UNIT-III

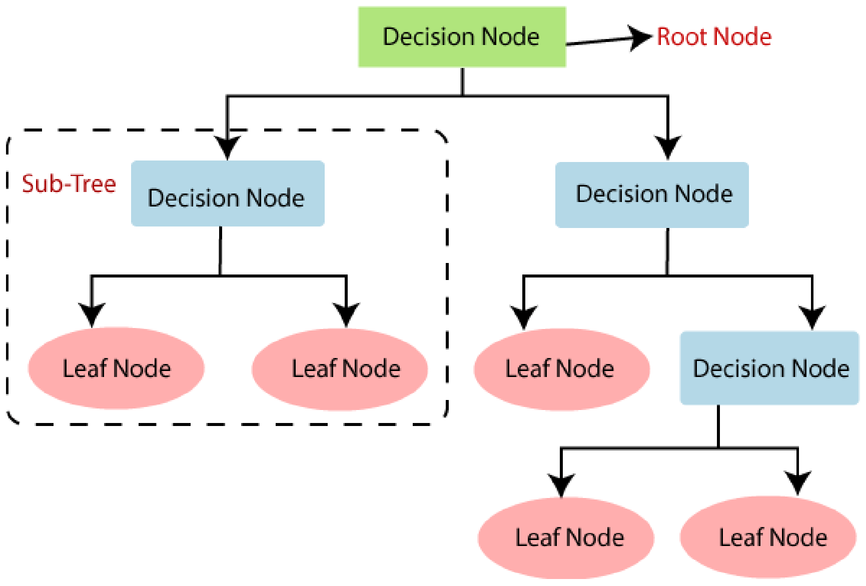
Learning with Trees: Decision Trees, Constructing Decision Trees, Classification and Regression Trees.

Ensemble Learning: Boosting, Bagging, Different ways to combine classifiers, Basic Statistics, Gaussian Mixture Models, Nearest Neighbour Methods.

Unsupervised Learning: K Means Algorithm.

Decision Trees:

- 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 tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules **and** each leaf node represents the outcome.
- In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node.
- Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
- The decisions or the tests are performed on the basis of features of the given dataset.
- It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.
- 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.



Example:

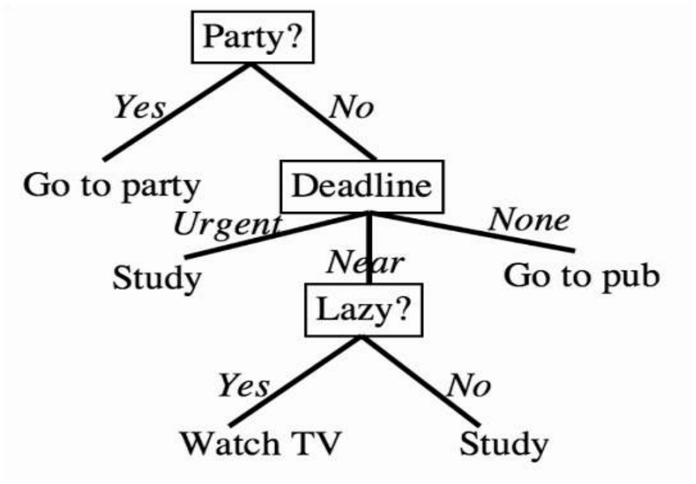


FIGURE 12.1 A simple decision tree to decide how you will spend the evening.

- One of the reasons that decision trees are popular is that we can turn them into a set of logical disjunctions (if ... then rules) that then go into program code very simply.
  Ex: if there is a party then go to it
  if there is not a party and you have an urgent deadline then study

Constructing Decision Trees:

Types of Decision Tree Algorithms:

- ID3: This algorithm measures how mixed up the data is at a node using something called entropy. It then chooses the feature that helps to clarify the data the most.
- C4.5: This is an improved version of ID3 that can handle missing data and continuous attributes.
- CART: This algorithm uses a different measure called Gini impurity to decide how to split the data. It can be used for both classification (sorting data into categories) and regression (predicting continuous values) tasks.

ID3 Algorithm:

Entropy in Information Theory:

- Entropy measures the amount of impurity in a set of features.
- The entropy H of a set of probabilities p_i is:

$$\mathrm{Entropy}(p) = -\sum_i p_i \log_2 p_i,$$

- where the logarithm is base 2 because we are imagining that we encode everything using binary digits (bits), and we define 0 log 0 = 0.

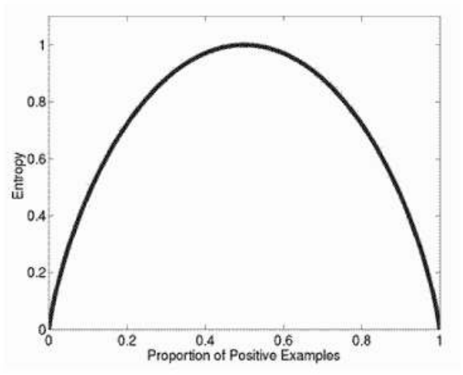


FIGURE 12.2 A graph of entropy, detailing how much information is available from finding out another piece of information given what you already know.

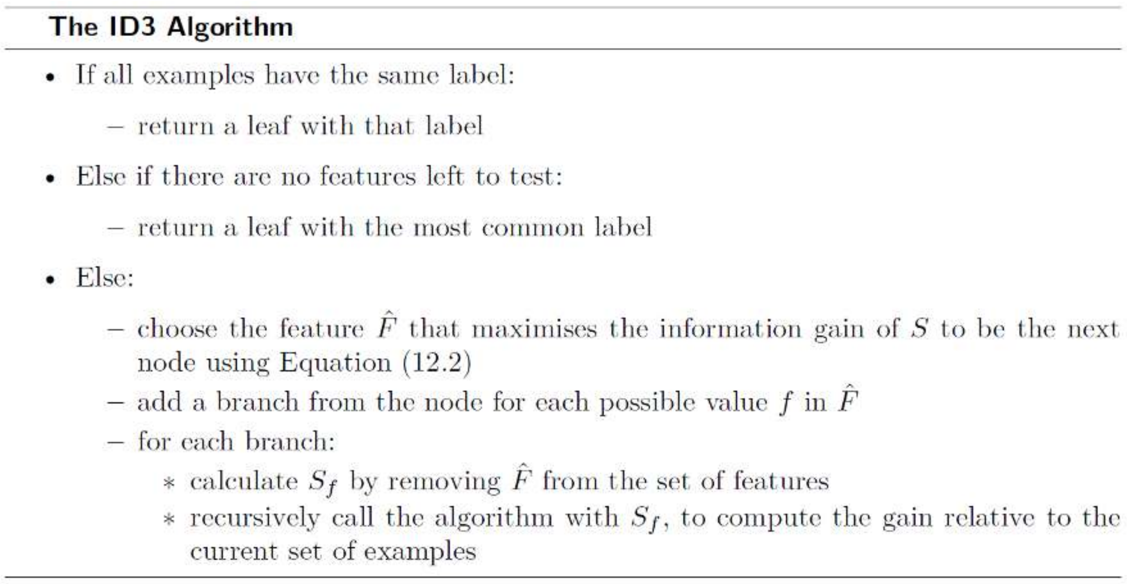
- If all of the examples are positive, then we don't get any extra information from knowing the value of the feature for any particular example, since whatever the value of the feature, the example will be positive. Thus, the entropy of that feature is 0.
- However, if the feature separates the examples into 50% positive and 50% negative, then the amount of entropy is at a maximum, and knowing about that feature is very useful to us.
- For our decision tree, the best feature to pick as the one to classify on now is the one that gives you the most information, i.e., the one with the highest entropy.

