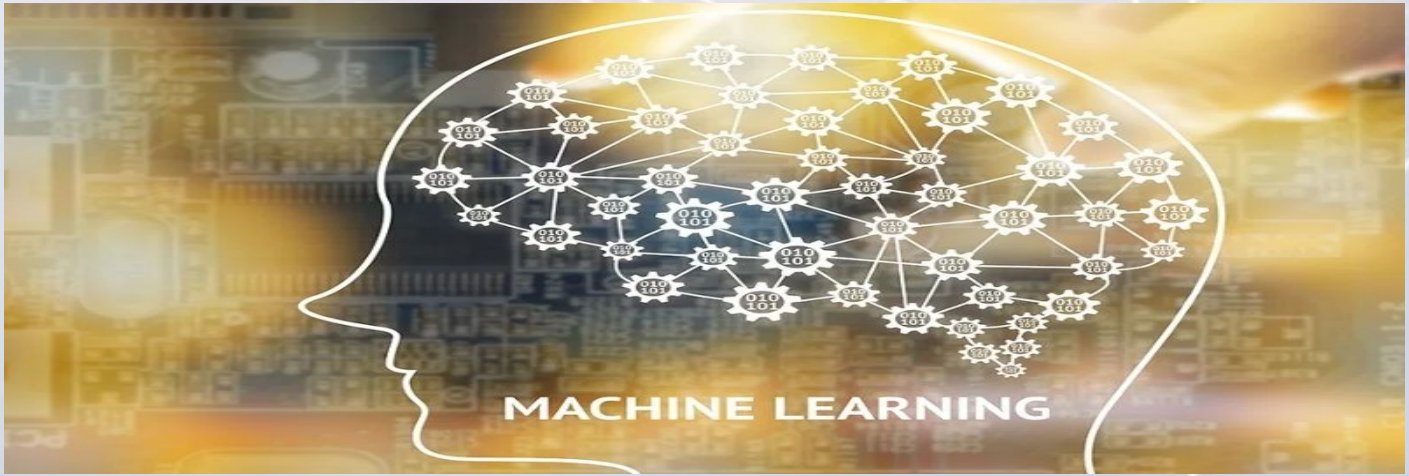
Machine Learning



Unit II

Supervised Learning (Regression/Classification)
Basic methods: Distance-based methods, Nearest-Neighbors,Decision Trees, Naive Bayes, Linear models: Linear Regression, Logistic Regression, Generalized Linear Models, Support Vector Machines, Nonlinearity and Kernel Methods, Beyond Binary Classification: Multiclass/Structured Outputs,Ranking

Unsupervised Learning Clustering:
K-means/Kernel K-means, Dimensionality Reduction -PCA, CCA, LDA, ICA, MNF – Canonical Variates - Feature Selection vs Feature Extraction, Generative Models (mixture models and latent factor models)

Classification & Regression

1. Classification - Discrete Valued Output (ie is this email spam or not spam)
2. Regression - Continuous Valued Output / Real valued output

Supervised Learning: Linear Regression

- Linear regression analysis is used to **predict the value of a variable based on the value of another variable.**
- The variable you want to predict is called the **dependent variable.** The variable you are using to predict the other variable's value is called the **independent variable**.
- Linear regression fits a **straight line** or surface that minimizes the discrepancies between predicted and actual output values.
- There are simple linear regression calculators that use a “least squares” method to discover the best-fit line for a set of paired data. You can then estimate the value of X (dependent variable) from Y (independent variable).
- Data: Dependent and independent variables should be **quantitative**. Categorical variables, such as religion, major field of study or region of residence, need to be recoded to binary (dummy) variables or other types of contrast variables.

Linear Regression

- **Apply Linear Regression in case Data meets following assumptions:**
  - The variables should be measured at a continuous level. Examples of continuous variables are time, sales, weight and test scores.
  - Use a scatter plot to find out quickly if there is a linear relationship between those two variables.
  - Your data should have no significant outliers.
  - Check for homoscedasticity — a statistical concept in which the variances along the best-fit linear-regression line remain similar all through that line.
  - The residuals (errors) of the best-fit regression line follow normal distribution.

Linear Regression

- **Evaluating trends and sales estimates**

  to predict a salesperson's total yearly sales (the dependent variable) from independent variables such as age, education and years of experience.

- **Analyze pricing elasticity**

  Changes in pricing often impact consumer behavior.
  If the price of a particular product keeps changing, you can use regression analysis to see whether consumption drops as the price increases. What if consumption does not drop significantly as the price increases? At what price point do buyers stop purchasing the product? This information would be very helpful for leaders in a retail business.

- **Sports analysis**

  The number of games won by a basketball team in a season is related to the average number of points the team scores per game. A scatterplot indicates that these variables are linearly related. The number of games won and the average number of points scored by the opponent are also linearly related. These variables have a negative relationship. As the number of games won increases, the average number of points scored by the opponent decreases. With linear regression, you can model the relationship of these variables. A good model can be used to predict how many games teams will win.

Linear Regression

Hypothesis:

$h(x) = \theta_0 + \theta_1 x$

Parameters:

θ_0 , θ_1

Cost Function:

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$$

Gradient Descent

$$Repeat\ \{$$

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

$$\}$$

α : Learning Rate

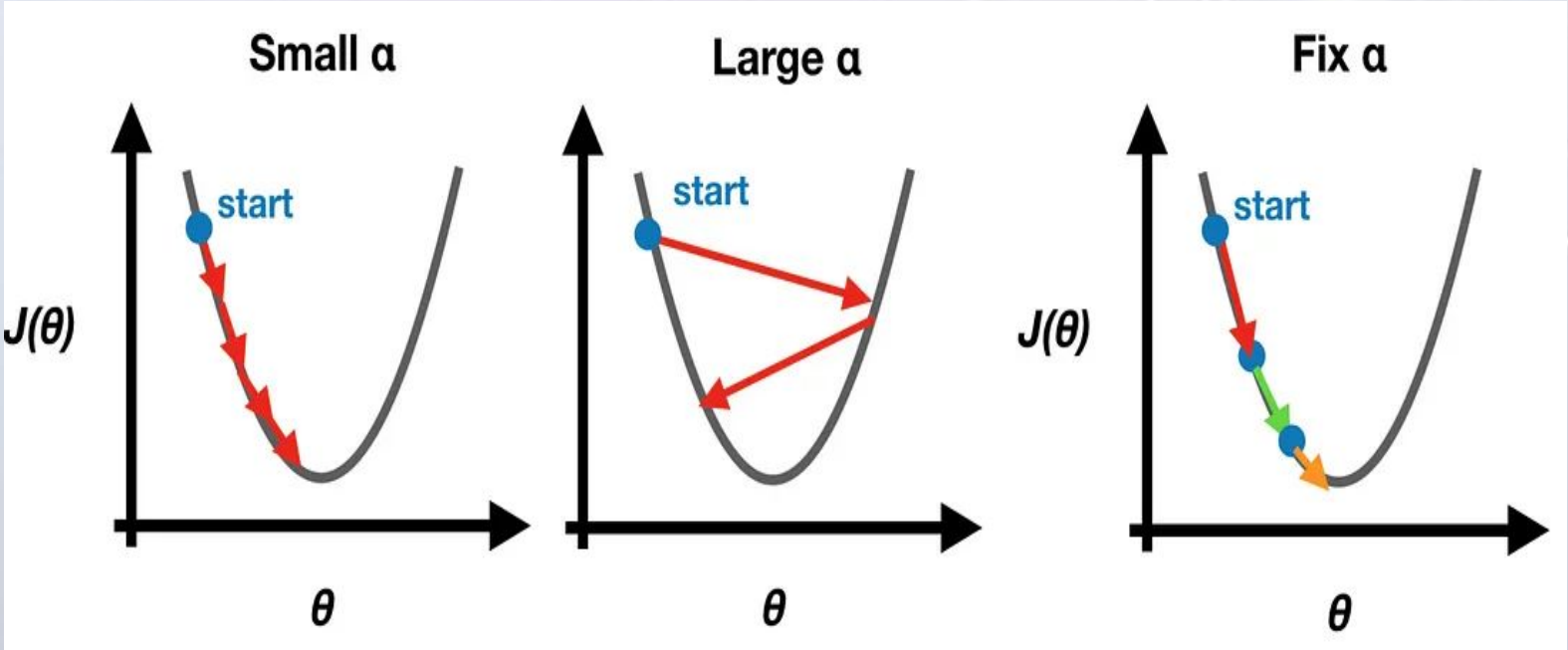
$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} ((h_\theta(x_i) - y_i)x_i)$$

- If the learning rate alpha is too small, gradient descent can be slow.
- If learning rate alpha is too large, it may not converge at all, or even diverge.

