PROGRAMMING FOR DATA ANALYTICS

Task 2

Kieran Glezer-Jones
ST10041889

Contents

1 Introduction

Heart disease is a condition where your heart experiences diseased vessels, blood clots, and structural problems that resulted in 868 662 casualties in the US alone in 2017 (American Heart Association, 2021).

However, the disease is by no means a death sentence, your chances are slim but there is a treatment for the disease. But it also means that failure to get this treatment can heavily reduce your odds of survival (Rossignol, et al., 2019).

So, I set out to collect a dataset that holds records of many different cases of people with and without heart disease. The aim is to analyse the data, come to some conclusions and then ultimately build a model that can accurately detect if someone has heart disease or not. If I am successful in creating this model then a patient can visit a doctor, who will take their stats and use the model to help confirm whether the patient is experiencing potential heart disease.

Many lives could be saved or slightly extended this way as this algorithm would make it very easy for anyone, even the patient themselves, to get a quick and instant response to this. In fact, with the way phones are advancing now, you could even make an app at collects much of this data in the near future. Assuming the technology for this will exist soon enough, getting the appropriate X inputs might not be difficult at all, it may just require a scanner and a bit of personal information.

With the algorithm embedded into everyone's phone, it could save hundreds of thousands of lives as your phone could pick up potential problems as soon as it becomes a problem and people can start treatment right away. With enough backing from a big company like Google or Apple, these features could be built into future iterations of their phones.

2 Background

This section covers the background of the data analysis report and looks at what is known before we start coding.

2.1 Data Set

The dataset I will be using for the analysis and creation of the model is the Heart Failure Prediction Dataset from Kaggle.com by (Fedesoriano, 2021). The dataset has a healthy number of records from a variety of different places and where sources from various hospitals within said places.

Source	No. of records.
Cleveland	303
Hungarian	294
Switzerland	123
Long Beach VA	200
Stalog (Heart) Data Set	270

Source: (Fedesoriano, 2021)

This means that the dataset has a total of 918 records.

The actual dataset itself consists of 12 features:

Variable Name	Units or Categories	Description	Qual/Quan	Type
Age	Years	How many years the person has been alive.	Quantitative	Discrete
Sex	Male (M) Female(F)	What gender the person is.	Qualitative	Nominal
ChestPainType	Typical Angina (TA) Atypical Angina (ATA) Non-Anginal Pain (NAP) Asymptomatic (ASY)	What type of chest pain the person is experiencing.	Qualitative	Nominal
RestingBP	mm Hg	The Patient's resting Blood pressure.	Quantitative	Discrete
Cholesterol	mm/dl	The patient's cholesterol levels.	Quantitative	Discrete
FastingBS	1: if FastingBS > 120 mg/dl, 0: otherwise	The patient's fasting blood sugar	Qualitative	Nominal
RestingECG	Normal (Normal) Having ST-T wave abnormality (ST) Showing probable or definite left ventricular hypertrophy by Estes' criteria (LVH)	The patients Resting ECG.	Qualitative	Nominal
MaxHR	60 - 202	The patient's maximum heart rate achieved.	Quantitative	Discrete
ExerciseAngina	Yes (Y) No (N)	If there was an exercise-induced angina.	Qualitative	Nominal
Oldpeak	Numeric value	ST depression induced by exercise relative to rest.	Quantitative	Continuous
ST_Slope	Upsloping (Up) Flat (Flat) Down sloping (Down)	The slope of the peak exercise ST segment.	Qualitative	Nominal
HeartDisease	Heart Disease (1) Normal (0)	Output class that describes if someone has heart disease or not.	Qualitative	Nominal

Source: (Fedesoriano, 2021)

2.2 Correlation

The correlation of data describes the strength of the relationship between them and is expressed as a number between -1 and +1 (Akoglu, 2018). Where a -1 is a perfect negative correlation and +1 is a perfect positive correlation. 0 would be a non-existent correlation.

Correlation Coefficient		Dancey & Reidy (Psychology)	Quinnipiac University (Politics)	Chan YH (Medicine)
+1	−1	Perfect	Perfect	Perfect
+0.9	−0.9	Strong	Very Strong	Very Strong
+0.8	−0.8	Strong	Very Strong	Very Strong
+0.7	−0.7	Strong	Very Strong	Moderate
+0.6	−0.6	Moderate	Strong	Moderate
+0.5	−0.5	Moderate	Strong	Fair
+0.4	−0.4	Moderate	Strong	Fair
+0.3	−0.3	Weak	Moderate	Fair
+0.2	−0.2	Weak	Weak	Poor
+0.1	−0.1	Weak	Negligible	Poor
0	0	Zero	None	None

Figure 1 Correlation strengths by (Akoglu, 2018)

We see that if the correlation is between -1 and -0.7 then it is a strong negative correlation if it is between -0.69 and -0.3 it is a moderate negative correlation and if it is between -0.29 and 0 it is a weak negative correlation. If the correlation is between 1 and 0.7 then it is a strong positive correlation if it is between 0.69 and 0.3 it is a moderate positive correlation and if it is between 0.29 and 0 it is a weak positive correlation.

If two variables have a strong positive correlation then as the independent variable goes up, the dependent variable will also go up. If the two variables have a strong negative relationship, then as the independent variable goes up the dependent variable will go down.

2.3 Classification

Classification is a type of supervised machine learning algorithm that will take input features as an X variable and predict a Y output. The important distinction between regression and classification is that regression aims to output a continuous value while classification aims to output a category. The output from a classification model can be numeric, but it will have to be discrete.

When we say that classification is a type of supervised learning, it means that we train the model with known inputs and known outputs. We know what we put into the model, and we know what we want out of it. We give it with both an X variable and a Y variable.

The dataset we will be using is good for classification as it has the HeartDisease feature which is either a 0 or a 1. This means that by giving all the other features to the model we should be able to classify the data as either 0 or 1.

2.4 Descriptive Analysis

Descriptive analytics revolves around the concept of taking raw historical data and then data mining is to discover trends and relationships (The Independent Institute of Education, 2022). The process of data mining involves breaking the raw data down and comparing, often visually, to other elements within the dataset to see how they interact with each other. The rationality behind performing this type of analysis is that it answers the question of “what happened”, or “What is happening”.

2.5 Predictive Analysis

The predictive analysis aims to take a large amount of historical data to find the patterns, trends, and relationships in an attempt to predict "what is going to happen next" (Cote, 2021).

To perform this analysis, we will be ultimately building a model based on a classification algorithm. We will feed the algorithm historical data and hopefully, it will output, with great accuracy, whether you have heart disease. The model doesn't know if you have heart disease or not, it is simply making a prediction that you do.

Where the model is ultimately flawed is that you will likely only know you have heart disease once you already have it. So, in that regard what the model is actually predicting is whether or not you are likely to have heart failure.

2.6 Useability

Ultimately, we are going to be using a classification model to input data and then get a prediction which is either a 0 or a 1. To make sure we can use this dataset there are a few factors that need to be met.

2.6.1 Legitimate source

If the model is trained on a dataset that was created by randomly generating numbers and categories, then it's not very likely that it will perform when handed real-life data. This is because the model won't accurately capture the true relationships behind these variables.

The provided dataset, however, was sourced from credible sources who were proud enough to put their names down as contributors to it. They are real-life records from various places that were merged to create a single reliable dataset.

2.6.2 Categoric target

The dataset needs to have an actual target that the model can work towards. You can't just throw data at the model and ask it to predict something it has never even heard about. So, you must supply the model with an appropriate Y variable that the X feature/s can predict. It's also important that the Y variable records are in parallel to the X variables. In the case of a classification model, it's also important that the output is a categoric prediction. The classification algorithms are not designed to output a quantitative Y prediction, so expecting them to do so will fail to fit the model.

In the case of our current dataset, we have the HeartDisease variable which is a one-hot binary variable that is either a 0 or a 1. This is something the classification algorithms can work with without issue. The feature also comes from the same dataset as the other features we will be using to train the model. This means that there is no issue with pulling the HeartDisease feature as your Y variable and using the other features as your X variable as the records will be parallel to each other by default.

2.6.3 Relationships between the variables

You can't feed the model an X variable that asks people what their favorite color is and then a Y variable that predicts the queen's next meal. These two factors likely have nothing to do with each other. This means that feeding them into a model would not output accurate results

with future predictions. This is why it's important to have variables with a fair level of correlation between them. As if one variable does not affect another then it's not going to add value to the model's accuracy.

Luckily the dataset we are using will be features that are considered important to verify the health of your heart. Such as your age and max heart rate. We can also quickly draw up a correlation matrix to see how each input variable affects the target variable.

Figure 2 Correlation Matrix for Heart Disease

We see that there are quite a few variables with a decent correlation to the target. None with a correlation strong enough to justify using them as the only feature in the X variable, but even the small correlations add a little bit of value towards the accuracy when you put enough of them together.

2.6.4 Enough features

When working with simple linear regression you only really need one good feature to put into the X variable to accurately predict the Y variable. Where this could technically happen with a classification model, it isn't very likely, and if it does happen, then that model likely isn't very useful, to begin with as it could just be replaced by a switch statement. So, classification will require enough relevant features to put into the X variable to come to an accurate prediction.

