**Heart Attack Risk Prediction**

Name: Gonugunta Raga Swarupa

Matrix number: 2301804G

# Introduction

**Topic**:

Heart attack risk prediction predicts if a person will have heart attack or not. This topic focuses on developing a machine learning model to detect the heart attack risk. In the recent days, everyone is getting diagnosed with various diseases and their health is deteriorating. All this can lead to heart attacks. The goal is to build a robust model that can accurately identify heart attack risk. With a prediction model, we can always check and make sure we are safe and healthy. People who believe that they are in risk can also confirm and be assured. If the model predicts that they might be in risk, they can get it checked and identify the issue early to prevent any damages.

**Describe the choice of dataset:**

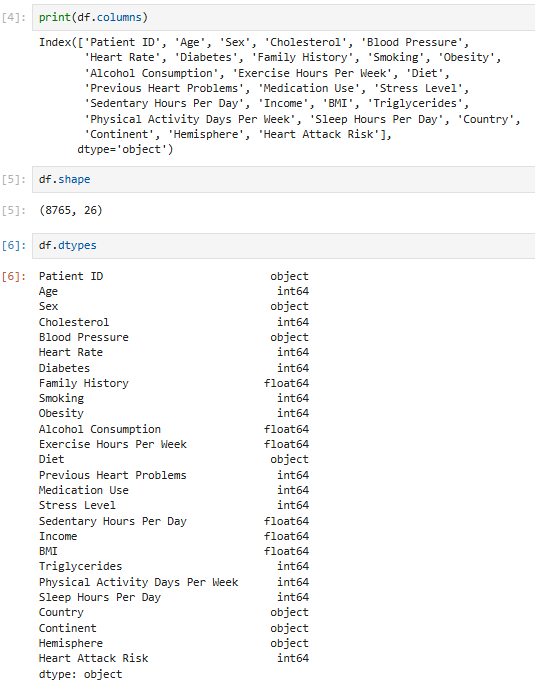
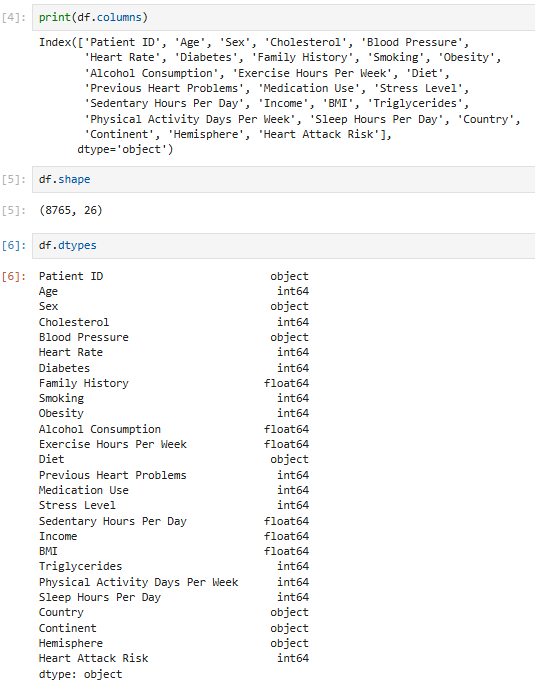
This dataset is from Kaggle and there are many features provided which is relevant to the heart attack risk. This dataset contains various health-related attributes and information (Patient ID, Age, Sex, Country, Continent, Hemisphere), health information (Cholesterol, Blood Pressure, Heart Rate, BMI, Triglycerides), lifestyle details (Smoking, Alcohol Consumption, Exercise Hours Per Week, Diet, Sedentary Hours Per Day, Physical Activity Days Per Week, Sleep Hours Per Day), medical history (Diabetes, Family History, Previous Heart Problems, Medication Use), and other factors (Income, Stress Level), all contributing to the assessment of Heart Attack Risk. It contains 8771 rows and 26 columns. The column that I am predicted would be the “Heart Attack Risk”.

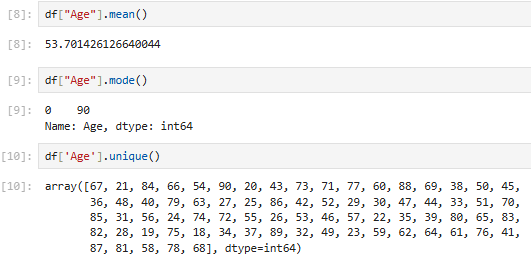
I chose this dataset because I want to create a useful classification predicting model to help people quickly and easily identify if they have a risk of having a heart attack. To begin with this project firstly I am going to get the data, explore and process, shuffle the data, split, set some model hyperparameters, train the model, evaluate, and make more changes if needed and use the model.

# Data exploration and preprocessing of data.

Firstly. it is important to understand the data. We need to know how many rows and columns it has, check if it has any missing values, understand the unique values for each column, find out the count of values in the columns, get to see the head & tail of the dataset and it is also good to find out the mean and mode for the columns.

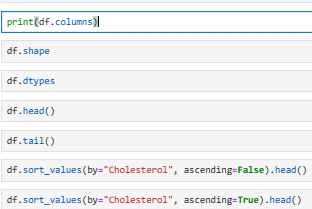
These codes allow me to better understand the data, it tells me the columns I have as well as the number of columns and rows. I also can see the data type for each column.

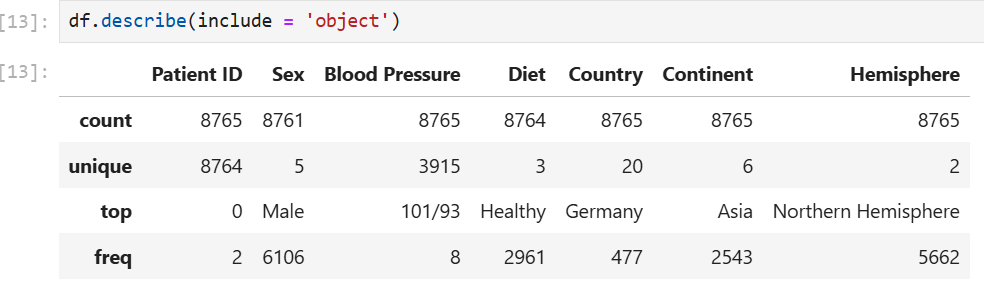


I can understand that the mean age is around 53 and the most occurring age is 90. I can also see all the unique values my column contains by using the .unique

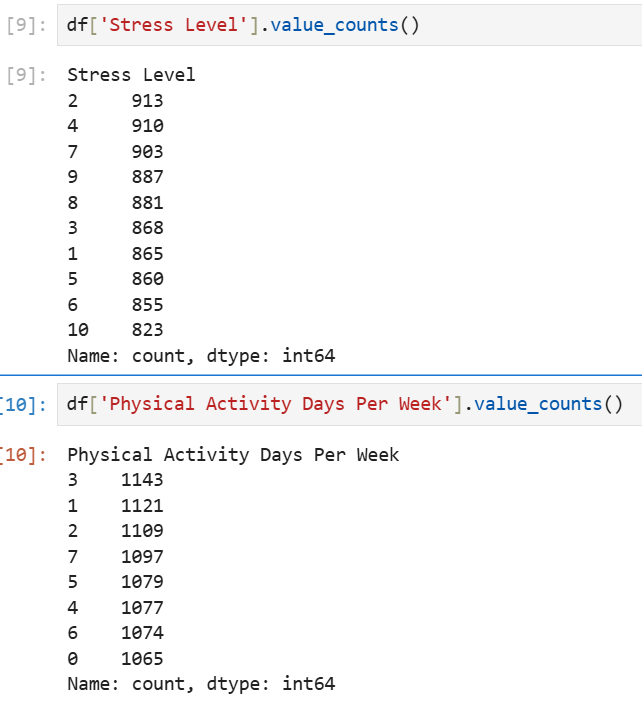
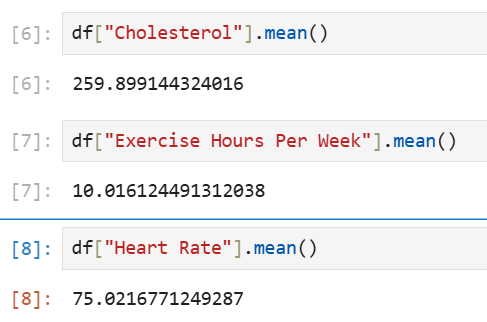
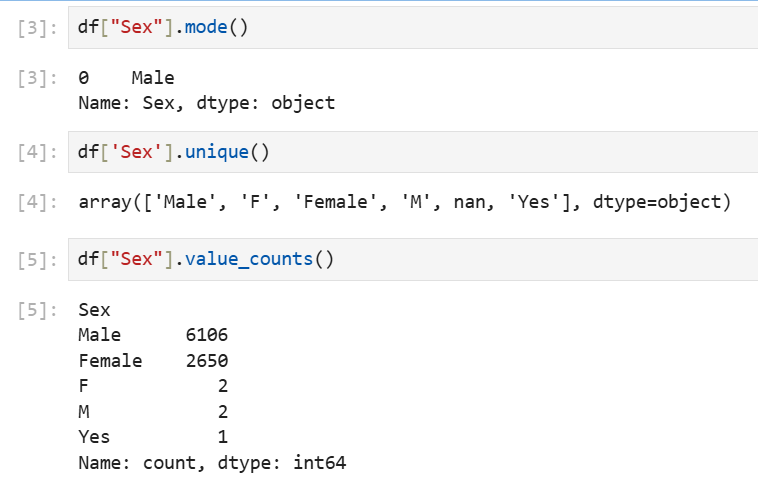




These codes here allow me to explore the data better before I use visualizations to understand it. I can find out any data inconsistencies and missing values that I need to take care of. I can also view the columns and slightly get an idea on what is needed and what might be removed later in the feature engineering. I can also view the column’s unique values and the counts for each unique value. That way, I can understand how distributed it is. I can also understand the mean and mode for numeric and categorical data and understand the counts.

Df.columns, df.shape, df.head, df.tail, df.describe, df.describe(include = ‘object’), Df.sort: These codes will generally let me know more about the dataset by returning the columns of the dataset, telling me how many rows and columns It has, allows me to also see the first 5 rows of the dataset as well as the last 5 of the dataset. Df.describe will let me view the statistics for the numerical columns by showing the mean, max, top, unique, frequency and many more details. We can also view details for the categorical data like count, unique, top and frequency. Df.sort will allow us to sort a column based on ascending or descending order that way, we can understand the highest and lowest values.

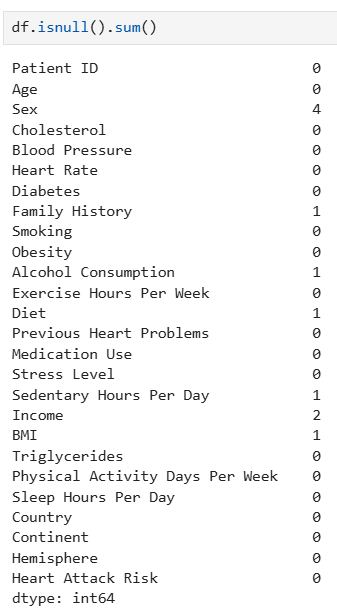
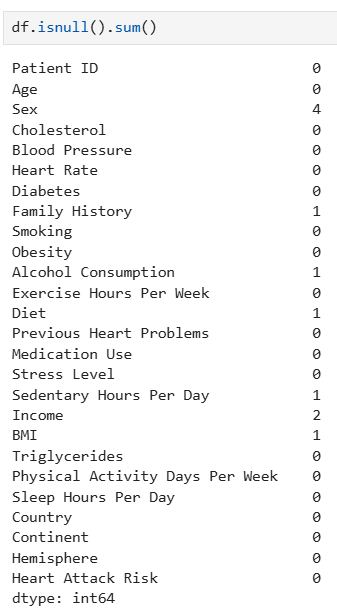


Value counts, unique, mean, mode: Value counts tell us how many of each unique value is present.