Learning Rate

- If the learning rate alpha is too small, gradient descent can be slow.
- If learning rate alpha is too large, it may not converge at all, or even diverge.



Multivariable Linear Regression

- *It is Linear Regression with multiple features.*
- Following is the example of house price prediction (data, features, label) and its hypothesis

$$h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_n x_n$$

Parameters: $\theta = \{\theta_0, \theta_1, \theta_2, \ldots . \theta_n\}$

Features: $x = \{x_0, x_1, x_2, \ldots . x_n\}$

$$h_\theta(x) = \theta_0 + 2104\theta_1 + 5\theta_2 + \theta_3 + 45\theta_4 = 460$$
$$h_\theta(x) = \theta_0 + 1416\theta_1 + 3\theta_2 + 2\theta_3 + 40\theta_4 = 232$$
$$h_\theta(x) = \theta_0 + 1534\theta_1 + 2\theta_2 + 2\theta_3 + 30\theta_4 = 315$$

	Size ($feet^2$) x_1	Number of bedrooms x_2	Number of floors x_3	age of home (years) x_4	Price($1000) $h_\theta(x) = y$
A data	2104	5	1	45	460
	1416	3	2	40	232
	1534	2	2	30	315
	…	…	…	…	…

Features (x) — Label ($h_\theta(x) = y$)

Multivariable Linear Regression

- When tuning the hypothesis, our model learns parameters θ which makes hypothesis function h(x) a 'good' predictor. A Good predictor means the hypothesis is closed enough to the true model.
- But how do we know the hypothesis h(x) is good enough or not? The answer is using a cost function, which we defined to measure the error that the hypothesis made.
- In Regression Problem, the most popular cost function is **Mean Error Square(MSE).** The formulation is defined as below.

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left(\underbrace{h_\theta(x^{(i)})}_{\text{Predicted value}}, \underbrace{y^{(i)}}_{\text{True value}} \right)^2$$

Square Error of data i

Multivariable Linear Regression

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}), y^{(i)}\right)^2$$

Square Error of data i

$h_\theta(x^{(i)})$: Predicted value

$y^{(i)}$: True value

- The cost function J(θ) is the sum of the square error of each data. The larger the error, the worser the performance of the hypothesis. Therefore, we want to minimize the error, that is, minimize the J(θ).
-
- To minimize the error, we will use **gradient descent algorithm**.

Repeat until converge {

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

} where j represents the feature index number.

Feature scaling

Feature Scaling is a technique to standardize the independent features present in the data in a fixed range. It is performed during the data pre-processing.

Working:

Given a data-set with features- Age, Salary, BHK Apartment with the data size of 5000 people, each having these independent data features.

Each data point is labeled as:

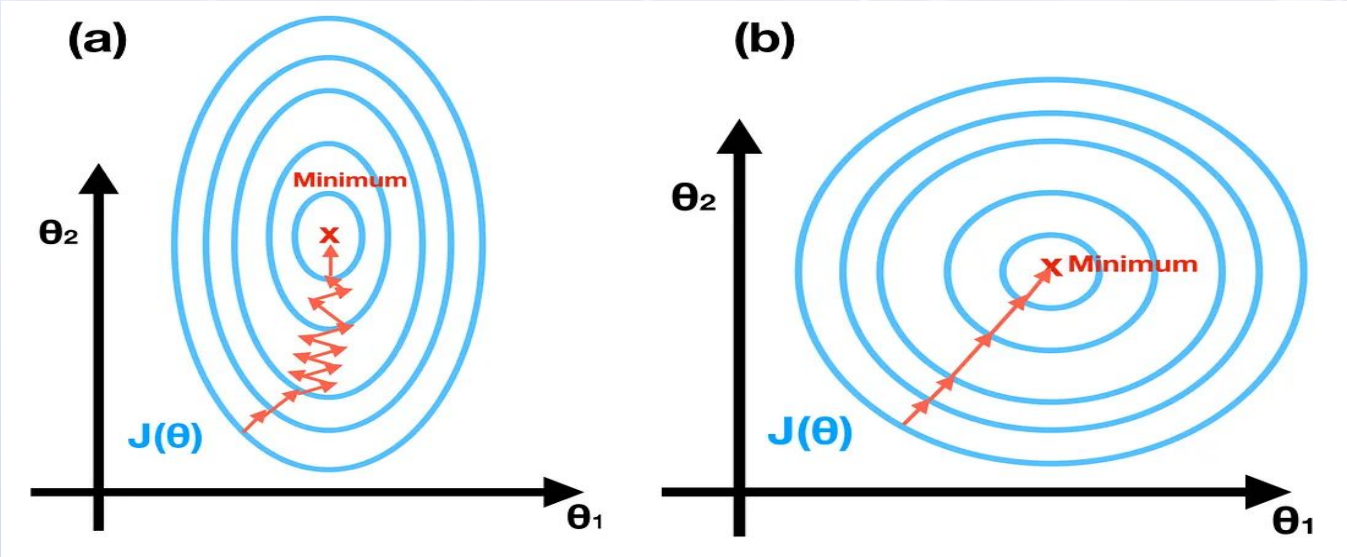
Class 1 - YES (means with the given Age, Salary, BHK Apartment feature value one can buy the property)

Class 2 - NO (means with the given Age, Salary, BHK Apartment feature value one can't buy the property).

Using a dataset to train the model, one aims to build a model that can predict whether one can buy a property or not with given feature values.

Feature scaling

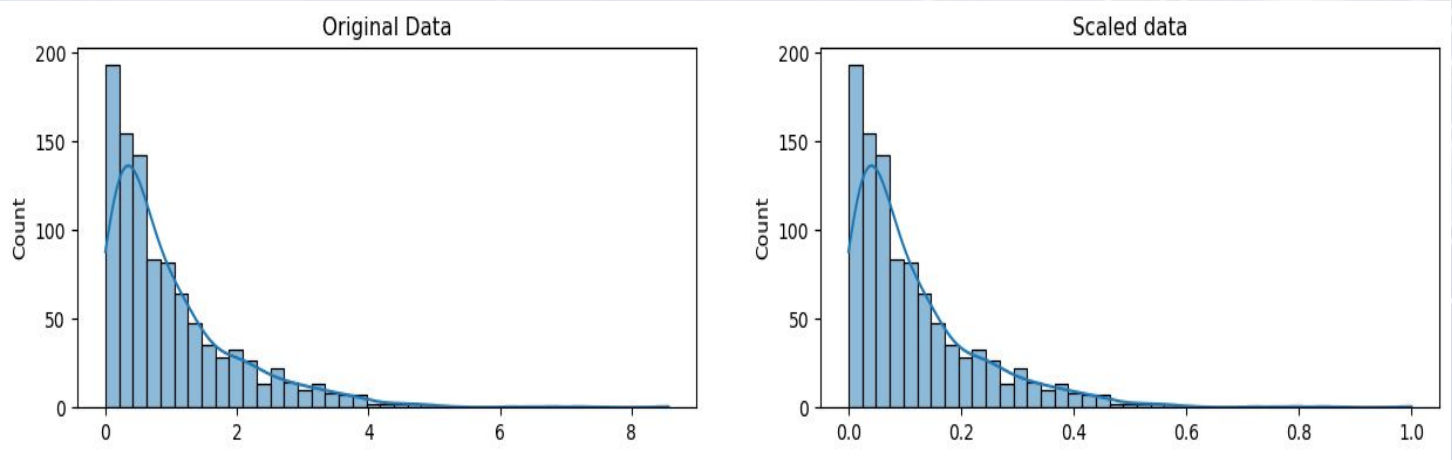
- Contour plot according to gradient descent