Luckily, when we look at the correlation matrix, we see there are quite a few variables that hold a strong enough relationship to the target that they would without a doubt add value to the accuracy.

2.6.5 Enough data

Too few records could result in the data not being able to grasp a real trend. Ultimately the model is likely being trained on a sample that represents the full population. This means that a sample size that is too small may not accurately represent the population. Thus, your model may not perform well in production or even testing.

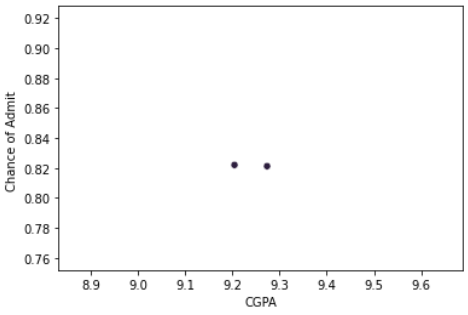

Figure 4 Too little data

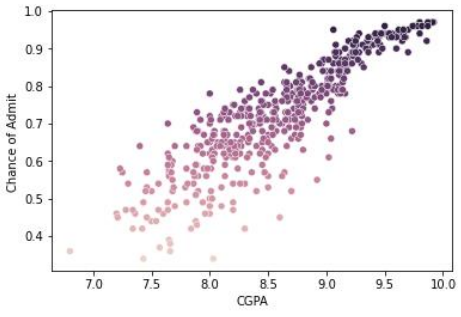

Figure 3 Enough data

Even though it's a drastic example, we see that in figure 3 the trend looks like it's horizontal, but as soon as we add the rest, we see the full picture regarding the direction of the data.

3 Data Analytics

We are going to perform descriptive analytics on the dataset in an attempt to uncover any valuable information that could be used to either further increase our knowledge of the situation or vastly optimize our model. For a more in-depth look at the data analysis process for this specific dataset, please refer to the first section in the Jupyter Notebook file provided.

3.1 Data collection and cleaning

The first step of the analysis process involves collecting the data and making sure that it is clean and ready to use. If the data has null or incorrect values, then they need to be fixed before we start working with them as they may throw off our perception of the data.

We run a test to find if there are any null values and the output shows that there are none. This does not mean, however, that all the values are perfect. The next thing we do is describe the data and look for anything that could be considered abnormal.

	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	HeartDisease
count	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000
mean	53.510893	132.396514	198.799564	0.233115	136.809368	0.887364	0.553377
std	9.432617	18.514154	109.384145	0.423046	25.460334	1.066570	0.497414
min	28.000000	0.000000	0.000000	0.000000	60.000000	-2.600000	0.000000
25%	47.000000	120.000000	173.250000	0.000000	120.000000	0.000000	0.000000
50%	54.000000	130.000000	223.000000	0.000000	138.000000	0.600000	1.000000
75%	60.000000	140.000000	267.000000	0.000000	156.000000	1.500000	1.000000
max	77.000000	200.000000	603.000000	1.000000	202.000000	6.200000	1.000000

Figure 5 Describing Numeric features

We quickly notice that there is an issue with cholesterol and resting blood pressure as they both have a minimum value of 0. This is most likely a mistake as it's extremely unlikely that these values would be 0 as that low of a cholesterol level is hard to believe, but that low of a resting blood pressure level would probably result in a dead patient.

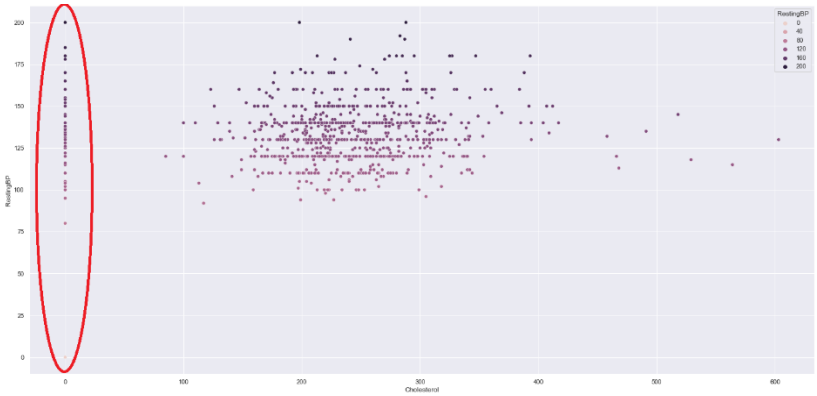

Figure 6 Scatterplot of RestingBP vs Cholesterol

From the scatterplot, we see quite a few cholesterol records being 0 and a single resting BP level being 0. We convert these records to null and drop them.

	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	HeartDisease
count	746.000000	746.000000	746.000000	746.000000	746.000000	746.000000	746.000000
mean	52.882038	133.022788	244.635389	0.167560	140.226542	0.901609	0.477212
std	9.505888	17.282750	59.153524	0.373726	24.524107	1.072861	0.499816
min	28.000000	92.000000	85.000000	0.000000	69.000000	-0.100000	0.000000
25%	46.000000	120.000000	207.250000	0.000000	122.000000	0.000000	0.000000
50%	54.000000	130.000000	237.000000	0.000000	140.000000	0.500000	0.000000
75%	59.000000	140.000000	275.000000	0.000000	160.000000	1.500000	1.000000
max	77.000000	200.000000	603.000000	1.000000	202.000000	6.200000	1.000000

Figure 7 Describing data after dropping nulls

We see that we only have about 81% of the original data left, but that should still be enough to train and test our model. The data integrity among these variables is also looking significantly better.

Something quite notable to add is that the average age of the data is quite high. The minimum value for the data is also quite high. Already from this, we get the idea that heart disease is more prominent in people of older ages.

The next thing we need to do is create some dummy values for the categorical data. This will split the data from those categoric features into multiple binary columns.

RestingECG
Normal
Normal
ST
Normal
Normal
...
Normal
Normal
Normal
LVH
Normal

Figure 8 Before dummies

RestingECG_Normal	RestingECG_ST
1	0
1	0
0	1
1	0
1	0
...	...
1	0
1	0
1	0
0	0
1	0

Figure 9 After dummies

We notice how it breaks the categories within one feature into multiple features which are either a 0 or a 1. If the value is 1 in [RestingECG_ST] then the value for that record is ST. It is also interesting how there are three categories which are Normal, ST, and LVH but there is no column for [RestingECG_LVH]. This is because if [RestingECG_Normal] is 0 and

[RestingECG_ST] is 0 then by default [RestingECG_LVH] has to be 1. It might be difficult for us to see this, but the computer does not care.

We however do care since it will mess with our analysis as we will be missing a few columns. So, we bring the columns back by setting drop_first to false when parsing the parameters to the get_dummies() method.

The next thing we can do is check the scatterplot of all the variables, even the categorical ones now that we have converted them into numerical values.