Df. duplicated().sum(): This tells us if any rows are duplicated. It is good to remove duplicated rows to ensure data consistency and to avoid overfitting. For my case, there are 0 duplicated rows.

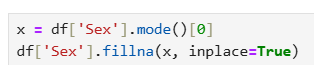


Upon exploring the dataset, I found out that the dataset was also not clean. There were some missing values, and some values are not appropriate. Eg. F, M, male, female, yes. I then replaced and removed some rows. I used the mean to replace some numeric values and used the mode to replace some values with categorical data.

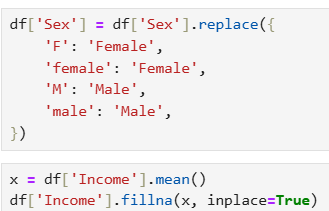


This shows that there are some null values in my rows. It also shows the exact number of null values for each column making it easier for me to remove and fix.

**Data cleaning:**



There were some missing values in the “Sex” column, so I found the mode and replaced it with it.

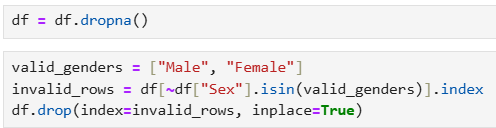


Some of the rows in the sex column is also not properly placed, there were variety of inputs, I have then replaced the rows with the proper value.

There were some missing values for the income column as well, since this is a numeric column, I have replaced it using the mean.

I then dropped the rows with the NA values to make the dataset cleaner.

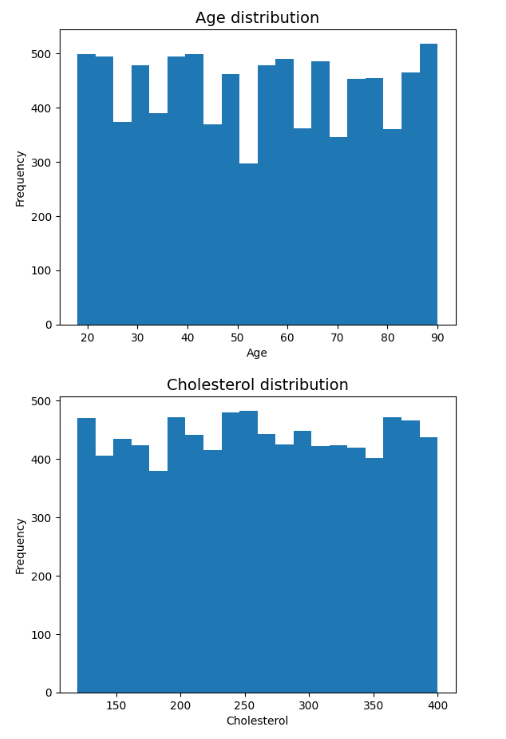
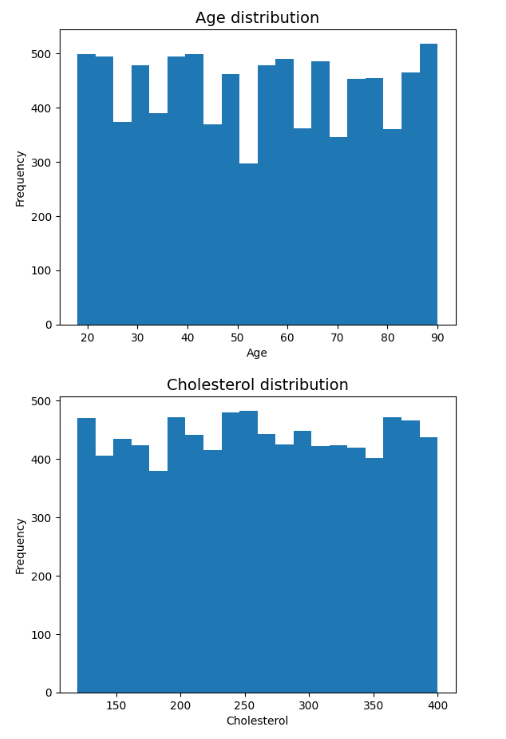
Then the sex column still had values which were not meant to be there like “Yes”. I then removed rows which contains anything other than Male or Female.

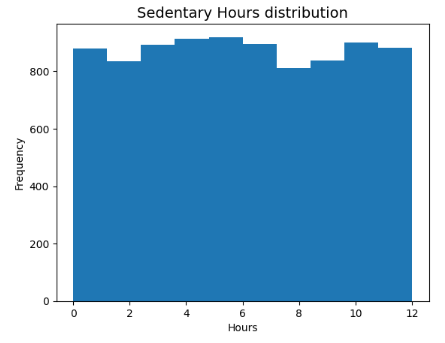
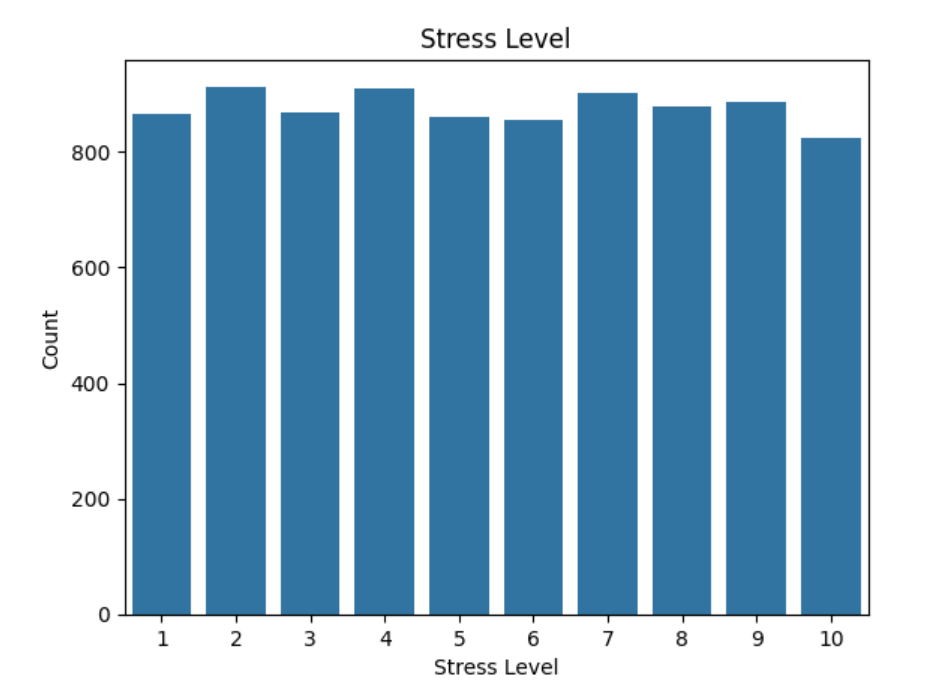
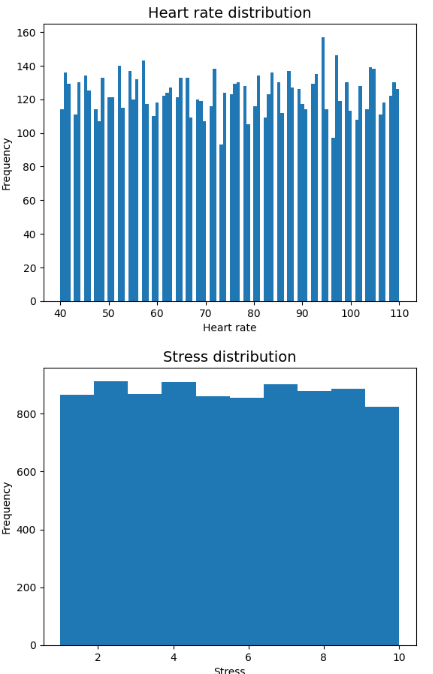


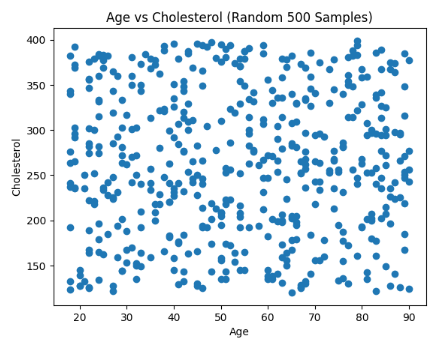
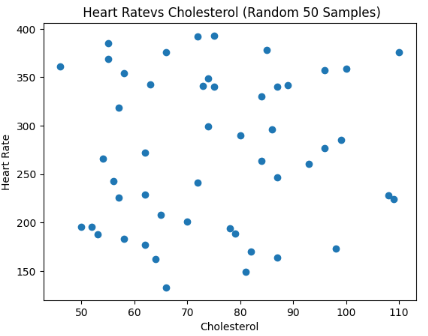
**Visualizations**:

I have used various charts to visualize my data example, Histogram, Scatter plot, Pie chart, Line plot, checking the linear Corr, bar charts, count plots, Box and whiskers, scatter matrix, Hexabin and violin plot.

I have plotted the histogram for some continuous variables to look at the distributions. For the cholesterol, generally most of the cholesterol distribution is distributed and is uniform. For the age, we can see that some plots are lower compared to others, age around 50 is the lowest and the highest would be the age 90. Age groups from 20, 40, 60, The highest heart rate is around 94 and the lowest is around 75. There are many different types of heart rate collected. The stress level chart indicates that the mode is 2 and the lowest is 10. There is no low count for any. The sedentary hours chart is well distributed and highest is at 5-6 hours and lowest is 8 hours.

