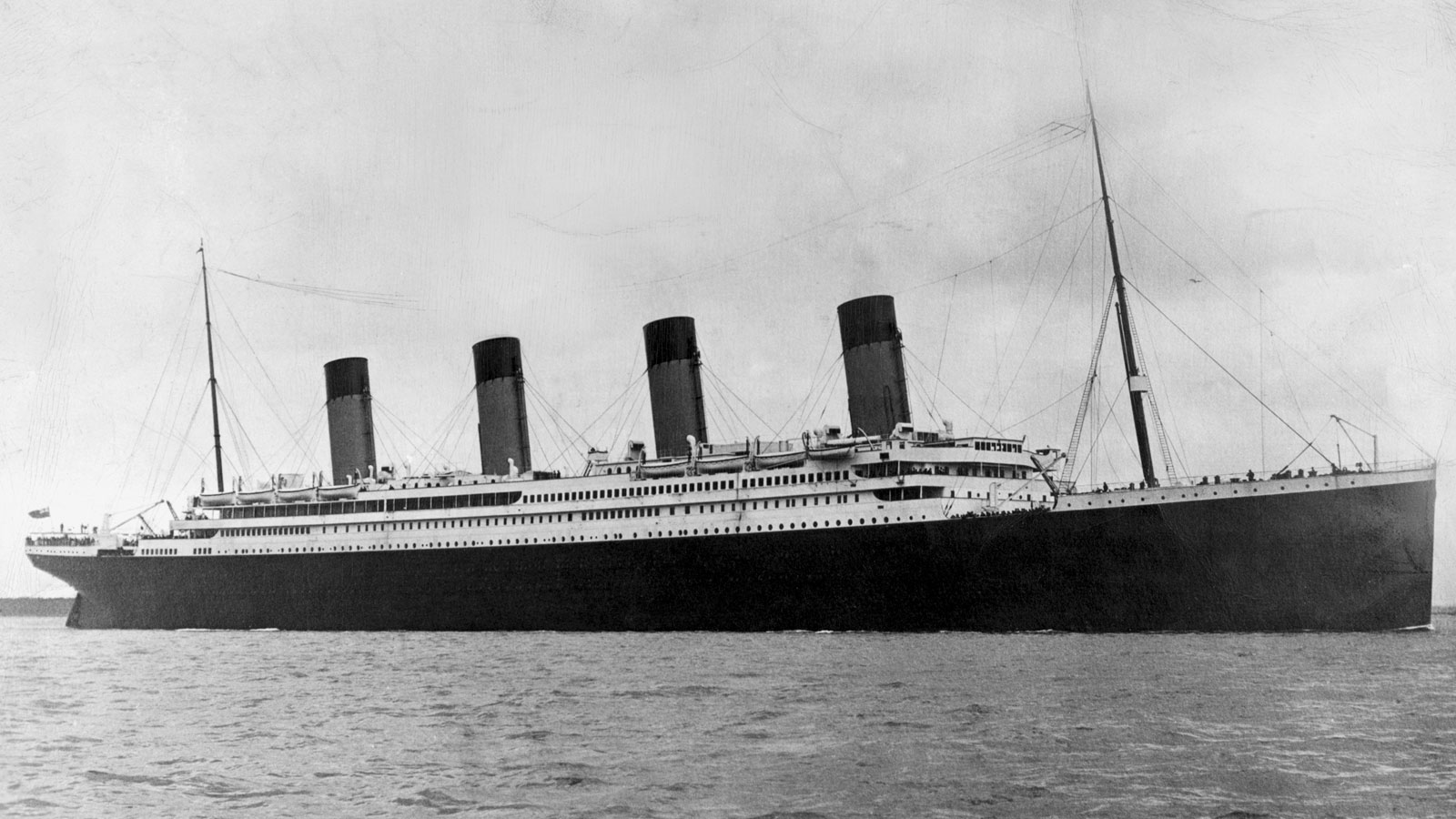
Predicting the survival of titanic passengers



