Franklin Thomas (user id: frathom)

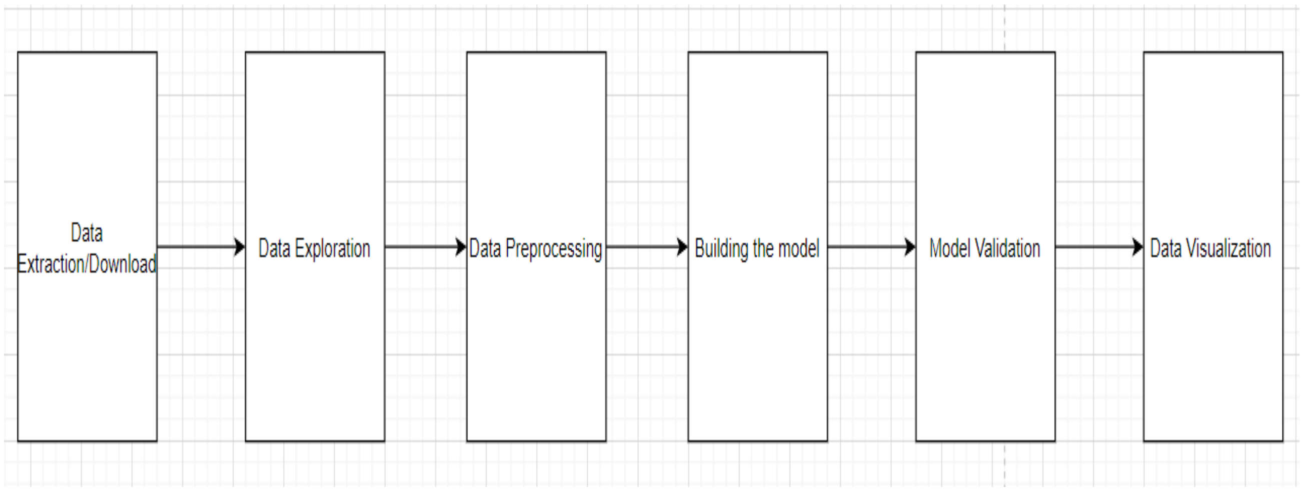
Introduction

The dataset I chose for the project is the 'Framingham Heart study' dataset which consists of data collected from the residents of Framingham, Massachusetts as a part of a cardiovascular study. The target attribute of the dataset is whether a particular patient has a 10year risk of coronary heart disease (CHD) (1 indicates the patient does have a ten year risk of coronary heart disease and 0 indicates the patient does not).Through this project I was able to apply the knowledge and skills I gained from the course. I was able to develop a data pipeline, clean the data(using python libraries), use pyspark and various other python tools/libraries for data understanding/exploration, develop predictive analysis models, visualize data through visualization tools(Tableau) etc.

Background

According to the CDC (Centers for Disease Control and Prevention) heart disease is the leading cause of death for men, women and people of most racial and ethnic groups in the united states [1], one person dies every 37 seconds in the United States from cardiovascular disease [1], in the United States someone has a heart attack every 40 seconds [1], about 1 in 5 heart attacks are silent(person's not aware but the damage is done) [1], Coronary heart disease is the most common type of heart disease, killing 365,914 people in 2017 [1]. I took up this project as I was really inspired by the idea of a model which could successfully predict whether a person has a 10 year risk of coronary heart disease. The early detection of such a risk in a person can help in making decisions on lifestyle changes for example change of diet, quit habits that may increase the risk and so on.

