Heart Failure Prediction

BHAVYA MANISH SHARMA(B20EE013)
GOHEL MRUGANSHI JATINBHAI(B20CS014)
MAMIDIPALLI DIVYA MEGHANA(B20CS032)

Abstract-This paper reports our experience to build the best model for correctly predicting heart failure on the heart failure prediction dataset.This dataset was created by combining different datasets already available independently but not combined before. In this dataset, 5 heart datasets are combined over 11 common features which makes it the largest heart disease dataset available so far for research purposes.

INTRODUCTION

Heart failure is a common event caused by Cardiovascular diseases (CVDs) and this dataset contains 11 features that can be used to predict a possible heart disease. People with cardiovascular disease or who are at high cardiovascular risk (due to the presence of one or more risk factors such as hypertension, diabetes, hyperlipidaemia or already established disease) need early detection and management wherein a machine learning model can be of great help.

Datasets :

- Cleveland: 303 observations
- Hungarian: 294 observations
- Switzerland: 123 observations
- Long Beach VA: 200 observations
- Stalog (Heart) Data Set: 270 observations

Total: 1190 observationS
Duplicated: 272 observations
Final dataset: 918 observations

METHODOLOGY

Pre-processing of dataset:
(TO MAKE THE DATASET IDEAL FOR APPLYING VARIOUS ML MODELS)

1. All the important and required libraries were imported
2. The excel sheet in .csv format containing the dataset was opened, read and loaded to form a panda dataframe.

3. Attribute Information:
    a. Age: age of the patient [years]
    b. Sex: sex of the patient [M: Male, F: Female]
    c. ChestPainType: chest pain type [TA: Typical Angina, ATA: Atypical Angina, NAP: Non-Anginal Pain, ASY: Asymptomatic]
    d. RestingBP: resting blood pressure [mm Hg]
    e. Cholesterol: serum cholesterol [mm/dl]
    f. FastingBS: fasting blood sugar [1: if FastingBS > 120 mg/dl, 0: otherwise]
    g. RestingECG: resting electrocardiogram results [Normal: Normal, ST: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV), LVH: showing probable or definite left ventricular hypertrophy by Estes' criteria]
    h. MaxHR: maximum heart rate achieved [Numeric value between 60 and 202]
    i. ExerciseAngina: exercise-induced angina [Y: Yes, N: No]
    j. Oldpeak: oldpeak = ST [Numeric value measured in depression]
    k. ST_Slope: the slope of the peak exercise ST segment [Up: upsloping, Flat: flat, Down: downsloping]
    l. HeartDisease: output class [1: heart disease, 0: Normal]
4. Converted the chest pain type encoded column to the real name of the chest pain for better understanding of this feature
   'ATA' : "Atypical Angina"
   'TA' : "Typical Angina"
   'NAP' : "Non-Anginal Pain"
   'ASY' : "Asymptomatic'
5. Checked for any null, NaN values in the dataset as these are missing data points and missing data creates imbalanced observations, cause biased estimates, and in extreme cases, can even lead to invalid conclusions

Count of null values in all features:

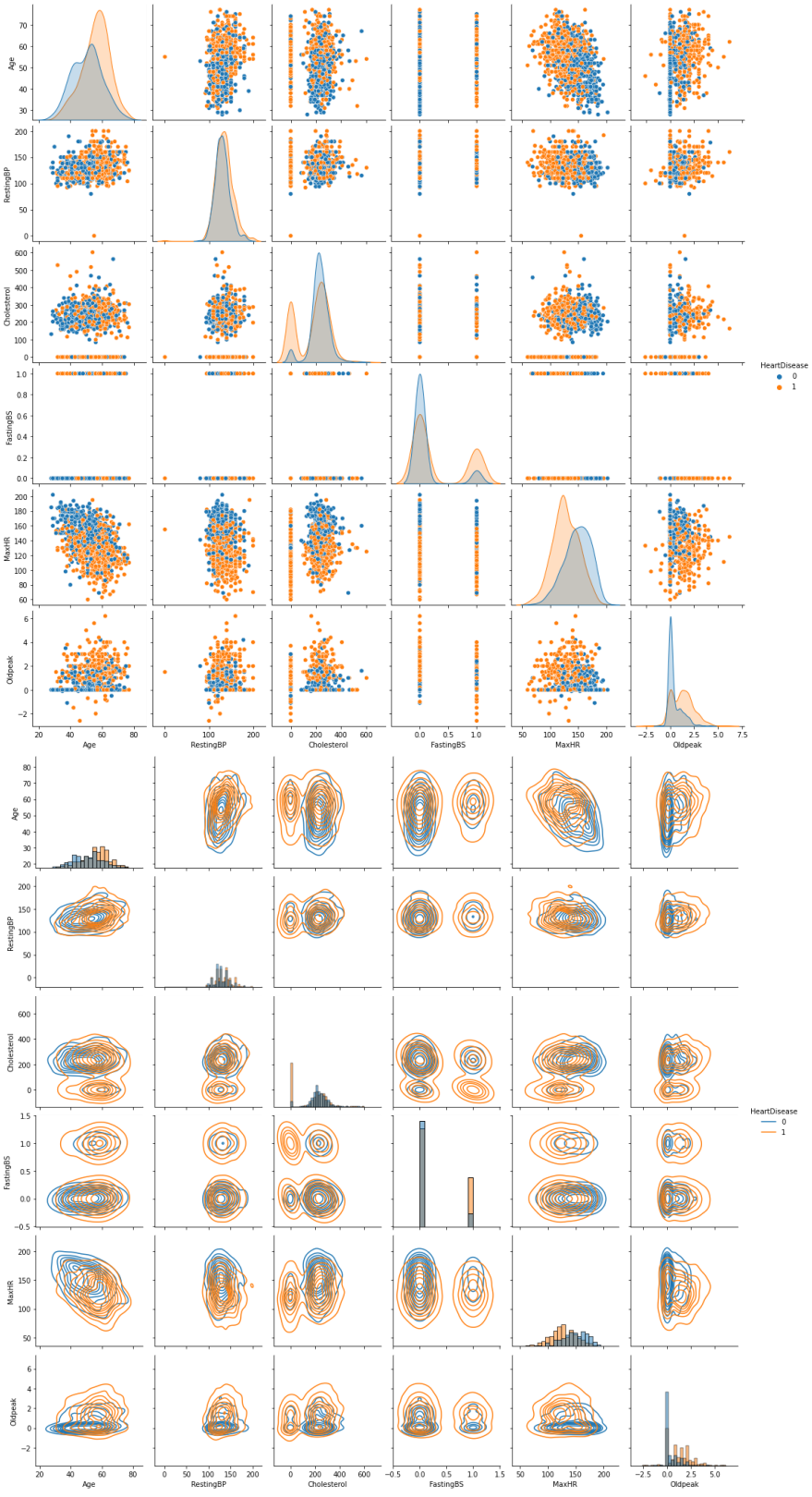
```
Age              0
Sex              0
ChestPainType    0
RestingBP        0
Cholesterol      0
FastingBS        0
RestingECG       0
MaxHR            0
ExerciseAngina   0
Oldpeak          0
ST_Slope         0
HeartDisease     0
```

```
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Age             918 non-null    int64
 1   Sex             918 non-null    object
 2   ChestPainType   918 non-null    object
 3   RestingBP       918 non-null    int64
 4   Cholesterol     918 non-null    int64
 5   FastingBS       918 non-null    int64
 6   RestingECG      918 non-null    object
 7   MaxHR           918 non-null    int64
 8   ExerciseAngina  918 non-null    object
 9   Oldpeak         918 non-null    float64
 10  ST_Slope        918 non-null    object
 11  HeartDisease    918 non-null    int64
```