# Problem definition:-

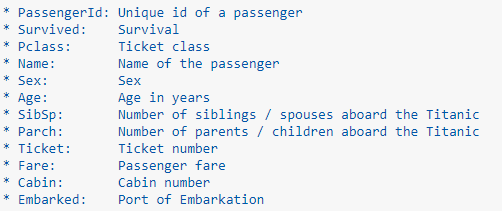
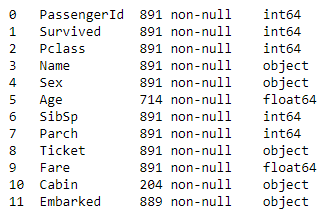
**RMS Titanic** was a British passenger liner operated by the white star line that sank in the North Atlantic Ocean on 15 April 1912, after striking an iceberg during her maiden voyage from Southampton to New York City. Of the estimated 2,224 passengers and crew aboard, more than 1500 died, making the sinking at the time on of the deadliest of the sinking ship and the deadliest peacetime sinking of a superliner or cruise ship to date. With much public attention, the disaster has since been the material of many artistic works and a founding material of the disaster film genre.

Our main objective is to predict if any arbitrary passenger on titanic would survive the sinking or not.

## Data analysis:-

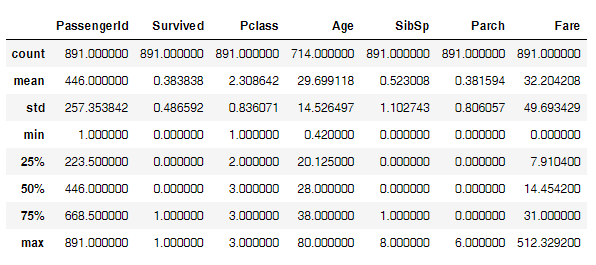
In this project, we have a dataset which has the details of each passenger in the titanic ship. It also has the column named survived to show whether that person survived or not.

The given dataset contains 891 rows and 12 columns. The description and type of each column given below.



The dataset has 891 examples and 11 features + the target variable (Survived), 2 of the features are float, 5 are integers, 5 are objects.

As we can see the dataset contains 891 entries which means 891 passengers was aboard but if you compare the total number of actual passengers aboard, it was given that around 2,224 passengers that means our dataset is not complete it is just the sample from the actual data.

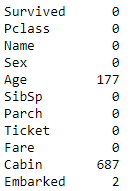


Above we can see that 38% out of the dataset survived the Titanic.

We can also see that the passenger ages range from 0.4 to 80. On top of that we can already detect some features, which contain missing values, like the ‘Age’ feature can process them. Furthermore, we can see that the features have widely different ranges, that we will need to convert into roughly the same scale. We can also spot some more features, that contain missing values (Nan = not a number), that we need to deal with.

From the table above, we can note a few things. First of all, that we need to convert a lot of features into numeric ones later on, so that the machine learning algorithms

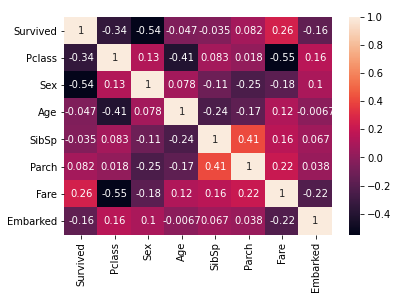
It also contains some missing values. Let’s see those missing values.



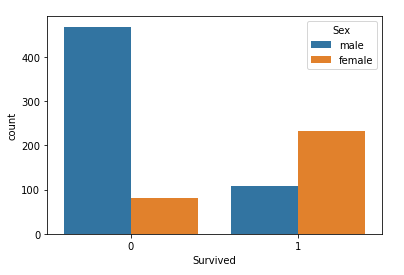
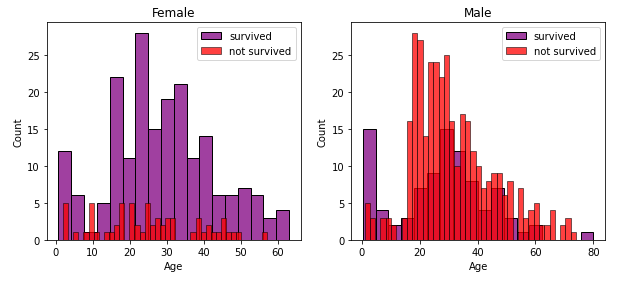
The embarked feature has only 2 missing values, which can easily be filled. It will be much more tricky, to deal with the ‘Age’ feature, which has 177 missing values. The ‘Cabin’ feature needs further investigation, but it looks like that we might want to drop it from the dataset, since 77% of it are missing.