Methodology



Data Pipeline

Franklin Thomas (user id: frathom)

a. Data Download:

The data was obtained from Kaggle. The dataset is the 'Framingham Heart Study' dataset and it consists of 16 attributes and over 4000 records. The attributes are:

- Male : Whether the person is male or female (0-female, 1-male)
- Age : Age of the person
- Education : The education level of the person(value of 1 indicates some high school education, 2- High school or GED, 3- Some college or vocational school, 4- college)
- Current Smoker : Whether the person is currently a smoker or not (1- yes, 0-no)
- Cigs Per Day : The number of cigarettes smoked per day(estimated average)
- BPMeds : Whether the person was on blood pressure medication (0 - no, 1 – yes)
- Prevalent Stroke : Whether the person previously had a stroke (0 - no, 1 - yes)
- Prevalent Hyp : Whether the person was hypertensive (0 - no, 1 - yes)
- Diabetes : Whether the person had diabetes (0 - no, 1 - yes)
- Sys BP : Person's systolic blood pressure
- Tot Chol : Person's total cholesterol level
- Dia BP : Person's diastolic blood pressure
- BMI : Person's Body Mass Index
- Heart Rate : Person's heart rate
- Glucose : Person's glucose level
- TenYearCHD : 10 year risk of coronary heart disease (CHD) of a person (0 - no, 1 - yes)

b. Data Exploration(EDA):

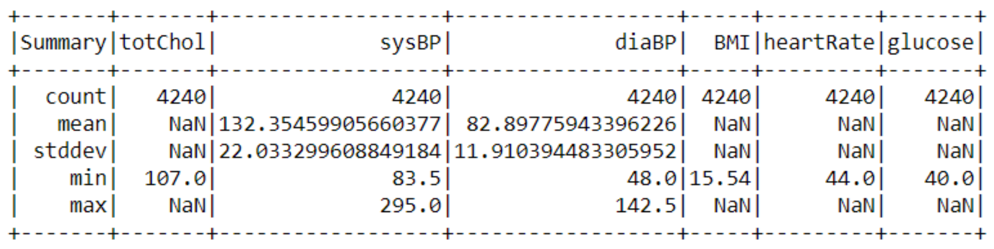
After downloading the data, exploratory analysis was done to observe the following :- how the data looked like, if there are any missing values present, if any attribute is correlated to another attribute, The stats of some attributes(for example mean, variance, std deviation etc), how many of the 4000+ records in the target variable were 1s and how many were 0s, how many of the males have ten year risk of CHD and how many females have the risk, how does age affect your risk of CHD etc.

I used python and its libraries for the project. For exploratory data analysis I used the following libraries:

- Pandas
- Pyspark
- Seaborn
- Matplotlib

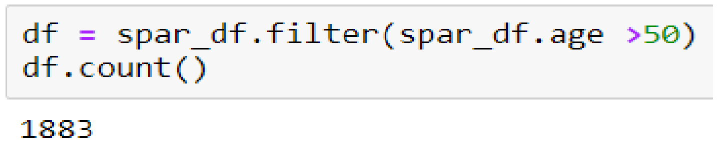
A few observations made from the data are:

Franklin Thomas (user id: frathom)



```
+-------+-------+------------------+------------------+-----+---------+-------+
|Summary|totChol|             sysBP|             diaBP|  BMI|heartRate|glucose|
+-------+-------+------------------+------------------+-----+---------+-------+
|  count|   4240|              4240|              4240| 4240|     4240|   4240|
|   mean|    NaN|132.35459905660377| 82.89775943396226|  NaN|      NaN|    NaN|
| stddev|    NaN|22.033299608849184|11.910394483305952|  NaN|      NaN|    NaN|
|    min|  107.0|              83.5|              48.0|15.54|     44.0|   40.0|
|    max|    NaN|             295.0|             142.5|  NaN|      NaN|    NaN|
+-------+-------+------------------+------------------+-----+---------+-------+
```

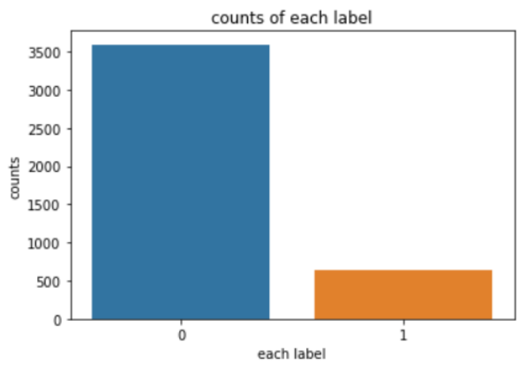
1. **Stats of some attributes before preprocessing using pyspark**