Information Gain:

- It is defined as the entropy of the whole set minus the entropy when a particular feature is chosen.

$$\text{Gain}(S,F) = \text{Entropy}(S) - \sum_{f \in values(F)} \frac{|S_f|}{|S|} \text{Entropy}(S_f). \qquad (12.2)$$

- The ID3 algorithm computes this information gain for each feature and chooses the one that produces the highest value.



The ID3 Algorithm

- If all examples have the same label:
    - return a leaf with that label
- Else if there are no features left to test:
    - return a leaf with the most common label
- Else:
    - choose the feature $\hat{F}$ that maximises the information gain of S to be the next node using Equation (12.2)
    - add a branch from the node for each possible value f in $\hat{F}$
    - for each branch:
        * calculate S_f by removing $\hat{F}$ from the set of features
        * recursively call the algorithm with S_f, to compute the gain relative to the current set of examples

C4.5 Algorithm:

- It is an improved version of ID3.
- Pruning is another method that can help us avoid overfitting.
- It helps in improving the performance of the Decision tree by cutting the nodes or sub-nodes which are not significant.
- Additionally, it removes the branches which have very low importance.
- There are mainly 2 ways for pruning:
- **Pre-pruning** – we can stop growing the tree earlier, which means we can prune/remove/cut a node if it has low importance while growing the tree.
- **Post-pruning** – once our tree is built to its depth, we can start pruning the nodes based on their significance.
- C4.5 uses a different method called rule post-pruning.
- This consists of taking the tree generated by ID3, converting it to a set of if-then rules, and then pruning each rule by removing preconditions if the accuracy of the rule increases without it.
- The rules are then sorted according to their accuracy on the training set and applied in order.
- The advantages of dealing with rules are that they are easier to read and their order in the tree does not matter, just their accuracy in the classification.
- For Continuous Variables, the simplest solution is to discretise the continuous variable.
- Computation complexity of Decision Tree is O(dnlogn) where n is number of data points, d is number of dimensions.

Classification Example: construct the decision tree to decide what to do in the evening

Deadline?	Is there a party?	Lazy?	Activity
Urgent	Yes	Yes	Party
Urgent	No	Yes	Study
Near	Yes	Yes	Party
None	Yes	No	Party
None	No	Yes	Pub
None	Yes	No	Party
Near	No	No	Study
Near	No	Yes	TV
Near	Yes	Yes	Party
Urgent	No	No	Study

We start with which feature has to selected as a root node?

Compute Entropy of S:

$$\begin{aligned}
\text{Entropy}(S) &= -p_{\text{party}} \log_2 p_{\text{party}} - p_{\text{study}} \log_2 p_{\text{study}} \\
&- p_{\text{pub}} \log_2 p_{\text{pub}} - p_{\text{TV}} \log_2 p_{\text{TV}} \\
&= -\frac{5}{10} \log_2 \frac{5}{10} - \frac{3}{10} \log_2 \frac{3}{10} - \frac{1}{10} \log_2 \frac{1}{10} - \frac{1}{10} \log_2 \frac{1}{10} \\
&= 0.5 + 0.5211 + 0.3322 + 0.3322 = 1.6855 \qquad (12.11)
\end{aligned}$$

find which feature has the maximal information gain:

$$\begin{aligned}
\text{Gain}(S, \text{Deadline}) &= 1.6855 - \frac{|S_{\text{urgent}}|}{10} \text{Entropy}(S_{\text{urgent}}) \\
&- \frac{|S_{\text{near}}|}{10} \text{Entropy}(S_{\text{near}}) - \frac{|S_{\text{none}}|}{10} \text{Entropy}(S_{\text{none}}) \\
&= 1.6855 - \frac{3}{10} \left(-\frac{2}{3} \log_2 \frac{2}{3} - \frac{1}{3} \log_2 \frac{1}{3} \right) \\
&- \frac{4}{10} \left(-\frac{2}{4} \log_2 \frac{2}{4} - \frac{1}{4} \log_2 \frac{1}{4} - \frac{1}{4} \log_2 \frac{1}{4} \right) \\
&- \frac{3}{10} \left(-\frac{1}{3} \log_2 \frac{1}{3} - \frac{2}{3} \log_2 \frac{2}{3} \right) \\
&= 1.6855 - 0.2755 - 0.6 - 0.2755 \\
&= 0.5345 \qquad (12.12)
\end{aligned}$$

$$\begin{aligned}\text{Gain}(S,\text{Party}) &= 1.6855 - \frac{5}{10}\left(-\frac{5}{5}\log_2\frac{5}{5}\right) \\ &- \frac{5}{10}\left(-\frac{3}{5}\log_2\frac{3}{5} - \frac{1}{5}\log_2\frac{1}{5} - \frac{1}{5}\log_2\frac{1}{5}\right) \\ &= 1.6855 - 0 - 0.6855 \\ &= 1.0 \qquad (12.13)\end{aligned}$$

$$\begin{aligned}\text{Gain}(S,\text{Lazy}) &= 1.6855 - \frac{6}{10}\left(-\frac{3}{6}\log_2\frac{3}{6} - \frac{1}{6}\log_2\frac{1}{6} - \frac{1}{6}\log_2\frac{1}{6} - \frac{1}{6}\log_2\frac{1}{6}\right) \\ &- \frac{4}{10}\left(-\frac{2}{4}\log_2\frac{2}{4} - \frac{2}{4}\log_2\frac{2}{4}\right) \\ &= 1.6855 - 1.0755 - 0.4 \\ &= 0.21 \qquad (12.14)\end{aligned}$$

- Therefore, the root node will be the party feature, which has two feature values ('yes' and 'no'), so it will have two branches coming out of it.
-

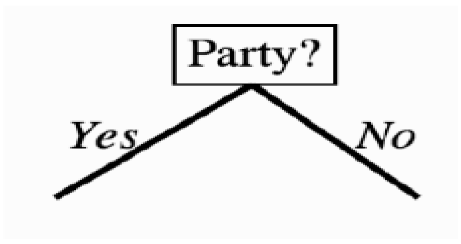


FIGURE 12.6 The decision tree after one step of the algorithm.