## Exploratory data analysis:-

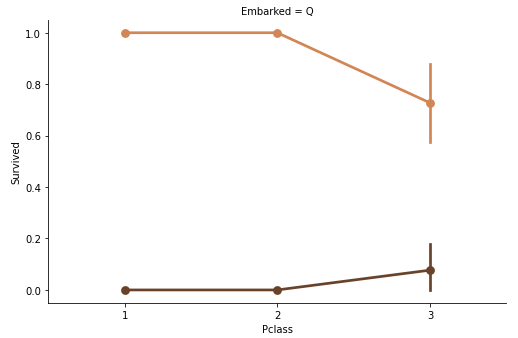
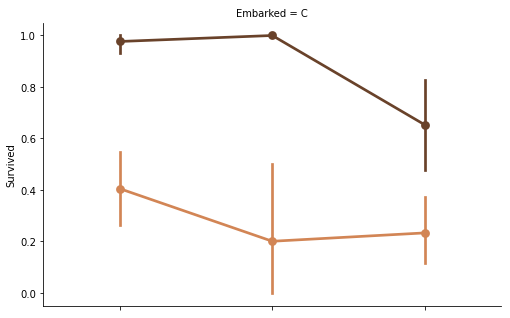
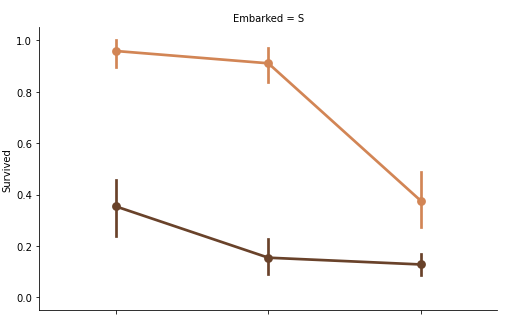
* Correlation among variables:- We can see that there is not much correlation between the variable. Only the column Sex has minimum negative correlation with the target variable.



* Visualizing variables:-



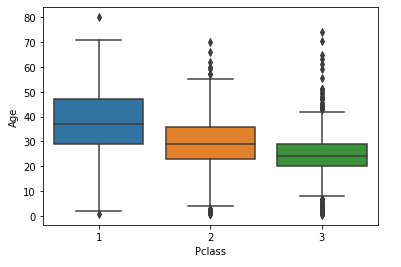
From the above histogram we can conclude that the survival probabilities of men are in the age between 20 to 35, while in case of women probability is between 15 to 40. The ratio between men and women who survived is greater in women.



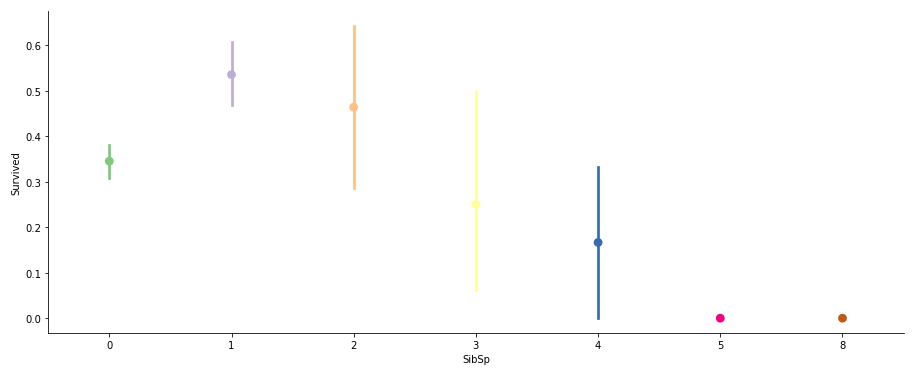
From the above point plot, we can say that the women Embarked from S and Q have higher chances of survival as compare to the women Embarked from C.

And for men, those embarked from C have the higher chance of survival as compare to those men who embarked from Q and S.

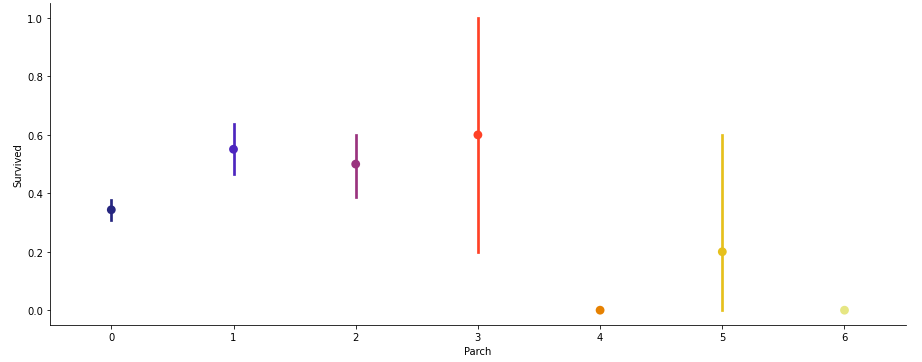
It is also clear that the men or women who are present in the Pclass 1 and 2 have the highest probability of survival.