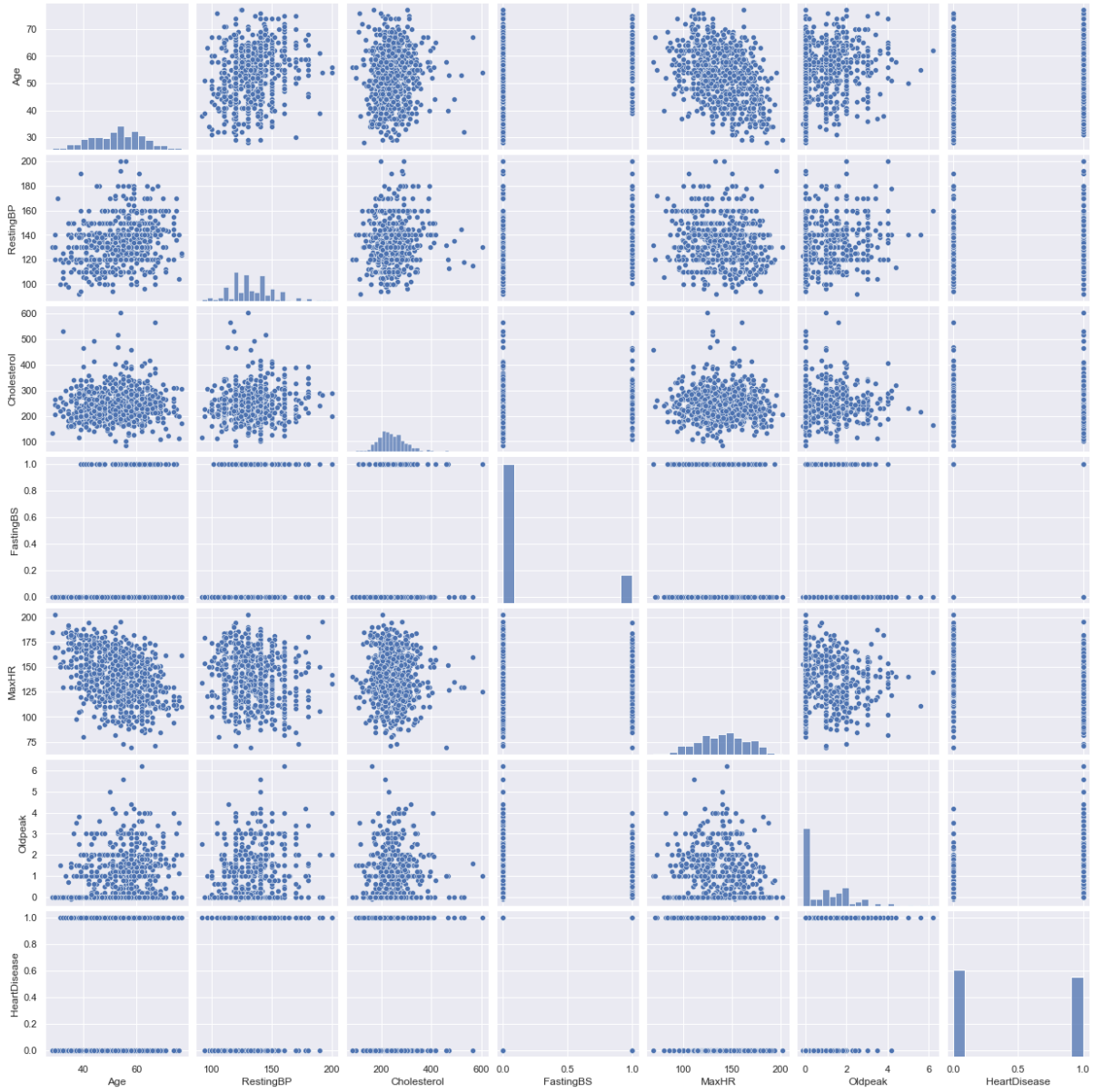

Figure 10 Scatterplot of all features

From figure 10 we see that all the variables have a good level of integrity with nothing looking out of place. We can also see that HeartDisease has a good spread between 0s and 1s. This is a good sign as it's exactly what we are looking for to make sure it's able to accurately classify both outcomes.

Next, we can check the distribution of the data to see if we find something.

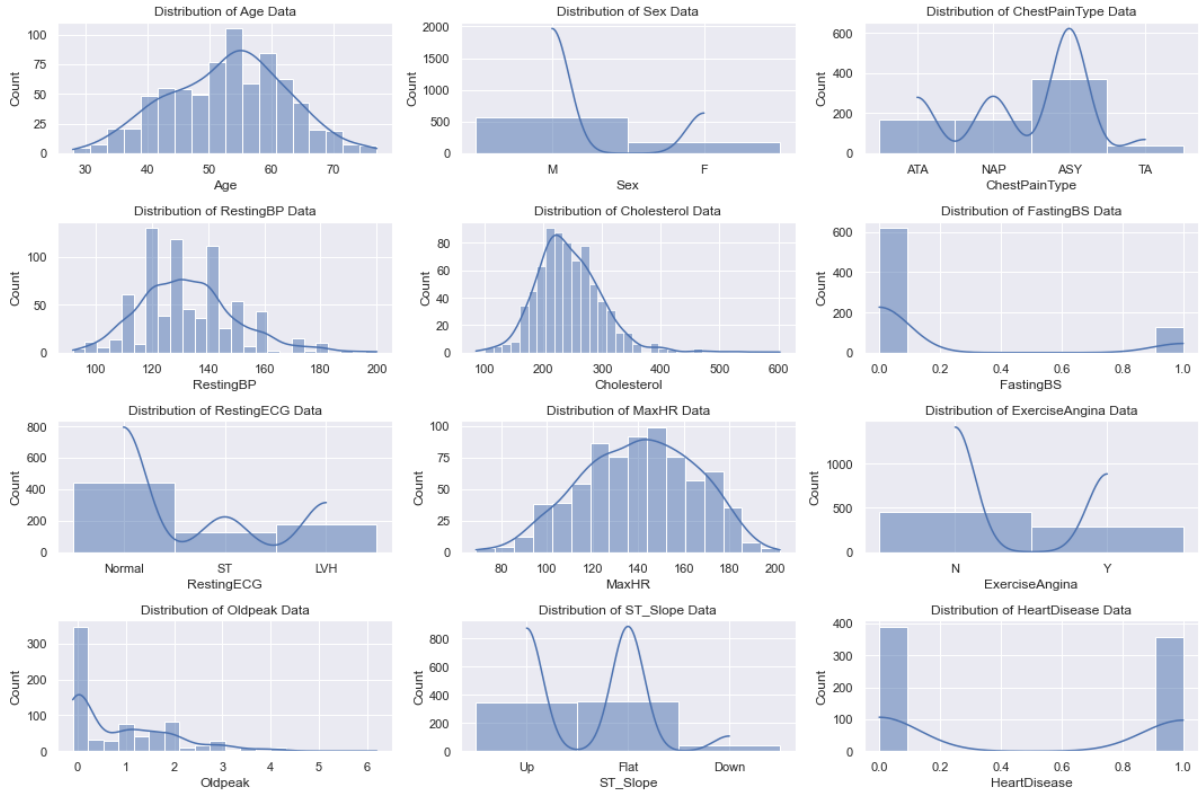

Figure 11 Distribution of data

We see Age, RestingBP, Cholesterol, and MaxHR are all normally distributed. We also see that there are very few records where ST_Slope is set to down. But most concerningly we see that is a significantly larger number of males than females. This may throw off the accuracy of the model, but there are some tricks we can use to get around this, which will use section 3.3.

We don't need to use dummy variables when creating graphs that revolve around counts. So we can convert the dummy values back when we perform those tasks.

3.2 Correlation

We can draw a correlation matrix to see what kind of relationship HeartDisease has with the other variables.

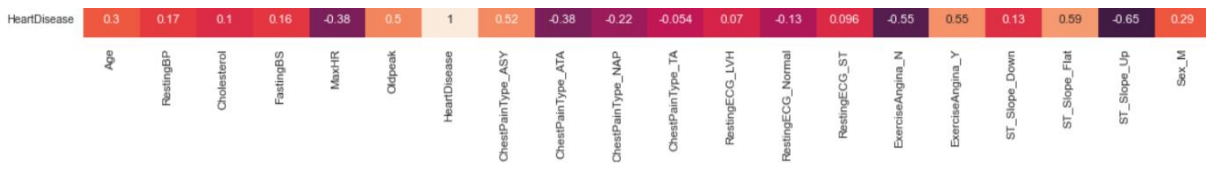

Figure 12 Correlation Matrix 2

Not only should we look at the features which have a high correlation to [HeartDisease], but we can also gain information about the features which have close to no correlation to the [HeartDisease] variable. Because if the variable does not correlate with the target, then we know it doesn't affect it either. One notable thing is that [ExerciseAngina_N] and [ExerciseAngina_Y] have -0.55 and +0.55 respectively. Which makes it pretty clear that people

who experienced exercise angina have a high chance of heart disease while people that didn't experience it likely don't have heart disease.
Other notable variables are the [ST_Slope_Flat] and the [ST_Slope_Up] which have fairly strong correlations. There is no doubt that these variables will play a large role in how accurate our model is. People experiencing upsloping have a low chance of heart disease while people who experience a flat ST slope have a high chance of heart disease. Down sloping doesn't seem to matter much.

3.3 Visualising the data

Working with numbers exclusively can make it difficult for our human brains to interpret the data. So, to make it easier for ourselves we can plot graphs to help us get a better understanding of what is happening. The following section only describes the most interesting findings from the analysis process. **This is just a summary**, for a more in-depth description of what was found and the opinions surrounding those findings, refer to the Jupyter Notebook file provided.

3.3.1 Creating graphs

The first thing we can look at is how many males and females have heart disease. The imbalance of male to female records makes this quite hard. We can, however, take the male records and female records and compare them separately.

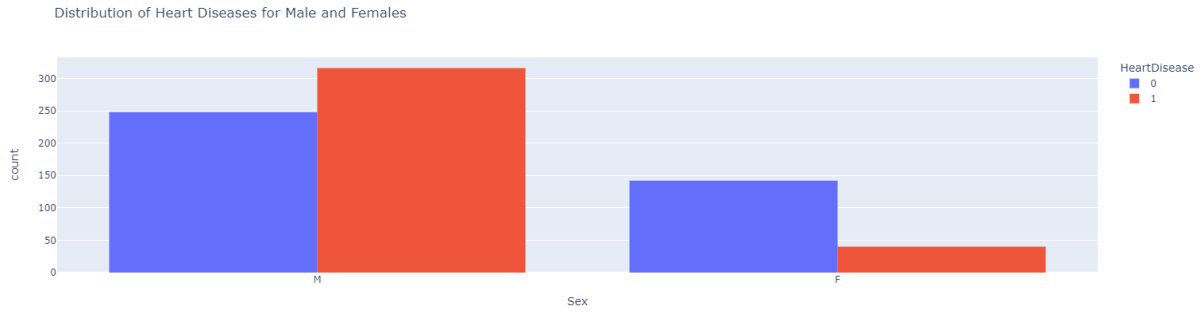

Figure 13 heart disease for males and females

A very interesting find is that it seems that men are significantly more likely to have heart disease than women. This will likely be a big factor in our model as very few females seem to have heart disease.