When we look at the 'yes' branch, we see that in all five cases where there was a party we went to it, so we just put a leaf node there, saying 'party'.

- For the 'no' branch, out of the five cases there are three different outcomes, so now we need to choose another feature.
- The five cases we are looking at are:

Deadline?	Is there a party?	Lazy?	Activity
Urgent	No	Yes	Study
None	No	Yes	Pub
Near	No	No	Study
Near	No	Yes	TV
Urgent	No	Yes	Study

- We've used the party feature, so we just need to calculate the information gain of the other two over these five examples:

$$\begin{aligned}
\text{Gain}(S,\text{Deadline}) &= 1.371 - \frac{2}{5}\left(-\frac{2}{2}\log_2\frac{2}{2}\right) \\
&- \frac{2}{5}\left(-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2}\right) - \frac{1}{5}\left(-\frac{1}{1}\log_2\frac{1}{1}\right) \\
&= 1.371 - 0 - 0.4 - 0 \\
&= 0.971 \qquad (12.15) \\
\text{Gain}(S,\text{Lazy}) &= 1.371 - \frac{4}{5}\left(-\frac{2}{4}\log_2\frac{2}{4} - \frac{1}{4}\log_2\frac{1}{4} - \frac{1}{4}\log_2\frac{1}{4}\right) \\
&- \frac{1}{5}\left(-\frac{1}{1}\log_2\frac{1}{1}\right) \\
&= 1.371 - 1.2 - 0 \\
&= 0.1710 \qquad (12.16)
\end{aligned}$$

- Here, Deadline feature has maximum information gain. Hence, we selected Deadline feature for splitting data.

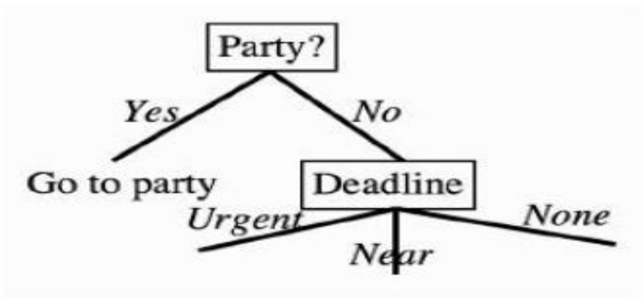
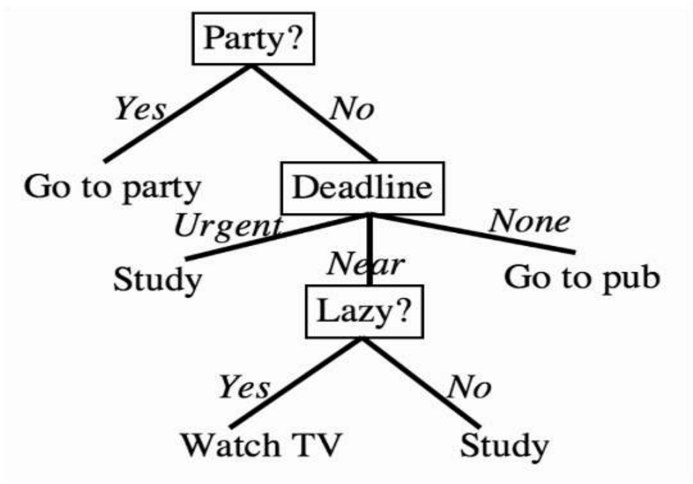


FIGURE 12.7 The tree after another step.

- Finally, we will get the following decision tree.



Classification and Regression Trees(CART):

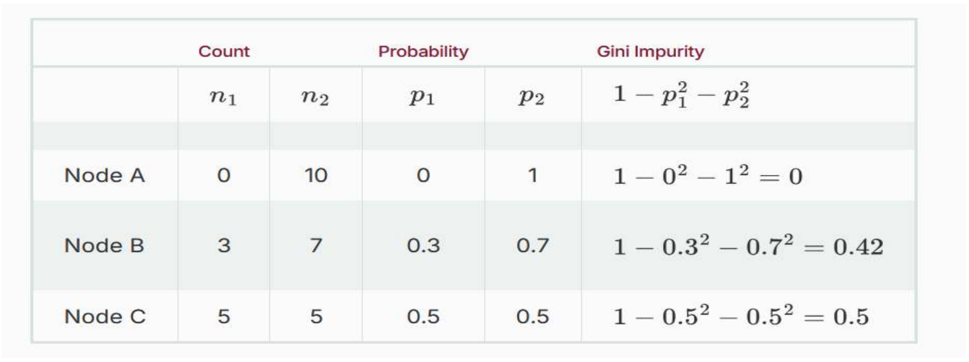
- It is another well-known tree-based algorithm, CART, whose name indicates that it can be used for both classification and regression.

Gini Impurity:

- It is the probability of misclassifying a randomly chosen element in a set.
- The 'impurity' in the name suggests that the aim of the decision tree is to have each leaf node represent a set of data points that are in the same class, so that there are no mismatches. This is known as purity.
- If a leaf is pure then all of the training data within it have just one class.
- Consider a dataset D that contains samples from k classes.
- The probability of samples belonging to class i at a given node can be denoted as p_i. Then the Gini Impurity of is defined as:

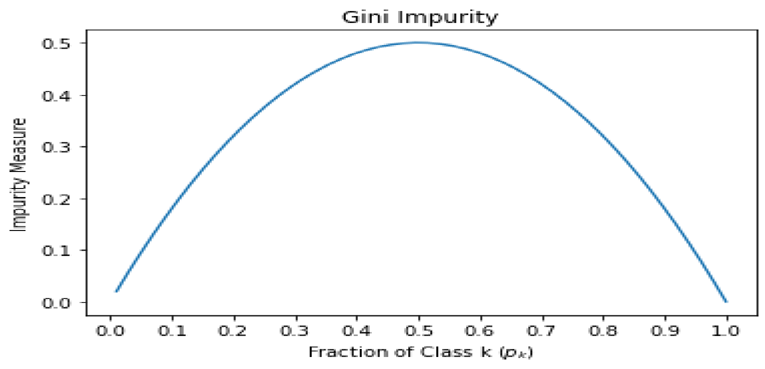
$$Gini(D) = 1 - \sum_{i=1}^{k} p_i^2$$

- The node with uniform class distribution has the highest impurity.
- The minimum impurity is obtained when all records belong to the same class.



