Statistical Learning Final Project Report

Heart Disease Prediction

Nora Nikoloska

nora.nikoloska@studenti.unipd.it

Sara Kartalovic

sara.kartalovic@studenti.unipd.it

December 30, 2021

1 Introduction

In this project the Heart Disease Data Set is explored in the problem of heart disease classification. The data is combined from four different hospital data sources, preprocessed and normalized. After the dataset exploration, the predictions are generated in two different approaches. The first approach is taking into account all four classes and making a prediction based on the combined decision between single binary classifiers for each class. The second approach is combining all disease states in a single class and differentiating between healthy and disease states. In this binary classification problem several techniques are used in order to generate the best model among which: cross-validation, variable selection and shrinkage models. Other models such as discriminant analysis and KNN are also included. The report contains executed code all of the different experiments, results visualisations and obtained accuracy values.

2 Dataset Decription

The dataset used is publicly available as Heart Disease Data Set[1]. It is a collection of data from four different hospitals:

1. Hungarian Institute of Cardiology. Budapest
2. University Hospital, Zurich, Switzerland
3. University Hospital, Basel, Switzerland
4. V.A. Medical Center, Long Beach and Cleveland Clinic Foundation

The number of subjects for each hospital are the following: Cleveland - 281, Hungarian - 294 Switzerland - 123, Long Beach VA - 200 amounting to 898 entries in total. The goal is to predict the heart disease class from different attributes and measurements for each subject. The original dataset contains 76 different features, but only a subset of 14 of them have been used in published experiments.

2.1 Features description

The medical meaning of all 14 used features is given bellow:

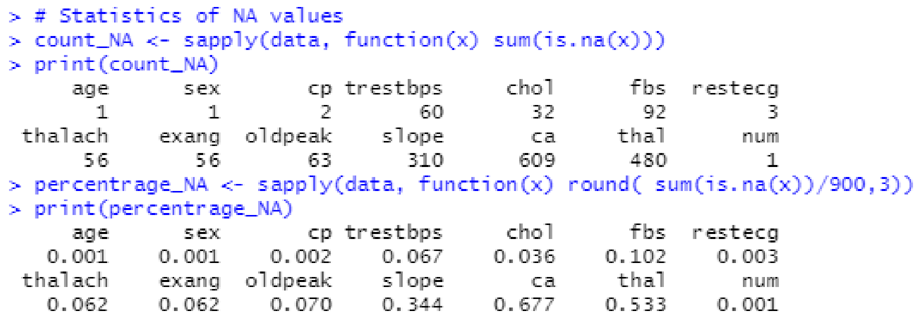
1. age: age in years
2. sex: sex (1 = male; 0 = female)
3. cp: chest pain type (1 = typical angina; 2 = atypical angina; 3 = non-anginal pain; 4 = asymptomatic)
4. trestbps: resting blood pressure (in mm Hg on admission to the hospital)
5. chol: serum cholesterol in mg/dl
6. fbs: (fasting blood sugar $>$ 120 mg/dl) (1 = true; 0 = false)
7. restecg: resting electrocardiographic results
   - Value 0: normal
   - Value 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of $>$ 0.05 mV)
   - Value 2: showing probable or definite left ventricular hypertrophy by Estes' criteria
8. thalach: maximum heart rate achieved
9. exang: exercise induced angina (1 = yes; 0 = no)
10. oldpeak = ST depression induced by exercise relative to rest
11. slope: the slope of the peak exercise ST segment (1 = upsloping; 2 = flat; 3 = downsloping)
12. ca: number of major vessels (0-3) colored by flouroscopy
13. thal: 3 = normal; 6 = fixed defect; 7 = reversible defect
14. num: diagnosis of heart disease (angiographic disease status). The class values range from 1 to 4 and should correspond to the four heart failure medical stages [2], although this information is not available in the dataset description.

2.2 Dataset generation

Since all available processed files have binary class values to indicate presence or absence of heart disease, in this project the raw unprocessed files from each hospital are used. Each file contains 10 rows of features per subject which are flattened and transformed into comma separated files. The files for each hospital (Cleveland, Switzerland, Hungarian and V.A.) are merged and represent the working dataset.

3 Data cleaning and filtering

The missing values in the original dataset are represented as -9. Firstly, they are replaced as missing values and their amount by column is counted. Figure 1 shows that the columns 'slope', 'ca' and 'thal' have a high percentage of missing values of 34%, 67% and 53% respectively. These columns are removed completely, and the remaining rows that contain missing values in other columns are removed after.



```
> # Statistics of NA values
> count_NA <- sapply(data, function(x) sum(is.na(x)))
> print(count_NA)
     age      sex       cp trestbps     chol      fbs  restecg 
       1        1        2       60       32       92        3 
 thalach    exang  oldpeak    slope       ca     thal      num 
      56       56       63      310      609      480        1 
> percentrage_NA <- sapply(data, function(x) round( sum(is.na(x))/900,3))
> print(percentrage_NA)
     age      sex       cp trestbps     chol      fbs  restecg 
   0.001    0.001    0.002    0.067    0.036    0.102    0.003 
 thalach    exang  oldpeak    slope       ca     thal      num 
   0.062    0.062    0.070    0.344    0.677    0.533    0.001 
```

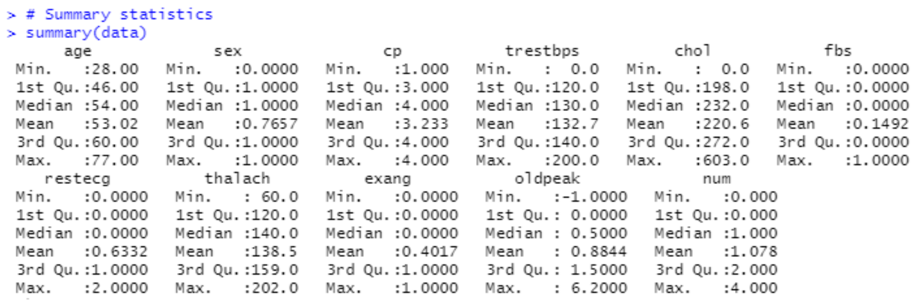
Figure 1: Number and ratio of missing values by column

4 Summary statistics

The data summary before normalization is shown in Figure 2.

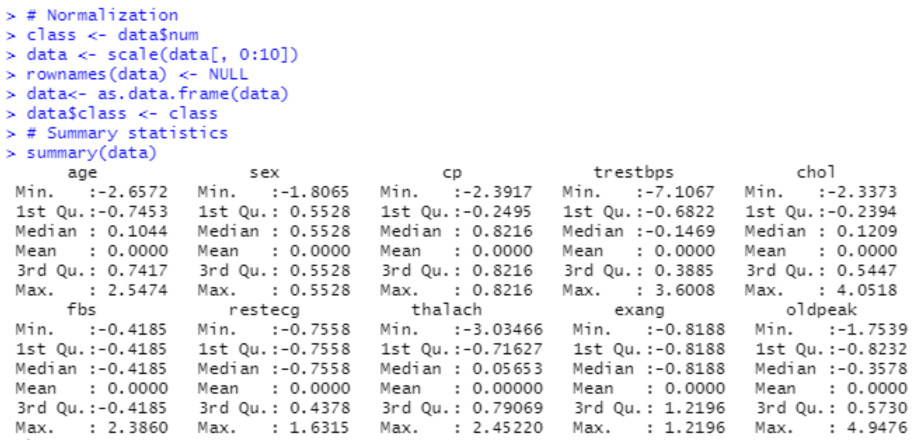
The data is normalized using the scale function and the summary after normalization is shown in Figure 3. The num attribute is not normalized as it represents the target class.

Summary statistics as bar-plots for several features are given in Figure 4. It can be seen that the age of the subjects ranges from 28 to 77 with more subjects concentrated in the range 50-65. The dataset is unbalanced by gender, with male subjects amounting to three times more than the female ones. The class distribution is also unbalanced, with the large portion of the dataset representing healthy individuals. Another categorical feature is the chest pain symptom reported by the patients. It can be seen that most of the subject have not reported this symptom.



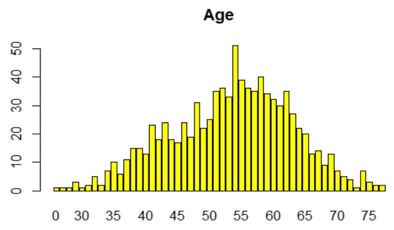
```
> # Summary statistics
> summary(data)
      age             sex                cp            trestbps          chol             fbs        
 Min.   :28.00   Min.   :0.0000   Min.   :1.000   Min.   :  0.0   Min.   :  0.0   Min.   :0.0000  
 1st Qu.:46.00   1st Qu.:1.0000   1st Qu.:3.000   1st Qu.:120.0   1st Qu.:198.0   1st Qu.:0.0000  
 Median :54.00   Median :1.0000   Median :4.000   Median :130.0   Median :232.0   Median :0.0000  
 Mean   :53.02   Mean   :0.7657   Mean   :3.233   Mean   :132.7   Mean   :220.6   Mean   :0.1492  
 3rd Qu.:60.00   3rd Qu.:1.0000   3rd Qu.:4.000   3rd Qu.:140.0   3rd Qu.:272.0   3rd Qu.:0.0000  
 Max.   :77.00   Max.   :1.0000   Max.   :4.000   Max.   :200.0   Max.   :603.0   Max.   :1.0000  
    restecg          thalach          exang           oldpeak             num       
 Min.   :0.0000   Min.   : 60.0   Min.   :0.0000   Min.   :-1.0000   Min.   :0.000  
 1st Qu.:0.0000   1st Qu.:120.0   1st Qu.:0.0000   1st Qu.: 0.0000   1st Qu.:0.000  
 Median :0.0000   Median :140.0   Median :0.0000   Median : 0.5000   Median :1.000  
 Mean   :0.6332   Mean   :138.5   Mean   :0.4017   Mean   : 0.8844   Mean   :1.078  
 3rd Qu.:1.0000   3rd Qu.:159.0   3rd Qu.:1.0000   3rd Qu.: 1.5000   3rd Qu.:2.000  
 Max.   :2.0000   Max.   :202.0   Max.   :1.0000   Max.   : 6.2000   Max.   :4.000  
```

Figure 2: Data summary

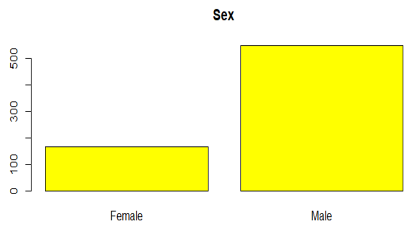


```
> # Normalization
> class <- data$num
> data <- scale(data[, 0:10])
> rownames(data) <- NULL
> data<- as.data.frame(data)
> data$class <- class
> # Summary statistics
> summary(data)
      age               sex                cp               trestbps             chol        
 Min.   :-2.6572   Min.   :-1.8065   Min.   :-2.3917   Min.   :-7.1067   Min.   :-2.3373  
 1st Qu.:-0.7453   1st Qu.: 0.5528   1st Qu.:-0.2495   1st Qu.:-0.6822   1st Qu.:-0.2394  
 Median : 0.1044   Median : 0.5528   Median : 0.8216   Median :-0.1469   Median : 0.1209  
 Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000  
 3rd Qu.: 0.7417   3rd Qu.: 0.5528   3rd Qu.: 0.8216   3rd Qu.: 0.3885   3rd Qu.: 0.5447  
 Max.   : 2.5474   Max.   : 0.5528   Max.   : 0.8216   Max.   : 3.6008   Max.   : 4.0518  
      fbs              restecg            thalach              exang             oldpeak       
 Min.   :-0.4185   Min.   :-0.7558   Min.   :-3.03466   Min.   :-0.8188   Min.   :-1.7539  
 1st Qu.:-0.4185   1st Qu.:-0.7558   1st Qu.:-0.71627   1st Qu.:-0.8188   1st Qu.:-0.8232  
 Median :-0.4185   Median :-0.7558   Median : 0.05653   Median :-0.8188   Median :-0.3578  
 Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.0000  
 3rd Qu.:-0.4185   3rd Qu.: 0.4378   3rd Qu.: 0.79069   3rd Qu.: 1.2196   3rd Qu.: 0.5730  
 Max.   : 2.3860   Max.   : 1.6315   Max.   : 2.45220   Max.   : 1.2196   Max.   : 4.9476  
```

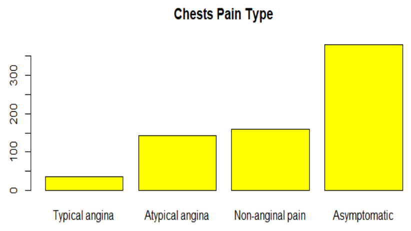
Figure 3: Normalized data summary

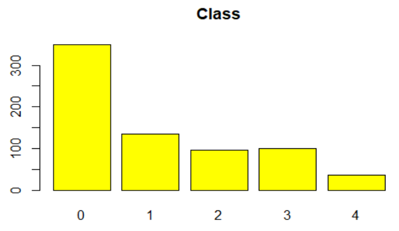


(a) Age distribution



(b) Sex distribution





(c) Chest pain type distribution

(d) Class distribution

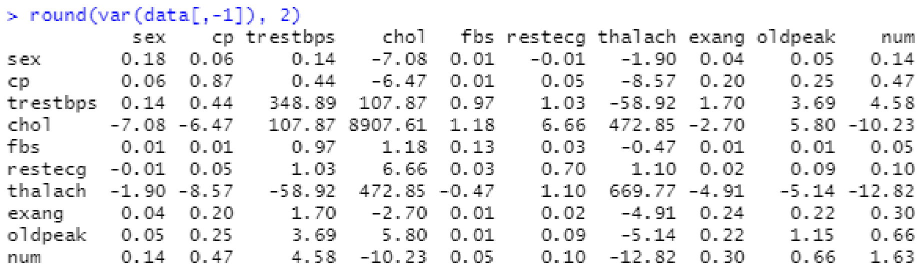
Figure 4: Summary statistics

5 Feature covariance and correlation

Covariance between two variables is defined as:

$$cov_{x,y} = \frac{\sum(x-\bar{x})(y-\bar{y})}{N-1}$$

where N is the number of data values. It measures the relationship between two variables. Figure 5 shows the variance - covariance matrix for all variables. Along the main diagonal are the variance values for each feature before normalization.



