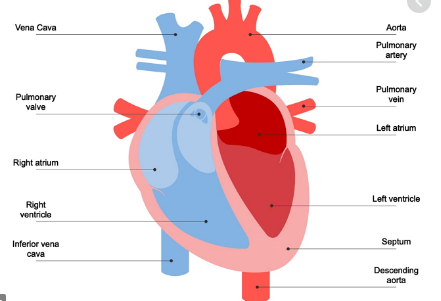
**Heart Diseases Prediction using Machine Learning**

**General Description for Heart Diseases**

Cardiovascular disease (CVD) is a class of diseases that involve the heart or blood vessels. CVD includes coronary artery diseases (CAD) such as angina and myocardial infarction (commonly known as a heart attack). Other CVDs include stroke, heart failure, hypertensive heart disease, rheumatic heart disease, cardiomyopathy, abnormal heart rhythms, congenital heart disease, valvular heart disease, carditis, aortic aneurysms, peripheral artery disease, thromboembolic disease, and venous thrombosis.

**Heart Diagram**



We are using here Machine Learning algorithms to predict if the patient is prone to Heart diseases.

**Data Set Information:**

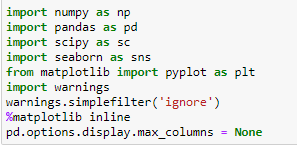
This database contains 14 attributes. In particular, the Cleveland database is the only one that has been used by ML researchers to this date. The "goal" field refers to the presence of heart disease in the patient. It is integer valued from 0 (no presence) to 4. Experiments with the Cleveland database have concentrated on simply attempting to distinguish presence (values 1,2,3,4) from absence (value 0).

Only 14 attributes used:

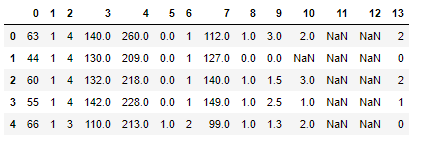
Only 14 attributes used:

(age),(sex),(cp),(trestbps),(chol),(fbs),(restecg),(thalach),(exang),(oldpeak),(slope),(ca),(thal),(num) (the predicted attribute)

We have imported all the required libraries which we will need for the data analysis(Data Preparation)



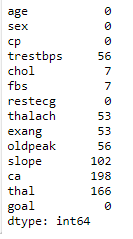
Below is the Data which is loaded in the jupyter notebook :



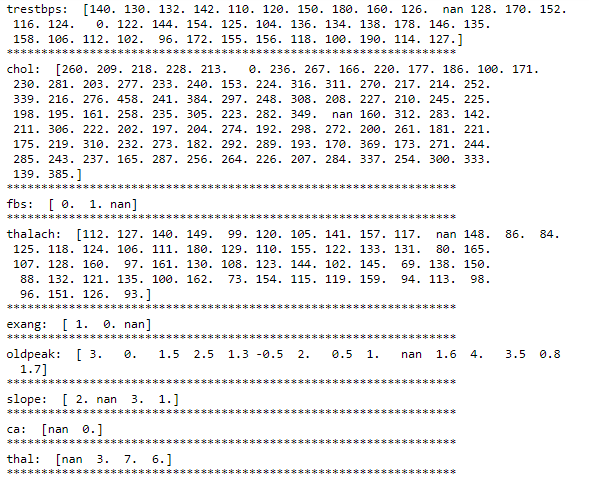
The columns are numbered as per the sequence. This way it is difficult to relate the values in the column with its nature/type. Hence we have updated the column names in the Dataset as below:

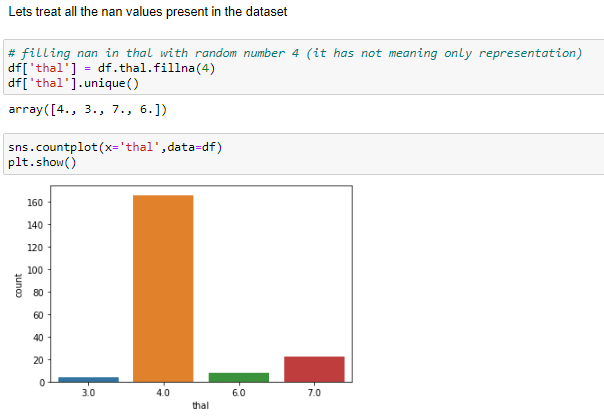


We have null values also in this dataset and in order to successfully able to built the model, we need to replace the null values with the relevant values for that particular column. For numerical data/column we can use either Mean, Median or Mode. Below process deals with the Null values treatment:

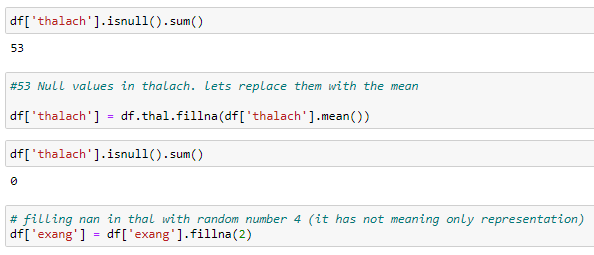


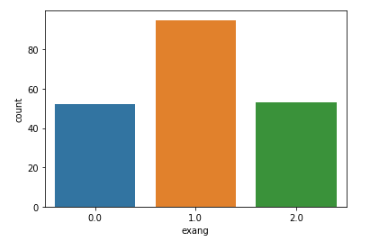
Unique values and Null values check in the dataframe:



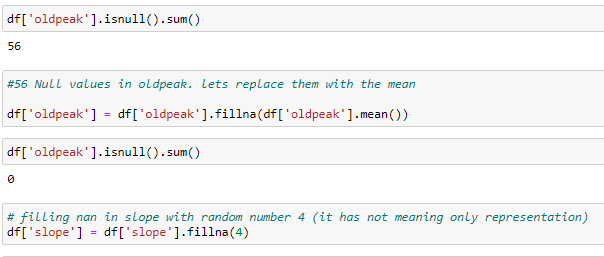


Lets look now at the thalach and exang filed and fill the null values with the relevant data. We are using the mean for thalach to replace null values. For exang column we are using value 4. It is used only for representation.

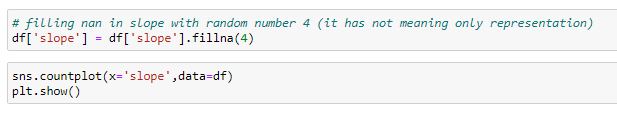




Similarly we will treat the oldpeak and slope column and replace the Null values.

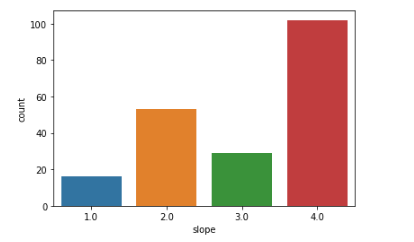


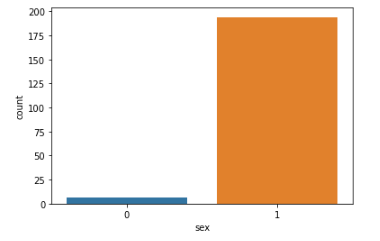
Lets now visualize the trend in the dataset with the help of some graphs. We will use here seaborn library for the same.