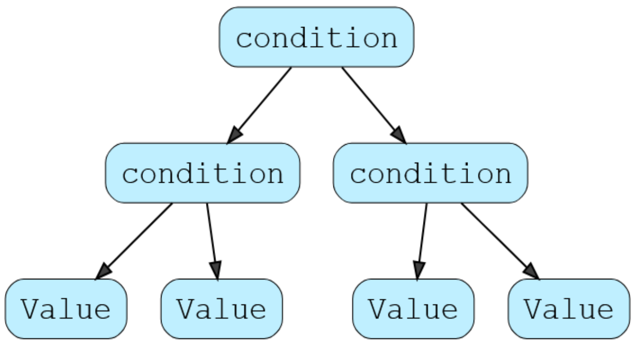
	Count		Probability		Gini Impurity
	n_1	n_2	p_1	p_2	$1 - p_1^2 - p_2^2$
Node A	0	10	0	1	$1 - 0^2 - 1^2 = 0$
Node B	3	7	0.3	0.7	$1 - 0.3^2 - 0.7^2 = 0.42$
Node C	5	5	0.5	0.5	$1 - 0.5^2 - 0.5^2 = 0.5$

- An attribute with the smallest Gini Impurity is selected for splitting the node.



Regression in Trees:

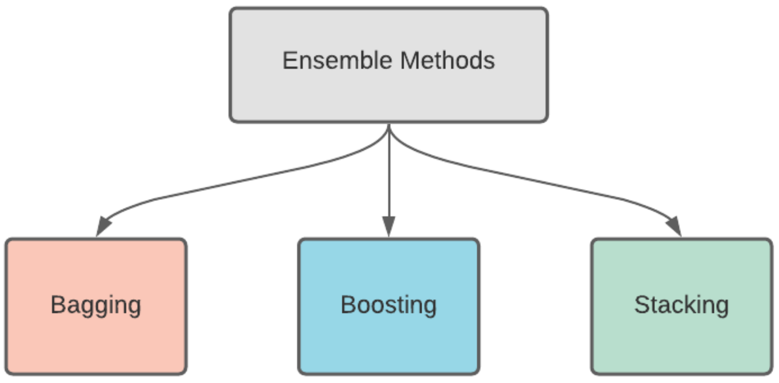
- A Regression tree is an algorithm where the target variable is continuous and the tree is used to predict its value**.**



- Regression Tree works by splitting the training data recursively into smaller subsets based on specific criteria.
- The objective is to split the data in a way that minimizes the residual reduction (Sum of Squared Error) in each subset.
- **Residual Reduction-** Residual reduction is a measure of how much the average squared difference between the predicted values and the actual values for the target variable is reduced by splitting the subset. The lower the residual reduction, the better the model fits the data.
- **Splitting Criteria-** CART evaluates every possible split at each node and selects the one that results in the greatest reduction of residual error in the resulting subsets. This process is repeated until a stopping criterion is met, such as reaching the maximum tree depth or having too few instances in a leaf node.

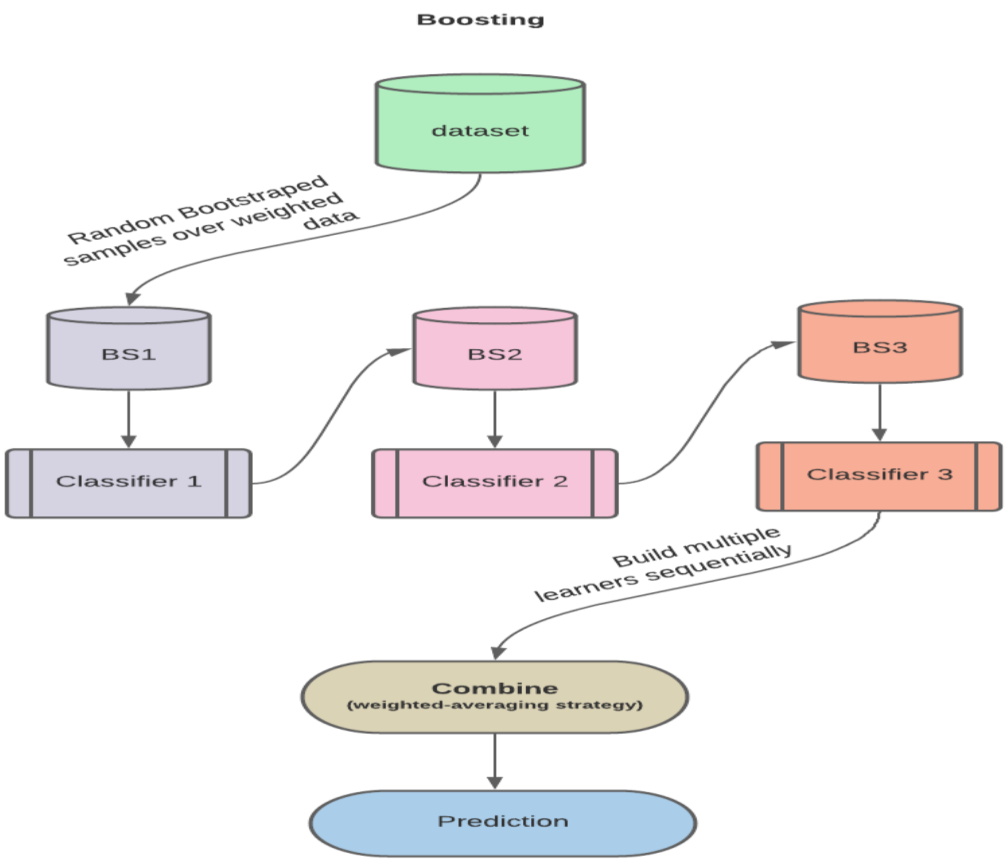
Ensemble Learning:

- Ensemble learning refers to the approach of combining multiple ML models to produce a more accurate and robust prediction compared to any individual model.
- The conventional ensemble methods include **bagging**, **boosting**, and **stacking-based** methods



Boosting:

- Boosting is an ensemble technique that combines multiple weak learners to create a strong learner.


Boosting
dataset
Random Bootstraped samples over weighted data
BS1
BS2
BS3
Classifier 1
Classifier 2
Classifier 3
Build multiple learners sequentially
Combine (weighted-averaging strategy)
Prediction

- The ensemble of weak models are trained in series such that each model that comes next, tries to correct errors of the previous model until the entire training dataset is predicted correctly.
- One of the most well-known boosting algorithms is AdaBoost (Adaptive Boosting).

AdaBoost:

- AdaBoost short for Adaptive Boosting is an ensemble learning used in machine learning for classification and regression problems.
- The main idea behind AdaBoost is to iteratively train the weak classifier on the training dataset with each successive classifier giving more weightage to the data points that are misclassified.
- The final AdaBoost model is decided by combining all the weak classifier that has been used for training with the weightage given to the models according to their accuracies.
- The model which has the highest accuracy is given the highest weightage while the model which has the lowest accuracy is given a lower weightage.