(a) Before Feature Scaling (b) After Feature Scaling

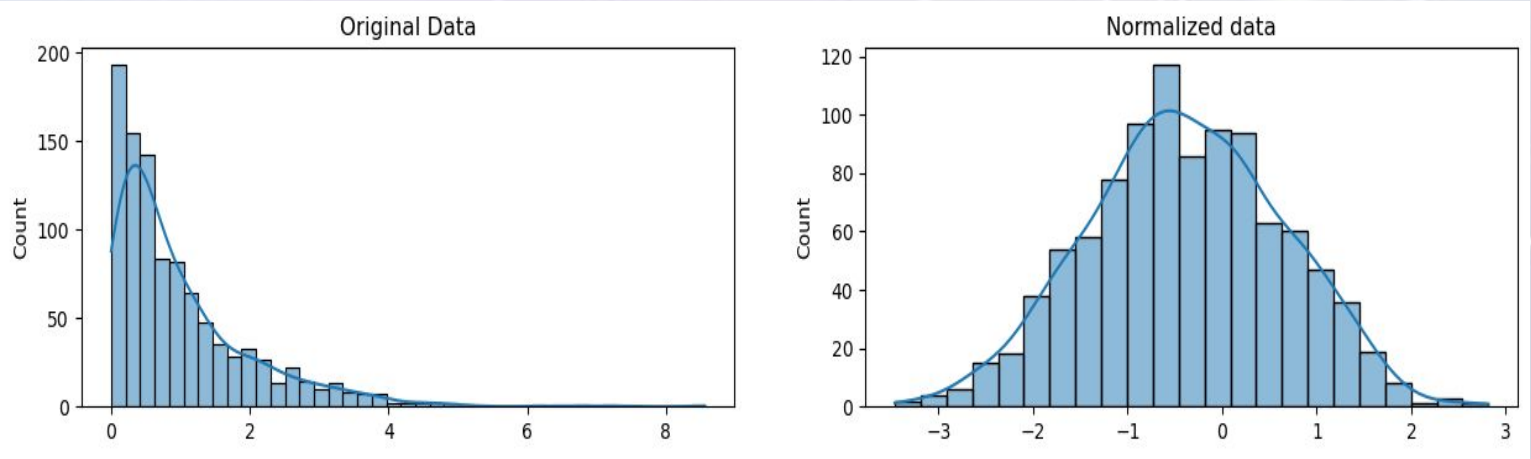
Feature scaling & Normalization

- Feature scaling is transforming data, so that it fits within a specific scale, like 0-100 or 0-1



Feature scaling & Normalization

- **Normalization** is to change observations so that they can be described as a normal distribution (bell curve).



Logistic Regression

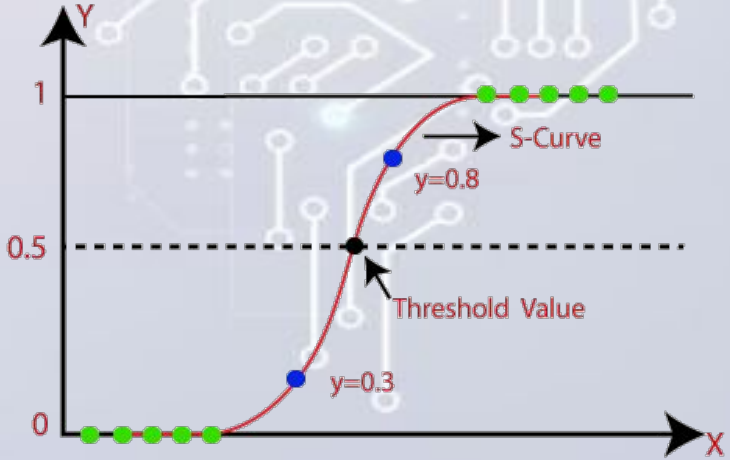
- Logistic regression is used in classification problems where the labels are a discrete number of classes as compared to linear regression, where labels are continuous variables.
- Logistic regression **hypothesis** is defined as:

$$h_\theta(x)=g(\theta^T x)$$

where function g is the sigmoid function.

- The **sigmoid function** is defined as:

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1+e^{-\theta^T x}},$$



Logistic Regression

- Cost Function:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}\log(h_\theta(x^{(i)})) + (1-y^{(i)})\log(1-h_\theta(x^{(i)}))]$$

- Gradient

$$\theta_j := \theta_j + \alpha\left(y^{(i)} - h_\theta(x^{(i)})\right)x_j^{(i)}$$

Logistic Regression Example

- The data set of pass or fail in exam of 5 students are given below table and Use logistic regression to classify the data
  - Calculate the probability of pass for the student who studied for 33 hours.
  - At Least how many hours student should study that make him pass the course with probability of more than 95%
  - Assume the model suggested by the optimizer for odds of passing the course is,

    log(odds) = -64 + 2* hours

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

Logistic Regression Example

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

   Use Sigmoid to calculate the probability: $s(x) = 1 / 1 + e^{-x}$

   Here, log(odds) = x = -64 + 2* hours = -64 + 2 * 33 = 2

   $s(x) = 1 / 1 + e^{-2} = 0.88$

   88% of pass for the student who studied for 33 hours.

2. At Least how many hours student should study that make him pass the course with probability of more than 95%

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

   $0.95 = 1 / 1 + e^{-x}$

   $e^{-x} = 0.0526$

   $-X = -2.94$

   log(odds) = x = -64 + 2* hours

   -2.94 = -64 + 2 * hours

   Hours = 33. 47 hours

Distance Based Methods

- Examples of Distance Based Methods
  - K-Nearest Neighbors
  - Learning Vector Quantization (LVQ)
  - Self-Organizing Map (SOM)
  - K-Means Clustering

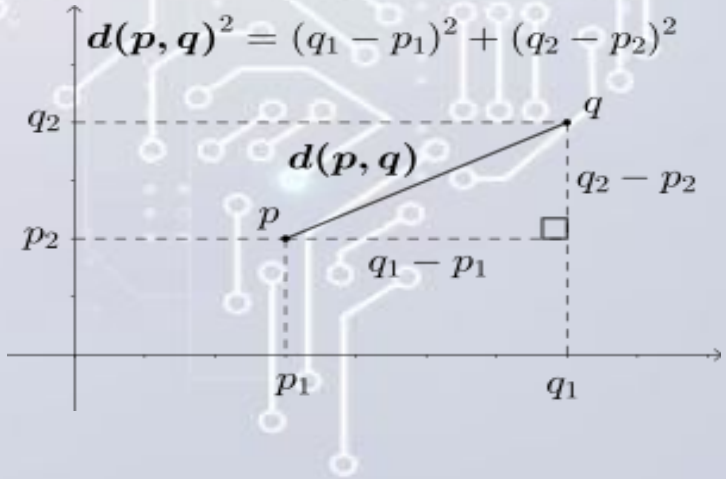
- Types of Distance
  - Hamming Distance
  - Euclidean Distance
  - Manhattan Distance
  - Minkowski Distance

Types of Distance

- **Hamming Distance**
  - Hamming distance between two strings (binary) of equal length is the number of positions at which the corresponding symbols are different.
  - Integer coding (natural) and one-hot encoding (binary)
    - String 1 = 1 0 1 1
    - String 2 = 1 1 0 1
    - Hamming Distance = 2

- **Euclidean Distance**
  - The Euclidean distance between two points in Euclidean space is the length of a line segment between the two points. It can be calculated from the Cartesian coordinates of the points using the Pythagorean theorem, therefore occasionally being called the Pythagorean distance.

row1 = [10, 20, 15, 10, 5]
row2 = [12, 24, 18, 8, 7]
Euclidean Distance =



Types of Distance

- **Manhattan Distance**
  - Manhattan distance is a distance measure that is calculated by taking the sum of distances between the x and y coordinates. The Manhattan distance is also known as Manhattan length.