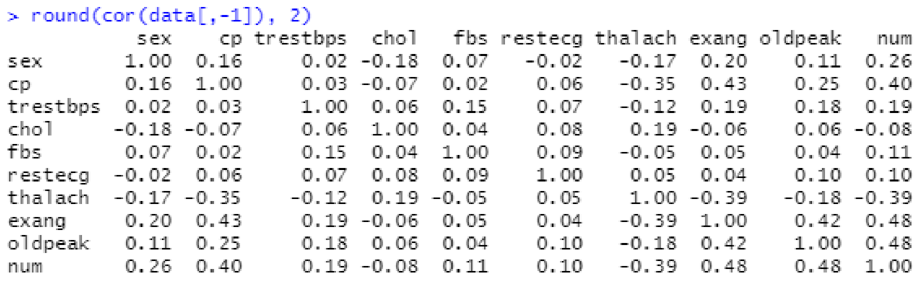
```
> round(var(data[,-1]), 2)
           sex    cp trestbps    chol   fbs restecg thalach exang oldpeak    num
sex       0.18  0.06     0.14   -7.08  0.01   -0.01   -1.90  0.04    0.05   0.14
cp        0.06  0.87     0.44   -6.47  0.01    0.05   -8.57  0.20    0.25   0.47
trestbps  0.14  0.44   348.89  107.87  0.97    1.03  -58.92  1.70    3.69   4.58
chol     -7.08 -6.47   107.87 8907.61  1.18    6.66  472.85 -2.70    5.80 -10.23
fbs       0.01  0.01     0.97    1.18  0.13    0.03   -0.47  0.01    0.01   0.05
restecg  -0.01  0.05     1.03    6.66  0.03    0.70    1.10  0.02    0.09   0.10
thalach  -1.90 -8.57   -58.92  472.85 -0.47    1.10  669.77 -4.91   -5.14 -12.82
exang     0.04  0.20     1.70   -2.70  0.01    0.02   -4.91  0.24    0.22   0.30
oldpeak   0.05  0.25     3.69    5.80  0.01    0.09   -5.14  0.22    1.15   0.66
num       0.14  0.47     4.58  -10.23  0.05    0.10  -12.82  0.30    0.66   1.63
```

Figure 5: Variance - covariance matrix

Correlation coefficient is defined as:

$$r = \frac{\sum(x-\bar{x})(y-\bar{y})}{\sqrt{\sum(x-\bar{x})^2\sum(y-\bar{y})^2}}$$

A correlation coefficient value close to 1 implies a strong positive linear relationship, whereas value close to -1 implies negative linear relationship. Figure 6 shows the correlation values between the features.



```
> round(cor(data[,-1]), 2)
           sex    cp trestbps  chol   fbs restecg thalach exang oldpeak   num
sex       1.00  0.16     0.02 -0.18  0.07   -0.02   -0.17  0.20    0.11  0.26
cp        0.16  1.00     0.03 -0.07  0.02    0.06   -0.35  0.43    0.25  0.40
trestbps  0.02  0.03     1.00  0.06  0.15    0.07   -0.12  0.19    0.18  0.19
chol     -0.18 -0.07     0.06  1.00  0.04    0.08    0.19 -0.06    0.06 -0.08
fbs       0.07  0.02     0.15  0.04  1.00    0.09   -0.05  0.05    0.04  0.11
restecg  -0.02  0.06     0.07  0.08  0.09    1.00    0.05  0.04    0.10  0.10
thalach  -0.17 -0.35    -0.12  0.19 -0.05    0.05    1.00 -0.39   -0.18 -0.39
exang     0.20  0.43     0.19 -0.06  0.05    0.04   -0.39  1.00    0.42  0.48
oldpeak   0.11  0.25     0.18  0.06  0.04    0.10   -0.18  0.42    1.00  0.48
num       0.26  0.40     0.19 -0.08  0.11    0.10   -0.39  0.48    0.48  1.00
```

Figure 6: Correlation matrix

The correlation matrix is visualised in Figure 7 in order to detect which features are correlated. The correlation is the highest between 'cp' and 'exang' and 'exang' with 'oldpeak', whereas the negative correlation value is highest between 'cp' and 'thalach' and 'thalach' with 'exang'. However, the correlation coefficient is not higher than 0.5 (or lower than -0.5) which shows that there is no linear correlation between features. Therefore, all features can be used in the modelling.

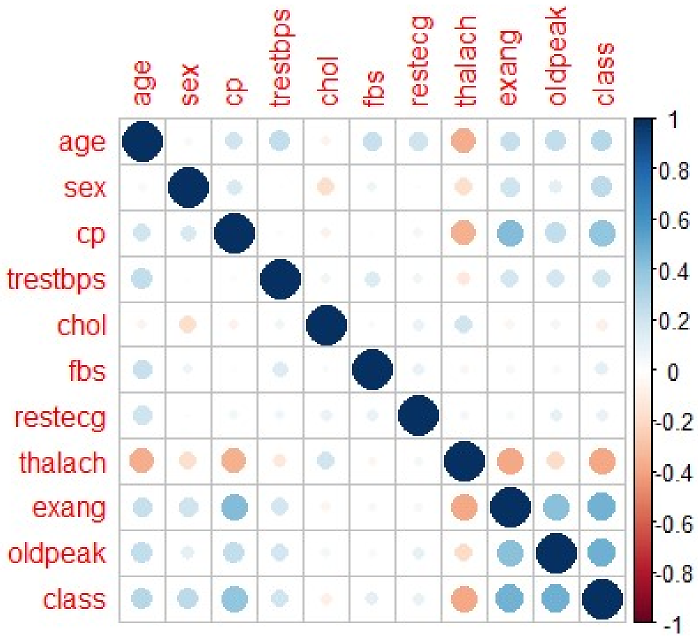


Figure 7: Correlation plot

6 One-vs-All Generalized Linear Modelling

One-vs-all approach is training a binary classifier for each class against all other classes and combining the prediction among all classifiers. Because of the nature of the dataset, the healthy class can be used in binary classification against other classes in order for the models to learn the class specific differences. Each binary model is trained on a subset of the data comprised of the given class against the healthy samples in a binary fashion.

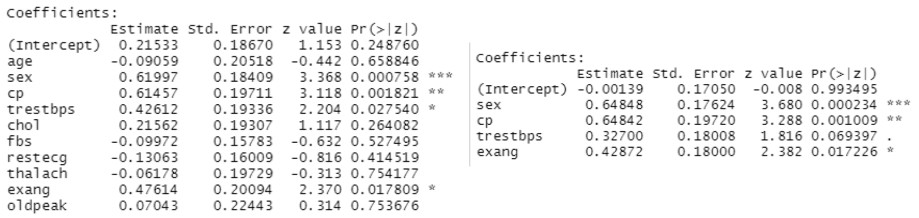
6.1 Dataset balancing and feature selection

Because of the nature of the dataset, 4 classifiers are constructed against the data from the class 0 healthy samples. For each class, the number of samples from that class is labeled with 1 and an equal number healthy samples labeled with 0.

For each of the classifiers, two models are compared: full and reduced. The full model consists of all variables in the dataset, whereas the reduced model consists only of the variables chosen trough cross validation and variance reduction. For the first classifier this procedure selects 4 variables, for the second and third class 5 and for the fourth class only one variable. The accuracy values of the reduced model for all classifiers are lower than the full models. The train - test split for each classifier is 70%.

6.2 Class 1

The data for the first classifier consists of 135 samples from class number 1 and an equal number of healthy samples. This data is then divided into a train set of 190 samples and test set of 80 samples using 70-30 proportion.



```
Coefficients:
            Estimate Std. Error z value Pr(>|z|)    
(Intercept)  0.21533    0.18670   1.153 0.248760    
age         -0.09059    0.20518  -0.442 0.658846    
sex          0.61997    0.18409   3.368 0.000758 ***
cp           0.61457    0.19711   3.118 0.001821 ** 
trestbps     0.42612    0.19336   2.204 0.027540 *  
chol         0.21562    0.19307   1.117 0.264082    
fbs         -0.09972    0.15783  -0.632 0.527495    
restecg     -0.13063    0.16009  -0.816 0.414519    
thalach     -0.06178    0.19729  -0.313 0.754177    
exang        0.47614    0.20094   2.370 0.017809 *  
oldpeak      0.07043    0.22443   0.314 0.753676    
```

```
Coefficients:
            Estimate Std. Error z value Pr(>|z|)    
(Intercept) -0.00139    0.17050  -0.008 0.993495    
sex          0.64848    0.17624   3.680 0.000234 ***
cp           0.64842    0.19720   3.288 0.001009 ** 
trestbps     0.32700    0.18008   1.816 0.069397 .  
exang        0.42872    0.18000   2.382 0.017226 *  
```

Figure 8: Models for class 1

The full model shown on Figure 8 with all features assigns the highest importance to the following features: 'sex', 'cp', 'trestbps' and 'exang'. The accuracy obtained from the full model is 71,25% and the area under the ROC curve shown on Figure 9 is 0.713. The number of false positives is 15 whereas false negatives 8.

The reduced model also shown on Figure 8 consist only of the features 'sex', 'cp', 'trestbps' and 'exang'. The accuracy obtained by this model is 70.00% and the number of both false positives and false negatives is 12. The ROC curve is shown on Figure 9.

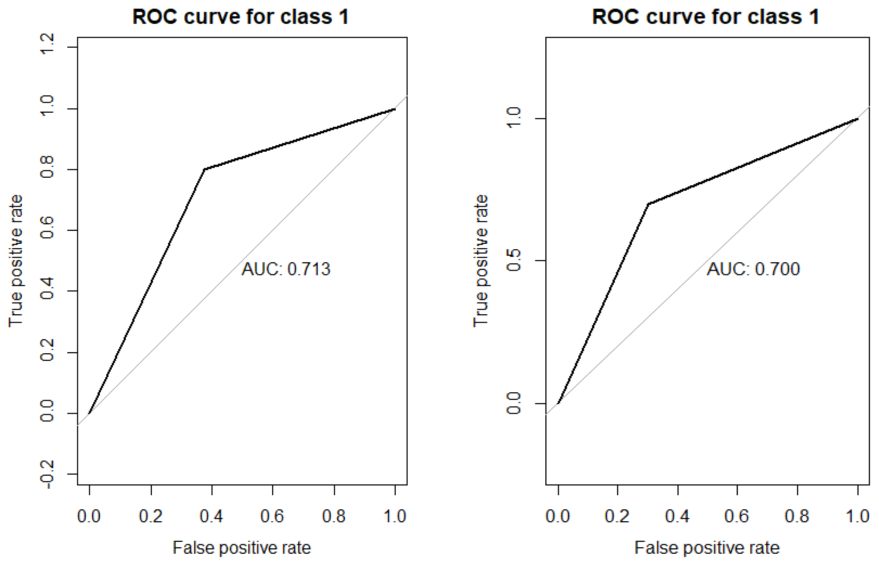
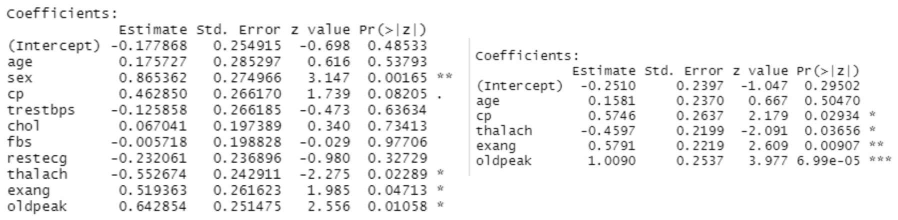


Figure 9: ROC curves for class 1

6.3 Class 2

For the second class the data consists of 97 samples from both the second class and the healthy samples. The 70-30 train and test dataset division results in 136 samples for training and 58 for the testing phase.



```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.177868   0.254915  -0.698  0.48533
age          0.175727   0.285297   0.616  0.53793
sex          0.865362   0.274966   3.147  0.00165 **
cp           0.462850   0.266170   1.739  0.08205 .
trestbps    -0.125858   0.266185  -0.473  0.63634
chol         0.067041   0.197389   0.340  0.73413
fbs         -0.005718   0.198828  -0.029  0.97706
restecg     -0.232061   0.236896  -0.980  0.32729
thalach     -0.552674   0.242911  -2.275  0.02289 *
exang        0.519363   0.261623   1.985  0.04713 *
oldpeak      0.642854   0.251475   2.556  0.01058 *
```

```
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -0.2510     0.2397  -1.047  0.29502
age           0.1581     0.2370   0.667  0.50470
cp            0.5746     0.2637   2.179  0.02934 *
thalach      -0.4597     0.2199  -2.091  0.03656 *
exang         0.5791     0.2219   2.609  0.00907 **
oldpeak       1.0090     0.2537   3.977 6.99e-05 ***
```

Figure 10: Models for class 2

In the full model shown on Figure 10 the obtained accuracy is 77.58% with 11 false positives and 2 false negatives. The area under the ROC curve shown on Figure 11 is 0.776. The most significant features found are ’sex’, ’thalch’, ’exang’, ’oldpeak’ and ’cp’.

On the other hand, the reduced model also shown on Figure 10 is only using the features ’age’, ’cp’, ’thelach’, ’exang’ and ’oldpeak’. The obtained accuracy by this model is 67.24% with 10 false positives and 9 false negatives. The ROC curve is shown on Figure 11.