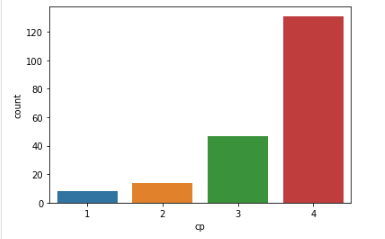
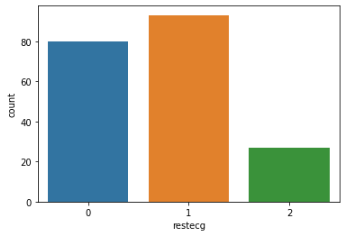
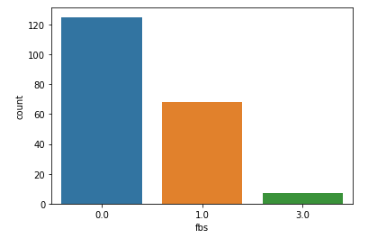


Column slope is a categorical variable and we have seen it has 3 categories and some null values. so we have included all the null values together and created a separate column and gave it value representation as 4.

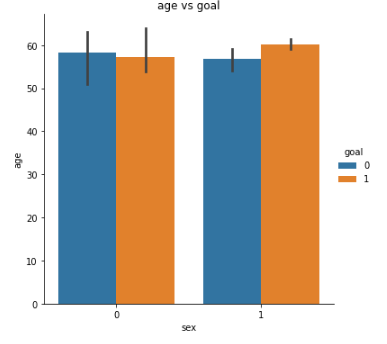
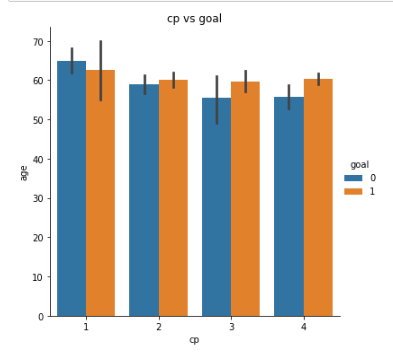
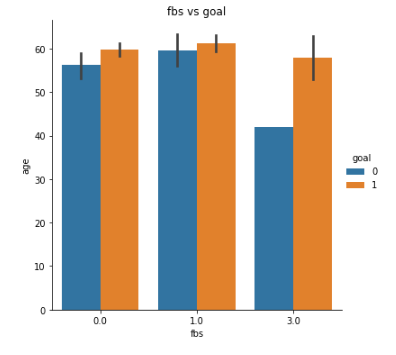
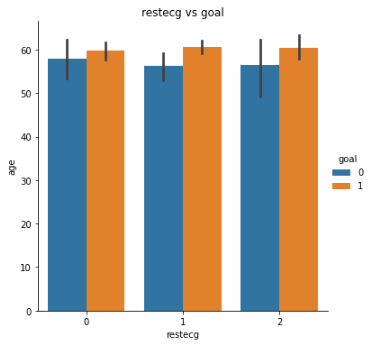
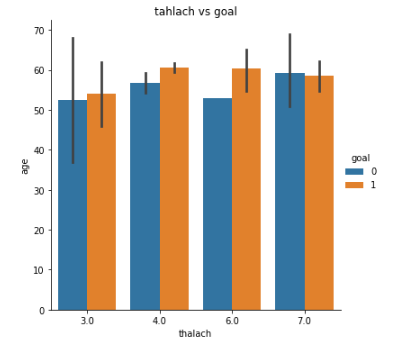
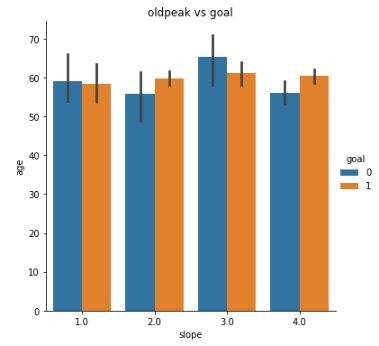
Below in the count plot for the slope, sex, cp and other columns:



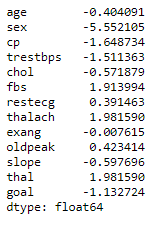


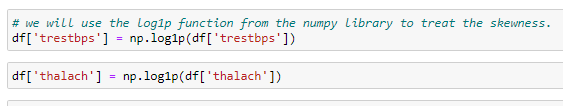
Lets now look at the catplot of various columns with our target variable goal:

