Decision Tree

Advantages:

- Compared to other algorithms decision trees requires less effort for data preparation during pre-processing.
- A decision tree does not require normalization of data.
- A decision tree does not require scaling of data as well.
- Missing values in the data also do NOT affect the process of building a decision tree to any considerable extent.
- A Decision tree model is very intuitive and easy to explain to technical teams as well as stakeholders.
- A significant advantage of a decision tree is that it forces the consideration of all possible outcomes of a decision and traces each path to a conclusion. It creates a comprehensive analysis of the consequences along each branch and identifies decision nodes that need further analysis.

Disadvantage:

- A small change in the data can cause a large change in the structure of the decision tree causing instability.
- For a Decision tree sometimes calculation can go far more complex compared to other algorithms.
- Decision tree often involves higher time to train the model.
- Decision tree training is relatively expensive as the complexity and time has taken are more.
- The Decision Tree algorithm is inadequate for applying regression and predicting continuous values.

Naive Bayes

Advantages of Using Naive Bayes Classifier

- Simple to Implement. The conditional probabilities are easy to evaluate.
- Very fast – no iterations since the probabilities can be directly computed. So this technique is useful where speed of training is important.
- If the conditional Independence assumption holds, it could give great results.

Disadvantages of Using Naive Bayes Classifier

- Conditional Independence Assumption does not always hold. In most situations, the feature show some form of dependency.
- Zero probability problem : When we encounter words in the test data for a particular class that are not present in the training data, we might end up with zero class probabilities. See the example below for more details: P(bumper | Ham) is 0 since bumper does not occuer in any ham (non-spam) documents in the training data.
- Bad binning of continuous variables with Multinomial naive bayes: Gaussian Naive Bayes
- Not great for imbalanced data: Complement Naive Bayes

K-Means Advantages :

1) If variables are huge, then K-Means most of the times computationally faster than hierarchical clustering, if we keep k smalls.

2) K-Means produce tighter clusters than hierarchical clustering, especially if the clusters are globular.

K-Means Disadvantages :

1) Difficult to predict K-Value.
2) With global cluster, it didn't work well.
3) Different initial partitions can result in different final clusters. Being dependent on initial values.
4) It does not work well with clusters (in the original data) of Different size and Different density

Apriori Algorithm :

Apriori is an algorithm for frequent itemset mining and association rule learning over relational databases. It proceeds by identifying the frequent individual items in the database and extending them to larger and larger item sets as long as those item sets appear sufficiently often in the database. The frequent itemsets determined by Apriori can be used to determine association rules which highlight general trends in the database: this has applications in domains such as market basket analysis.

Apriori uses a "bottom-up" approach, where frequent subsets are extended one item at a time (a step known as candidate generation), and groups of candidates are tested against the data. The algorithm terminates when no further successful extensions are found.Using breadth-first search and a Hash tree structure, apriori counts candidate item sets efficiently. It generates candidate item sets of length k from item sets of length k-1. Then it prunes the candidates which have an infrequent subpattern. According to the downward closure lemma, the candidate set contains all frequent k-length item sets. After that, it scans the transaction database to determine frequent item sets among the candidates.

The apriori algorithm advantages are as follows:

- This is the most simple and easy-to-understand algorithm among association rule learning algorithms
- The resulting rules are intuitive and easy to communicate to an end-user
- It doesn't require labeled data as it is fully unsupervised; as a result, you can use it in many different situations because unlabeled data is often more accessible
- Many extensions were proposed for different use cases based on this implementation—for example, there are association learning algorithms that take into account the ordering of items, their number, and associated timestamps
- The algorithm is exhaustive, so it finds all the rules with the specified support and confidence

One of the biggest limitations of the Apriori Algorithm is that it is slow. This is so because of the bare decided by the:

➔ A large number of itemsets in the Apriori algorithm dataset.
➔ Low minimum support in the data set for the Apriori algorithm.
➔ The time needed to hold a large number of candidate sets with many frequent itemsets.
➔ Thus it is inefficient when used with large volumes of datasets.

Many methods are available for improving the efficiency of the algorithm.

Hash-Based Technique
Transaction Reduction
Partitioning
Sampling
Dynamic Itemset Counting