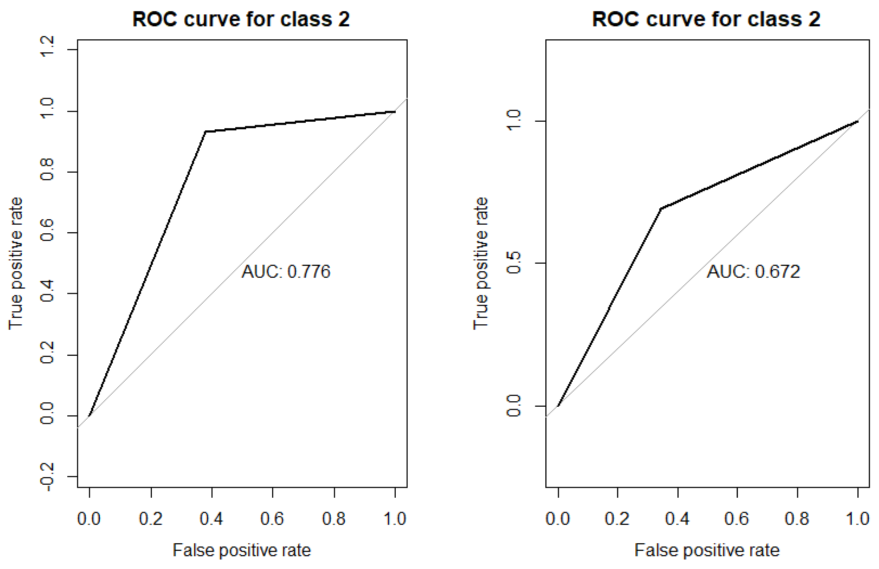
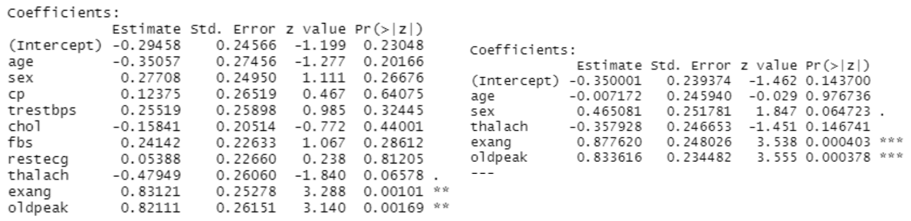


Figure 11: ROC curves for class 2

6.4 Class 3

The number of samples available for the third class is 100. From the total 200 samples for the third classifier, 140 are used for training and 60 for testing.



```
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.29458    0.24566  -1.199  0.23048
age         -0.35057    0.27456  -1.277  0.20166
sex          0.27708    0.24950   1.111  0.26676
cp           0.12375    0.26519   0.467  0.64075
trestbps     0.25519    0.25898   0.985  0.32445
chol        -0.15841    0.20514  -0.772  0.44001
fbs          0.24142    0.22633   1.067  0.28612
restecg      0.05388    0.22660   0.238  0.81205
thalach     -0.47949    0.26060  -1.840  0.06578 .
exang        0.83121    0.25278   3.288  0.00101 **
oldpeak      0.82111    0.26151   3.140  0.00169 **
```

```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.350001   0.239374  -1.462 0.143700
age         -0.007172   0.245940  -0.029 0.976736
sex          0.465081   0.251781   1.847 0.064723 .
thalach     -0.357928   0.246653  -1.451 0.146741
exang        0.877620   0.248026   3.538 0.000403 ***
oldpeak      0.833616   0.234482   3.555 0.000378 ***
---
```

Figure 12: Models for class 3

The accuracy obtained with the full model shown on Figure 12 is 85%. The number of false positives is 5 and false negatives 4. The area under the ROC curve shown on Figure 13 is 0.850. This model shows high significance values for the features: ’oldpeak’, ’chol’, and ’fbs’.

The reduced model consists of the features ’age’, ’sec’, ’thalach’, ’exang’ and ’oldpeak’. The accuracy obtained is 83.33% with 4 false positives and 6 false negatives.

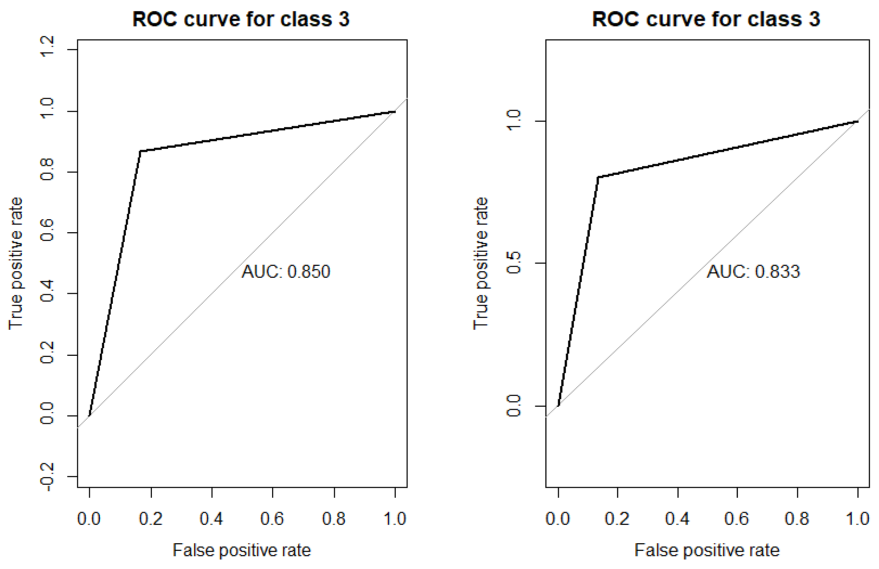
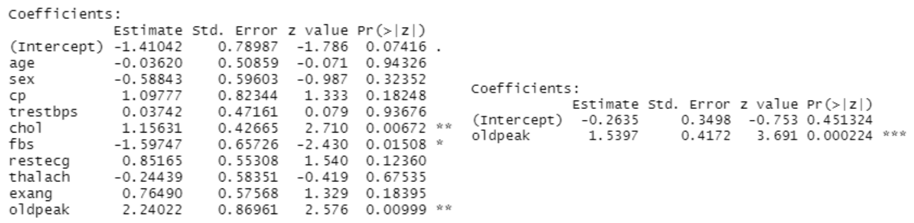


Figure 13: ROC curves for class 3

6.5 Class 4

The forth classifier is trained on 49 samples and tested on 21. The number of samples available from this class is 36.



```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.41042    0.78987  -1.786  0.07416 .
age         -0.03620    0.50859  -0.071  0.94326
sex         -0.58843    0.59603  -0.987  0.32352
cp           1.09777    0.82344   1.333  0.18248
trestbps     0.03742    0.47161   0.079  0.93676
chol         1.15631    0.42665   2.710  0.00672 **
fbs         -1.59747    0.65726  -2.430  0.01508 *
restecg      0.85165    0.55308   1.540  0.12360
thalach     -0.24439    0.58351  -0.419  0.67535
exang        0.76490    0.57568   1.329  0.18395
oldpeak      2.24022    0.86961   2.576  0.00999 **
```

```
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -0.2635     0.3498  -0.753 0.451324
oldpeak       1.5397     0.4172   3.691 0.000224 ***
```

Figure 14: Models for class 4

The full model shown on Figure 14 obtains accuracy of 80.00% and AUC shown on Figure 15 of 0.80. The number of both false positives and false negatives is 2.

The reduced model consists of only one feature - oldpeak. The accuracy obtained by this model is 65% with 6 false positives and 1 false negative. The ROC curve is shown on Figure 15

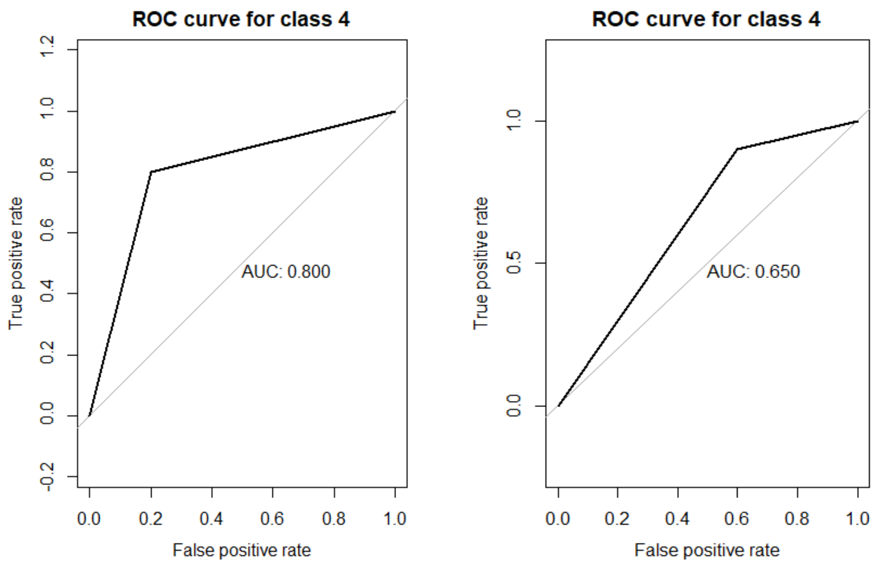
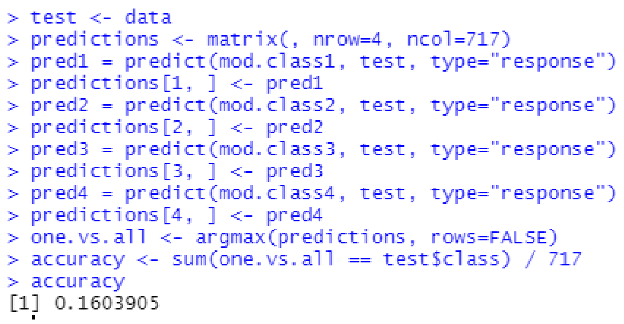


Figure 15: ROC curves for class 4

6.6 Prediction

The final class prediction is calculated as shown in Figure 16. All four classifiers are tested on the whole dataset and the class is assigned to the one which calculates the maximum probability. The overall accuracy is calculated by comparing the whole dataset with the vector of predicted class. When using the full models for all classifiers the overall accuracy is 16.04%. However even though the individual classifiers perform better when using all of the variables, the overall accuracy when using the individual reduced models is 17.15%.

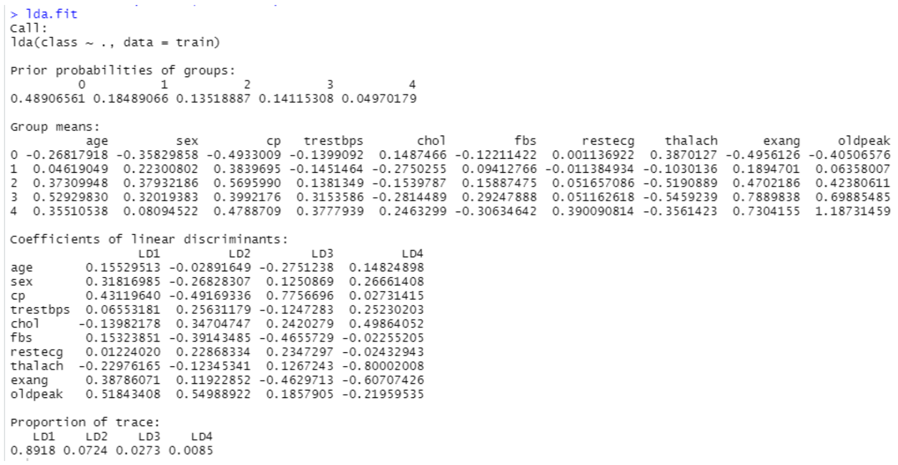


```
> test <- data
> predictions <- matrix(, nrow=4, ncol=717)
> pred1 = predict(mod.class1, test, type="response")
> predictions[1, ] <- pred1
> pred2 = predict(mod.class2, test, type="response")
> predictions[2, ] <- pred2
> pred3 = predict(mod.class3, test, type="response")
> predictions[3, ] <- pred3
> pred4 = predict(mod.class4, test, type="response")
> predictions[4, ] <- pred4
> one.vs.all <- argmax(predictions, rows=FALSE)
> accuracy <- sum(one.vs.all == test$class) / 717
> accuracy
[1] 0.1603905
```

Figure 16: Prediction calculation

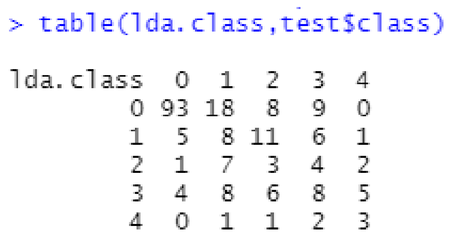
7 Linear Discriminant Analysis

One of the next models used in order to differentiate between the five different classes is LDA. This model preforms dimensionality reduction while maximizing the separability between the given classes by maximizing the distance between classes and minimizing the variance. Even though this dataset does not have a large number of features, this technique has shown higher accuracy in the testing phase compared with others. In linear discriminant analysis a coefficient is computed for each class which is used to generate the prediction. Figure 53 shows the correlation plots between the coefficients. Figure 17 shows the feature means for each class and the values obtained for the coefficients. The confusion matrix is shown on Figure 18 and the accuracy values per class are: 0 (healthy) - 90.29%, 1 - 19.04%, 2 - 10%, 3 - 27.58% and 4 - 27.27%. From the confusion matrix we can see that the model accurately detects the healthy instances, but does not differentiate well between the other four classes. The overall accuracy for this model among all classes is 53.74%.



```
> lda.fit
Call:
lda(class ~ ., data = train)

Prior probabilities of groups:
         0          1          2          3          4 
0.48906561 0.18489066 0.13518887 0.14115308 0.04970179 

Group means:
          age         sex         cp   trestbps       chol         fbs      restecg    thalach      exang     oldpeak
0 -0.26817918 -0.35829858 -0.4933009 -0.1399092  0.1487466 -0.12211422  0.001136922  0.3870127 -0.4956126 -0.40506576
1  0.04619049  0.22300802  0.3839695 -0.1451464 -0.2750255  0.09412766 -0.011384934 -0.1030136  0.1894701  0.06358007
2  0.37309948  0.37932186  0.5695990  0.1381349 -0.1539787  0.15887475  0.051657086 -0.5190889  0.4702186  0.42380611
3  0.52929830  0.32019383  0.3992176  0.3153586 -0.2814489  0.29247888  0.051162618 -0.5459239  0.7889838  0.69885485
4  0.35510538  0.08094522  0.4788709  0.3777939  0.2463299 -0.30634642  0.390090814 -0.3561423  0.7304155  1.18731459

Coefficients of linear discriminants:
                 LD1         LD2        LD3         LD4
age       0.15529513 -0.02891649 -0.2751238  0.14824898
sex       0.31816985 -0.26828307  0.1250869  0.26661408
cp        0.43119640 -0.49169336  0.7756696  0.02731415
trestbps  0.06553181  0.25631179 -0.1247283  0.25230203
chol     -0.13982178  0.34704747  0.2420279  0.49864052
fbs       0.15323851 -0.39143485 -0.4655729 -0.02255205
restecg   0.01224020  0.22868334  0.2347297 -0.02432943
thalach  -0.22976165 -0.12345341  0.1267243 -0.80002008
exang     0.38786071  0.11922852 -0.4629713 -0.60707426
oldpeak   0.51843408  0.54988922  0.1857905 -0.21959535

Proportion of trace:
   LD1    LD2    LD3    LD4 
0.8918 0.0724 0.0273 0.0085 
```