Next, we look at what chest pain symptoms each gender experienced.

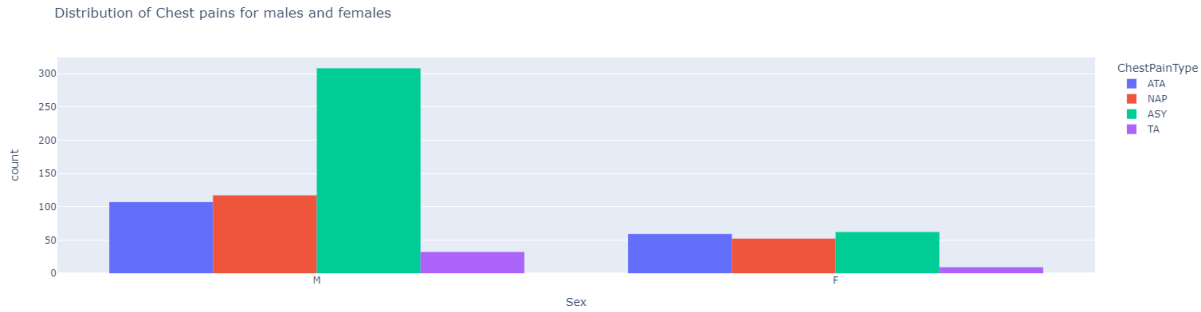

Figure 14 Symptoms for males and females

We see that men seem more likely to have asymptomatic chest pains. This is strange when you think about heart disease in general as you would expect it to be painful. We can output the symptoms for people with and without heart disease.

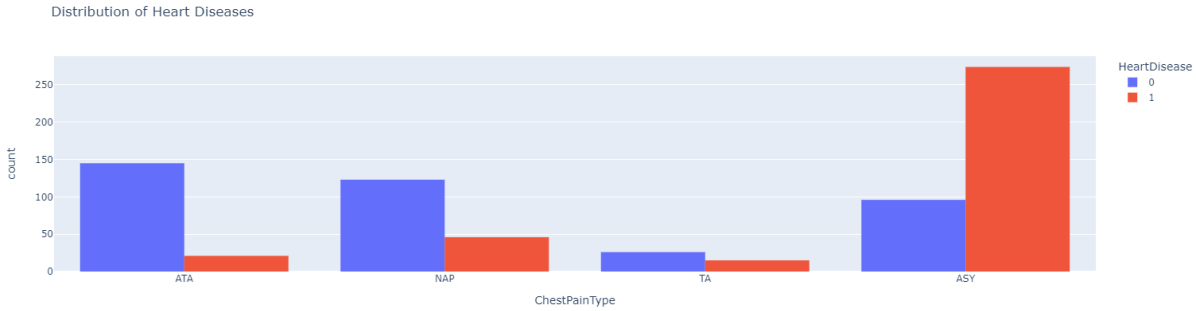

Figure 15 Chest pains for heart disease

We see that the vast majority of people with heart disease experience no chest pain. It now makes sense why the majority of males were asymptomatic as the majority of males were the ones with heart disease.

We now look at the [ExerciseAngina] variable to hopefully get some clues as to what is happening.

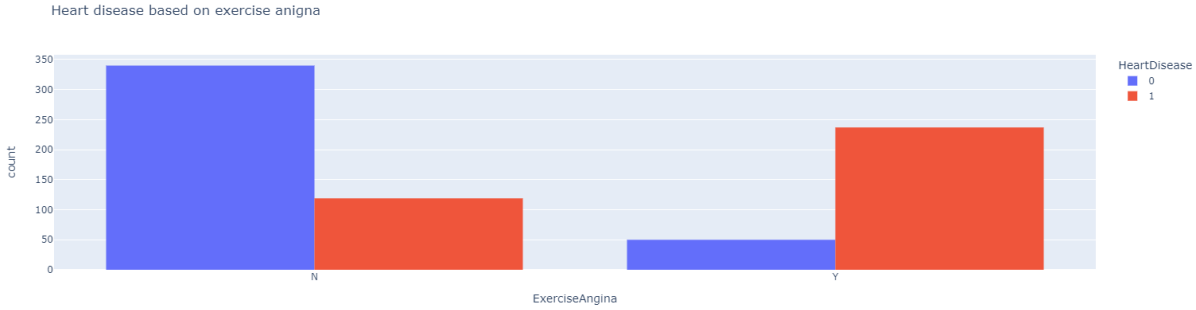

Figure 16 heart disease and exercise angina

We see that the majority of people who experience pain from exercising had heart disease, while those that didn't experience exercise angina might not have been exercising in the first place. This means that **unusual pains while exercising may be a good indicator of heart disease**.

With this information we can answer the questions of where the hospitals got this data, and why are there so many male records compared to female records. The hospitals likely got this data when men went to the gym and performed high-intensity workouts which would aggravate their hearts and result in pain. They would then go to a doctor who would add them to the records. Women are less likely to do such high-intensity workouts and are more resistant to developing heart disease so they probably don't get the signs as often as men would.

There are still a few more variables to analyse so we can move on to looking at the ages of people with and without heart disease. We can also check to see if being a male or a female matters here.

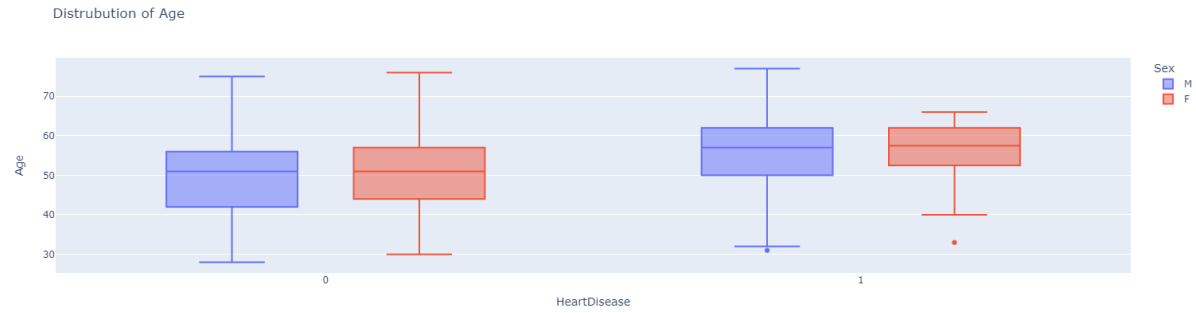

Figure 17 Box and whisker of age, gender, and heart disease

The first thing to note is that being male or female doesn't seem to matter here. But we do see an obvious increase in average ages for people with heart disease. I conclude from this that the older you get the more likely you are to develop heart disease. The problem with this statement is that the data is normally distributed instead of following a constant upward curve. This means there are fewer people with heart disease at age 70 than at age 55. I wouldn't go as far as to say this means that you have less chance of getting heart disease at age 70 compared to age 55. The lack of people in the very high age range is likely due to there being fewer people of that age.

Next, we can look at the [MaxHR] variable.

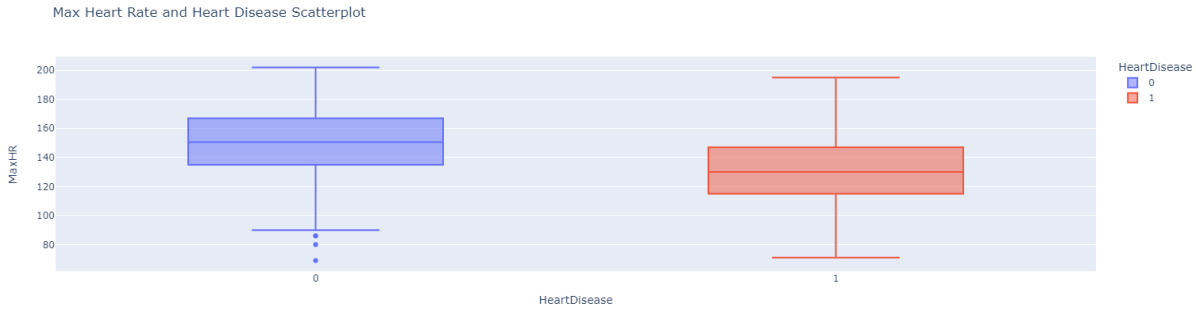

Figure 18 Max heart rate and heart disease

Simply put, we see that people with a higher max heart rate have less chance of getting heart disease. This isn't surprising as we already established that as your age goes up your maximum heart rate achieved goes down in figure 10. This graph further justifies our claim regarding age and heart disease.

Next, we can look at the [Oldpeak] variable.