  - Manhattan Distance = sum for i to N sum |v1[i] – v2[i]|
  - row1 = [10, 20, 15, 10, 5]
  - row2 = [12, 24, 18, 8, 7]

- **Minkowski Distance**
  - The Minkowski distance or Minkowski metric is a metric in a normed vector space which can be considered as a generalization of both the Euclidean distance and the Manhattan distance.

    Minkowski Distance = (sum for i to N (abs(v1[i] – v2[i]))^p)^(1/p)
    Where “p” is the order parameter.

    When p is set to 1, the calculation is the same as the Manhattan distance. When p is set to 2, it is the same as the Euclidean distance.

- p=1: Manhattan distance.
- p=2: Euclidean distance.

Nearest Neighbour

- The k-nearest neighbors algorithm, also known as KNN or k-NN, is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point.

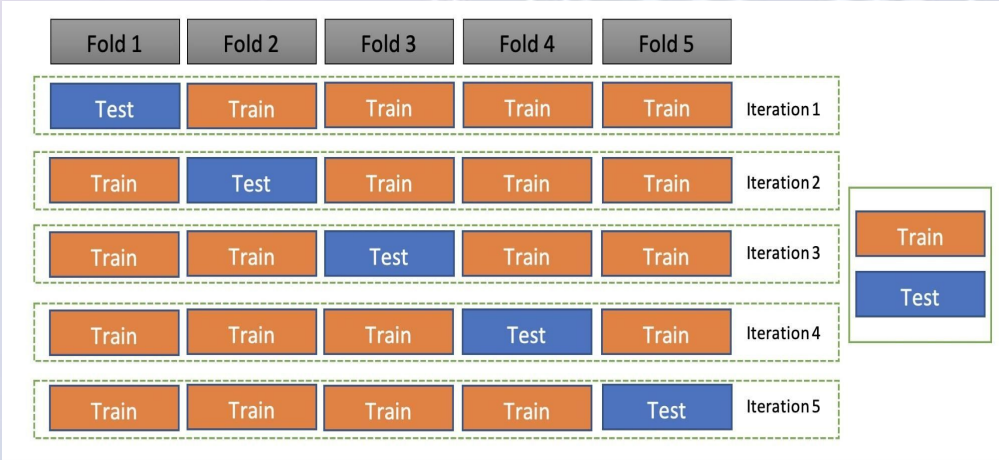
- It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.

Nearest Neighbour

- Choose the value of k for KNN Algorithm
  - **Cross Validation** can be used to choose the value of k
    - K-fold cross-validation.
    - Hold-out cross-validation.
    - Stratified k-fold cross-validation.
    - Leave-p-out cross-validation.
    - Leave-one-out cross-validation.
    - Monte Carlo (shuffle-split)
    - Time series (rolling cross-validation)
- Compute KNN: distance metrics

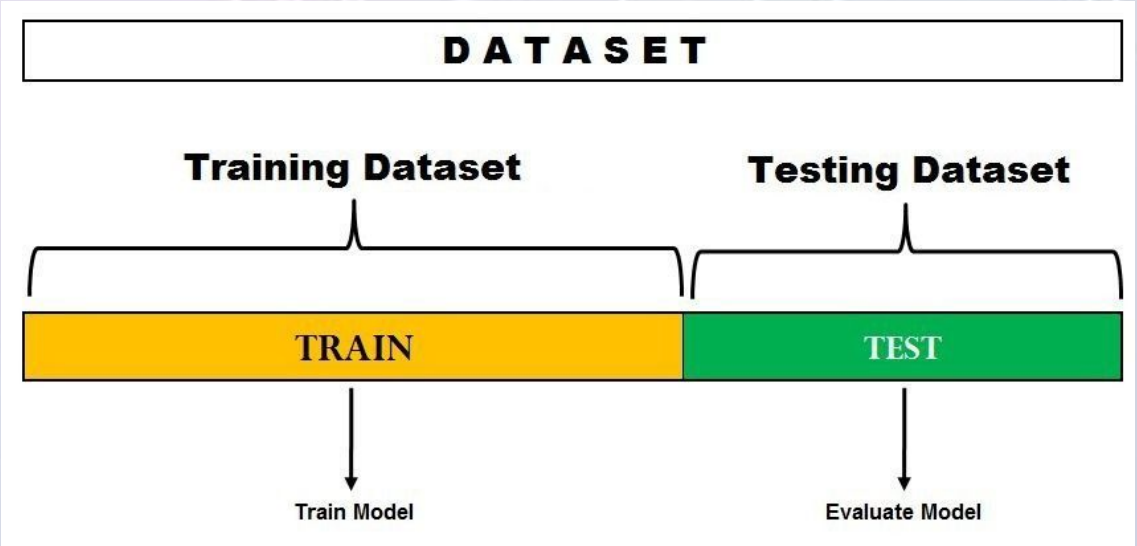
Cross Validation (K-fold)

- The whole dataset is partitioned in k parts of equal size and each partition is called a fold. It's known as k-fold since there are k parts where k can be any integer.
- One fold is used for validation and other K-1 folds are used for training the model. To use every fold as a validation set and other left-outs as a training set, this technique is repeated k times until each fold is used once.



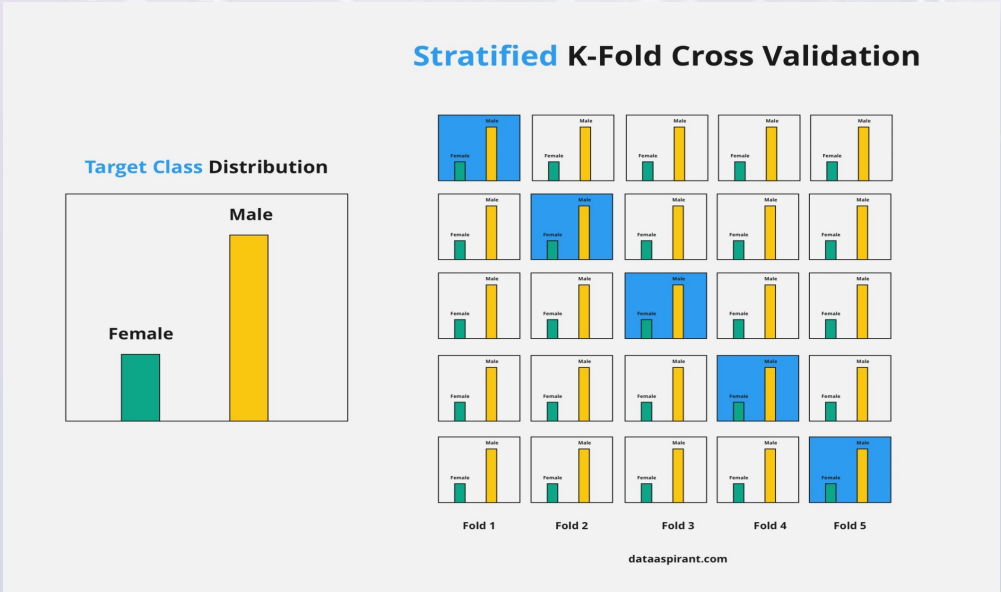
Cross Validation (hold out)

- A train-test split, holdout cross-validation has the entire dataset partitioned randomly into a training set and a validation set. A rule of thumb to partition data is that nearly 70% of the whole dataset will be used as a training set and the remaining 30% will be used as a validation set.



Cross Validation (Stratified k-fold cross-validation.)

- It splits the dataset into k equal folds, each fold has the same ratio of instances of target variables that are in the complete dataset. This enables it to work perfectly for imbalanced datasets, but not for time-series data.