No null,NaN values found in the dataset.

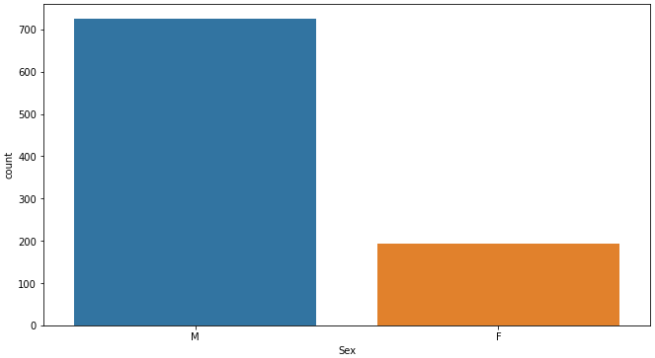
Visualization of dataset:

1. Pair plots were plotted with help of sns and matplotlib library. Pair plot is used to understand the best set of features to explain a relationship between two variables or to form the most separated clusters. It also helps to form some simple classification models by drawing some simple lines or making linear separation in our data-set.

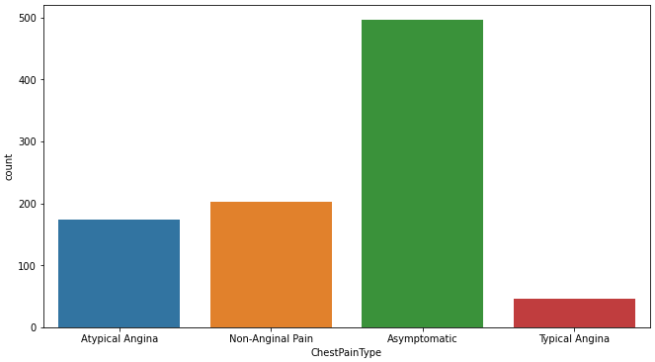