Steps in AdaBoost:

1. Weight Initialization

At the start, every instance is assigned an identical weight. These weights determine the importance of every example.

2. Model Training

A weak learner is skilled at the dataset, with the aim of minimizing classification errors.

3. Weighted Error Calculation

The weighted mistakes are then calculated by means of summing up the weights of the misclassified times. This step emphasizes the importance of the samples which are tough to classify.

4. Model Weight Calculation

The weight of the susceptible learner is calculated primarily based on their Performance in classifying the training data. Models that perform properly are assigned higher weights, indicating that they're more reliable.

5. Update Instance Weights

The example weights are updated to offer more weight to the misclassified samples from the previous step.

6. Repeat

Steps 2 through 5 are repeated for a predefined variety of iterations or till a distinctive overall performance threshold is met.

7. Final Model Creation

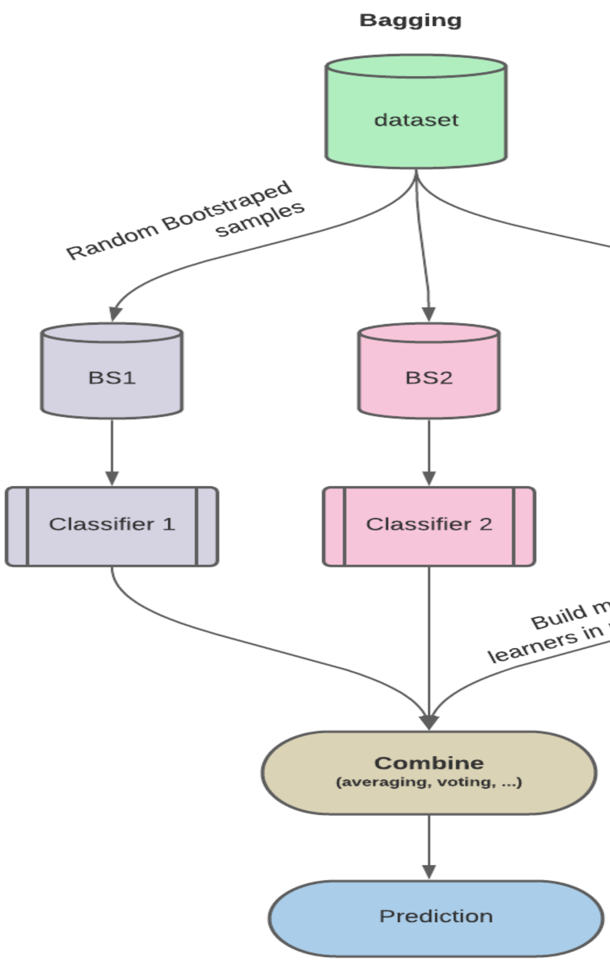
The very last sturdy model (also referred to as the ensemble) is created by means of combining the weighted outputs of all weak learners.

8. Classification

To make predictions on new records, AdaBoost uses the very last ensemble model.

Bagging:

- Bagging is a supervised learning technique that can be used for both regression and classification tasks.

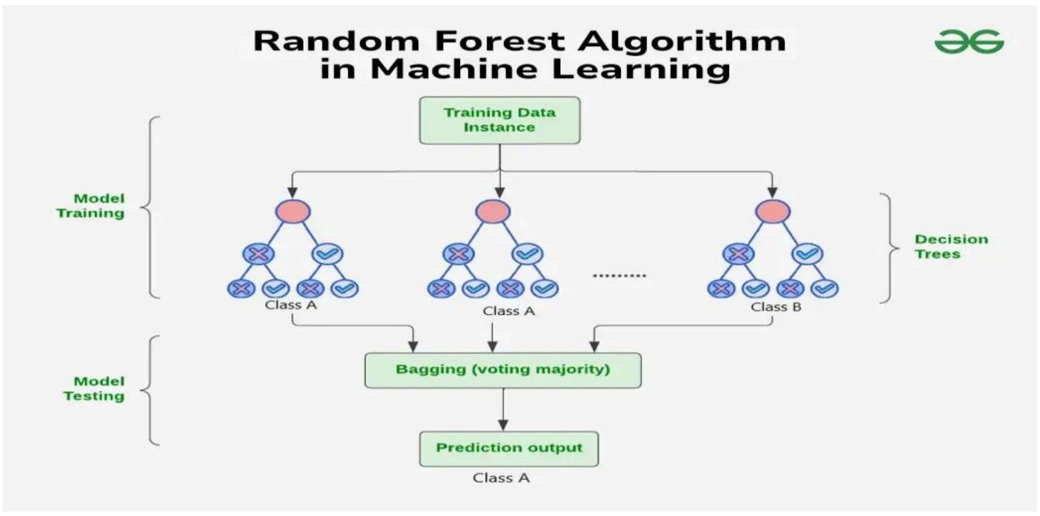

Bagging
dataset
Random Bootstraped samples
BS1
BS2
BS3
Classifier 1
Classifier 2
Classifier 3
Build multiple learners in parallel
Combine (averaging, voting, ...)
Prediction

- Here is an overview of the steps including Bagging classifier algorithm:
- **Bootstrap Sampling:** Divides the original training data into 'N' subsets and randomly selects a subset with replacement in some rows from other subsets. This step ensures that the base models are trained on diverse subsets of the data and there is no class imbalance.
- **Base Model Training**: For each bootstrapped sample, train a base model independently on that subset of data. These weak models are trained in parallel to increase computational efficiency and reduce time consumption.

- **Prediction Aggregation:** To make a prediction on testing data combine the predictions of all base models. For classification tasks, it can include majority voting or weighted majority while for regression, it involves averaging the predictions.
- **Out-of-Bag (OOB) Evaluation:** Some samples are excluded from the training subset of particular base models during the bootstrapping method. These “out-of-bag” samples can be used to estimate the model’s performance without the need for cross-validation.
- **Final Prediction:** After aggregating the predictions from all the base models, Bagging produces a final prediction for each instance.

