Exercise 5

Konstantinos Vakalopoulos 12223236

2022-12-1

Premilinary work

First, the data (hcvdat1.csv) were loaded

```
bank <- read.csv2("C:/Users/vaka1/Desktop/bank.csv") #must be modified
```

and missing values were omitted.

```
bank <- na.omit(bank)
```

We transform all the character variables to factors and all int variables to numeric. Furthermore, the value "yes" was replaced by 1 and the value "no" to zero.

```
bank$job <- as.factor(bank$job)
bank$marital <- as.factor(bank$marital)
bank$education <- as.factor(bank$education)
bank$default <- as.factor(bank$default)
bank$housing <- as.factor(bank$housing)
bank$loan <- as.factor(bank$loan)
bank$contact <- as.factor(bank$contact)
bank$month <- as.factor(bank$month)
bank$poutcome <- as.factor(bank$poutcome)


bank$y <- ifelse(bank$y=="yes",1,0)
bank$y <- as.factor(bank$y)
```

```
bank$age <- as.numeric(bank$age)
bank$balance <- as.numeric(bank$balance)
bank$day <- as.numeric(bank$day)
bank$duration <- as.numeric(bank$duration)
bank$campaign <- as.numeric(bank$campaign)
bank$pdays <- as.numeric(bank$pdays)
bank$previous <- as.numeric(bank$previous)
```

Thus, for addition information about the data, the following commands were used

```
summary(bank)
```

```
##       age                  job          marital         education    default
##  Min.   :19.00   management :969   divorced: 528   primary  : 678   no :4445
##  1st Qu.:33.00   blue-collar:946   married :2797   secondary:2306   yes:  76
##  Median :39.00   technician :768   single  :1196   tertiary :1350
##  Mean   :41.17   admin.     :478                   unknown  : 187
##  3rd Qu.:49.00   services   :417
##  Max.   :87.00   retired    :230
##                  (Other)    :713
##     balance      housing     loan           contact          day
##  Min.   :-3313   no :1962   no :3830   cellular :2896   Min.   : 1.00
##  1st Qu.:   69   yes:2559   yes: 691   telephone: 301   1st Qu.: 9.00
##  Median :  444                         unknown  :1324   Median :16.00
##  Mean   : 1423                                          Mean   :15.92
##  3rd Qu.: 1480                                          3rd Qu.:21.00
##  Max.   :71188                                          Max.   :31.00
##
##      month         duration       campaign          pdays
##  may    :1398   Min.   :   4   Min.   : 1.000   Min.   : -1.00
##  jul    : 706   1st Qu.: 104   1st Qu.: 1.000   1st Qu.: -1.00
##  aug    : 633   Median : 185   Median : 2.000   Median : -1.00
##  jun    : 531   Mean   : 264   Mean   : 2.794   Mean   : 39.77
##  nov    : 389   3rd Qu.: 329   3rd Qu.: 3.000   3rd Qu.: -1.00
##  apr    : 293   Max.   :3025   Max.   :50.000   Max.   :871.00
##  (Other): 571
##     previous          poutcome    y
##  Min.   : 0.0000   failure: 490   0:4000
##  1st Qu.: 0.0000   other  : 197   1: 521
##  Median : 0.0000   success: 129
##  Mean   : 0.5426   unknown:3705
##  3rd Qu.: 0.0000
##  Max.   :25.0000
##
```

```
str(bank)
```

```
## 'data.frame':    4521 obs. of  17 variables:
##  $ age      : num  30 33 35 30 59 35 36 39 41 43 ...
##  $ job      : Factor w/ 12 levels "admin.","blue-collar",..: 11 8 5 5 2 5 7 10 3 8 ...
##  $ marital  : Factor w/ 3 levels "divorced","married",..: 2 2 3 2 2 3 2 2 2 2 ...
##  $ education: Factor w/ 4 levels "primary","secondary",..: 1 2 3 3 2 3 3 2 3 1 ...
##  $ default  : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ balance  : num  1787 4789 1350 1476 0 ...
##  $ housing  : Factor w/ 2 levels "no","yes": 1 2 2 2 2 1 2 2 2 2 ...
##  $ loan     : Factor w/ 2 levels "no","yes": 1 2 1 2 1 1 1 1 1 2 ...
##  $ contact  : Factor w/ 3 levels "cellular","telephone",..: 1 1 1 3 3 1 1 1 3 1 ...
##  $ day      : num  19 11 16 3 5 23 14 6 14 17 ...
##  $ month    : Factor w/ 12 levels "apr","aug","dec",..: 11 9 1 7 9 4 9 9 9 1 ...
##  $ duration : num  79 220 185 199 226 141 341 151 57 313 ...
##  $ campaign : num  1 1 1 4 1 2 1 2 2 1 ...
##  $ pdays    : num  -1 339 330 -1 -1 176 330 -1 -1 147 ...
##  $ previous : num  0 4 1 0 0 3 2 0 0 2 ...
##  $ poutcome : Factor w/ 4 levels "failure","other",..: 4 1 1 4 4 1 2 4 4 1 ...
##  $ y        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
```

