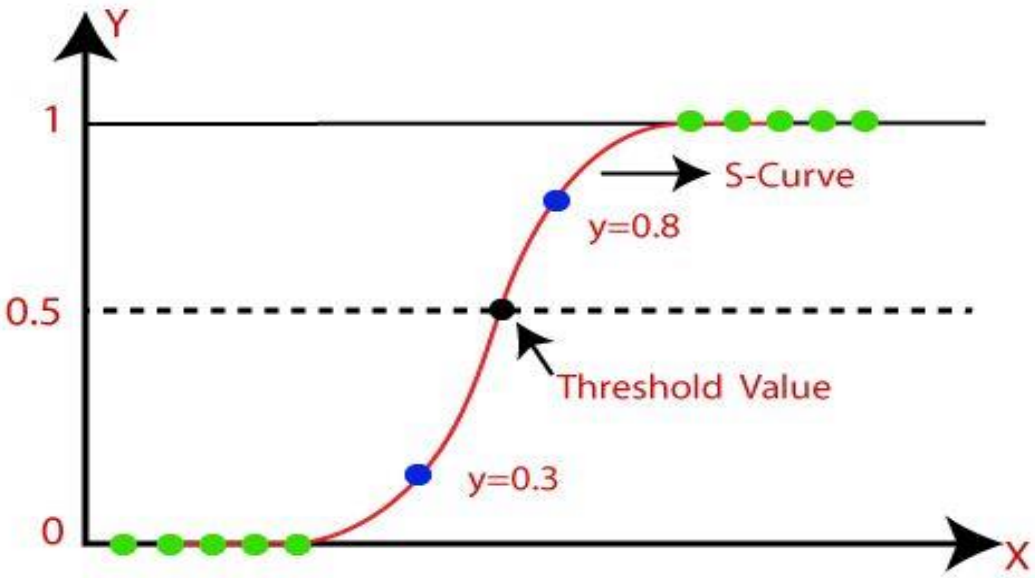
# Logistic Regression

- Logistic regression is one of the most popular Machine Learning algorithms, which comes under the supervised Learning technique.
- It is used for predicting the categorical dependent variable using a given set of independent variables.
- Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value.
- Logistic regression is a calculation used to predict a binary outcome: either something happens, or does not. This can be exhibited as Yes/No, Pass/Fail, Alive/Dead, etc.
- It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, **it gives the probabilistic values which lie between 0 and 1**.

# Logistic Regression

- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1)
- The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc

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



# Logistic Function (Sigmoid Function)

- The sigmoid function is a mathematical function used to map the predicted values to probabilities.
- It maps any real value into another value within a range of 0 and 1.
- The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the Sigmoid function or the logistic function.
- In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

# Type of Logistic Regression

On the basis of the categories, Logistic Regression can be classified into three types:

1. **Binomial:** In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.
2. **Multinomial:** In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep".
3. **Ordinal:** In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

# Contd.

| Feature | Binomial Logistic Regression | Multinomial Logistic Regression | Ordinal Logistic Regression |
|---|---|---|---|
| Dependent Variable Types | Two possible outcomes | Three or more unordered categories | Three or more ordered categories |
| Examples | 0 or 1, Pass or Fail | "cat", "dog", "sheep" | "low", "medium", "high" |
| Use Case | Binary classification problems | Multiclass classification problems | Ordered classification problems |
| Model Complexity | Simpler | More complex | More complex |
| Order of Categories | No order | No order | Ordered |

# DEMO

---

# Logistic Regression

- https://colab.research.google.com/drive/1Z504ZYibatAsE2_WlHzobJyboSiy4lUk?usp=sharing

# Example:

- 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 |

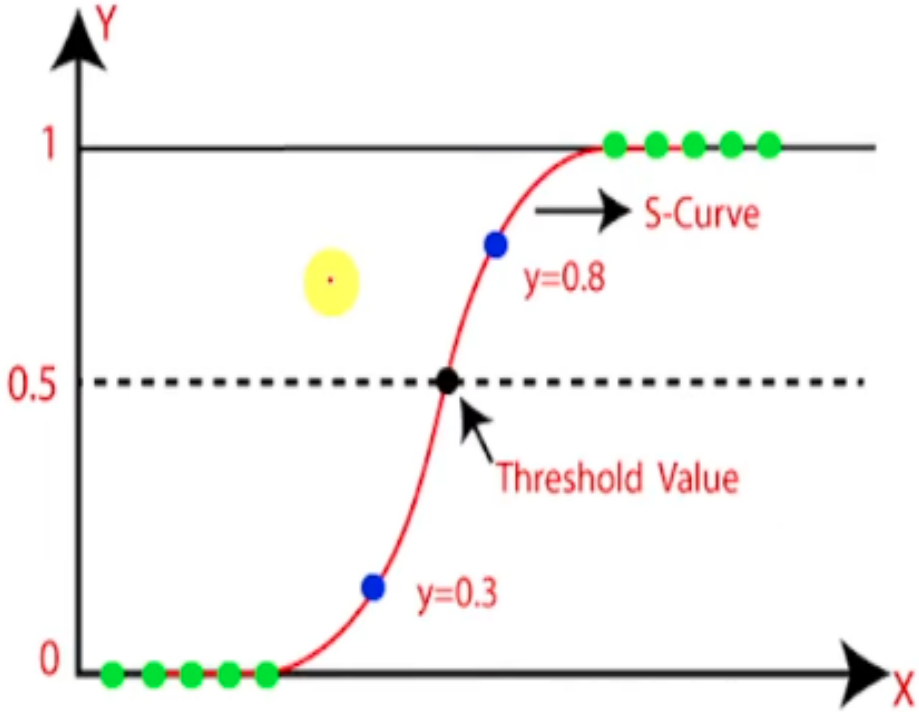
Assume the model suggested by the optimizer for odds of passing the course is,

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

# Contd.

- We use Sigmoid Function in logistic regression

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



# Contd.

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 |

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

# Contd.

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 |

- $p = \frac{1}{1+e^{-z}} = 0.95$
- $0.95 * (1 + e^{-z}) = 1$
- $0.95 * e^{-z} = 1 - 0.95$
- $e^{-z} = \frac{0.05}{0.95} = 0.0526$
- $\ln(e^{-z}) = \ln(0.0526)$

$\ln(e^x) = x$

$-z = ln(0.0526) = -2.94$

# Contd.

- $z = 2.94$
- $\log(odds) = z = -64 + 2 * hours$
- $2.94 = -64 + 2 * hours$
- $2 * hours = 2.94 + 64$
- $2 * hours = 66.94$
- $hours = \frac{66.94}{2}$
- $\boldsymbol{hours = 33.47\ Hours}$

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

- The student should study **at least 33.47 hours**, so that he will pass the exam with more than 95% probability

# Example:

- Simple example with one feature X and o target variable y. Here's dataset:

| X | Y |
|---|---|
| 2 | 0 |
| 3 | 0 |
| 5 | 1 |
| 6 | 1 |

- $z=-7+2\cdot X$

Apply Logistic Regression to answer the following Questions

Q- 1: X=4, the predicted probability of Y.

Q-2:At least which value of x is required to get Y 89%

# Contd.

- Ans:
- Q-1: X=4, the predicted probability Y: 0.7311
- Q-2:At least which value of x is required to get Y 89%

  X≈4.5454

# Example:

| X | Y |
|---|---|
| 2 | 0 |
| 3 | 0 |
| 5 | 1 |
| 6 | 1 |

Given a logistic regression model defined by:

$z = -5 + 1.5 \cdot X$

- Let's answer the following questions:
- Q-1: What is the predicted probability of Y for X=6?
- Q-2: At least what value of X is required to get Y with 95% probability?

# Ans:

- Q-1: So, the predicted probability of Y when X=6 is approximately 0.982 (or 98.2%).
- Q-2: $X$ needs to be at least approximately 5.296 to get a probability of $Y$ being 1 at 95% or higher.

# Advantages and Disadvantages of Logistic Regression

**Advantages of Logistic Regression Algorithm:**

- Logistic regression is easier to implement, interpret and very efficient to train.
- Logistic Regression performs well when the **dataset is linearly separable**.
- Logistic Regression not only gives a measure of how relevant a predictor (coefficient size) is, but also its direction of association (positive or negative).

**Disadvantages of Logistic Regression Algorithm:**

- If the number of observations are lesser than the number of features, Logistic Regression should not be used, otherwise it may lead to overfit.
- Main limitation of Logistic Regression is the **assumption of linearity** between the dependent variable and the independent variables. In the real world, the data is rarely linearly separable..

# Support Vector Machine (SVM)

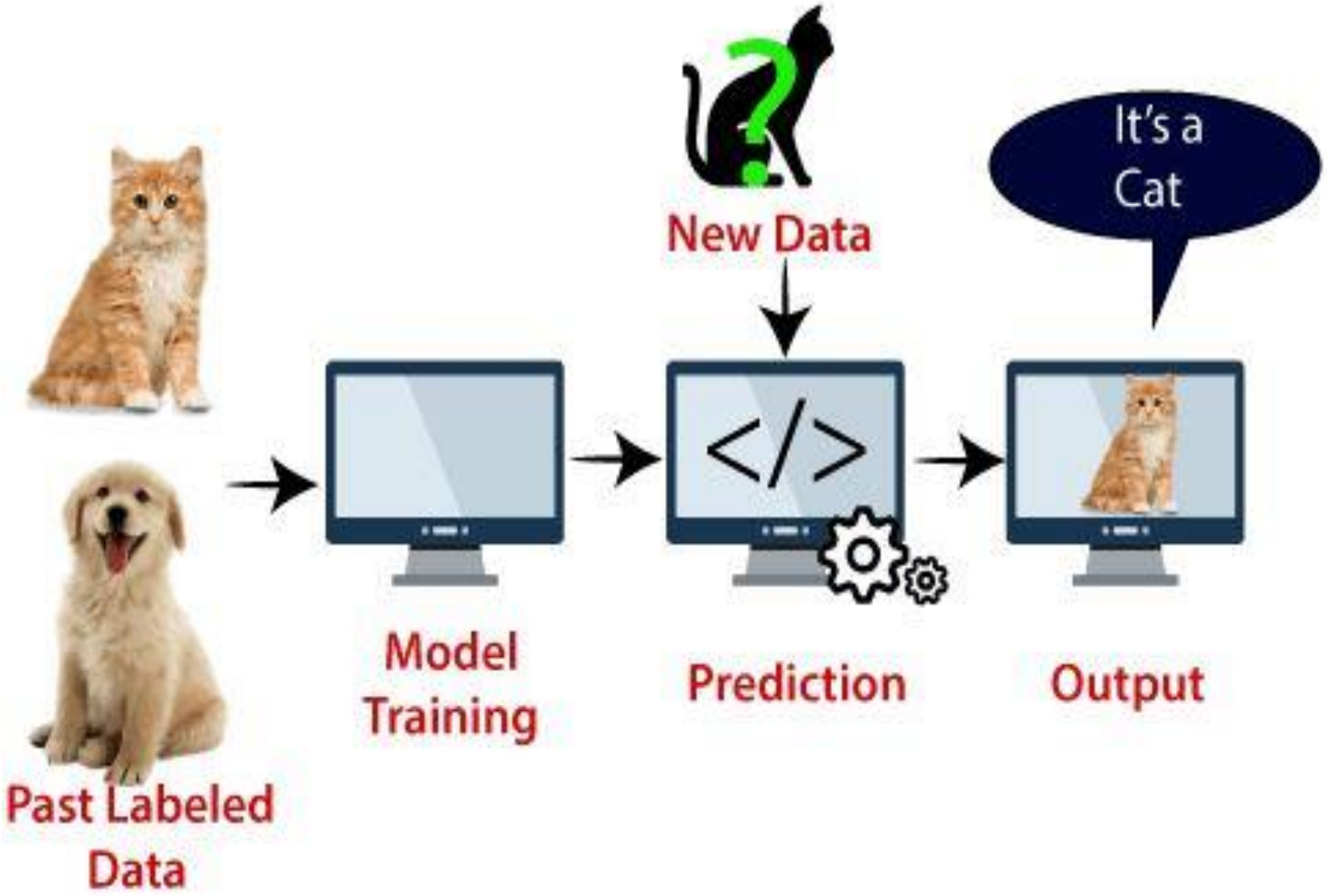
Introduction, Type of SVM, Different Kernel Functions, Advantage & Disadvantage

# SVM

- SVM is a powerful supervised algorithm that works **best on smaller datasets but on complex ones**.
- Support Vector Machine, abbreviated as SVM can be **used for both regression and classification** tasks, but generally, they **work best in classification** problems.
- It is a supervised machine learning problem where we **try to find a hyperplane that best separates the two classes**.
- SVM algorithm can be used for **Face detection, image classification, text categorization,** etc.

# Contd.

- Example:



# Contd.

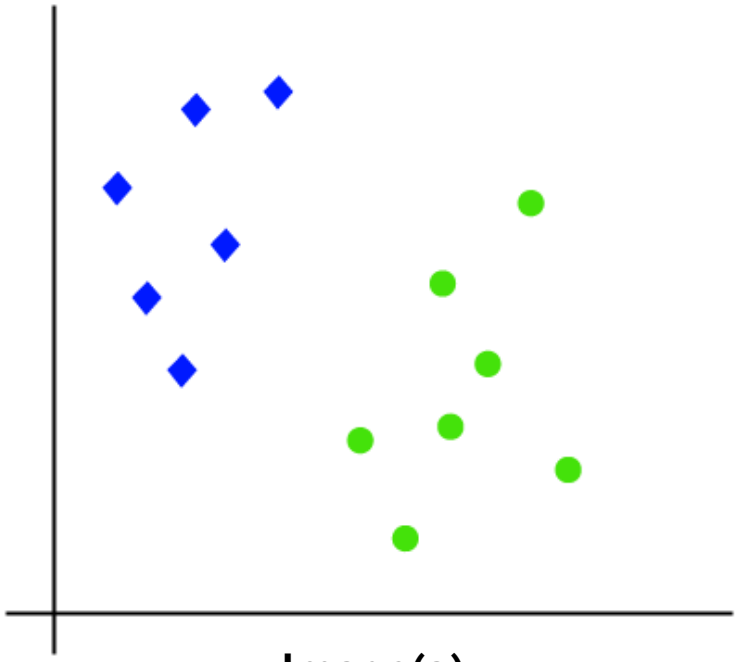
**Note:** Don't get confused between SVM and logistic regression.

- Both the algorithms try to find the best hyperplane, but the main difference is **logistic regression is a probabilistic approach** whereas **support vector machine is based on statistical approaches.**
- SVM works best when the dataset is small and complex. It is usually **advisable to first use logistic regression** and see how does it performs, **if it fails to give a good accuracy you can go for SVM** without any kernel

# Types of Support Vector Machine (SVM) Algorithms

- **Linear SVM**: When the data is perfectly linearly separable only then we can use Linear SVM. Perfectly linearly separable means that the data points can be classified into 2 classes by using a single straight line(if 2D).
- **Non-Linear SVM**: When the data is not linearly separable then we can use Non-Linear SVM, which means when the data points cannot be separated into 2 classes by using a straight line (if 2D) then we use some advanced techniques like kernel tricks to classify them. In most real-world applications we do not find linearly separable datapoints hence we use kernel trick to solve them.

# Linear SVM



Image(a)

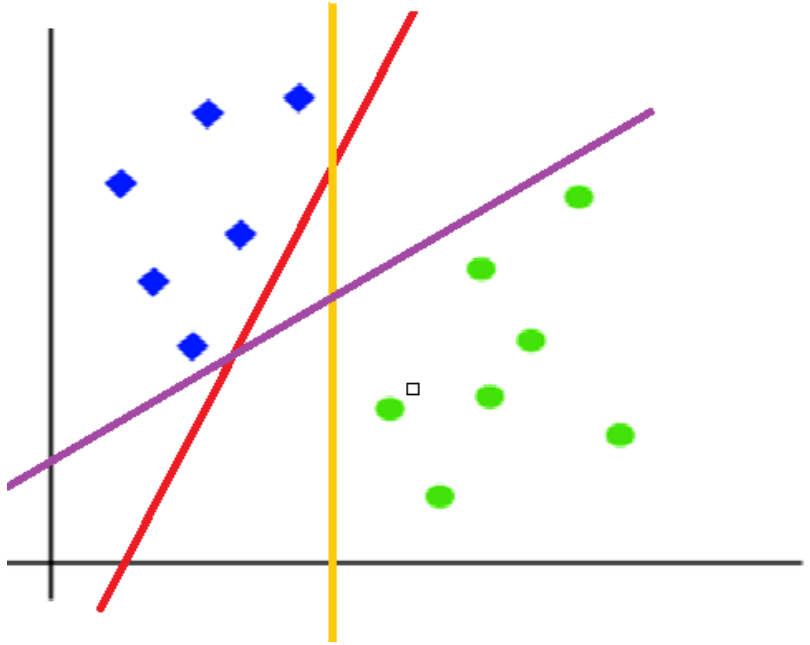
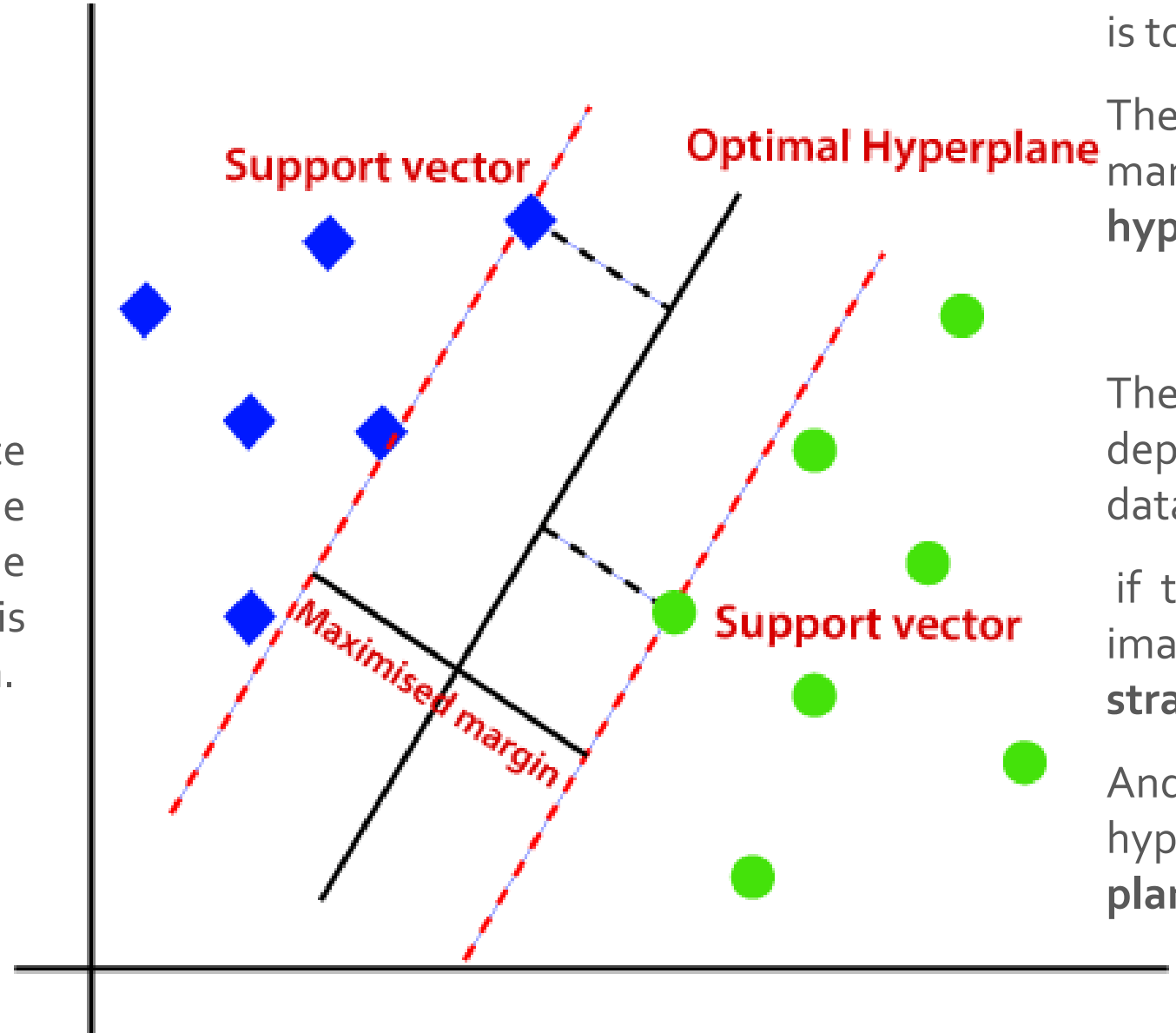


Image (b)

- Suppose we have a dataset that has two tags (green and blue), and the dataset has two features x1 and x2. We want a classifier that can classify the pair(x1, x2) of coordinates in either green or blue. Consider the below image (a). So as it is 2-d space so by just using a straight line, we can easily separate these two classes. But there can be multiple lines that can separate these classes. Consider the below image (b)

SVM algorithm finds the closest point of the lines from both the classes. These points are called **support vectors**.