Random Forest:

- The idea is largely that if one tree is good, then many trees (a forest) should be better, provided that there is enough variety between them.
- It works by creating a number of Decision Trees during the training phase.
- Each tree is constructed using a random subset of the data set to measure a random subset of features in each partition.
- This randomness introduces variability among individual trees, reducing the risk of overfitting and improving overall prediction performance.
- In prediction, the algorithm aggregates the results of all trees, either by voting (for classification tasks) or by averaging (for regression tasks)

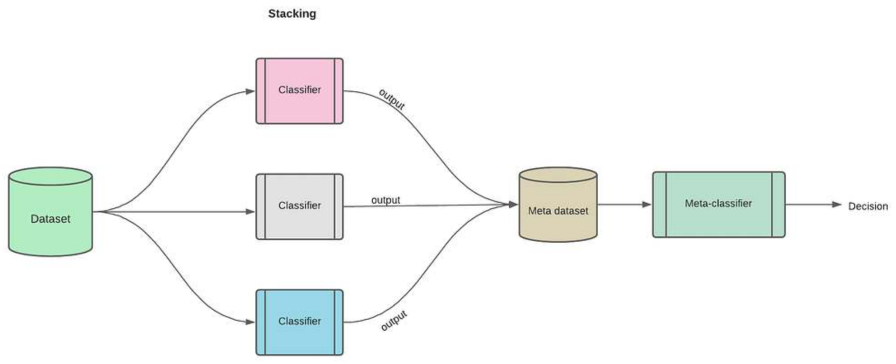


The Basic Random Forest Training Algorithm

- For each of N trees:
  - create a new bootstrap sample of the training set
  - use this bootstrap sample to train a decision tree
  - at each node of the decision tree, randomly select m features, and compute the information gain (or Gini impurity) only on that set of features, selecting the optimal one
  - repeat until the tree is complete

Stacking:

- Stacking combines many ensemble methods in order to build a meta-learner.
- Stacking has two levels of learning: 1) base learning and 2) meta-learning.
- In the first one, the base learners are trained with training data set.
- Once trained, the base learners create a new data set for a meta-learner.
- The meta-learner is then trained with that new training data set.
- Finally, the trained meta-learner is used to classify new instances.



Different ways to combine classifiers:

- If the number of classifiers is odd and the classifiers are each independent of each other, then majority voting will return the correct label if more than half of the classifiers agree.
- For regression problems, rather than taking the majority vote, it is common to take the mean of the outputs.
- However, the mean is heavily affected by outliers, with the result that the median is a more common average to use.
- It is the use of the median that produces the bagging algorithm, which is meant to imply ‘robust bagging’.

Basic Statistics:

Mean:

- The "mean" is the average value of a dataset.
- It is calculated by adding up all the values in the dataset and dividing by the number of observations.
- The mean is a useful measure of central tendency because it is sensitive to outliers, meaning that extreme values can significantly affect the value of the mean.

Median:

- The "median" is the middle value in a dataset.
- It is calculated by arranging the values in the dataset in order and finding the value that lies in the middle.
- If there are an even number of values in the dataset, the median is the average of the two middle values.

- The median is a useful measure of central tendency because it is not affected by outliers, meaning that extreme values do not significantly affect the value of the median.

Mode:

- The "mode" is the most common value in a dataset.
- It is calculated by finding the value that occurs most frequently in the dataset.
- If there are multiple values that occur with the same frequency, the dataset is said to be bimodal, trimodal, or multimodal.
- The mode is a useful measure of central tendency because it can identify the most common value in a dataset.
- However, it is not a good measure of central tendency for datasets with a wide range of values or datasets with no repeating values.

Variance:

- Variance is a measure of how much the data for a variable varies from it's mean.

$$\text{Variance } (s^2) = \sum \frac{(x_i - \bar{x})^2}{N-1}$$

Where,

- x_i is the ith observation,
- $\bar{x}$ is the mean, and
- N is the number of observations

Covariance:

- Covariance is a measure of relationship between two variables that is scale dependent, i.e. how much will a variable change when another variable changes.

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

Where,

- x_i is the ith observation in variable x,
- $\bar{x}$ is the mean for variable x,
- y_i is the ith observation in variable y,
- $\bar{y}$ is the mean for variable y, and
- N is the number of observations

Standard Deviation:

- The square root of the variance is known as the standard deviation

Mahalanobis Distance:

- Mahalanobis Distance is a statistical tool used to measure the distance between a point and a distribution.
- It is a powerful technique that considers the correlations between variables in a dataset, making it a valuable tool in various applications such as outlier detection, clustering, and classification.
  $D^2 = (x-\mu)^T\Sigma^{-1}(x-\mu)$

Where D^2 is the squared Mahalanobis Distance, x is the point in question, μ is the mean vector of the distribution, Σ is the covariance matrix of the distribution, and ^{T} denotes the transpose of a matrix.

The Gaussian / Normal Distribution:

- Normal distribution, also known as the Gaussian distribution, is a continuous probability distribution that is symmetric about the mean, depicting that data near the mean are more frequent in occurrence than data far from the mean.

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

$$\begin{aligned} \mu &= \text{ mean of } x \\ \sigma &= \text{ standard deviation of } x \\ \pi &\approx 3.14159 \ldots \\ e &\approx 2.71828 \ldots \end{aligned}$$

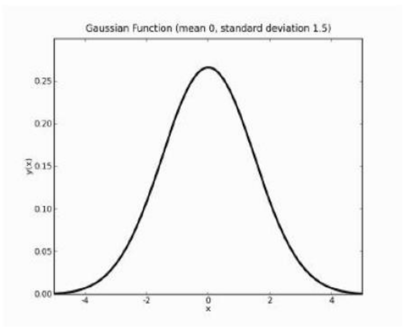
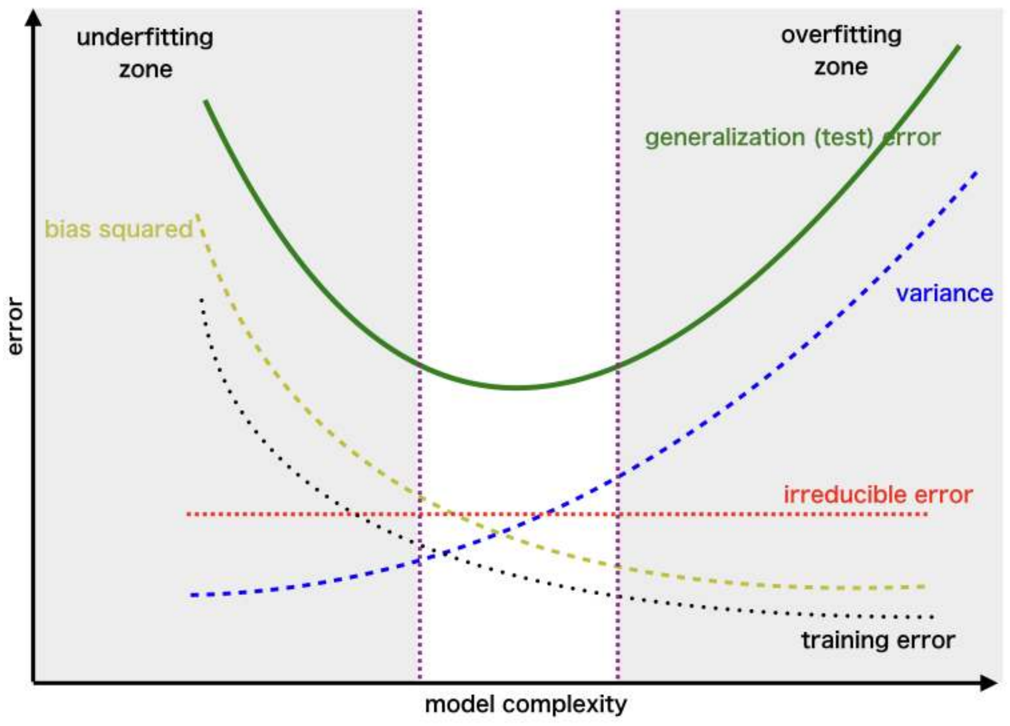


FIGURE 2.14 Plot of the one-dimensional Gaussian curve.

