Data Analytics

Alex Andrew

B00756527

BEng Computing Technology

CW1 Assignment

Table of Contents

Introduction

The dataset which the developer has chosen for my project is called the Heart Disease Health Indicators dataset and is found on Kaggle.com. This dataset is one that has been derived from a much larger dataset called the Behavioural Risk Factor Surveillance System, which has over 400 columns. This dataset contains information about thousands of patients and their health conditions and health behaviours. The Heart Disease dataset creator ran a series of queries on the Behavioural Risk System records for 2015 and collected a series of 22 variables on heart disease. This is still a large dataset, even after the creators cleaning, as it has over 250000 records.

The developer will analyse whether an individual is at risk of heart disease, based on various factors in the dataset. An example of some of these factors will be their Body Mass Index (BMI), their diabetic status, if they smoke, drink and exercise, among many other factors.

Full Feature Details	Feature Name	Feature Categories	Type
Heart Disease or Attack	HeartDiseaseorAttack	n/a	Nominal
High Blood Pressure	HighBP	n/a	Nominal
High Cholesterol Level	HighChol	n/a	Nominal
Cholesterol Check within past 5 years	CholCheck	n/a	Nominal
Body Mass Index	BMI	n/a	Continuous
Smoked over 100 Cigarettes	Smoker	n/a	Nominal
Had a stroke	Stroke	n/a	Nominal
Have Diabetes or Pre-Diabetes	Diabetes	2-Diabetes 1-Pre-Diabetes 0-None	Ordinal
Exercise done in the last 30 days	PhysActivity	n/a	Nominal
Consume Fruit more than once a day	Fruits	n/a	Nominal
Consume Vegetables more than once a day	Veggies	n/a	Nominal
Heavy Drinking (men - 14 drinks, women – 7)	HvyAlcoholConsump	n/a	Nominal
Any Healthcare Coverage	AnyHealthcare	n/a	Nominal
Last year, avoided healthcare because of cost	NoDocbcCost	n/a	Nominal
General state of health	GenHlth	5-Excellent 4-Good 3-Ok 2-Poor 1-Very Poor	Ordinal
In last 30 days, how many days with bad mental health	MentHlth	n/a	Discrete

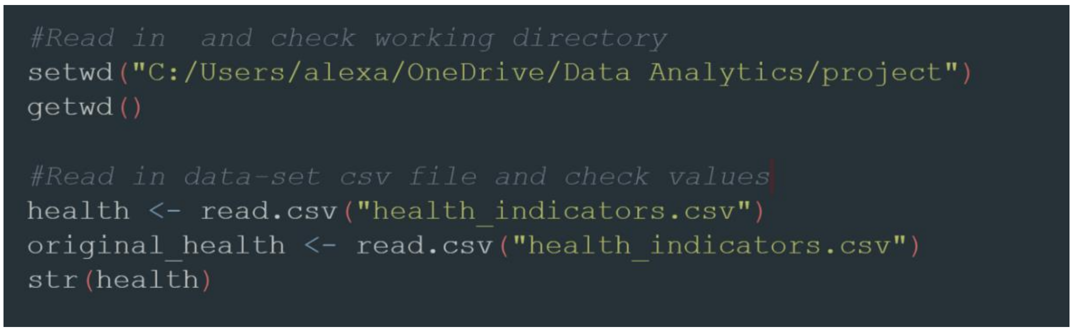
In last 30 days, how many days with bad physical health	PhysHlth	n/a	Discrete
Difficulty Walking	DiffWalk	n/a	Nominal
Sex	Sex	Male Female	Nominal
Age	Age	1- "18-24" 2- "25-29" 3- "30-34" 4- "35-39" 5- "40-44" 6- "45-49" 7- "50-54" 8- "55-59" 9- "60-64" 10- "65-69" 11- "70-74" 12- "75-79" 13- "80+"	Ordinal
Stage of Education	Education	1 2 3 4 5 6	Ordinal
Level of income	Income	1- "<$10000" 2- ">=$10000" 3- ">=$25000" 4- ">=$35000" 5- ">=$45000" 6- ">=$55000" 7- ">=$65000" 8- ">=$75000"	Ordinal

Methods

Data Cleaning

First, the developer will discuss how the data was cleaned and prepared for analysis. Data modelling techniques, as well as the different features, will be discussed.

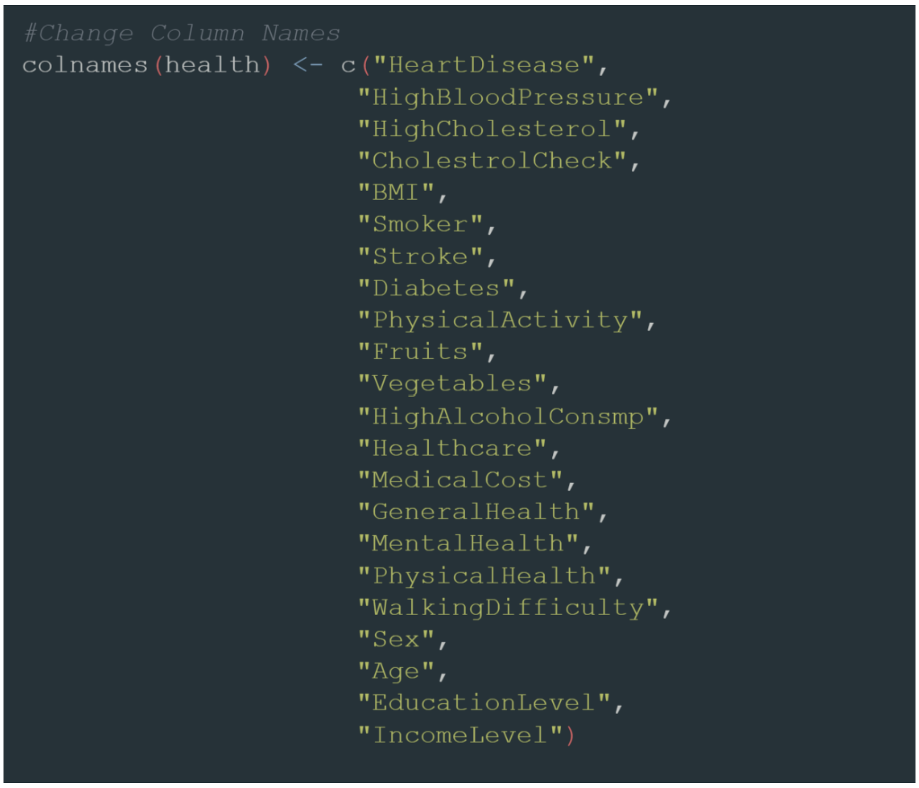
First, the developer began by cleaning the dataset. First the dataset is downloaded from the Kaggle webpage and is loaded into R via csv format and is stored in both the health and original_health variables. Original health is not touched, while health is the dataset that will be used.



```r
#Read in  and check working directory
setwd("C:/Users/alexa/OneDrive/Data Analytics/project")
getwd()

#Read in data-set csv file and check values
health <- read.csv("health_indicators.csv")
original_health <- read.csv("health_indicators.csv")
str(health)
```

Figure 1

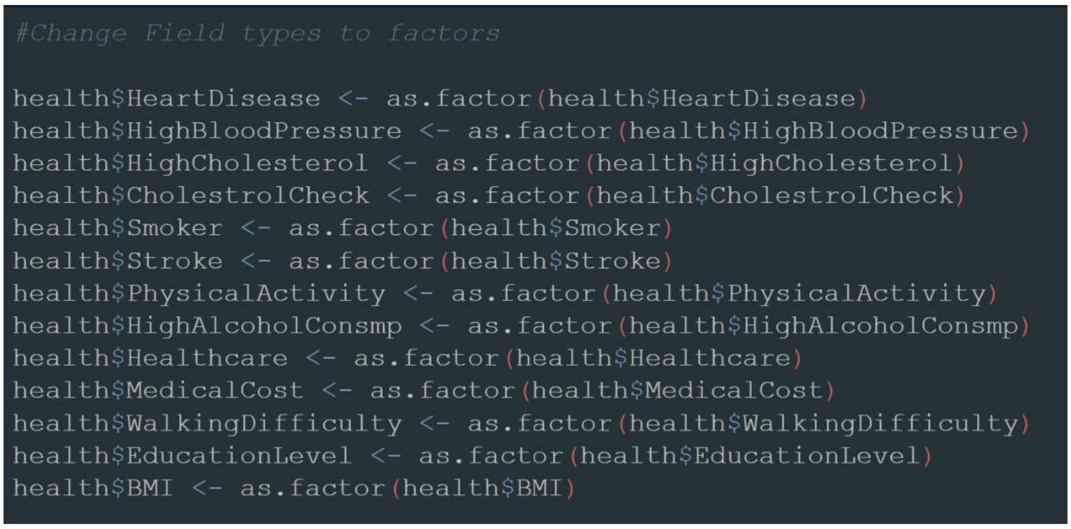
The first cleaning step was to was to change column names of the datasets variables to make then neater and much easier to understand as can be seen in figure 2.



```r
#Change Column Names
colnames(health) <- c("HeartDisease",
                      "HighBloodPressure",
                      "HighCholesterol",
                      "CholestrolCheck",
                      "BMI",
                      "Smoker",
                      "Stroke",
                      "Diabetes",
                      "PhysicalActivity",
                      "Fruits",
                      "Vegetables",
                      "HighAlcoholConsmp",
                      "Healthcare",
                      "MedicalCost",
                      "GeneralHealth",
                      "MentalHealth",
                      "PhysicalHealth",
                      "WalkingDifficulty",
                      "Sex",
                      "Age",
                      "EducationLevel",
                      "IncomeLevel")
```

Figure 2

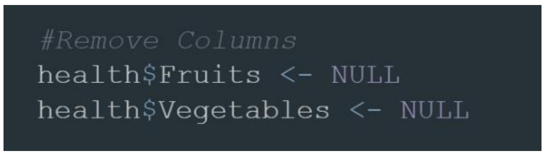
The next step was then to factorise the variables that are nominal as seen in figure 3. This renders the fields useful for later for statistical analysis as they are fixed to their values and the values then won't be changed by the sub sampling techniques and can be used in machine learning algorithms.



```r
#Change Field types to factors

health$HeartDisease <- as.factor(health$HeartDisease)
health$HighBloodPressure <- as.factor(health$HighBloodPressure)
health$HighCholesterol <- as.factor(health$HighCholesterol)
health$CholestrolCheck <- as.factor(health$CholestrolCheck)
health$Smoker <- as.factor(health$Smoker)
health$Stroke <- as.factor(health$Stroke)
health$PhysicalActivity <- as.factor(health$PhysicalActivity)
health$HighAlcoholConsmp <- as.factor(health$HighAlcoholConsmp)
health$Healthcare <- as.factor(health$Healthcare)
health$MedicalCost <- as.factor(health$MedicalCost)
health$WalkingDifficulty <- as.factor(health$WalkingDifficulty)
health$EducationLevel <- as.factor(health$EducationLevel)
health$BMI <- as.factor(health$BMI)
```

Figure 3

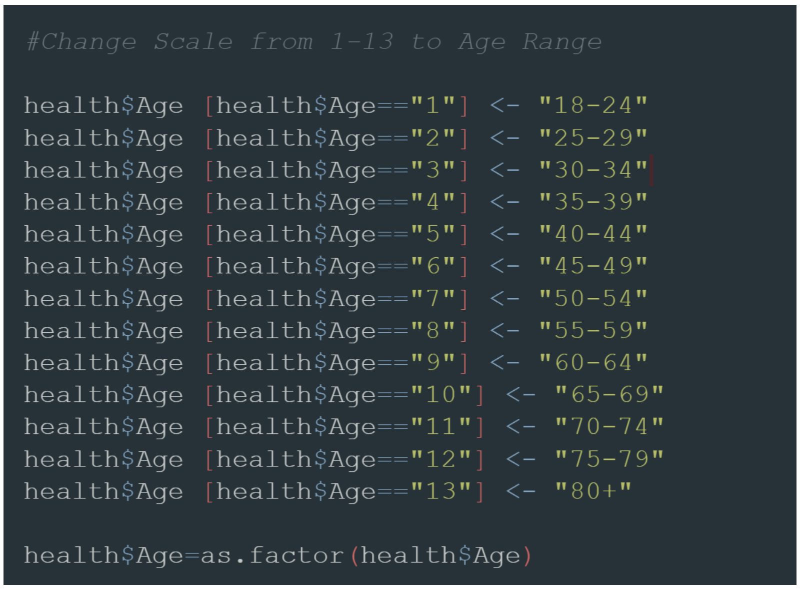
The developer then identified 2 features which were deemed to be unusual to the analysis and were removed from the data as seen in figure 4. The first column is "Fruit", which is a binary class in which the individual was asked if they consumed more than 1 piece of fruit a day. "Vegetable" was the second feature, which was the same as "Fruit" apart from the individual was asked if they consumed more than 1 vegetable.



```r
#Remove Columns
health$Fruits <- NULL
health$Vegetables <- NULL
```