The distance between the vectors and the hyperplane is called as **margin**.

Hence, the SVM algorithm helps to find the best line or decision boundary; this best boundary or region is called as a **hyperplane**. And the goal of SVM is to maximize this margin.

The **hyperplane** with maximum margin is called the **optimal hyperplane**.

The dimensions of the hyperplane depend on the features present in the dataset, which means-

if there are **2 features** (as shown in image), then hyperplane will be a **straight line**.

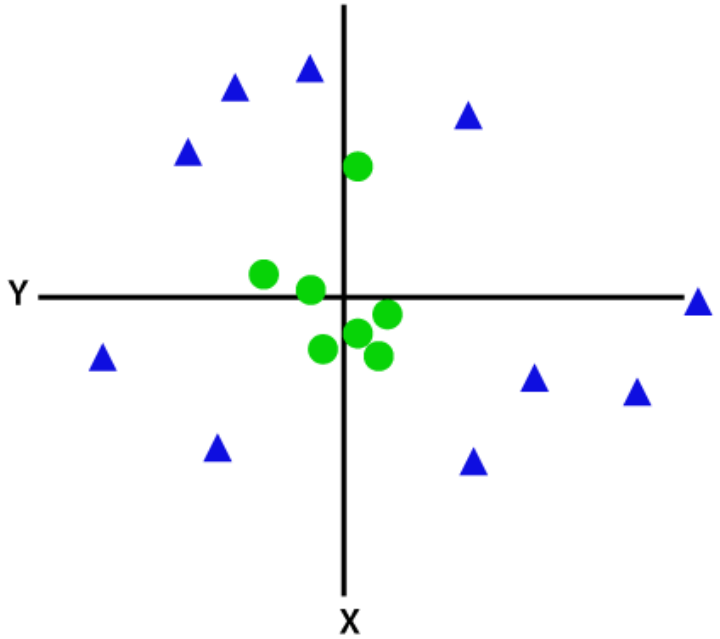
And if there are **3 features**, then hyperplane will be a **2-dimension plane**.

# Contd.

- Hence, the SVM algorithm helps to find the best line or decision boundary; this best boundary or region is called as a **hyperplane**.
- SVM algorithm finds the closest point of the lines from both the classes. These points are called support vectors.
- The distance between the vectors and the hyperplane is called as **margin**.
- And the goal of SVM is to maximize this margin. The **hyperplane** with maximum margin is called the **optimal hyperplane**.

# Non-Linear SVM

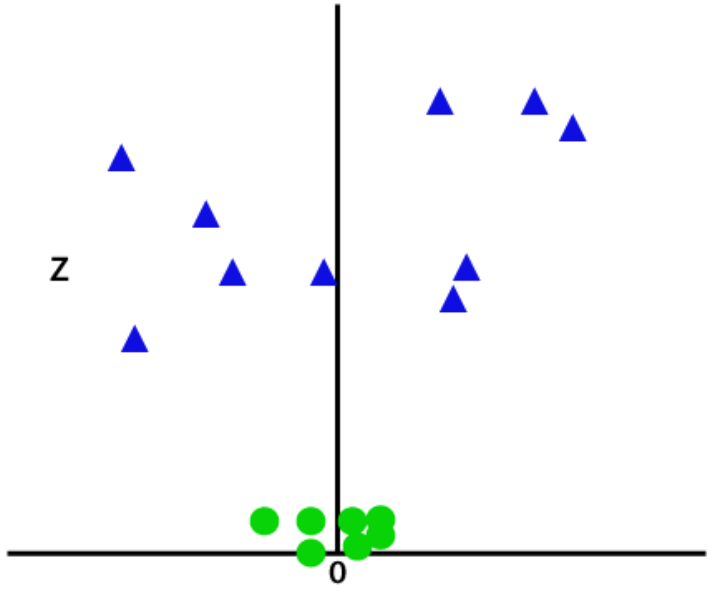
- If data is linearly arranged, then we can separate it by using a straight line, but for non-linear data, we cannot draw a single straight line. Consider the below image:




- So to separate these data points, we need to add one more dimension. For linear data, we have used two dimensions x and y, so for non-linear data, we will add a third dimension z. It can be calculated as:

# Contd.

- $z=x^2+y^2$
- By adding the third dimension, the sample space will become as below image (a).
- So now, SVM will divide the datasets into classes in the following way. Since we are in 3-d Space, hence it is looking like a plane parallel to the x-axis. If we convert it in 2d space with z=1, then it will become as, Consider the below image (b)



Image(a)

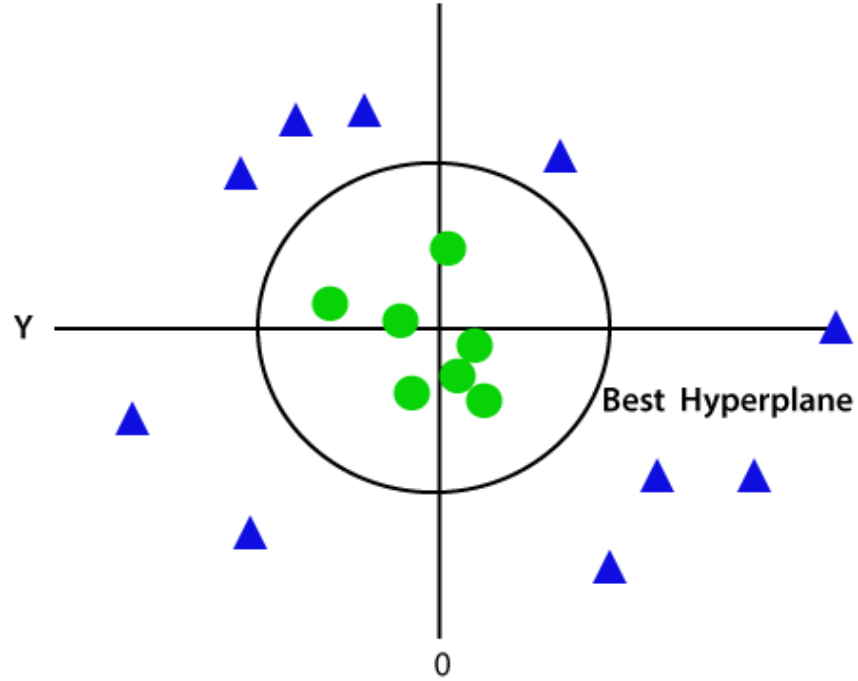
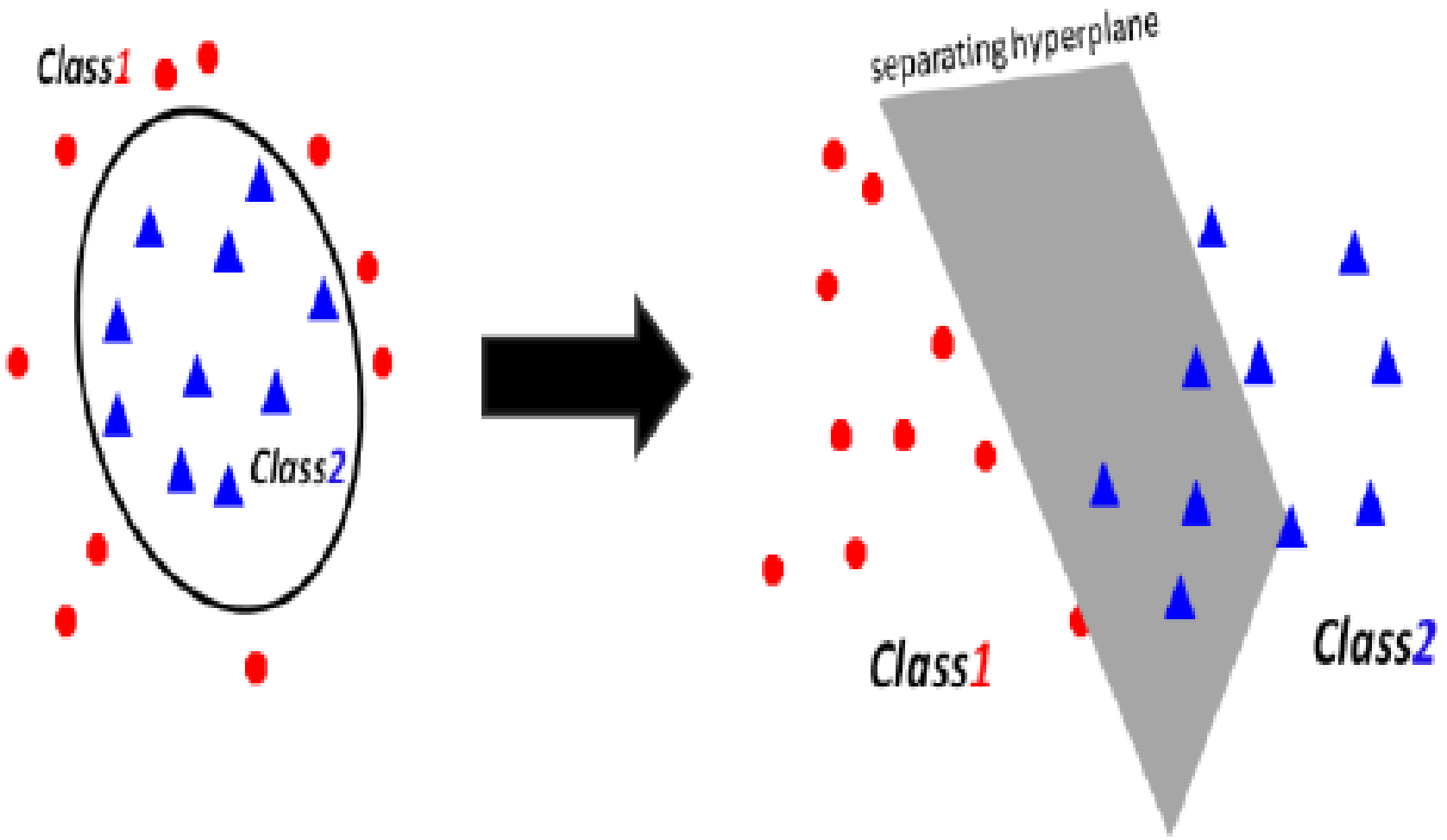


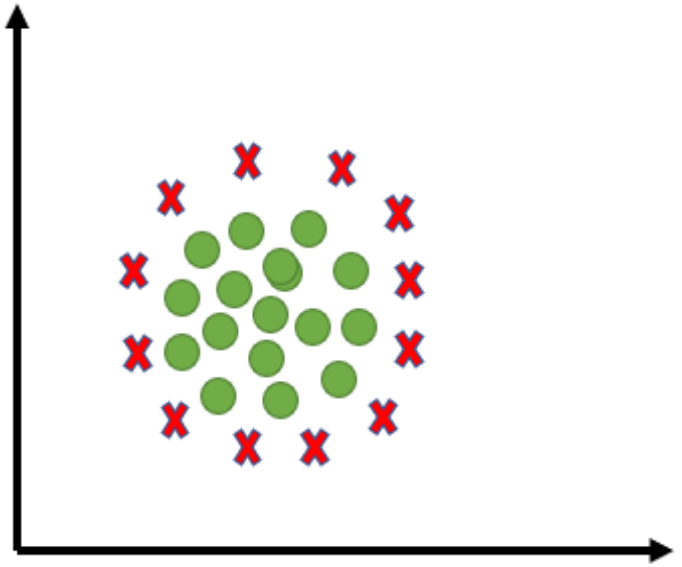
Image (b)

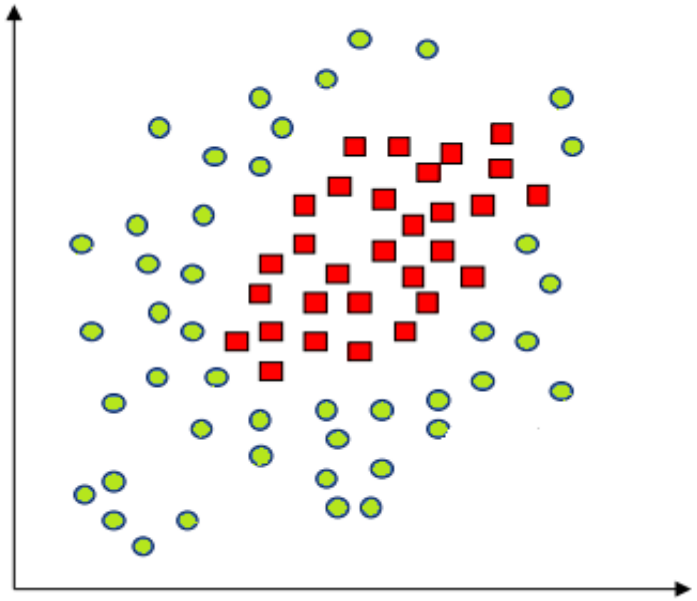
# Contd.

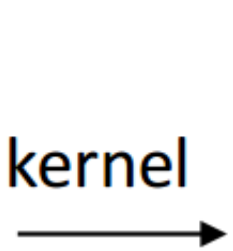


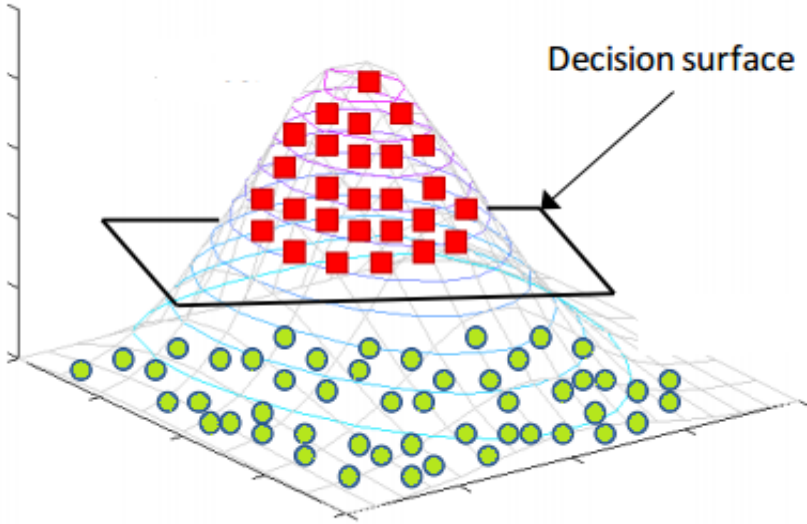
- Kernels in Support Vector Machine

# Contd.





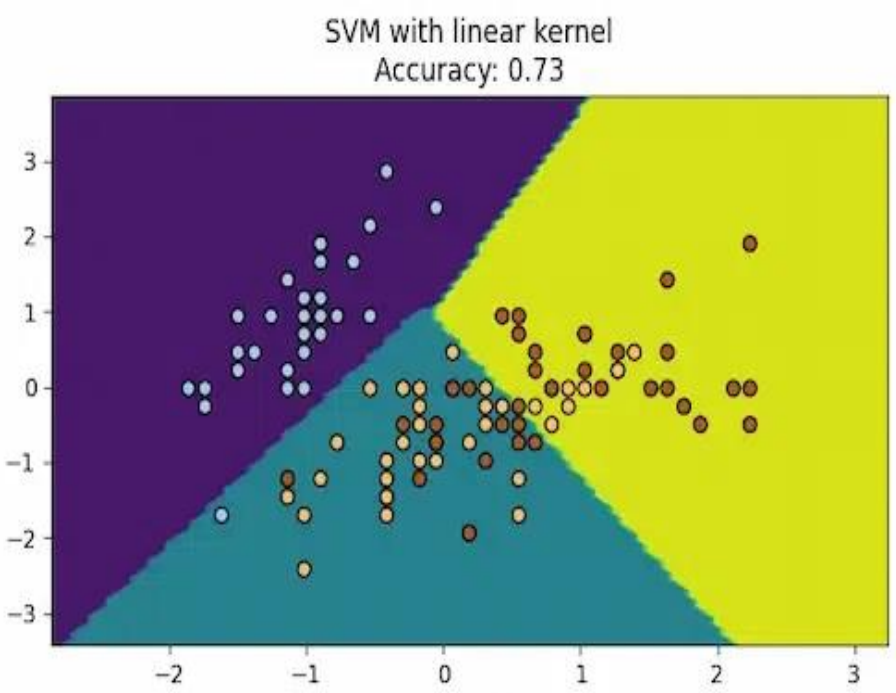


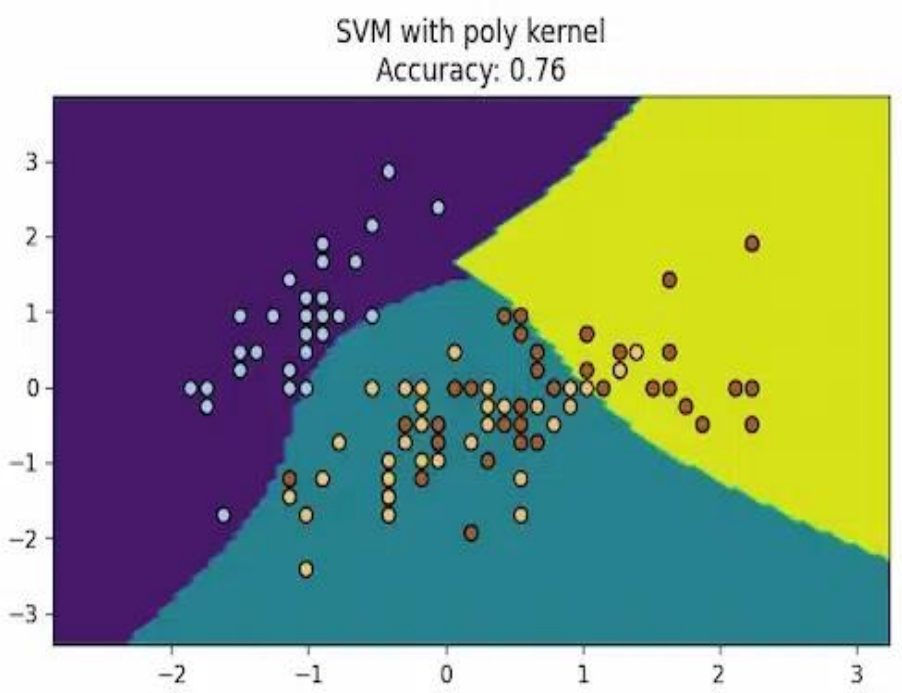



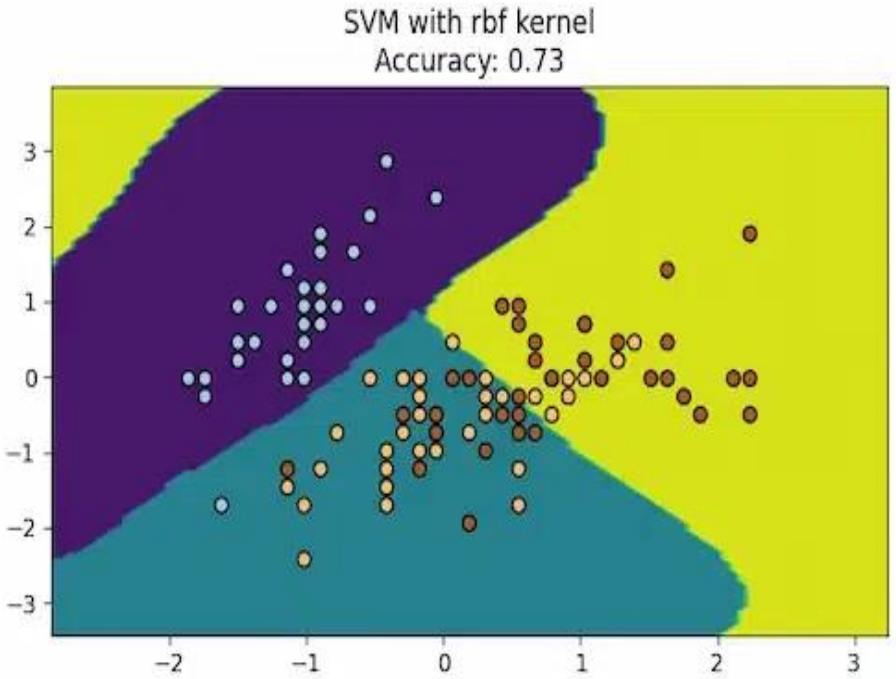
# Different Kernel Functions

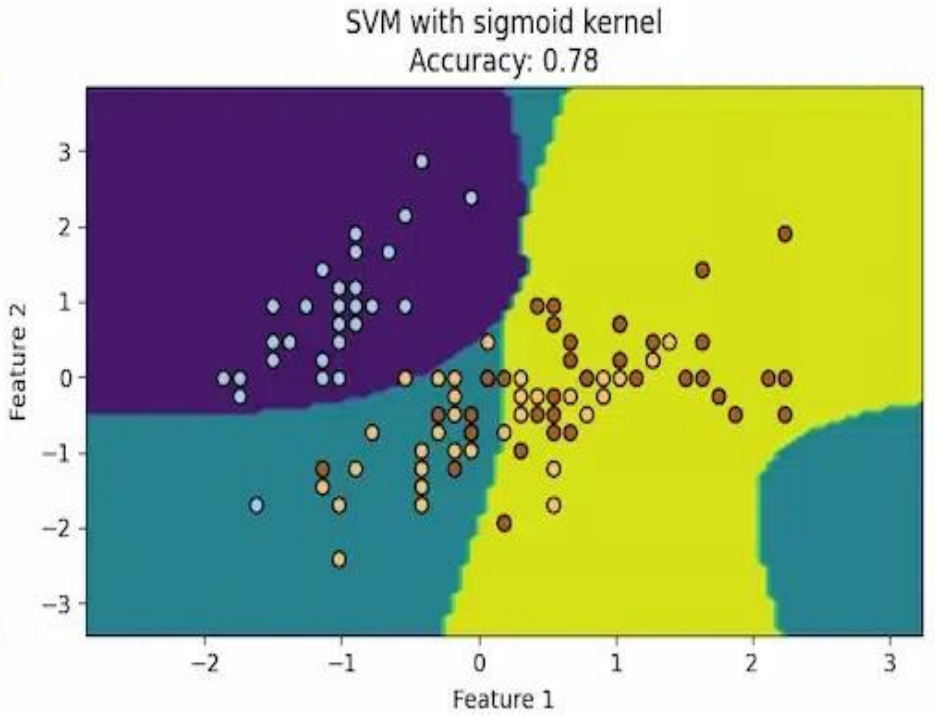
- Linear Kernel
- Polynomial Kernel
- Sigmoid Kernel
- RBF Kernel