Below, it is presented the number of no and yes for the response variable y using the library ggplot2. Based on the histogram, it can be seen that the data are heavily imbalanced

```
library(ggplot2)
```

```
ggplot(bank, aes(x=y)) +
  geom_histogram(stat="count")
```

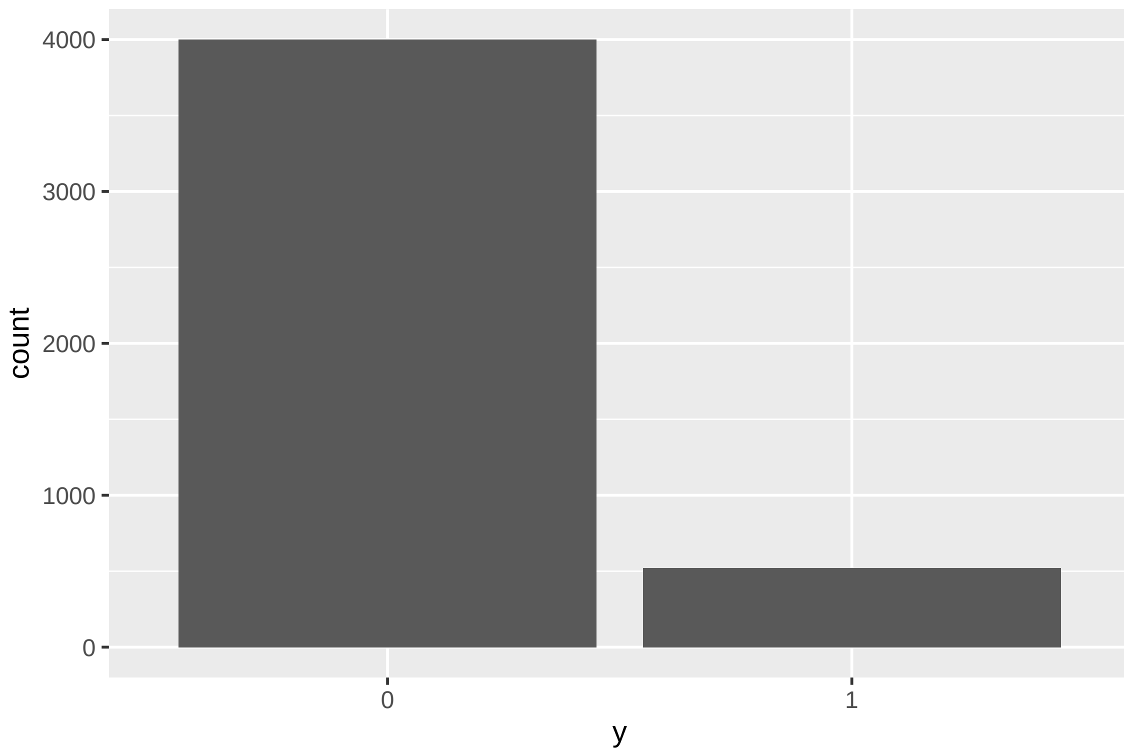

Figure 1: Frequency of the response variable

More information about the data are presented to the plots using the library Hmisc.

```
library(Hmisc)
```

```
hist.data.frame(bank[,which(sapply(bank, is.numeric)==TRUE)])
```