Figure 4

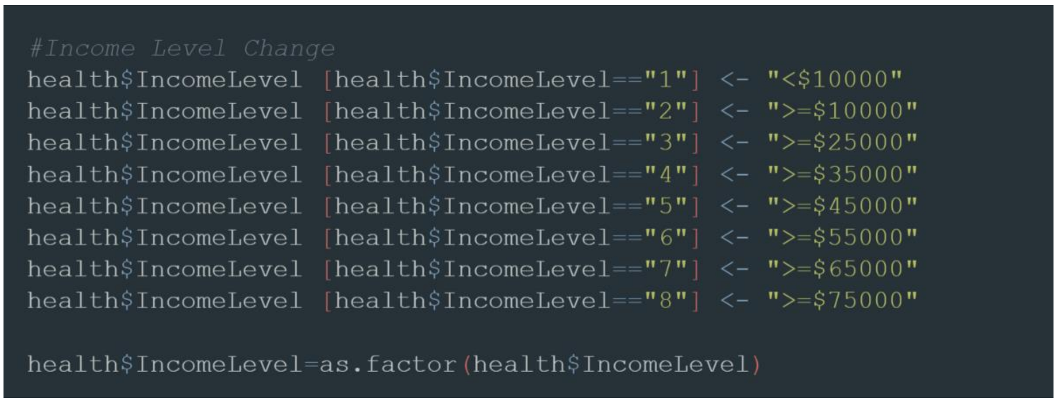
The next section of cleaning is changing the ranges in the ordinal features from a numerical value to a word and factoring these. The reason for changing the numerical value to a word is that it makes the data easier to understand when being observed. The first feature changed was "Age". The age range runs from 1-13 and each of the numbers in this range was changed to a word. This ranged from 18-24 to 80+, and increased in intervals of 5 as shown in figure 5. The range is then factored.



```r
#Change Scale from 1-13 to Age Range

health$Age [health$Age=="1"] <- "18-24"
health$Age [health$Age=="2"] <- "25-29"
health$Age [health$Age=="3"] <- "30-34"
health$Age [health$Age=="4"] <- "35-39"
health$Age [health$Age=="5"] <- "40-44"
health$Age [health$Age=="6"] <- "45-49"
health$Age [health$Age=="7"] <- "50-54"
health$Age [health$Age=="8"] <- "55-59"
health$Age [health$Age=="9"] <- "60-64"
health$Age [health$Age=="10"] <- "65-69"
health$Age [health$Age=="11"] <- "70-74"
health$Age [health$Age=="12"] <- "75-79"
health$Age [health$Age=="13"] <- "80+"

health$Age=as.factor(health$Age)
```

Figure 5

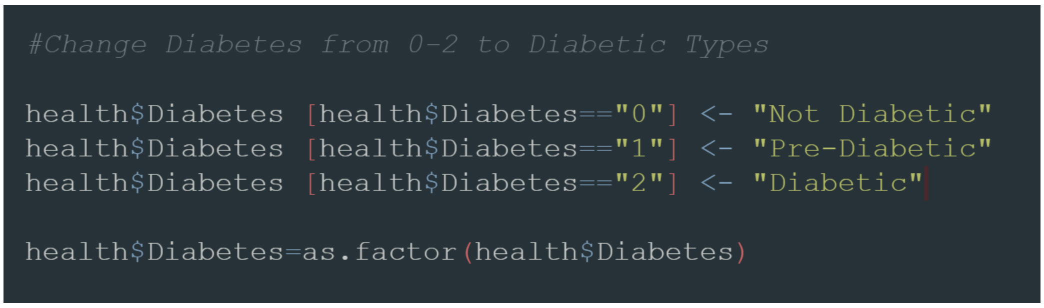
Next changed was “Income”. The income range ran from 1-8, so the range for income went from <$10,000 up to <=$75,000 as shown in figure 6. The incomes are then factored.



```r
#Income Level Change
health$IncomeLevel [health$IncomeLevel=="1"] <- "<$10000"
health$IncomeLevel [health$IncomeLevel=="2"] <- ">=$10000"
health$IncomeLevel [health$IncomeLevel=="3"] <- ">=$25000"
health$IncomeLevel [health$IncomeLevel=="4"] <- ">=$35000"
health$IncomeLevel [health$IncomeLevel=="5"] <- ">=$45000"
health$IncomeLevel [health$IncomeLevel=="6"] <- ">=$55000"
health$IncomeLevel [health$IncomeLevel=="7"] <- ">=$65000"
health$IncomeLevel [health$IncomeLevel=="8"] <- ">=$75000"

health$IncomeLevel=as.factor(health$IncomeLevel)
```

Figure 6

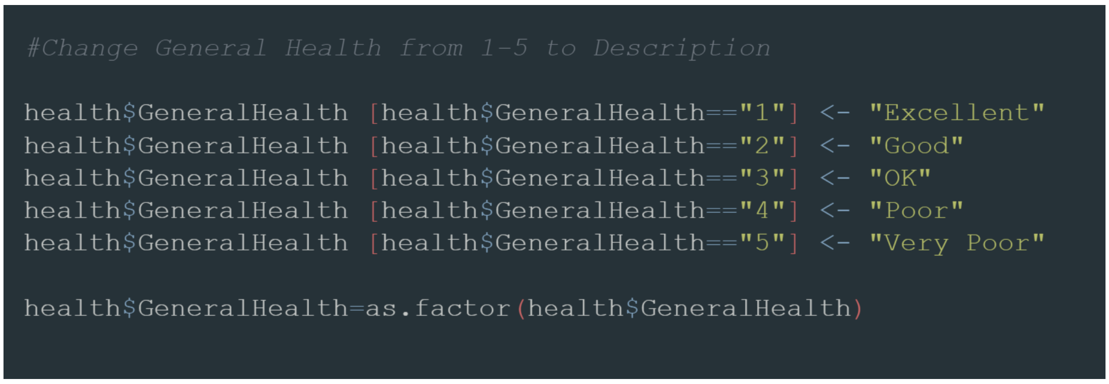
“Diabetes” was changed from a range of 0 to 2 to diabetic status as shown in figure 7. Range then factored



```r
#Change Diabetes from 0-2 to Diabetic Types

health$Diabetes [health$Diabetes=="0"] <- "Not Diabetic"
health$Diabetes [health$Diabetes=="1"] <- "Pre-Diabetic"
health$Diabetes [health$Diabetes=="2"] <- "Diabetic"

health$Diabetes=as.factor(health$Diabetes)
```

Figure 7

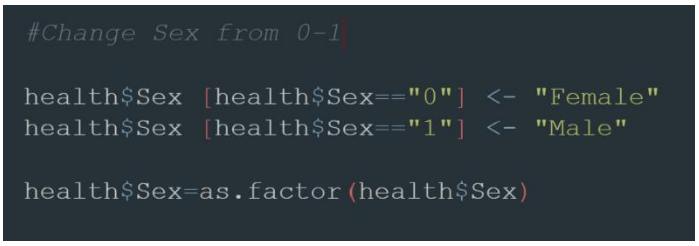
“General Health” was changed from a range of 1-5, to a range of 1 being “Excellent”, 2 being “Good”, 3 being “Ok”, 4 being “Poor” to 5 being very poor as shown in figure 8. The range was then factored.



```r
#Change General Health from 1-5 to Description

health$GeneralHealth [health$GeneralHealth=="1"] <- "Excellent"
health$GeneralHealth [health$GeneralHealth=="2"] <- "Good"
health$GeneralHealth [health$GeneralHealth=="3"] <- "OK"
health$GeneralHealth [health$GeneralHealth=="4"] <- "Poor"
health$GeneralHealth [health$GeneralHealth=="5"] <- "Very Poor"

health$GeneralHealth=as.factor(health$GeneralHealth)
```

Figure 8

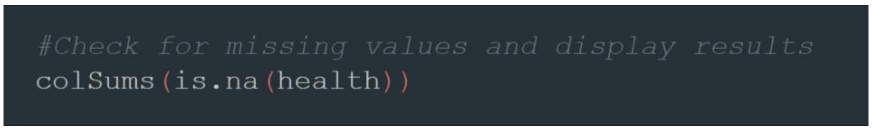
“Sex” was changed from 0 and 1 to Female and Male respectively as shown in figure 9. The range was then factored.



```r
#Change Sex from 0-1

health$Sex [health$Sex=="0"] <- "Female"
health$Sex [health$Sex=="1"] <- "Male"

health$Sex=as.factor(health$Sex)
```

Figure 9

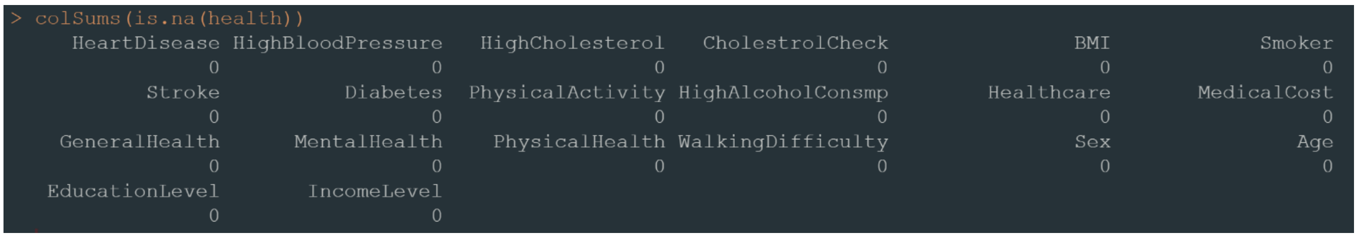
When this Heart Disease Health Indicators dataset was created by extracting features from the larger BRFSS 2015 dataset, the creator of the dataset, removed all NULL values from the dataset. It is important that these are removed so later analysis is successful and accurate. To prove, the data contains no null values, the code in figure 10 is then run.



```r
#Check for missing values and display results
colSums(is.na(health))
```

Figure 10

The results of this will be shown in the console. The results for health are as shown in figure 11. The developer can now proceed with any analysis as the dataset is complete and is not missing any values.

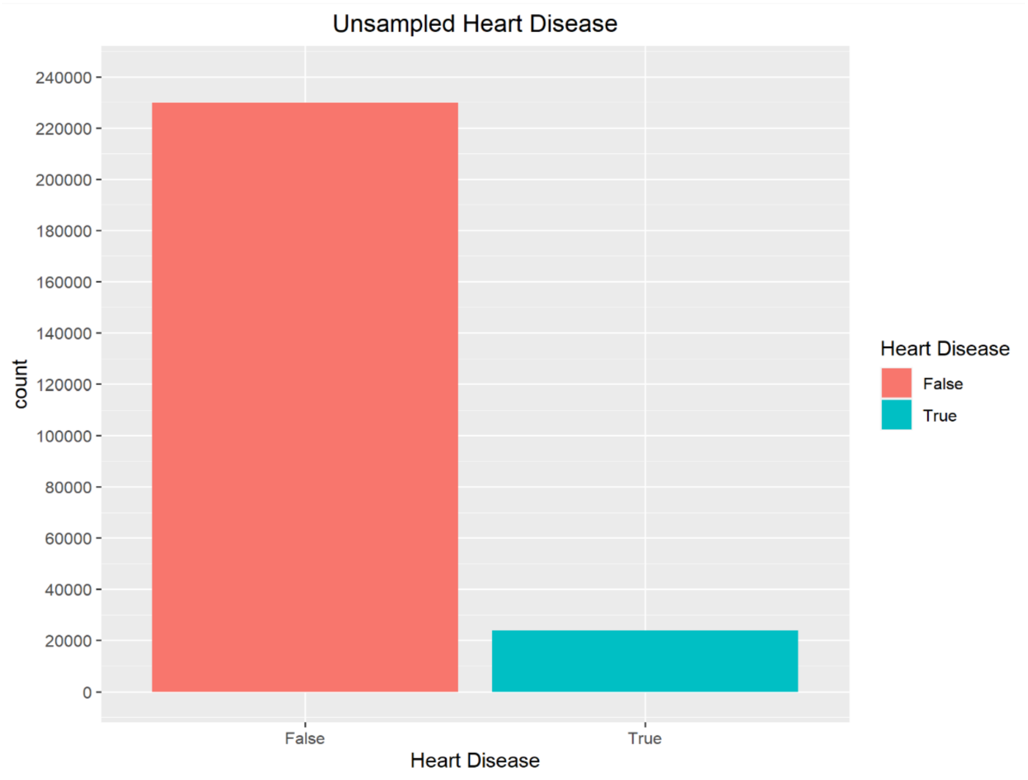


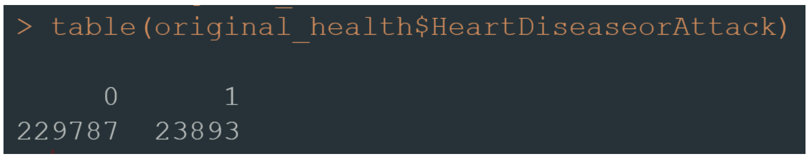
```
> colSums(is.na(health))
     HeartDisease HighBloodPressure    HighCholesterol    CholestrolCheck                BMI             Smoker
                0                 0                  0                  0                  0                  0
           Stroke          Diabetes   PhysicalActivity HighAlcoholConsmp         Healthcare        MedicalCost
                0                 0                  0                  0                  0                  0
    GeneralHealth      MentalHealth     PhysicalHealth WalkingDifficulty                Sex                Age
                0                 0                  0                  0                  0                  0
   EducationLevel       IncomeLevel
                0                 0
```

Figure 11

Data Preparation

The developers first step in data preparation was to balance the target variable split. As shown in figure 12, the numbers of people that had heart disease where only about 10% of the total number of people asked, compared to 90% that did not have heart disease.