# Contd.

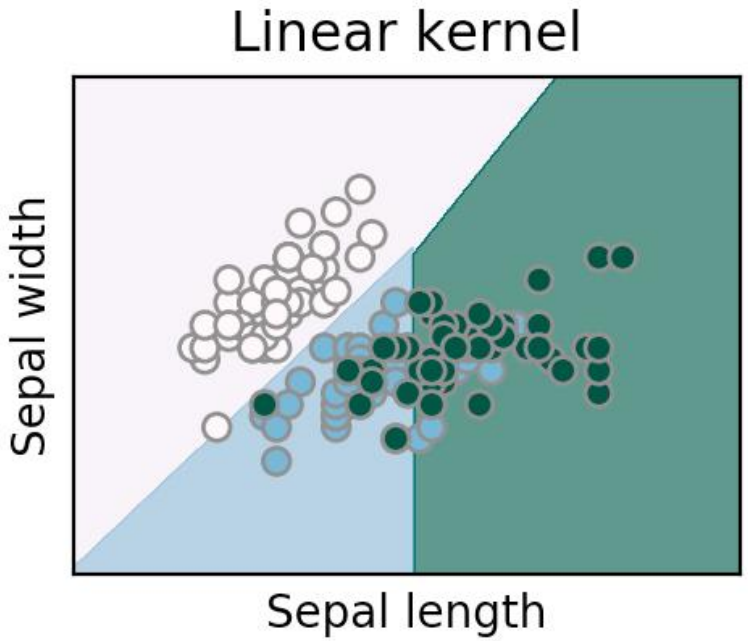

SVM with linear kernel
Accuracy: 0.73

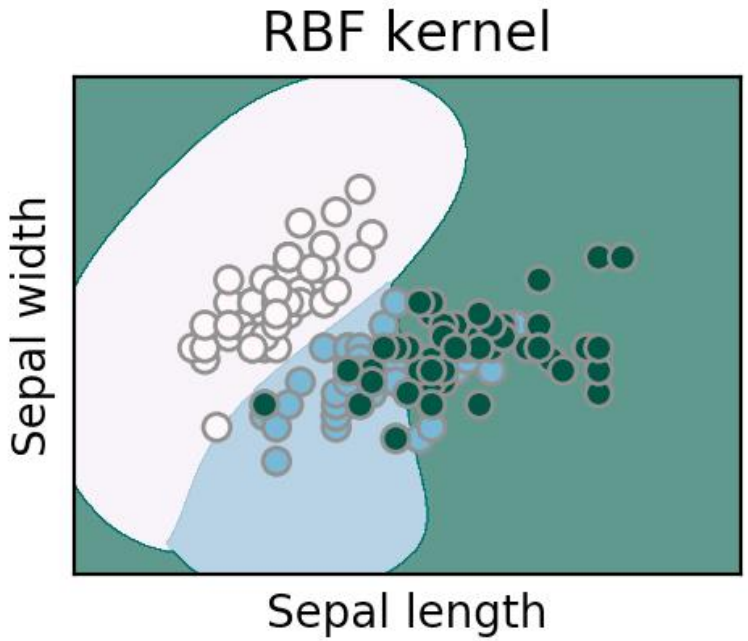

SVM with poly kernel
Accuracy: 0.76

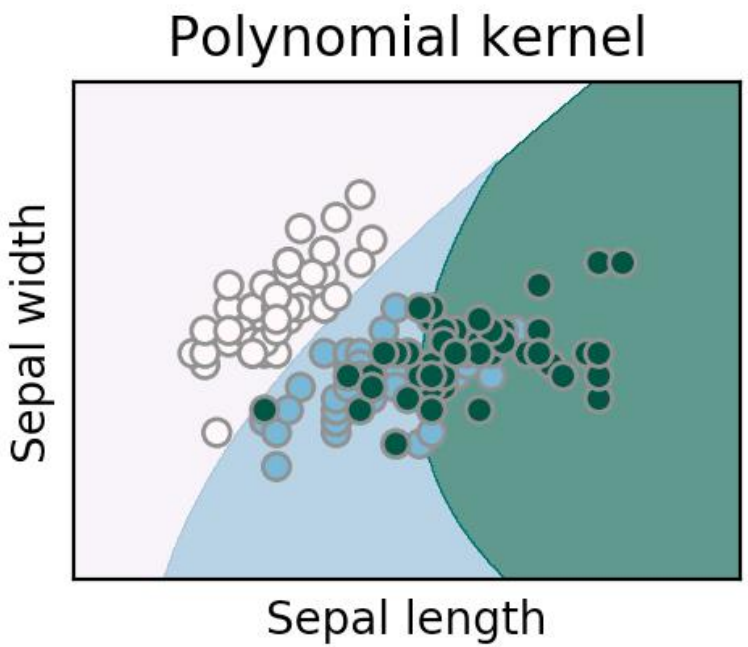

SVM with rbf kernel
Accuracy: 0.73

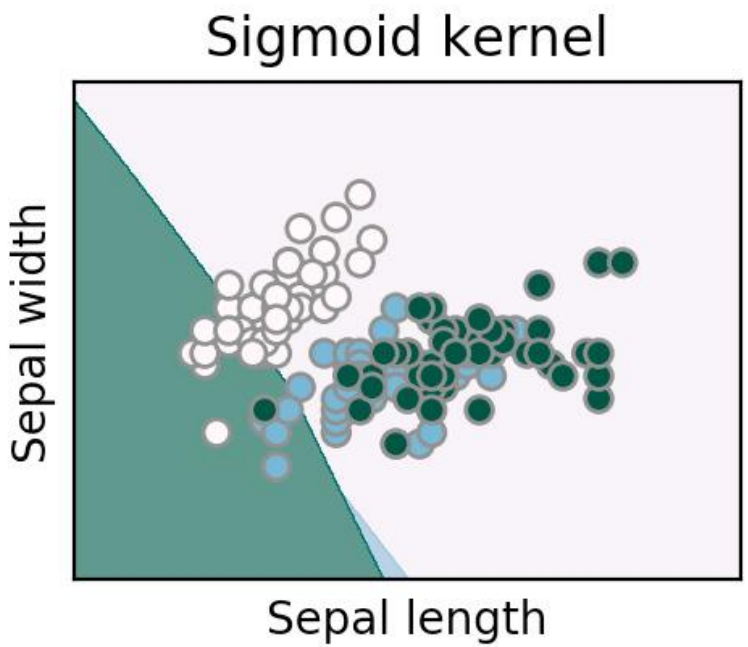

SVM with sigmoid kernel
Accuracy: 0.78
Feature 2
Feature 1

# Contd.

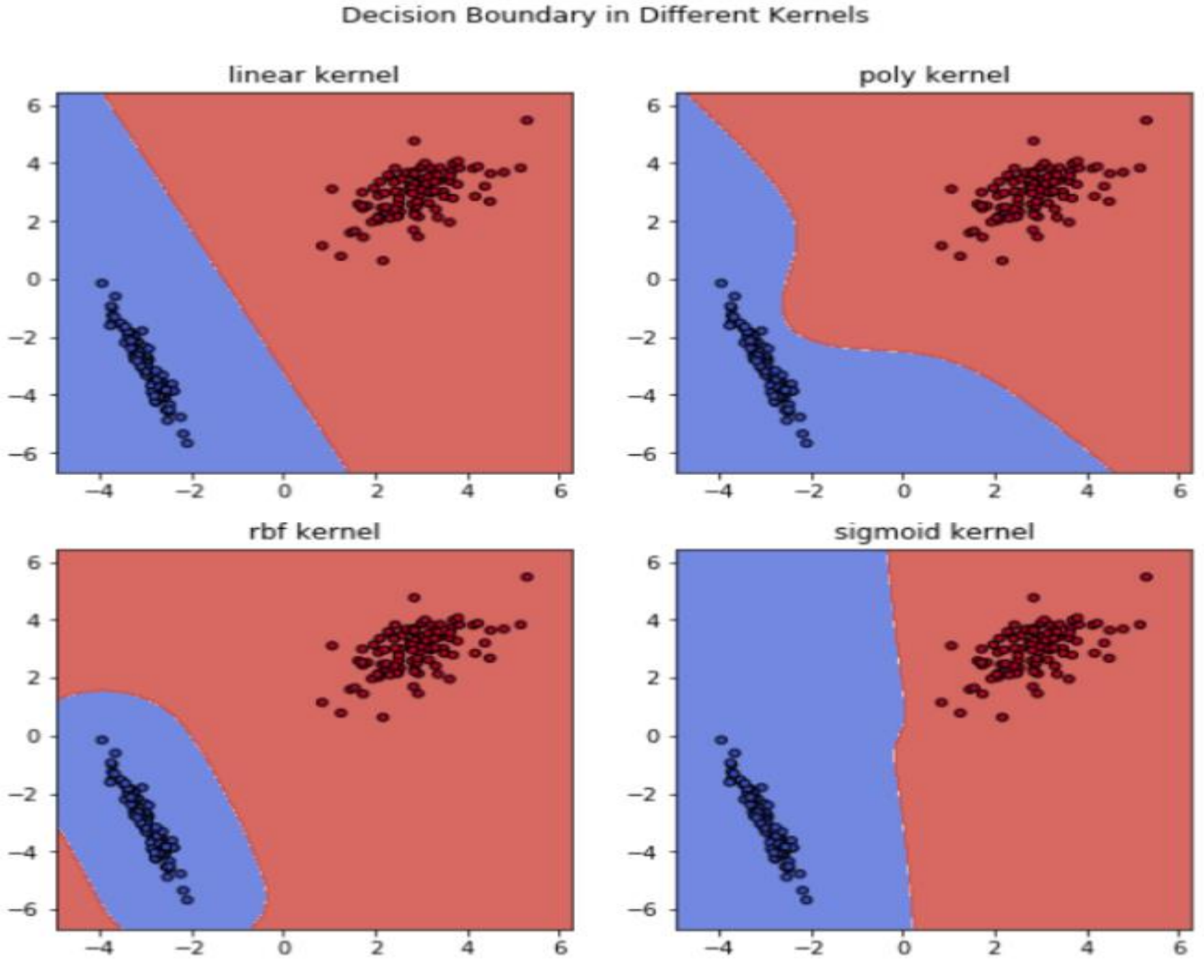

Linear kernel
Sepal width
Sepal length


RBF kernel
Sepal width
Sepal length


Polynomial kernel
Sepal width
Sepal length


Sigmoid kernel
Sepal width
Sepal length

# Contd.


Decision Boundary in Different Kernels
linear kernel
poly kernel
rbf kernel
sigmoid kernel

# Advantages and Disadvantages of SVM

**Advantages of SVM Algorithm:**

- It works really well with a clear margin of separation
- It is effective in high dimensional spaces.
- It is effective in cases where the number of dimensions is greater than the number of samples.

**Disadvantages of SVM Algorithm:**

- It doesn't perform well when we have large data set because the required training time is higher
- It also doesn't perform very well, when the data set has more noise i.e. target classes are overlapping.

# DEMO

---

# SVM

- https://colab.research.google.com/drive/1Yj4t10E7jqxLkGTJeNc-aV7_8epgY3iR?usp=sharing

# K- Nearest Neighbours (KNN)

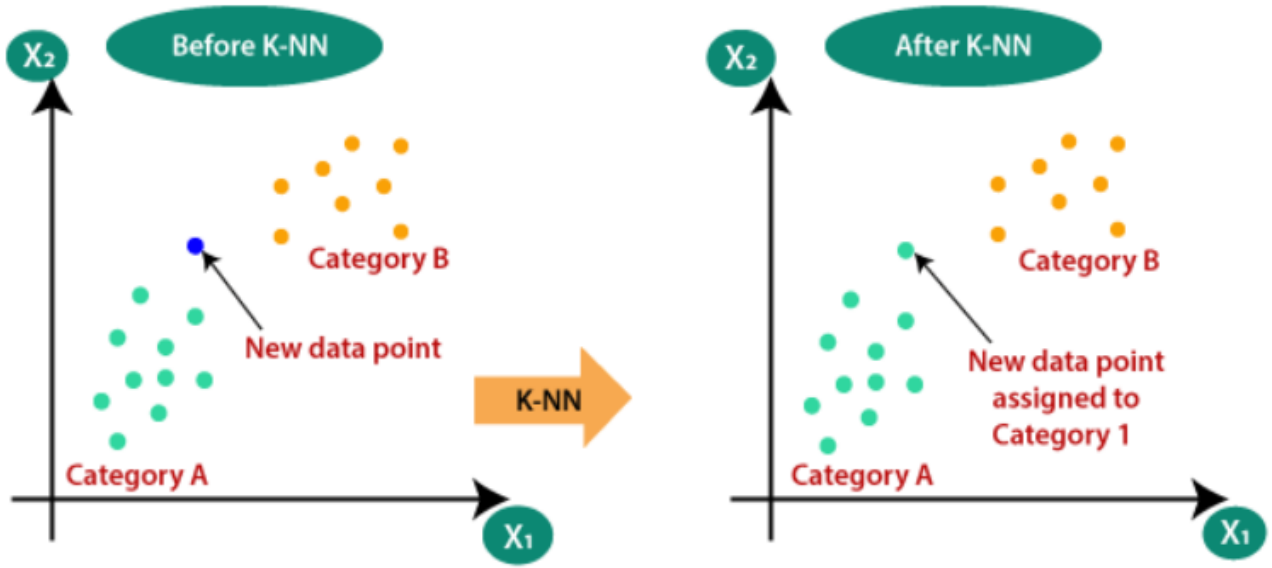
Introduction, Algorithm, Example

# K- Nearest Neighbours (KNN)

- The K-NN algorithm compares a new data entry to the values in a given data set (with different classes or categories).
- Based on its closeness or similarities in a given range (**K**) of neighbors, the algorithm assigns the new data to a class or category in the data set (training data).
- It is also called a **lazy learner algorithm** because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.

# Why do we need a K-NN Algorithm?

- Suppose there are two categories, i.e., Category A and Category B, and we have a new data point x1, so this data point will lie in which of these categories.
- To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset. Consider the below diagram:



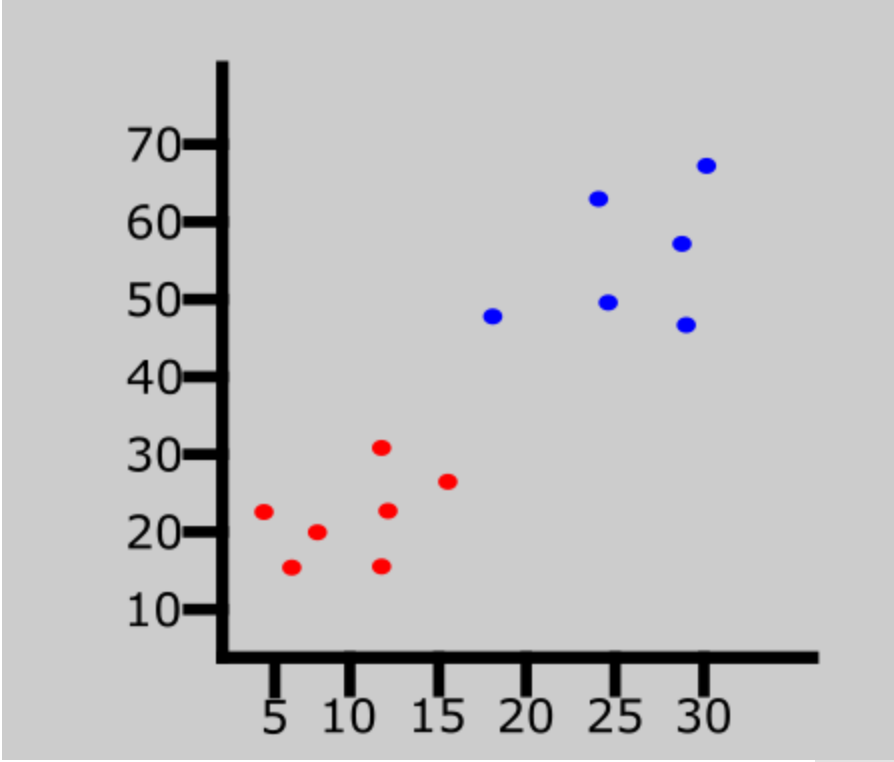
# KNN Algorithm

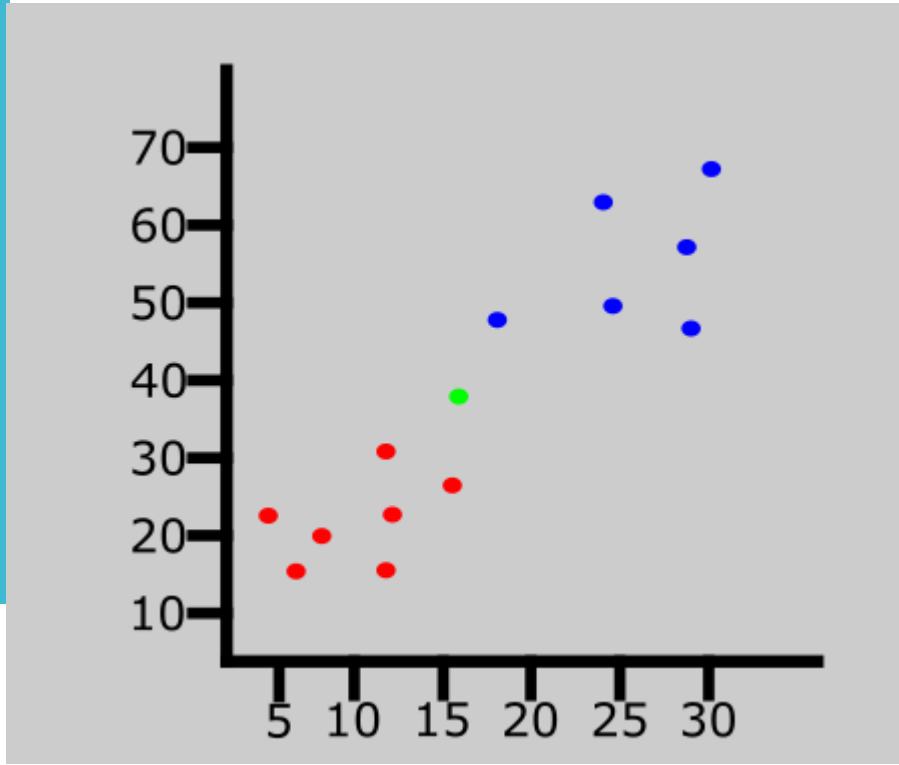
The K-NN working can be explained on the basis of the below algorithm:

- **Step #1 -** Assign a value to **K**.
- **Step #2 -** Calculate the distance between the new data entry and all other existing data entries. Arrange them in ascending order. (Use Euclidean distance formula)
- **Step #3 -** Find the **K** nearest neighbors to the new entry based on the calculated distances.
- **Step #4 -** Assign the new data entry to the majority class in the nearest neighbors.