From the above box plot, we can observe that the wealthier passenger in the first class and second class tends to actually a bit older than passengers in the third class. May be this is due to accumulate wealth it take time.



Here we can see that the chances of survival decreases as the number of Siblings/Spouse aboard on the titanic.



Here we can see that the person who has 3 Parents/children with them has the highest chances of survival.

# Pre-processing pipeline:-

Data preprocessing is a predominant step in machine learning to yield highly accurate and insightful results. Greater the quality of data, the greater is the reliability of the produced results. **Incomplete, noisy, and inconsistent data** are the inherent nature of real-world datasets. Data preprocessing helps in increasing the quality of data by filling in missing incomplete data, smoothing noise, and resolving inconsistencies.

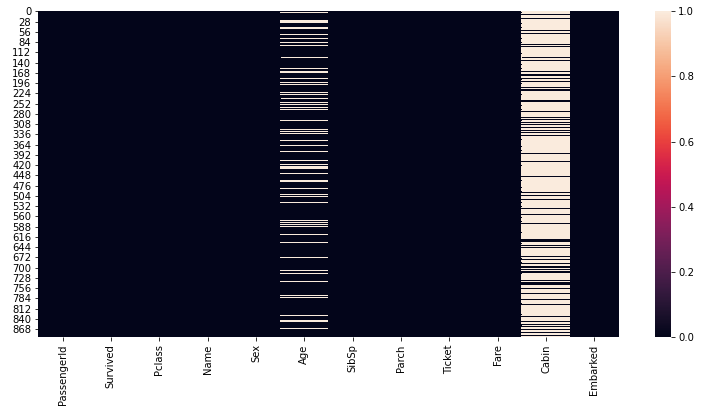
* **Incomplete data** can occur due to many reasons. Appropriate data may not be persisted due to a misunderstanding, or because of instrument defects and misfunctions.
* **Noisy data** can occur for a number of reasons (having incorrect feature values). The instruments used for the data collection might be faulty. Data entry may contain human or instrument errors. Data transmission errors might occur as well.

There are many stages involved in data preprocessing.

* **Data cleaning** attempts to impute missing values, removing outliers.
* **Data integration** integrates data from a multitude if source into single data warehouse.
* **Data transformation** such as normalization, may be applied. For example, normalization may improve the accuracy and efficiency of mining algorithms involving distance measurement.
* **Data reduction** can reduce the data size by dropping out redundant features. Feature selection and feature extraction technique can be used.

**Treating null values:-**

Sometimes there are certain columns which contains the null value, use to indicating missing or unknown values or may be the value doesn’t exist.



In our dataset there are two columns which contain the null values namely Age and Cabin. We can treat the Age column but for Cabin column the number of null values is greater than 70%, so it’s better to drop this column.

We can replace the null value in Age column by the help Pclass column, i.e. we calculate the mean age of each Pclass column and replace the age of the person of the corresponding Pclass.

**Converting labels into numeric:-**

In machine learning, we usually deal with datasets which contains multiple labels in one or more than one columns. These labels can be in the form of words or numbers. To make the data understandable or in human readable form, the training data is often labeled in words.

In our dataset there are columns like Name, Sex, Ticket, Embarked. These columns have to be treated with one hot encoding or the label encoder.

The column Name and Ticket doesn’t have to do with the target variable. So we will drop these columns. For columns Sex and Embarked we use label encoder to convert into numeric column.

**Label Encoder:-** refers to converting the labels into numeric form so as to convert it into the machine readable form. Machine learning algorithms can then decide in a better way on how those labels must be operated. It is an important pre-processing step for the structured dataset in supervised learning.

Label encoding in python can be imported from Sklearn library. Sklearn provides very efficient tool for encoding. Label encoder encode labels with a value between 0 and n\_classes-1.

Taking an example from our dataset.