Figure 17: Linear Discriminant Analysis coefficient values

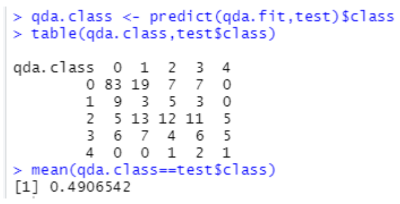


```
> table(lda.class,test$class)

lda.class  0  1  2  3  4
        0 93 18  8  9  0
        1  5  8 11  6  1
        2  1  7  3  4  2
        3  4  8  6  8  5
        4  0  1  1  2  3
```

Figure 18: LDA confusion matrix

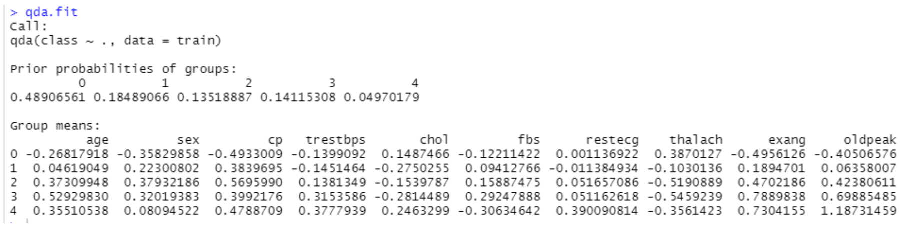
8 Quadratic Discriminant Analysis

Another model used is the Quadratic Discriminant Analysis which generates quadratic functions for class separation. The overall accuracy when using this model is 49.06% which is lower than the accuracy obtained with LDA. The class specific accuracies are: 0 (healthy) - 80.58%, 1 - 7.14%, 2 - 41.37%, 3 - 20.68%, 4 - 10%. Similarly, the model has the highest accuracy in predicting the healthy samples. These values are obtained from the confusion matrix shown in Figure 19. Figure 20 shows the values of the group means per feature.



```
> qda.class <- predict(qda.fit,test)$class
> table(qda.class,test$class)

qda.class  0  1  2  3  4
        0 83 19  7  7  0
        1  9  3  5  3  0
        2  5 13 12 11  5
        3  6  7  4  6  5
        4  0  0  1  2  1
> mean(qda.class==test$class)
[1] 0.4906542
```

Figure 19: QDA confusion matrix

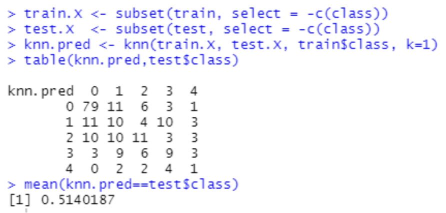


```
> qda.fit
Call:
qda(class ~ ., data = train)

Prior probabilities of groups:
         0          1          2          3          4 
0.48906561 0.18489066 0.13518887 0.14115308 0.04970179 

Group means:
          age         sex         cp   trestbps       chol         fbs      restecg    thalach      exang     oldpeak
0 -0.26817918 -0.35829858 -0.4933009 -0.1399092  0.1487466 -0.12211422  0.001136922  0.3870127 -0.4956126 -0.40506576
1  0.04619049  0.22300802  0.3839695 -0.1451464 -0.2750255  0.09412766 -0.011384934 -0.1030136  0.1894701  0.06358007
2  0.37309948  0.37932186  0.5695990  0.1381349 -0.1539787  0.15887475  0.051657086 -0.5190889  0.4702186  0.42380611
3  0.52929830  0.32019383  0.3992176  0.3153586 -0.2814489  0.29247888  0.051162618 -0.5459239  0.7889838  0.69885485
4  0.35510538  0.08094522  0.4788709  0.3777939  0.2463299 -0.30634642  0.390090814 -0.3561423  0.7304155  1.18731459
```

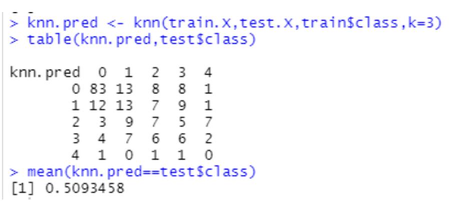
Figure 20: Quadratic Discriminant Analysis values

9 KNN

Another model used for multiclass classification is K-Nearest Neighbours. This method assigns the class to each of the training samples according to the class of it's K nearest neighbours in the feature space. On Figure 21 the confusion matrices for prediction when using 1 and 3 neighbours, respectively are shown. In this instance, the overall accuracy generally decreases when increasing the number of neighbours. The overall accuracy obtained with K = 1 is 51.40%, whereas for K = 3 it is 50.95%. With further increase of the number of neighbours, the obtained accuracy is 49.06% for K = 5 and 48.59% for K = 8.



```
> train.X <- subset(train, select = -c(class))
> test.X  <- subset(test, select = -c(class))
> knn.pred <- knn(train.X, test.X, train$class, k=1)
> table(knn.pred,test$class)

knn.pred  0  1  2  3  4
       0 79 11  6  3  1
       1 11 10  4 10  3
       2 10 10 11  3  3
       3  3  9  6  9  3
       4  0  2  2  4  1
> mean(knn.pred==test$class)
[1] 0.5140187
```



```
> knn.pred <- knn(train.X,test.X,train$class,k=3)
> table(knn.pred,test$class)

knn.pred  0  1  2  3  4
       0 83 13  8  8  1
       1 12 13  7  9  1
       2  3  9  7  5  7
       3  4  7  6  6  2
       4  1  0  1  1  0
> mean(knn.pred==test$class)
[1] 0.5093458
```

Figure 21: Confusion matrices and accuracy for KNN with N = 1 and N = 3

10 Binary classification

The results from the One-vs-All modeling show that models can be generated to differentiate well between pairs of the classes in this dataset, but the combined prediction has very poor accuracy. The discriminant analysis methods used (both LDA and QDA) further suggest that the accuracy is the highest in the detection of the healthy samples. These are the motivating factors to continue a binary analysis of the data and differentiate between healthy and disease samples. This is done by combining all disease status classes: 1-4 into a single class. The generated optimal model can be used for screening and detection of a healthy against a disease state, but further analysis of the patient would be required for accurate diagnosis of the disease type. By combining the disease classes into one, the dataset is balanced for the binary classification task. In order to generate the optimal model, several approaches are performed among which: resampling methods, variable selection, statistics comparison among models and shrinkage models.

10.1 Resampling Methods

Resampling methods "involve repeatedly drawing samples from a training set and refitting a model of interest on each sample in order to obtain additional information about the fitted model" [3]. Cross-validation, the one of the most common methods, is used along with its variation, k-fold cross-validation. Missclassification error (Err) is computed in order to calculate training and validation error. We compute

$$CV_{(K)} = \sum_{k=1}^{K} \frac{n_k}{n} Err_k$$

where

$$Err_k = \frac{1}{n_k} \sum_{i \in C_k} I(y_i \neq \hat{y}_i)$$

Validation error is used as the biased estimate of the test error. It is an estimate because we do not have available large test data set that would be used only as the test set and that would provide the exact test error value. Also, it is biased because it depends on exactly which samples are included in the training set and which samples are included in the validation set; moreover, "statistical methods tend to perform worse when trained on fewer observations, this suggests that the validation set error rate may tend to overestimate the test error rate for the model fit on the entire data set" [3].

10.1.1 Cross-Validation

The validation process includes randomly splitting the data into two different data sets, training and validation, where the model is fitted on the training set and tested on the validation set. On the Figure 22, by using the cross-validation method, it's shown that the model that has the smallest validation error is the model which uses 8 variables. Red point on the plot denotes the number of variables where the validation error has the smallest value.

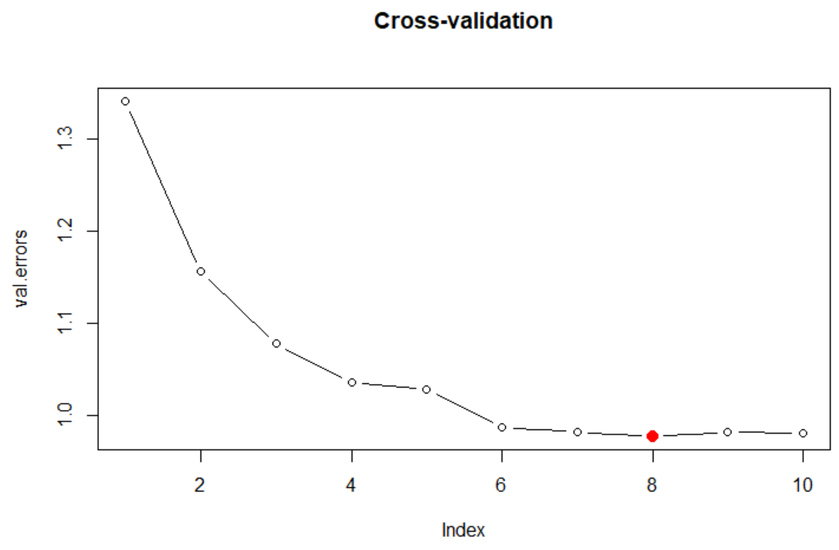


Figure 22: Cross validation

The Figure 23 shows the difference between validation error and training error. The line in blue color, which represents the training error, is a lot smaller that the black line, which represents the validation error. Also, it can be seen that validation error rapidly increases towards the end when more than 8 variables are used.

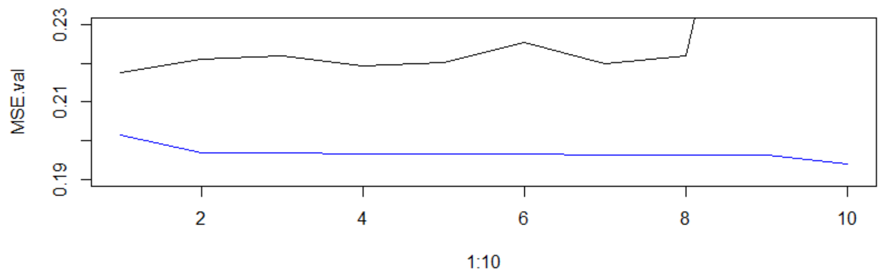
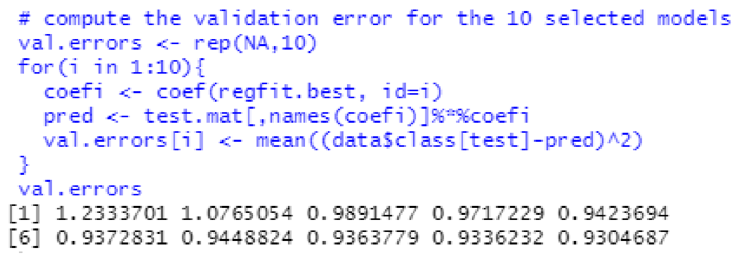


Figure 23: Feature selection comparison

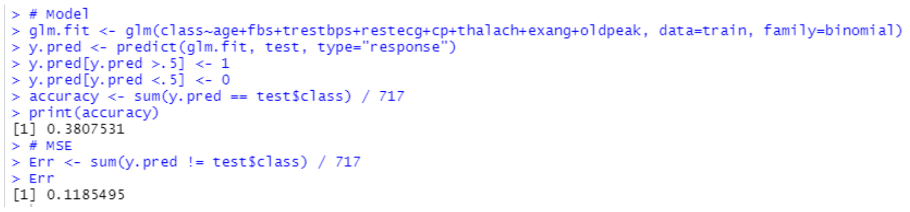
The error that we are interested in is validation error since it is used to estimate the test error and we want it to be as low as possible. On the Figure 24 we can see that validation error is decreasing. It is not important that our model performs good on the already seen training data if it does not classify well the unseen data. The test accuracy should be as high as possible and the test error very small because that suggests that the model is making correct predictions.



```
  # compute the validation error for the 10 selected models
  val.errors <- rep(NA,10)
  for(i in 1:10){
    coefi <- coef(regfit.best, id=i)
    pred <- test.mat[,names(coefi)]%*%coefi
    val.errors[i] <- mean((data$class[test]-pred)^2)
  }
  val.errors
[1] 1.2333701 1.0765054 0.9891477 0.9717229 0.9423694
[6] 0.9372831 0.9448824 0.9363779 0.9336232 0.9304687
```

Figure 24: Validation error for 10 selected models

After splitting the data in half and making training and test sets and fitting the model, the accuracy obtained is 38.07% as shown in the Figure 25.



```
> # Model
> glm.fit <- glm(class~age+fbs+trestbps+restecg+cp+thalach+exang+oldpeak, data=train, family=binomial)
> y.pred <- predict(glm.fit, test, type="response")
> y.pred[y.pred >.5] <- 1
> y.pred[y.pred <.5] <- 0
> accuracy <- sum(y.pred == test$class) / 717
> print(accuracy)
[1] 0.3807531
> # MSE
> Err <- sum(y.pred != test$class) / 717
> Err
[1] 0.1185495
```

Figure 25: Cross-Validation Model

10.1.2 10-Fold Cross-Validation

In the K-fold cross-validation approach, the data was randomly divided into 10 approximately equal folds. After dividing the data, the model is fitted on the 9 folds combined, and tested on the remaining 1 fold. The process is repeated 10 times in turn, for each part, and then the results are combined. As shown in the Figure 26, the model with the smallest mean cross-validation error is the model that uses all 10 features.