2. Count plot for different features in the dataset were plotted

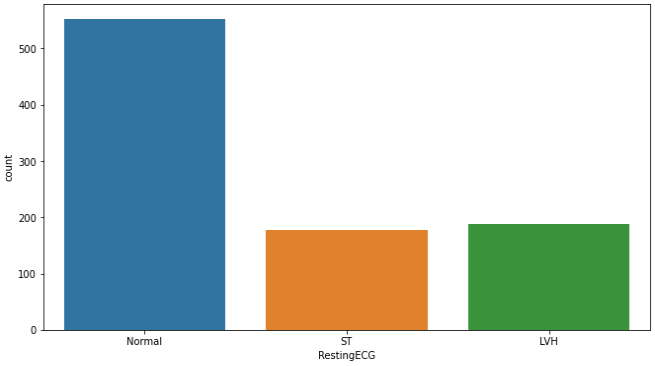
Feature:sex



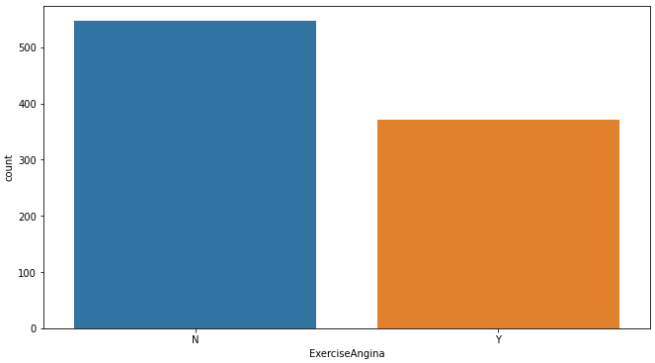
Feature:chest pain type



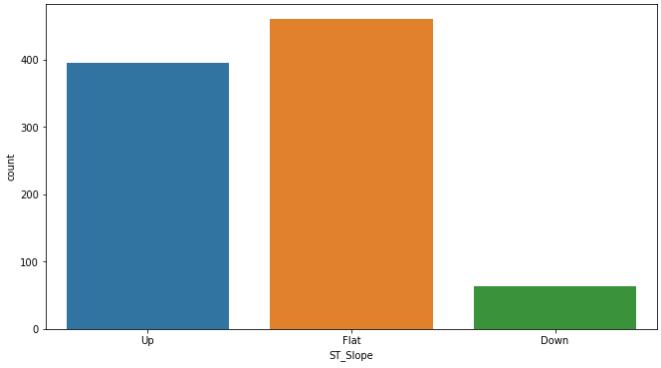
Feature:resting ECG



Feature:exercise angina

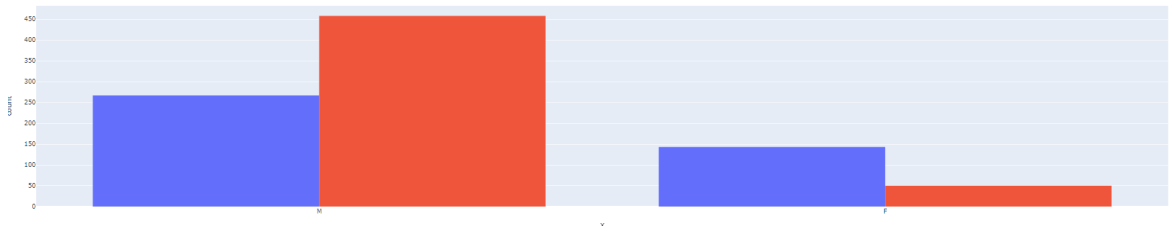


Feature:ST_slope

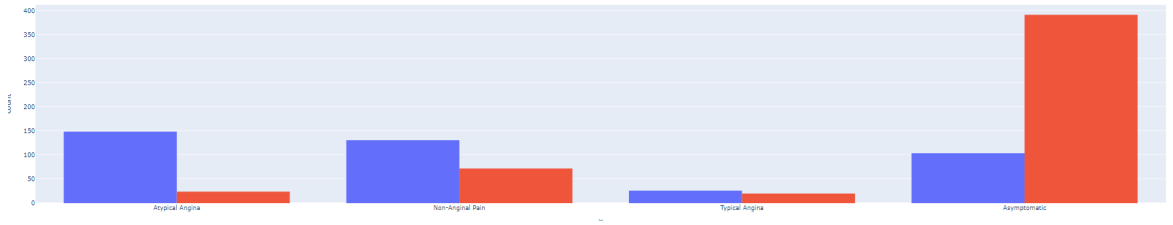


3. Class-wise cout plot for different features in the dataset were plotted