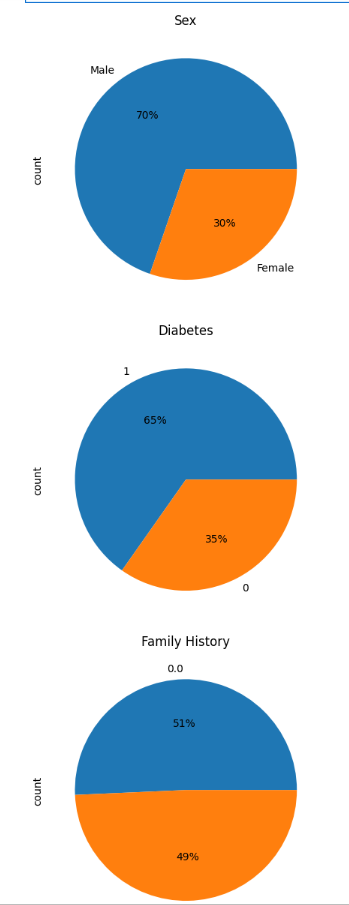
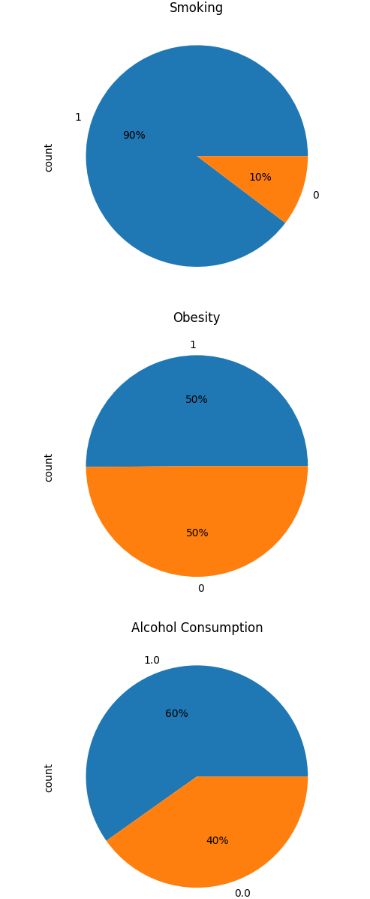
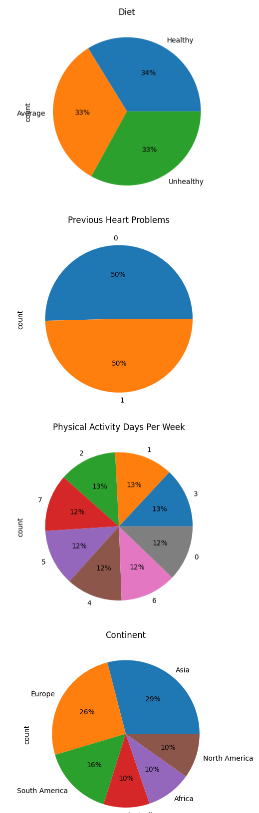


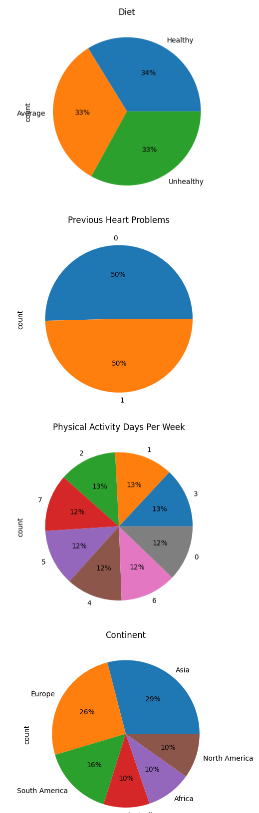
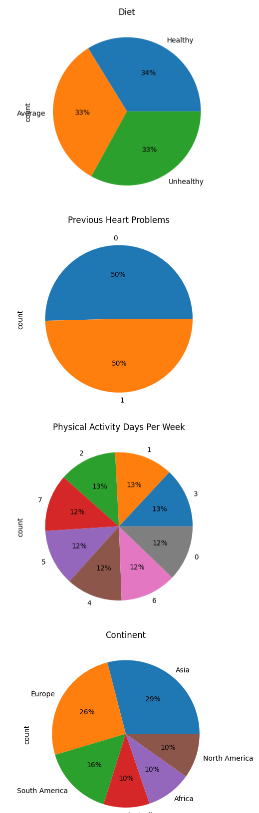
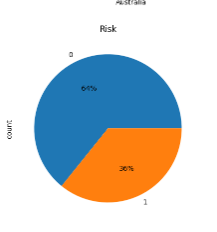


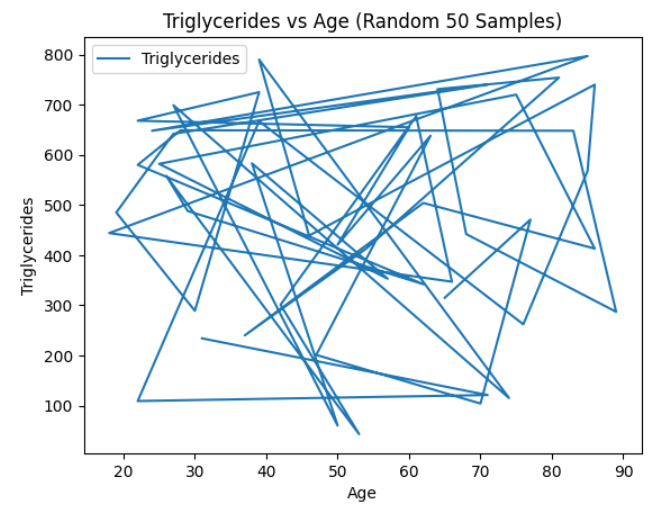
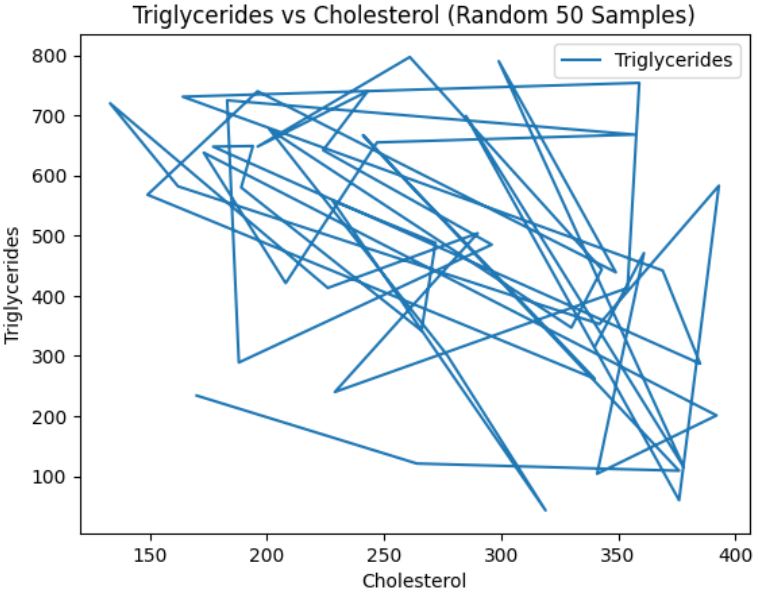
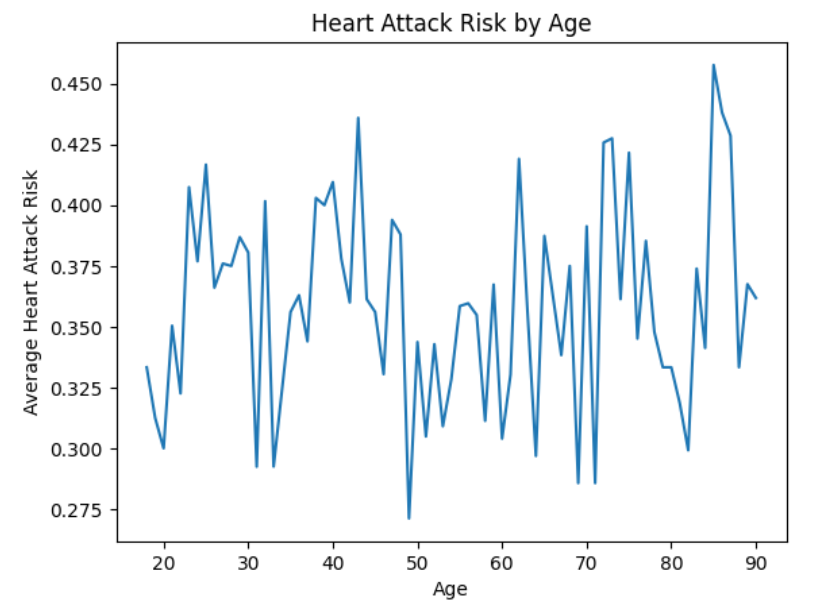
Although both the variables are numeric and continuous, I was not able to make an insightful scatter plot. There were too many samples and there was so relationship between the features. This happened to most of the columns. I also reduced the sample size to 500 and even 50 but I could not identify a relationship at all.

The distribution for the pie chart has more males than females. It also has more diabetic patients than people without diabetes. The percentage of patients who have had family members with a history of the heart attack is almost equal to that of people whose family members did not have diabetes but there is 2% more of the people whose family members did not have. 90% of the people are smokers. The distribution of people who are obese and people who are not obese is equal. 60% of these people consume alcohol. Healthy, unhealthy, and average diets almost have an equal distribution. There is just one more percent of people with a healthy diet.  This chart shows an even distribution of the  people who had heart problems before and people who didn't. The number of days with physical activity is spread out nearly evenly at percentages 12 and 13 only from 0 days a week to 7 days a week. The highest percentage is Asia at 29 percent while the lowest are Australia, Africa, and north America at 10 percent. Overall, when looking at the predicting column, there are more people with no risk compared to those with risk > 20%

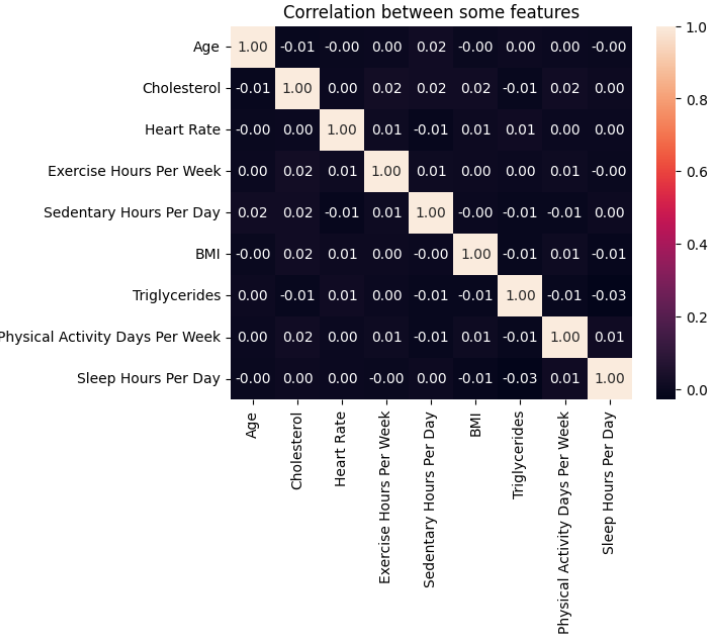
  

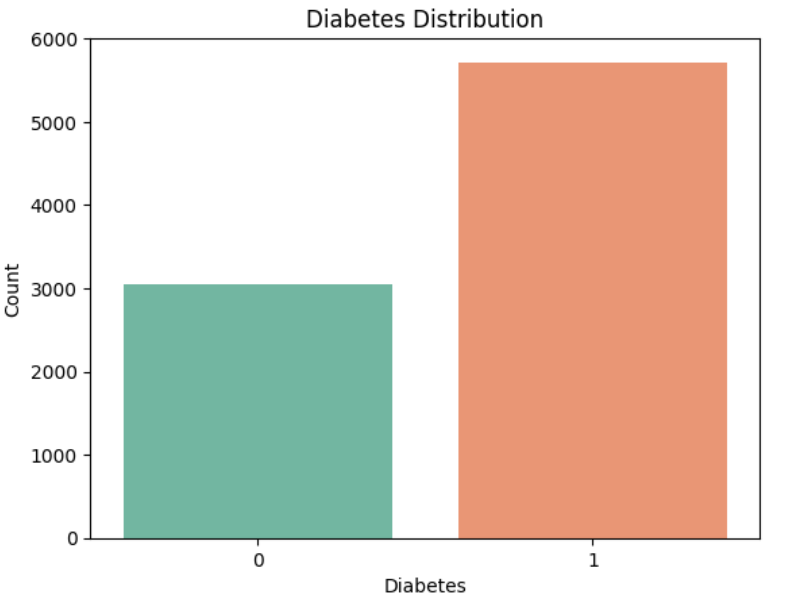
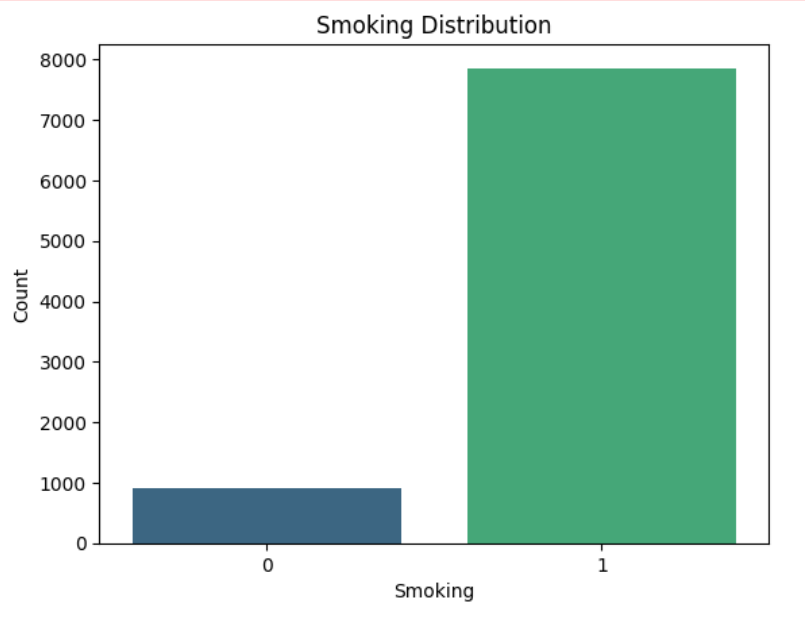
I have also created some line plots. The graphs are not insightful as no relation was observed. I have only taken 50 samples since using all was very messy.