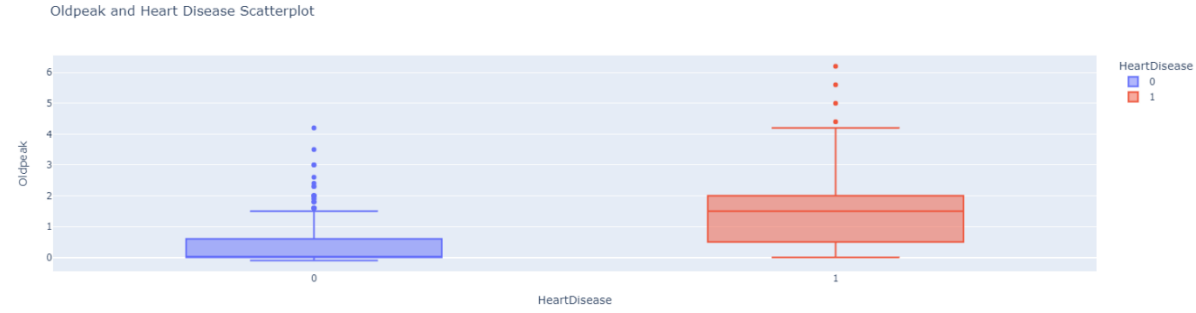

Figure 19 Oldpeak and heart disease

Interestingly enough we see that if your ST depression induced by exercise relative to rest is 0 then you have a low chance of having heart disease, but if the value is greater than 0 you have a high chance of having heart disease. This evidence is pretty strong and already we know that this variable will play a big role in predicting the target.

Lastly, let's look at the slopes.

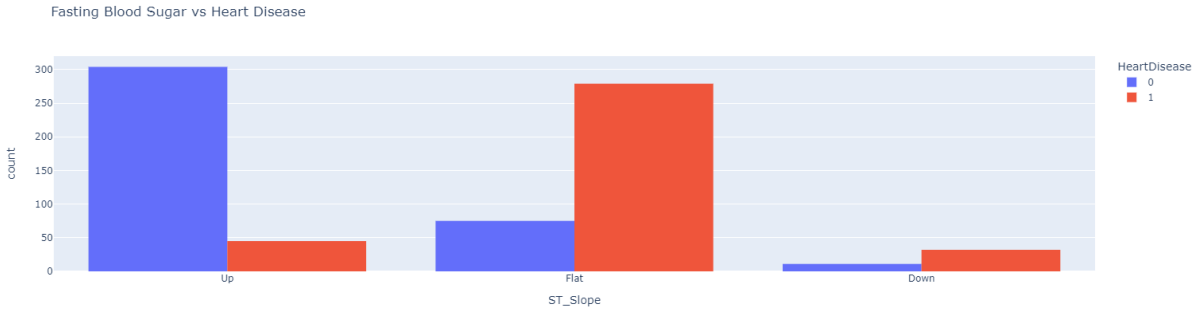

Figure 20 Slopes and heart disease

If your [ST_Slope] is going up, then you have a dramatically reduced chance of having heart disease, but if it's flat or down then the opposite is true. This will very likely play a big role in predicting our target.

3.3.2 Conclusion

The first major conclusion we came to is the fact that these records were likely sourced from people doing intense training and then going to the doctor since it caused unusual pains.

Regarding the question of "what is happing" that descriptive analysis naturally asks we can conclude that:

- Men are more likely to develop heart disease.
- Exercise is a good indicator of heart disease.
- Heart disease doesn't passively cause chest pains.
- The older you get the more likely you are to experience heart disease.
- An Oldpeak of > 0 drastically increases your chance of having heart disease.
- As your max heart rate achieved goes up so do your chances of not having heart disease.
- A flat ST slope means you likely have heart disease.
- A down ST slope could mean you have heart disease.
- An up ST slope means you likely don't have heart disease.

4 Model Creation

A model will be created that takes the wisdom gained from the descriptive analysis and then applies predictive analysis to output a prediction. The below steps summarize the process taken to create the model. For more detail on how the model was created, please refer to the Jupyter Notebook file provided.

4.1 Libraries

Below is a list of libraries that were used during the creation and visualisation of the model.

Library	Use	Example Code
Pandas	Allows the importing of a CSV file into a variable. Also allows for the processing and displaying of datasets.	import pandas as pd
Seaborn	Allows for the plotting of scatterplots and other types of graphical outputs.	import seaborn as sns
Matplotlib.pyplot	Allows for advanced processing of data into graphical outputs.	import matplotlib.pyplot as plt
Ipython. display (Image)	Allows us to display an inline image that's loaded from a file.	from IPython.display import Image
NumPy	Allows you to store data in a NumPy array which acts like a grid to store columns and rows.	import numpy as np
Scikit Learn	Comes with many methods to fit your models, as well as allows us to create training and test data from our data set. Most importantly it will allow us to import the method which will fit our linear regression model.	from sklearn.linear_model import LinearRegression
missingno	Allows us to visualise null values within a dataset.	import missingno as msno

4.2 Performing Predictive Analysis

The below section will discuss the process for performing predictive analysis. The below examples are outputs from the Random Forest regression algorithm. Other algorithms were used and compared. For more detail into how those algorithms were utilised please refer to the Jupyter Notebook file provided.

4.2.1 Separating the values

We need to isolate a X variable that holds all the important features and the Y variable that holds the target feature. The descriptive analysis showed that not all the variables are important, like the [FastingBS] variable. But even if its contribution is extremely minor, it will still contribute a tiny amount to the overall performance. In some cases, this might not be a

big deal, and the hassle of having to constantly input an extra variable may not be worth it. However, this is a model that is designed to save lives, so an extra 0.01% accuracy from these variables may results in thousands of lives being saved. We are going to use all the other variables, regardless of how important, for our X value. Our Y value will hold the [HeartDisease] feature.

4.2.2 Splitting the data

We can't just build a model and hope it's accurate, so we need to split the values into training and testing data. To do this we use the train_test_split method from sklearn to split the data into 80% training data and 20% testing data.

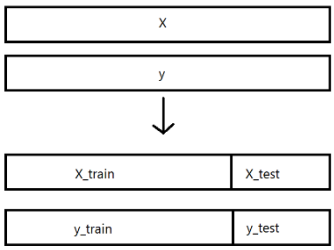

Figure 21 Splitting data

We take the X value and split it into two variables, the X_train variable which holds 80% of the previous X records and the X_test variable which holds the remaining 20%. The same is done for the Y variable. It's important to note that we will only be working with the training variables when fitting our models. The testing variables will only be used to test the accuracy of our model. If we use the training variables to train the model then the model's accuracy becomes quite unreliable, even if it's high. This is because the model has seen this testing data before and know how to handle it.

4.2.3 Verifying the data

To verify the data, we are going to use the concept of the k-folds validation. The K-folds validation breaks the dataset into k number of folds. It can then iterate k times through the data, each time retraining the model and using 1 of those folds to test it. This eliminates the worry that you simply got lucky with your output accuracy. It's also important to note that we will not be using the testing data we created in the previous step. That will be used to test the accuracy later on. The k-folds technique is just a way to validate that the data isn't too random in its accuracy outputs. You can average up the results from all the iterations.

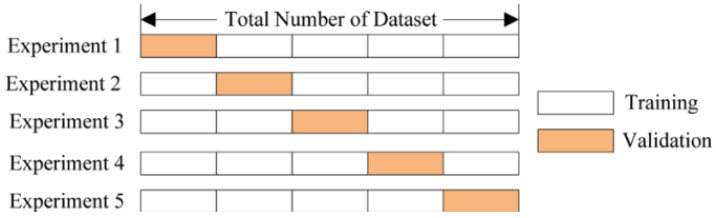

Figure 22 k-folds by (Koehrsen, 2018)

We will not manually write out the iteration logic for this, we will instead use the cross_val_score function from the sci-kit learn library which does the same thing automatically for you. As a bonus the cross_val_score library also shuffles your data before you run it, this

allows you to run the test a few times to further verify the accuracy. It also means that your data won't be affected by where in the record it was originally.

4.2.4 Fitting the Random Forest Classifier Algorithm

The random forest classification algorithm works by gathering an ensemble of decision trees and allowing them to vote on what they think the outcome should be (Pal, 2005). This works off the concept that the majority vote is more valuable than a single vote.