# K-Nearest Neighbors Classifiers and Model Example With Diagrams

- This data is given, The graph above represents a data set consisting of two classes — red and blue.




- A new data entry has been introduced to the data set. This is represented by the green point in the graph above. Apply KNN to classify Green Point into red or Blue

# Contd.

- **Step #1 -** Assign a value to **K**.

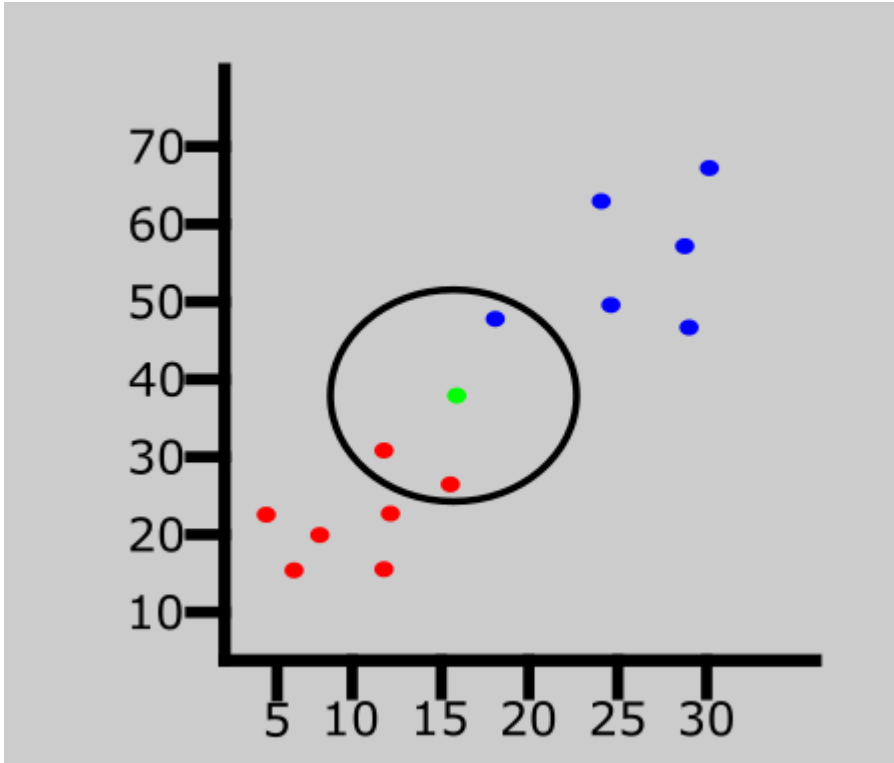
  the value of **K** is 3

- **Step #2 -** Calculate the distance between the new data entry and all other existing data entries (you'll learn how to do this shortly). Arrange them in ascending order.

# Contd.

- **Step #3 -** Find the **K** nearest neighbors to the new entry based on the calculated distances.

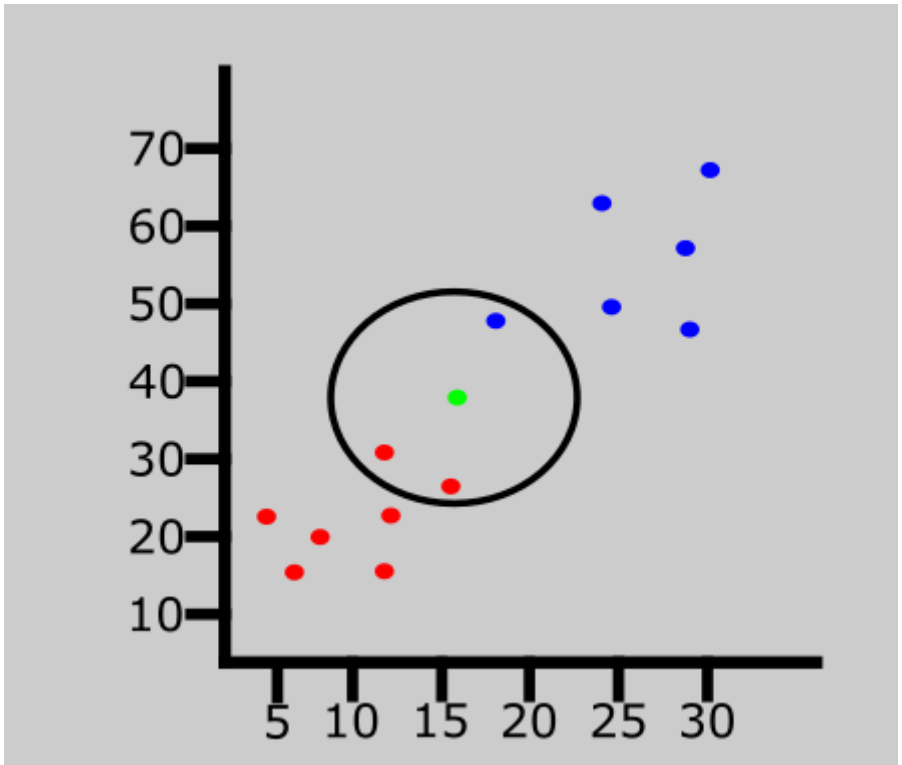
the algoritham will only consider the 3 nearest neighbors to the green point (new entry). This is represented in the graph above.

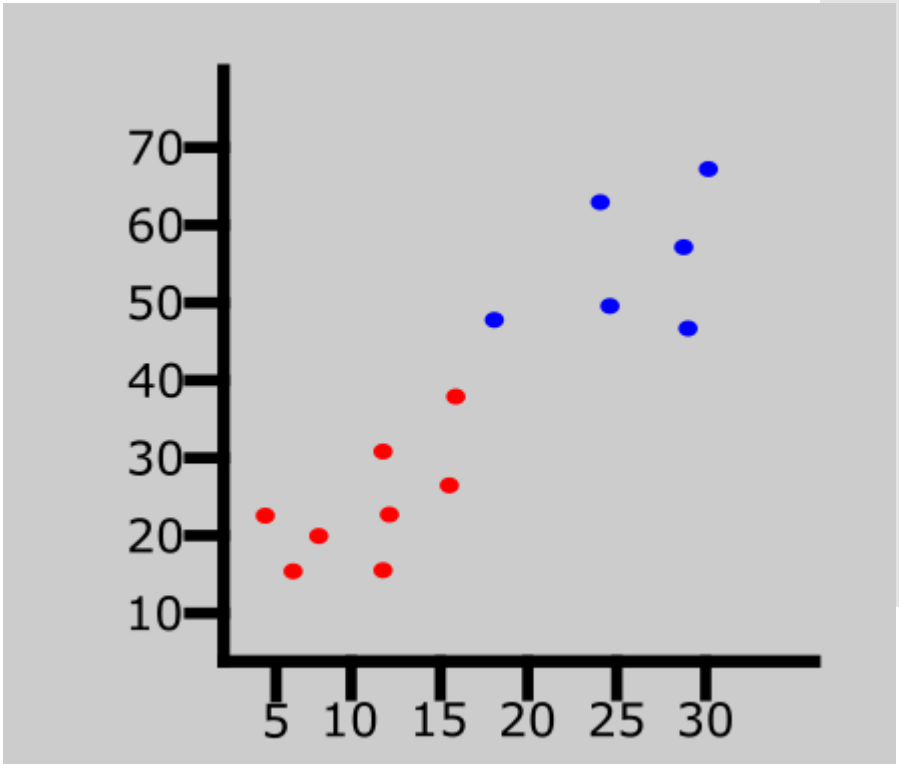


# Contd.

- **Step #4 -** Assign the new data entry to the majority class in the nearest neighbors.

Out of the 3 nearest neighbors in the diagram above, the majority class is red so the new entry will be assigned to that class. The last data entry has been classified as red.



# How to Choose the Value of K in the K-NN Algorithm

- There is no particular way of choosing the value **K**, but here are some common conventions to keep in mind:
- Choosing a very low value will most likely lead to inaccurate predictions.
- The commonly used value of **K** is 5.
- Always use an odd number as the value of **K**.

# K-Nearest Neighbors Classifiers and Model Example With Data Set

| BRIGHTNESS | SATURATION | CLASS |
|---|---|---|
| 40 | 20 | Red |
| 50 | 50 | Blue |
| 60 | 90 | Blue |
| 10 | 25 | Red |
| 70 | 70 | Blue |
| 60 | 10 | Red |
| 25 | 80 | Blue |

The table above represents our data set. We have two columns - **Brightness** and **Saturation**. Each row in the table has a **class of either Red or Blue**. let's assume the value of **K** is **5**. we introduce a new data entry,

| BRIGHTNESS | SATURATION | CLASS |
|---|---|---|
| 20 | 35 | ? |

Apply KNN to find the class of New Entry.

# Contd.

- **Step #1 -** Assign a value to **K**.

  here the value of **K** is 5 (Given)

- **Step #2 -** Calculate the distance between the new data entry and all other existing data entries. Arrange them in ascending order.

Consider Distance = Euclidean distance formula

# Contd.

- To know its class, we have to calculate the distance from the new entry to other entries in the data set using the Euclidean distance formula.
- Here's the formula: $d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$
- Where:
- $X_2$ = New entry's brightness (20).
- $X_1$= Existing entry's brightness.
- $Y_2$ = New entry's saturation (35).
- $Y_1$ = Existing entry's saturation.

| BRIGHTNESS **(X2)** | SATURATION **(Y2)** | CLASS | $d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$ |
|---|---|---|---|
| 40 | 20 | Red | d1 = √(20 - 40)² + (35 - 20)²<br>= √400 + 225 = √625 = 25 |
| 50 | 50 | Blue | d2 = √(20 - 50)² + (35 - 50)²<br>= √900 + 225 = √1125 = 33.54 |
| 60 | 90 | Blue | d3 = √(20 - 60)² + (35 - 90)²<br>= √1600 + 3025 = √4625 = 68.01 |
| 10 | 25 | Red | ?? |
| 70 | 70 | Blue | ?? |
| 60 | 10 | Red | ?? |
| 25 | 80 | Blue | ?? |
| X1 = 20 | Y1 = 35 | ?? | New Entry |

| BRIGHTNESS | SATURATION | CLASS | DISTANCE |
|---|---|---|---|
| 40 | 20 | Red | 25 |
| 50 | 50 | Blue | 33.54 |
| 60 | 90 | Blue | 68.01 |
| 10 | 25 | Red | 14.14 |
| 70 | 70 | Blue | 61.03 |
| 60 | 10 | Red | 47.17 |
| 25 | 80 | Blue | 45.28 |
| **X1 = 20** | **Y1 = 35** | **??** | **New Entry** |

Let's rearrange the distances in ascending order:

| BRIGHTNESS | SATURATION | CLASS | DISTANCE |
|---|---|---|---|
| 10 | 25 | Red | 14.14 |
| 40 | 20 | Red | 25 |
| 50 | 50 | Blue | 33.54 |
| 25 | 80 | Blue | 45.28 |
| 60 | 10 | Red | 47.17 |
| 70 | 70 | Blue | 61.03 |
| 60 | 90 | Blue | 68.01 |
| **X1 = 20** | **Y1 = 35** | **??** | **New Entry** |

# Contd.

- **Step #3 -** Find the **K** nearest neighbors to the new entry based on the calculated distances.

Since we chose 5 as the value of **K**, we'll only consider the first five rows. That is:

| BRIGHTNESS | SATURATION | CLASS | DISTANCE | CONSIDER |
|---|---|---|---|---|
| 10 | 25 | Red | 10 | Yes |
| 40 | 20 | Red | 25 | Yes |
| 50 | 50 | Blue | 33.54 | Yes |
| 25 | 80 | Blue | 45 | Yes |
| 60 | 10 | Red | 47.17 | Yes |
| 70 | 70 | Blue | 61.03 | No |
| 60 | 90 | Blue | 68.01 | No |
| **X1 = 20** | **Y1 = 35** | **??** | **New Entry** | - |

# Contd.

- **Step #4 -** Assign the new data entry to the majority class in the nearest neighbors.
- As you can see above, the majority class within the 5 nearest neighbors to the new entry is **Red**. Therefore, we'll classify the new entry as **Red**.

| BRIGHTNESS | SATURATION | CLASS | DISTANCE | CONSIDER |
|---|---|---|---|---|
| 10 | 25 | Red | 10 | Yes |
| 40 | 20 | Red | 25 | Yes |
| 50 | 50 | Blue | 33.54 | Yes |
| 25 | 80 | Blue | 45 | Yes |
| 60 | 10 | Red | 47.17 | Yes |
| **X1 = 20** | **Y1 = 35** | **Red** | **New Entry** | - |

| BRIGHTNESS | SATURATION | CLASS |
|---|---|---|
| 40 | 20 | Red |
| 50 | 50 | Blue |
| 60 | 90 | Blue |
| 10 | 25 | Red |
| 70 | 70 | Blue |
| 60 | 10 | Red |
| 25 | 80 | Blue |

Given Old Table

- Here's the updated table:

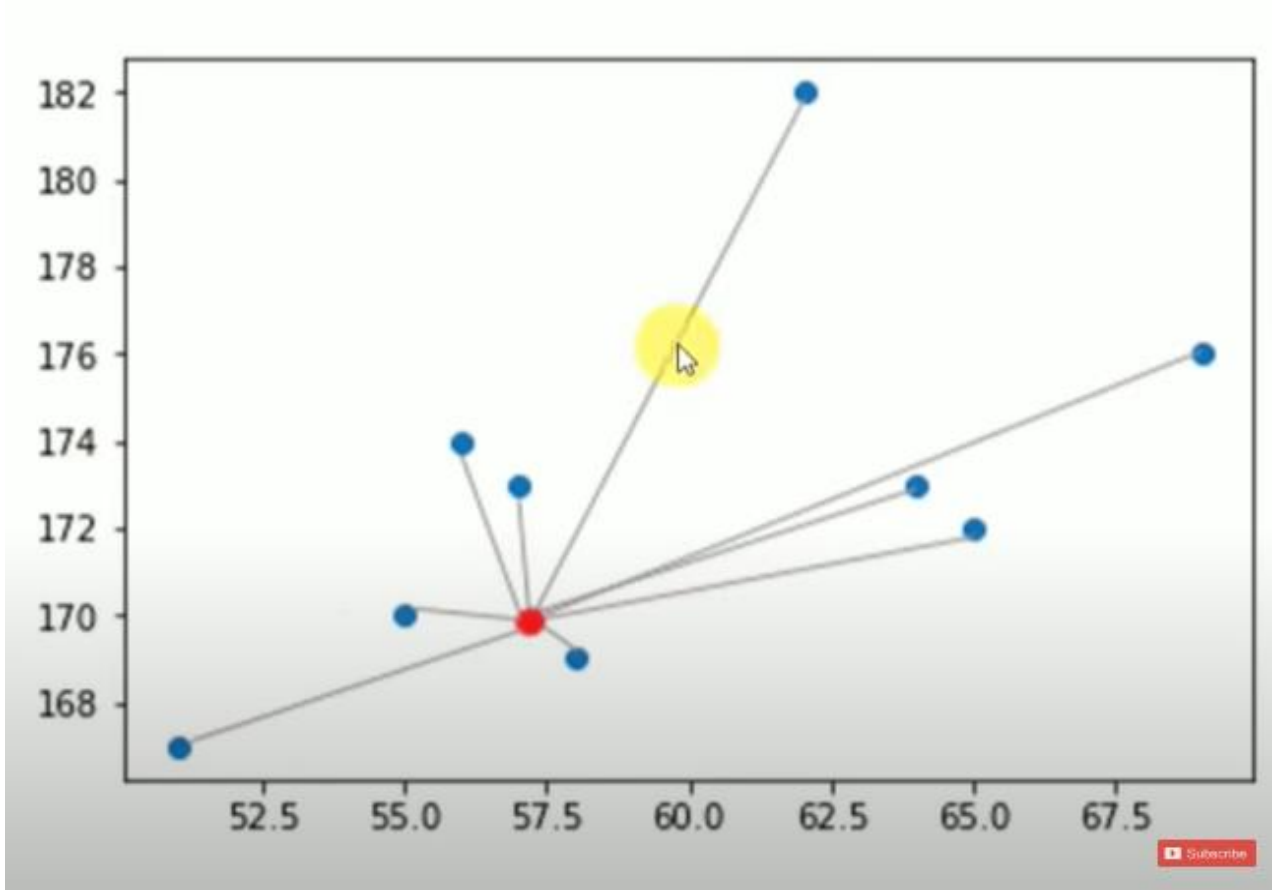
| BRIGHTNESS | SATURATION | CLASS |
|---|---|---|
| 40 | 20 | Red |
| 50 | 50 | Blue |
| 60 | 90 | Blue |
| 10 | 25 | Red |
| 70 | 70 | Blue |
| 60 | 10 | Red |
| 25 | 80 | Blue |
| **20** | **35** | **Red** |

# Example:

Apply KNN to find the class of New Entry. (Take k =5)

| Height (CM) | Weight (KG) | Class |
|---|---|---|
| 167 | 51 | Underweight |
| 182 | 62 | Normal |
| 176 | 69 | Normal |
| 173 | 64 | Normal |
| 172 | 65 | Normal |
| 174 | 56 | Underweight |
| 169 | 58 | Normal |
| 173 | 57 | Normal |
| 170 | 55 | Normal |
| 170 | 57 | ? |

# Contd.



# Contd.

| Height (CM) | Weight (KG) | Class | Distance |
|---|---|---|---|
| 167 | 51 | Underweight | 6.7 |
| 182 | 62 | Normal | 13 |
| 176 | 69 | Normal | 13.4 |
| 173 | 64 | Normal | 7.6 |
| 172 | 65 | Normal | 8.2 |
| 174 | 56 | Underweight | 4.1 |
| 169 | 58 | Normal | 1.4 |
| 173 | 57 | Normal | 3 |
| 170 | 55 | Normal | 2 |
| 170 | 57 | ? | |

# Contd.

| Height (CM) | Weight (KG) | Class | Distance | Rank |
|---|---|---|---|---|
| 169 | 58 | Normal | 1.4 | 1 ✓ |
| 170 | 55 | Normal | 2 | 2 |
| 173 | 57 | Normal | 3 | 3 |
| 174 | 56 | Underweight | 4.1 | 4 |
| 167 | 51 | Underweight | 6.7 | 5 |
| 173 | 64 | Normal | 7.6 | 6 |
| 172 | 65 | Normal | 8.2 | 7 |
| 182 | 62 | Normal | 13 | 8 |
| 176 | 69 | Normal | 13.4 | 9 |
| 170 | 57 | Normal | | |

# Example

Apply KNN to find the class of New Entry. (Take k =1,2,5)

| 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 |

ANS:

| 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**

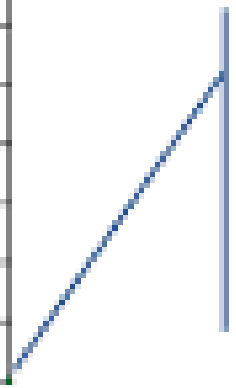
**If k = 1 – Setosa**

**If k = 2 – Setosa**

**If k = 5 – Setosa**

# Example

| Customer | Age | Loan | Default |
|---|---|---|---|
| John | 25 | 40000 | N |
| Smith | 35 | 60000 | N |
| Alex | 45 | 80000 | N |
| Jade | 20 | 20000 | N |
| Kate | 35 | 120000 | N |
| Mark | 52 | 18000 | N |
| Anil | 23 | 95000 | Y |
| Pat | 40 | 62000 | Y |
| George | 60 | 100000 | Y |
| Jim | 48 | 220000 | Y |
| Jack | 33 | 150000 | Y |
| Andrew | 48 | 142000 | ? |



We need to predict Andrew default status by using Euclidean distance

# Advantages and Disadvantages of KNN

**Advantages of KNN Algorithm:**

- It is simple to implement.
- It is robust to the noisy training data
- It can be more effective if the training data is large.

**Disadvantages of KNN Algorithm:**

- Always needs to determine the value of K which may be complex some time.
- The computation cost is high because of calculating the distance between the data points for all the training samples.

# DEMO

---

# KNN

- https://colab.research.google.com/drive/1Yj4t10E7jqxLkGTJeNc-aV7_8epgY3iR?usp=sharing

# Naive Bayes Classifier

Introduction, Algorithm, Types of Distribution, Naive Bayes Classifier Approach with Example (For Single Feature and For Multiple Feature)

# Naive Bayes Classifier

- Naive Bayes is a statistical classification technique based on Bayes Theorem.
- It is one of the simplest supervised learning algorithms. Naive Bayes classifier is the fast, accurate and reliable algorithm.
- Naive Bayes classifiers have high accuracy and speed on large datasets.