It is observed that the ages above 80 have higher risks of heart attacks while the ages around 48 have the lowest but it is not consistent as the ones, I have mentioned are just the highest and lowest points.

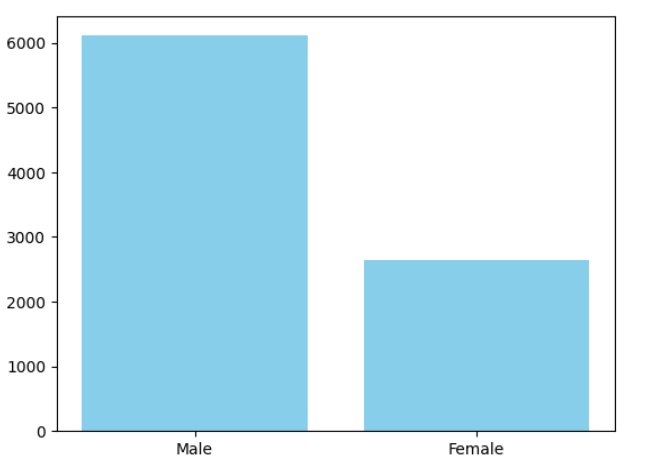
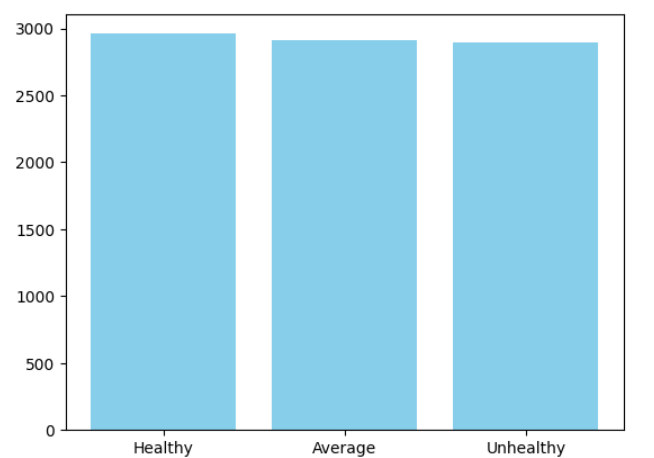
I have done linear correlation however, none of my features are correlated at all.



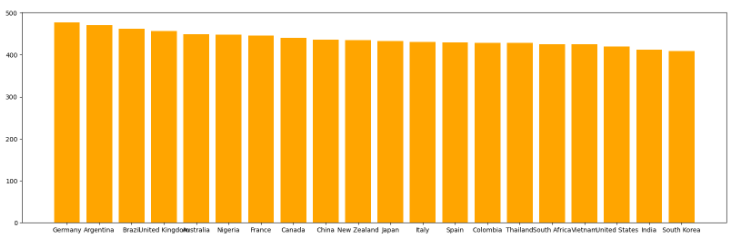
Diabetes: Most people have diabetes

Smoking: Most people are smokers, there are very little people who do not smoke.

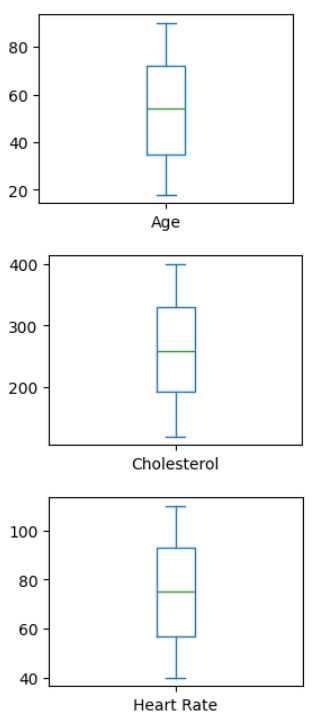
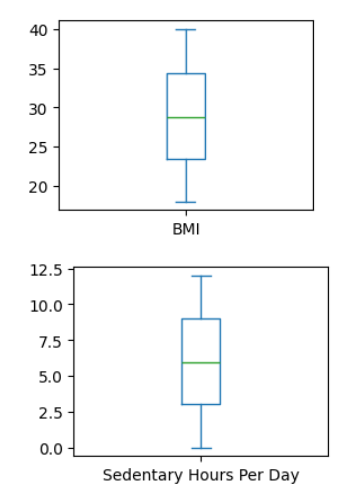
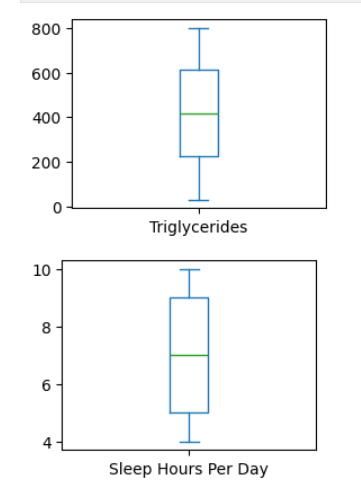
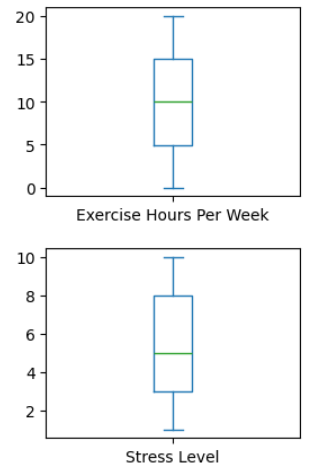


For the diet, we can see that the count for the “Healthy”, “Unhealthy” and “Average” are about the same however, healthy is still the one with more counts.

There are more males than females. About 6000 males and only 2500 females

Germany is the highest at around 480 while the rest are equally distributed.

|  |  |
| --- | --- |
|  |  |
| Generally, those who had a family history of heart attacks, have a higher rate of getting diabetes. | Those who have obesity mostly have diabetes as well. The count for the diabetes is higher for those who have obesity compared to those who do not have obesity have low count of diabetes. |
|  |  |
| Those who did not have a heart problem previopusly have more chance of being in a risk for a heart attack. | There are more people with diabetes but do not have a heart attack risk compared to those who have diabetes but have heart attack risk. Those who do not have diabetes, the number of people with heart attack risk is also low. |
|  |  |
| We can see that most people are males and generally they do not have a heart attack risk. Same for females, their risk of getting a heart attack is also half. | Those who smoke, have a higher chance of getting diabetes compared to those who don’t. |
|  |  |
| Those who drink alcohol have a higher chance of getting diabetes. | The counts seem to be stable for any level of stress. There is no big increase for a particular level of stress count. |

For the age’s box plot, we can see that the lowest value is around 20 and the highest is >80. The median age is around 55. Lowest cholesterol level is < 200 and the highest is at 400 the median for this is around 250. The lowest heart rate recorded would be 40 and the highest is > 100 the median rate is around 75. Lowest BMI id < 20 and highest is 40 with the median being at 28. Some people have 0 hours of sedentary lifestyle while some have the highest at around 12.5 hours a day and the media is around 6 hours a day. Lowest triglycerides are at 0 and the highest is at 800 the median is 400. Some people only sleep for 4 hours a day lowest, and some do for 10 hours a day, but the median is 7 the values for this is very spread. Some don’t exercise at all every week, and some do for 20 hours, and the median is 10 hours a week. Most people’s stress level seems to be distributed but the lowest is 1 and the highest is 10 and the median is 5.

|  |  |
| --- | --- |
|  |  |
| Here we can see that the purple-colored areas would be the ones with the low count and if it is brighter, it represents a higher count. On this plot we can see that if the BMI level is high then the cholesterol will also be high. | It is a bit more diverse here as some indicate that when the triglycerides are low then the cholesterol is also low. There are also some points which indicate if triglycerides are at 600 – 700 then the cholesterol is around 250. There is also another point where triglycerides are around 150 but the cholesterol is around 350. |
|  |  |
| I also made some violin plots to understand my data better. This plot shows the age distribution across the Male and Female. Generally, It shows a similar spread with the median age about the same for both genders. Seems like the Male age’s distribution is more complex compared to the female’s age. | I also made a scatter matrix however it is not very insightful. There don’t seem to be a specific trend or a pattern |
| Non-smokers have a slightly broader distribution than smokers, indicating more broad BMI values. | People from the Asia continent have the highest risk of heart attack. |

# Methods and Improvements

**Feature engineering:**

* **Delete columns**: I have also performed feature engineering where I have removed some columns which I thought might not be impactful and might cause biasness like the Patients ID, income, Country, Hemisphere, Continent, Mediation use, Physical activity per day. I took out id because it is just a unique identifier. Income does not determine if a person will have a heart attack. There are only some countries and not all thus it might create biasness. It is unclear on what medicine they are using thus I removed it and since we have the column exercise hours per week I removed the physical activity per day. The hemisphere and continent columns do not directly impact the heart attack risk also . This dataset focuses on the health-related information only.
* **Binning**: I have also binned the Cholesterol column into 4 bins, Cholesterol\_Normal,

Cholesterol\_At\_Risk, Cholesterol\_High, Cholesterol\_Dangerous.

* **One hot encoding**: When I am training the model, I need to make sure that all my columns are numeric and are in proper format. If I do not OHE my categorical columns, I am unable to create a model. I have OHEed the Sex, Cholesterol, Diet.
* **Derive a new column**: I also wanted to make the existing column (Blood Pressure) more informative as currently it gets displayed as e.g. “194/234” displaying the systolic and diabolic blood pressure metrics. I have then decided to split this into 2 different columns Systolic and Diastolic. This is more informative especially when I am classifying it. Since it only contains numbers, it is already in the required format which is numeric. Now it is more detailed and better formatted.

|  |  |
| --- | --- |
|  |  |

Upon exploring the dataset, I have noticed that the dataset is imbalanced. The majority was class 0 and it was at about 5000 rows and the class 1 was a minority at about 3000 rows. When I was exploring the dataset and used some models, the results were very poor. I have used the models with the highest accuracy score like the LDA, NB and SVM but the model could not even identify the 1 class. All the results were 0. This issue was likely due to the imbalance in the dataset, the majority was class 0 at about 5000 rows and the class 1 was a minority at about 3000 rows. This might have led to a biased model that only predicted the 0 class.