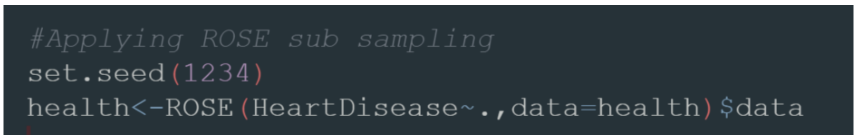


```
> table(original_health$HeartDiseaseorAttack)

     0      1 
229787  23893 
```

Figure 12

Class imbalance will mean that the machine learning, when complete, will lean towards the majority class and give inaccurate results. To solve this, the developer decided to use subsampling, to balance out the minority/majority classes. There are multiple sampling methods, such as down sampling, up sampling or hybrid methods. Up sampling was not chosen as it would balance the classes out at a count of over 450,000 records, which would have a major impact on any machine learning algorithms performance and run times. Down sampling was not chosen as it reduces the classes to a count of 40,000, which will reduce the machine learning accuracies and not produce accurate results. The developer decided to choose ROSE (Random Over-Sampling Examples), which balances the classes out at over 200,000, with around equal in each class. ROSE works well with a binary classification and samples the dataset to an appropriate number of records, reducing records to reduce run times but also having enough records to keep algorithm results accurate.

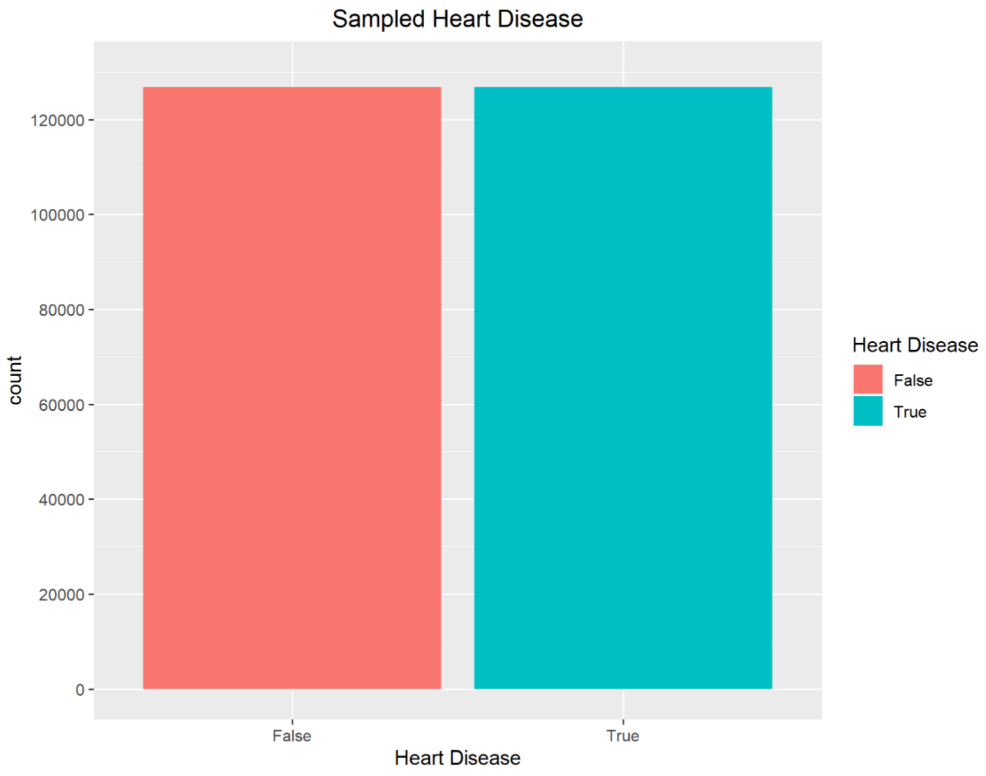
Applying ROSE to the data is a simple task. The first step is to set a seed, which enables the sampling results to be reproduced accurately after any randomization occurs. The ROSE algorithm is applied via a single line as shown in figure 13 below,

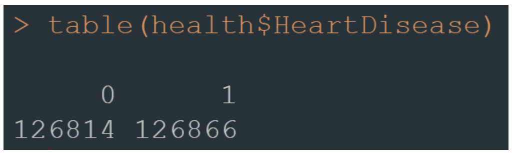


```r
#Applying ROSE sub sampling
set.seed(1234)
health<-ROSE(HeartDisease~.,data=health)$data
```

Figure 13

Health is now updated with its records sampled. The target variable will now be roughly equal as shown below in figure 14.

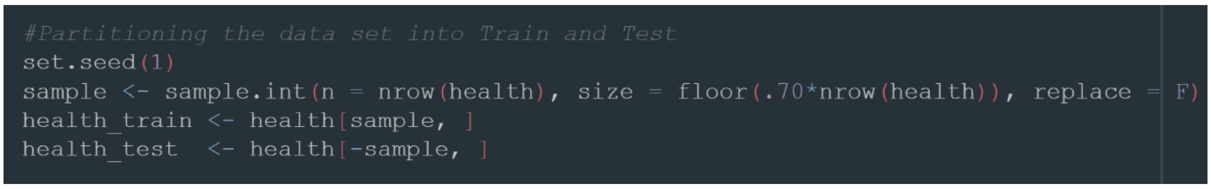




```
> table(health$HeartDisease)

     0      1 
126814 126866 
```

Figure 14

The developer at this point then split the dataset into the train and test sections. A split of 70/30 was chosen. This gives the dataset enough records in the train dataset, which is over 170,000 and a large test dataset which is around 70,000 records. First the seed is set, like sub sampling, to enable reproduceable randomization. The number of rows are calculated and 70% of these are assigned to the variable sample. Train takes the first 70% and test then takes the last 30% records. The test/train split is important for machine learning as it enables the algorithm to compare the trained data to untouched test data.



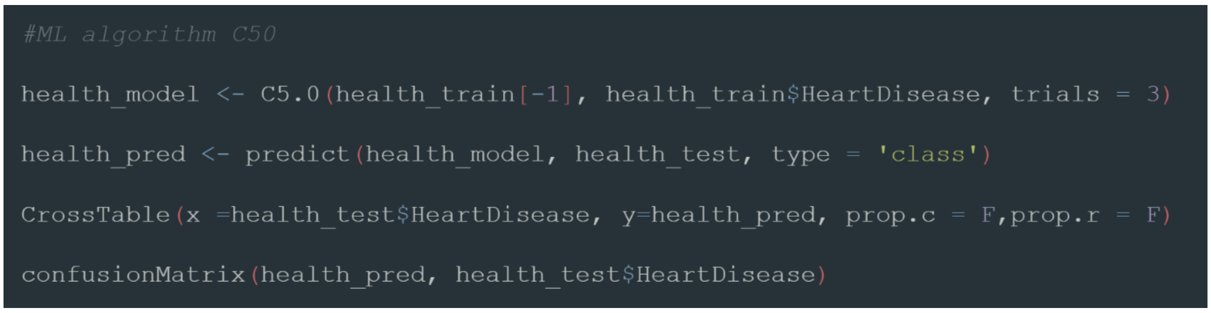
```r
#Partitioning the data set into Train and Test
set.seed(1)
sample <- sample.int(n = nrow(health), size = floor(.70*nrow(health)), replace = F)
health_train <- health[sample, ]
health_test  <- health[-sample, ]
```

Figure 15

Data Modelling Techniques

C5.0 Classification Model

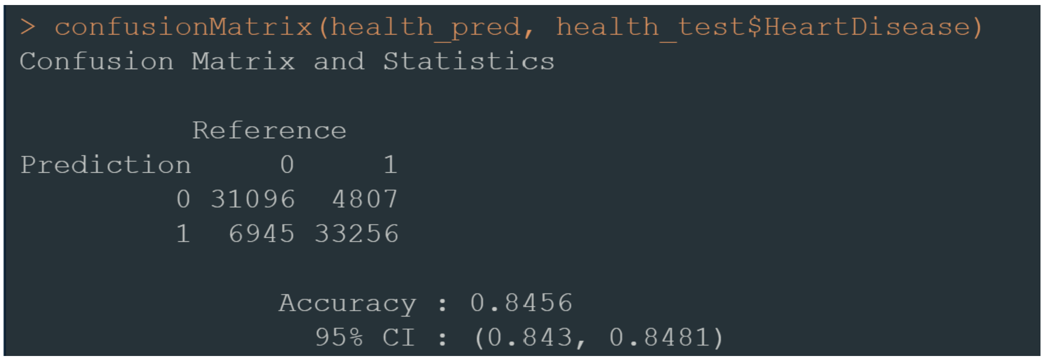
The first modelling technique the developer uses is the C5.0 classification model. This model is a decision tree library, like RPART and is a comparatively easy to use model, compared to Neural Networks and XGBoost. The below code shown in figure 16 is how the developer has run C5.0.



```
#ML algorithm C50

health_model <- C5.0(health_train[-1], health_train$HeartDisease, trials = 3)

health_pred <- predict(health_model, health_test, type = 'class')

CrossTable(x =health_test$HeartDisease, y=health_pred, prop.c = F,prop.r = F)

confusionMatrix(health_pred, health_test$HeartDisease)
```

Figure 16

As shown in the confusion matrix below in figure 17, the accuracy of C5.0 is 84.56%, which is the highest of the models which the developer used. This model provides the least false positives compared to the other algorithms.



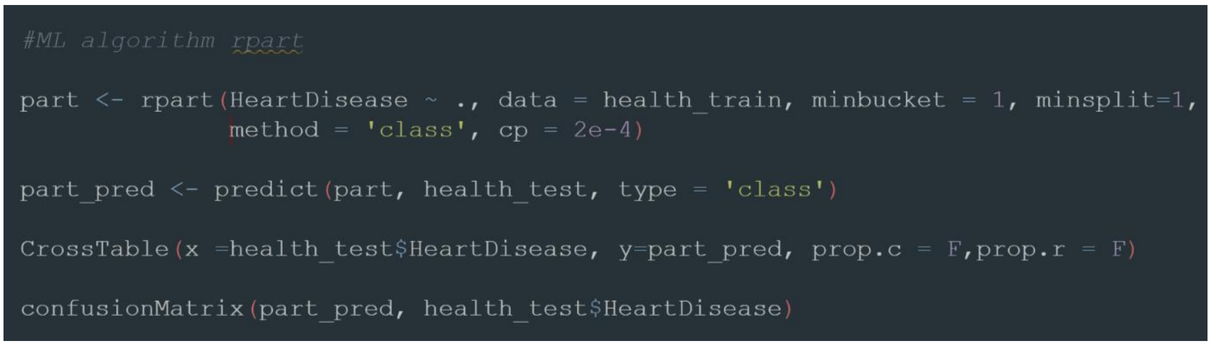
```
> confusionMatrix(health_pred, health_test$HeartDisease)
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 31096  4807
         1  6945 33256

               Accuracy : 0.8456
                 95% CI : (0.843, 0.8481)
```

Figure 17

The developer chose C5.0 as it is easy to understand and implement. It also produces quick and accurate results, as seen from the confusion matrix. It also requires very little tuning and little parameters are needed.

Recursive Partitioning and Regression Trees (RPART)

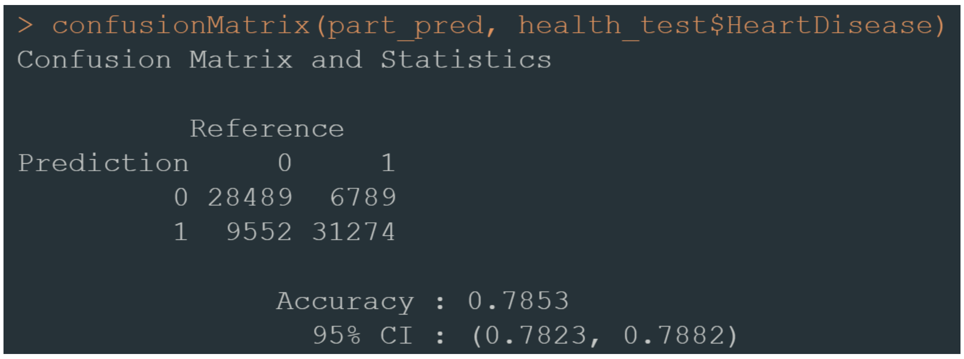
RPART is a package which is used by R to grow classification and recursive decision trees. This algorithm is a decision tree library, like C5.0. It employs the CART (classification and regression trees) algorithm. The below code in figure 18 shows the code used to run the rpart model.



```
#ML algorithm rpart

part <- rpart(HeartDisease ~ ., data = health_train, minbucket = 1, minsplit=1,
              method = 'class', cp = 2e-4)

part_pred <- predict(part, health_test, type = 'class')

CrossTable(x =health_test$HeartDisease, y=part_pred, prop.c = F,prop.r = F)

confusionMatrix(part_pred, health_test$HeartDisease)
```

Figure 18

The algorithm, when run, has an accuracy rate of 78%, as can be seen in the confusion matrix shown below in figure 19. This model is one of the less accurate of the models that the developer has run.