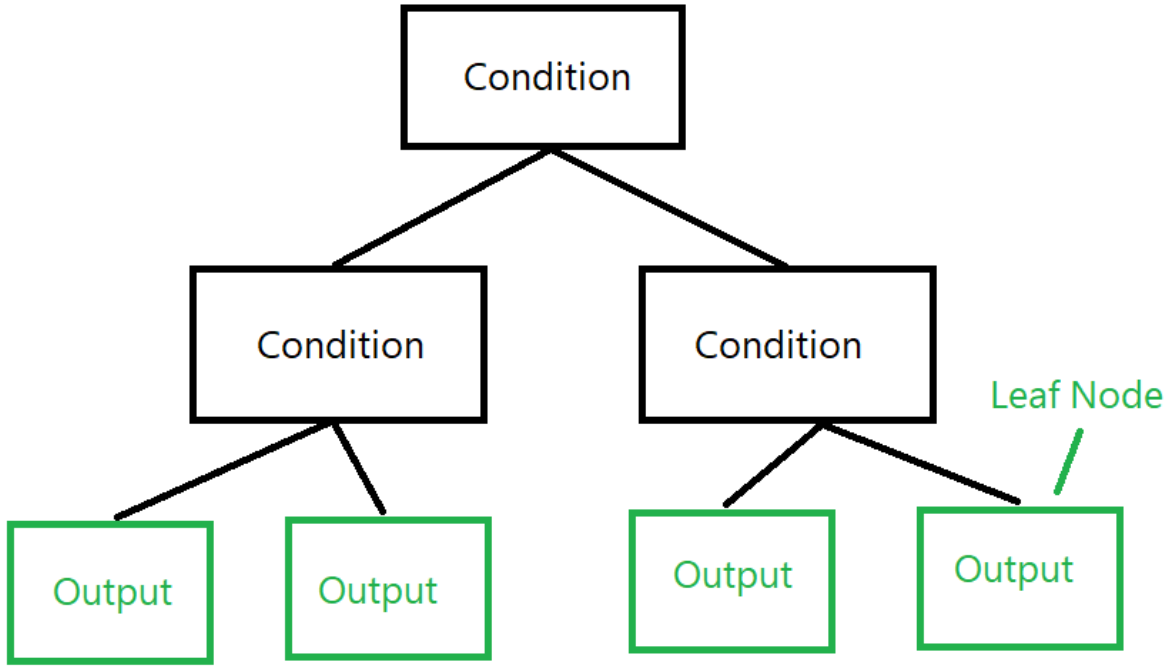

Figure 23 Decision Tree

Decision trees work by moving the input down a hierarchy of nodes, constantly making decisions which choose the path the input will take until it reaches a leaf node, and an output is concluded. The random forest classification algorithm works by making a few of these trees and taking the majority vote as the sole answer to the problem.

The first thing we do is create a pipeline which will take the data and scale it down into a number between 0 and 1. This helps the machine read the data as the points won't be so far apart. And an important thing to note about this is that the overall distribution of the data does not change.

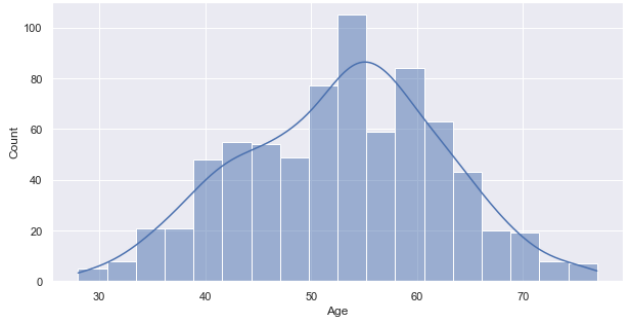

Figure 25 Before scaling

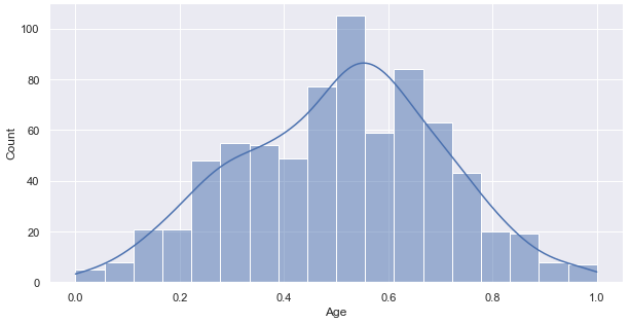

Figure 24 After scaling

The next part of the pipe will be the model itself. The pipe will take any input data, scale it and then feed that scaled data into the model. Once we fit the model through the pipe, we can create predictions from it. We then take those predictions and compare them to the actual results we were expecting. We can visualise this with a confusion matrix and output a classification report.

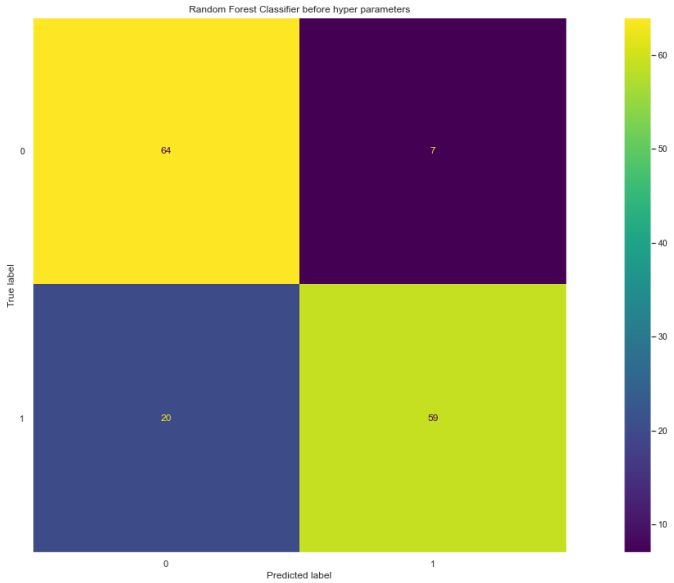

Figure 26 random forest confusion matrix before hyper parameters

We see that the model did quite well with the confusion matrix. It guessed 64 true negatives and 59 true positives. But very concerningly it got 20 false negatives. Those could be 20 people who came in and where told they didn't have heart disease when they did. Let's check the classification report and see where if anything stands out.

	precision	recall	f1-score	support
0	0.76	0.90	0.83	71
1	0.89	0.75	0.81	79
accuracy			0.82	150
macro avg	0.83	0.82	0.82	150
weighted avg	0.83	0.82	0.82	150

Figure 27 random forest classification report before hyper parameters

The precision describes the ratio of how many positive predictions were predicted correctly to the total predicted positive observations (Exsilio Solutions, 2016). The equation can be expressed as

$$\frac{[True\ Positives]}{[True\ Positives]+[False\ Positives]} = \frac{64}{64+20} = 0.76\ or\ \frac{59}{59+7} = 0.89$$

The recall is the ratio between the correctly predicted positives and all the observations in that class (Exsilio Solutions, 2016). It will be expressed as

$$\frac{[True\ Positives]}{[True\ Positives]+[False\ Negative]} = \frac{64}{64+7} = 0.9\ or\ \frac{59}{59+7} = 0.75$$

The F1 score is the weighted average between the two (Exsilio Solutions, 2016). It can be expressed as:

$$\frac{2([Recall] * [Precision])}{[Recall] + [Precision]} = \frac{2(0.9 * 0.76)}{0.9 + 0.76} = 0.82 \; or \; \frac{2(0.75 * 0.89)}{0.75 + 0.89} = 0.81$$

We see that the classification report did quite well. All the scores where alright, but there were a lot of variations in the scores with some recalls being low and some precisions being low. But the average accuracy was decent and above 80%.

We can further verify the data by running the cross_val_score method to see if the folds were consistent.

```
[0.83333333 0.83193277 0.84033613 0.70588235 0.83193277]
The average score was 0.8086834733893558
```

Figure 28 Cross Val Score for random forest before hyper parameters

We see that the cross-validation score was fairly consistent. This means that the data should be reliable.

However, when referring to how serious of a topic heart disease is, any extra percentage we can squeeze out of the accuracy score could results in hundreds or thousands of lives being saved. We can look to improve the model.

4.2.5 Improving the Random Forest Classifier model

To improve the model, we will look at improving the parameters we pass to it. We will be making use of a grid search which will break the data into folds similar to the f-folds method and then test each fold against a list of parameters you supply it. For more information on what parameters where used and how they were implemented, please refer to the Jupyter Notebook file provided.

After running the grid search on our data, we get these parameters.

```
n_estimators: 80
min_samples_split: 2
min_samples_leaf: 1
max_features: auto
max_depth: 4
bootstrap: False
```

Figure 29 Hyper parameters random forest classification

We can now take these parameters and plug them into the model. Let's compare the cross-validation score.

```
[0.85       0.85714286 0.86554622 0.82352941 0.86554622]
0.8523529411764705
```

Figure 30 Cross validation score

We see that the cross-validation score is performing significantly better with an average increase of almost 5%. That is many more lived saved. Now we can introduce the testing data we split at the start and test it against our new model with hyper parameters.

	precision	recall	f1-score	support
0	0.93	0.93	0.93	71
1	0.94	0.94	0.94	79
accuracy			0.93	150
macro avg	0.93	0.93	0.93	150
weighted avg	0.93	0.93	0.93	150