# Building Machine learning model

For building machine learning model there are several models present inside the Sklearn module.

Sklearn provides two types of models i.e. regression and classification. Our dataset’s target variable is to predict whether fraud reported or not. So for this kind of problem we use classification models.

But before fitting our dataset to its model first we have to separate the predictor variable and the target variable, then we pass this variable to the train\_test\_split method to create random test and train subset.

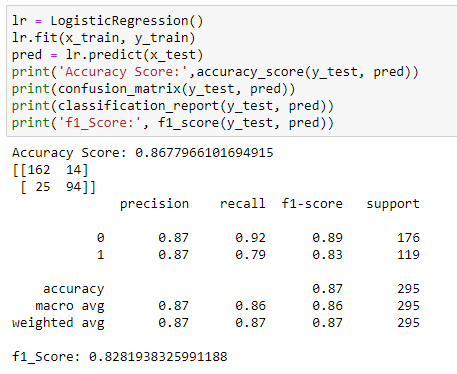
**What is train\_test\_split** is a function in sklearn model selection for splitting data arrays into two subsets for training data and testing data. With this function, you don’t need to divide dataset manually. By default, sklearn train\_test\_split will make random partitions for the two subsets. However, you can also specify a random state for the operation. It gives four output x\_train, x\_test, y\_train and y\_test. The x\_train and x\_test contains the training and testing predictor variables while y\_train and y\_test contains the training and testing target variable.

After performing train\_test\_split we have to choose the models to pass the training variable.

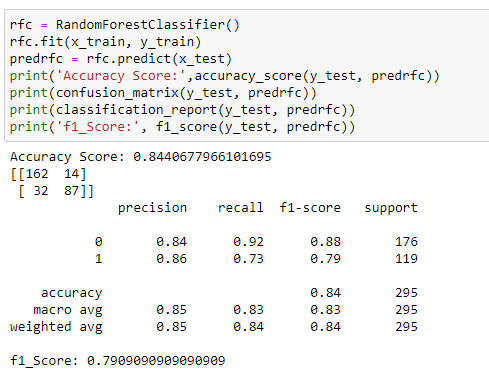
We can build as many models as we want to compare the accuracy given by these models and to select best model among them.

I have selected 5 model:-

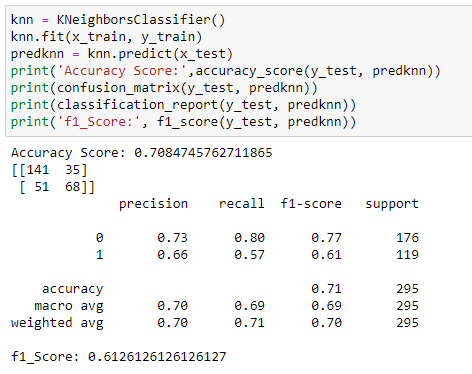
1. **Logistic Regression from sklearn.linear\_model**:- Logistic regression is a supervised learning classification algorithm used to predict the probability of a target variable. The nature of target or dependent variable is binary, which means there would be only two possible classes 1 (stands for success/yes) or 0 (stands for failure/no). Mathematically, a logistic regression model predicts P(Y=1) as a function of X. It is one of the simplest ML algorithms that can be used for various classification problems such as spam detection, Diabetes prediction, cancer detection etc.



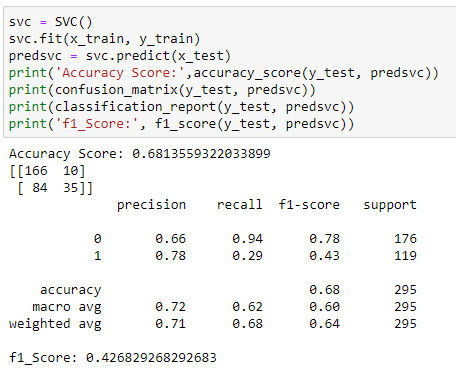
1. **RandomForestClassifier from sklearn.ensemble**:- As we know that a forest is made up of trees and more trees means more robust forest. Similarly, random forest algorithm creates decision trees on data samples and then gets the prediction from each of them and finally selects the best solution by means of voting. It is an ensemble method which is better than a single decision tree because it reduces the over-fitting by averaging the result.