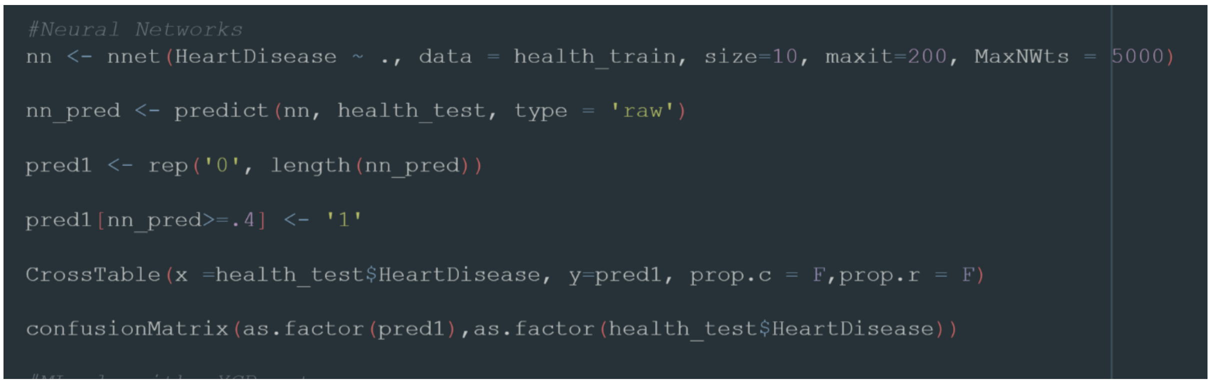
```
> confusionMatrix(part_pred, health_test$HeartDisease)
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 28489  6789
         1  9552 31274

               Accuracy : 0.7853
                 95% CI : (0.7823, 0.7882)
```

Figure 19

The developer has chosen RPART as an algorithm to compare with C5.0 as they are similar. The algorithm is easy to use and implement. Decision tree models are good at handling both numerical and categorical data which the health dataset contains.

Neural Networks
Neural networks are made up of simple I/O units called neurons. These neurons are all interconnected and each neuron have weights associated to them. Neural networks work similar to that of the human brain. Neural networks can be used for both classification and regression. Neural networks is the slowed running model and takes time to produce a prediction.

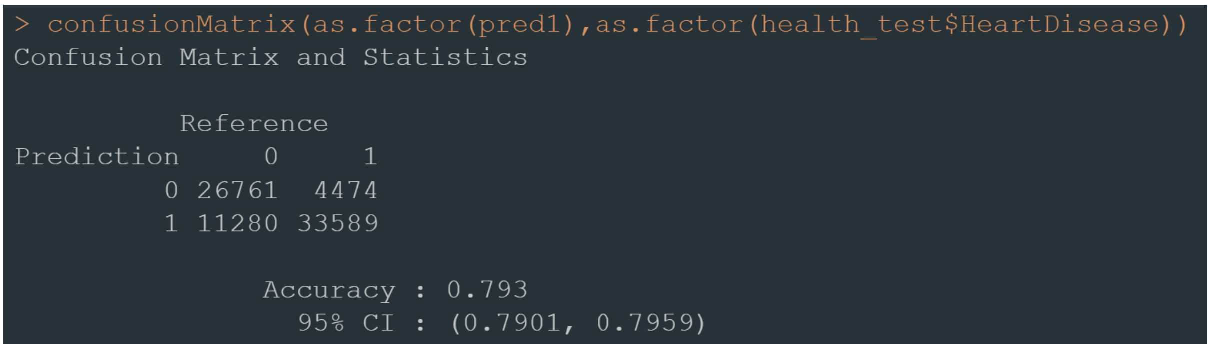
In Figure 20, the code used to run the neural network. The neural network runs for 200 interactions and the number of hidden nodes is set as 10.



```
#Neural Networks
nn <- nnet(HeartDisease ~ ., data = health_train, size=10, maxit=200, MaxNWts = 5000)

nn_pred <- predict(nn, health_test, type = 'raw')

pred1 <- rep('0', length(nn_pred))

pred1[nn_pred>=.4] <- '1'

CrossTable(x =health_test$HeartDisease, y=pred1, prop.c = F,prop.r = F)

confusionMatrix(as.factor(pred1),as.factor(health_test$HeartDisease))
```

Figure 20

The algorithm produces an accuracy of 79.3%, which is slightly lower than XGBoost but higher than RPART. This can be seen in the confusion matrix displayed in figure 21 below.



```
> confusionMatrix(as.factor(pred1),as.factor(health_test$HeartDisease))
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 26761  4474
         1 11280 33589

               Accuracy : 0.793
                 95% CI : (0.7901, 0.7959)
```

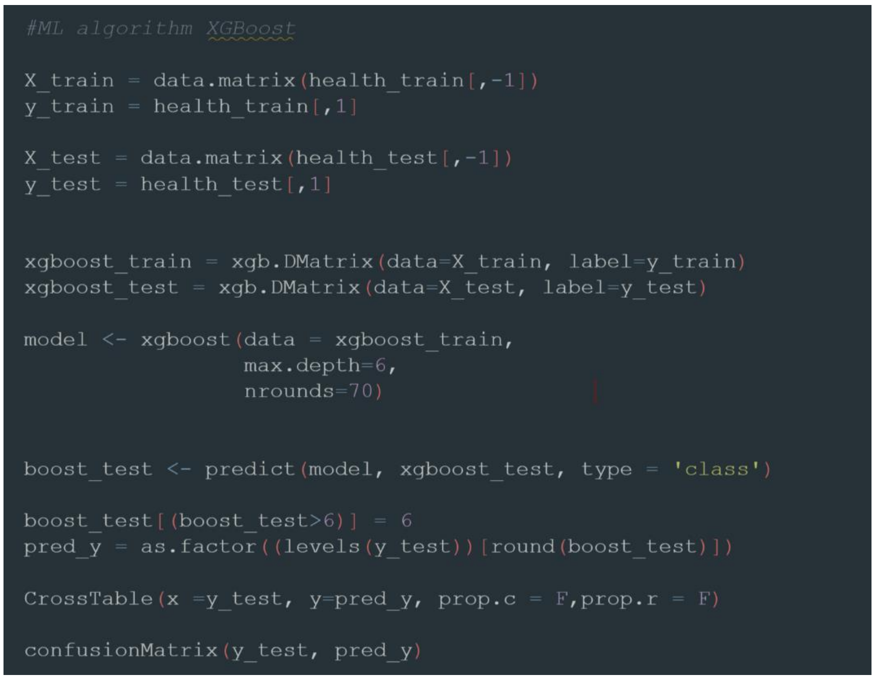
Figure 21

The developer chose neural networks as it provides a relatively accurate response. The algorithm is harder to implement, compared to the others. It is flexible and also deals well with large amounts of data, which the health training dataset contains. However, they are prone to overfitting and the number of hidden nodes has been tuned to 10 to prevent this as much as possible

XGBoost

XGBoost is a gradient boosting algorithm and has various functions, such as regression and classification. It has been used to win various awards on machine learning. The algorithm is much faster than other gradient boosting algorithms and is one of the faster algorithms to create decision trees on the health data.

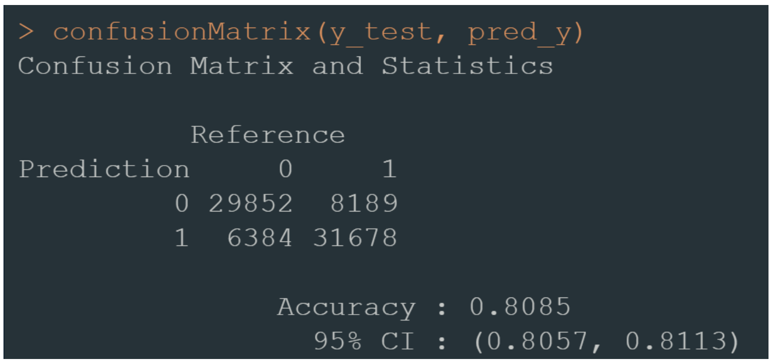
In figure 22, the code shown to run XGBoost is displayed. The data has to be converted into a data matrix before the algorithm can be run, and then is converted into a XGB matrix.



```
#ML algorithm XGBoost

X_train = data.matrix(health_train[,-1])
y_train = health_train[,1]

X_test = data.matrix(health_test[,-1])
y_test = health_test[,1]


xgboost_train = xgb.DMatrix(data=X_train, label=y_train)
xgboost_test = xgb.DMatrix(data=X_test, label=y_test)

model <- xgboost(data = xgboost_train,
                 max.depth=6,
                 nrounds=70)


boost_test <- predict(model, xgboost_test, type = 'class')

boost_test[(boost_test>6)] = 6
pred_y = as.factor((levels(y_test))[round(boost_test)])

CrossTable(x =y_test, y=pred_y, prop.c = F,prop.r = F)

confusionMatrix(y_test, pred_y)
```

Figure 22

In the confusion matrix shown in figure 23, the accuracy of the XGBoost algorithm is high, sitting at 80.85%, which is slightly lower than C5.0 but higher than Neural Networks and RPART.



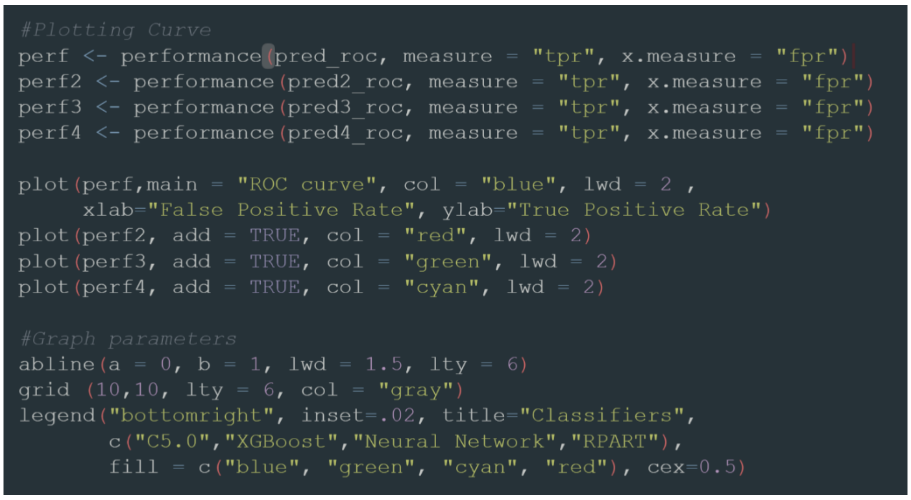
```
> confusionMatrix(y_test, pred_y)
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 29852  8189
         1  6384 31678

               Accuracy : 0.8085
                 95% CI : (0.8057, 0.8113)
```

Figure 23

The developer choose to include XGBoost as it is a boosting algorithm which makes it different to the other models as it prioritises speed and tends to give highly accurate results. It also supports regularization to reduce overfitting.

Results

To show results of the analysis, the developer has chosen a ROC graph to show the results of the 4 models. An ROC was chosen as it allows the 4 results to be simply compared and plotted against each other. It shows clearly the false positive rate against the true possible rate at each stage of the prediction. The code shown below in figure 24 produces the ROC graph displayed in figure 25.



```
#Plotting Curve
perf <- performance(pred_roc, measure = "tpr", x.measure = "fpr")
perf2 <- performance(pred2_roc, measure = "tpr", x.measure = "fpr")
perf3 <- performance(pred3_roc, measure = "tpr", x.measure = "fpr")
perf4 <- performance(pred4_roc, measure = "tpr", x.measure = "fpr")

plot(perf,main = "ROC curve", col = "blue", lwd = 2 ,
     xlab="False Positive Rate", ylab="True Positive Rate")
plot(perf2, add = TRUE, col = "red", lwd = 2)
plot(perf3, add = TRUE, col = "green", lwd = 2)
plot(perf4, add = TRUE, col = "cyan", lwd = 2)

#Graph parameters
abline(a = 0, b = 1, lwd = 1.5, lty = 6)
grid (10,10, lty = 6, col = "gray")
legend("bottomright", inset=.02, title="Classifiers",
       c("C5.0","XGBoost","Neural Network","RPART"),
       fill = c("blue", "green", "cyan", "red"), cex=0.5)
```