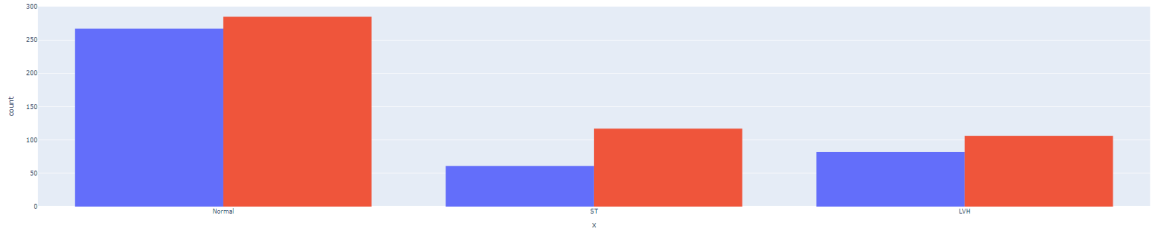
Feature:sex



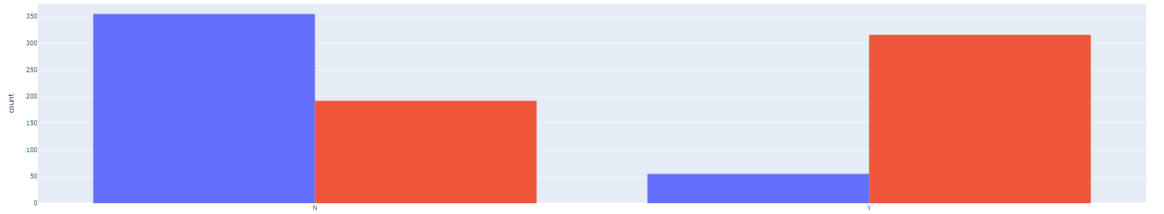
Feature:chest pain type



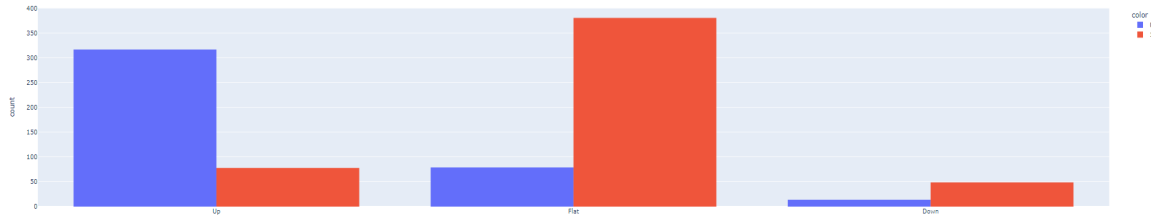
Feature:resting ECG



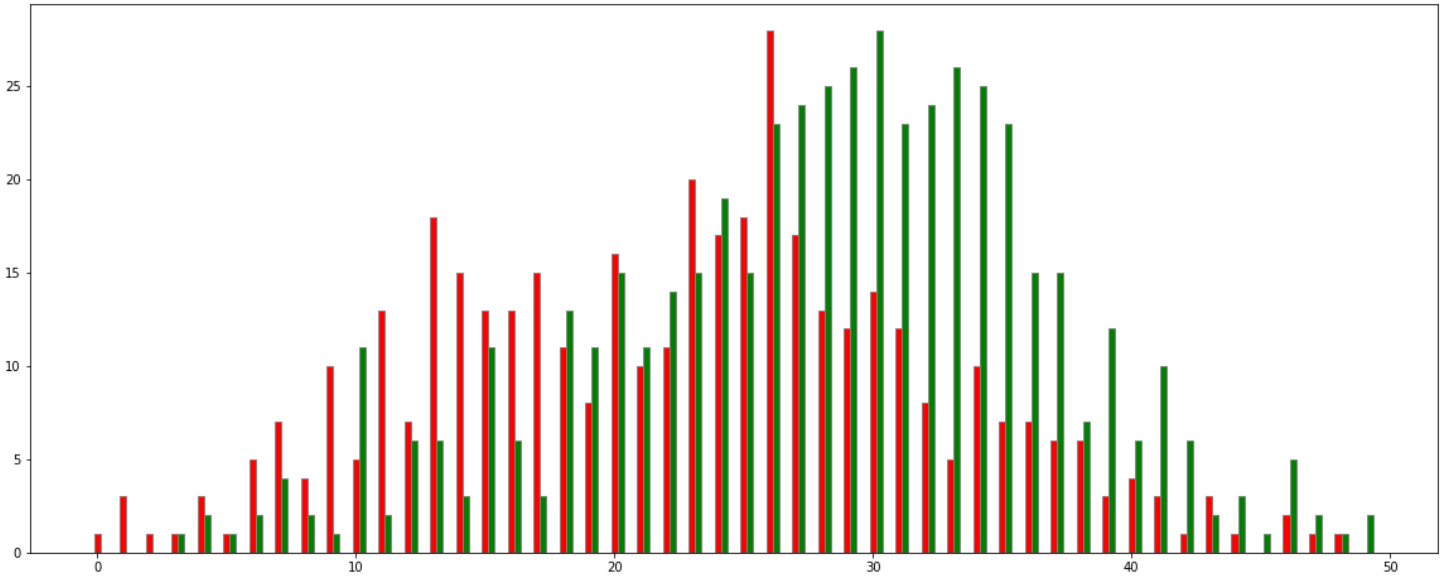
Feature:exercise angina



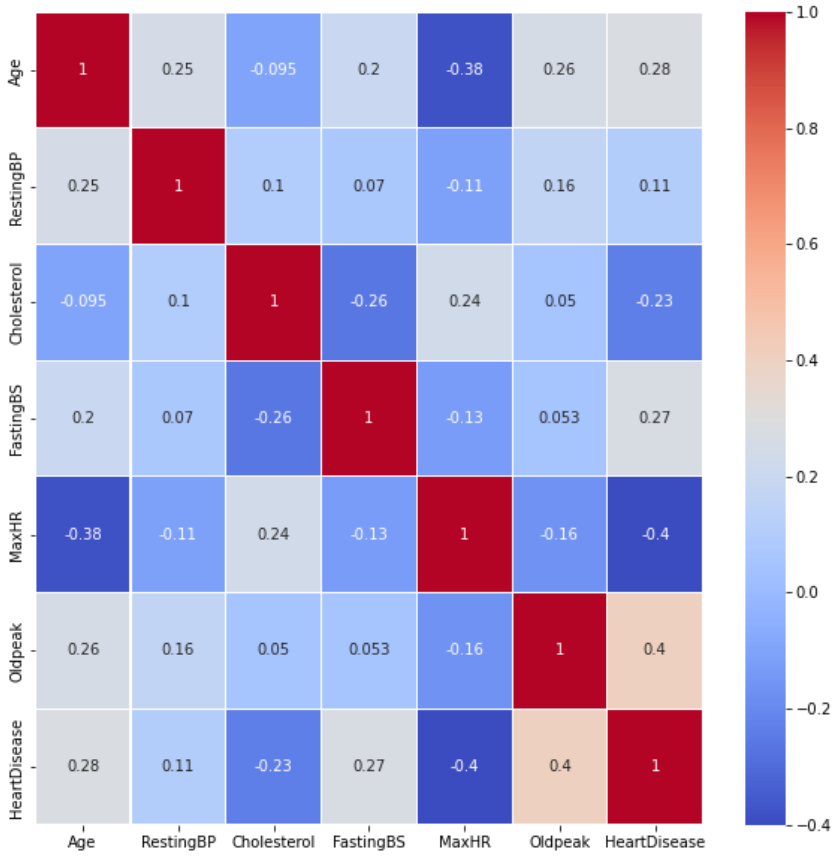
Feature:ST_slope



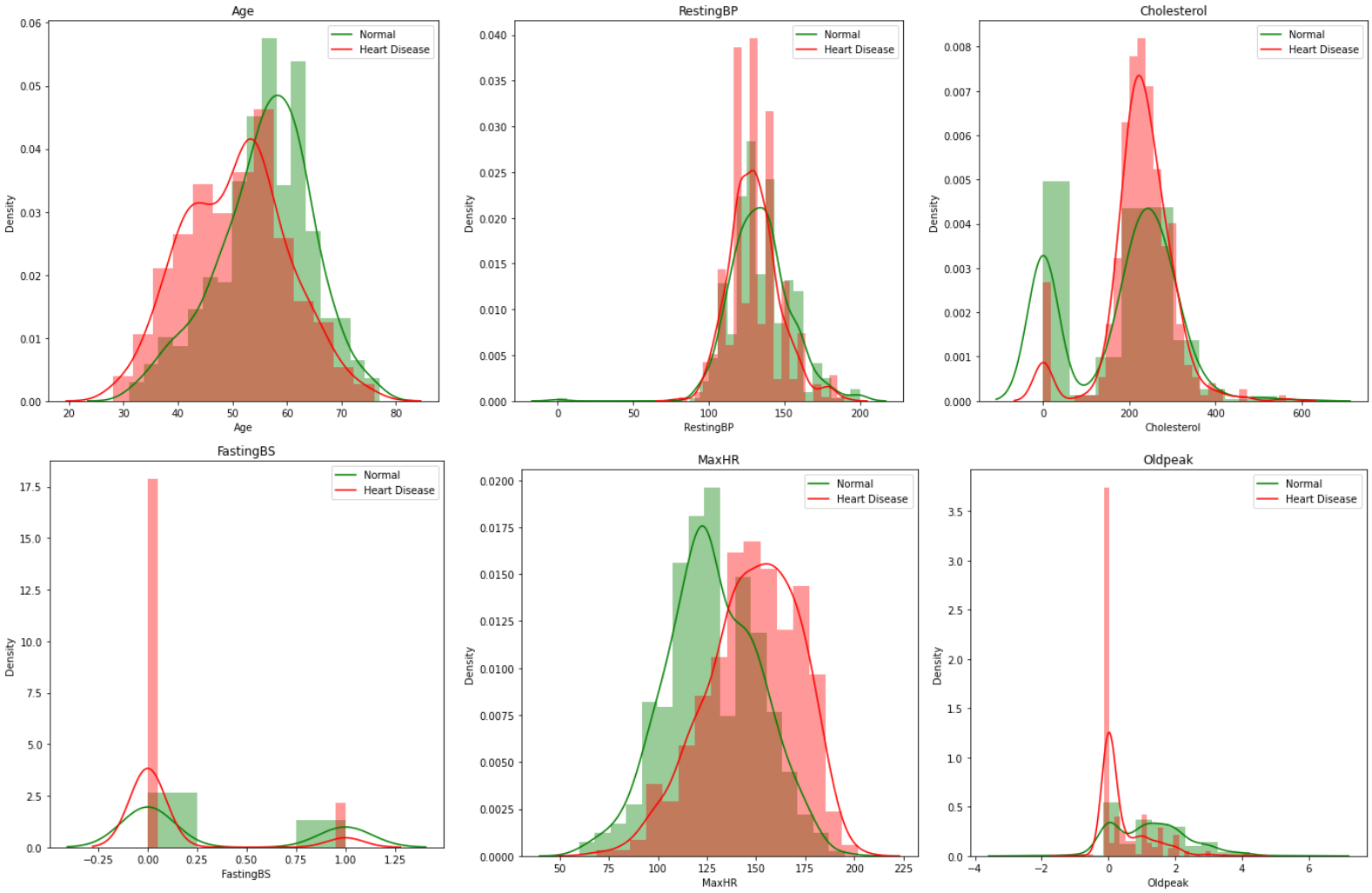
4. A bar graph to look at the age-wise frequency count for heart disease absent and heart disease absent was plotted