# Assumption of Naive Bayes

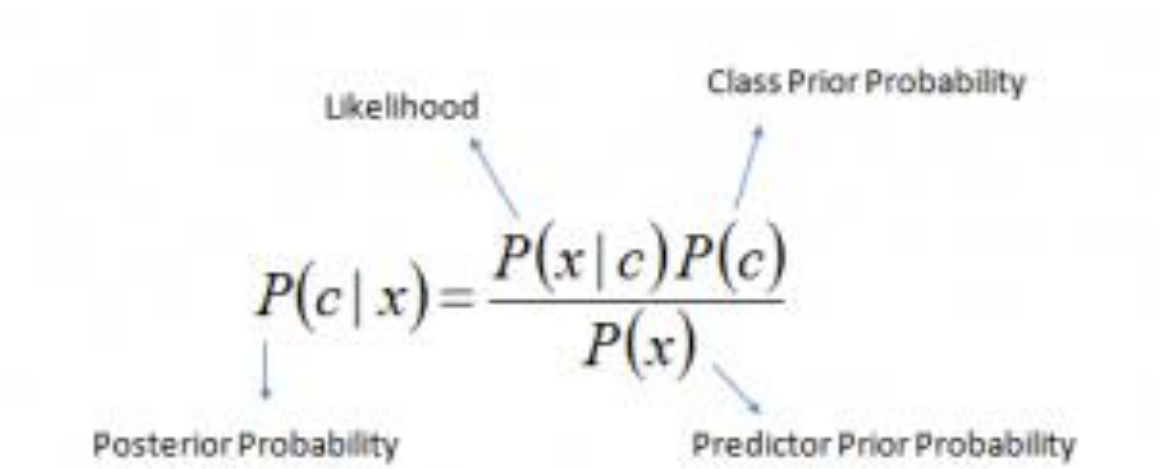
- **Feature independence:** The features of the data are conditionally independent of each other, given the class label.
- **Continuous features are normally distributed:** If a feature is continuous, then it is assumed to be normally distributed within each class.
- **Discrete features have multinomial distributions:** If a feature is discrete, then it is assumed to have a multinomial distribution within each class.
- **Features are equally important:** All features are assumed to contribute equally to the prediction of the class label.
- **No missing data:** The data should not contain any missing values.

# Multivariate Bernoulli Distribution, Gaussian Distribution, and Multinomial Distribution

| Aspect | Multivariate Bernoulli Distribution | Gaussian Distribution | Multinomial Distribution |
|---|---|---|---|
| **Type of Data** | Binary (0 or 1) for each feature | Continuous data | Categorical data with multiple classes |
| **Example Use Case** | Email spam detection (presence or absence of words) | Predicting house prices based on features | Text classification (frequency of words) |
| **Parameters** | Probability of each feature being 1 | Mean and variance for each feature | Probability of each category |
| **Assumption** | Features are independent given the class | Features are normally distributed | Features are categorical and independent given the class |
| **Application** | Naive Bayes classifiers for binary features | Gaussian Naive Bayes classifier | Multinomial Naive Bayes classifier |
| **Distribution** | Each feature follows a Bernoulli distribution | Each feature follows a Gaussian (Normal) distribution | Each feature follows a Multinomial distribution |
| **Output** | Probability of binary feature occurrence | Probability density of continuous features | Probability of categorical feature occurrence |

# Bayes' Theorem

- Bayes theorem provides a way of computing posterior probability P(c|x) from P(c), P(x) and P(x|c). Look at the equation below:




Above, $$P(c \mid \mathbf{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

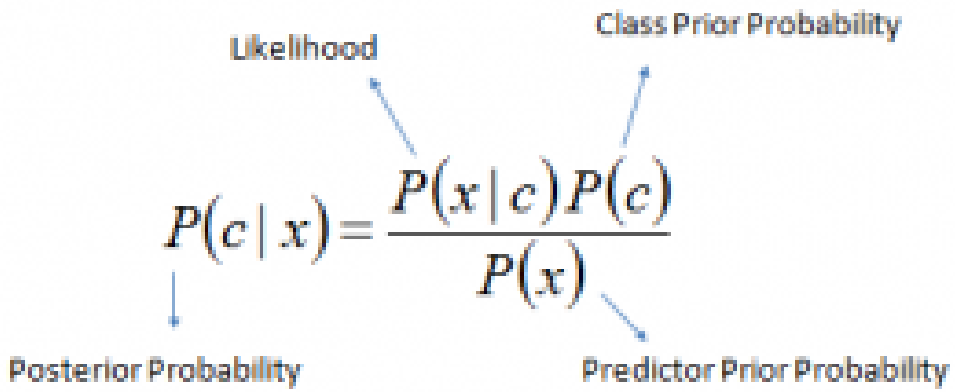
- $P(c|x)$ is the posterior probability of *class* (c, *target*) given *predictor* (x, *attributes*).
- $P(c)$ is the prior probability of *class*.
- $P(x|c)$ is the likelihood which is the probability of the *predictor* given *class*.
- $P(x)$ is the prior probability of the *predictor*.

# Naive Bayes Classifier Approach

- **First Approach (In case of a single feature)**
- **Second Approach (In case of multiple features)**

# First Approach (In case of a single feature)

- **Step 1:** Convert the data set into a frequency table
- **Step 2:** Create Likelihood table by finding the probabilities
- **Step 3:** Apply Bayes Formula and calculate posterior probability for each target value




- **Step 4**: See which class has a higher probability, given the input belongs to the higher probability class.

# Example:

- Apply Naive Bayes Classifier.
  - Players will play if the weather is overcast. Is this statement correct
  - Players will play if the weather is sunny. Is this statement correct?
  - Players will play if the weather is Rainy. Is this statement correct?

| Whether | Play |
|---|---|
| Sunny | No |
| Sunny | No |
| Overcast | Yes |
| Rainy | Yes |
| Rainy | Yes |
| Rainy | No |
| Overcast | Yes |
| Sunny | No |
| Sunny | Yes |
| Rainy | Yes |
| Sunny | Yes |
| Overcast | Yes |
| Overcast | Yes |
| Rainy | No |

# Contd.

| Whether | Play |
|---|---|
| Sunny | No |
| Sunny | No |
| Overcast | Yes |
| Rainy | Yes |
| Rainy | Yes |
| Rainy | No |
| Overcast | Yes |
| Sunny | No |
| Sunny | Yes |
| Rainy | Yes |
| Sunny | Yes |
| Overcast | Yes |
| Overcast | Yes |
| Rainy | No |

- **Step 1:** Convert the data set into a frequency table

| Whether | No | Yes |
|---|---|---|
| Overcast | 0 | 4 |
| Sunny | 3 | 2 |
| Rainy | 2 | 3 |
| Total: | 5 | 9 |

**Step 2:** Create Likelihood table by finding the probabilities

| Whether | No | Yes | |
|---|---|---|---|
| Overcast | 0 | 4 | P(Overcast) = 4/14 = 0.29 |
| Sunny | 3 | 2 | P(Sunny) = 5/14 = 0.36 |
| Rainy | 2 | 3 | P(Rainy) = 5/14 = 0.36 |
| Total: | 5 | 9 | |
| | P(No) = 5/14 = 0.36 | P(Yes) = 9/14 = 0.64 | |

**Likelihood Table 1**

| Whether | No | Yes | Posterior probabilities (No) | Posterior probabilities (Yes) |
|---|---|---|---|---|
| Overcast | 0 | 4 | P(Overcast \| No) = 0 / 5 = 0 | P(Overcast \| Yes) = 4/9 = 0.44 |
| Sunny | 3 | 2 | P(Sunny \| No) = 3 / 5 = 0.6 | P(Sunny \| Yes) = 2/9 =0.22 |
| Rainy | 2 | 3 | P(Rainy \| No) = 2/ 5 = 0.4 | P(Overcast \| Yes) =3 /9 =0.33 |
| Total: | 5 | 9 | | |

**Likelihood Table 2**

**Step 3:** Apply Bayes Formula and calculate posterior probability for each target value

Now suppose you want to calculate the probability of playing when the weather is overcast.

- Probability of playing:
- P(Yes | Overcast) = [ P(Overcast | Yes) * P(Yes) ] / P (Overcast)

  = (0.44 * 0.64 ) / 0.29 = 0.98

Now suppose you want to calculate the probability of playing when the weather is overcast.

- Probability of not playing:
- P(No | Overcast) = [ P(Overcast | No) * P(No) ] / P (Overcast)

  = (0 * 0.36 ) / 0.29 = 0

# Contd.

- **Step 4**: See which class has a higher probability, given the input belongs to the higher probability class.
- P(Yes | Overcast) = 0.98
- P(No | Overcast) = 0

Ans: Yes, Players will play if the weather is overcast.

this statement correct

# Contd.

- Players will play if the weather is sunny. Is this statement correct?
- Ans: ???

- Players will play if the weather is Rainy. Is this statement correct?
- Ans: ???

# Example:

| Fruit ID | Color (Feature) | Class |
|---|---|---|
| 1 | Red | Apple |
| 2 | Red | Apple |
| 3 | Green | Apple |
| 4 | Orange | Orange |
| 5 | Orange | Orange |
| 6 | Green | Apple |

Apply Naive Bayes Classifier to classify whether a fruit is "Apple" or "Orange" if new fruit that is "Red", color (our single feature).

# Contd.

| Fruit ID | Color (Feature) | Class |
|---|---|---|
| 1 | Red | Apple |
| 2 | Red | Apple |
| 3 | Green | Apple |
| 4 | Orange | Orange |
| 5 | Orange | Orange |
| 6 | Green | Apple |

- **Step 1:** Convert the data set into a frequency table

| Color | Apple | Orange |
|---|---|---|
| Red | 2 | 0 |
| Green | 2 | 0 |
| Orange | 0 | 2 |
| Total: | 4 | 2 |

**Step 2:** Create Likelihood table by finding the probabilities

| Color | Apple | Orange | |
|---|---|---|---|
| Red | 2 | 0 | P(Red) = 2/6 = 0.33 |
| Green | 2 | 0 | P(Green) = 2/6 = 0.33 |
| Orange | 0 | 2 | P(Orange) = 2/6 = 0.33 |
| Total: | 4 | 2 | |
| | P(Apple) = 4/6 = 0.67 | P(Orange) = 2/6 = 0.33 | |

**Likelihood Table 1**

**Likelihood Table 2**

| Color | Apple | Orange | Posterior probabilities (Apple) | Posterior probabilities (Orange) |
|---|---|---|---|---|
| Red | 2 | 0 | P(Red \| Apple) = 2 /4 = 0.5 | P(Red \| Orange) = 0/2 = 0 |
| Green | 2 | 0 | P(Green \| Apple) = 2 / 4 = 0.5 | P(Green \| Orange) = 0/2 = 0 |
| Orange | 0 | 2 | P(Orange \| Apple) = 0 / 4 = 0 | P(Orange \| Orange) = 2/2 =1 |
| Total: | 4 | 2 | | |

**Step 3:** Apply Bayes Formula and calculate posterior probability for each target value

Now suppose you want to calculate the probability of Apple when the Color is Red.

- Probability of Apple:
- P(Apple | Red) = [ P(Red | Apple) * P(Apple) ] / P (Red)

  = (0.5 * 0.66 ) / 0.33 = 1

Now suppose you want to calculate the probability of Orange when the Color is Red.

- Probability of Orange:
- P(Orange | Red) = = [ P(Red | Orange) * P(Orange) ] / P (Red)

  = (0 * 0.33 ) / 0.33 = 0

# Contd.

- **Step 4**: See which class has a higher probability, given the input belongs to the higher probability class.
- P(Apple | Red) = 1
- P(Orange | Red) = 0

Ans: if new fruit that is "Red"color, then it is classify as Apple.

# Example

| Person ID | Favorite Drink (Feature) | Class |
|---|---|---|
| 1 | Soda | Teen |
| 2 | Coffee | Adult |
| 3 | Soda | Teen |
| 4 | Juice | Teen |
| 5 | Coffee | Adult |
| 6 | Soda | Teen |
| 7 | Coffee | Adult |
| 8 | Juice | Teen |
| 9 | Soda | Teen |
| 10 | Coffee | Adult |

Apply Naive Bayes Classifier to classify whether new Person is Teen or Adult whose favorite drink is "Juice".

# Ans:

Prior Probabilities:

- P(Teen) = 0.6
- P(Adult) = 0.4

Likelihoods:

- P(Favorite Drink = Soda | Teen) = 4/6 = 0.67
- P(Favorite Drink = Coffee | Teen) = 0/6 = 0.0
- P(Favorite Drink = Juice | Teen) = 2/6 = 0.33
- P(Favorite Drink = Soda | Adult) = 0/4 = 0.0
- P(Favorite Drink = Coffee | Adult) = 4/4 = 1.0
- P(Favorite Drink = Juice | Adult) = 0/4 = 0.0
- Classification: For a new person whose favorite drink is "Juice", the person is classified as "Teen" based on the higher posterior probability.

# Example

| Age Group | Buys Computer? |
|---|---|
| Youth | No |
| Youth | No |
| Middle | Yes |
| Senior | Yes |
| Senior | Yes |
| Senior | No |
| Middle | Yes |
| Youth | No |
| Youth | Yes |
| Senior | Yes |

Apply Naive Bayes for

Given a new instance

with an input feature

(e.g., Age Group = Youth),

calculate the probability

of each class

# Answer:

- P(Yes) = 5 / 10 = 0.5
- P(No) = 5 / 10 = 0.5
- P(Buys Computer = Yes | Age Group = Youth) = 0.1 / (0.1 + 0.3) = 0.25
- P(Buys Computer = No | Age Group = Youth) = 0.3 / (0.1 + 0.3) = 0.75

# Second Approach (In case of multiple features)

- **Step 1:** Convert the data set into a frequency tables (According to No of Input and Output)
- **Step 2:** Create Likelihood table by finding the probabilities ((According to No of Input and Output)
- **Step 3:** Apply Bayes Formula and calculate posterior probability for each target value

$$P(c|X) = P(x_1|c) \times P(x_2|c) \times \cdots \times P(x_n|c) \times P(c)$$

- **Step 4**: See which class has a higher probability, given the input belongs to the higher probability class.

# Second Approach (In case of multiple features)

| Email ID | Free (Feature 1) | Win (Feature 2) | Class |
|---|---|---|---|
| 1 | Yes | Yes | Spam |
| 2 | Yes | No | Spam |
| 3 | No | Yes | Not Spam |
| 4 | Yes | Yes | Not Spam |
| 5 | No | No | Not Spam |
| 6 | Yes | Yes | Spam |
| 7 | No | Yes | Spam |
| 8 | No | No | Not Spam |
| 9 | Yes | No | Not Spam |
| 10 | Yes | Yes | Spam |

- Apply Naive Bayes Classifier to find the class of New email classification (Free = Yes, Win = No).

# Example

- **Step 1:** Convert the data set into a frequency table

| Email ID | Free (Feature 1) | Win (Feature 2) | Class |
|---|---|---|---|
| 1 | Yes | Yes | Spam |
| 2 | Yes | No | Spam |
| 3 | No | Yes | Not Spam |
| 4 | Yes | Yes | Not Spam |
| 5 | No | No | Not Spam |
| 6 | Yes | Yes | Spam |
| 7 | No | Yes | Spam |
| 8 | No | No | Not Spam |
| 9 | Yes | No | Not Spam |
| 10 | Yes | Yes | Spam |

| Free (Feature 1) | Spam | Not Spam |
|---|---|---|
| Yes | 4 | 2 |
| No | 1 | 3 |
| Total | 5 | 5 |

| Win (Feature 2) | Spam | Not Spam |
|---|---|---|
| Yes | 4 | 2 |
| No | 1 | 3 |
| Total | 5 | 5 |

**Step 2:** Create Likelihood table by finding the probabilities

P(Spam) = 5 /10 = 0.5

P (Not Spam) = 5/10 = 0.5

| Free (Feature 1) | Spam | Not Spam | Posterior probabilities (Spam) | Posterior probabilities (Not Spam) |
|---|---|---|---|---|
| Yes | 4 | 2 | P(Free=Yes \| Spam) = 4/5 = 0.8 | P(Free=Yes \| Not Spam) = 2/5 = 0.4 |
| No | 1 | 3 | P(Free=No \| Spam) = 1/5 = 0.2 | P(Free=No \| Not Spam) = 3/5 = 0.6 |
| Total | 5 | 5 | | |

| Win (Feature 1) | Spam | Not Spam | Posterior probabilities (Spam) | Posterior probabilities (Not Spam) |
|---|---|---|---|---|
| Yes | 4 | 2 | P(Win=Yes \| Spam) = 4/5 = 0.8 | P(Win=Yes \| Not Spam) = 2/5 = 0.4 |
| No | 1 | 3 | P(Win=No \| Spam) = 1/5 = 0.2 | P(Win=No \| Not Spam) = 3/5 = 0.6 |
| Total | 5 | 5 | | |

**Step 3:** Apply Bayes Formula and calculate posterior probability for each target value

- Calculate the probability of Spam with New email classification with Free = Yes, Win = No.

P(Spam | Free = Yes, Win = No) = P(Free = Yes | Spam) * P(Win = No | Spam) * P(Spam)

= 0.8 * 0.2 * 0.5 = 0.08

- Calculate the probability of Not Spam with New email classification with Free = Yes, Win = No.

P(Not Spam | Free = Yes, Win = No)

= P(Free = Yes | Not Spam) * P(Win = No | Not Spam) * P(Not Spam)

= 0.4 * 0.6 * 0.5 = 0.12

# Contd.

- **Step 4**: See which class has a higher probability, given the input belongs to the higher probability class.
- P(Spam | Free = Yes, Win = No) = 0.08
- P(Not Spam | Free = Yes, Win = No) = 0.12
- Since 0.12 > 0.08, we classify the new email as "Not Spam".

Ans: Class of New email classification (Free = Yes, Win = No) is Not Spam

# Example:

Estimate conditional probabilities of each attributes {color, legs, height, smelly} for the species classes: {M, H} using the data given in the table.