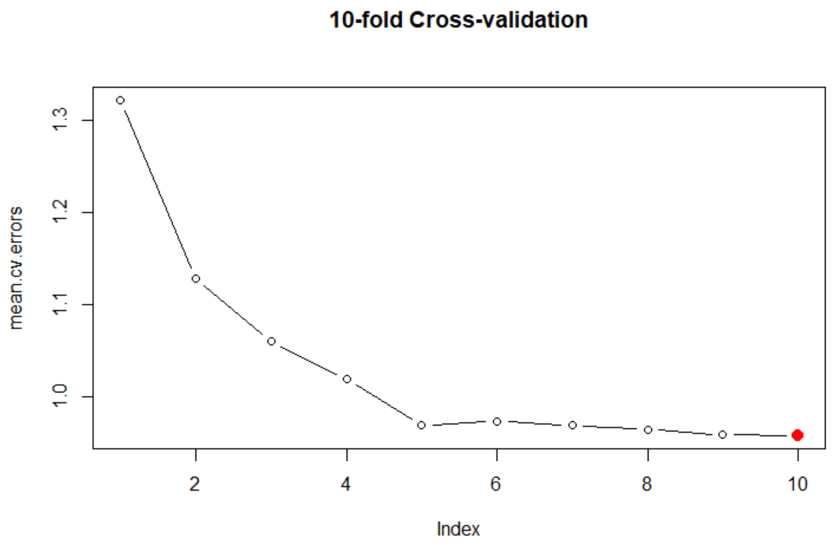


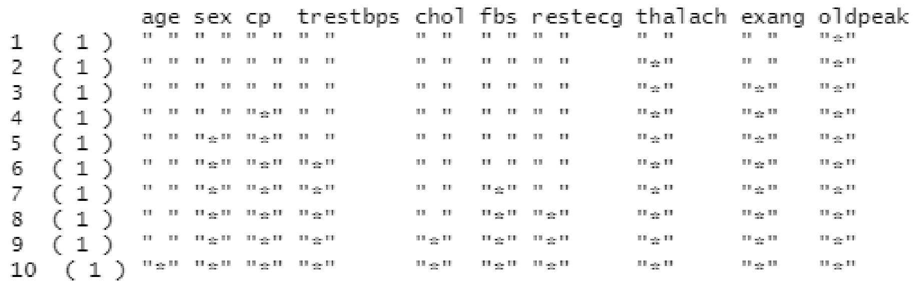
Figure 26: 10 fold cross validation

10.2 Variable selection

Variable selection is a very important process because it possibly reduces the complexity of the model. Sometimes, not all the variables are used in the final model or have influence and role in the model construction. These methods use least squares to fit a linear model that contains a subset of the predictors. For the optimal variable selection several methods are used such as: best subset selection and stepwise selection methods: forward and backward selection.

10.2.1 Best subset selection

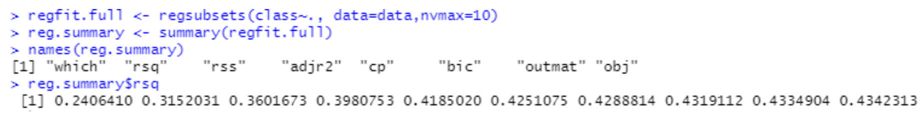
The best subset selection is used for variable selection in order to select the best model that contains a given number of predictors. It considers all the possible models containing the subset of the predictors. The best model is chosen based on the residual sum of squares (RSS). The asterisks ('*') on the Figure 27 indicate that the best five-variable model contains only 'age', 'cp', 'thalach', 'exang', and 'oldpeak'. Removing one or more features may lead to the increase the value of the RSS which lead to the decrease in the value of R2 statistic since these quantities are related to the training error. With this being said, it is easy to see that the model which includes all the features will have the smallest RSS and the largest R2 statistic.



```
           age sex cp  trestbps chol fbs restecg thalach exang oldpeak
1  ( 1 )   " " " " " " " "      " "  " " " "     " "     " "   "*"
2  ( 1 )   " " " " " " " "      " "  " " " "     "*"     " "   "*"
3  ( 1 )   " " " " " " " "      " "  " " " "     "*"     "*"   "*"
4  ( 1 )   " " " " "*" " "      " "  " " " "     "*"     "*"   "*"
5  ( 1 )   " " "*" "*" " "      " "  " " " "     "*"     "*"   "*"
6  ( 1 )   " " "*" "*" "*"      " "  " " " "     "*"     "*"   "*"
7  ( 1 )   " " "*" "*" "*"      " "  "*" " "     "*"     "*"   "*"
8  ( 1 )   " " "*" "*" "*"      " "  "*" "*"     "*"     "*"   "*"
9  ( 1 )   " " "*" "*" "*"      "*"  "*" "*"     "*"     "*"   "*"
10  ( 1 ) "*" "*" "*" "*"      "*"  "*" "*"     "*"     "*"   "*"
```

Figure 27: Subset selection

It can be seen on the Figure 28. that R2 statistic increases for about 20% when all variables are used. When one variable is used, it is 24%, whereas when all variables are used it is 43%. This suggests that R2 statistic increases monotonically when more variables are included.



```
> regfit.full <- regsubsets(class~., data=data,nvmax=10)
> reg.summary <- summary(regfit.full)
> names(reg.summary)
[1] "which"  "rsq"    "rss"    "adjr2"  "cp"     "bic"    "outmat" "obj"
> reg.summary$rsq
 [1] 0.2406410 0.3152031 0.3601673 0.3980753 0.4185020 0.4251075 0.4288814 0.4319112 0.4334904 0.4342313
```

Figure 28: Best subset selection

10.3 Stepwise Selection

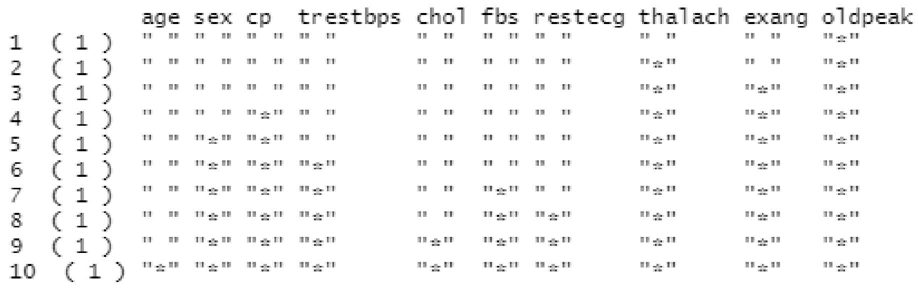
Forward and backward selection are the two alternatives to the best subset selection model that can be applied when the number of predictors is large. They can be computed also by using regsubsets() function by specifying the method to be "forward" or "backward". This way we produce the sequence of nested models which makes it to be the main difference between this procedure and best subset selection method where we consider only the subset of the variables.

10.3.1 Forward selection

Forward stepwise selection is computed the following way:

```
regfit.fwd <- regsubsets(class~.,data=data,nvmax=10,method="forward")
```

This method considers a smaller set of the predictors. It starts with null model where there are no predictors and adds one predictor at the time to the model until it includes all the predictors in the model. Particularly, the variable that is added at the each step is the variable that gives the greatest additional improvement. The same results as in the best subset selection was obtained. The asterisks ('*') on the Figure 29 indicate that the best five-variable model contains only 'age', 'cp', 'thalach', 'exang', and 'oldpeak'.



```
          age sex cp  trestbps chol fbs restecg thalach exang oldpeak
1  ( 1 )  " " " " " " " "      " "  " " " "     " "     " "   "*"
2  ( 1 )  " " " " " " " "      " "  " " " "     "*"     " "   "*"
3  ( 1 )  " " " " " " " "      " "  " " " "     "*"     "*"   "*"
4  ( 1 )  " " " " "*" " "      " "  " " " "     "*"     "*"   "*"
5  ( 1 )  " " "*" "*" " "      " "  " " " "     "*"     "*"   "*"
6  ( 1 )  " " "*" "*" "*"      " "  " " " "     "*"     "*"   "*"
7  ( 1 )  " " "*" "*" "*"      " "  "*" " "     "*"     "*"   "*"
8  ( 1 )  " " "*" "*" "*"      " "  "*" "*"     "*"     "*"   "*"
9  ( 1 )  " " "*" "*" "*"      "*"  "*" "*"     "*"     "*"   "*"
10  ( 1 ) "*" "*" "*" "*"      "*"  "*" "*"     "*"     "*"   "*"
```

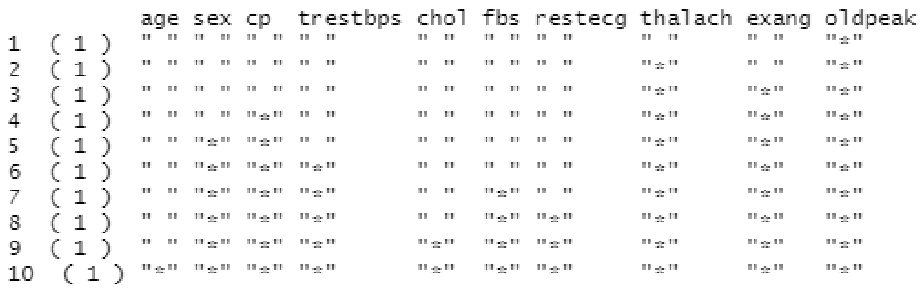
Figure 29: Forward selection

10.3.2 Backward selection

The backward stepwise selection is computed the following way:

```
regfit.bwd <- regsubsets(class~.,data=data,nvmax=10,method="backward")
```

The backward selection is another alternative to the best subset selection and it starts from the full least square model that includes all the predictors. Then, one at the time, it removes the least useful predictors. Since the backward selection requires that the number of samples is larger than the number of the variables, it is appropriate to use it on our dataset and model. It gave the same results as the previous two methods. The asterisks ('*') on the Figure 30 indicate that the best five-variable model contains only 'age', 'cp', 'thalach', 'exang', and 'oldpeak'.



```
          age sex cp  trestbps chol fbs restecg thalach exang oldpeak
1  ( 1 )  " " " " " " " "      " "  " " " "     " "     " "   "*"
2  ( 1 )  " " " " " " " "      " "  " " " "     "*"     " "   "*"
3  ( 1 )  " " " " " " " "      " "  " " " "     "*"     "*"   "*"
4  ( 1 )  " " " " "*" " "      " "  " " " "     "*"     "*"   "*"
5  ( 1 )  " " "*" "*" " "      " "  " " " "     "*"     "*"   "*"
6  ( 1 )  " " "*" "*" "*"      " "  " " " "     "*"     "*"   "*"
7  ( 1 )  " " "*" "*" "*"      " "  "*" " "     "*"     "*"   "*"
8  ( 1 )  " " "*" "*" "*"      " "  "*" "*"     "*"     "*"   "*"
9  ( 1 )  " " "*" "*" "*"      "*"  "*" "*"     "*"     "*"   "*"
10  ( 1 ) "*" "*" "*" "*"      "*"  "*" "*"     "*"     "*"   "*"
```

Figure 30: Backward selection

10.4 Choosing Optimal Model

By plotting RSS, adjusted R2, Cp, and BIC for all the possible models, as shown in the Figure 31, it helps us to decide which model to choose among all the models that we got by performing best subset selection, froward selection, and backward selection.

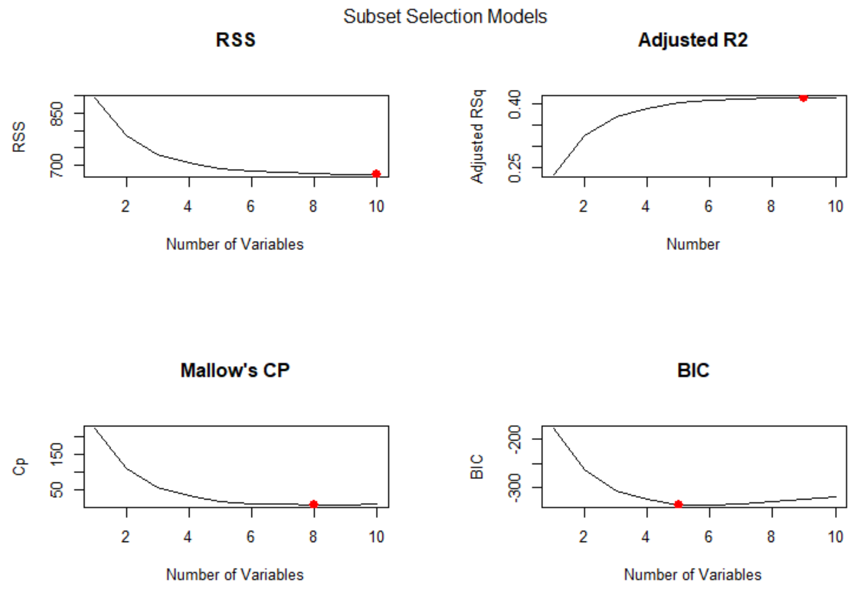
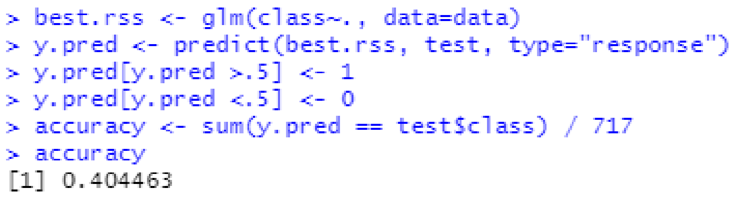


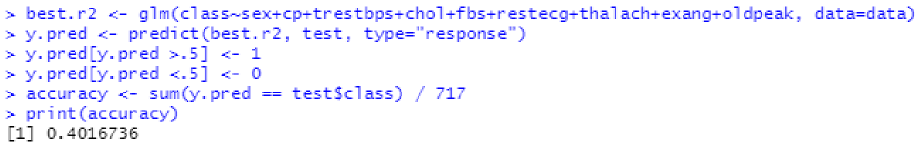
Figure 31: Subset plot

The accuracy of these four models is computed as shown in the Figures 32, 33, 34, and 35. It can be seen that BIC model has the highest accuracy with 41%. Behind him is RSS model with accuracy of 40.45%. Adjusted R2 and Mallow's CP models had accuracy of 40.17% and 39.89% respectively.