5. Checking correlation of dependent and independent features by plotting correlation matrix for all the features using heatmap



6. The features with feature values of type ‘object’ were ordinally encoded
   Features :
   ```
   'Sex','ChestPainType','RestingECG','ExerciseAngina','ST_Slope'
   ```
7. Density plots for numeric features were plotted
   Features:
   ```
   'Age','RestingBP','Cholesterol','FastingBS','MaxHR','Oldpeak'
   ```
   Plots:



8. The data was splitted to training and testing data using the sklearn library keeping test ratio 0.3.

```
len(x_train)
642
len(y_train)
642
len(x_test)
276
len(y_test)
276
```

CLASSIFICATION ON DATASET

Implementation of classification algorithms:

A. The goal in this step is to develop a benchmark model that serves us as a baseline, upon which we will measure the performance of a better and more tuned algorithm.
B. We are using different Classification Techniques and comparing them to see which algorithm is giving better performance then other and see how our model is predicting.
C. We initially form the following 8 ML algorithms using the sklearn library we have for them.
    a. **Decision tree classifier:**The decision tree classifier (Pang-Ning et al., 2006) creates the classification model by building a decision tree. Each node in the tree specifies a test on an attribute, each branch descending from that node corresponds to one of the possible values for that attribute.
    b. **K-Neighbors:**The k-nearest neighbors (KNN) algorithm is a simple, supervised machine learning algorithm that can be used to solve both classification and regression problems**.**
    c. **Gradient boosting classifier:**Gradient boosting classifiers are a group of machine learning algorithms that combine many weak learning models together to create a strong predictive model.
    d. **Gaussian Naive Baye**s:Gaussian Naive Bayes supports continuous valued features and models each as conforming to a Gaussian (normal) distribution. An approach to create a simple model is to assume that the data is described by a Gaussian distribution with no co-variance (independent dimensions) between dimensions.
    e. **Logistic Regression**:Logistic regression is a process of modeling the probability of a discrete outcome given an input variable
    f. **Multi-layer classifier:**Multi-layer Perceptron (MLP) is a supervised learning algorithm that learns a function f (·) : R m → R o by training on a dataset, where is the number of dimensions for input and is the number of dimensions for output.
    g. **K-means**:KMeans is a clustering algorithm which divides observations into k clusters. Since we can dictate the amount of clusters, it can be easily used in classification where we divide data into clusters which can be equal to or more than the number of classes.
    h. **Random forest Classifier**: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
D. Following are the accuracy for all the 8 models where the models were fitted on the training data(x_train,y_train) and the same model was used to predict testing data(x_test) and the predictions were compared with the original y_test:

Models	accuracy_Score
Decision Tree Classifier	77.898551
KNeighbors	70.652174
GradientBoosting	88.405797
Gaussian Naivebayes	85.144928
Logistic Regression	86.594203
Multilayer Perceptron	86.594203
K-means	54.710145
Random forest regressor	88.768116

Hyperparameter tuning:
Hyperparameter tuning is choosing a set of optimal hyperparameters for a learning algorithm. A hyperparameter is a model argument whose value is set before the learning process begins. Hyperparameter tuning is the key to machine learning algorithms.Hyperparameter tuning is an essential part of controlling the behavior of a machine learning model. If we don't correctly tune our hyperparameters, our estimated model parameters produce suboptimal results, as they don't minimize the loss function. This means our model makes more errors. Hence, hyperparameter tuning results in increase in accuracy.
If after hyperparameter tuning the accuracy doesn't increase or reduce this shows that the initial model was overfitting the datset.

A. **Decision tree classifier** - The parameters max_depth , min_samples_split were varied and the best parameters were selected from the graph of testing-training errors.
B. **K-Neighbors** - best parameters were selected using gridsearchCV.
C. **Gradient boosting classifier**- best parameters were selected using gridsearchCV.
D. **Gaussian Naive Bayes**- best parameters were selected using gridsearchCV.
E. **Logistic Regression**- best parameters were selected using gridsearchCV.
F. **Multi-layer Classifier**- best parameters were selected using gridsearchCV.
G. **K-means** - The parameter init was varied and the best parameter was selected from the graph of testing-training errors.
H. **Random forest Classifier**- The parameters max_depth,min_samples_split,min_sample_leaf were varied and the best parameters were selected from the graph of testing-training errors.

EVALUATION OF MODELS

The models implemented were evaluated using techniques like - Classification report : precision , recall , f1 score and support ,ROC plots,accuracy score,bar plots.