Using these probabilities estimate the probability values for the new instance – (Color=Green, legs=2, Height=Tall, and Smelly=No).

| No | Color | Legs | Height | Smelly | Species |
|---|---|---|---|---|---|
| 1 | White | 3 | Short | Yes | M |
| 2 | Green | 2 | Tall | No | M |
| 3 | Green | 3 | Short | Yes | M |
| 4 | White | 3 | Short | Yes | M |
| 5 | Green | 2 | Short | No | H |
| 6 | White | 2 | Tall | No | H |
| 7 | White | 2 | Tall | No | H |
| 8 | White | 2 | Short | Yes | H |

| No | Color | Legs | Height | Smelly | Species |
|---|---|---|---|---|---|
| 1 | White | 3 | Short | Yes | M |
| 2 | Green | 2 | Tall | No | M |
| 3 | Green | 3 | Short | Yes | M |
| 4 | White | 3 | Short | Yes | M |
| 5 | Green | 2 | Short | No | H |
| 6 | White | 2 | Tall | No | H |
| 7 | White | 2 | Tall | No | H |
| 8 | White | 2 | Short | Yes | H |

**New Instance**

**(Color=Green, legs=2, Height=Tall, and Smelly=No)**

$$P(M) = \frac{4}{8} = 0.5 \quad P(H) = \frac{4}{8} = 0.5$$

| Color | M | H |
|---|---|---|
| White | 2/4 | 3/4 |
| Green | 2/4 | 1/4 |

| Legs | M | H |
|---|---|---|
| 2 | 1/4 | 4/4 |
| 3 | 3/4 | 0/4 |

| Height | M | H |
|---|---|---|
| Tall | 3/4 | 2/4 |
| Short | 1/4 | 2/4 |

| Smelly | M | H |
|---|---|---|
| Yes | 3/4 | 1/4 |
| No | 1/4 | 3/4 |

$$P(M) = \frac{4}{8} = 0.5 \quad P(H) = \frac{4}{8} = 0.5$$

| Color | M | H |
|---|---|---|
| White | 2/4 | 3/4 |
| Green | 2/4 | 1/4 |

| Legs | M | H |
|---|---|---|
| 2 | 1/4 | 4/4 |
| 3 | 3/4 | 0/4 |

| Height | M | H |
|---|---|---|
| Tall | 3/4 | 2/4 |
| Short | 1/4 | 2/4 |

| Smelly | M | H |
|---|---|---|
| Yes | 3/4 | 1/4 |
| No | 1/4 | 3/4 |

$$p(M|New\ Instance) = p(M) * p(Color = Green|M) * p(Legs = 2|M) * p(Height = tall|M) * p(Smelly = no\ |M)$$

$$p(M|New\ Instance) = 0.5 * \frac{2}{4} * \frac{1}{4} * \frac{3}{4} * \frac{1}{4} = 0.0117$$

$$p(H|New\ Instance) = p(H) * p(Color = Green|H) * p(Legs = 2|H) * p(Height = tall|H) * p(Smelly = no\ |H)$$

$$p(H|New\ Instance) = 0.5 * \frac{1}{4} * \frac{4}{4} * \frac{2}{4} * \frac{3}{4} = 0.047$$

$p(H|New\ Instance) > p(M|New\ Instance)$

**Hence the new instance belongs to Speices H**

# Example

| Example 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 |

$$\boldsymbol{New\ Instance} = (\boldsymbol{Red}, \boldsymbol{SUV}, \boldsymbol{Domestic}) \rightarrow (\boldsymbol{Yes\ or\ No})$$

# Ans:

| Example 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 |

$$p(Yes) = \frac{5}{10} = 0.5$$

$$p(No) = \frac{5}{10} = 0.5$$

| Color | Yes | No |
|---|---|---|
| Red | 3/5 | 2/5 |
| Yellow | 2/5 | 3/5 |

| Type | Yes | No |
|---|---|---|
| Sports | 4/5 | 2/5 |
| SUV | 1/5 | 3/5 |

| Origin | Yes | No |
|---|---|---|
| Domestic | 2/5 | 3/5 |
| Imported | 3/5 | 2/5 |

$$New\ Instance = (Red, SUV, Domestic)$$

$$P(Yes|New\ Instance) = p(Yes) * P(Color = Red|Yes) * P(Type = SUV|Yes) * P(Origin = Domestic|Yes)$$

$$P(Yes|New\ Instance) = \frac{5}{10} * \frac{3}{5} * \frac{1}{5} * \frac{2}{5} = \frac{3}{125} = 0.024$$

$$P(No|New\ Instance) = p(No) * P(Color = Red|No) * P(Type = SUV|No) * P(Origin = Domestic|No)$$

$$P(No|New\ Instance) = \frac{5}{10} * \frac{2}{5} * \frac{3}{5} * \frac{3}{5} = \frac{9}{125} = 0.072$$

$$P(No|New\ Instance) > P(Yes|New\ Instance)$$

# Example

| Day | *Outlook* | *Temperature* | *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 |

$\langle Outlook = sunny, Temperature = cool, Humidity = high, Wind = strong \rangle$

Ans:

$$P(PlayTennis = yes) = 9/14 = .64$$

$$P(PlayTennis = no) = 5/14 = .36$$

| Outlook | Y | N |
|---|---|---|
| sunny | 2/9 | 3/5 |
| overcast | 4/9 | 0 |
| rain | 3/9 | 2/5 |
| Tempreature | | |
| hot | 2/9 | 2/5 |
| mild | 4/9 | 2/5 |
| cool | 3/9 | 1/5 |

| Humidity | Y | N |
|---|---|---|
| high | 3/9 | 4/5 |
| normal | 6/9 | 1/5 |
| | | |
| Windy | | |
| Strong | 3/9 | 3/5 |
| Weak | 6/9 | 2/5 |

# Contd.

$$\langle Outlook = sunny, Temperature = cool, Humidity = high, Wind = strong \rangle$$

$$v_{NB} = \underset{v_j \in \{yes,no\}}{\text{argmax}} \; P(v_j) \prod_i P(a_i|v_j)$$

$$= \underset{v_j \in \{yes,no\}}{\text{argmax}} \; P(v_j) \quad P(Outlook = sunny|v_j) P(Temperature = cool|v_j)$$

$$\cdot \; P(Humidity = high|v_j) P(Wind = strong|v_j)$$

$$v_{NB}(yes) = P(yes) \; P(sunny|yes) \; P(cool|yes) \; P(high|yes) \; P(strong|yes) = .0053$$

$$v_{NB}(no) = P(no) \; P(sunny|no) \; P(cool|no) \; P(high|no) \; P(strong|no) = .0206$$

$$v_{NB}(yes) = \frac{v_{NB}(yes)}{v_{NB}(yes) + v_{NB}(no)} = 0.205 \qquad v_{NB}(no) = \frac{v_{NB}(no)}{v_{NB}(yes) + v_{NB}(no)} = 0.795$$

# DEMO

---

# Naive Bayes Classifier

- https://colab.research.google.com/drive/1Yj4t10E7jqxLkGTJeNc-aV7_8epgY3iR?usp=sharing

# Advantage & Disadvantage of Naive Bayes Classifier

**Advantages of Naïve Bayes Classifier:**

- Naïve Bayes is one of the fast and easy ML algorithms to predict a class of datasets.
- It can be used for Binary as well as Multi-class Classifications.
- It performs well in Multi-class predictions as compared to the other Algorithms.
- It is the most popular choice for **text classification problems**.

**Disadvantages of Naïve Bayes Classifier:**

- Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

# Decision Tree

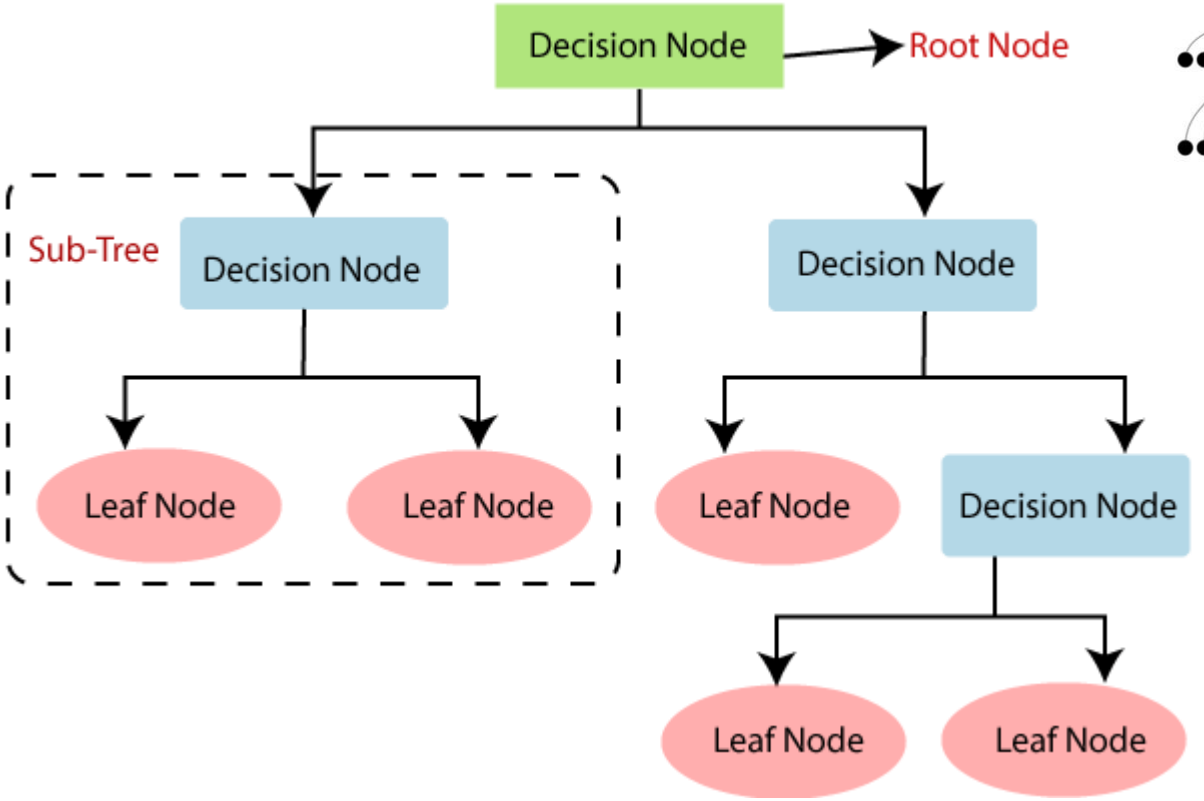
Introduction, Decision Tree Terminologies, Algorithm - Attribute Selection Measures (Information Gain & Entropy , Gini Index),ID3 Algorithm & Examples, Advantages & Disadvanatages

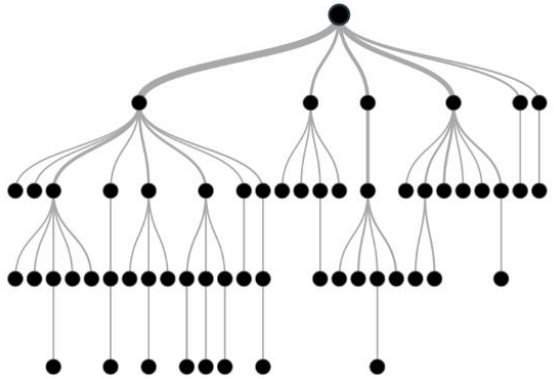
# Decision Tree

- Decision Tree is a **Supervised learning technique** that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems.
- It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.
- It has a hierarchical tree structure consisting of a root node, branches, internal nodes, and leaf nodes.
- It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.

# Contd.

- It is a tree-structured classifier, where **internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome.**



# Decision Tree Terminologies

- Root Node: Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.
- Leaf Node: Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.
- Splitting: Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.
- Branch/Sub Tree: A tree formed by splitting the tree.
- Pruning: Pruning is the process of removing the unwanted branches from the tree.
- Parent/Child node: The root node of the tree is called the parent node, and other nodes are called the child nodes.

# Algorithm

- **Step-1:** Begin the tree with the root node, says S, which contains the complete dataset.
- **Step-2:** Find the best attribute in the dataset using **Attribute Selection Measure (ASM).**
- **Step-3:** Divide the S into subsets that contains possible values for the best attributes.
- **Step-4:** Generate the decision tree node, which contains the best attribute.
- **Step-5:** Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

# Attribute Selection Measures

- While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as **Attribute selection measure or ASM.**
- By this measurement, we can easily select the best attribute for the nodes of the tree.
- There are two popular techniques for ASM, which are:
  - **Information Gain & Entropy**
  - **Gini Index**

# Information Gain

- Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.
- It **calculates how much information a feature provides us about a class.**
- According to the value of information gain, we split the node and build the decision tree.
- A decision tree algorithm always tries to **maximize the value of information gain**, and a node/attribute having the highest information gain is split first. It can be calculated using the below formula:

**IG= Entropy(S) - [(Weighted Avg) *Entropy(each feature)]**

# Entropy

- Entropy is a metric to **measure the impurity in a given attribute**. It specifies randomness in data. Entropy can be calculated as:

**Entropy(s)= -P(yes)log2 P(yes)- P(no) log2 P(no)**

- Where,
- S= Total number of samples
- P(yes)= probability of yes
- P(no)= probability of no

# Gini Index

- Gini index is a measure of impurity or purity used while creating a decision tree in the CART(Classification and Regression Tree) algorithm.
- An attribute with the low Gini index should be preferred as compared to the high Gini index.

$$\text{Gini Index} = 1 - \sum_j P_j^2$$

# ID3 Algorithm:

- **Step-1:** Begin the tree with the root node, says S, which contains the complete dataset.
- **Step-2:** Find the best attribute in the dataset using **Attribute Selection Measure (ASM) – Consider Highest value of Information Gain Feature as best Feature**
- **Step-3:** Divide the S into subsets that contains possible values for the best attributes.
- **Step-4:** Generate the decision tree node, which contains the best attribute.
- **Step-5:** Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

# Example:

Draw decision tree for the given dataset.

| Instance | Classification | a1 | a2 |
|---|---|---|---|
| 1 | + | T | T |
| 2 | + | T | T |
| 3 | - | T | F |
| 4 | + | F | F |
| 5 | - | F | T |
| 6 | - | F | T |

| Instance | Classification | a1 | a2 |
|---|---|---|---|
| 1 | + | T | T |
| 2 | + | T | T |
| 3 | - | T | F |
| 4 | + | F | F |
| 5 | - | F | T |
| 6 | - | F | T |

**Attribute: a1**

$Values\,(a1) = T, F$

$S = [3+, 3-]$ $\quad Entropy(S) = 1.0$

$S_T = [2+, 1-]$ $\quad Entropy(S_T) = -\frac{2}{3}log_2\frac{2}{3} - \frac{1}{3}log_2\frac{1}{3} = 0.9183$

$S_F \leftarrow [1+, 2-]$ $\quad Entropy(S_F) = -\frac{1}{3}log_2\frac{1}{3} - \frac{2}{3}log_2\frac{2}{3} = 0.9183$

$$Gain\,(S, a1) = Entropy(S) - \sum_{v \in \{T,F\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, a1) = Entropy(S) - \frac{3}{6}Entropy(S_T) - \frac{3}{6}Entropy(S_F)$$

$$Gain(S, a1) = 1.0 - \frac{3}{6} * 0.9183 - \frac{3}{6} * 0.9183 = 0.0817$$

| Instance | Classification | a1 | a2 |
|---|---|---|---|
| 1 | + | T | T |
| 2 | + | T | T |
| 3 | - | T | F |
| 4 | + | F | F |
| 5 | - | F | T |
| 6 | - | F | T |

**Attribute: a2**

$Values\ (a2) = T, F$

$S = [3+, 3-]$ $\quad Entropy(S) = 1.0$

$S_T = [2+, 2-]$ $\quad Entropy(S_T) = 1.0$

$S_F \leftarrow [1+, 1-]$ $\quad Entropy(S_F) = 1.0$

$$Gain\ (S, a2) = Entropy(S) - \sum_{v \in \{T,F\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

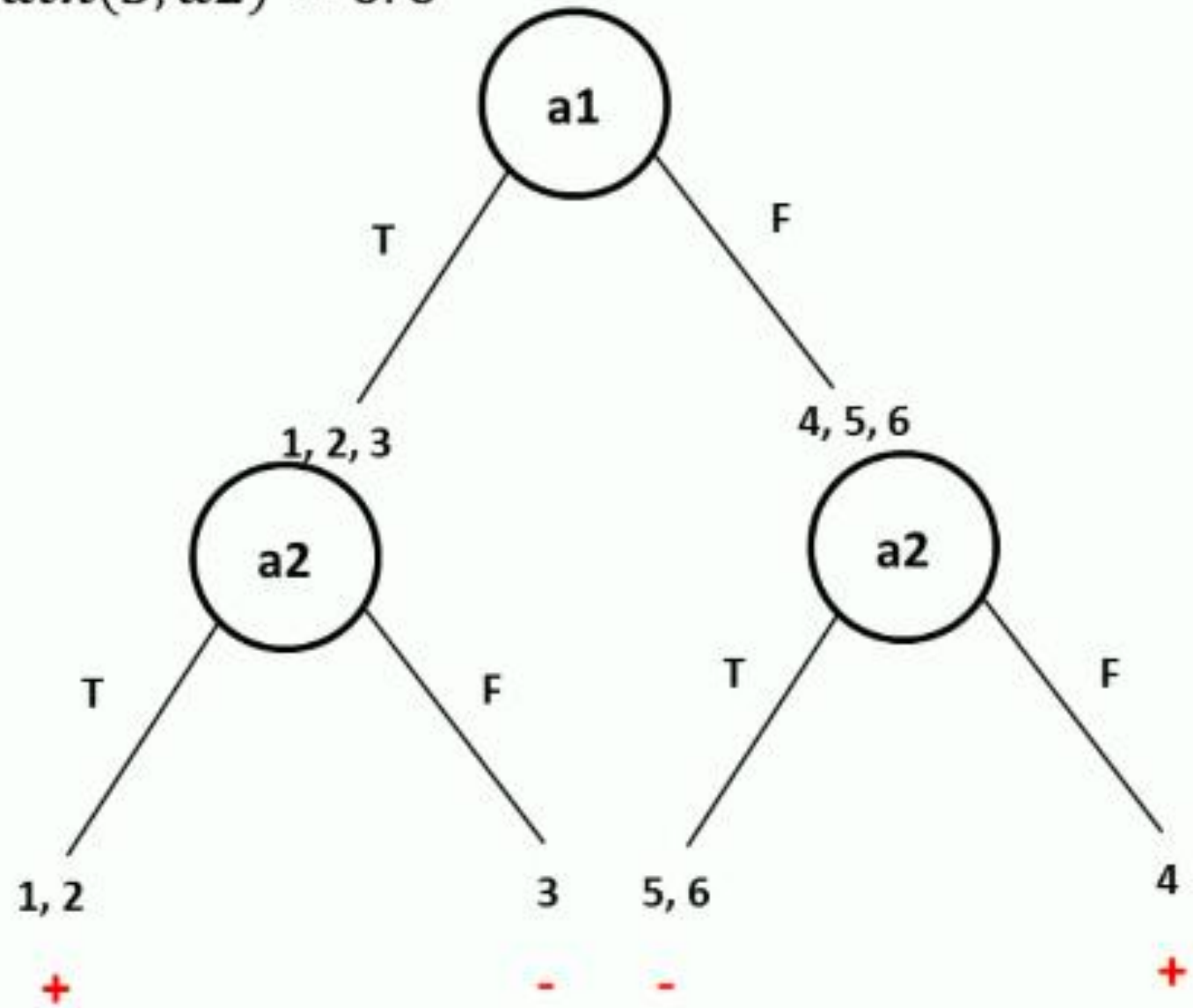
$$Gain(S, a2) = Entropy(S) - \frac{4}{6} Entropy(S_T) - \frac{2}{6} Entropy(S_F)$$

$$Gain(S, a2) = 1.0 - \frac{4}{6} * 1.0 - \frac{2}{6} * 1.0 = 0.0$$

| Instance | Classification | a1 | a2 |
|---|---|---|---|
| 1 | + | T | T |
| 2 | + | T | T |
| 3 | - | T | F |
| 4 | + | F | F |
| 5 | - | F | T |
| 6 | - | F | T |

$$Gain(S, a1) = 0.0817 \quad - \quad Maximum\ Gain$$

$$Gain(S, a2) = 0.0$$



# Example:

| Temperature | Humidity | Play |
|---|---|---|
| T | T | Yes |
| T | T | No |
| F | T | Yes |
| F | F | No |
| T | F | Yes |
| T | F | No |
| T | F | No |

Draw decision tree for the given dataset.

# Ans:

- ??????

# Example:

Draw decision tree for the given dataset.

| Instance | a1 | a2 | a3 | Classification |
|---|---|---|---|---|
| 1 | True | Hot | High | No |
| 2 | True | Hot | High | No |
| 3 | False | Hot | High | Yes |
| 4 | False | Cool | Normal | Yes |
| 5 | False | Cool | Normal | Yes |
| 6 | True | Cool | High | No |
| 7 | True | Hot | High | No |
| 8 | True | Hot | Normal | Yes |
| 9 | False | Cool | Normal | Yes |
| 10 | False | Cool | High | Yes |

| Instance | a1 | a2 | a3 | Classification |
|---|---|---|---|---|
| 1 | True | Hot | High | No |
| 2 | True | Hot | High | No |
| 3 | False | Hot | High | Yes |
| 4 | False | Cool | Normal | Yes |
| 5 | False | Cool | Normal | Yes |
| 6 | True | Cool | High | No |
| 7 | True | Hot | High | No |
| 8 | True | Hot | Normal | Yes |
| 9 | False | Cool | Normal | Yes |
| 10 | False | Cool | High | Yes |

**Attribute: a1**

$$Values(a1) = True,\ False$$

$$S = [6+,\ 4-] \qquad Entropy(S) = -\frac{6}{10}log_2\frac{6}{10} - \frac{4}{10}log_2\frac{4}{10} = 0.9709$$

$$S_{True} = [1+, 4-] \qquad Entropy(S_{True}) = -\frac{1}{5}log_2\frac{1}{5} - \frac{4}{5}log_2\frac{4}{5} = 0.7219$$

$$S_{Flase} \leftarrow [5+,\ 0-] \qquad Entropy(S_{False}) = 0.0$$

$$Gain\ (S, a1) = Entropy(S) - \sum_{v \in \{True, False\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, a1) = Entropy(S) - \frac{5}{10}Entropy(S_{True}) - \frac{5}{10}Entropy(S_{False})$$

$$Gain(S, a1) = 0.9709 - \frac{5}{10} * 0.7219 - \frac{5}{10} * 1 = 0.6099$$

| Instance | a1 | a2 | a3 | Classification |
|---|---|---|---|---|
| 1 | True | Hot | High | No |
| 2 | True | Hot | High | No |
| 3 | False | Hot | High | Yes |
| 4 | False | Cool | Normal | Yes |
| 5 | False | Cool | Normal | Yes |
| 6 | True | Cool | High | No |
| 7 | True | Hot | High | No |
| 8 | True | Hot | Normal | Yes |
| 9 | False | Cool | Normal | Yes |
| 10 | False | Cool | High | Yes |