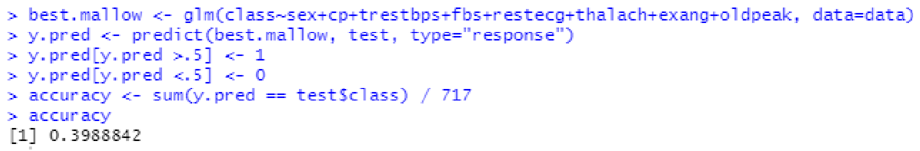
```
> best.rss <- glm(class~., data=data)
> y.pred <- predict(best.rss, test, type="response")
> y.pred[y.pred >.5] <- 1
> y.pred[y.pred <.5] <- 0
> accuracy <- sum(y.pred == test$class) / 717
> accuracy
[1] 0.404463
```

Figure 32: Best RSS model accuracy



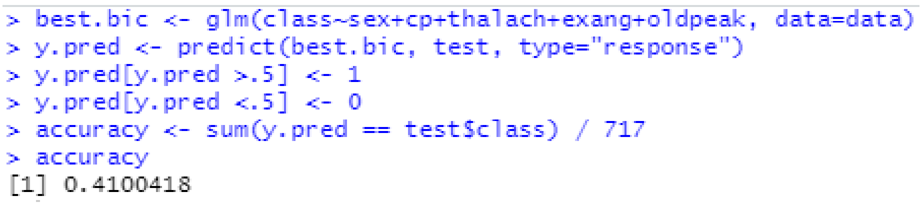
```
> best.r2 <- glm(class~sex+cp+trestbps+chol+fbs+restecg+thalach+exang+oldpeak, data=data)
> y.pred <- predict(best.r2, test, type="response")
> y.pred[y.pred >.5] <- 1
> y.pred[y.pred <.5] <- 0
> accuracy <- sum(y.pred == test$class) / 717
> print(accuracy)
[1] 0.4016736
```

Figure 33: Best Adjusted R2 model accuracy



```
> best.mallow <- glm(class~sex+cp+trestbps+fbs+restecg+thalach+exang+oldpeak, data=data)
> y.pred <- predict(best.mallow, test, type="response")
> y.pred[y.pred >.5] <- 1
> y.pred[y.pred <.5] <- 0
> accuracy <- sum(y.pred == test$class) / 717
> accuracy
[1] 0.3988842
```

Figure 34: Best Mallow’s CP model accuracy



```
> best.bic <- glm(class~sex+cp+thalach+exang+oldpeak, data=data)
> y.pred <- predict(best.bic, test, type="response")
> y.pred[y.pred >.5] <- 1
> y.pred[y.pred <.5] <- 0
> accuracy <- sum(y.pred == test$class) / 717
> accuracy
[1] 0.4100418
```

Figure 35: Best BIC model accuracy

The model that we chose to fit is BIC (Bayesian information criterion) model because it reduces complexity by using only 5 features out of 10. Moreover, among all four model, BIC was the only model that reached accuracy of 41%. On the bottom right plot on the Figure 31 we can see the increase of BIC estimate of test error after five variables are used.

Figure 36 shows the best models and the variables that are in that model’s construction and it is computed by using built-in plot() command that function regsubsets() has:

```
par(mfrow=c(2,2))
plot(regfit.full,scale="r2")
plot(regfit.full,scale="adjr2")
plot(regfit.full,scale="Cp")
plot(regfit.full,scale="bic")
par(mfrow=c(1,1))
```

It the top row of the each model there is a black square for the each feature that is chosen according to the optimal model associated with that statistic. Once again, we can see that the best BIC model uses only 5 variables, which are 'sex', 'cp', 'thalach', 'exang', and 'oldpeak'.

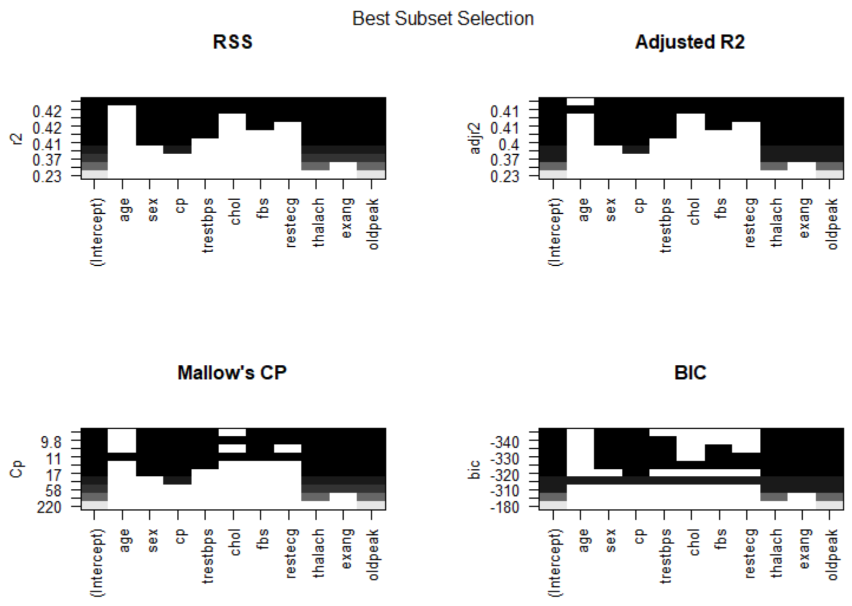


Figure 36: Subset built in plot

10.4.1 BIC Model

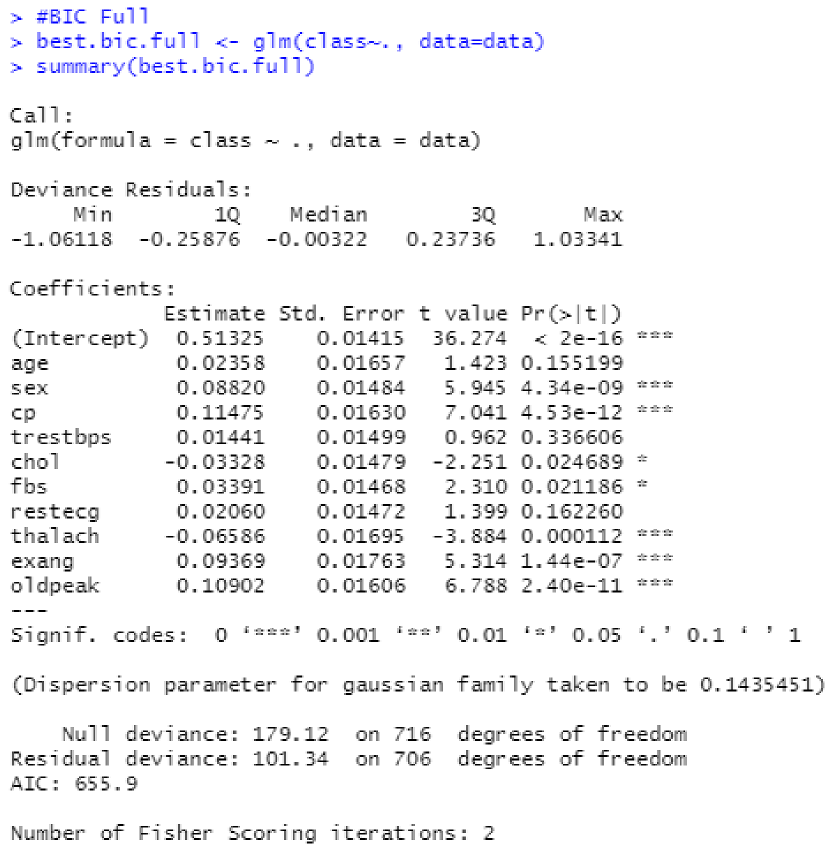
BIC or Schwarz Information Criterion represents the BIC estimation of the test error and is defined as:

$$BIC = -2 * l(\hat{\theta}) + log(n)d$$

BIC tends to take on a small value for a particular model that has a small test error, which leads us to take the model with the smallest BIC value. Also, BIC statistic "generally places a heavier penalty on models with many variables."

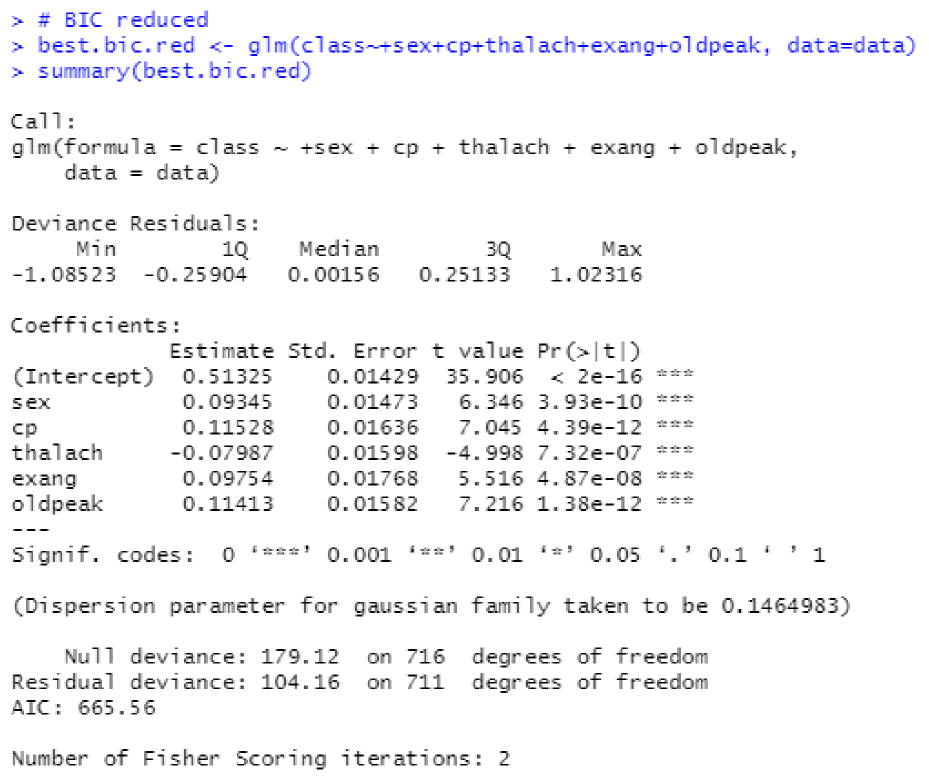
After partitioning the data into training and test set, we fitted the full and reduced BIC models. The full model, which is equal to the RSS model since it uses all the features had an accuracy of 40.45%, while the reduced model had accuracy of 41% which proves that by reducing complexity and using less variables in the final model, the model is going to perform better.

Figures 37 and 38 represent summaries of the full and reduced BIC models. It is easy to see that the features that have the most influence on the final model are 'sex', 'age', 'thalach', 'exang', and 'oldpeak'. Those variables are used in both models and all 5 of them have 3 asterisks which means that they have the highest influence. Next to those 5 features, full model also uses features 'chol' and 'fbs'; however, the have one asterisk each which indicates that those two variables have much less influence on the final model.



```
> #BIC Full
> best.bic.full <- glm(class~., data=data)
> summary(best.bic.full)

Call:
glm(formula = class ~ ., data = data)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.06118  -0.25876  -0.00322   0.23736   1.03341

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.51325    0.01415  36.274  < 2e-16 ***
age          0.02358    0.01657   1.423 0.155199
sex          0.08820    0.01484   5.945 4.34e-09 ***
cp           0.11475    0.01630   7.041 4.53e-12 ***
trestbps     0.01441    0.01499   0.962 0.336606
chol        -0.03328    0.01479  -2.251 0.024689 *
fbs          0.03391    0.01468   2.310 0.021186 *
restecg      0.02060    0.01472   1.399 0.162260
thalach     -0.06586    0.01695  -3.884 0.000112 ***
exang        0.09369    0.01763   5.314 1.44e-07 ***
oldpeak      0.10902    0.01606   6.788 2.40e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 0.1435451)

    Null deviance: 179.12  on 716  degrees of freedom
Residual deviance: 101.34  on 706  degrees of freedom
AIC: 655.9

Number of Fisher Scoring iterations: 2
```

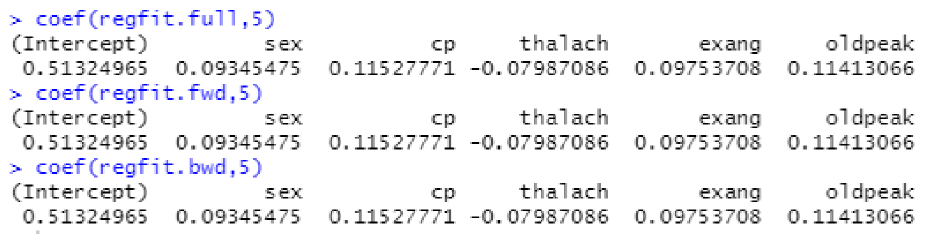
Figure 37: BIC Full Model



```
> # BIC reduced
> best.bic.red <- glm(class~+sex+cp+thalach+exang+oldpeak, data=data)
> summary(best.bic.red)

Call:
glm(formula = class ~ +sex + cp + thalach + exang + oldpeak,
    data = data)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.08523  -0.25904   0.00156   0.25133   1.02316

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.51325    0.01429  35.906  < 2e-16 ***
sex          0.09345    0.01473   6.346 3.93e-10 ***
cp           0.11528    0.01636   7.045 4.39e-12 ***
thalach     -0.07987    0.01598  -4.998 7.32e-07 ***
exang        0.09754    0.01768   5.516 4.87e-08 ***
oldpeak      0.11413    0.01582   7.216 1.38e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 0.1464983)

    Null deviance: 179.12  on 716  degrees of freedom
Residual deviance: 104.16  on 711  degrees of freedom
AIC: 665.56

Number of Fisher Scoring iterations: 2
```

Figure 38: BIC Reduced Model

By looking at the values of coefficients on the Figure 39, it is easy to see that best subset selection procedure, as well as forward and backward stepwise procedures, all produce the same 5 variable models.



```
> coef(regfit.full,5)
(Intercept)         sex          cp     thalach       exang     oldpeak
 0.51324965  0.09345475  0.11527771 -0.07987086  0.09753708  0.11413066
> coef(regfit.fwd,5)
(Intercept)         sex          cp     thalach       exang     oldpeak
 0.51324965  0.09345475  0.11527771 -0.07987086  0.09753708  0.11413066
> coef(regfit.bwd,5)
(Intercept)         sex          cp     thalach       exang     oldpeak
 0.51324965  0.09345475  0.11527771 -0.07987086  0.09753708  0.11413066
```

Figure 39: Feature selection comparison

10.5 Shrinkage models

Ridge regression and Lasso represent the two best-known techniques for shrinking the regression coefficients towards zero. By doing this they reduce the model complexity and may prevent the over-fitting. This way we do not remove any parameters, all parameters are included in the final model, but we reduce the parameter space. We fit the model containing all predictors; however, we put constraints on the coefficient estimates, and that way we regularize them. They are useful because they also report the influence that the given variable has on the final result. The main difference between these two shrinking methods is that ridge regression uses quadratic shrinking and the lasso uses absolute-value shrinking.

