**AIDS Lab**

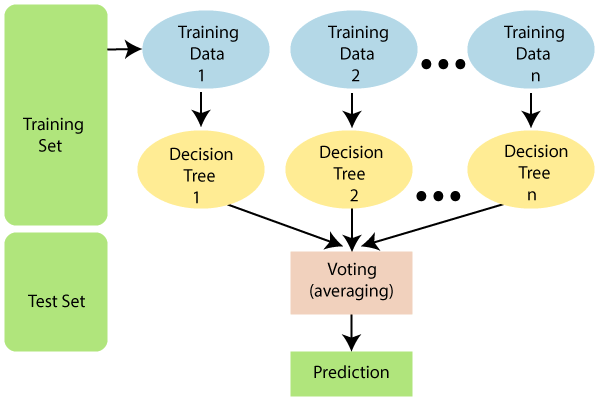
**EXPERIMENT NO. 11**

**Aim**: Implement supervised learning algorithms like Random Forest.

**Theory**:

Supervised learning is the type of machine learning in which machines are trained using well "labelled" training data, and on the basis of that data, machines predict the output. The labelled data means some input data is already tagged with the correct output. Supervised learning is a process of providing input data as well as correct output data to the machine learning model. The aim of a supervised learning algorithm is to find a mapping function to map the input variable(x) with the output variable(y).

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.



As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random forest classifier:

1. There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
2. The predictions from each tree must have very low correlations.

Random Forest works in two-phase first is to create the random forest by combining N decision trees, and second is to make predictions for each tree created in the first phase. The Working process can be explained in the below steps and diagram:

1. **Step 1**: Select random K data points from the training set.
2. **Step 2**: Build the decision trees associated with the selected data points (Subsets).
3. **Step 3**: Choose the number N for decision trees that you want to build.
4. **Step 4**: Repeat Step 1 & 2.
5. **Step 5**: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

**Code and Output**:

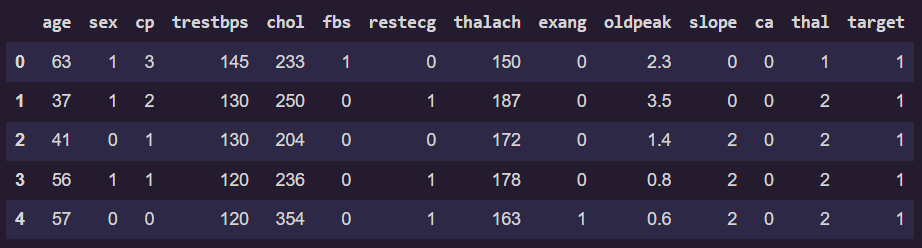
A simple record of Heart Patients monitored - Binary Classification Problem.

Heart disease is the leading cause of death among all other diseases, even cancers. Due to lack of resources in the medical field, the prediction of heart disease occasionally may be a problem. Utilisation of suitable technology support in this regard can prove to be highly beneficial to the medical fraternity and patients. The machine learning techniques can be very well adapted to do the prediction of heart disease.

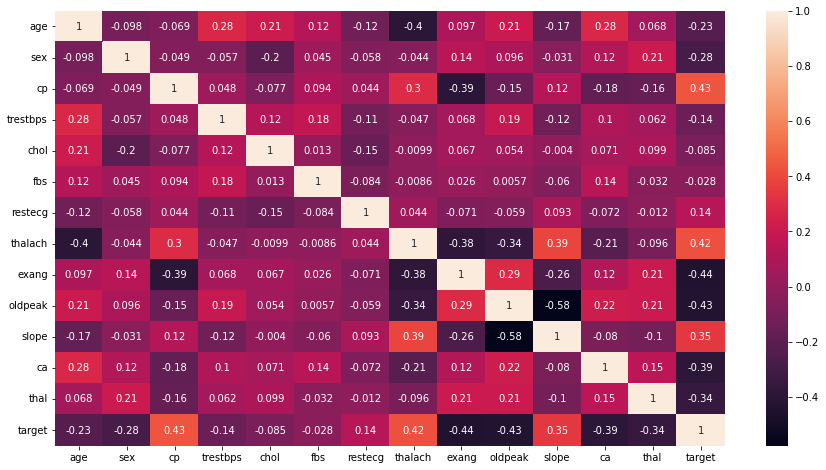
| import matplotlib.pyplot as plt import sklearn import pandas as pd df = pd.read\_csv('heart.csv') df.shape |
| --- |



| df.head() |
| --- |



| import seaborn as sns x = df.corr() plt.figure(figsize=(15, 8)) sns.heatmap(x, annot=True) |
| --- |



| X = df.drop('target', 1) y = df['target'] print('Shape of X and y respectively :', X.shape, y.shape) |
| --- |



| from sklearn.model\_selection import train\_test\_split X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=0) print('shape of X and y respectively (train) :', X\_train.shape, y\_train.shape) print('shape of X and y respectively (test) :', X\_test.shape, y\_test.shape) |
| --- |

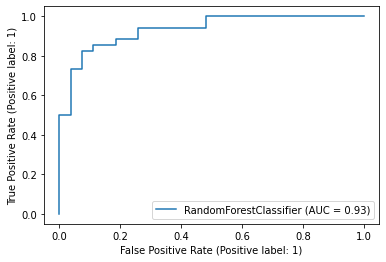


from sklearn.ensemble import RandomForestClassifier

| from sklearn.metrics import confusion\_matrix rfc = RandomForestClassifier(n\_estimators=500, criterion='entropy', max\_depth=8, min\_samples\_split=5) model = rfc.fit(X\_train, y\_train) prediction = model.predict(X\_test) cm = confusion\_matrix(y\_test, prediction) TP = cm[0][0] TN = cm[1][1] FN = cm[1][0] FP = cm[0][1] print('Testing Accuracy for Random Forest:', (TP+TN)/(TP+TN+FN+FP)\*100) print('Testing Precision for Random Forest:', (TP/(TP+FP))\*100) |
| --- |



| from sklearn.metrics import RocCurveDisplay ax = plt.gca() rfc\_disp = RocCurveDisplay.from\_estimator(model, X\_test, y\_test, ax=ax) |
| --- |



**Conclusion**:

Thus we studied an overview of the supervised learning algorithms and implemented Random Forest algorithm on a Heart Disease Prediction dataset.