Models	Final_Score	Initial_Score
Decision Tree Classifier	80.434783	77.898551
KNeighbors	74.275362	70.652174
GradientBoosting	87.681159	88.405797
Gaussian Naivebayes	85.507246	85.144928
Logistic Regression	85.144928	86.594203
Multilayer Perceptron	81.521739	86.594203
K-means	54.710145	54.710145
Random forest regressor	86.956522	88.768116

Classification reports:

DECISION TREE CLASSIFIER

	precision	recall	f1-score	support
0	0.78	0.80	0.79	127
1	0.82	0.81	0.82	149

K-NEIGHBORS

	precision	recall	f1-score	support
0	0.73	0.70	0.71	127
1	0.75	0.78	0.77	149

GRADIENT BOOSTING CLASSIFIER

	precision	recall	f1-score	support
0	0.89	0.83	0.86	127
1	0.87	0.91	0.89	149

GAUSSIAN NAIVE BAYES

	precision	recall	f1-score	support
0	0.83	0.86	0.84	127
1	0.88	0.85	0.86	149

LOGISTIC REGRESSION

	precision	recall	f1-score	support
0	0.84	0.84	0.84	127
1	0.86	0.86	0.86	149

MULTI-LAYER CLASSIFIER

	precision	recall	f1-score	support
0	0.76	0.87	0.81	127
1	0.88	0.77	0.82	149

K-MEANS

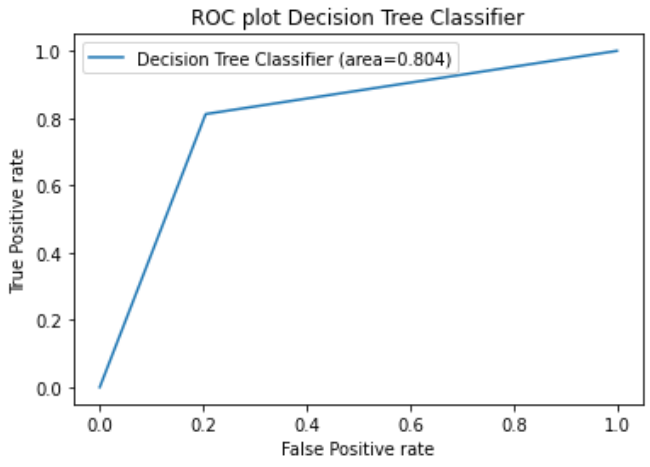
	precision	recall	f1-score	support
0	0.50	0.94	0.66	127
1	0.80	0.21	0.34	149

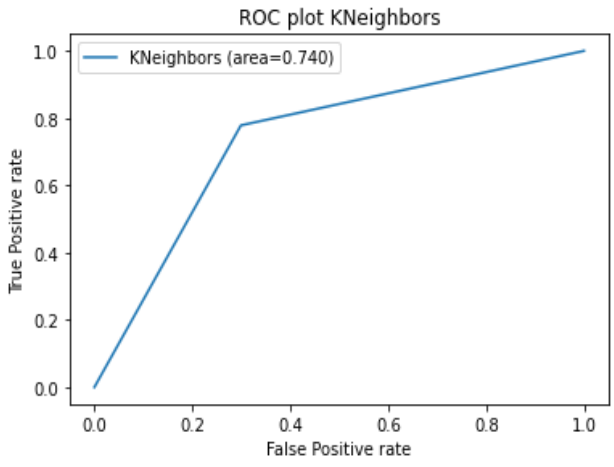
RANDOM FOREST CLASSIFIER

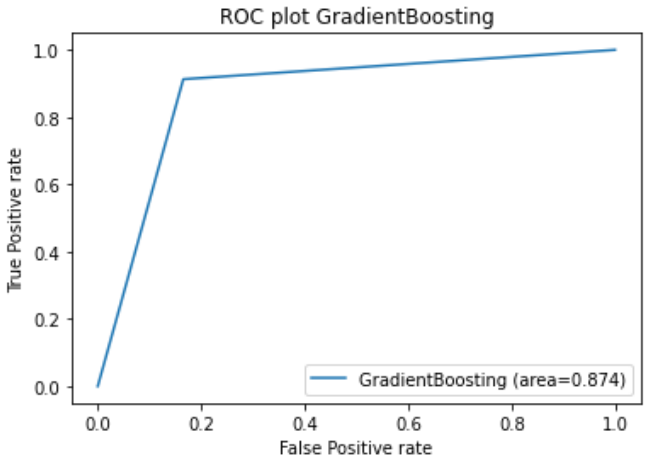
	precision	recall	f1-score	support
0	0.89	0.82	0.85	127
1	0.86	0.91	0.88	149

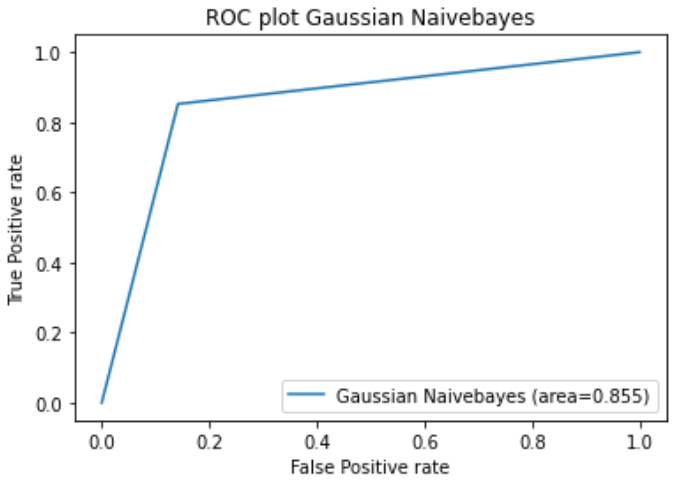
ROC curves

An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters: True Positive Rate. False Positive Rate.