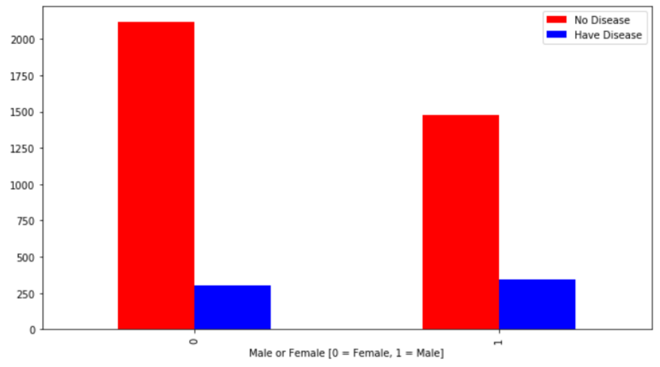
```
df = spar_df.filter(spar_df.age >50)
df.count()
```

```
1883
```

2. Pyspark command to see how many people above age 50 are present in the data

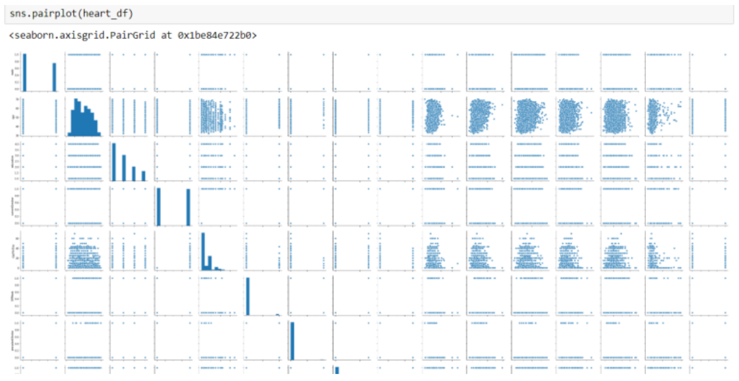


3. The distribution of the target variable in the dataset



4. Distribution of males and females having 10 year risk of CHD

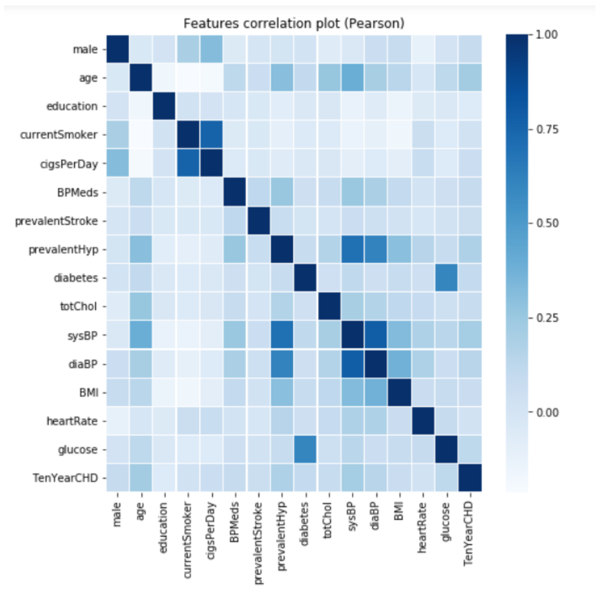
Franklin Thomas (user id: frathom)