Figure 24

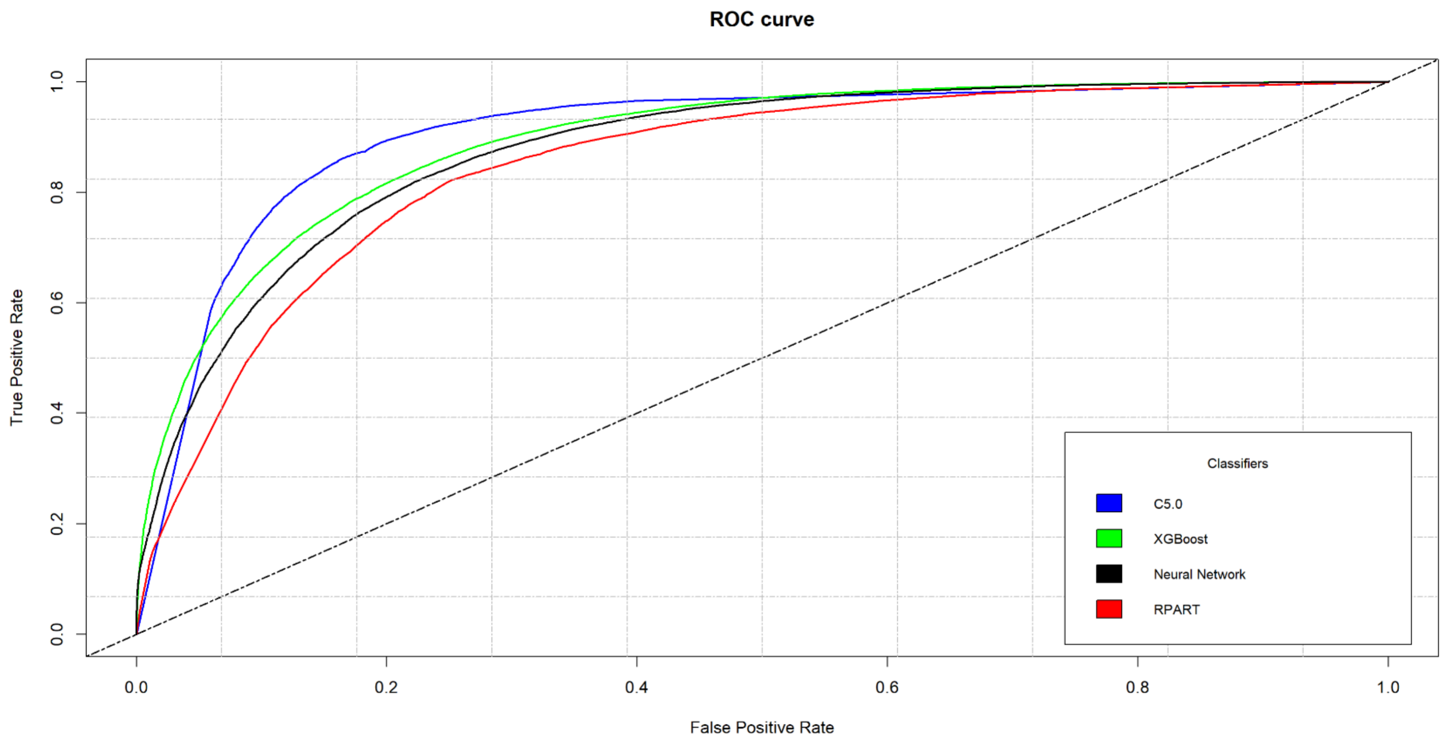
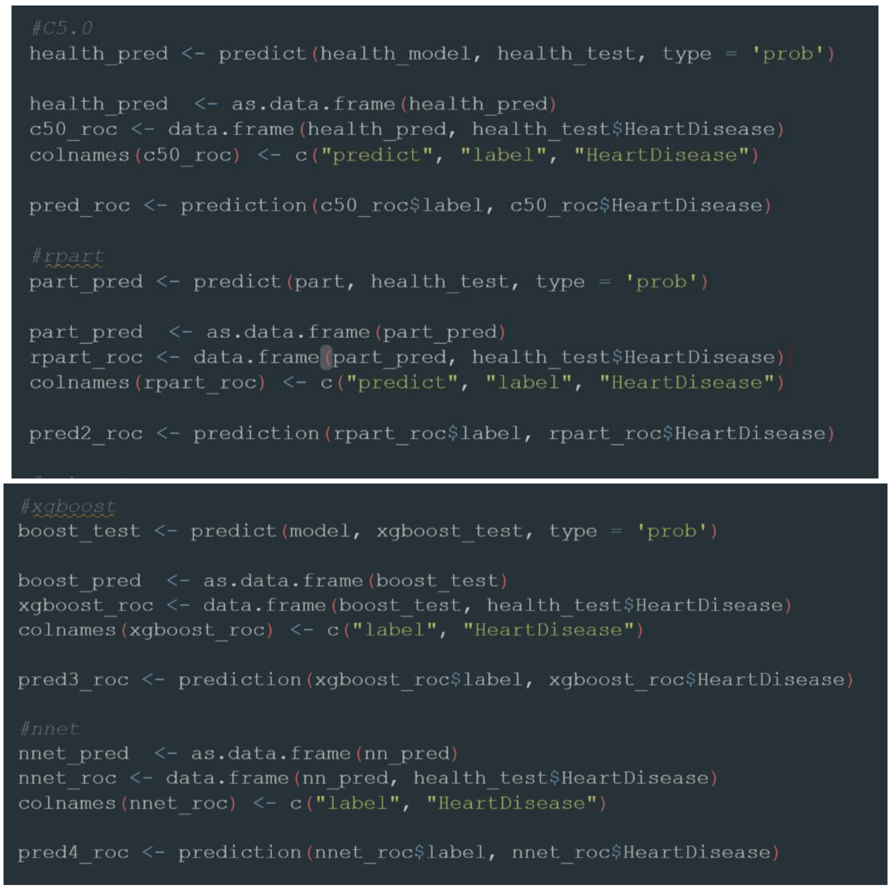


Figure 25

For each of the prediction results, each of these are predicted with a type of 'prob'. The results are placed into a data frame which is then added to a prediction object, along with the target variable. This is done before the ROC graph is plotted. The code shown below in figure 26 show this process.

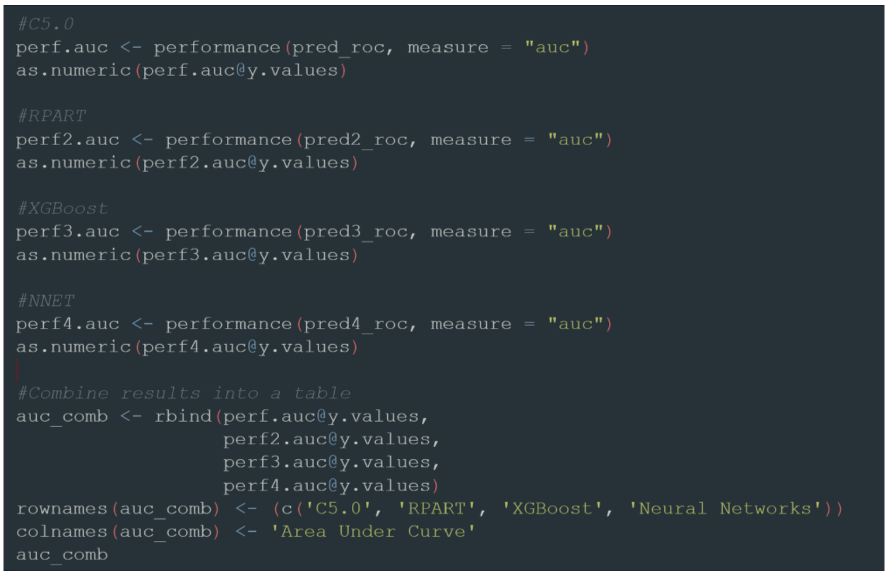


```r
#C5.0
health_pred <- predict(health_model, health_test, type = 'prob')

health_pred  <- as.data.frame(health_pred)
c50_roc <- data.frame(health_pred, health_test$HeartDisease)
colnames(c50_roc) <- c("predict", "label", "HeartDisease")

pred_roc <- prediction(c50_roc$label, c50_roc$HeartDisease)

#rpart
part_pred <- predict(part, health_test, type = 'prob')

part_pred  <- as.data.frame(part_pred)
rpart_roc <- data.frame(part_pred, health_test$HeartDisease)
colnames(rpart_roc) <- c("predict", "label", "HeartDisease")

pred2_roc <- prediction(rpart_roc$label, rpart_roc$HeartDisease)
```

```r
#xgboost
boost_test <- predict(model, xgboost_test, type = 'prob')

boost_pred  <- as.data.frame(boost_test)
xgboost_roc <- data.frame(boost_test, health_test$HeartDisease)
colnames(xgboost_roc) <- c("label", "HeartDisease")

pred3_roc <- prediction(xgboost_roc$label, xgboost_roc$HeartDisease)

#nnet
nnet_pred  <- as.data.frame(nn_pred)
nnet_roc <- data.frame(nn_pred, health_test$HeartDisease)
colnames(nnet_roc) <- c("label", "HeartDisease")

pred4_roc <- prediction(nnet_roc$label, nnet_roc$HeartDisease)
```

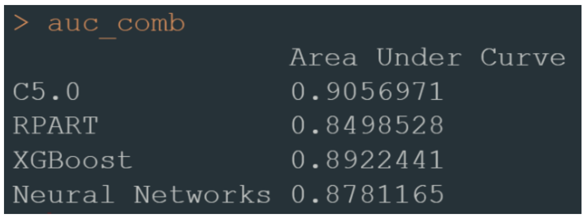
Figure 26

For each of the graphs, the area under the curve is calculated using the following code in figure 27. When run, the results are displayed as shown in figure 28.



```r
#C5.0
perf.auc <- performance(pred_roc, measure = "auc")
as.numeric(perf.auc@y.values)

#RPART
perf2.auc <- performance(pred2_roc, measure = "auc")
as.numeric(perf2.auc@y.values)

#XGBoost
perf3.auc <- performance(pred3_roc, measure = "auc")
as.numeric(perf3.auc@y.values)

#NNET
perf4.auc <- performance(pred4_roc, measure = "auc")
as.numeric(perf4.auc@y.values)

#Combine results into a table
auc_comb <- rbind(perf.auc@y.values,
                  perf2.auc@y.values,
                  perf3.auc@y.values,
                  perf4.auc@y.values)
rownames(auc_comb) <- (c('C5.0', 'RPART', 'XGBoost', 'Neural Networks'))
colnames(auc_comb) <- 'Area Under Curve'
auc_comb
```

Figure 27



```
> auc_comb
                Area Under Curve
C5.0                   0.9056971
RPART                  0.8498528
XGBoost                0.8922441
Neural Networks        0.8781165
```

Figure 28

The AUC results are in the same order as the analysis results. C5.0 has the highest area, and RPART has the least area under the curve.

Discussion

There are multiple factors which can determine if someone is at higher risk of heart disease. These variables will be discussed in the following section with data visualisation.

The first variable that the developer has plotted is ‘Age’ against ‘Heart Disease’. The developer has predicted that as the age bracket increases, there will be a higher chance of heart disease. The displayed results below in figure 29 agree with this trend. In each age range, the risk of heart disease increases. This ranges from small changes at the lower end of the age scale, to large percentage increases between ages towards the middle and end of the ages. Figure 29 shows a clear correlation between age and risk of heart disease.

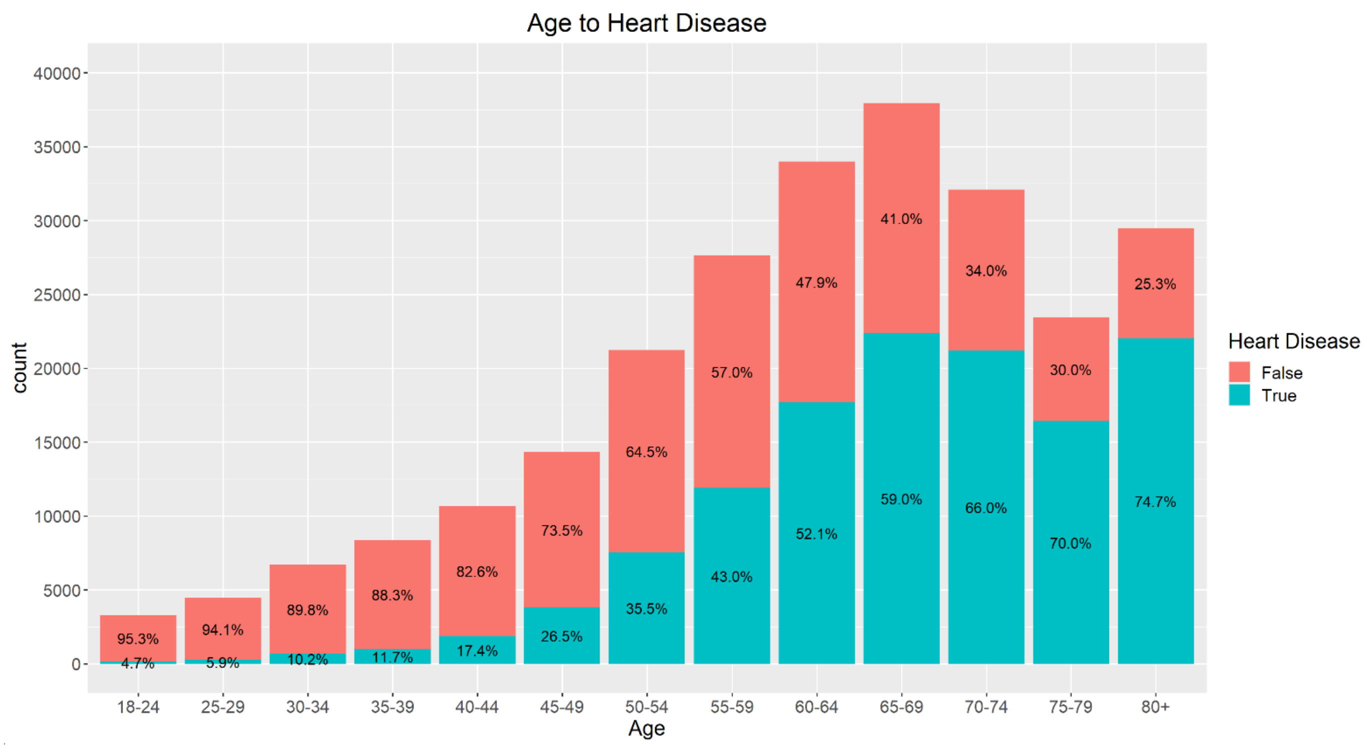


Figure 29

The next variable to be plotted was "General Health" vs "Heart Disease". The developer predicted that the higher the general health was, the lower the risk of heart disease. This was proved correct by the histogram in figure 30 as the heart disease risk increases from 17.90% in "Excellent", to "83.06%" in poor, with the middle section percentages all increasing as the general health decreased, for example, from "OK" to "Poor". "General Health" shows a clear correlation to "Heart Disease".