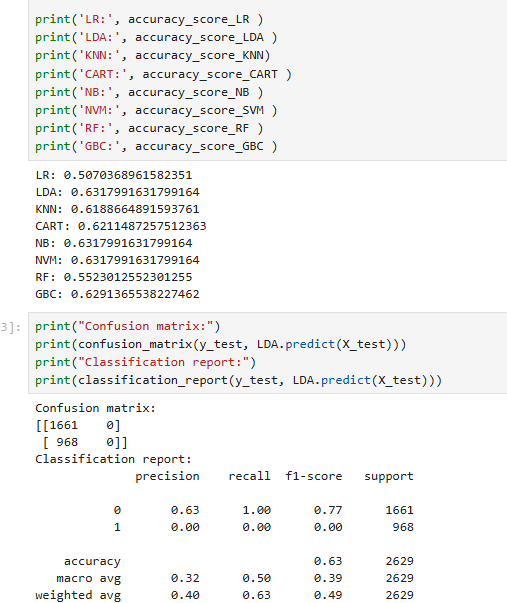
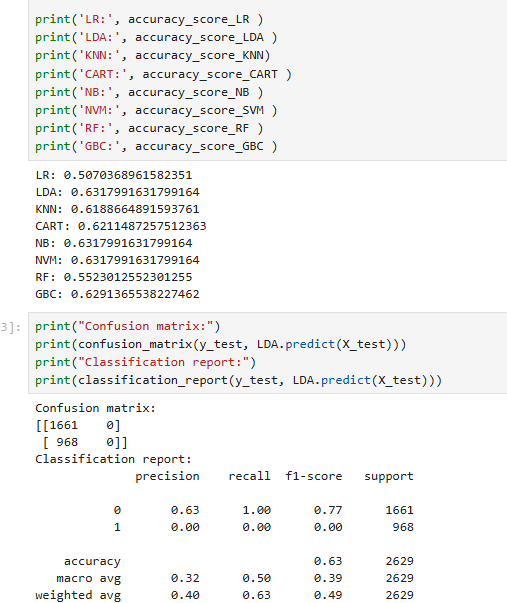
To fix this issue, I have tried out various techniques like SMOTE and Under sampling to balance the class distribution and improve the model’s ability to correctly classify the minority class.

Smote: Synthetic Minority Over-Sampling Technique is a way to handle class imbalance for datasets Unlike under sampling, which reduces the size of the majority class, SMOTE creates synthetic rows for the minority class to balance the dataset. Since it creates fake rows, there might also be a concern that the model might overfit. It might also create noise and increase the complexity.

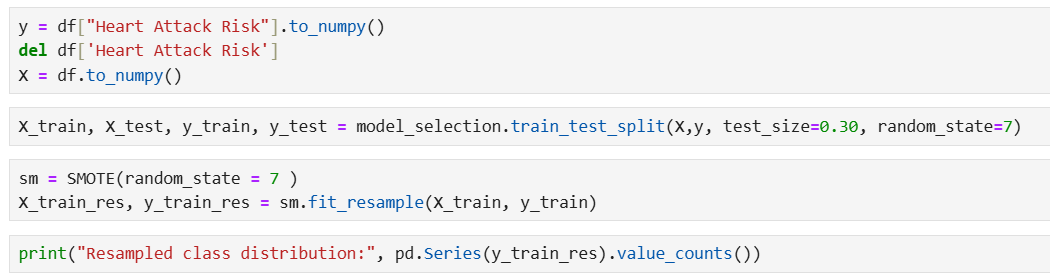
Under sampling: I reduce the size of the majority class 0, I chose to then have the same number of rows in the majority class as in the minority class. After under sampling the majority class, I finally managed to balance the data with about 3000 rows for each class.

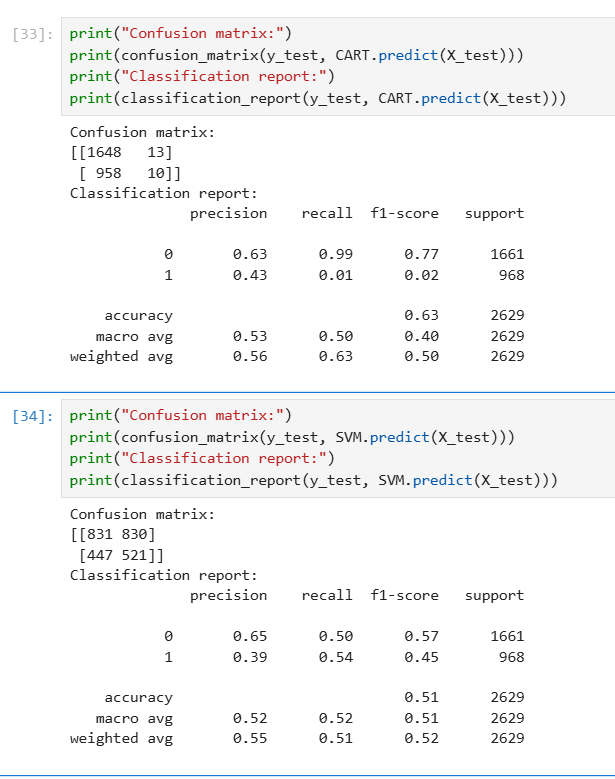
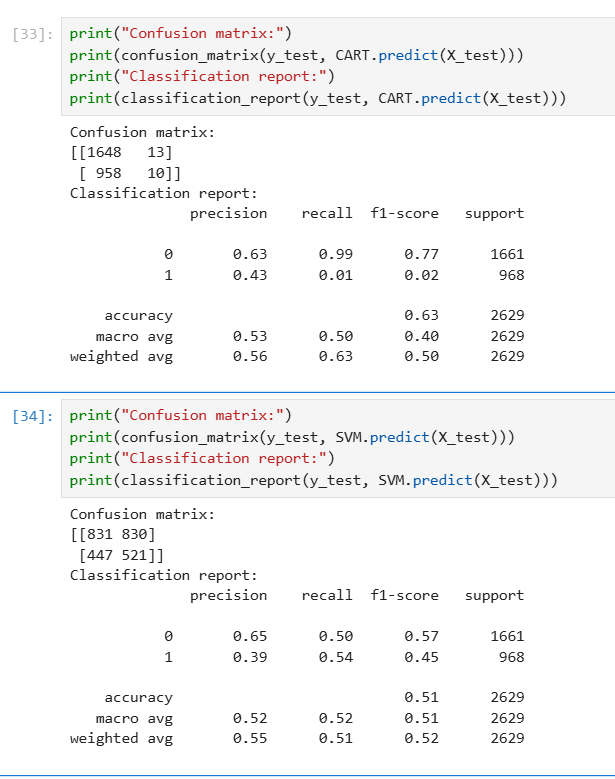
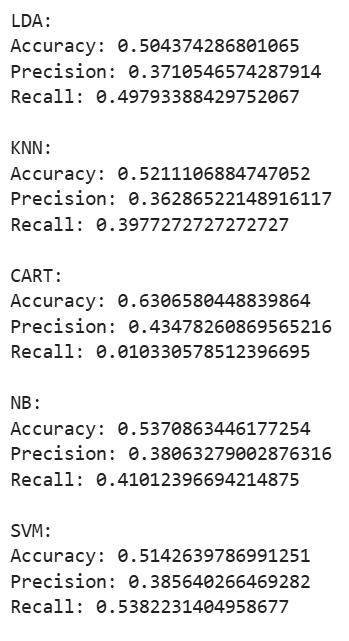
To make a fair decision when choosing which method to go with, I have modelled with all the possibilities. (Original data, Dataset which was smoted, Undersampled data)

This was the model’s result after I used the **original dataset** to evaluate. I concluded that I should not use the original dataset at all as the result for the positive class is just 0. This shows that the model is biased and cannot identify the 1 class at all.

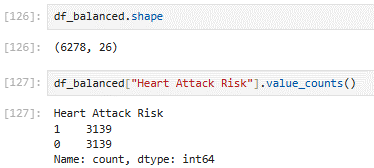
I also evaluated with smote and the prediction is not the best also. For CART, I can see that the recall is very low and it cannot predict the positive class and its very weak. The F1 score is also very low. I also used SVM and the precision is quite low for the positive class and the F1 score is also low. This method is also not very suitable.



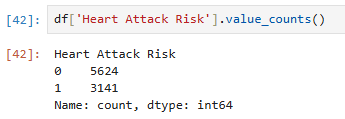


I have chosen to go with Under sampling as after modelling with different datasets. (Original, Smote, Resampled) Resample works the best.

After:



Before:



Once I was done, with the feature engineering, I have chosen to train and split my data to 80 / 20. This ensures that I have enough data for training considering that I only have 6000 rows. Then I needed to explore different models to train and test on. I have explored these models I have chosen these because these are good models which can handle the columns and rows well.

**Logistic Regression**: It is a model used for binary or multi class classification. It also can predict the probability of a class.

**Linear Discriminant Analysis**: We can use this to solve multi class classification problems as it separates the classes with multi features through data dimension reduction.

**K Neighbours Classifier**: It classifies based on what is near to the data point. It will locate the nearest neighbour of a particular data point and learn.

**Decision Tree Classifier**: This uses a tree like structure to classify the features and predict.

**Gaussian NB**: This works on a continuous data well. It follows the Gaussian distribution.

**Support Vector Machine**: It classifies the data by finding the best line or hyperplane of the data points to separate them into different classes.

**Random Forest Classifier**: An ensemble method that builds multiple decision trees and combines their outputs

**Gradient Boosting Classifier**: It combines multiple weak models to create a strong model

**Random Forest Regressor:** Ensemble method that builds multiple decision trees and shows the average prediction. It reduces overfitting by combining predictions from many trees.

**Gradient Boosting Regressor:** Ensemble method that builds trees sequentially where the errors are correct based on the previous ones. It is very accurate. It might overfit.

I have made use of various models to ensure that I am trying both the classifiers and regression models to predict. If I use the classifier, it gives me a value of either 1 or 0. If I make use of the regression, I have a wider range of value and my model might predict better as it will be based on percentages. I can set a threshold to define what will be considered a positive class. Example, if it is above 50% it can be classified as a positive case.

Metrics I used to evaluate the model. The most important metric in this case would be the F1 score as it is the harmonic mean of the precision and the recall which are both very important and critical when predicting heart attack risk.

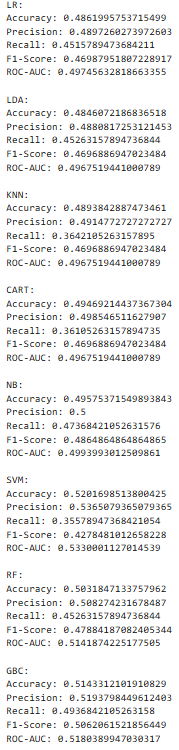
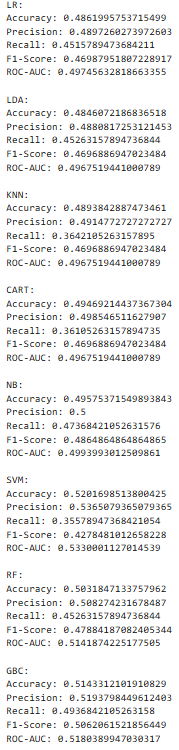
**Accuracy**: measures how well a model correctly predicts for the outcome.