Figure 32 Classification report for random forest with hyper parameters

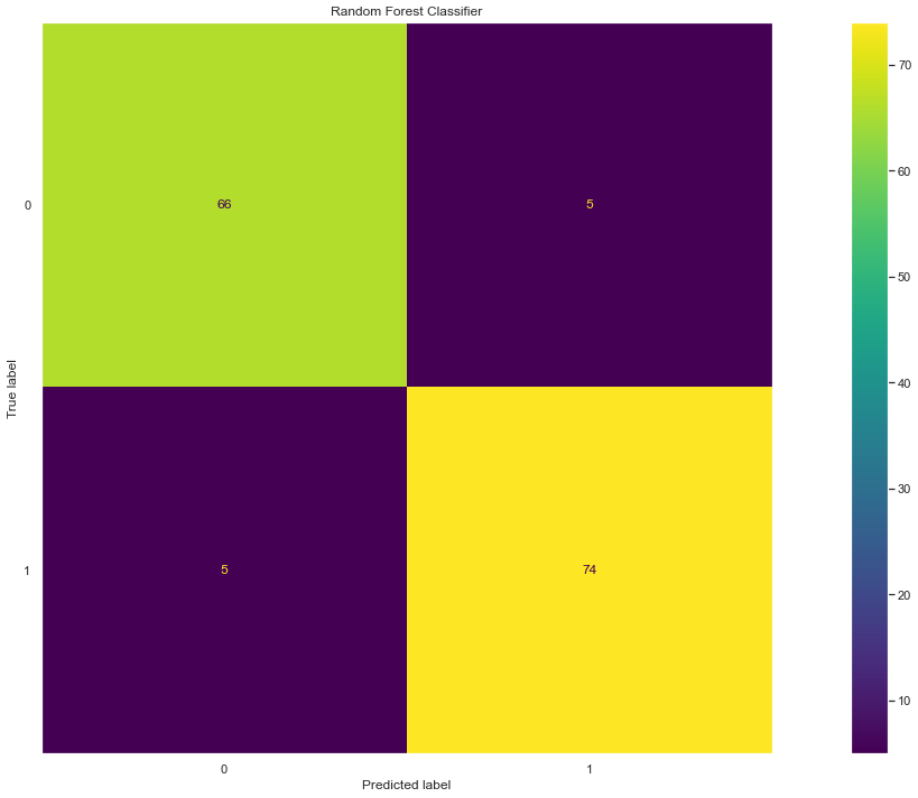

Figure 31 Confusion matrix random forest with hyper parameters

The model is performing significantly better than before with an accuracy of 93% and only 5 false negatives.

One final output we can get from the random forest classification model is once it is fit, you can create a graph that shows how important each feature is to the algorithm.

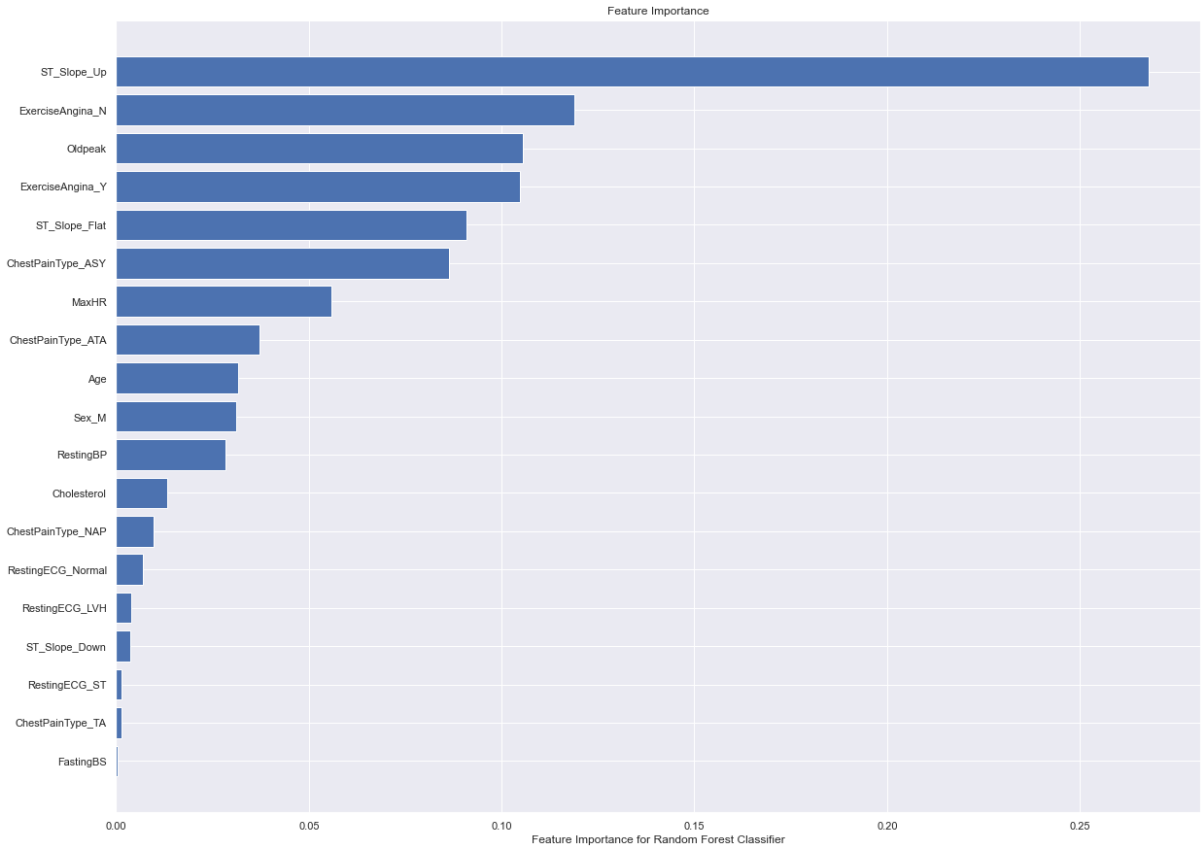

Figure 33 Feature importance

From looking at the feature important we know which variables to prioritize in the future. We also see that our descriptive analysis was pretty much spot on. Nothing really surprising except for how unimportant [ST_Slope_Down] is. This is probably due to there simply being not enough records of it to contribute anything substantial to the algorithm. We also see that [ST_Slope_Up] is incredibly important. So important that even without the model it should be considered a good indicator that you don't have heart disease if you experience it.

4.2.6 Fitting and improving the Naïve Bayes Classification algorithm

The naïve bayes classification algorithm is a probability-based algorithm that works under the assumption that none of the features are relevant to each other. This is where the term naïve comes into play (Sunil, 2017). It will try to classify each feature independently and each one of those independent classifications will add towards the probability that you have heart disease or not.

The first step is to create the pipe and fit the training data to the cross-validation score function.

```
[0.61666667 0.54621849 0.58823529 0.6302521  0.6302521 ]
The average score was 0.6023249299719888
```

Figure 34 NB cross validation before hyper parameters

We see that the algorithm is performing quite poorly compared to the Random Forest classification model. But it is performing consistently poorly. There are no random spikes in the accuracy, which means that there is hope for hyper parameters to drastically increase the accuracy.

Let's look at the classification report and confusion matrix first so we get an idea of how well the algorithm improves after hyper parameters.

	precision	recall	f1-score	support
0	0.52	1.00	0.69	71
1	1.00	0.18	0.30	79
accuracy			0.57	150
macro avg	0.76	0.59	0.49	150
weighted avg	0.77	0.57	0.48	150

Figure 35 NB classification report before hyper parameters

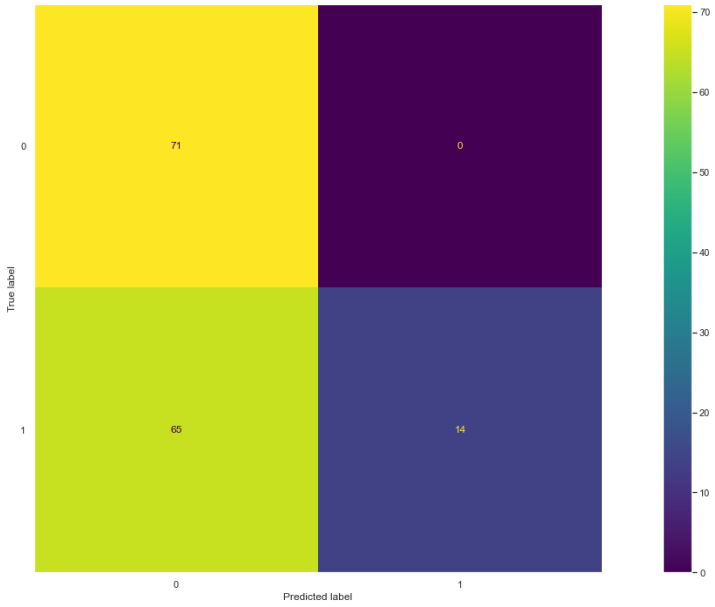

Figure 36 NB confusion matrix before hyper parameters

The accuracy score is completely misleading as the algorithm is performing horribly with 65 cases of it failing to predict positive heart disease. It was good as not producing a false positive, because it almost never produced a positive to begin with.