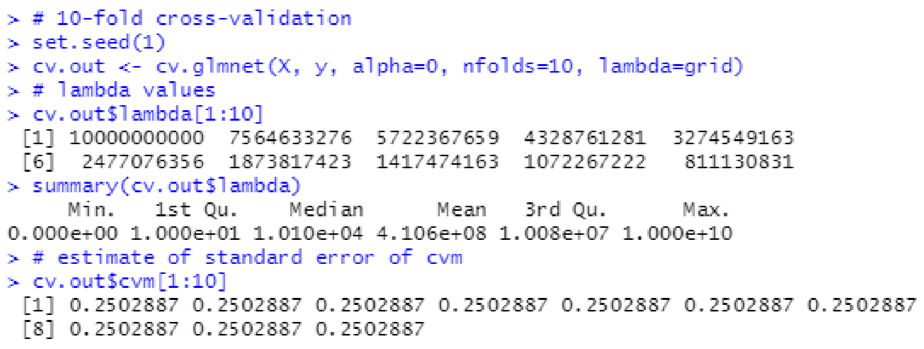
10.5.1 Ridge Regression

By minimizing the RSS, the Ridge regression is looking for coefficient estimates that fit the model well while regularizing the norm of the weights. So, ridge regression is used to reduce the model complexity by shrinking the coefficients and includes all predictors in the final model. In other words, it reduces the complexity of the model and keeps all the variables included at the same time.

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 + \lambda\sum_{j=1}^{p}\beta_j^2 = RSS + \lambda\sum_{j=1}^{p}\beta_j^2$$

"The second term, $\lambda\sum_{j=1}^{p}\beta_j^2$, called a shrinkage penalty, is shrinkage small when β_1,..., β_p are close to zero, and so it has the effect of shrinking penalty the estimates of β_j towards zero. The tuning parameter λ serves to control the relative impact of these two terms on the regression coefficient estimates. "When the penalty term λ is 0, it has no effect, so the final model will be the least square model"[3]. However, when λ -> ∞ the penalty grows which means that ridge regression coefficients will approach 0, but none of them will be exactly zero. This is the reason why the final model includes all the predictors. With λ increasing, bias is increasing but the variance is decreasing. What is specific is that for each λ, ridge regression produces a different set of coefficient estimates.

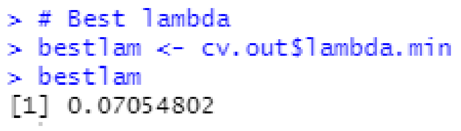
It is critical to select a good value for λ, and cross-validation is used for it. We used 10-fold cross-validation procedure to select the right λ value, which means that for each λ the 10 models are fitted. The figure 40 shows 10 λ values and 10 values for mean cross-validated error. We need to find a λ value that provides the smallest cross-validated error.



```
> # 10-fold cross-validation
> set.seed(1)
> cv.out <- cv.glmnet(X, y, alpha=0, nfolds=10, lambda=grid)
> # lambda values
> cv.out$lambda[1:10]
 [1] 10000000000  7564633276  5722367659  4328761281  3274549163
 [6]  2477076356  1873817423  1417474163  1072267222   811130831
> summary(cv.out$lambda)
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max. 
0.000e+00 1.000e+01 1.010e+04 4.106e+08 1.008e+07 1.000e+10 
> # estimate of standard error of cvm
> cv.out$cvm[1:10]
 [1] 0.2502887 0.2502887 0.2502887 0.2502887 0.2502887 0.2502887 0.2502887
 [8] 0.2502887 0.2502887 0.2502887
```

Figure 40: Finding best λ value with 10-fold cross-validation

The red dots on the Figure 41 represent the mean squared error for each $\log(\lambda)$ value. It is shows that with the increase of λ, the mean squared error is increasing as well. On the other hand, the left vertical line shows the minimum mean squared error. The best λ value is roughly 0.0705 and it can be obtained by computing the following:



```
> # Best lambda
> bestlam <- cv.out$lambda.min
> bestlam
[1] 0.07054802
```

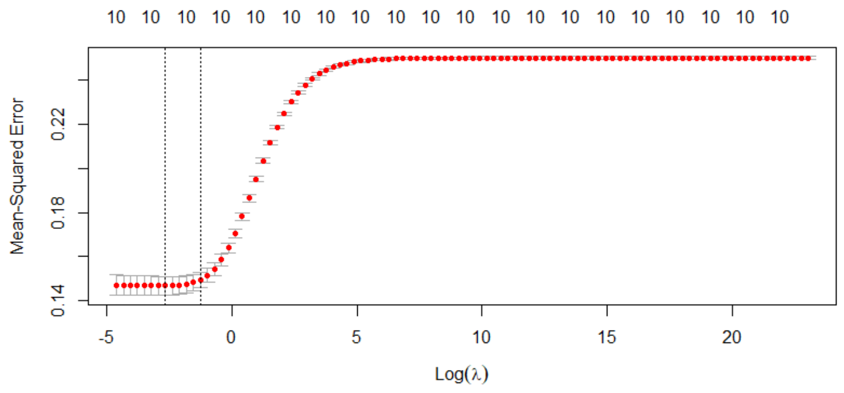
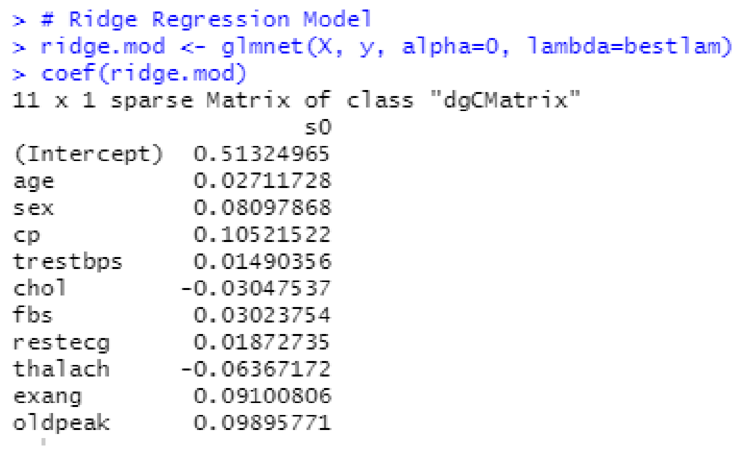


Figure 41: Best λ value plot

After fitting the models and finding the best λ value, we can fit a model using this λ which will give us the coefficients of our ridge regression model as shown in the Figure 42.



```
> # Ridge Regression Model
> ridge.mod <- glmnet(X, y, alpha=0, lambda=bestlam)
> coef(ridge.mod)
11 x 1 sparse Matrix of class "dgCMatrix"
                      s0
(Intercept)  0.51324965
age          0.02711728
sex          0.08097868
cp           0.10521522
trestbps     0.01490356
chol        -0.03047537
fbs          0.03023754
restecg      0.01872735
thalach     -0.06367172
exang        0.09100806
oldpeak      0.09895771
```

Figure 42: Ridge Regression Model coefficients

The visualization of ridge regression model coefficients can be done by using the plot() method. The output of the mentioned method is on the Figure 43, where each curve represents a variable."It shows the path of its coefficient against the l1-norm of the whole coefficient vector as λ varies. The axis above indicates the number of nonzero coefficients at the current λ."

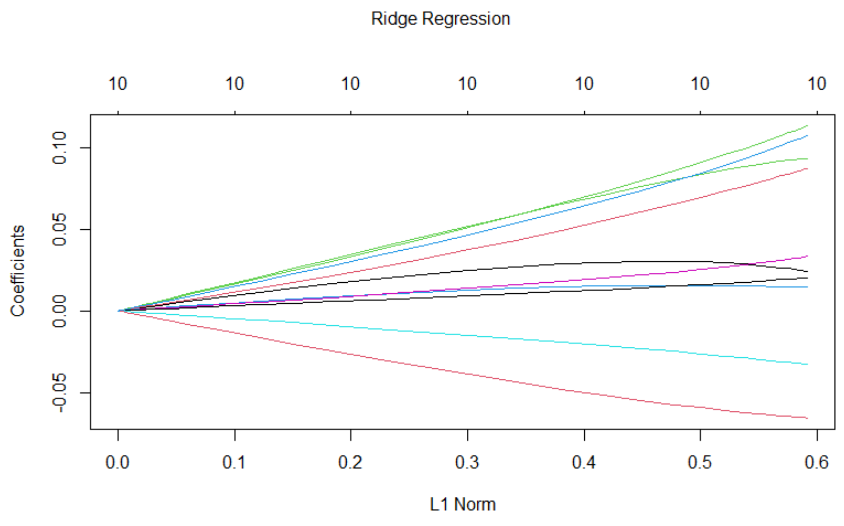
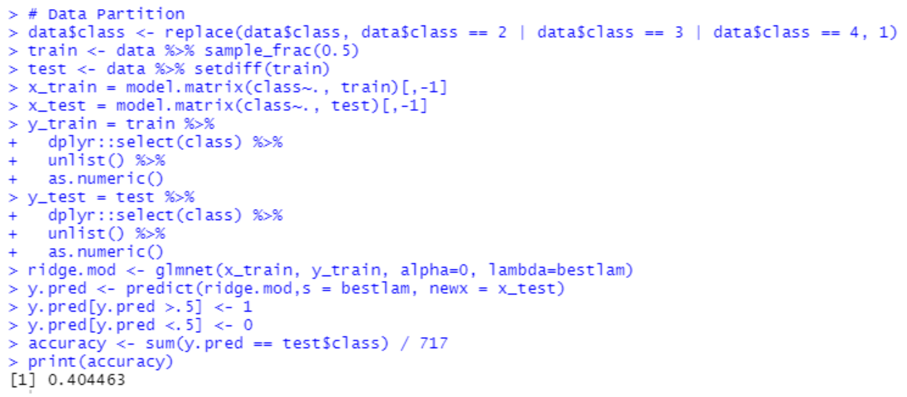


Figure 43: Visualization of ridge regression model coefficients

After splitting the dataset in half and creating test and training sets, using the best λ value, the model is fitted as shown in the Figure 44. The accuracy obtained by this model was 40.44%.



```
> # Data Partition
> data$class <- replace(data$class, data$class == 2 | data$class == 3 | data$class == 4, 1)
> train <- data %>% sample_frac(0.5)
> test <- data %>% setdiff(train)
> x_train = model.matrix(class~., train)[,-1]
> x_test = model.matrix(class~., test)[,-1]
> y_train = train %>%
+   dplyr::select(class) %>%
+   unlist() %>%
+   as.numeric()
> y_test = test %>%
+   dplyr::select(class) %>%
+   unlist() %>%
+   as.numeric()
> ridge.mod <- glmnet(x_train, y_train, alpha=0, lambda=bestlam)
> y.pred <- predict(ridge.mod,s = bestlam, newx = x_test)
> y.pred[y.pred >.5] <- 1
> y.pred[y.pred <.5] <- 0
> accuracy <- sum(y.pred == test$class) / 717
> print(accuracy)
[1] 0.404463
```

Figure 44: The best ridge regression model

10.5.2 The Lasso

The lasso (least absolute shrinkage and selection operator) is very similar to the ridge regression, but as penalization term it takes into account the magnitudes instead of the square of the coefficients. With this being said, it also forces some of the coefficient estimates towards zero, but by using L1 penalty, it actually forces some of the coefficient estimates to be exactly equal to zero and by doing this performs feature selection. Also, it can be said that the lasso yields sparse models which are models that include only the subset of variables. Since there is feature selection, the lasso models are usually less complex and easier to interpret than the ridge models which uses all predictors in the final model.

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 + \lambda\sum_{j=1}^{p}|\beta_j| = RSS + \lambda\sum_{j=1}^{p}|\beta_j|$$

By doing cross-validation we can find the best lambda value that will be used to fit the model. The red dots on the Figure 45 shows the mean squared errors for each log(λ) value. The left vertical line on the plot shows the minimum mean squared error for given log(λ) value. After performing the cross validation (Figure 46), the value obtained for the best λ is 0.01. This lambda will be used to fit the final model.