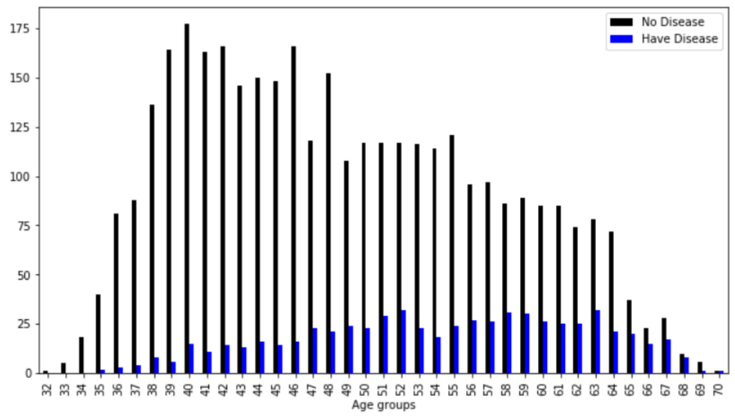
```
sns.pairplot(heart_df)
```

```
<seaborn.axisgrid.PairGrid at 0x1be84e722b0>
```

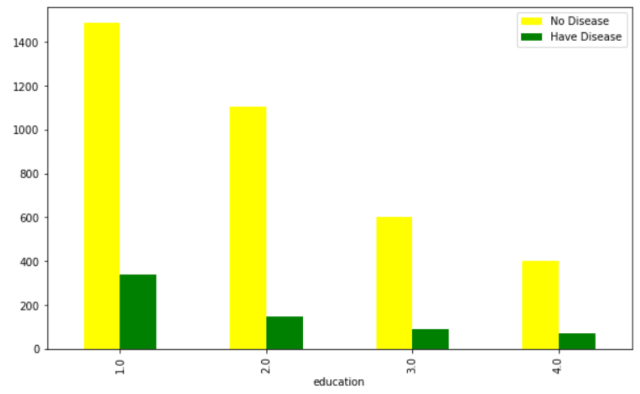
5. Pairplot (every attributed plotted against each other) to check for correlation



6. Feature correlation plot



7. Distribution of 10 year risk of CHD for different age groups



8. Distribution of 10 year risk of CHD for different education group

c. Data Preprocessing:

The data consisted of null values. The distribution of null values in each attribute is shown as below:

```
The number of missing values per column are:
male                 0
age                  0
education          105
currentSmoker        0
cigsPerDay          29
BPMeds              53
prevalentStroke      0
prevalentHyp         0
diabetes             0
totChol             50
sysBP                0
diaBP                0
BMI                 19
heartRate            1
glucose            388
TenYearCHD           0
dtype: int64
```

The null values in each column/attribute were imputed with their respective mode.

The stats of some columns after preprocessing is shown below:

```
+-------+------------------+------------------+------------------+------------------+------------------+------------------+
|Summary|           totChol|             sysBP|             diaBP|               BMI|         heartRate|           glucose|
+-------+------------------+------------------+------------------+------------------+------------------+------------------+
|  count|              4240|              4240|              4240|              4240|              4240|              4240|
|   mean|236.73844339622642|132.35459905660377| 82.89775943396226|25.784620283018874| 75.87877358490566| 81.32641509433962|
| stddev|44.328953627206694|22.033299608849184|11.910394483305952| 4.077826102272104|12.023937060119845|22.919884887634762|
|    min|             107.0|              83.5|              48.0|             15.54|              44.0|              40.0|
|    max|             696.0|             295.0|             142.5|              56.8|             143.0|             394.0|
+-------+------------------+------------------+------------------+------------------+------------------+------------------+
```

As seen in the previous figure 3 the dataset is imbalanced i.e., the target variable (TenYearCHD) has unequal number of 0s and 1s (3596 0s and 644 1s). Imbalance in the datasets cause poor performance of models. I tried different methods to handle this(I dropped few attributes to check if the model’s accuracy would improve, I tried methods like up-sampling and down-sampling however none of these gave me desirable outputs). I finally tried SMOTE (Synthetic Minority Oversampling Technique) which synthesizes new examples from the minority class. It works by selecting examples that are close in the feature space, drawing a line between the examples in the feature space and drawing a new sample at a point along that line. This proved fruitful as the model's accuracy improved considerably.

d. Building The Model :

I implemented the following predictive analysis models:

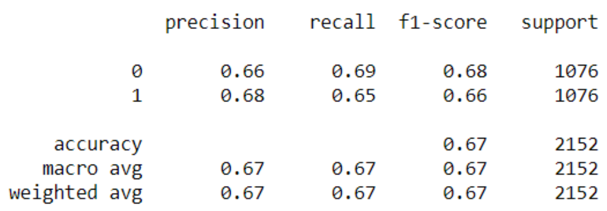
- Logistic Regression
- Naïve Bayes Classifier
- Decision Tree
- KNN (K nearest neighbors)
- SVM
- MLP (multilayer perceptron) Classifier
- Random forest
- Gradient Boosting Classifier
- Adaboost Classifier

Each one of these algorithms gave different accuracies and F1 scores and the results of these will be discussed in detail in the results section.

Results

When the models were validated it was observed that while the accuracies were quite high(logistic regression accuracy was around 85%), the F1 scores(It is the harmonic mean between precision and recall where precision is (true positive)/(true positive + false positive) and recall is (true positive)/(true positive + false negative)) were quite low. The best value of F1 score is 1 and the worst is 0. For my models I observed that the F1 scores were very low(some were as low as 0.10). The models correctly classified 0s(the majority class) and predicted 1s as 0s.Out of lets say 200 1s in the test data 180+ were classified as 0s. Using SMOTE(Synthetic Minority Oversampling Technique) I was able to correctly classify the 1s to a good extend .Model validation was done using F1 score(a very good metric which is used when datasets are imbalanced) and accuracy. I used 'classification report' from the sklearn library to show the classification metrics for each model (it shows us F1 score along with accuracy). The results are shown below:

Franklin Thomas (user id: frathom)

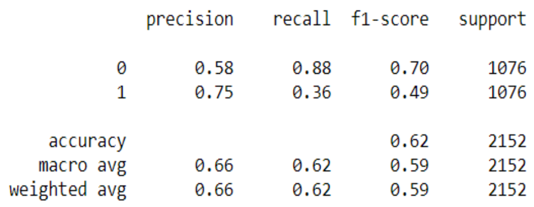


```
              precision    recall  f1-score   support

           0       0.66      0.69      0.68      1076
           1       0.68      0.65      0.66      1076

    accuracy                           0.67      2152
   macro avg       0.67      0.67      0.67      2152
weighted avg       0.67      0.67      0.67      2152
```

Logistic Regression classification report

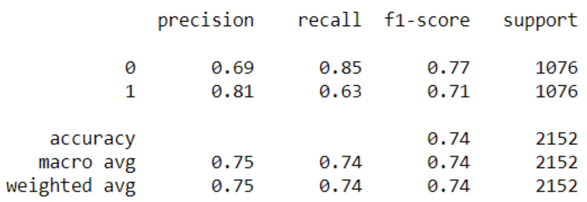


```
              precision    recall  f1-score   support

           0       0.58      0.88      0.70      1076
           1       0.75      0.36      0.49      1076

    accuracy                           0.62      2152
   macro avg       0.66      0.62      0.59      2152
weighted avg       0.66      0.62      0.59      2152
```

Naive Bayes classification report

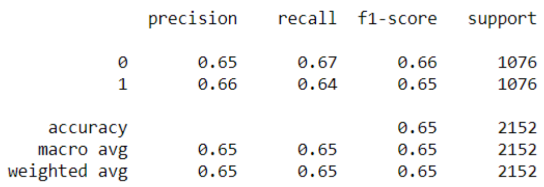


```
              precision    recall  f1-score   support

           0       0.69      0.85      0.77      1076
           1       0.81      0.63      0.71      1076

    accuracy                           0.74      2152
   macro avg       0.75      0.74      0.74      2152
weighted avg       0.75      0.74      0.74      2152
```