This would normally indicate that there aren't enough records with [HeartDisease] set to 1. If this was the case, we could use SMOTE to generate synthetic variables to compensate. However, we know from the descriptive analysis that there are more than enough records with

positive heart disease. The next thing we will look at is optimizing the parameters to see if that yields better results.

```
[0.84166667 0.8487395  0.86554622 0.80672269 0.84033613]
0.8406022408963585
```

Figure 37 NB cross validation score after hyper parameters

We see that by adding hyper parameters we are able to more than significantly increase the average accuracy of the model when put through the cross-validation method. Now we can introduce our testing data and see how well the full model performs.

	precision	recall	f1-score	support
0	0.85	0.87	0.86	71
1	0.88	0.86	0.87	79
accuracy			0.87	150
macro avg	0.87	0.87	0.87	150
weighted avg	0.87	0.87	0.87	150

Figure 38 Classification report for NB after hyper parameters

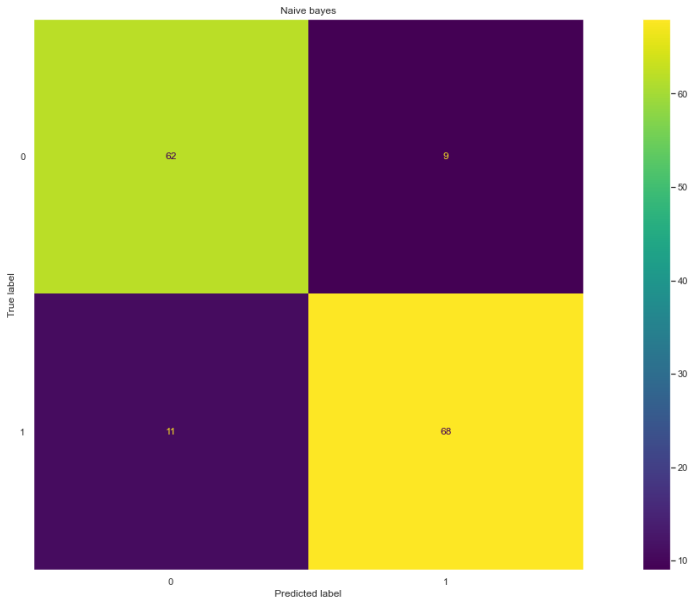

Figure 39 Confusion Matrix for NB after hyper parameters

The model is performing much better than it was before. But with 11 missed heart diseases, the model does not perform as well as the Random Forest model does.

4.2.7 Overfitting and Underfitting considerations

Overfitting occurs when a model is fit too closely to the training data. By doing this you prevent the model from being able to predict accurately as it was affected too severely by the noise in the data set. A model that suffers from underfitting cannot make accurate predictions as it simply cannot capture the trend of the data (Muller & Guido, 2017).

4.2.7.1 Overfitting

It can be quite difficult to avoid overfitting the data if the data you have already has too much bias. The naïve bayes algorithm should suffer too much with overfitting as it doesn't follow the trend of the data regardless. It makes assumptions based on each feature independently. If you do have overfitting in the data, then you can try reducing or increasing the smoothing variable in the parameters.

There is also a trick to preventing overfitting with the random forest and decision tree algorithms. You can set a maximum depth of the tree. By doing this you force the algorithm to make more drastic decisions instead of taking it's time to follow the data closely.

4.2.7.2 Underfitting

The biggest contributor to underfitting is having bad data to begin with. There needs to be enough records to train the data with and there needs to be a fair relationship between the features you are using.

If you don't have enough data, then you either need to collect more or you need to increase your sample size. It's also possible that the data gets underfit if there aren't enough records of one of the two possible target outcomes. In this case the data could be underfitted if there were not enough people with heart disease in the dataset. If this is the case, then you can create synthetic variables with SMOTE.

If the data is unclean, then you can also find yourself underfitting the data. What's worse is that if you drop all the nulls, you may lose too many records and still underfit. It might be beneficial to interpolate the data instead of just dropping it.

Bad parameters can also cause underfitting, such as having your smoothing too high in the naïve bayes classification algorithm or having a max depth that is too low in the random forest classification algorithm.

4.2.7.3 Testing for overfitting or underfitting

It's quite easy to test if your data is overfit or underfit. The key is to split your dataset into training and testing data. You then keep the testing data away from the model while you are testing it so that the testing data does not impact the model at all. By doing this you can use the testing data to act as 'real life data that just got added'. This will show how well your model performs in a real-life situation where it needs to predict on data it's never seen before.

5 Conclusion

The descriptive analysis brought to light many interesting things that may not have been obvious at first. Such as heart disease causing asymptomatic chest pains and an upwards ST slope being a very likely indicator that you're save from it. Men also seem significantly more likely to develop heart disease and also more likely to spot it as exercising can be a good tell-tale sign. The older you get the more worried you should be about heart disease and an Oldpeak of over 0 should require immediate attention.

We find that the predictive analysis produced two great models. Both the random forest classification and naïve bayes algorithms work quite well with the dataset that was provided with an accuracy well over 80%. However, the random forest goes above and beyond the naïve bayes by often producing an accuracy score of over 90%.

6 Recommendations

These findings are quite solid in how well they perform. With the help of the feature importance graph and the descriptive analysis, people in general could be educated about these symptoms in an attempt to save a few extra lives. Not all these symptoms are common knowledge to everyone, or some are communicated in a way that could mislead people.

For example, if someone types exercise angina into Google, they are presented with this suggestion:

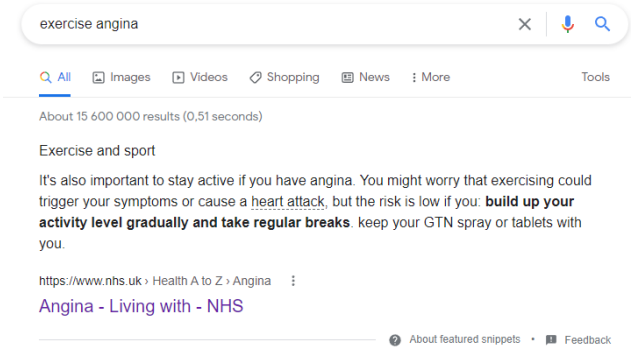

Figure 40 exercise angina by google.com

The first thing you read is that you should keep exercising. It doesn't recommend going to the doctor at all. If someone experiences this pain and looks it up, they might believe that they just need to keep exercising to make the pain go away. This quick glance makes it look like their muscles were just too weak. It should rather heavily suggest going to see a doctor as their highest priority because they may be experiencing early signs of heart disease, which if left untreated could result in a heart attack and potentially their funeral. My recommendation is that google should update the results to specify this immediately, the same way they recommend a help service if it picks up that you're suicidal.

Another suggestion is for huge phone companies like Apple or Google to build their future models with scanning functionality that allows you to input many of the required features above to give you a fairly accurate warning on your heart's health. This could be a great business

move implement this as it would be quite attractive to people of older ages, especially if these symptoms become common knowledge among all households. You don't even need all the features. You just need a scanner to measure ST slope, you could input if you had pain while exercising, or who knows, maybe the phone could figure it out based on stress levels or something. Oldpeak could be hard to implement but not impossible. Age is an easy input, your Max heart rate can be measured, it can ask if you have chest pain passively, your biological sex is an easy input, but it will have to be specified that it's not the gender you identify with, it will have to be either xx or xy. Those make up the most important variables to get a fairly accurate idea of the persons health. Your fasting blood sugar levels, cholesterol, and resting ECG aren't that important to the model.

My last suggestion is to not only use one of the two mentioned classification methods. To increase the accuracy of the prediction the outputs of all tested classification methods that we test that scored above 85% accuracy. We can then take the majority vote among them. Similar to how the random forest algorithm works.

7 References

Akoglu, H., 2018. User's guide to correlation coefficients. *Turkish journal of emergency medicine,* 18(3), pp. 91-93.

American Heart Association, 2021. *2021 Heart Disease and Stroke Statistics,* United States: Americaon Heart Association.

Cote, C., 2021. *Predictive analytics.* [Online]
Available at: https://online.hbs.edu/blog/post/predictive-analytics
[Accessed 22 April 2022].

Exsilio Solutions, 2016. *Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures.* [Online]
Available at: https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/
[Accessed 4 June 2022].

Fedesoriano, 2021. *Heart Failure Prediction Dataset.* [Online]
Available at: https://www.kaggle.com/datasets/fedesoriano/heart-failure-prediction
[Accessed 01 June 2022].

Koehrsen, W., 2018. Overfitting vs. underfitting: A complete example. *Towards Data Science.*

Muller, A. & Guido, S., 2017. *Introduction to Machine Learning with Python.* 1st ed. United States of America: O'Reilly.

Pal, M., 2005. Random forest classifier for remote sensing classification.. *International journal of remote sensing,* 26(1), pp. 217-222.

Rossignol, P., Hernandez, A., Solomon, S. & Zannad, F., 2019. Heart failure drug treatment. *The Lancet,* 393(10175), pp. 1034-1044.

Sunil, R., 2017. *6 Easy Steps to Learn Naive Bayes Algorithm with codes in Python and R.* [Online]
Available at: https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/#:~:text=Naive%20Bayes%20Model-,What%20is%20Naive%20Bayes%20algorithm%3F,presence%20of%20any%20other%20feature.
[Accessed 05 June 2022].

The Independent Institute of Education, 2022. *Data Analytics Module Manual ,* s.l.: s.n.