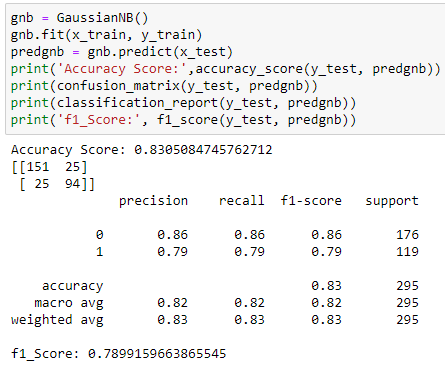
1. **KNeighborsClassifiers from sklearn.neighbors**:- K-nearest neighbors (KNN) algorithm uses ‘feature similarity’ to predict the values of new data points which further means that the new data point will be assigned a value based on how closely it matches the points in the training set.



1. **Support vector classfier:- ‘**Support vector machine’ is a supervised machine learning algorithm which can be used for both classification or regression challenge. However, it is mostly used in classification problem. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is the number of features you have) with the value of each feature being the value of a particular co-ordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well.



1. **GaussianNB from sklearn.naive\_bayes**:- Naive Bayes algorithms are a classification technique based on applying Bayes’ theorem with a strong assumption that all the predictors are independent to each other. In simple words, the assumption is that the presence of a feature in a class is independent to the presence of any other feature in the same class. It is the simplest Naïve Bayes classifier having the assumption that the data from each label is drawn from a simple Gaussian distribution.



**Conclusion:**

We got our best model i.e. Logistic Regression with the accuracy score of 86.7%.

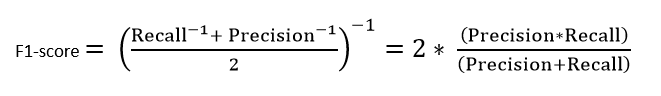
Here our model predicts 162 true positive cases out of 176 positive cases and 94 true negative cases out of 119 cases.

It predicts 14 false positive cases out of 176 positive cases and 25 false negative cases out of 119 cases.

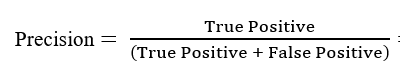
It gives the f1 score of 82.8%.

**Let’s understand what does precision recall and f1 score and accuracy means.**

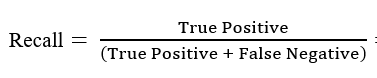
* **F1 score**: this is the harmonic mean of precision and recall and gives a better measure of the incorrectly classified cases than the accuracy matrix.



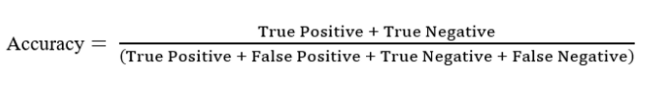
* **Precision:** It is implied as the measure of the correctly identified positive cases from all the predicted positive cases. Thus, it is useful when the costs of False Positives are high.



* **Recall:** It is the measure of the correctly identified positive cases from all the actual positive cases. It is important when the cost of False Negatives is high.

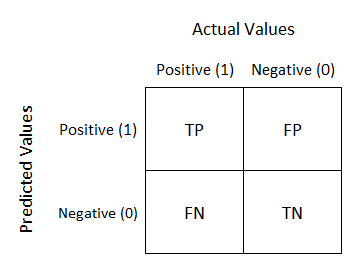


* **Accuracy:** One of the more obvious metrics, it is the measure of all the correctly identified cases. It is most used when all the classes are equally important.



**Let’s understand what confusion matrix is:**

A confusion matrix is a table that is often used to describe the performance of a classification model (or ‘classifier’) on a set of test data for which the true values are known.