**Precision**: How often the prediction for the positive class (have risk )is correct.

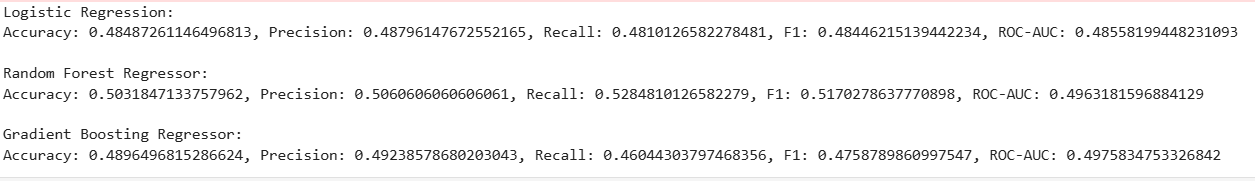
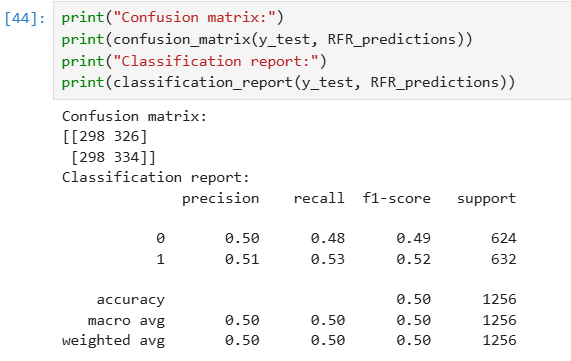
**Recall**: Measures how many of the predicted positive class outcomes are correctly identified. False negatives might be crucial as the user would not know they are in risk.

**F1 score**: Harmonic mean of precision and recall. It provides a number that balances both.

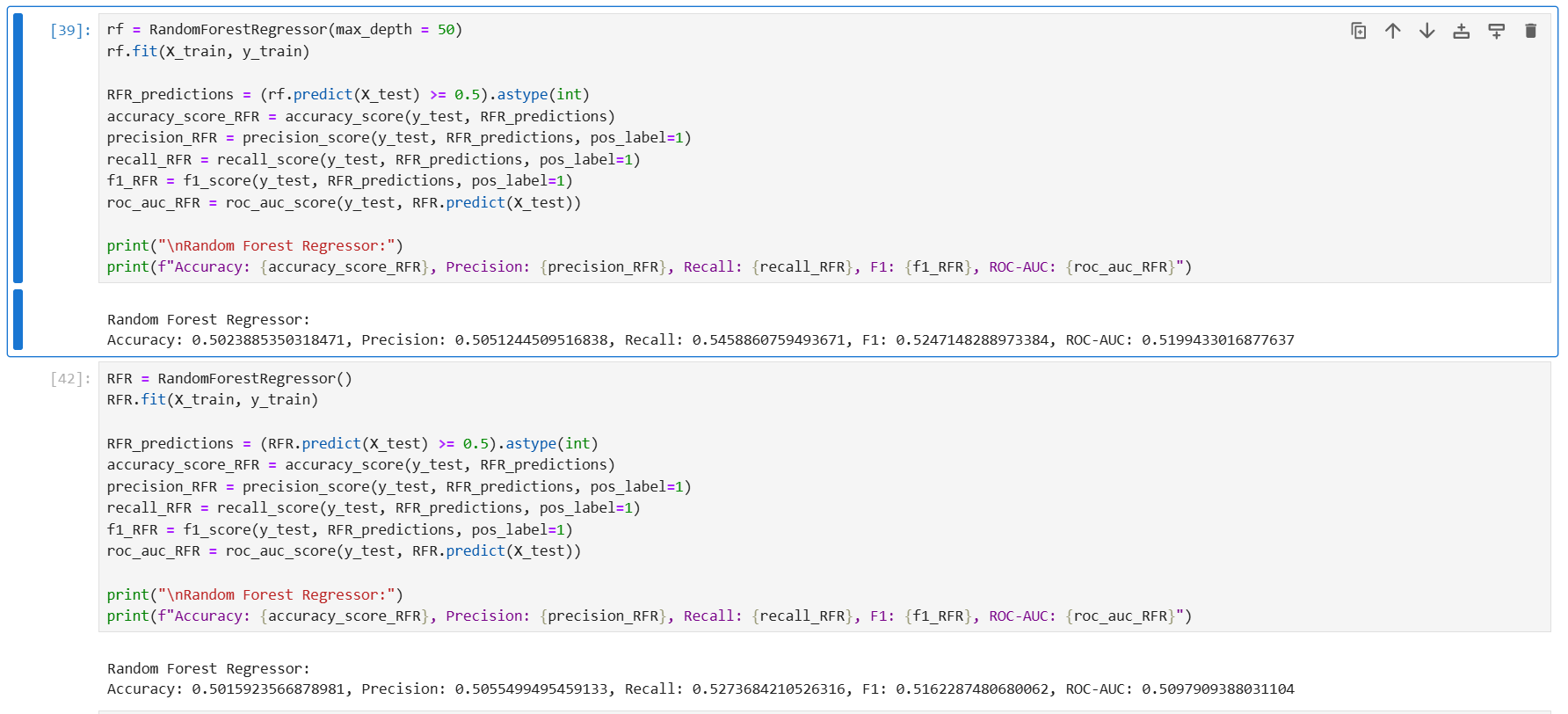
**ROC AUC**: Measures a model's ability to distinguish between classes by plotting the true positive rate (recall) against the false positive rate.

Overall, the classifier models are not doing great as the accuracy for all is very low. The models are unable to learn the data and make proper predictions. In terms of accuracy, SVM is performing the best amongst the rest of the models at 0.52 and the worst performing model would be the LDA at 0.48. As we are predicting the heart attack risk, it is important that we minimise false negatives and false positives and make sure to have the highest number of true negative and true positives when predicting to make sure our model is good and accurate. High precision and high recall is very important for our model and F1 score usually tells us the mean for both the metrics. High F1 score is good. GBC has the highest F1 score at 0.50 this means that it is average. The lowest F1 score is SVM at only 0.42. This is the worst performing model.

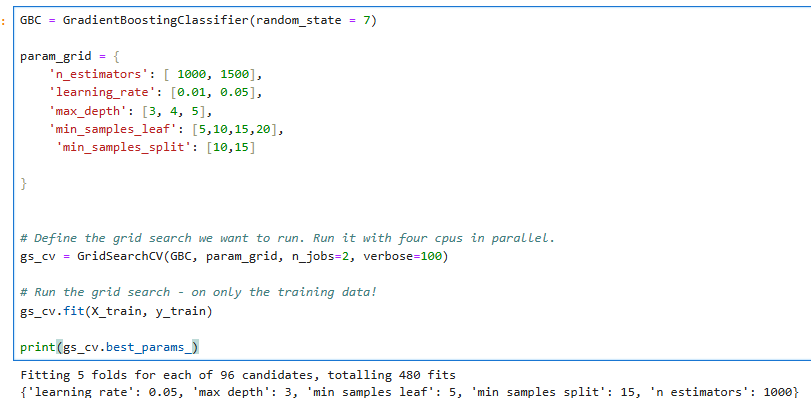
 

I have also used more regression models and in these regression models, the model which is performing the best in terms of accuracy would be Random Forest regressor. The highest F1 score observed is also the Random forest regressor at 0.51. This is even better compared to the Classification models as observed above. In our case, precision and recall is very important to minimise false positive and false negatives. Thus, I feel like a random forest regressor is the best performing model. Recall is also higher at 0.52

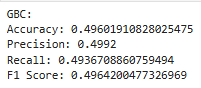


Improving models with **Grid search , Optuna** & **hyperparameter** **tuning**.

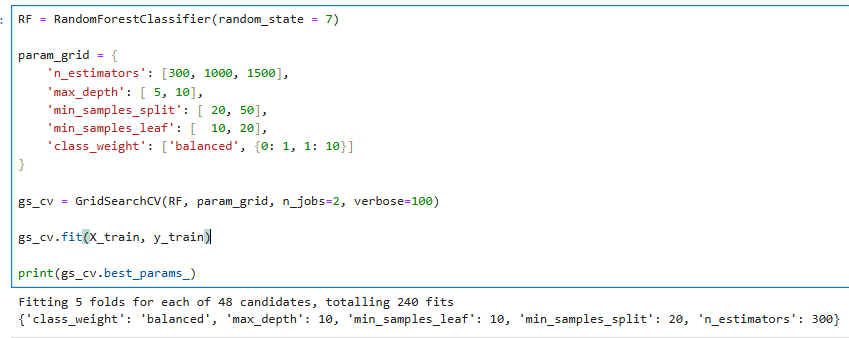
I have tried out multiple and various ways to improve the model’s performance. I have set a lot of hyperparameters however, that is time consuming as we need to test which is the best performing 1 by 1. I have then implemented Grid Search. Just to see if any model’s performance increases.



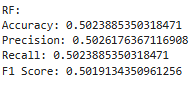
This gives me:



Based on this grid search results, these parameters do not seem to be very suitable as the results have dropped.

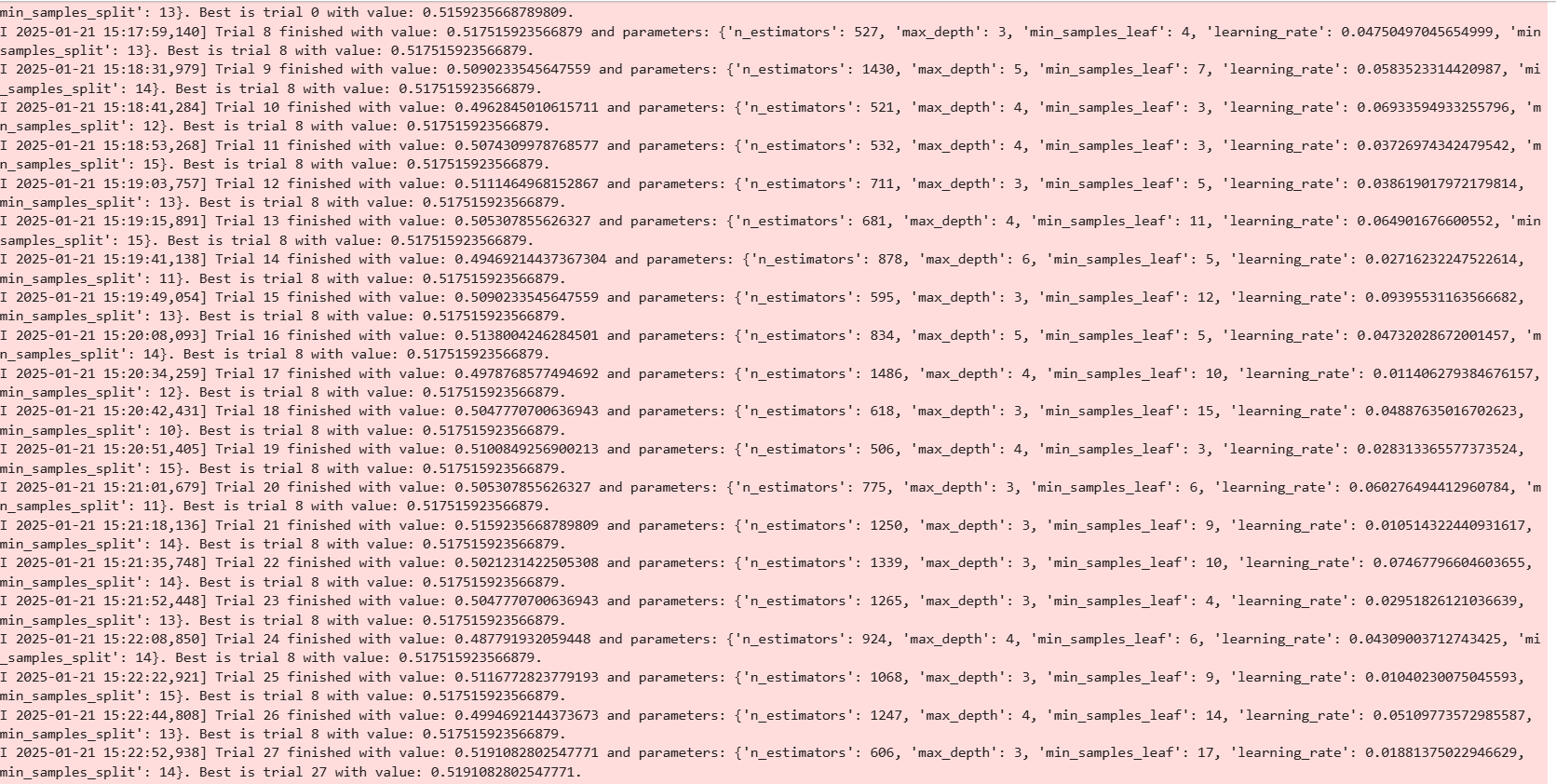


This gives me:



GBC and RF, from the results printed and re modelled, i can understand that the Random forest works better.