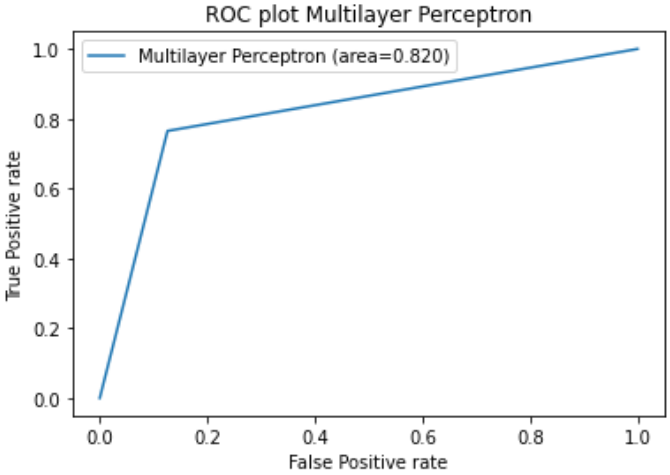


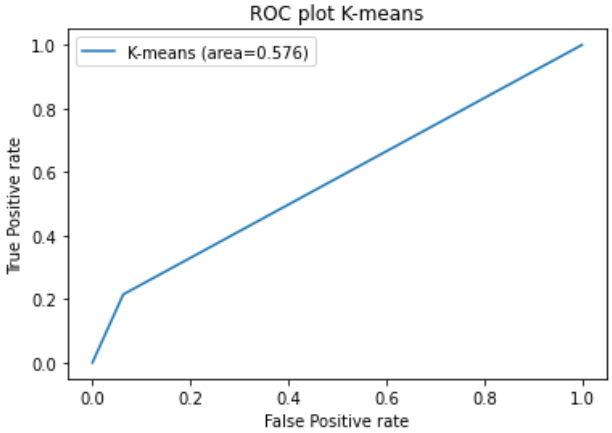


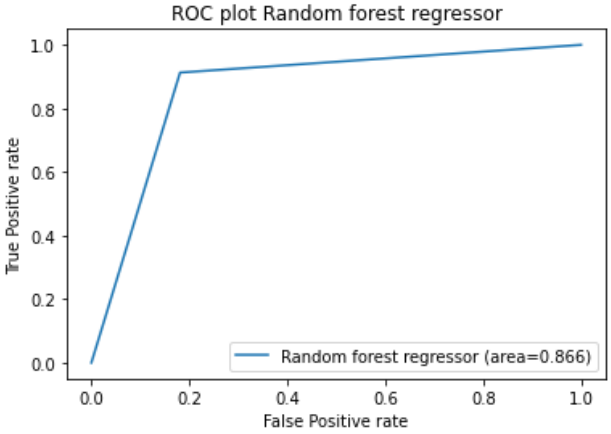

ROC plot GradientBoosting
True Positive rate
False Positive rate
GradientBoosting (area=0.874)

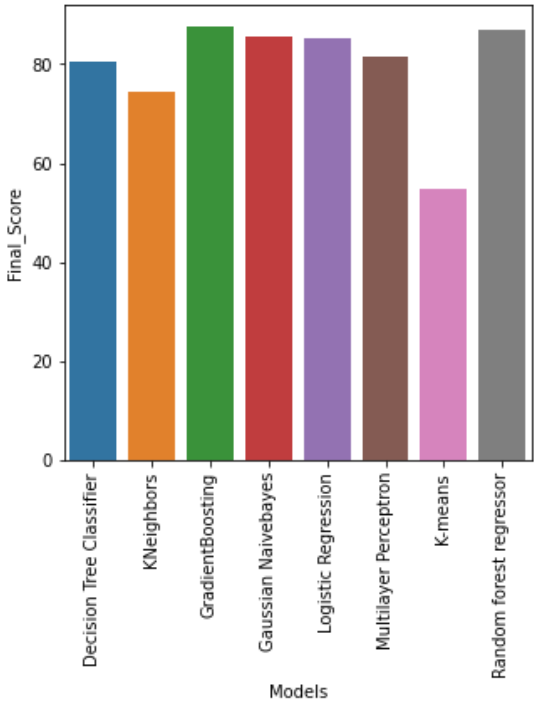

ROC plot Gaussian Naivebayes
True Positive rate
False Positive rate
Gaussian Naivebayes (area=0.855)


ROC plot Logistic Regression
True Positive rate
False Positive rate
Logistic Regression (area=0.851)


ROC plot Multilayer Perceptron
True Positive rate
False Positive rate
Multilayer Perceptron (area=0.820)

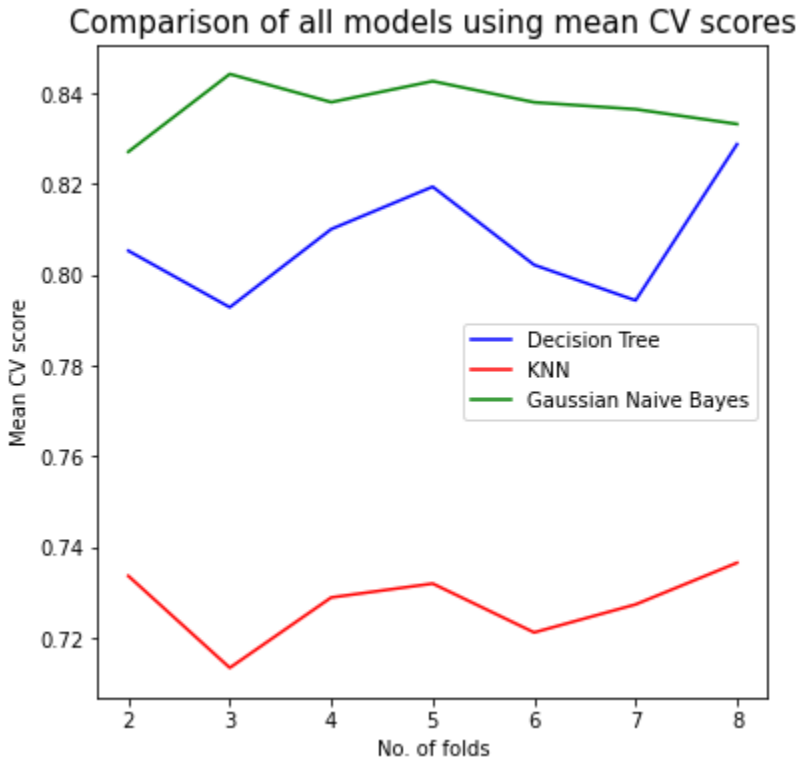

ROC plot K-means
True Positive rate
False Positive rate
K-means (area=0.576)


ROC plot Random forest regressor
True Positive rate
False Positive rate
Random forest regressor (area=0.866)



Cross-validation on the three best models:

- We perform the cross validation of our model to check if the model has any overfitting issue, by checking the ability of the model to make predictions on new data, using k-folds.
- We test the five fold cross validation for Decision tree classifier, K-neighbors and Gaussian Naive bayes below its visualization is shown.