**Attribute: a2**

$$Values(a2) = Hot,\ Cool$$

$$S = [6+,\ 4-] \qquad Entropy(S) = -\frac{6}{10}log_2\frac{6}{10} - \frac{4}{10}log_2\frac{4}{10} = 0.9709$$

$$S_{Hot} = [2+, 3-] \qquad Entropy(S_{Hot}) = -\frac{2}{5}log_2\frac{2}{5} - \frac{3}{5}log_2\frac{3}{5} = 0.9709$$

$$S_{Cool} \leftarrow [4+,\ 1-] \qquad Entropy(S_{Cool}) = -\frac{4}{5}log_2\frac{4}{5} - \frac{1}{5}log_2\frac{1}{5} = 0.7219$$

$$Gain\ (S, a2) = Entropy(S) - \sum_{v \in \{Hot, Cool\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, a2) = Entropy(S) - \frac{5}{10}Entropy(S_{Hot}) - \frac{5}{10}Entropy(S_{Cool})$$

$$Gain(S, a2) = 0.9709 - \frac{5}{10} * 0.9709 - \frac{5}{10} * 0.7219 = 0.1245$$

| Instance | a1 | a2 | a3 | Classification |
|---|---|---|---|---|
| 1 | True | Hot | High | No |
| 2 | True | Hot | High | No |
| 3 | False | Hot | High | Yes |
| 4 | False | Cool | Normal | Yes |
| 5 | False | Cool | Normal | Yes |
| 6 | True | Cool | High | No |
| 7 | True | Hot | High | No |
| 8 | True | Hot | Normal | Yes |
| 9 | False | Cool | Normal | Yes |
| 10 | False | Cool | High | Yes |

**Attribute: a3**

$$Values\,(a3) = High,\ Normal$$

$$S = [6+,\ 4-] \qquad Entropy(S) = -\frac{6}{10}log_2\frac{6}{10} - \frac{4}{10}log_2\frac{4}{10} = 0.9709$$

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

$$S_{Normal} \leftarrow [4+,\ 0-] \qquad Entropy(S_{Normal}) = 0.0$$

$$Gain\,(S, a3) = Entropy(S) - \sum_{v \in \{High, Normal\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, a3) = Entropy(S) - \frac{6}{10}Entropy(S_{High}) - \frac{4}{10}Entropy(S_{Normal})$$

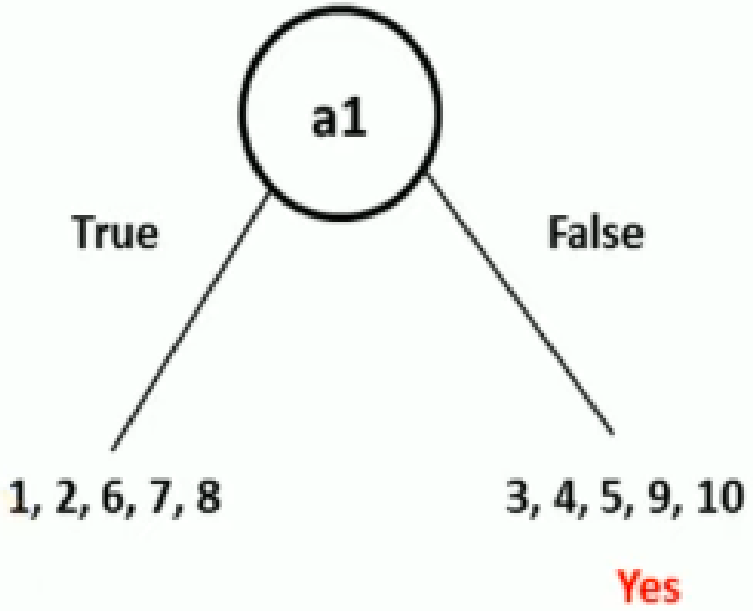
$$Gain(S, a3) = 0.9709 - \frac{6}{10} * 0.9183 - \frac{4}{10} * 0.0 = 0.4199$$

| Instance | a1 | a2 | a3 | Classification |
|---|---|---|---|---|
| 1 | True | Hot | High | No |
| 2 | True | Hot | High | No |
| 3 | False | Hot | High | Yes |
| 4 | False | Cool | Normal | Yes |
| 5 | False | Cool | Normal | Yes |
| 6 | True | Cool | High | No |
| 7 | True | Hot | High | No |
| 8 | True | Hot | Normal | Yes |
| 9 | False | Cool | Normal | Yes |
| 10 | False | Cool | High | Yes |

$$Gain(S, a1) = 0.6099 \quad - \quad Maximum\ Gain$$

$$Gain(S, a2) = 0.1245$$

$$Gain(S, a3) = 0.4199$$



## Attribute: a2

| Instance | a2 | a3 | Classification |
|---|---|---|---|
| 1 | Hot | High | No |
| 2 | Hot | High | No |
| 6 | Cool | High | No |
| 7 | Hot | High | No |
| 8 | Hot | Normal | Yes |

$$Values\,(a2) = Hot,\ Cool$$

$$S_{a1} = [1+,\ 4-] \qquad Entropy(S_{a1}) = -\frac{1}{5}log_2\frac{1}{5} - \frac{4}{5}log_2\frac{4}{5} = 0.7219$$

$$S_{Hot} = [1+, 3-] \qquad Entropy(S_{Hot}) = -\frac{1}{4}log_2\frac{1}{4} - \frac{3}{4}log_2\frac{3}{4} = 0.8112$$

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

$$Gain\,(S, a2) = Entropy(S) - \sum_{v \in \{Hot, Cool\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, a2) = Entropy(S) - \frac{4}{5}Entropy(S_{Hot}) - \frac{1}{5}Entropy(S_{Cool})$$

$$Gain(S, a2) = 0.9709 - \frac{4}{5} * 0.8112 - \frac{1}{5} * 0.0 = 0.3219$$

| Instance | a2 | a3 | Classification |
|---|---|---|---|
| 1 | Hot | High | No |
| 2 | Hot | High | No |
| 6 | Cool | High | No |
| 7 | Hot | High | No |
| 8 | Hot | Normal | Yes |

## Attribute: a3

$$Values\,(a3) = High,\ Normal$$

$$S_{a1} = [1+,\ 4-] \qquad Entropy(S_{a1}) = -\frac{1}{5}log_2\frac{1}{5} - \frac{4}{5}log_2\frac{4}{5} = 0.7219$$

$$S_{High} = [0+, 4-] \qquad Entropy(S_{High}) = 0.0$$

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

$$Gain\,(S, a3) = Entropy(S) - \sum_{v \in \{High, Normal\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

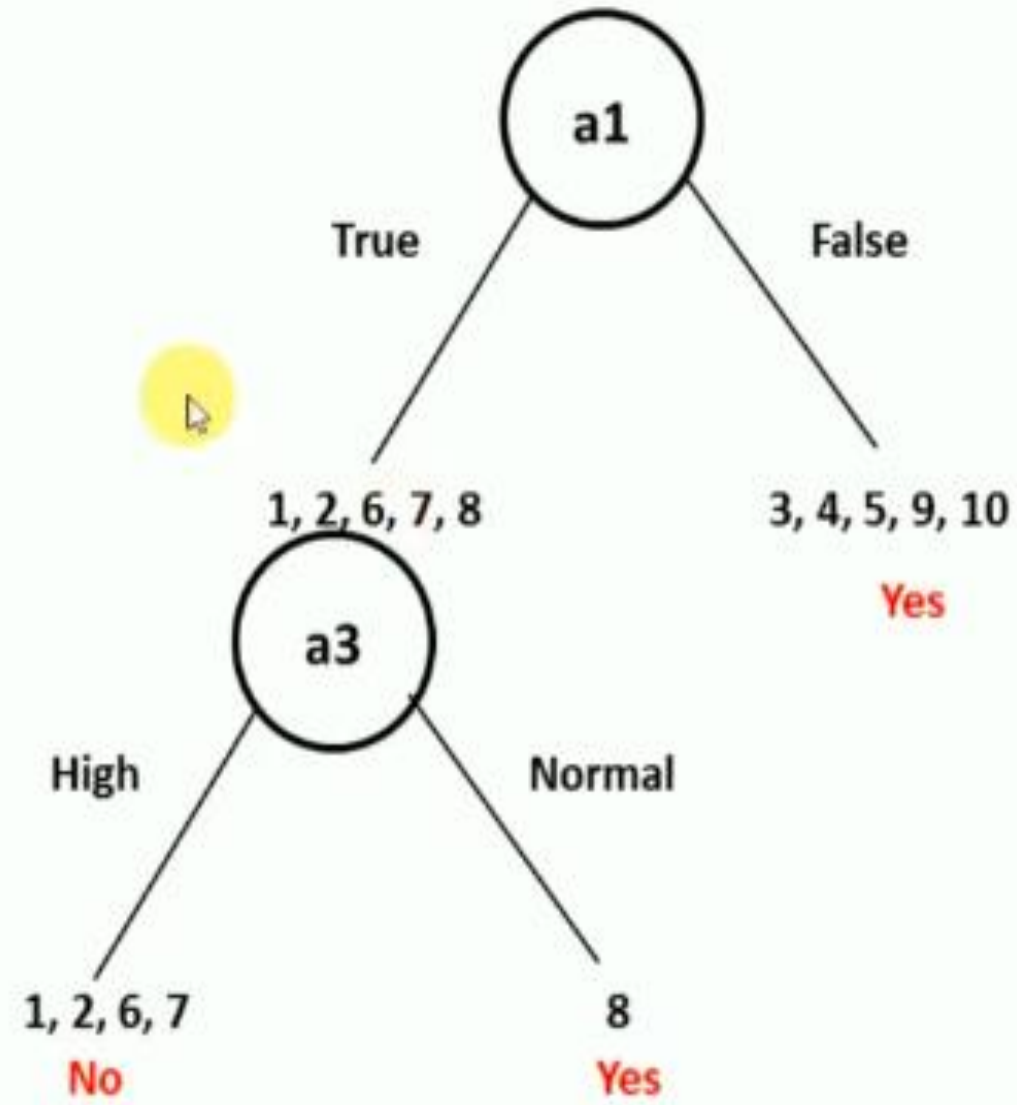
$$Gain(S, a3) = Entropy(S) - \frac{4}{5}Entropy(S_{High}) - \frac{1}{5}Entropy(S_{Normal})$$

$$Gain(S, a3) = 0.9709 - \frac{4}{5} * 0.0 - \frac{1}{5} * 0.0 = 0.7219$$

$$Gain(S_{a1}, a2) = 0.3219$$

$$Gain(S_{a1}, a3) = 0.7219 - \textit{Maximum Gain}$$

| Instance | a2 | a3 | Classification |
|---|---|---|---|
| 1 | Hot | High | No |
| 2 | Hot | High | No |
| 6 | Cool | High | No |
| 7 | Hot | High | No |
| 8 | Hot | Normal | Yes |

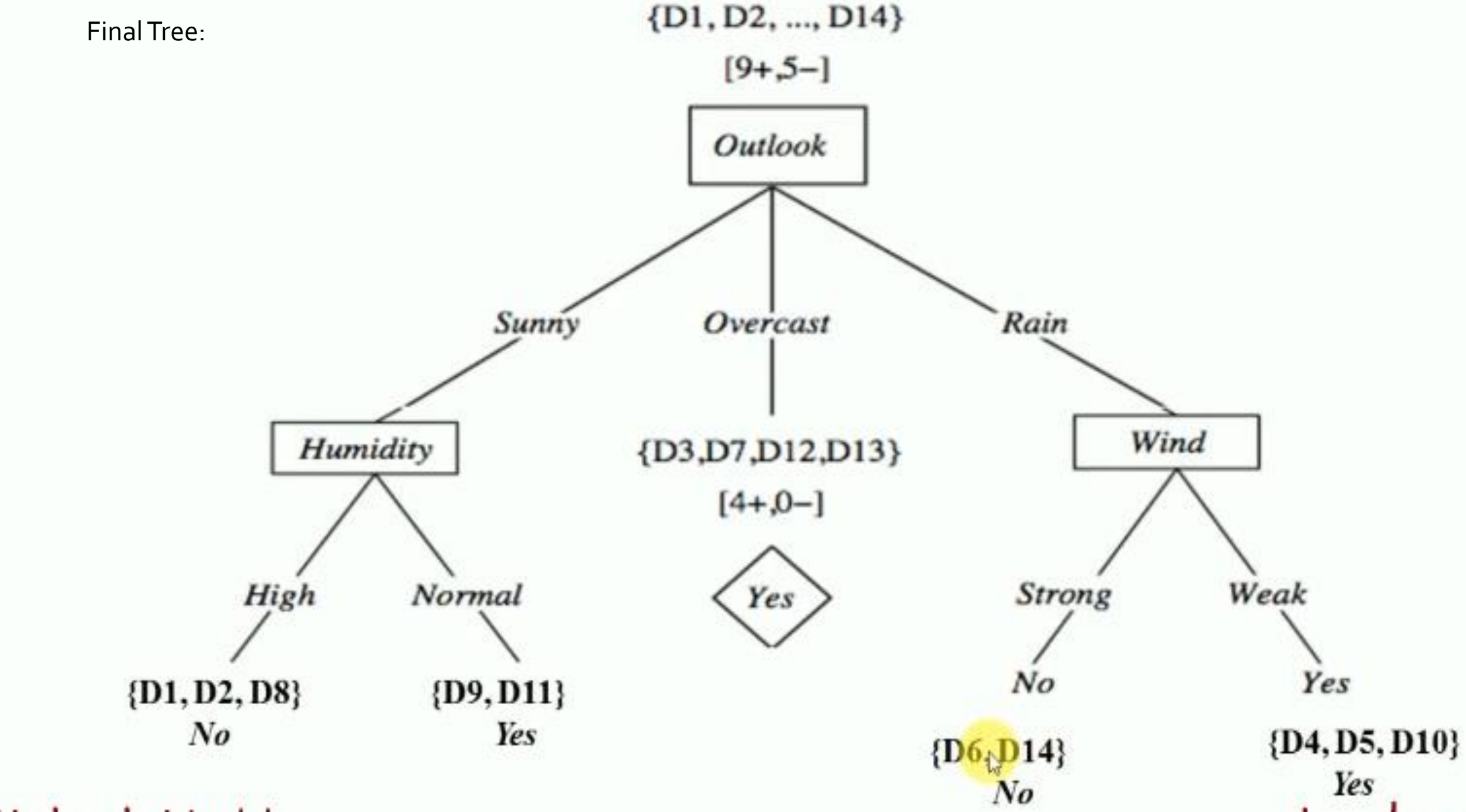


# Example:

Draw decision tree for the given dataset.

| 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 |

Final Tree:


{D1, D2, ..., D14}
[9+,5−]
Outlook
Sunny
Overcast
Rain
Humidity
{D3,D7,D12,D13}
[4+,0−]
Wind
High
Normal
Yes
Strong
Weak
{D1, D2, D8}
No
{D9, D11}
Yes
No
Yes
{D6, D14}
No
{D4, D5, D10}
Yes

# DEMO

---

# Decision Tree

- https://colab.research.google.com/drive/1Yj4t10E7jqxLkGTJeNc-aV7_8epgY3iR?usp=sharing

# Advantage & Disadvantage of Decision tress

**Advantages of the Decision Tree**

- It is simple to understand as it follows the same process which a human follow while making any decision in real-life.
- It can be very useful for solving decision-related problems.
- It helps to think about all the possible outcomes for a problem.
- There is less requirement of data cleaning compared to other algorithms.

**Disadvantages of the Decision Tree**

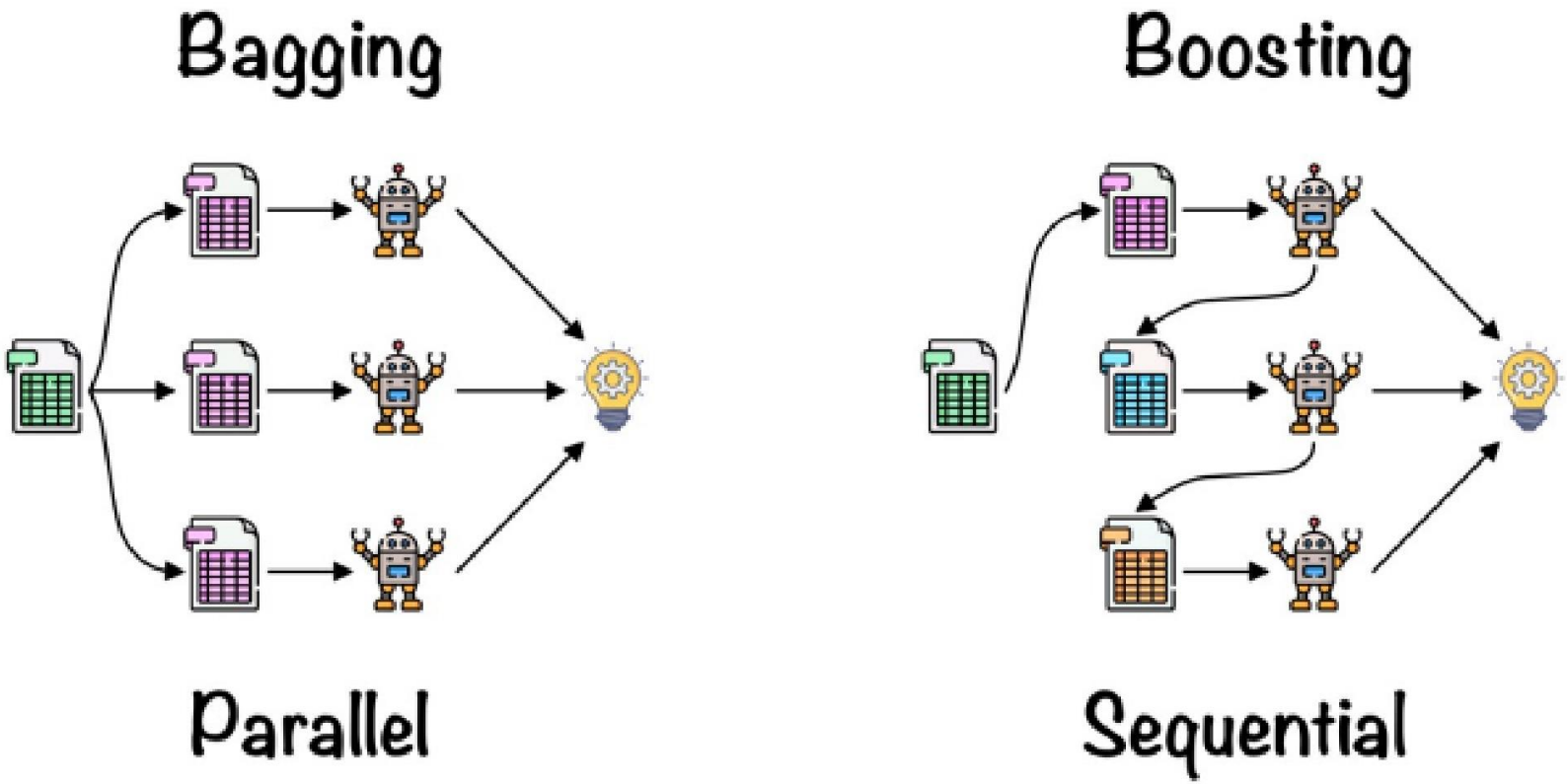
- The decision tree contains lots of layers, which makes it complex.
- It may have an overfitting issue, which can be resolved using the **Random Forest algorithm.**
- For more class labels, the computational complexity of the decision tree may increase.