Decision Tree classification report

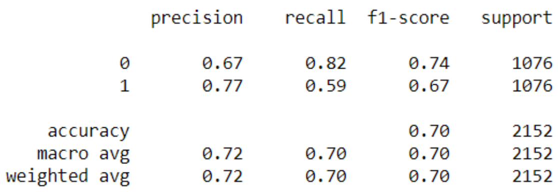


```
              precision    recall  f1-score   support

           0       0.65      0.67      0.66      1076
           1       0.66      0.64      0.65      1076

    accuracy                           0.65      2152
   macro avg       0.65      0.65      0.65      2152
weighted avg       0.65      0.65      0.65      2152
```

KNN classification report

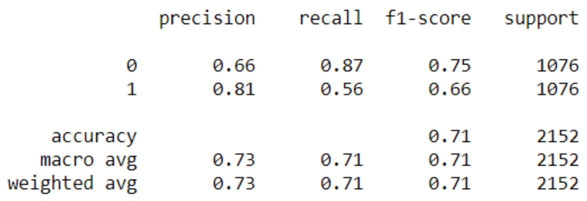


```
              precision    recall  f1-score   support

           0       0.67      0.82      0.74      1076
           1       0.77      0.59      0.67      1076

    accuracy                           0.70      2152
   macro avg       0.72      0.70      0.70      2152
weighted avg       0.72      0.70      0.70      2152
```

SVM classification report

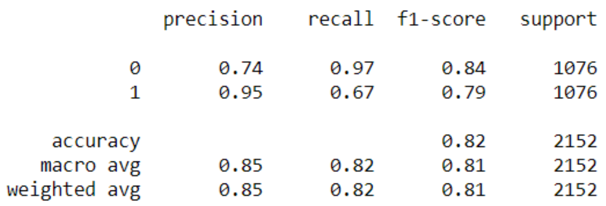


```
              precision    recall  f1-score   support

           0       0.66      0.87      0.75      1076
           1       0.81      0.56      0.66      1076

    accuracy                           0.71      2152
   macro avg       0.73      0.71      0.71      2152
weighted avg       0.73      0.71      0.71      2152
```

MLP Classifier classification report

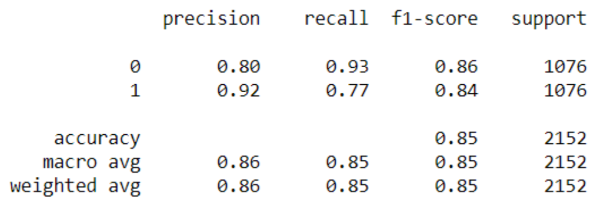


```
              precision    recall  f1-score   support

           0       0.74      0.97      0.84      1076
           1       0.95      0.67      0.79      1076

    accuracy                           0.82      2152
   macro avg       0.85      0.82      0.81      2152
weighted avg       0.85      0.82      0.81      2152
```

Random Forest classification report



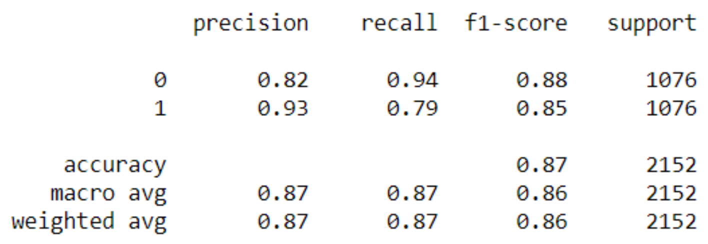
```
              precision    recall  f1-score   support

           0       0.80      0.93      0.86      1076
           1       0.92      0.77      0.84      1076

    accuracy                           0.85      2152
   macro avg       0.86      0.85      0.85      2152
weighted avg       0.86      0.85      0.85      2152
```