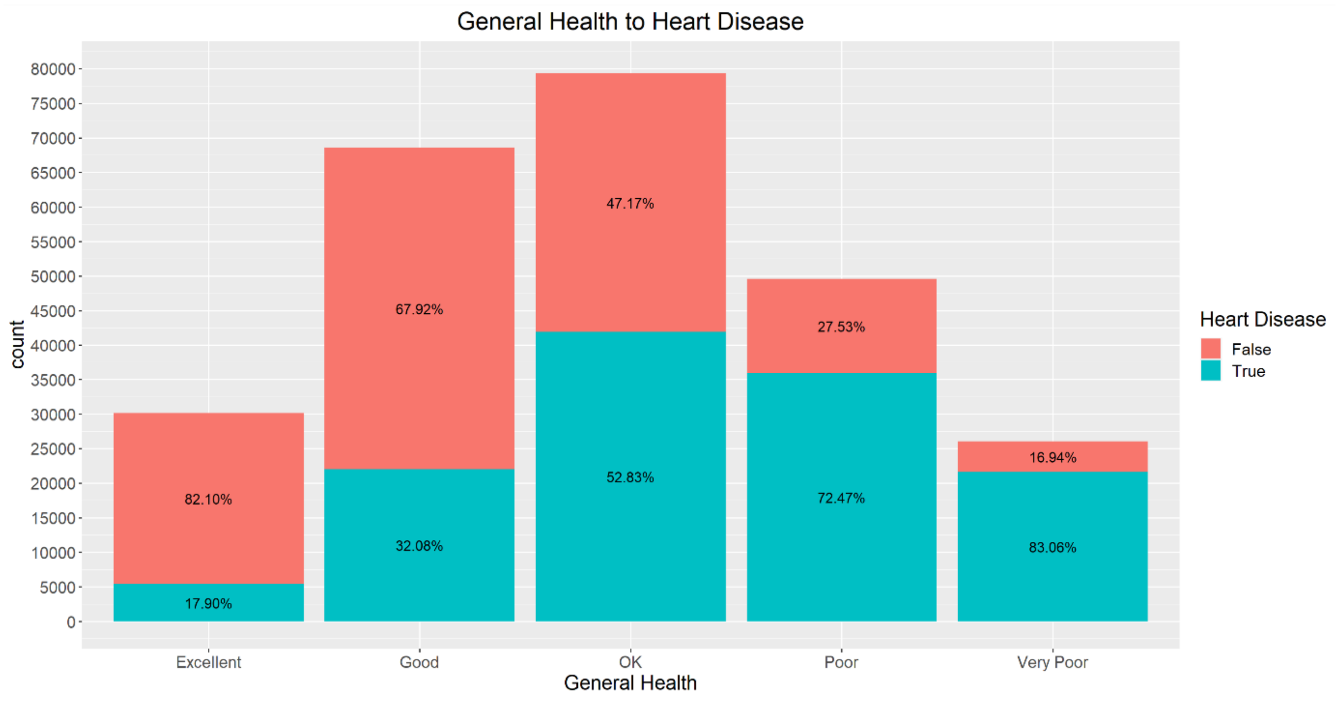


Figure 30

Diabetic status was compared to “Heart Disease” and this variable results were unsurprising, like the previous variables. The developer has predicted that those without diabetes would be significantly less at risk of heart disease than those with diabetes, and those with pre-diabetes would not be at a high of a risk with those with diabetes, but would still be at a greater risk of heart disease than those that were not diabetic. This was proven correct with in the histogram in figure 31. The only problem with this is that the pre-diabetic column does show many records and does not give an unbiased reading. The low amount of records has resulted in the graph percentages of pre-diabetes being very small. These read 38.1% without heart disease and 61.9% with.

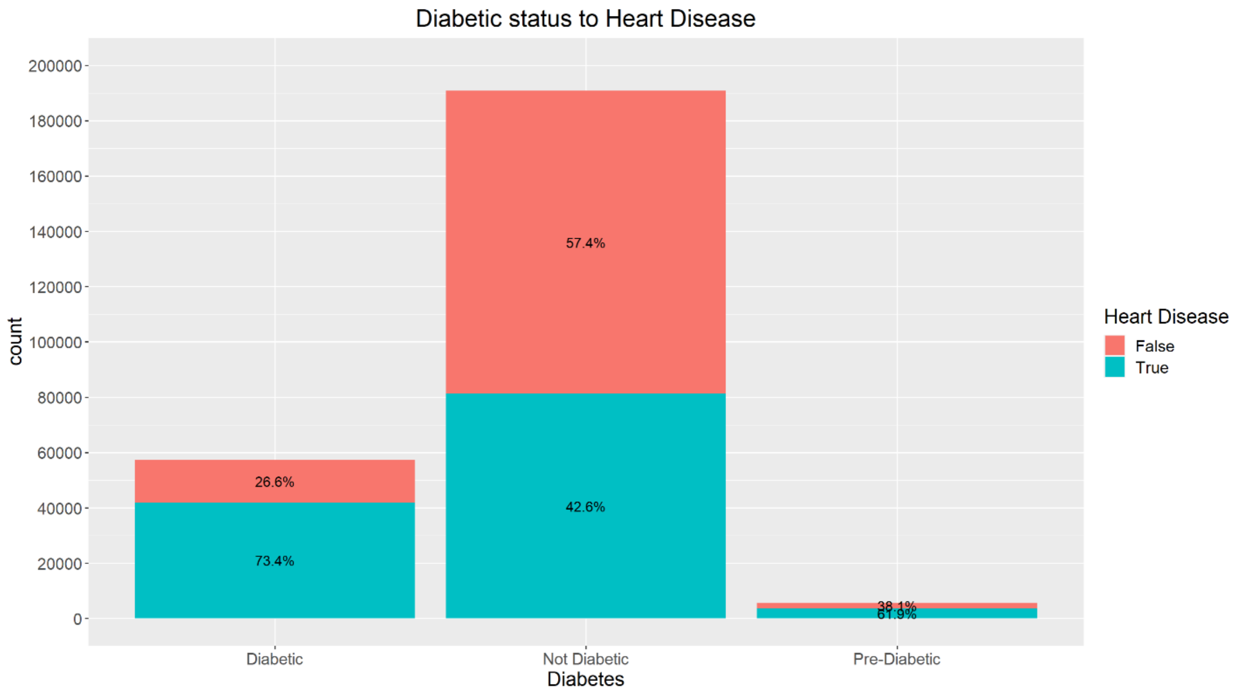


Figure 31

The next visualisation is different, as there are 3 variables shown. These are “Smoking” and “High Alcohol Consumption” to “Heart Disease”. The developer has decided to show these variables in a split histogram displayed below in figure 32. On the right, those who have heart disease are shown and on the left, those without are shown. The developer has predicted that both smoking and a high alcohol consumption will lead to a higher risk of heart disease. However, only one part of this is correct, although, there is a low number of records with a high alcohol consumption, leading to the results not being entirely accurate. This can only be solved with a higher number of alcoholics records in the dataset.

On the right side of the two histograms where “High Alcohol Consumption” is true, the bars is much smaller in size on the left graph compared to the right, regardless if the individual smokes or not. This brings the developer to the conclusion that there is not much correlation shown between high alcohol consumption and heart disease. However, smoking and heart disease are heavily correlated as can be seen from the right side graph as each section has a higher risk when the individual smokes compared to when they don’t smoke. The following graph is quite complex but displays multiple correlations.

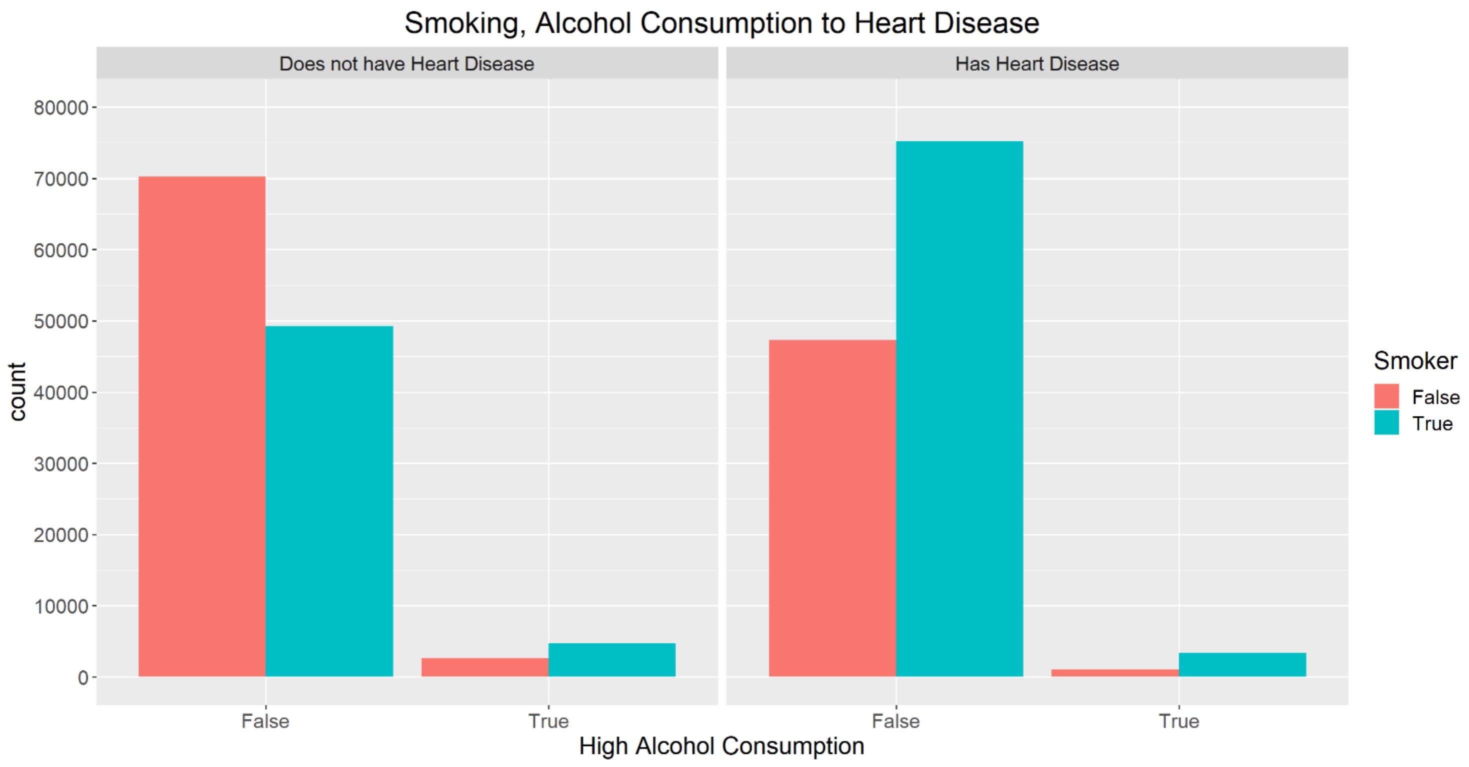


Figure 32

Figure 33 displayed below shows a split histogram, similar to figure 32, but this graph displays the correlation between “High Blood Pressure”, “High Cholesterol” and “Heart Disease”. As can be seen by the histogram on the right side of the graph, those who do not have high cholesterol and blood pressure is the smallest column. However, if either high cholesterol and blood pressure change to True, the risk of heart disease increases. If high blood pressure and high cholesterol are present, there is an extremely high chance of having heart disease. However, those who do not have heart disease, which is shown on the left, the most of these individuals did not have high blood pressure or high cholesterol. This shows a strong correlation between the 3 variables and this is not surprising to the developer.

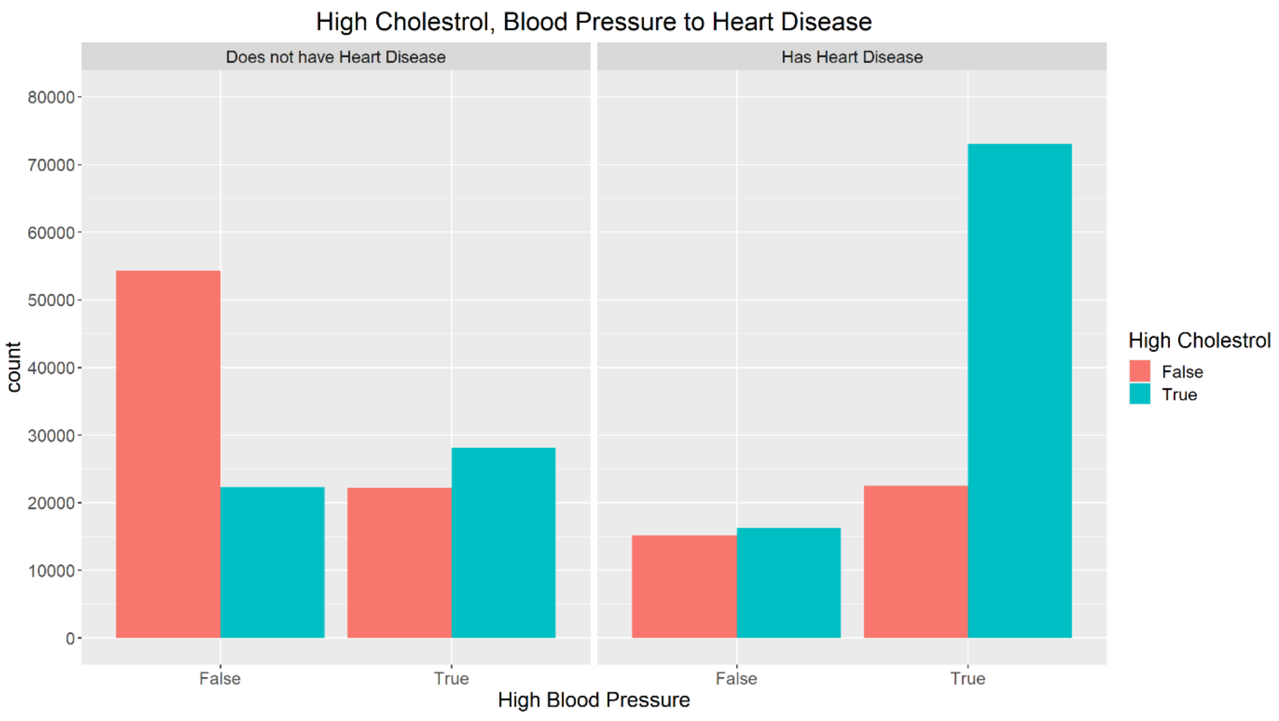


Figure 33

Figure 34 shows the correlation between "BMI" (Body Mass Index) and "Heart Disease". The developer has expected that around the healthy BMIs, which is around 17-25, the heart disease risk will be lowest and will increase when BMI increases. The displayed histogram shows exactly this. From the health range of 17-25, as the BMI increases, the risk of heart disease will increase rapidly. For example, at a BMI of 43, the rate of heart disease is at 65%. Below 17, the risk of heart disease increases slightly but a lack of data from this range may affect its accuracy.