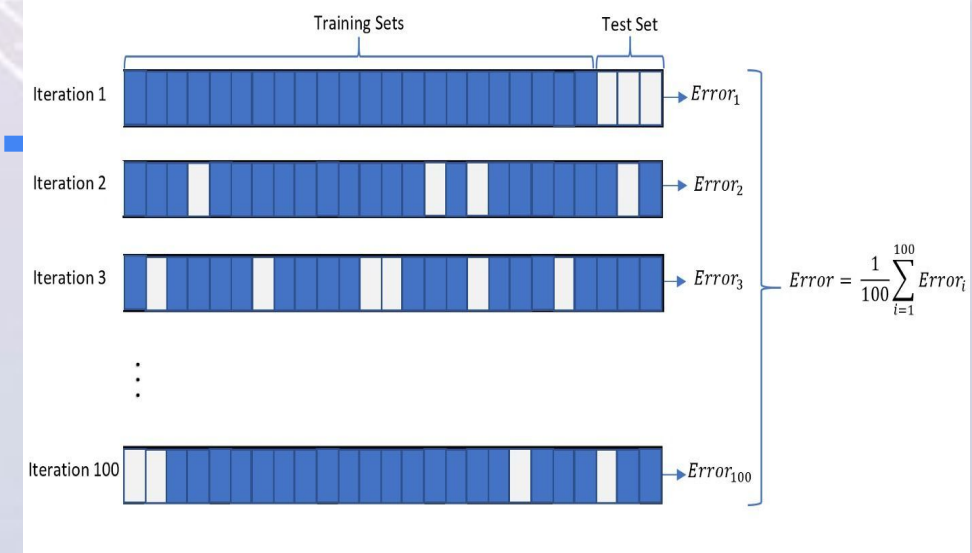
Cross Validation (Leave-one-out cross-validation.)

- In this technique, only 1 sample point is used as a validation set and the remaining n-1 samples are used in the training set. Think of it as a more specific case of the leave-p-out cross-validation technique with P=1.

Cross Validation (Monte Carlo)

- Known as shuffle split cross-validation and repeated random subsampling cross-validation, the Monte Carlo technique involves splitting the whole data into training data and test data. Splitting can be done in the percentage of 70-30% or 60-40% - or anything you prefer. The only condition for each iteration is to keep the train-test split percentage different.

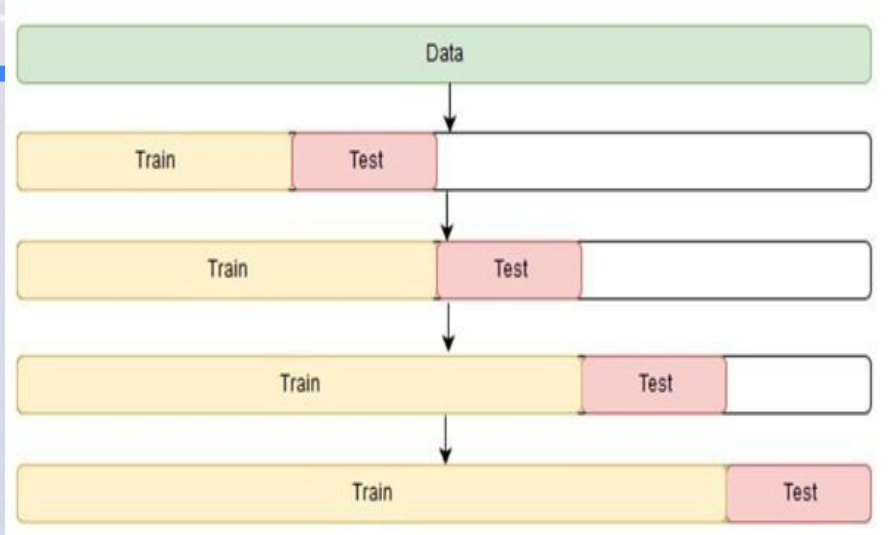
- The next step is to fit the model on the train data set in that iteration and calculate the accuracy of the fitted model on the test dataset. Repeat these iterations many times - 100,400,500 or even higher - and take the average of all the test errors to conclude how well your model performs.



Cross Validation (Time series (rolling cross-validation))

- Since the order of data is very important for time series-related problems, the dataset is split into training and validation sets according to time. Therefore, it's also called the forward chaining method or rolling cross-validation.

- Start the training with a small subset of data. Perform forecasting for the later data points and check their accuracy. The forecasted data points are then included as part of the next training dataset and the next data points are forecasted.The process goes on.



K Nearest Neighbour

- Step-1: Select the number K of the neighbors.
- Step-2: Calculate the Euclidean distance of K number of neighbors.
- Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.
- Step-4: Among these k neighbors, count the number of the data points in each category.
- Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.
- Step-6: Model is ready.

K Nearest Neighbour

Advantages

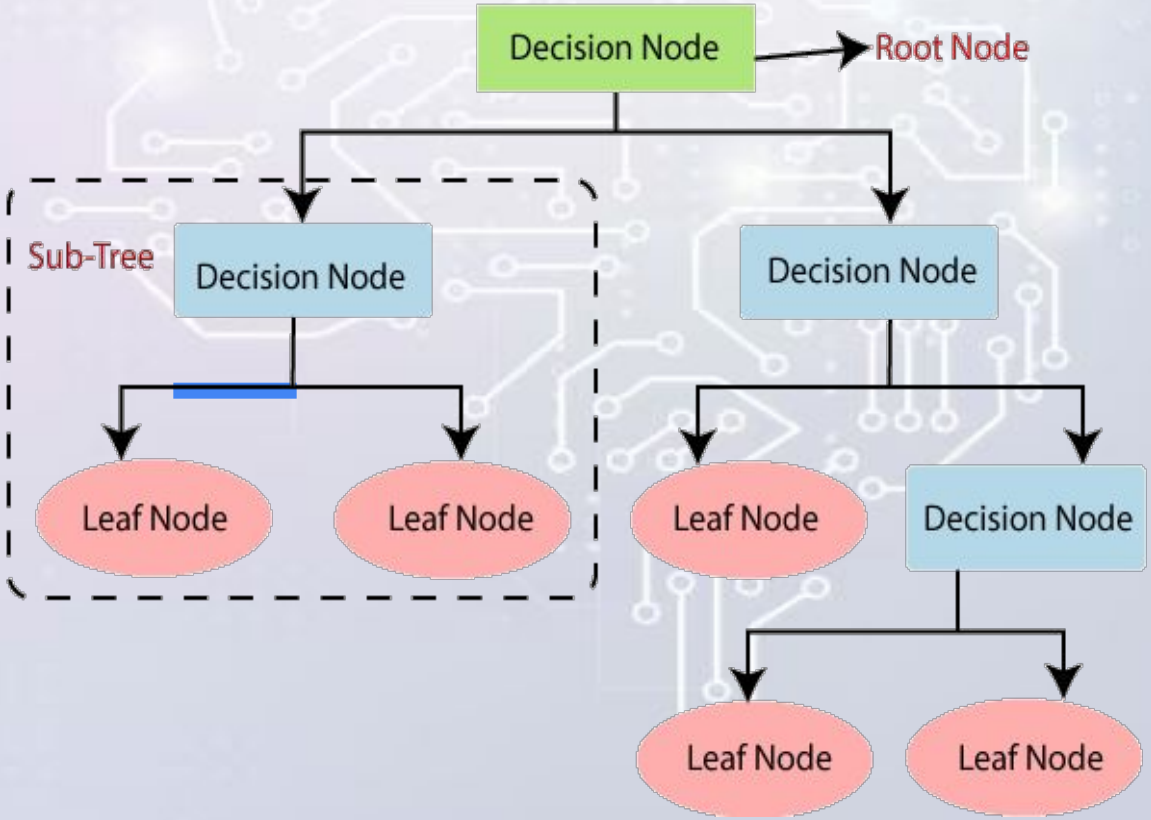
- Simple to implement
- Adapts Easily
- Few Hyperparameters

Disadvantages

- 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.
- Curse of Dimensionality
- Prone to Overfitting

Decision Tree

- Root Node
- Leaf Node
- Splitting
- Branch/Subtree
- Pruning
- Parent/Child node



Decision Tree

- **Entropy** : Measures the level of impurities present in samples.
- It is used to calculate the **Information Gain** and **Gini Index.**

Information Gain = Entropy(parent) – [Weighted Entropy(children)]

Impurity Criterion

Gini Index

$$I_G = 1 - \sum_{j=1}^{c} p_j^2$$

p_j: proportion of the samples that belongs to class c for a particular node

Entropy

$$I_H = -\sum_{j=1}^{c} p_j log_2(p_j)$$

p_j: proportion of the samples that belongs to class c for a particular node.