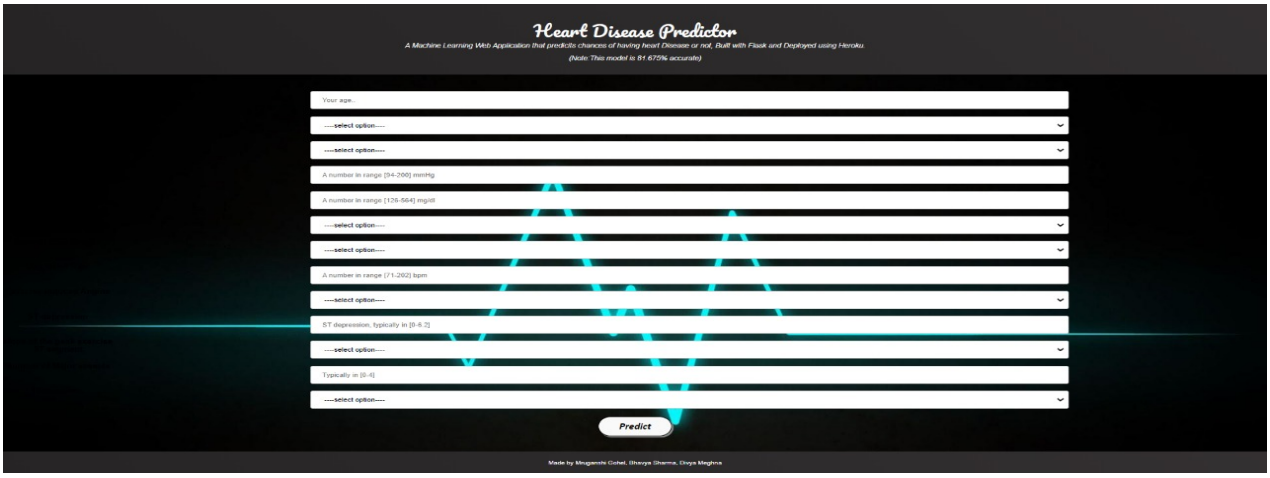


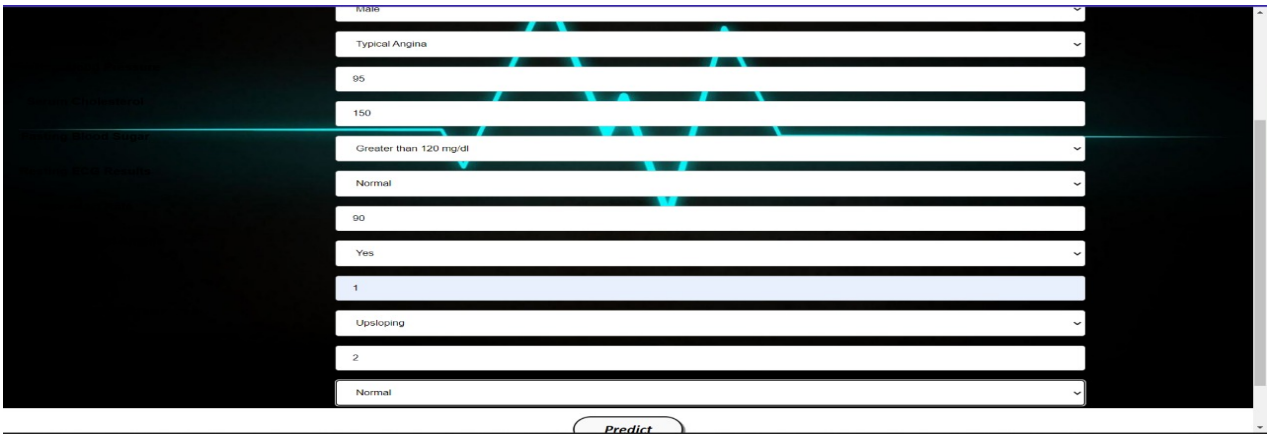
RESULTS AND ANALYSIS

Out of the eight models that we tried and implemented three models - Decision tree , K-neighbors and Gaussian Naive bayes performed well even after hyperparameter tuning. The parameters were
Decision tree : max_depth=9,min_samples_split=8
K-neighbors: 'metric': 'manhattan', 'n_neighbors': 16, 'weights': 'distance'
Gaussian Naive Bayes: 'var_smoothing': 5.336699231206313e-07
Before hyperparameter tuning, the accuracy for the algorithm Random Forest Regressor was the highest:88.768116. But , we do not consider this algorithm as its accuracy after hyperparameter tuning decreased and similarly the algorithms Gradient boosting classifier, Logistic Regression, Multi-layer classifier were also not chosen.
The best accuracy we obtain after hyperparameter tuning and from the chosen algorithms is for Gaussian Naive Bayes : **accuracy- 85.507246 . Gaussian Naive Bayes model also performs and gives the best results when cross-validation was performed on it.**

DEPLOYMENT

- We built a flask API endpoint that was hosted on a local webserver. The API endpoint takes in a request with a list of values and readings of various attributes needed to predict whether there are chances of heart disease or not for a person and returns an estimated price.
- Technologies used - python, Flask, HTML, CSS,heroku
- After entering details, heart failure prediction will be visible.
- Demo is shown below:





CONTRIBUTIONS

Bhavya Manish Sharma

- Data pre-processing, visualization and data exploratory analysis
- Models : Decision tree classifier,K-means,Random forest regressor
- Hyperparameter tuning : Decision tree classifier,Gaussian Naive Bayes,Random forest regressor,K-means
- Cross-validation: Decision Tree Classifier
- Report

Gohel Mruganshi Jatinbhai

- Models : K-neighbors,Gradient Boosting Classifier,Gaussian Naive Bayes
- Hyperparameter tuning : Logistic Regression, Multi-layer classifier,K-neighbors,Gradient Boosting Classifier
- Cross-validation: K-neighbors,Gaussian Naive Bayes
- Deployment

Mamidipalli Divya Meghana

- Models : Logistic Regression, Multi-layer classifier
- Report
- Deployment

REFERENCES

1. Sklearn library
2. Data Visualization using Python for Machine Learning and Data science|by sanat|towards data science
3. Cross validation in machine learning | geeks for geeks
4. https://scikit-learn.org/stable/modules/grid_search.html
5. https://analyticsindiamag.com/guide-to-hyperparameters-tuning-using-gridsearchcv-and-randomizedsearchcv/