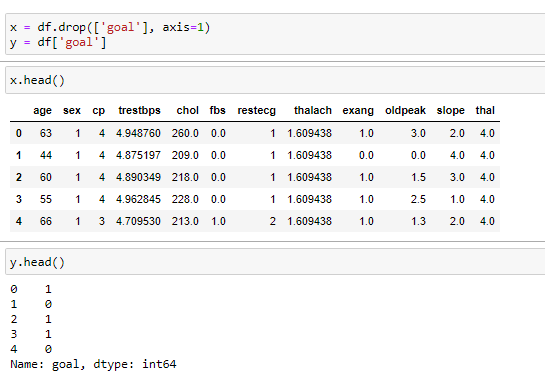
Lets check the skewness in the dataframe:



We can see that some of the columns have skewness outside the acceptable range in this case, i.e. (-1,1). Lets remove the skewness from the trestbps and thalach column. We will use the log1p method which is present in the numpy module. Below code snipet shows the process for the same.



Dividing the datset into x(predictors) and y(target)

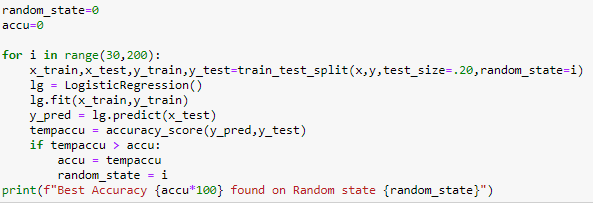


Lets find out the best random state for the train test split. We are using here the logistic regression to find out the best accuracy and check what the random state for this accuracy is.

Lets import the required libraries for this:



We used a for loop on a range of 30 to 200 and inside the loop we are building the logistic regression model and splitting the dataset into train and test. We are then printing the highest accuracy score and the value of I (Random state) at which this accuracy is achieved.

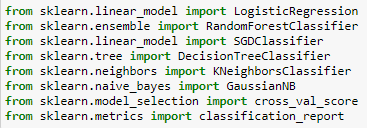


Best Accuracy 90.0 found on Random state 33

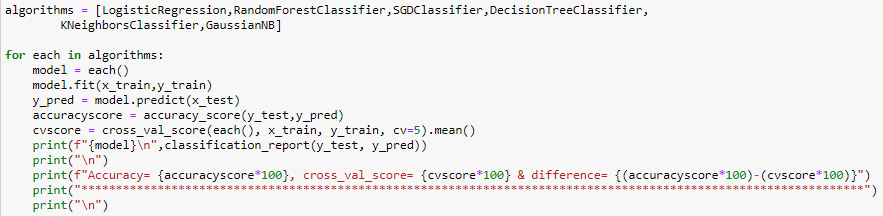
We achieved an accuracy of 90% on random state 33. We will use this random state now to do the train test split of the Dataset.



We will now use the training dataset on the below models and see which gives us the best accuracy and result:

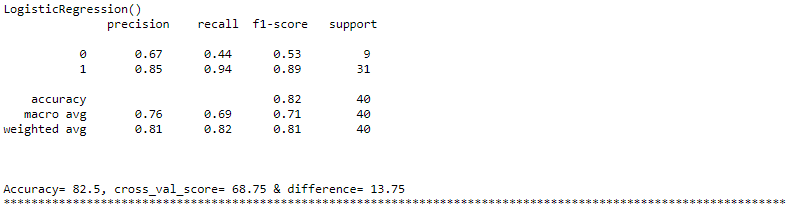


We created a list of the algorithms and passed it into the model in a for loop and printed the accuracy and cvscore as shown below:



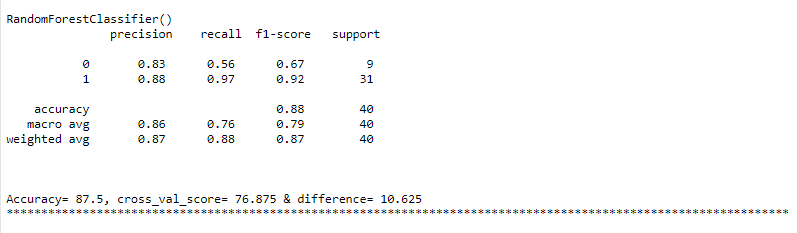
Below are results that we got the from the models that we used above in the for loop:

Logistic Regression:



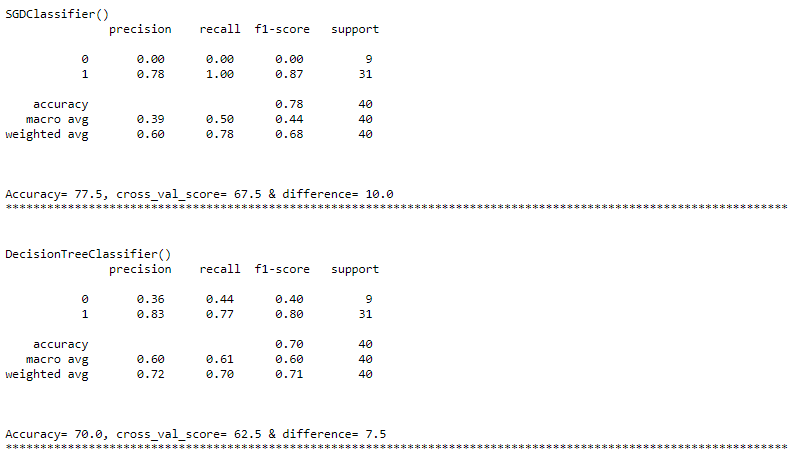
We achieved an accuracy of 82.5 and Cross\_val\_score of 68.75 from logistic regression.

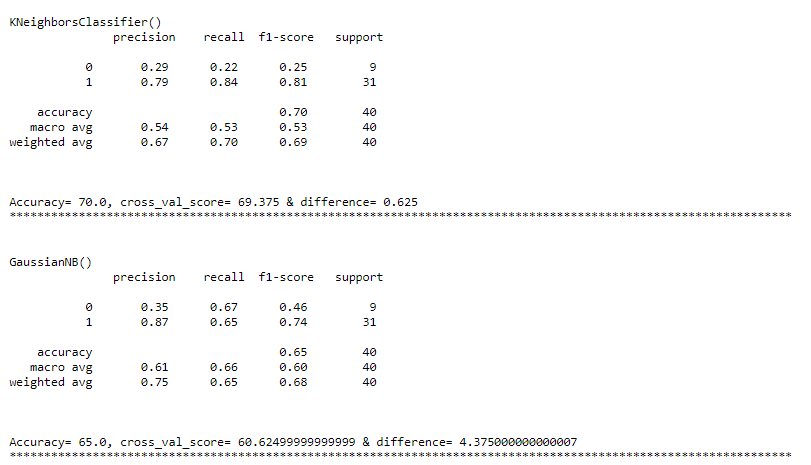
RandomForestClassifier()



We achieved an accuracy of 87.5 and Cross\_val\_score of 76.88 from logistic RandomForestClassifier which is slightly better than the LogisticRegression result. Similarly now we will check for the other models:

SGDClassifier and DecisionTree Classifier both have less accuracy than the RandomForestClassifier and Logistic Regression model.





Both the KNeighborsClassifier and GaussianNB have less accuracy than the previous models.

Hence for this project we can infer that the Random Forest Classifier gave us the best result i.e. an accuracy of 87.5% and

Source:

1. Heart Diagram : <https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.edrawmax.com%2Farticle%2Fa-guide-to-understand-human-heart-with-heart-diagram.html&psig=AOvVaw0a7s542ZGRT_kB3djjr9jV&ust=1614621420162000&source=images&cd=vfe&ved=0CAIQjRxqFwoTCMCO-tqTje8CFQAAAAAdAAAAABAD>