The bias and Variance Trade-off:

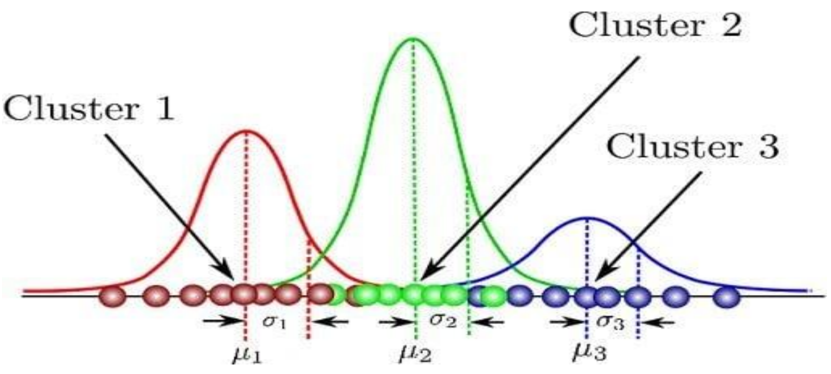
- Bias is the difference between the average prediction of our model and the correct value which we are trying to predict.
- Model with high bias pays very little attention to the training data and oversimplifies the model. It always leads to high error on training and test data.
- Variance is the variability of model prediction for a given data point or a value which tells us spread of our data.

- Model with high variance pays a lot of attention to training data and does not generalize on the data which it hasn't seen before.
- As a result, such models perform very well on training data but has high error rates on test data.
- If our model is too simple and has very few parameters then it may have high bias and low variance.
- On the other hand if our model has large number of parameters then it's going to have high variance and low bias.
- So we need to find the right/good balance without overfitting and underfitting the data.



Gaussian Mixture Models:

- GMM is blending multiple Gaussian distributions to form a single model.
- A Gaussian mixture model (GMM) is a machine learning method used to determine the probability each data point belongs to a given cluster. The model is a soft clustering method used in unsupervised learning.



- In soft clustering, instead of forcefully assigning a data point to a single cluster, GMM assigns probabilities that indicate the likelihood of that data point belonging to each of the Gaussian components.

Notation:

- **K:** Number of Gaussian components
- **N:** Number of data points
- **D:** Dimensionality of the data

GMM Parameters:

- **Means (μ):** Center locations of Gaussian components.
- **Covariance Matrices (Σ):** Define the shape and spread of each component.
- **Weights (π):** Probability of selecting each component.

Model Training

- Training a GMM involves setting the parameters using available data.
- The Expectation-Maximization (EM) technique is often employed, alternating between the Expectation (E) and Maximization (M) steps until convergence.

Expectation-Maximization:

- During the E step, the model calculates the probability of each data point belonging to each Gaussian component.
- The M step then adjusts the model's parameters based on these probabilities.

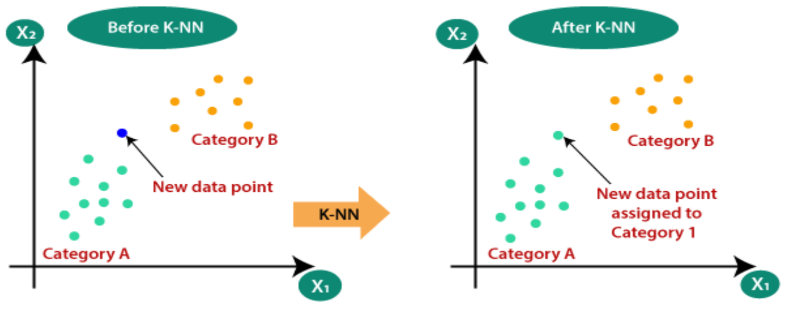
Clustering and Density Estimation:

- Post-training, GMMs cluster data points based on the highest posterior probability.
- They are also used for density estimation, assessing the probability density at any point in the feature space.

Nearest Neighbour Methods:

K-Nearest Neighbors Algorithm:

- The K-Nearest Neighbors (KNN) algorithm is a supervised machine learning method employed to tackle classification and regression problems.



Step 1: Selecting the optimal value of K

- K represents the number of nearest neighbors that needs to be considered while making prediction.

Step 2: Calculating distance

- To measure the similarity between target and training data points, Euclidean distance is used. Distance is calculated between each of the data points in the dataset and target point.

Step 3: Finding Nearest Neighbors

- The k data points with the smallest distances to the target point are the nearest neighbors.

Step 4: Voting for Classification or Taking Average for Regression

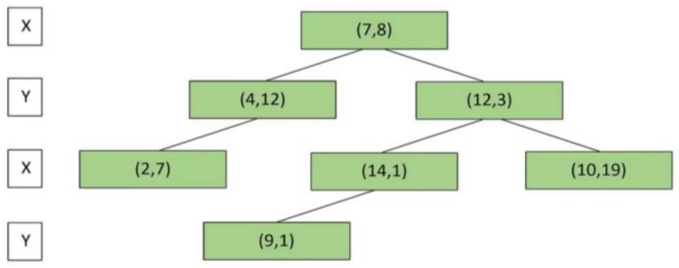
- In the classification problem, the class labels of K-nearest neighbors are determined by performing majority voting. The class with the most occurrences among the neighbors becomes the predicted class for the target data point.
- In the regression problem, the class label is calculated by taking average of the target values of K nearest neighbors. The calculated average value becomes the predicted output for the target data point.

K Dimensional Tree (KD Tree):

- KDTree is a space partitioning data structure for organizing points in K-Dimensional space.
- It is an improvement over KNN.
- It is useful for representing data efficiently.
- In KDTree the data points are organized and partitioned on the basis of some specific conditions.
- The purpose of the tree was to store spatial data with the goal of accomplishing:

  1. Nearest neighbor search.
  2. Range queries.
  3. Fast look-up.

Example:
A simple example to showcase the insertion into a K-Dimensional Tree, we will use a k = 2. The points we will be adding are: (7,8), (12,3), (14,1), (4,12), (9,1), (2,7), and (10,19).