Adaboost Classifier classification report

The best model :-



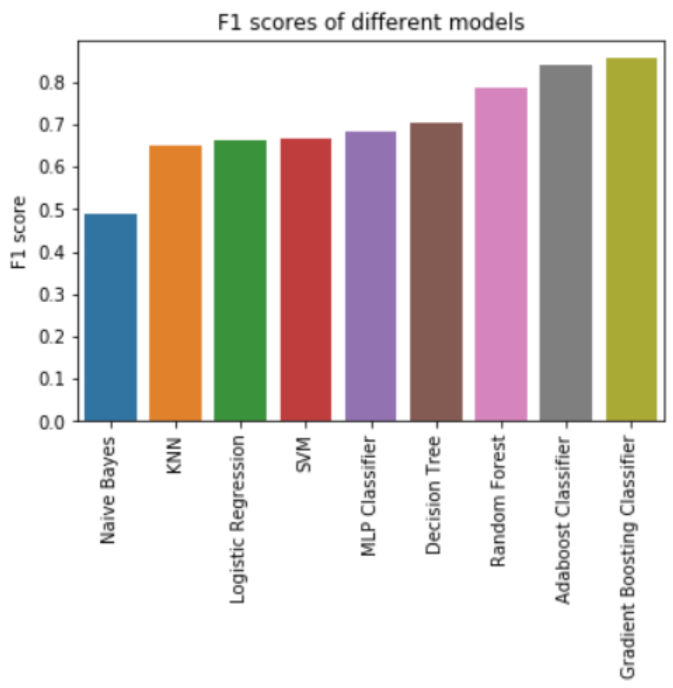
```
              precision    recall  f1-score   support

           0       0.82      0.94      0.88      1076
           1       0.93      0.79      0.85      1076

    accuracy                           0.87      2152
   macro avg       0.87      0.87      0.86      2152
weighted avg       0.87      0.87      0.86      2152
```

Gradient Boosting Classifier classification report

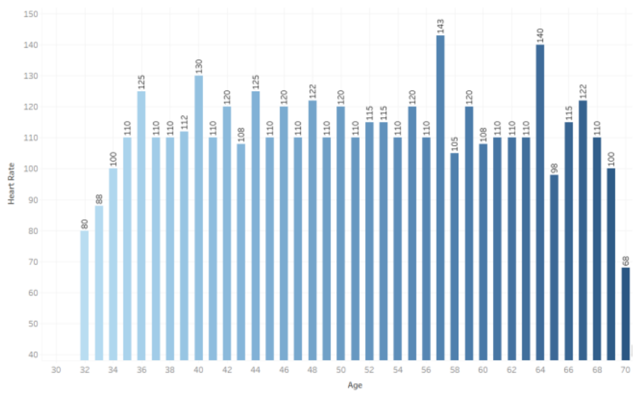
As seen from the classification reports gradient boosting algorithm performed better than the other models. It gave an accuracy of 87% and a F1 score of 0.86. Adaboost classifier performance was nearly as good however gradient boosting had a higher accuracy and F1 score for the same learning rate and number of estimators.



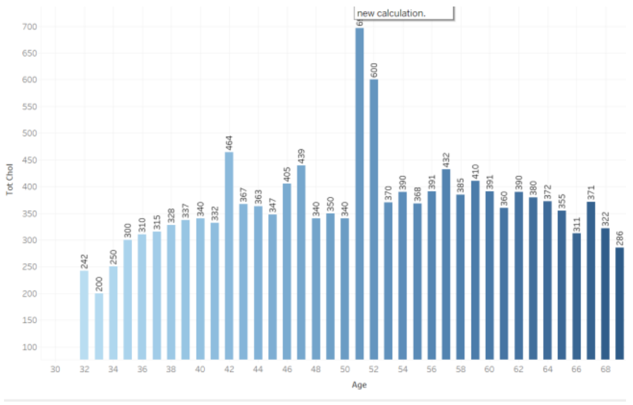
F1 Scores of the models created

Additional Visualizations Created :