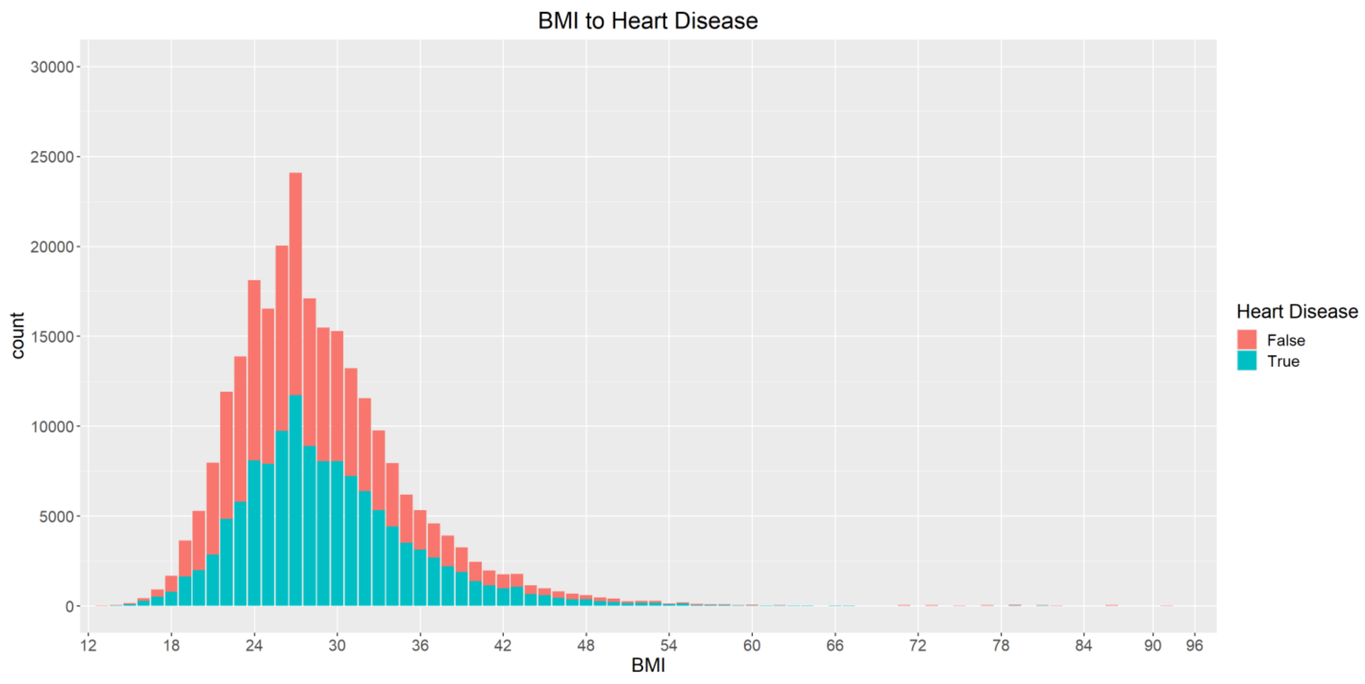


Figure 34

Finally, the correlation between “Sex” and “Heart Disease” is shown in figure 35 and as predicted by the developer, heart disease for male is significantly higher than in female. There is a correlation between sex and heart disease, although the lack of other variables may make this data visualisation a little biased.

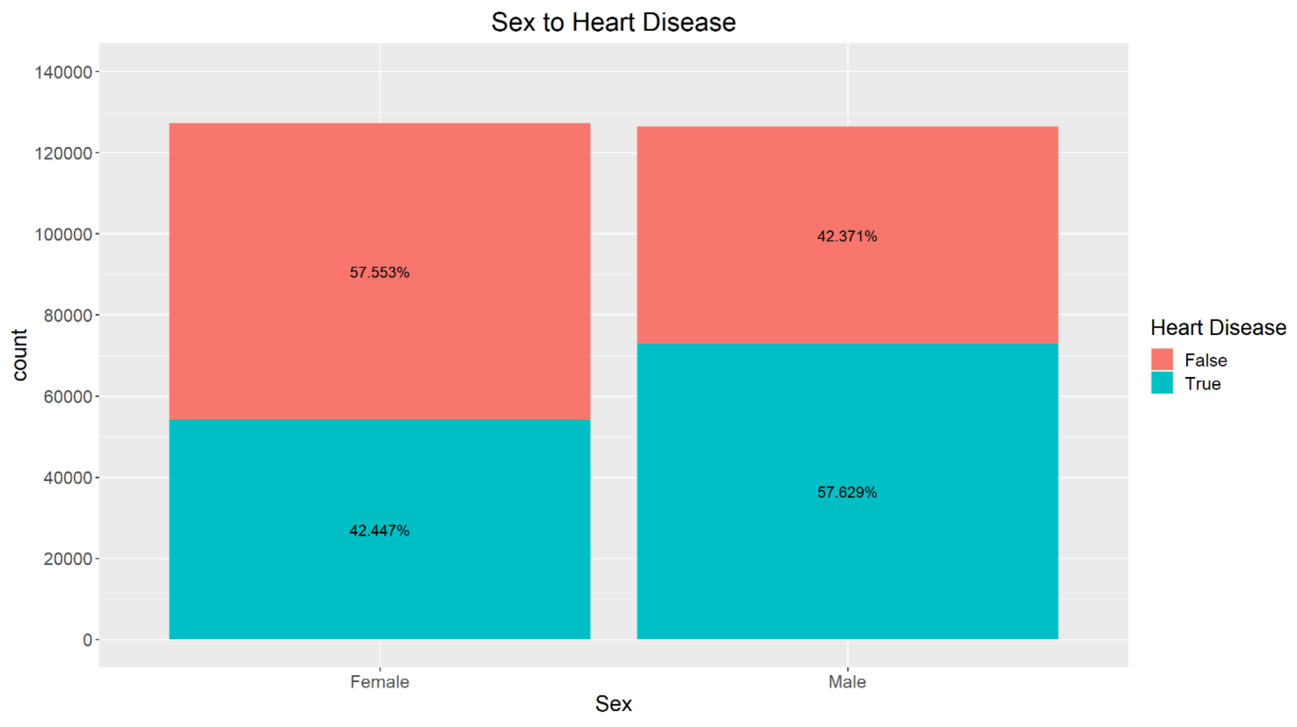


Figure 35

Conclusion

In conclusion, the developer has conducted thorough analysis on the Heart Disease Health Indicators dataset and would now be able to predict accurately, depending on a number of variables. if a individual was at risk of heart disease. The developer has learned in this assignment multiple processes, such as reading in a dataset, cleaning the dataset such as renaming and removing columns. The developer has also learned processes such as ROSE subsampling, splitting datasets into train/test and tuning then running several machine learning algorithms. The developer then undertook analysis on these results and used a ROC curve to plot analysis results, along with calculating the area under the curve. The results yielded from this are as follows. C5.0 had the highest AUC, followed by XGBoost, then Neural Networks. RPART had the lowest AUC.

If the developer was to carry out this analysis again, they would use different analytical models such as KNN to see the difference in accuracies and AUC. They would also ensure that the analysis accuracies would be higher on average, such as mid/high 80s. The developer would also perhaps choose another dataset, with more categorical variables, instead of binary ones, to gain a deeper understanding of the correlations between variables and to provide different types of data visualisation.

References

[1] University Lectures / Tutorials / Past Projects

[2] AWS (n.d.). *Predicting Earning Potential using the Adult Dataset*. [online] rstudio-pubs-static.s3.amazonaws.com. Available at: https://rstudio-pubs-static.s3.amazonaws.com/235617_51e06fa6c43b47d1b6daca2523b2f9e4.html . [Accessed 1 Apr. 2022].

[3] geeksforgeeks.org (2020). *How Neural Networks are used for Regression in R Programming?* [online] GeeksforGeeks. Available at: https://www.geeksforgeeks.org/how-neural-networks-are-used-for-regression-in-r-programming/. [Accessed 4 Apr. 2022].

[4] Holmes, J. (2021). *What is Rpart in R? - Cement Answers*. [online] cementanswers.com. Available at: https://cementanswers.com/what-is-rpart-in-r [Accessed 9 Apr. 2022].

[5] Jason Brownlee (2016). *A Gentle Introduction to XGBoost for Applied Machine Learning*. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/. [Accessed 8 Apr. 2022].

[6] Lunardon, N., Menardi, G. and Torelli, N. (2015). *Package 'ROSE' Title ROSE: Random Over-Sampling Examples*. [online] Available at: https://cran.r-project.org/web/packages/ROSE/ROSE.pdf.

[7] R, M. with (2019). *Methods for dealing with imbalanced data | R-bloggers*. [online] r-bloggers. Available at: https://www.r-bloggers.com/2019/04/methods-for-dealing-with-imbalanced-data/.

[8] Ripley, B., Venables, W. and Maintainer (2020). *Package 'nnet' NeedsCompilation yes*. [online] Available at: https://cran.r-project.org/web/packages/nnet/nnet.pdf. [Accessed 2 Apr. 2022].

[9] Stack Overflow. (n.d.). *r - Showing data values on stacked bar chart in ggplot2*. [online] Available at: https://stackoverflow.com/questions/6644997/showing-data-values-on-stacked-bar-chart-in-ggplot2 [Accessed 11 Apr. 2022].

[10] www.rdocumentation.org. (n.d.). *rpart function - RDocumentation*. [online] Available at: https://www.rdocumentation.org/packages/rpart/versions/4.1.16/topics/rpart [Accessed 30 Mar. 2022].

[11] kaggle.com. (n.d.). Machine Learning with XGBoost (in R). [online] Available at: https://www.kaggle.com/code/rtatman/machine-learning-with-xgboost-in-r/notebook [Accessed 1 Apr. 2022].