*This is the the definition of entropy for all non-empty classes ($p \neq 0$). The entropy is 0 if all samples at a node belong to the same class.

Decision Tree

- Entropy of a group in which all examples belong to the same class

$$\textbf{Entropy} = -1 \log_2 1 = 0$$

- It shows Minimum impurity
- It is not a good training set for learning

- Entropy of a group with 50% in either class

$$\textbf{Entropy} = -0.5 \log_2 0.5 - 0.5 \log_2 0.5 = 1$$

- It shows Maximum impurity
- It is a good training set for learning

Decision Tree (Example)

Day	Outlook	Temperature	Humidity	Wind	Play
1	SUNNY	HOT	HIGH	WEAK	NO
2	SUNNY	HOT	HIGH	STRONG	NO
3	OVERCAST	HOT	HIGH	WEAK	YES
4	RAIN	MILD	HIGH	WEAK	YES
5	RAIN	COOL	NORMAL	WEAK	YES
6	RAIN	COOL	NORMAL	STRONG	NO
7	OVERCAST	COOL	NORMAL	STRONG	YES
8	SUNNY	MILD	HIGH	WEAK	NO
9	SUNNY	COOL	NORMAL	WEAK	YES
10	RAIN	MILD	NORMAL	WEAK	YES
11	SUNNY	MILD	NORMAL	STRONG	YES
12	OVERCAST	MILD	HIGH	STRONG	YES
13	OVERCAST	HOT	NORMAL	WEAK	YES
14	RAIN	MILD	HIGH	STRONG	NO

Decision Tree

Total Instances = 14
Total Positive = 9
Total Negative = 5

Tota Entropy = - [((9\14) * $\log_2$(9\14)) + ((5\14) * $\log_2$(5\14))]

= - [(0.642 * (-0.639)) + (0.357 * (-1.486))]
= - [-0.410 + -0.530]
= - [-0.940]
= 0.940

- **Information Gain** of attribute **: Outlook**

Outlook values = Sunny, Rainy, Overcast

S_{Sunny} = [2+, 3-]

S_{Rainy} = [3+ , 2-]

$S_{Overcast}$ = [4+ , 0-]

Decision Tree

Total Entropy (Parent) = 0.940 ..... **equ(1)**

Information Gain of attribute **: Outlook**

Outlook values = Sunny, Rainy, Overcast

S_{Sunny} = [2+, 3-]

S_{Rainy} = [3+ , 2-]

$S_{Overcast}$ = [4+ , 0-]

Entropy S_{Sunny} = 0.970 # 2/5 , 3/5 ..... **equ (2)**

Entropy $S_{Overcast}$ = 0 # All sample belongs to same class ..... **equ (3)**

Entropy S_{Rainy} = 0.970 # 2/5 , ⅗ ..... **equ (4)**

Information Gain of Outlook = Total Entropy - Weighted Entropy of Child nodes
= 0.940 - [(5/14) * 0.970 + 0 + (5/14) * 0.970)]
= 0.246

Decision Tree

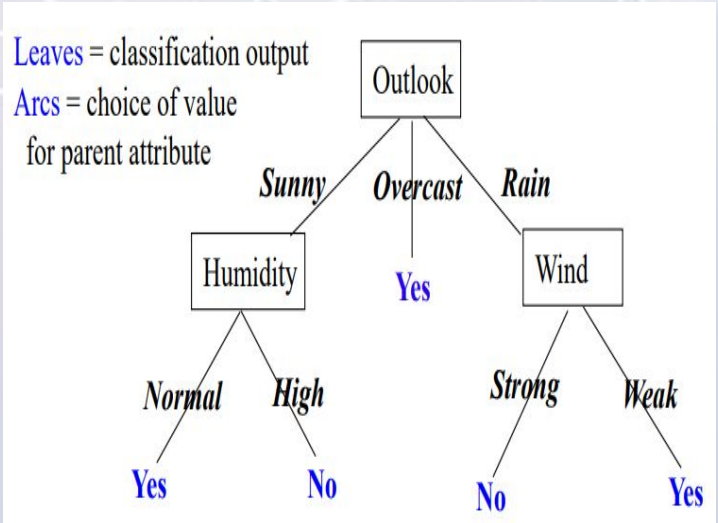
Total Entropy (Parent) = 0.940

Information Gain of Outlook = **0.246**
Information Gain of Temperature = 0.029
Information Gain of Humidity = 0.151
Information Gain of Wind = 0.029

So, Root node will be Outlook,
With subtrees of Sunny, Rainy, Overcast

Sunny is at instance numbers 1,2,8,9,11.

Next,
Information Gain of (Sunny, Humidity)
Information Gain of (Sunny, Temperature)
Information Gain of (Sunny, Wind)



Decision Tree

Sunny is at instance numbers 1,2,8,9,11.

Next,
Information Gain of (Sunny, Humidity)
Information Gain of (Sunny, Temperature)
Information Gain of (Sunny, Wind)

Now calculate,
Information Gain of (Sunny, Humidity) = 0.970 – [3/5 * 0 + 2/5 * 0]
= 0.970

Information Gain of (Sunny, Temperature) = 0.970 – [2/5 * 0 + 2/5 * 1 + 1/5 * 0]
= 0.570

Information Gain of (Sunny, Wind) = 0.970 – [3/5 * 0.918 + 2/5 * 1]
= 0.0192

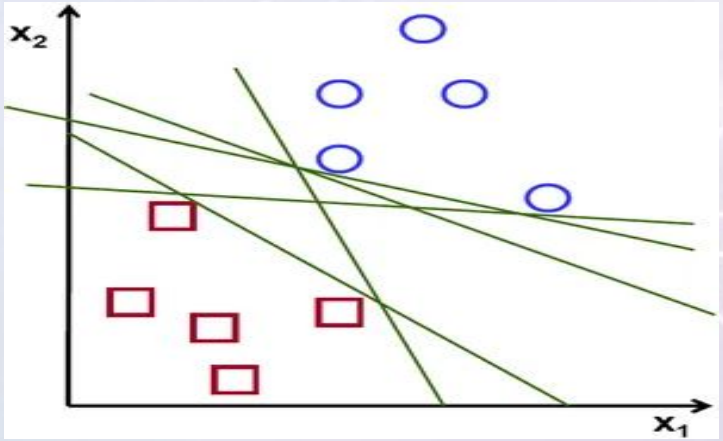
So, attribute Humidity will be next subtree.

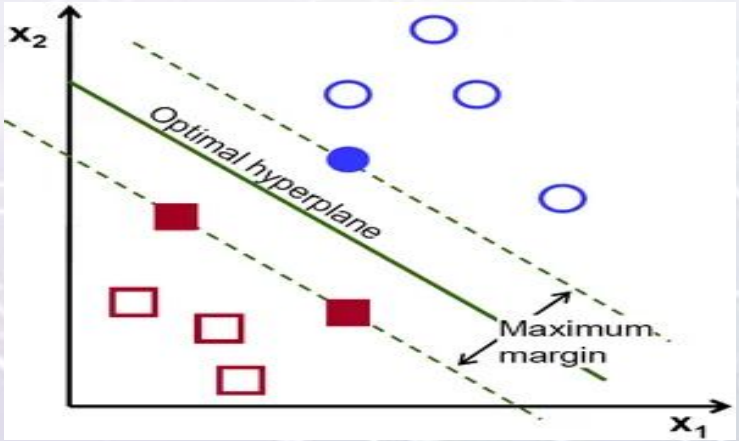
Decision Tree

Patient ID	Age	Sex	BP	Cholesterol	Drug
p1	Young	F	High	Normal	Drug A
p2	Young	F	High	High	Drug A
p3	Middle-age	F	Hiigh	Normal	Drug B
p4	Senior	F	Normal	Normal	Drug B
p5	Senior	M	Low	Normal	Drug B
p6	Senior	M	Low	High	Drug A
p7	Middle-age	M	Low	High	Drug B
p8	Young	F	Normal	Normal	Drug A
p9	Young	M	Low	Normal	Drug B
p10	Senior	M	Normal	Normal	Drug B
p11	Young	M	Normal	High	Drug B
p12	Middle-age	F	Normal	High	Drug B
p13	Middle-age	M	High	Normal	Drug B
p14	Senior	F	Normal	High	Drug A
p15	Middle-age	F	Low	Normal	?