# Ensemble Methods

Bagging – Random Forest Algorithm, Boosting – XGBoost

# Ensemble Methods

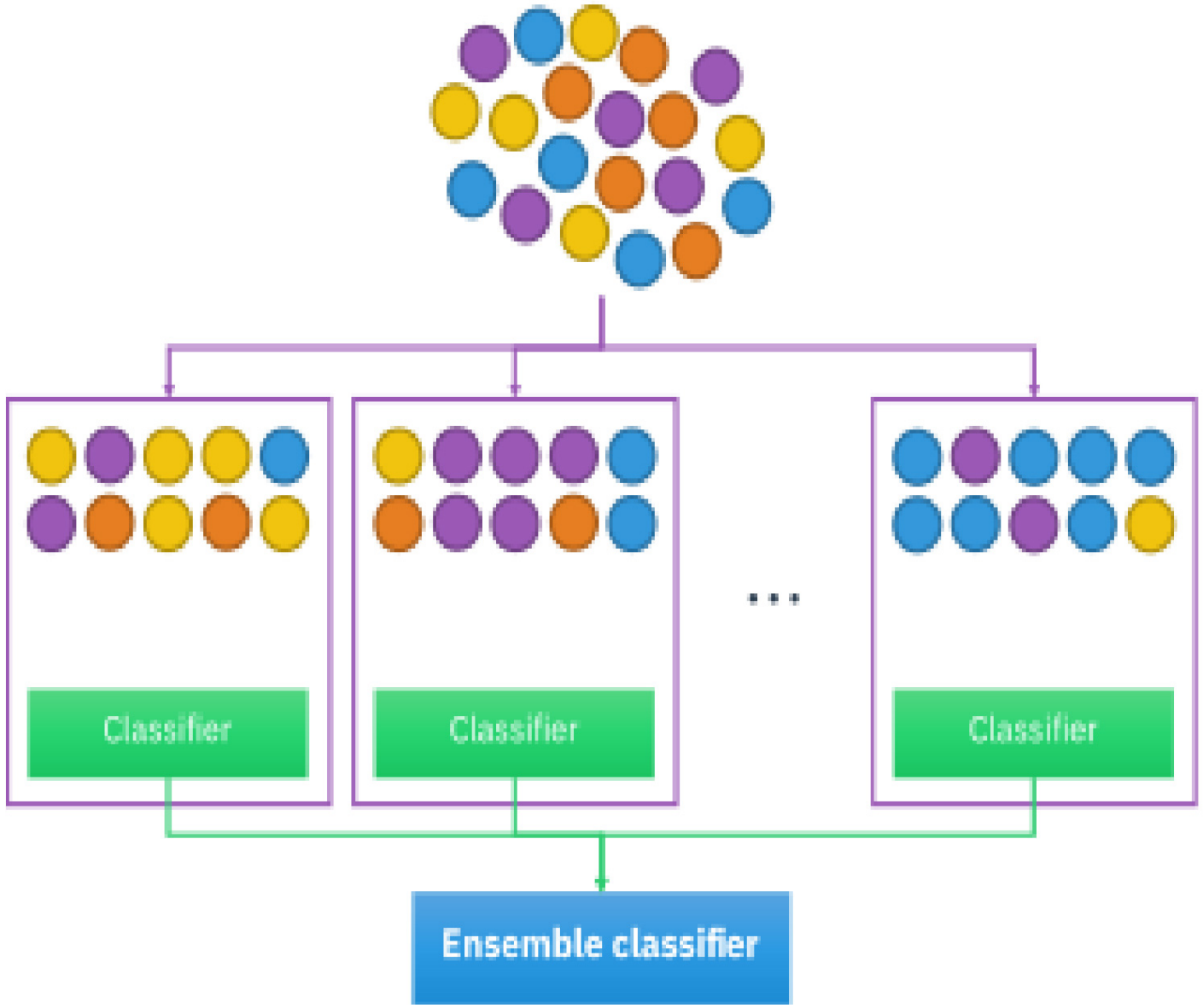
- ***Ensemble*** simply means combining multiple models.
- Thus a collection of models is used to make predictions rather than an individual model.
- Ensemble uses two types of methods:



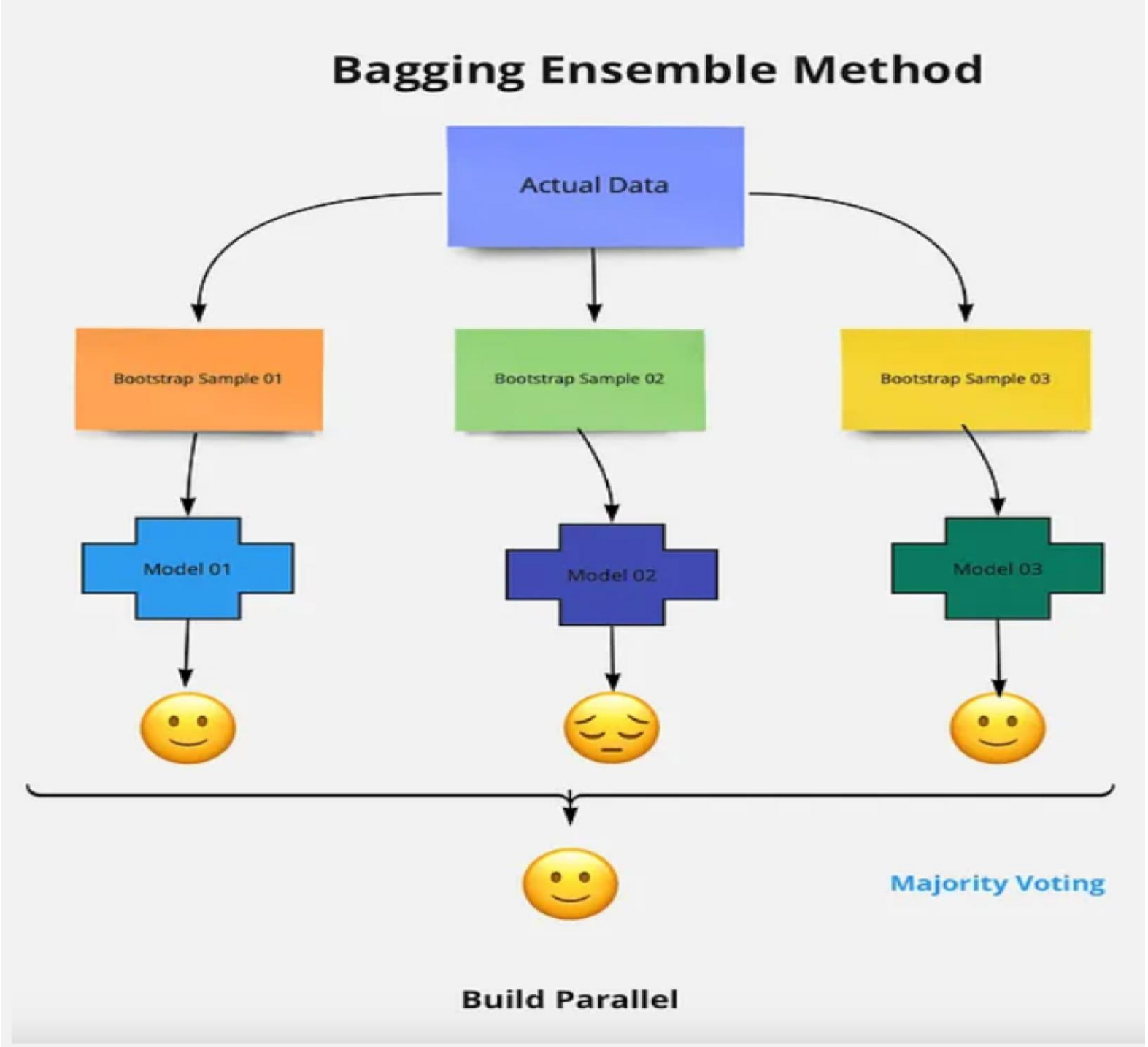
# Bagging

- Bagging, also known as Bootstrap Aggregation, serves as the ensemble technique in the Random Forest algorithm. Here are the steps involved in Bagging:
- **Selection of Subset**: Bagging starts by choosing a random sample, or subset, from the entire dataset.
- **Bootstrap Sampling**: Each model is then created from these samples, called Bootstrap Samples, which are taken from the original data with replacement. This process is known as row sampling.
- **Bootstrapping**: The step of row sampling with replacement is referred to as bootstrapping.
- **Independent Model Training**: Each model is trained independently on its corresponding Bootstrap Sample. This training process generates results for each model.
- **Majority Voting**: The final output is determined by combining the results of all models through majority voting. The most commonly predicted outcome among the models is selected.
- **Aggregation**: This step, which involves combining all the results and generating the final output based on majority voting, is known as aggregation.

# Bagging


Original Data
Classifier
Classifier
Classifier
Bootstrapping
Aggregating
Ensemble classifier
Bagging

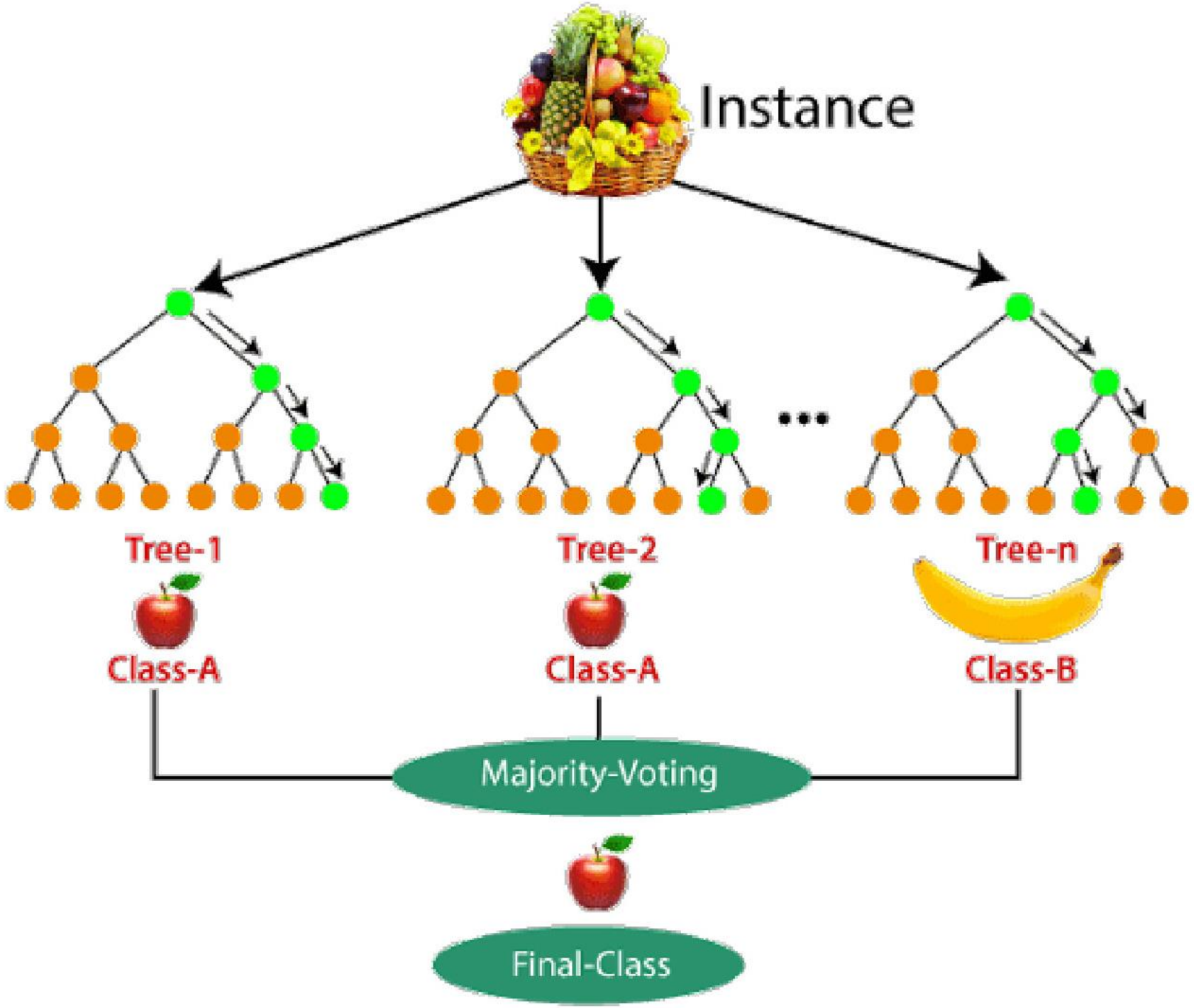
# Bagging


Bagging Ensemble Method
Actual Data
Bootstrap Sample 01
Bootstrap Sample 02
Bootstrap Sample 03
Model 01
Model 02
Model 03
Majority Voting
Build Parallel

# Random Forest

- **Step 1:** In the Random forest model, a subset of data points and a subset of features is selected for constructing each decision tree. Simply put, n random records and m features are taken from the data set having k number of records.
- **Step 2:** Individual decision trees are constructed for each sample.
- **Step 3:** Each decision tree will generate an output.
- **Step 4:** Final output is considered based on ***Majority Voting or Averaging*** for Classification and regression, respectively.

# Contd.


Instance
Tree-1
Tree-2
Tree-n
Class-A
Class-A
Class-B
Majority-Voting
Final-Class

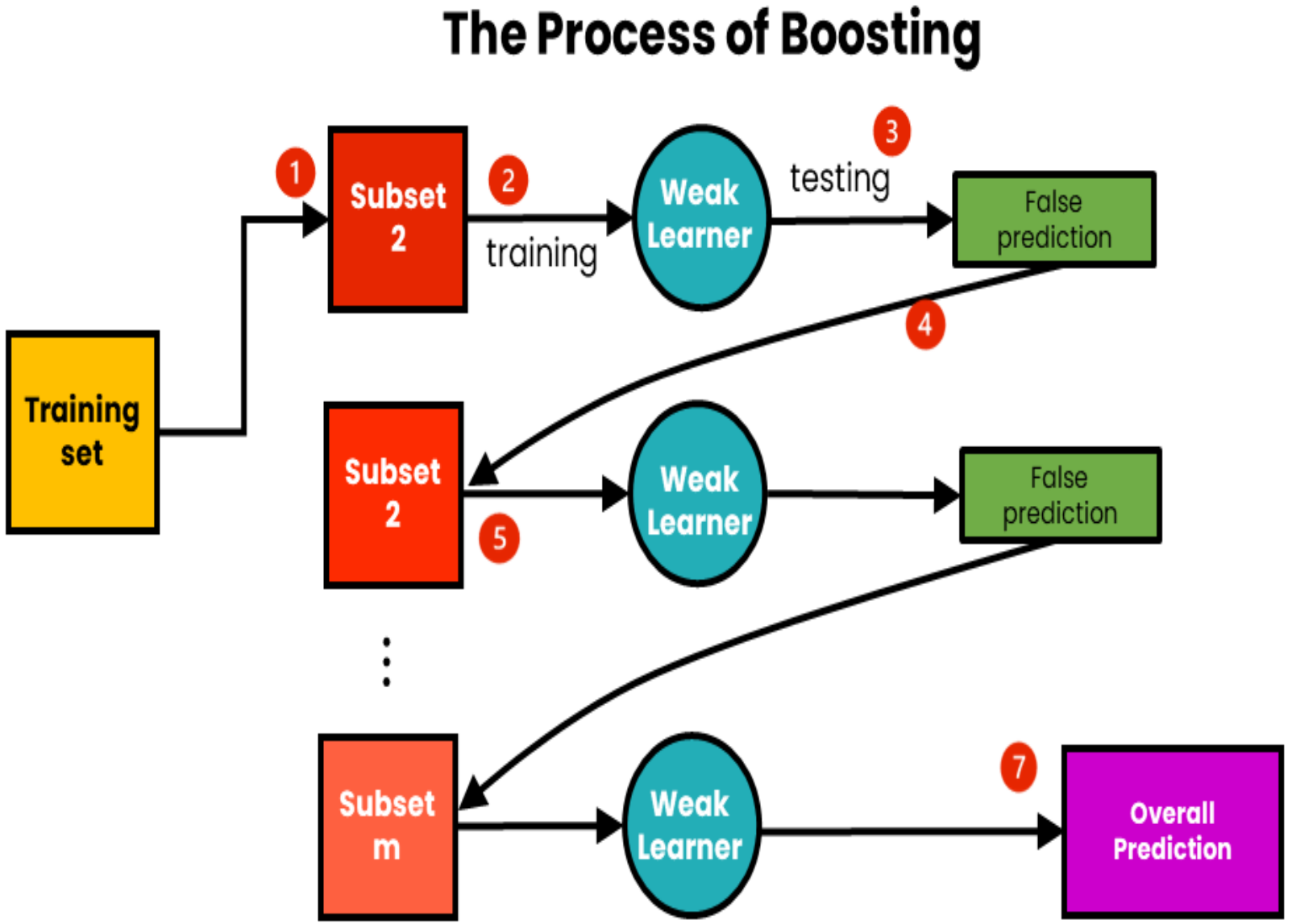
# Contd.

- Example: Consider the fruit basket as the data as shown in the figure below. Now n number of samples are taken from the fruit basket, and an individual decision tree is constructed for each sample. Each decision tree will generate an output, as shown in the figure. The final output is considered based on majority voting. In the below figure, you can see that the majority decision tree gives output as an apple when compared to a banana, so the final output is taken as an apple.

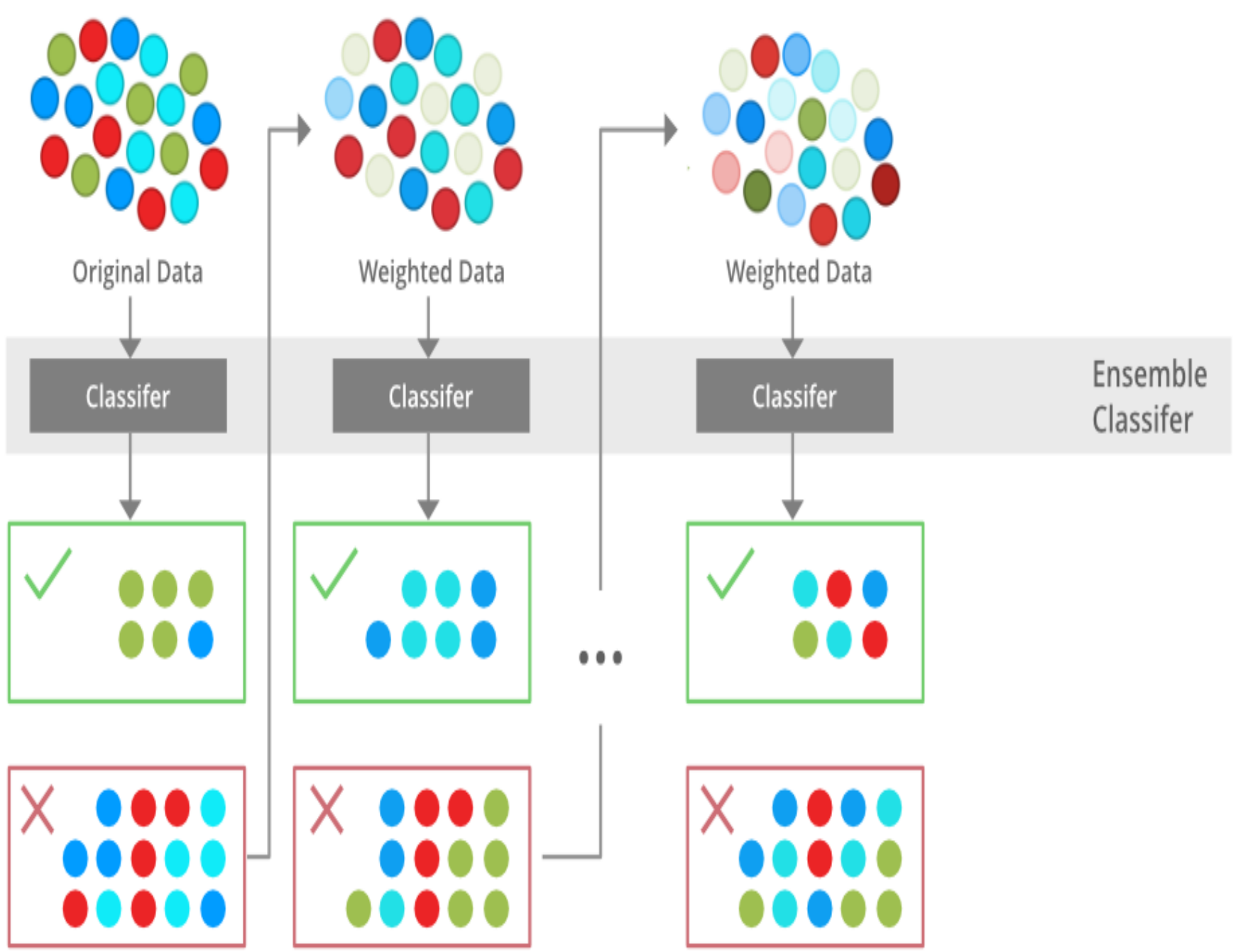
# Boosting

- Boosting is one of the techniques that use the concept of ensemble learning.
- A boosting algorithm combines multiple simple models (also known as weak learners or base estimators) to generate the final output.
- It is done by building a model by using weak models in series.
- There are several boosting algorithms; AdaBoost - Adaptive Boosting, Gradient Boosting Machine (GBM), Extreme Gradient Boosting Machine (XGBM)

# Contd.


The Process of Boosting
Training set
1
Subset 2
2
training
Weak Learner
3
testing
False prediction
4
Subset 2
5
Weak Learner
False prediction
Subset m
Weak Learner
7
Overall Prediction

# Contd.


Original Data
Weighted Data
Weighted Data
Classifer
Classifer
Classifer
Ensemble
Classifer

# Contd.

- You've built a linear regression model that gives you a decent 77% accuracy on the validation dataset.
- Next, you decide to expand your portfolio by building a k-Nearest Neighbour (KNN) model and a decision tree model on the same dataset. These models gave you an accuracy of 62% and 89% on the validation set respectively.
- It's obvious that all three models work in completely different ways. For instance, the linear regression model tries to capture linear relationships in the data while the decision tree model attempts to capture the non-linearity in the data.
- How about, instead of using any one of these models for making the final predictions, we use a combination of all of these models?
- I'm thinking of an average of the predictions from these models. By doing this, we would be able to capture more information from the data, right?

# DEMO

---

Ensemble Learning (Random Forest, XGBoost)

- https://colab.research.google.com/drive/1Yj4t10E7jqxLkGTJeNc-aV7_8epgY3iR?usp=sharing

# Any Queries..??

Thank you