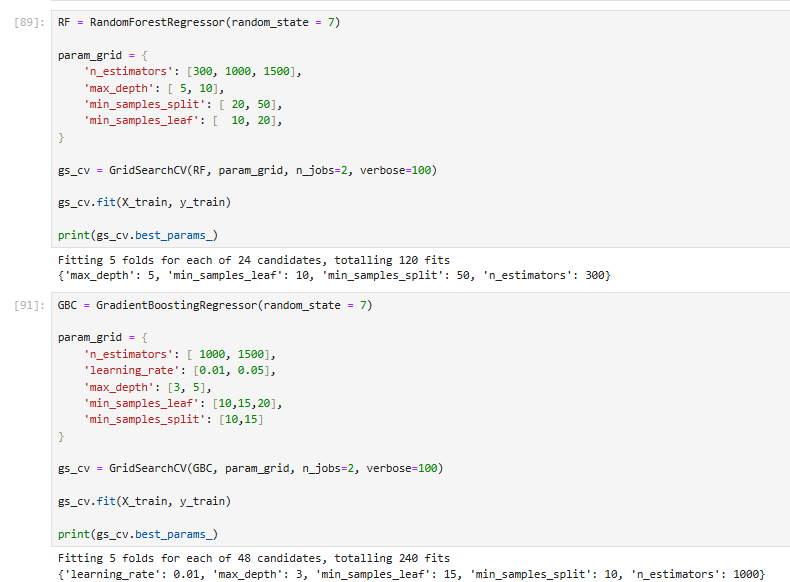
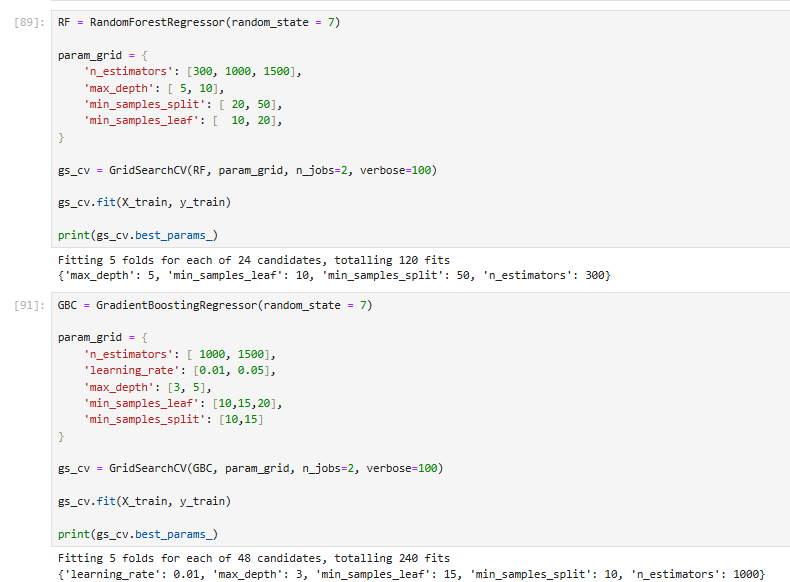
I have also explored another method to tune fand search for the best hyperparameters, I have used Optuna. This is another method where I can set some parameters and set some number of trails so that it can run through them and find the best parameters for the model.

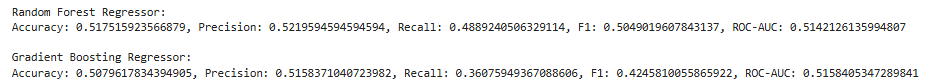




It then identifies and prints out the best result.

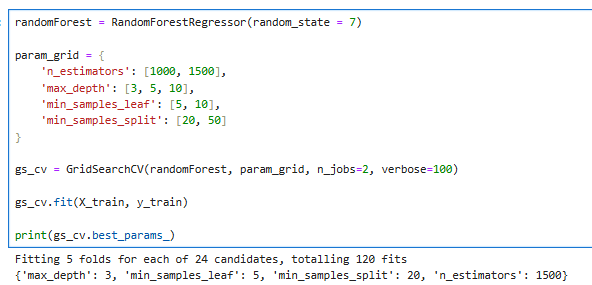
I have also done grid search for the regression-based models and these are best results identified. I then modelled them again and below are the results I got back. Comparing the 2 models, Random Forest Regressor still performs better.





This also proves that random forest is better.

I also did another grid search for the random forest regressor



This gives me:



This shows that the recall and F1 score is lower compared to the other random forest model meaning these parameters are not suitable when considering recall.

**Trying out more parameters:**

n est = 1500

Accuracy: 0.5135350318471338, Precision: 0.5171288743882545, Recall: 0.5015822784810127, F1: 0.5092369477911647, ROC-AUC: 0.5172985637779942

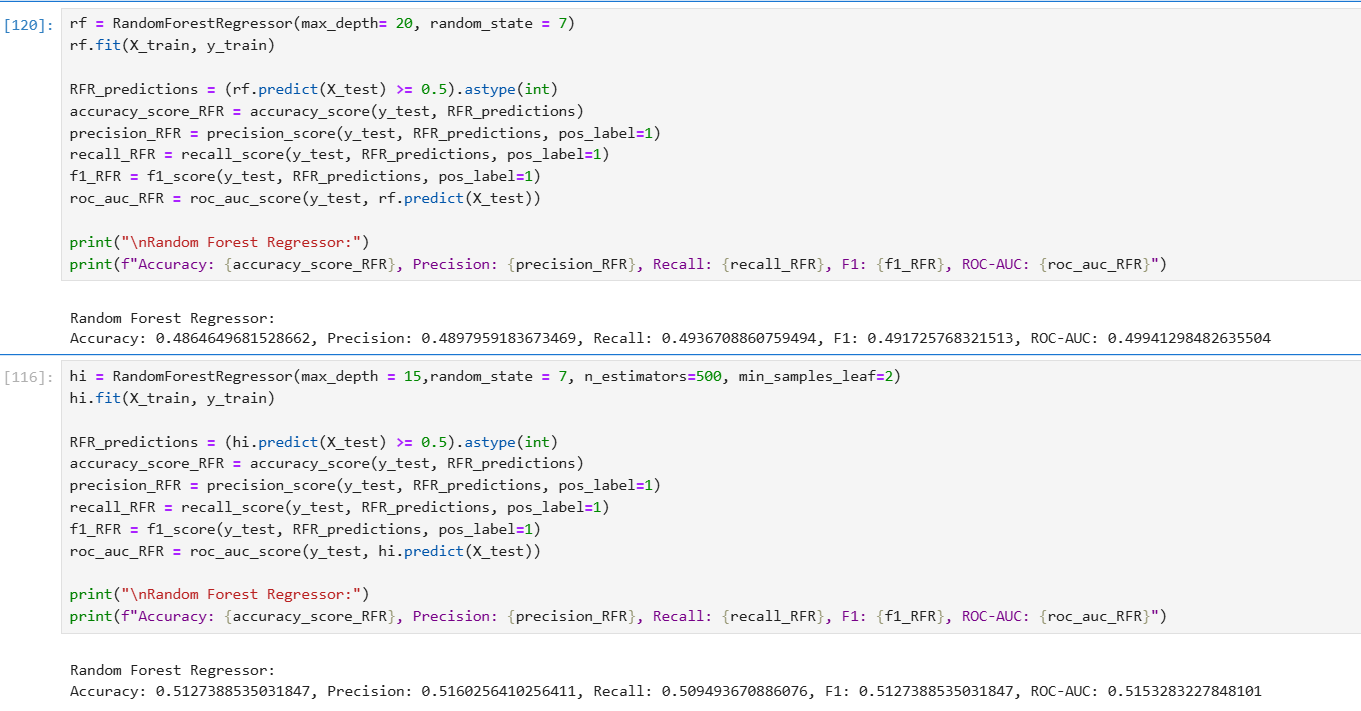
n\_est = 500

Accuracy: 0.5127388535031847, Precision: 0.5160256410256411, Recall: 0.509493670886076, F1: 0.5127388535031847, ROC-AUC: 0.5153283227848101

n est = 300

Accuracy: 0.5151273885350318, Precision: 0.5187601957585645, Recall: 0.5031645569620253, F1: 0.5108433734939759, ROC-AUC: 0.5106271299902629

These are some adjustments I made manually and after getting some suggestions from the grid search also, and I realised that n estimators = 500 is the best.

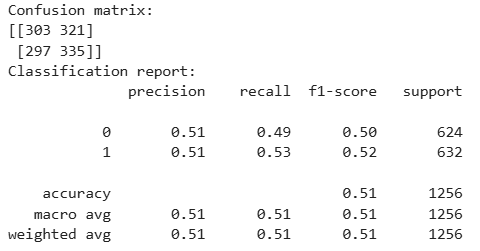


After

Before

I added more hyperparameters based on what I understood from the grid search and after manually testing out. Seems like grid search does not always give the best and most suitable parameters.

This shows that the model has improved



Like what I have mentioned above, There are many metrics to evaluate like precision, recall, accuracy, F1 score which is the mean of precision and recall. For a scenario like this, it is best to maintain a high recall and precision to minimise the false positives and ensure to identify the highest number of positive cases correctly (recall).