Appendix A

```
#-----------------Heart Disease Analysis-----------------------


#Install Libraries
install.packages("caret")
install.packages("ggplot2")
install.packages("C50")
install.packages("gmodels")
install.packages("ROCR")
install.packages("rpart")
install.packages("nnet")
install.packages("xgboost")
install.packages("ROSE")


#Libraries
library(ggplot2)
library(C50)
library(gmodels)
library(caret)
library(ROCR)
library(rpart)
library(nnet)
library(xgboost)
library(ROSE)
library(scales)


#Read in  and check working directory
setwd("C:/Users/alexa/OneDrive/Data Analytics/project")
```

```
getwd()

#Read in data-set csv file and check values
health <- read.csv("health_indicators.csv")
original_health <- read.csv("health_indicators.csv")
str(health)

#-------------Cleaning-----------------

#Change Column Names
colnames(health) <- c("HeartDisease",
                "HighBloodPressure",
                "HighCholesterol",
                "CholestrolCheck",
                "BMI",
                "Smoker",
                "Stroke",
                "Diabetes",
                "PhysicalActivity",
                "Fruits",
                "Vegetables",
                "HighAlcoholConsmp",
                "Healthcare",
                "MedicalCost",
                "GeneralHealth",
                "MentalHealth",
                "PhysicalHealth",
                "WalkingDifficulty",
                "Sex",
                "Age",
                "EducationLevel",
                "IncomeLevel")



#Change Field types to factors

health$HeartDisease <- as.factor(health$HeartDisease)
health$HighBloodPressure <- as.factor(health$HighBloodPressure)
health$HighCholesterol <- as.factor(health$HighCholesterol)
health$CholestrolCheck <- as.factor(health$CholestrolCheck)
health$Smoker <- as.factor(health$Smoker)
health$Stroke <- as.factor(health$Stroke)
health$PhysicalActivity <- as.factor(health$PhysicalActivity)
health$HighAlcoholConsmp <- as.factor(health$HighAlcoholConsmp)
health$Healthcare <- as.factor(health$Healthcare)
health$MedicalCost <- as.factor(health$MedicalCost)
health$WalkingDifficulty <- as.factor(health$WalkingDifficulty)
health$EducationLevel <- as.factor(health$EducationLevel)
health$BMI <- as.factor(health$BMI)



#Remove Columns
health$Fruits <- NULL
health$Vegetables <- NULL

#Change Scale from 1-13 to Age Range

health$Age [health$Age=="1"] <- "18-24"
```

```
health$Age [health$Age=="2"] <- "25-29"

health$Age [health$Age=="3"] <- "30-34"

health$Age [health$Age=="4"] <- "35-39"

health$Age [health$Age=="5"] <- "40-44"

health$Age [health$Age=="6"] <- "45-49"

health$Age [health$Age=="7"] <- "50-54"

health$Age [health$Age=="8"] <- "55-59"

health$Age [health$Age=="9"] <- "60-64"

health$Age [health$Age=="10"] <- "65-69"

health$Age [health$Age=="11"] <- "70-74"

health$Age [health$Age=="12"] <- "75-79"

health$Age [health$Age=="13"] <- "80+"


health$Age=as.factor(health$Age)



#Income Level Change

health$IncomeLevel [health$IncomeLevel=="1"] <- "<$10000"

health$IncomeLevel [health$IncomeLevel=="2"] <- ">=$10000"

health$IncomeLevel [health$IncomeLevel=="3"] <- ">=$25000"

health$IncomeLevel [health$IncomeLevel=="4"] <- ">=$35000"

health$IncomeLevel [health$IncomeLevel=="5"] <- ">=$45000"

health$IncomeLevel [health$IncomeLevel=="6"] <- ">=$55000"

health$IncomeLevel [health$IncomeLevel=="7"] <- ">=$65000"

health$IncomeLevel [health$IncomeLevel=="8"] <- ">=$75000"


health$IncomeLevel=as.factor(health$IncomeLevel)


#Change Diabetes from 0-2 to Diabetic Types
```

```
health$Diabetes [health$Diabetes=="0"] <- "Not Diabetic"
health$Diabetes [health$Diabetes=="1"] <- "Pre-Diabetic"
health$Diabetes [health$Diabetes=="2"] <- "Diabetic"

health$Diabetes=as.factor(health$Diabetes)


#Change General Health from 1-5 to Description

health$GeneralHealth [health$GeneralHealth=="1"] <- "Excellent"
health$GeneralHealth [health$GeneralHealth=="2"] <- "Good"
health$GeneralHealth [health$GeneralHealth=="3"] <- "OK"
health$GeneralHealth [health$GeneralHealth=="4"] <- "Poor"
health$GeneralHealth [health$GeneralHealth=="5"] <- "Very Poor"

health$GeneralHealth=as.factor(health$GeneralHealth)


#Change Sex from 0-1

health$Sex [health$Sex=="0"] <- "Female"
health$Sex [health$Sex=="1"] <- "Male"

health$Sex=as.factor(health$Sex)


#Check for missing values and display results
colSums(is.na(health))
```

```
#-------------Data set arrangement-----------

#Applying ROSE sub sampling
set.seed(1234)
health<-ROSE(HeartDisease~.,data=health)$data

#Comparison between sub sampled data and the original
table(health$HeartDisease)
table(original_health$HeartDiseaseorAttack)

#Partitioning the data set into Train and Test
set.seed(1)
sample <- sample.int(n = nrow(health), size = floor(.70*nrow(health)), replace = F)
health_train <- health[sample, ]
health_test  <- health[-sample, ]

str(health_train)
#-------------Method Graph Plots----------------

#Heart disease label
health_label <- c("False", "True")

#Plot unchanged Heart disease
ggplot(data = original_health, aes(x=as.factor(HeartDiseaseorAttack),
                              fill= as.factor(HeartDiseaseorAttack)))+
  geom_bar()+ scale_y_continuous(breaks = seq(0, 240000, 20000), lim = c(0, 240000))+
  scale_x_discrete(labels= health_label, "Heart Disease") +
  scale_fill_discrete(name = "Heart Disease", labels = c("False", "True"))+
  ggtitle("Unsampled Heart Disease")+
  theme(plot.title = element_text(hjust = 0.5))

#Plot Changed Heart Disease
ggplot(data = health, aes(x=HeartDisease,fill=HeartDisease))+geom_bar() +
  scale_y_continuous(breaks = seq(0, 130000, 20000), lim = c(0, 130000))+
  scale_x_discrete(labels= health_label, "Heart Disease") +
  scale_fill_discrete(name = "Heart Disease", labels = c("False", "True")) +
  ggtitle("Sampled Heart Disease")+
  theme(plot.title = element_text(hjust = 0.5))


#----------------Discussion Graphs------------------


#Histogram of Count of Ages to heart disease
ggplot(data = health, aes(x=Age,fill=HeartDisease))+geom_bar()+
  scale_y_continuous(breaks = seq(0, 40000, 5000), lim = c(0, 40000))+
  scale_fill_discrete(name = "Heart Disease", labels = c("False", "True")) +
  ggtitle("Age to Heart Disease")+
  theme(plot.title = element_text(hjust = 0.5), text=element_text(size=16))+
  geom_text(aes(label = scales::percent(..count../tapply(..count.., ..x.. ,sum)[..x..])),
            position = position_stack(vjust = 0.5), size = 4, stat = "count")


#Histogram of diabetes to heart disease
ggplot(data = health, aes(x=Diabetes,fill=HeartDisease))+geom_bar()+
  scale_y_continuous(breaks = seq(0, 200000, 20000), lim = c(0, 200000))+
  scale_fill_discrete(name = "Heart Disease", labels = c("False", "True")) +
  ggtitle("Diabetic status to Heart Disease")+
  theme(plot.title = element_text(hjust = 0.5),text=element_text(size=16))+
  geom_text(aes(label = scales::percent(..count../tapply(..count.., ..x.. ,sum)[..x..])),
            position = position_stack(vjust = 0.5), size = 4, stat = "count")


#Histogram of general health to heart disease
ggplot(data = health, aes(x=GeneralHealth,fill=HeartDisease))+geom_bar()+
  scale_x_discrete(name = "General Health")+
  scale_y_continuous(breaks = seq(0, 80000, 5000), lim = c(0, 80000))+
  scale_fill_discrete(name = "Heart Disease", labels = c("False", "True")) +
  ggtitle("General Health to Heart Disease")+
  theme(plot.title = element_text(hjust = 0.5), text=element_text(size=16))+
  geom_text(aes(label = scales::percent(..count../tapply(..count.., ..x.. ,sum)[..x..])),
            position = position_stack(vjust = 0.5), size = 4, stat = "count")

#Relationship between blood pressure and cholesterol
ggplot(data = health) +
  geom_bar(aes(x=HighBloodPressure, fill=HighCholesterol),
           position = "dodge") +
  facet_wrap(~HeartDisease, labeller = labeller(HeartDisease = c("0" = "Does not have Heart
Disease", "1" = "Has Heart Disease")))+
  ggtitle("High Cholestrol, Blood Pressure to Heart Disease")+
  scale_x_discrete(name ="High Blood Pressure", labels=c("0" = "False","1" = "True"))+
  scale_y_continuous(breaks = seq(0, 80000, 10000), lim = c(0, 80000))+
  scale_fill_discrete(name = "High Cholestrol", labels = c("False", "True")) +
  theme(plot.title = element_text(hjust = 0.5), text=element_text(size=16))

#Relationship between smoking, drinking and heart disease
ggplot(data = health) +
  geom_bar(aes(x=HighAlcoholConsmp, fill=Smoker),
           position = "dodge") +
  facet_wrap(~HeartDisease, labeller = labeller(HeartDisease = c("0" = "Does not have Heart
Disease", "1" = "Has Heart Disease")))+
  ggtitle("Smoking, Alcohol Consumption to Heart Disease")+
  scale_x_discrete(name ="High Alcohol Consumption", labels=c("0" = "False","1" = "True"))+
  scale_y_continuous(breaks = seq(0, 80000, 10000), lim = c(0, 80000))+
  scale_fill_discrete(name = "Smoker", labels = c("False", "True")) +
  theme(plot.title = element_text(hjust = 0.5), text=element_text(size=16))


#BMI Histogram
ggplot(data = health, aes(x=BMI,fill=HeartDisease))+geom_bar()+
  scale_y_continuous(breaks = seq(0, 30000, 5000), lim = c(0, 30000))+
  scale_x_discrete(name = "BMI", breaks = seq(12, 96, 6))+
  scale_fill_discrete(name = "Heart Disease", labels = c("False", "True")) +
  ggtitle("BMI to Heart Disease")+
  theme(plot.title = element_text(hjust = 0.5), text=element_text(size=16))


#Histogram of Sex to Heart disease
ggplot(data = health, aes(x=Sex,fill=HeartDisease))+geom_bar()+
  scale_y_continuous(breaks = seq(0, 140000, 20000), lim = c(0, 140000))+
  scale_fill_discrete(name = "Heart Disease", labels = c("False", "True")) +
  ggtitle("Sex to Heart Disease")+
  theme(plot.title = element_text(hjust = 0.5), text=element_text(size=16))+
  geom_text(aes(label = scales::percent(..count../tapply(..count.., ..x.. ,sum)[..x..])),
            position = position_stack(vjust = 0.5), size = 4, stat = "count")

#-------------Training the data set-----------------
```

```
#ML algorithm C50

health_model <- C5.0(health_train[-1], health_train$HeartDisease, trials = 3)

health_pred <- predict(health_model, health_test, type = 'class')

CrossTable(x =health_test$HeartDisease, y=health_pred, prop.c = F,prop.r = F)

confusionMatrix(health_pred, health_test$HeartDisease)

#ML algorithm rpart

part <- rpart(HeartDisease ~ ., data = health_train, minbucket = 1, minsplit=1,
        method = 'class', cp = 2e-4)

part_pred <- predict(part, health_test, type = 'class')

CrossTable(x =health_test$HeartDisease, y=part_pred, prop.c = F,prop.r = F)

confusionMatrix(part_pred, health_test$HeartDisease)

#Neural Networks
nn <- nnet(HeartDisease ~ ., data = health_train, size=10, maxit=200, MaxNWts = 5000)

nn_pred <- predict(nn, health_test, type = 'raw')

pred1 <- rep('0', length(nn_pred))
```

```
pred1[nn_pred>=.4] <- '1'

CrossTable(x =health_test$HeartDisease, y=pred1, prop.c = F,prop.r = F)

confusionMatrix(as.factor(pred1),as.factor(health_test$HeartDisease))

#ML algorithm XGBoost

X_train = data.matrix(health_train[,-1])
y_train = health_train[,1]

X_test = data.matrix(health_test[,-1])
y_test = health_test[,1]



xgboost_train = xgb.DMatrix(data=X_train, label=y_train)
xgboost_test = xgb.DMatrix(data=X_test, label=y_test)

model <- xgboost(data = xgboost_train,
           max.depth=6,
           nrounds=70)



boost_test <- predict(model, xgboost_test, type = 'class')

boost_test[(boost_test>6)] = 6
pred_y = as.factor((levels(y_test))[round(boost_test)])

CrossTable(x =y_test, y=pred_y, prop.c = F,prop.r = F)
```

```
confusionMatrix(y_test, pred_y)



#------------------ROCR graph---------------

#C5.0
health_pred <- predict(health_model, health_test, type = 'prob')

health_pred  <- as.data.frame(health_pred)
c50_roc <- data.frame(health_pred, health_test$HeartDisease)
colnames(c50_roc) <- c("predict", "label", "HeartDisease")

pred_roc <- prediction(c50_roc$label, c50_roc$HeartDisease)

#rpart
part_pred <- predict(part, health_test, type = 'prob')

part_pred  <- as.data.frame(part_pred)
rpart_roc <- data.frame(part_pred, health_test$HeartDisease)
colnames(rpart_roc) <- c("predict", "label", "HeartDisease")

pred2_roc <- prediction(rpart_roc$label, rpart_roc$HeartDisease)

#xgboost
boost_test <- predict(model, xgboost_test, type = 'prob')

boost_pred  <- as.data.frame(boost_test)
xgboost_roc <- data.frame(boost_test, health_test$HeartDisease)
```

```
colnames(xgboost_roc) <- c("label", "HeartDisease")

pred3_roc <- prediction(xgboost_roc$label, xgboost_roc$HeartDisease)

#nnet
nnet_pred  <- as.data.frame(nn_pred)
nnet_roc <- data.frame(nn_pred, health_test$HeartDisease)
colnames(nnet_roc) <- c("label", "HeartDisease")

pred4_roc <- prediction(nnet_roc$label, nnet_roc$HeartDisease)

#Plotting Curve
perf <- performance(pred_roc, measure = "tpr", x.measure = "fpr")
perf2 <- performance(pred2_roc, measure = "tpr", x.measure = "fpr")
perf3 <- performance(pred3_roc, measure = "tpr", x.measure = "fpr")
perf4 <- performance(pred4_roc, measure = "tpr", x.measure = "fpr")

plot(perf,main = "ROC curve", col = "blue", lwd = 2 ,
   xlab="False Positive Rate", ylab="True Positive Rate")
plot(perf2, add = TRUE, col = "red", lwd = 2)
plot(perf3, add = TRUE, col = "green", lwd = 2)
plot(perf4, add = TRUE, col = "black", lwd = 2)

#Graph parameters
abline(a = 0, b = 1, lwd = 1.5, lty = 6)
grid (10,10, lty = 6, col = "gray")
legend("bottomright", inset=.02, title="Classifiers",
     c("C5.0","XGBoost","Neural Network","RPART"),
     fill = c("blue", "green", "black", "red"), cex=0.8)
```

```
#AUC calculations


#C5.0

perf.auc <- performance(pred_roc, measure = "auc")

as.numeric(perf.auc@y.values)


#RPART

perf2.auc <- performance(pred2_roc, measure = "auc")

as.numeric(perf2.auc@y.values)


#XGBoost

perf3.auc <- performance(pred3_roc, measure = "auc")

as.numeric(perf3.auc@y.values)


#NNET

perf4.auc <- performance(pred4_roc, measure = "auc")

as.numeric(perf4.auc@y.values)


#Combine results into a table

auc_comb <- rbind(perf.auc@y.values,

                perf2.auc@y.values,

                perf3.auc@y.values,

                perf4.auc@y.values)

rownames(auc_comb) <- (c('C5.0', 'RPART', 'XGBoost', 'Neural Networks'))

colnames(auc_comb) <- 'Area Under Curve'

auc_comb
```