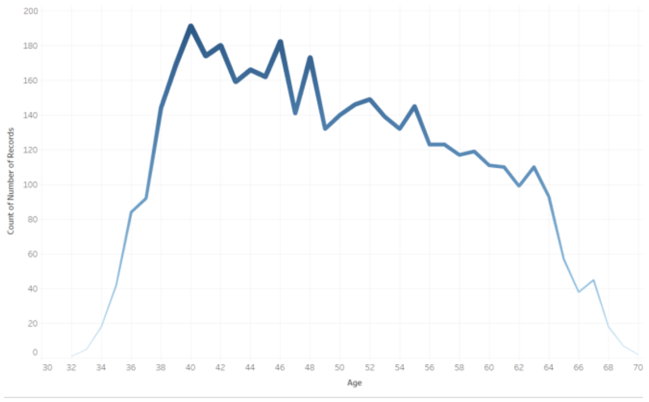
I created a few visualizations with Tableau to understand the data more. Through these visualizations I was able to visualize which age group has the highest heart rate, how the heart rate changes as age increases, the total cholesterol of different age groups, the age groups with the highest cholesterol, the distribution of number of records and their corresponding age groups(how many records of particular age group is present in the data).



Heart Rate vs Age



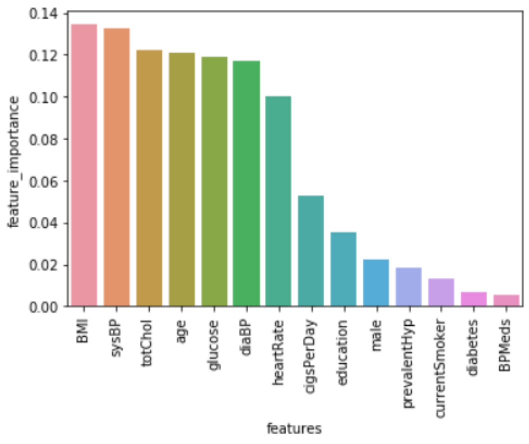
Tot Chol vs Age



Count of Number of Records vs Age

Challenges:

Improving the model accuracies and performance was the biggest challenge in this project. Various techniques were tried for example upsampling, downsampling etc. None of them worked as good as SMOTE. I also tried techniques like Recursive feature elimination to see the importance of each attribute and tried eliminating some of the attributes with low importance. I eliminated a few attributes and compared this accuracy to the SMOTE result and observed that SMOTE worked better and gave a much higher accuracy and F1 scores for each model. The feature importances obtained from recursive feature elimination is shown below:



As seen from the figure features such as ‘cigsPerDay’, ‘education’, ‘male’, ‘prevalentHyp’, ‘currentSmoker’, ‘diabetes’, ‘BPMeds’ have low feature importance. These features were dropped to see how the models would fare but it was seen that the models again predicted most of the 1s(minority class) of the target variable as 0s(majority class).

Conclusion

As discussed earlier heart health prediction of a person is of utmost importance and can help save lives if detected early. Machine learning models can identify underlying patterns in data that humans cannot. High dimensionality patterns can be found by these models helping them achieve high accuracy of prediction. The best performing model in this project was the gradient boosting classifier with an accuracy of 87% and a F1 score of 0.86. There are however ways to improve its performance like for eample by playing around with the learning rate and number of estimators. Same goes for adaboost classifier which performed second best (very close in terms of performance to gradient boosting algorithm).

Through this project I was able to successfully apply the knowledge I gained from this course and was able to refine my models with this knowledge. The data pipeline that was developed can be improved further by having a loop back from the model validation to the data preprocessing i.e., we can check if the model performance would improve by trying out different imputation techniques eg imputing missing values with mean, minimum value or maximum value to check the performance. Overall I was pretty satisfied with the performance of my models on the test data. For future work I would love to explore some more machine learning models like catboostclassifier, even try some other libraries in python and the machine learning models present in them.

References

1. https://www.cdc.gov/heartdisease/facts.htm
2. Dataset : https://www.kaggle.com/amanajmera1/framingham-heart-study-dataset
3. https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/
4. https://towardsdatascience.com/data-preprocessing-in-python-b52b652e37d5
5. https://towardsdatascience.com/understanding-gradient-boosting-machines-9be756fe76ab