I have also done feature importance to view and visualise the most important features in my model. I included all the features and created a model. \

**Feature importance:**

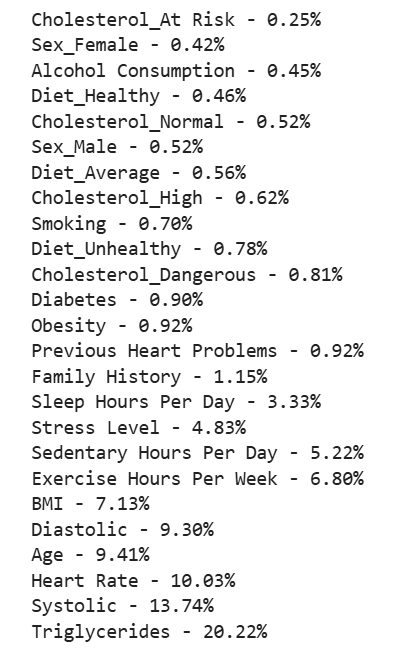
When I use all the features:



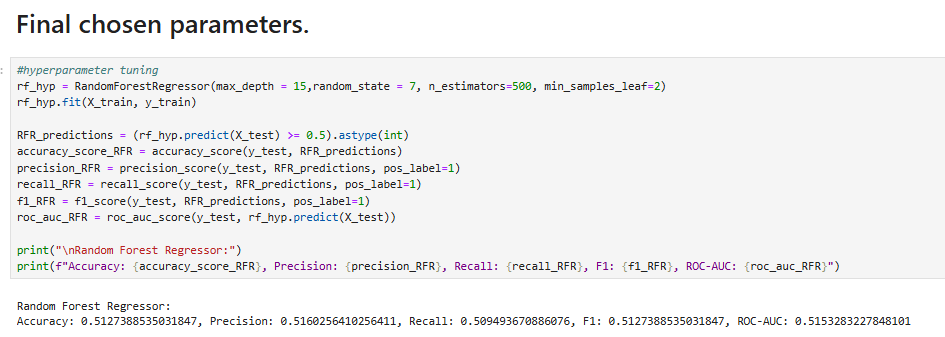
From this, I understand that having too many features (57 columns after one-hot encoding) does not significantly improve the model's performance. The highest feature importance is only 16.66%. This tells that not many features strongly help with the outcome.

Feature importance after feature engineering. (From this I can understand the most impactful features)



# Results and analysis

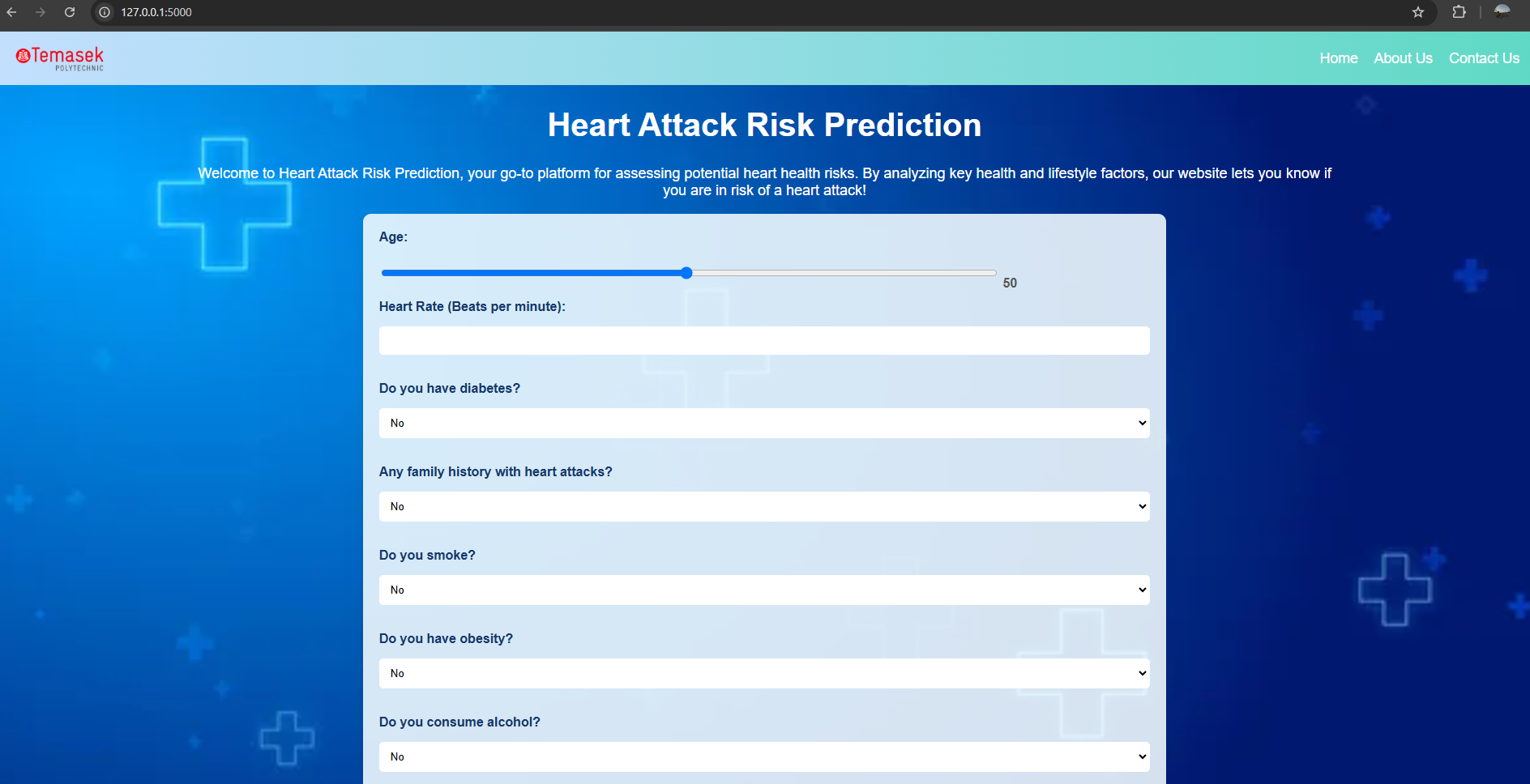
I chose Random Forest Regressor as my **final model** with some hyperparameters set such as max\_depth = 15,random\_state = 7, n\_estimators=500, min\_samples\_leaf=2. This model gives me this result:



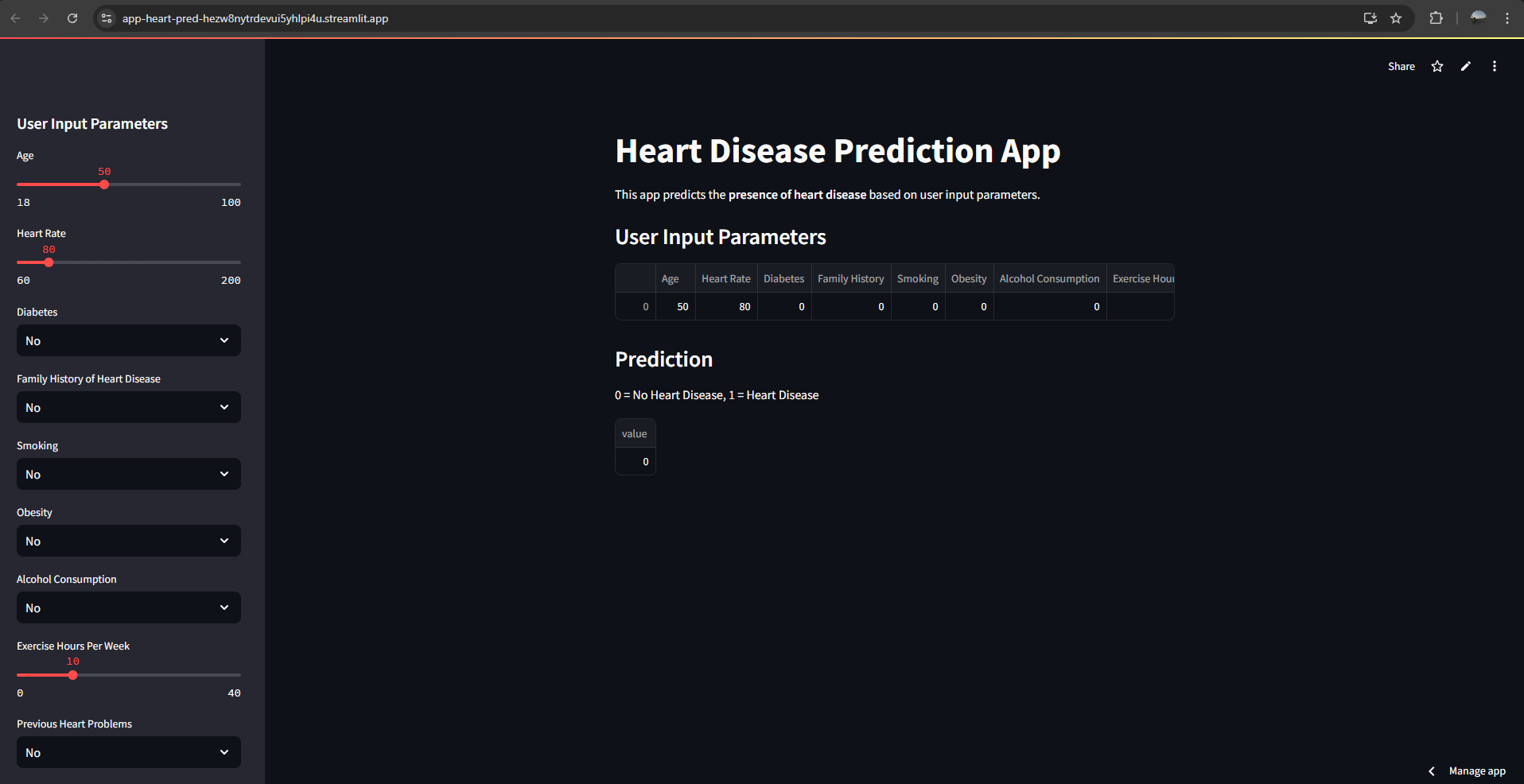
The accuracy is higher compared to other models and the F1 score is also better. F1 score is important because it is crucial to minimise false positive and false negative. We need to make sure the model identifies the risk properly and accuracy is high.

**Deployment:**

* Flask (Main)
  + This is a micro web framework written in python and I created a HTML file and created an app.ipynb to create the flask application. I loaded the model inside the app and then I defined a route and created a function which would render the index.html template. Then I also created another function responsible for the prediction. I made sure that the inputs and the web page are user friendly. I made use of dropdowns and sliders to make it easier for the users. I also made sure my application is robust, it does not accept empty values and I made sure no negative values are sent. I then collect back the responses for each feature as an array which is then needed for the model to predict based on the values. This application runs on the 127.0.0.1/5000



* Streamlit
  + I was able to host it locally as well as on the stream lit cloud for other people to access and use my model. I created a heart-pred.py which contains the Streamlit website code and inputs as well as the function which can predict based on the inputs taken I placed the codes to split and train. The dataset I am training it the processed dataset which is balanced, feature engineered and cleaned. I am able to run it locally as well but I also wanted to try to host it on the cloud. I added my codes to a GitHub repo and created a requirements.txt file which includes the libraries and imports that I need. I also uploaded the dataset into the repo. I then connected my repo to the streamlit.io and created an App to deploy.



# Conclusion

Overall, in this project I managed to build a model to predict the heart attack risk. I have done proper data exploration and cleaning to ensure that my dataset is suitable for modeling. I also then did feature engineering: Create new column, binning, one hot encoding to make sure that the columns are more suitable. I also then balanced my dataset to reduce biasness in my models. To select a good set of hyperparameters, I have made use of grid search and Optuna. After finding out which model is the best suited and which model’s hyperparameters I should use, I exported and saved the model.

I then continued to deploy this prediction model as a website using Flask. Other than flask, I also explored Streamlit to host it on the cloud. I have made sure that my user experience on the website is good and friendly. I made sure that the user does not have to click a lot of buttons to find out their risk. The inputs are all well labeled and made use of sliders and dropdown. Overall, through this project, I have learnt how to handle data better and what makes a model good and how to properly evaluate it. I also explored a variety of models and methods to improve them.

**References:**  
https://www.mathworks.com/campaigns/offers/next/choosing-the-best-machine-learning-classification-model-and-avoiding-overfitting.html

https://builtin.com/data-science/supervised-machine-learning-classification

https://optuna.readthedocs.io/en/stable/tutorial/

https://www.geeksforgeeks.org/smote-for-imbalanced-classification-with-python/

***The end***