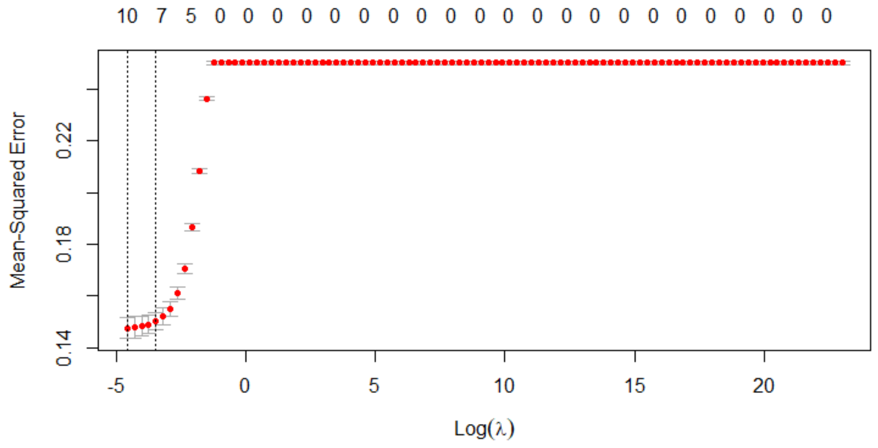
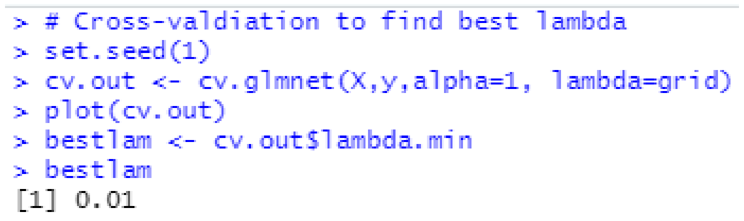


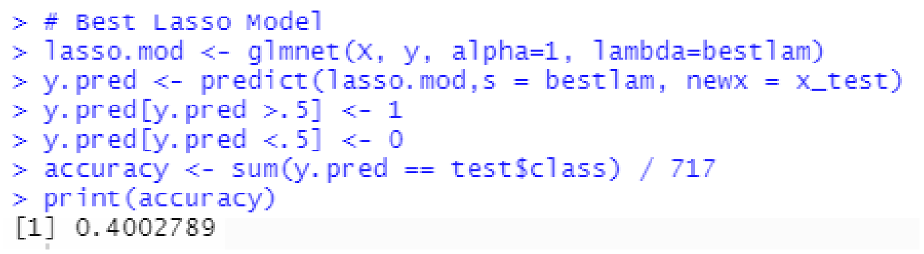
Figure 45: The best lasso lambda value



```
> # Cross-valdiation to find best lambda
> set.seed(1)
> cv.out <- cv.glmnet(X,y,alpha=1, lambda=grid)
> plot(cv.out)
> bestlam <- cv.out$lambda.min
> bestlam
[1] 0.01
```

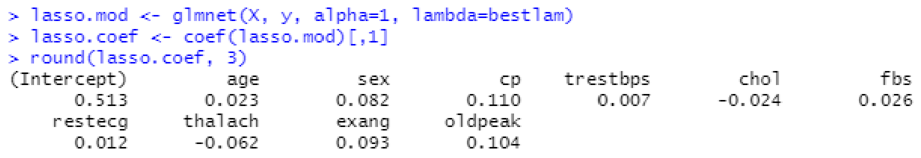
Figure 46: Cross-validation to find the best lambda value

The accuracy obtained by fitting the model using best lambda value which is 0.01 is 40.02%. The model is fitted the following way:



```
> # Best Lasso Model
> lasso.mod <- glmnet(X, y, alpha=1, lambda=bestlam)
> y.pred <- predict(lasso.mod,s = bestlam, newx = x_test)
> y.pred[y.pred >.5] <- 1
> y.pred[y.pred <.5] <- 0
> accuracy <- sum(y.pred == test$class) / 717
> print(accuracy)
[1] 0.4002789
```

We can see that the estimates of the coefficients are decreased and going towards 0. Next, it can be seen that some variables, such as 'trestbps', 'chol', 'fbs', and 'restecg', have very small coefficients and are closer to 0 than the rest of the variables, which indicates that they have less influence in the final model.



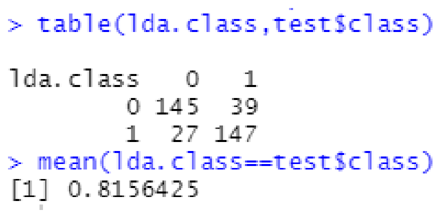
```
> lasso.mod <- glmnet(X, y, alpha=1, lambda=bestlam)
> lasso.coef <- coef(lasso.mod)[,1]
> round(lasso.coef, 3)
(Intercept)         age         sex          cp    trestbps        chol         fbs
      0.513       0.023       0.082       0.110       0.007      -0.024       0.026
    restecg     thalach       exang     oldpeak
      0.012      -0.062       0.093       0.104
```

10.6 Discriminant models and KNN

Since the accuracy of the different models is not satisfactory for classifying the samples, discriminant models and KNN models have been tested on the binary data. The train test split for these models is 50%.

10.6.1 Linear Discriminant Analysis

The accuracy obtained with LDA on the binary classification task is 81.56% and the confusion matrix is shown on Figure 47. The histograms for the two classes generated by LDA are shown on Figure 48.

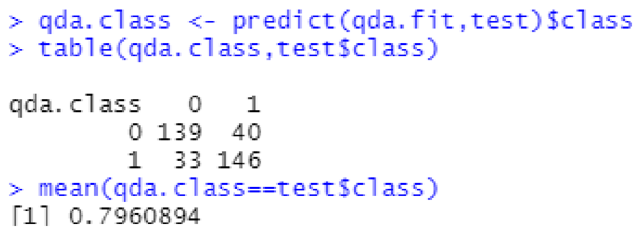


```
> table(lda.class,test$class)

lda.class   0   1
        0 145  39
        1  27 147
> mean(lda.class==test$class)
[1] 0.8156425
```

Figure 47: LDA for binary classification

10.6.2 Quadratic Discriminant Analysis

With the quadratic model, the accuracy is shown to be worse than LDA and it is 79.60%. The confusion matrix is shown on Figure 49.



```
> qda.class <- predict(qda.fit,test)$class
> table(qda.class,test$class)

qda.class   0   1
        0 139  40
        1  33 146
> mean(qda.class==test$class)
[1] 0.7960894
```

Figure 49: QDA for binary classification

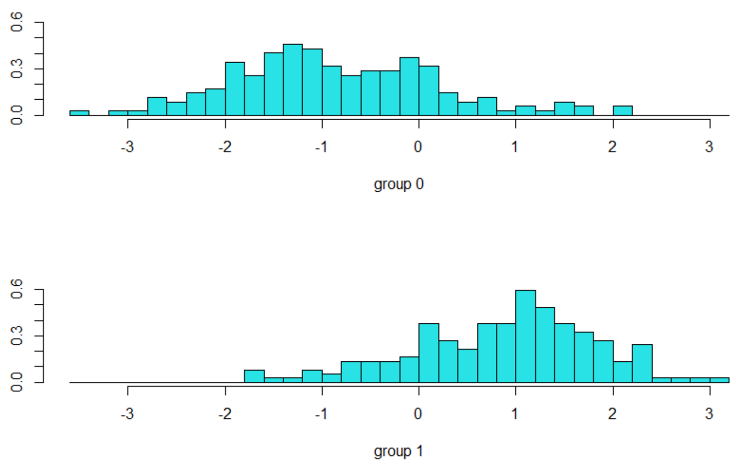
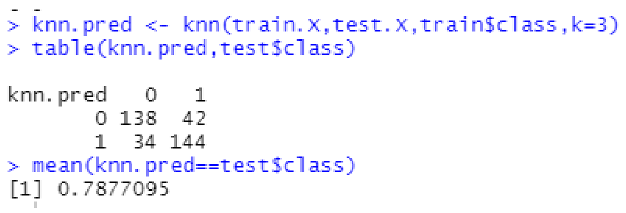


Figure 48: LDA mapping plot between classes

10.6.3 KNN

For the binary classification task, several KNN models were tested with different values of K. The highest accuracy is obtained for K = 3: 78.77%. Other values of K produced the following accuracies: K = 1: 73.46%, K = 2: 74.86%, K = 4: 77.09%, K = 5: 77.93% and K = 8: 75.69%. Figure 50 shows the confusion matrix for the best value of K.



```
> knn.pred <- knn(train.X,test.X,train$class,k=3)
> table(knn.pred,test$class)

knn.pred   0   1
       0 138  42
       1  34 144
> mean(knn.pred==test$class)
[1] 0.7877095
```

Figure 50: Binary KNN with K = 3

11 Conclusion

Some of the main conclusions from the analysis are:

- The features most important for each class are:
  - Class 1: ’sex’, ’cp’, ’trestbps’, ’exang’
  - Class 2: ’sex’, ’cp’, ’thlach’, ’exang’, ’oldpeak’

  - Class 3: ’oldpeak’, ’chol’, ’fbs’
  - Class 4: ’oldpeak’

- The data allows for better classification models between healthy and disease state samples
- Models such as LDA, QDA and KNN can reach 50% accuracy when differentiating between all disease classes, but up to 80% for the binary classification
- Discriminant analysis models classify healthy samples with the highest accuracy
- The accuracy in KNN decreases when reducing K for the multiclass classification
- According to best subset selection, forward and backward selection, best 5-variable model contains ’sex’, ’cp’, ’thlach’, ’exang’, and ’oldpeak’ features
- Models such as RSS, Adjusted R2, Mallow’s CP, and BIC reached approximately 40% accuracy
- Ridge regression and lasso model also reached the 40% accuracy
- Less complex models tend to have higher accuracy
- The highest accuracy obtained is with the LDA binary classification model

References

[1] Heart Disease Data Set

[2] Ammar KA, Jacobsen SJ, Mahoney DW, et al. Prevalence and prognostic significance of heart failure stages: application of the American College of Cardiology/American Heart Association heart failure staging criteria in the community

[3] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani: An Introduction to Statistical Learning

12 Additional plots

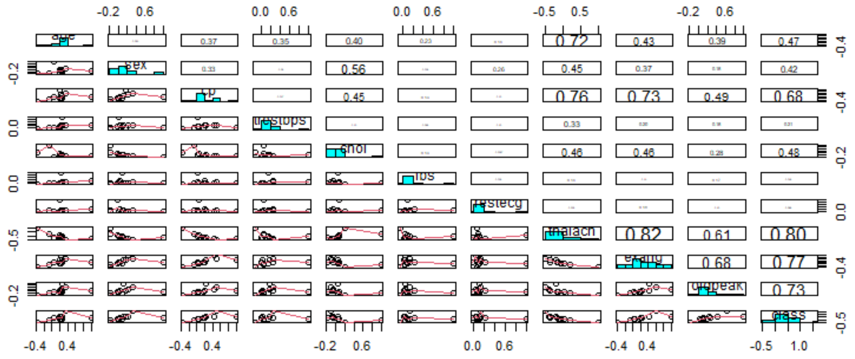


Figure 51: Pairs function plot

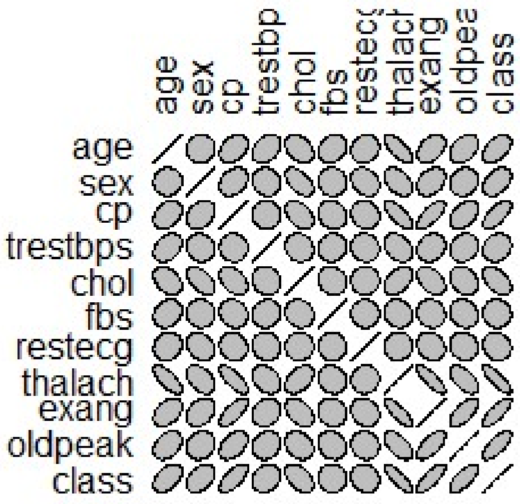


Figure 52: Ellipse function plot

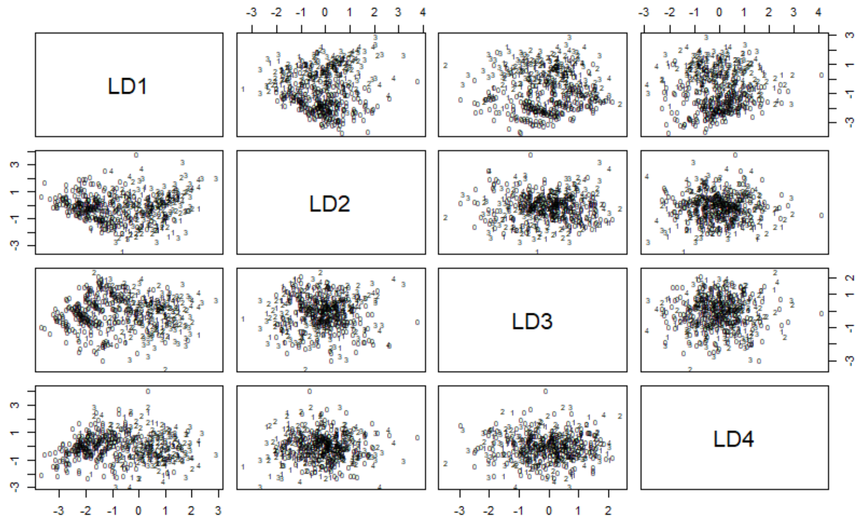


Figure 53: LDA mapping plot for all coefficients on the multiclass classification task

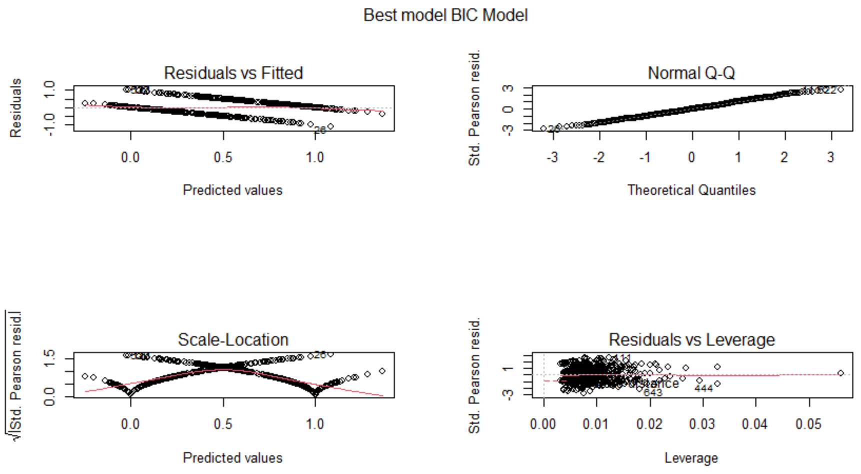


Figure 54: BIC reduced model plot