Naive Bayes

Support Vector Machine (SVM)

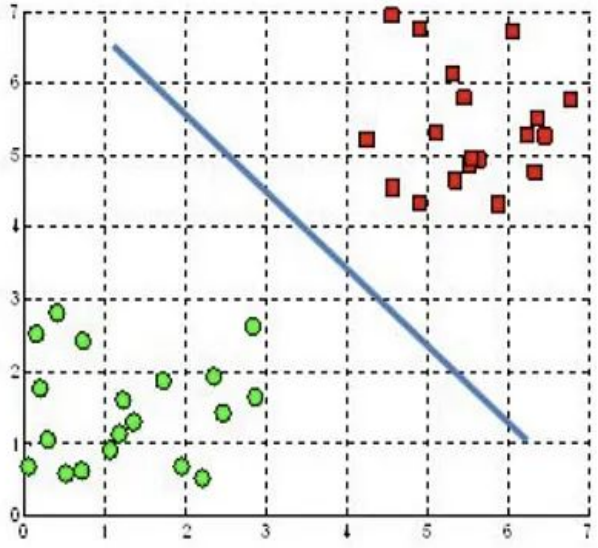




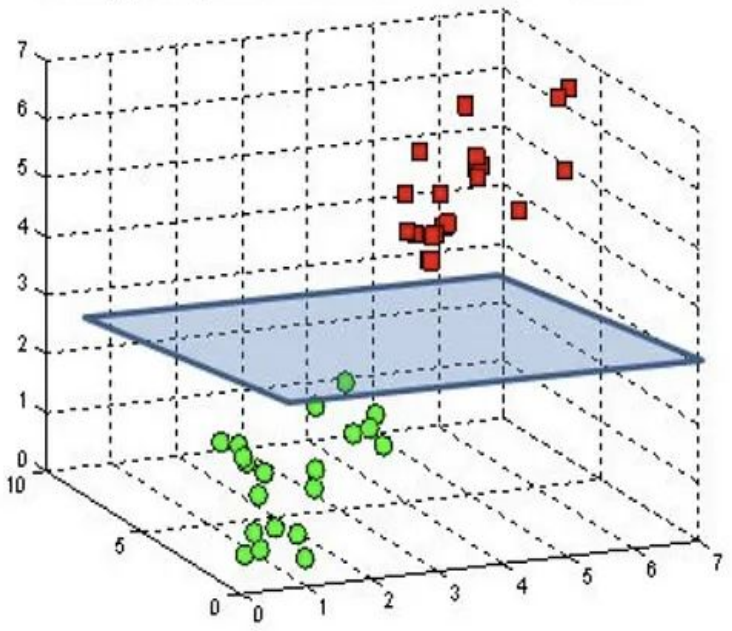
- Support Vector Machine (SVM) is a supervised machine learning algorithm used for linear or nonlinear classification, regression, and outlier detection tasks.
- SVMs can manage high-dimensional data and nonlinear relationships.
- The main objective of the SVM algorithm is to find the optimal hyperplane in an N-dimensional space that can separate the data points in different classes in the feature space.
- The hyperplane tries that the margin between the closest points of different classes should be as maximum as possible.
- The dimension of the hyperplane depends upon the number of features. If the number of input features is two, then the hyperplane is just a line. If the number of input features is three, then the hyperplane becomes a 2-D plane.

Support Vector Machine (SVM)

- A hyperplane in $\mathbb{R}^2$ is a line



- A hyperplane in $\mathbb{R}^3$ is a plane



Support Vector Machine (SVM)

- **Hyperplane:** Hyperplane is the decision boundary that is used to separate the data points of different classes in a feature space. In the case of linear classifications, it will be a linear equation i.e. wx+b = 0.
- **Support Vectors:** Support vectors are the closest data points to the hyperplane, which makes a critical role in deciding the hyperplane and margin.
- **Margin:** Margin is the distance between the support vector and hyperplane. The main objective of the support vector machine algorithm is to maximize the margin.
- **Kernel:** Kernel is the mathematical function, which is used in SVM to map the original input data points into high-dimensional feature spaces, so, that the hyperplane can be easily found out even if the data points are not linearly separable in the original input space. Some of the common kernel functions are linear, polynomial, radial basis function(RBF), and sigmoid.
- **Hard Margin:** The maximum-margin hyperplane or the hard margin hyperplane is a hyperplane that properly separates the data points of different categories without any misclassifications.

Support Vector Machine (SVM)

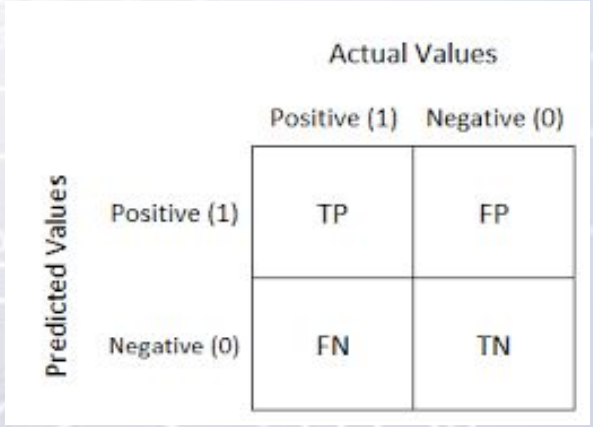
- **Soft Margin:** When the data is not perfectly separable or contains outliers, SVM permits a soft margin technique. Each data point has a slack variable introduced by the soft-margin SVM formulation, which softens the strict margin requirement and permits certain misclassifications or violations. It discovers a compromise between increasing the margin and reducing violations.

- **C:** Margin maximisation and misclassification fines are balanced by the regularisation parameter C in SVM. The penalty for going over the margin or misclassifying data items is decided by it. A stricter penalty is imposed with a greater value of C, which results in a smaller margin and perhaps fewer misclassifications.

- **Hinge Loss:** A typical loss function in SVMs is hinge loss. It punishes incorrect classifications or margin violations. The objective function in SVM is frequently formed by combining it with the regularisation term.

Performance measures for Supervised classification

- **Confusion Matrix:** A confusion matrix represents the prediction summary in matrix form. It shows how many prediction are correct and incorrect per class.
- **Precision** shows how often an ML model is correct when predicting the target class.
- **Recall** shows whether an ML model can find all objects of the target class.
- Precision can be seen as a measure of quality, and recall as a measure of quantity. Higher precision means irrelevant ones, and high recall means that an algorithm returns most of the relevant results (whether or not irrelevant ones are also returned).
- The **F-score**, also called the F1-score, is a measure of a model's accuracy on a dataset.



$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

Unsupervised Learning : Clustering

- **K-means/Kernel K-means**
  - Example:
  - following 7 data points are given:
  - A1 (2,10), A2(2,5), A3(8,4), B1(5,8), B2(7,5), B3(6,4), C1(1,2), C2(4,9)
  - Assume current centroid as A1(2,10) , B1(5,8), and C1(1,2)
  - Calculate euclidean distance from each centroids.
    - Distance (A1, A1) = 0
    - Distance (A2, A1) = sqrt ((2-2)^2 + (5-10)^2) = 5
    - Distance (A3, A1) =....
  - Recalculate centroid/mean by taking mean of each cluster.
  - Repeat the process until convergence.