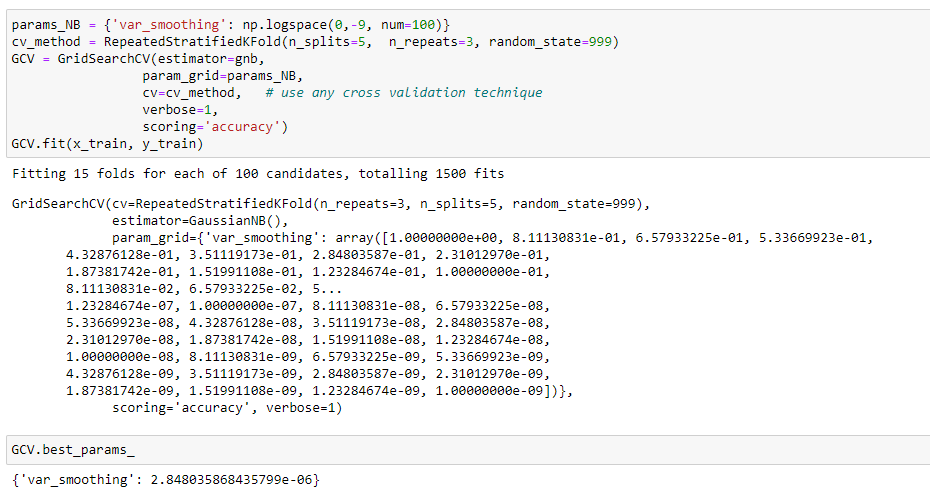
1. **TN/True Negative:** the cases were negative and predicted negative.
2. **TP/True Positive:** the cases were positive and predicted positive.
3. **FN/False Negative:** the cases were positive but predicted negative.
4. **TN/True Negative:** the cases were negative but predicted positive.

**Hyper parameter tuning:**

Hyper parameter optimization in machine learning intends to find the hyper parameters of a given machine learning algorithm that deliver the best performance as measured on a validation set. Hyper parameters, in contrast to model parameters, are set by the machine learning engineer before training. The number of trees in a random forest is a hyper parameter while the weights in a neural network are model parameters learned during training. I like to think of hyper parameters as the model settings to be tuned so that the model can optimally solve the machine learning problem.

We will use GridSearchCV for the hyper parameter tuning.

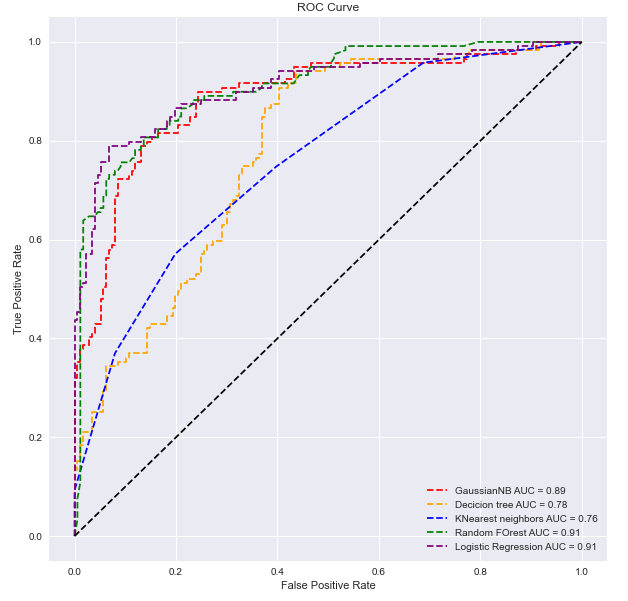
**GridSearchCV:** In GridSearchCV approach, machine learning model is evaluated for a range of hyper parameter values. This approach is called GridSearchCV, because it searches for best set of hyper parameters from a grid of hyper parameters values.



**Plotting ROC curve:**

AUC - ROC curve is a performance measurement for the classification problems at various threshold settings. ROC is a probability curve and AUC represents the degree or measure of separability. It tells how much the model is capable of distinguishing between classes. Higher the AUC, the better the model is at predicting 0s as 0s and 1s as 1s. By analogy, the Higher the AUC, the better the model is at distinguishing between patients with the disease and no disease.

The ROC curve is plotted with TPR against the FPR where TPR is on the y-axis and FPR is on the x-axis.



# Concluding remarks

We started with the data exploration where we got a feeling for the dataset, checked about missing data and learned which features are important. During this process we used seaborn and matplotlib to do the visualizations. During the data preprocessing part, we computed missing values, converted features into numeric ones, grouped values into categories and created a few new features. Afterwards we started training 5 different machine learning models, picked one of them (Logistic Regression). Then we discussed how Logistic regression works, took a look at the importance it assigns to the different features and tuned it’s performace through optimizing it’s hyperparameter values. Lastly, we looked at it’s confusion matrix and computed the models precision, recall and f-score. And at the end we plot the Receiver Operating Characteristic (ROC) of our models, and calculate AUC score.

Of course there is still room for improvement, like doing a more extensive feature engineering, by comparing and plotting the features against each other and identifying and removing the noisy features. Another thing that can improve the overall result would be a more extensive hyperparameter tuning on several machine learning models. You could also do some ensemble learning.