Unsupervised Learning:

- Unsupervised learning is a type of machine learning that learns from unlabeled data.
- This means that the data does not have any pre-existing labels or categories.
- The goal of unsupervised learning is to discover patterns and relationships in the data without any explicit guidance.

Types of Unsupervised Learning:

Unsupervised learning is classified into two categories of algorithms:

- **Clustering:** A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior. Clustering is a type of unsupervised learning that is used to group similar data points together.
- **Association:** An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

Applications of Unsupervised learning:

- **Anomaly detection:** Unsupervised learning can identify unusual patterns or deviations from normal behavior in data, enabling the detection of fraud, intrusion, or system failures.
- **Scientific discovery:** Unsupervised learning can uncover hidden relationships and patterns in scientific data, leading to new hypotheses and insights in various scientific fields.
- **Recommendation systems:** Unsupervised learning can identify patterns and similarities in user behavior and preferences to recommend products, movies, or music that align with their interests.
- **Customer segmentation:** Unsupervised learning can identify groups of customers with similar characteristics, allowing businesses to target marketing campaigns and improve customer service more effectively.
- **Image analysis:** Unsupervised learning can group images based on their content, facilitating tasks such as image classification, object detection, and image retrieval.

K Means Algorithm:

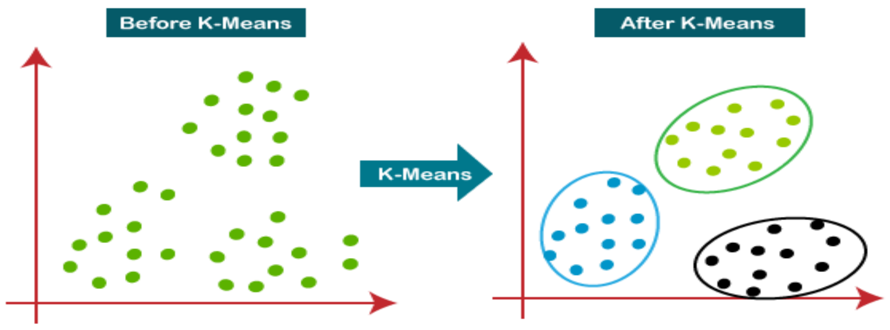
- K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters. Here K defines the number of pre-defined clusters

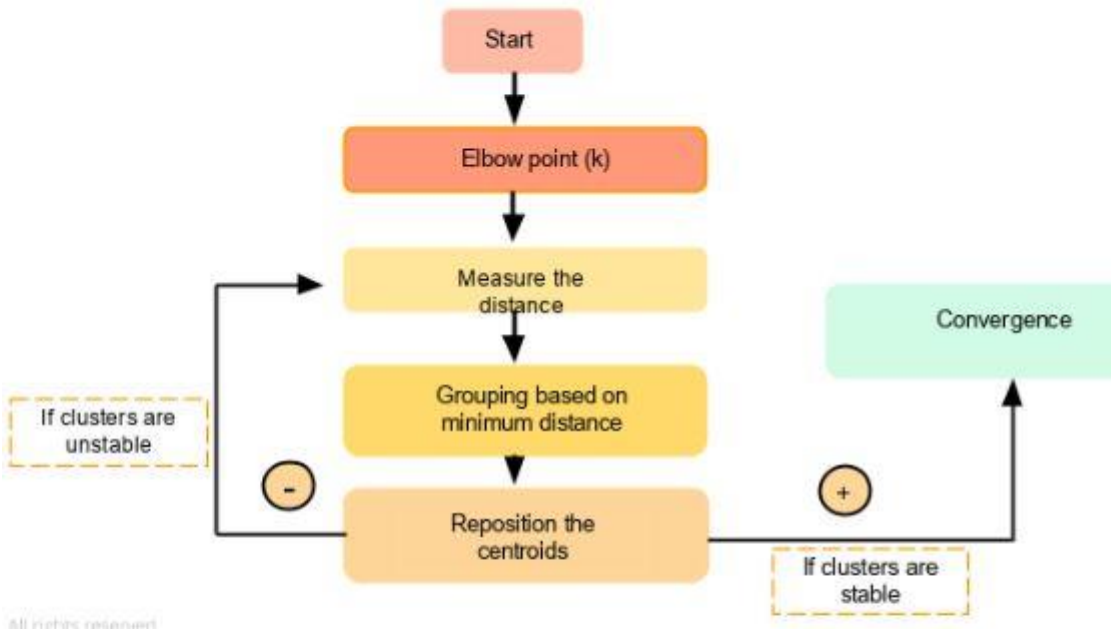
The k-Means Algorithm

```
• Initialisation
    – choose a value for $k$
    – choose $k$ random positions in the input space
    – assign the cluster centres $\mu_j$ to those positions
• Learning
    – repeat
        * for each datapoint $\mathbf{x}_i$:
            · compute the distance to each cluster centre
            · assign the datapoint to the nearest cluster centre with distance
                  $d_i = \min_j d(\mathbf{x}_i, \mu_j)$.    (14.1)
        * for each cluster centre:
            · move the position of the centre to the mean of the points in that cluster
              ($N_j$ is the number of points in cluster $j$):
                  $\mu_j = \frac{1}{N_j} \sum_{i=1}^{N_j} \mathbf{x}_i$    (14.2)
    – until the cluster centres stop moving
• Usage
    – for each test point:
        * compute the distance to each cluster centre
        * assign the datapoint to the nearest cluster centre with distance
              $d_i = \min_j d(\mathbf{x}_i, \mu_j)$.    (14.3)
```

The k-means clustering algorithm mainly performs two tasks:

- Determines the best value for K center points or centroids by an iterative process.
- Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.





Applications of K-Means Clustering:

- Customer Segmentation
- Document Clustering
- Image Segmentation
- Recommendation Engines
- Image Compression

Advantages of K-Means Clustering:

- **Simple and Easy to implement:** The K-means algorithm is easy to understand and implement.
- **Fast and Efficient:** K-means is computationally efficient and can handle large datasets with high dimensionality.
- **Scalability:** K-means can handle large datasets with many data points and can be easily scaled to handle even larger datasets.
- **Flexibility:** K-means can be easily adapted to different applications and can be used with varying metrics of distance and initialization methods.

Disadvantages of K-Means Clustering:

- **Sensitivity to initial centroids:** K-means is sensitive to the initial selection of centroids and can converge to a suboptimal solution.

- **Requires specifying the number of clusters:** The number of clusters k needs to be specified before running the algorithm, which can be challenging in some applications.
- **Sensitive to outliers:** K-means is sensitive to outliers, which can have a significant impact on the resulting clusters.

*****