Unsupervised Learning : K-means Clustering

- K-means/Kernel K-means: Example

Data Points			Distance to						Initial 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	

Unsupervised Learning : K mean Clustering

- New Centroid by taking,
  Mean of Cluster 1 = (2,10)
  Mean of Cluster 2 = [(8+5+7+6+4)/5] and [(4+8+5+4+9)/5] = (6, 6)
  Mean of Cluster 3 = [(2+1)/2] and [(5+2)/2] = (1.5, 3.5)

Data Points			Distance to						Initial Cluster	New Cluster 1
			2	10	6	6	1.5	3.5		
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

Unsupervised Learning : K mean Clustering

- New Centroid by taking,
  Mean of Cluster 1 = (3, 9.5)
  Mean of Cluster 2 = (6.5, 5.25)
  Mean of Cluster 3 = (1.5, 3.5)

Data Points			Distance to						Initial Cluster	New Cluster 1	New Cluster 2
			3	9.5	6.5	5.25	1.5	3.5			
A1	2	10					6.52		1	1	1
A2	2	5					1.58		3	3	3
A3	8	4					6.52		2	2	2
B1	5	8					5.70		2	2	2
B2	7	5					5.70		2	2	2
B3	6	4					4.53		2	2	2
C1	1	2					1.58		3	3	3
C2	4	9					6.04		2	1	1

Unsupervised Learning : K mean Clustering

Initialize k means with random values / wcss(within cluster sum of square)

--> For a given number of iterations/convergence:

--> Iterate through items:

--> Find the mean closest to the item by calculating

the euclidean distance of the item with each of the means

--> Assign item to mean

--> Update mean by shifting it to the average of the items in that cluster

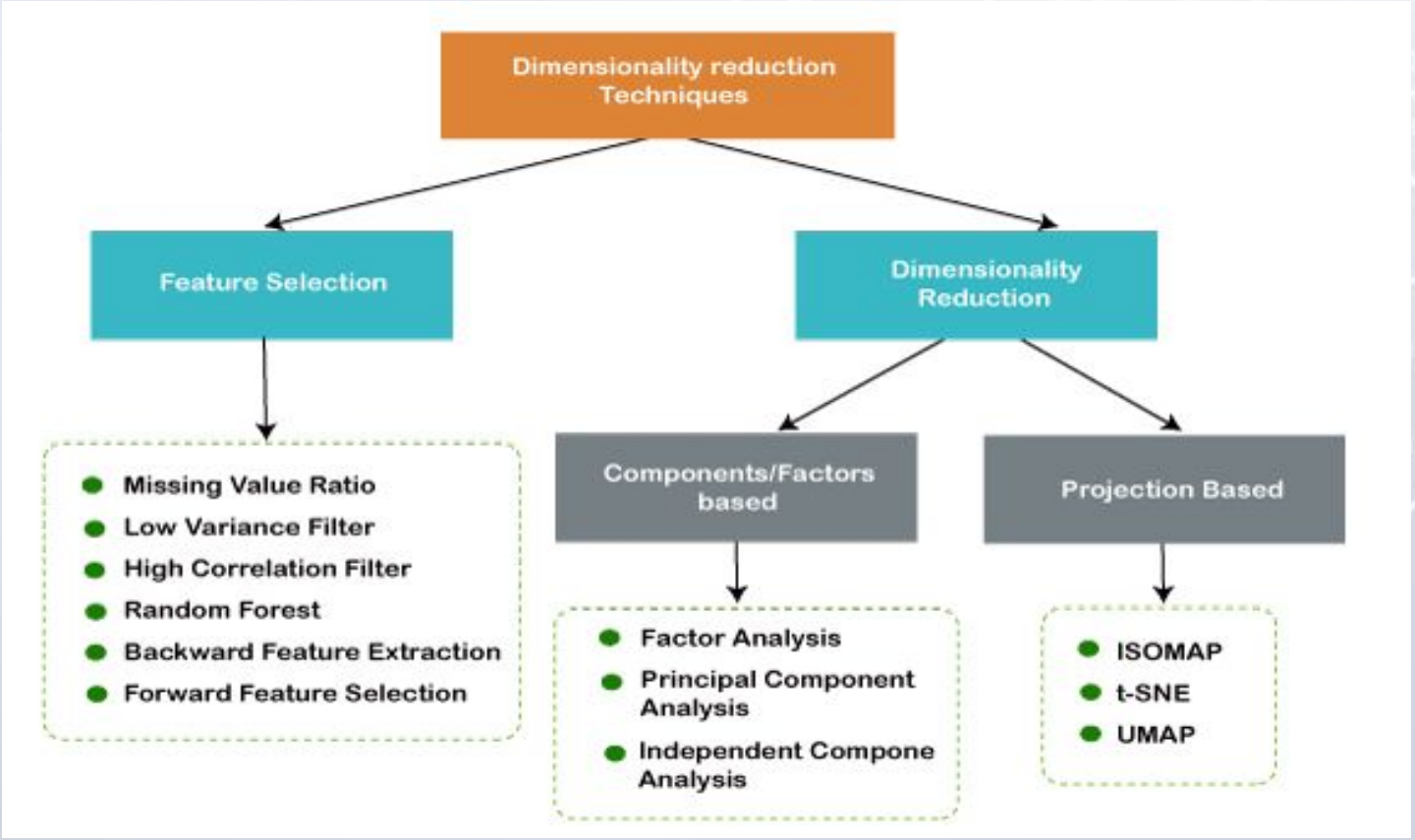
Unsupervised Learning : Clustering

Dimensionality Reduction:-

"It is a way of converting the higher dimensions dataset into lesser dimensions dataset ensuring that it provides similar information."

It is commonly used in the fields that deal with high-dimensional data, such as speech recognition, signal processing, bioinformatics, etc. It can also be used for data visualization, noise reduction, cluster analysis, etc.

Unsupervised Learning : Clustering



Unsupervised Learning : Clustering

Dimensionality Reduction -PCA, CCA, LDA, ICA

PCA : Principal Component Analysis
CCA : Canonical Correlation Analysis
LDA : Linear Discriminant Analysis
ICA : Independent Component Analysis

Unsupervised Learning : PCA

- The main goal of **Principal Component Analysis (PCA)** is to **reduce the dimensionality** of a dataset while preserving the most important patterns or relationships between the variables without any prior knowledge of the target variables.

Step 1: Standardization

To standardize dataset to ensure that each variable has a mean of 0 and a standard deviation of 1.

$$Z = (X- \mu) / \sigma$$

Here,

- μ is the mean of independent features $\{\mu1, \mu2, ..., \mu_m\}$
- σ is the standard deviation of independent features $\{\sigma1, \sigma2, ..., \sigma_m\}$

Unsupervised Learning : PCA

Step 2: **Covariance Matrix Computation**

Covariance measures the strength of joint variability between two or more variables, indicating how much they change in relation to each other.

To find the covariance we can use the formula:

$$cov_{x,y} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{N - 1}$$

N = number of data values

The value of covariance can be positive, negative, or zero.

- Positive: As the x increases y also increases.
- Negative: As the x increases y also decreases.
- Zero: No direct relation

Unsupervised Learning : PCA

Step 3: Compute Eigenvalues and Eigenvectors of Covariance Matrix to Identify Principal Components

Let A be a square nXn matrix and X be a non-zero vector for which

$$AX = \lambda X$$

for some scalar values λ.

then λ is known as the eigenvalue of matrix A and X is known as the eigenvector of matrix A for the corresponding eigenvalue.

It can also be written as :

$$AX - \lambda X = 0$$

$$AX = \lambda X$$

Unsupervised Learning : PCA & LDA

- LDA maximizes the distance between different classes, whereas PCA maximizes the variance of the data.
- When there are fewer samples in each class, PCA performs better. LDA, however, performs better on large datasets with many classes.

Unsupervised Learning : PCA & LDA

❑

Unsupervised Learning : Clustering

- MNF – Minimum Noise Fraction.
  It is a Dimensionality Reduction Technique.
  It reduces dimensions in terms of SNR (Signal to Noise Ratio)

- Canonical Variates
- Feature Selection vs Feature Extraction

Unsupervised Learning : Clustering

- Generative Models (mixture models and latent factor models)
- **Latent variables** are variables that are unobserved, but whose influence can be summarized through one or more indicator variables. They are useful for capturing complex or conceptual properties of a system that are difficult to quantify or measure directly.
- Ex: IQ, Quality of life,dedication,etc.