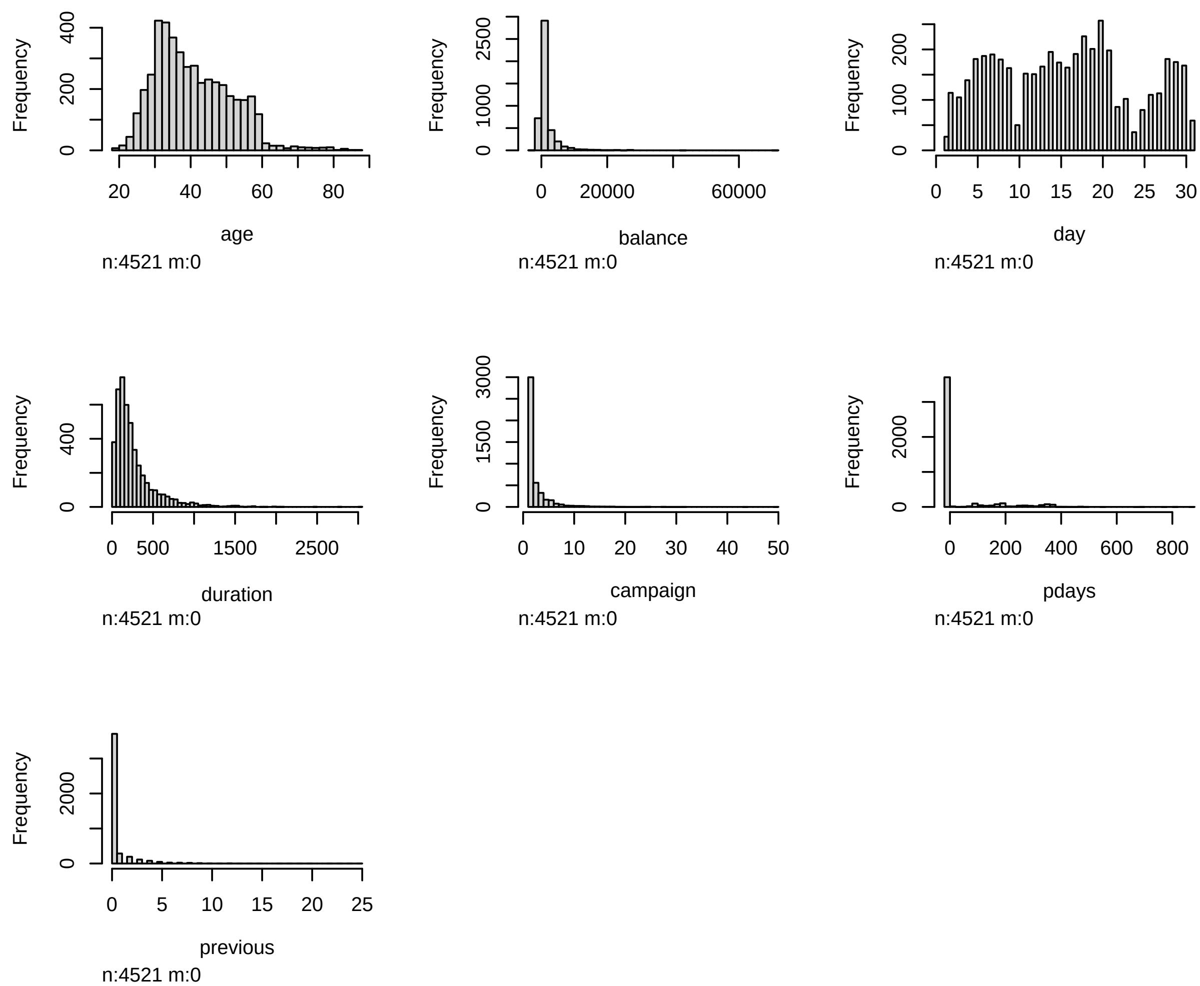

Figure 2: Histograms of the numeric variables

```
hist.data.frame(bank[,which(sapply(bank, is.numeric)==FALSE)])
```

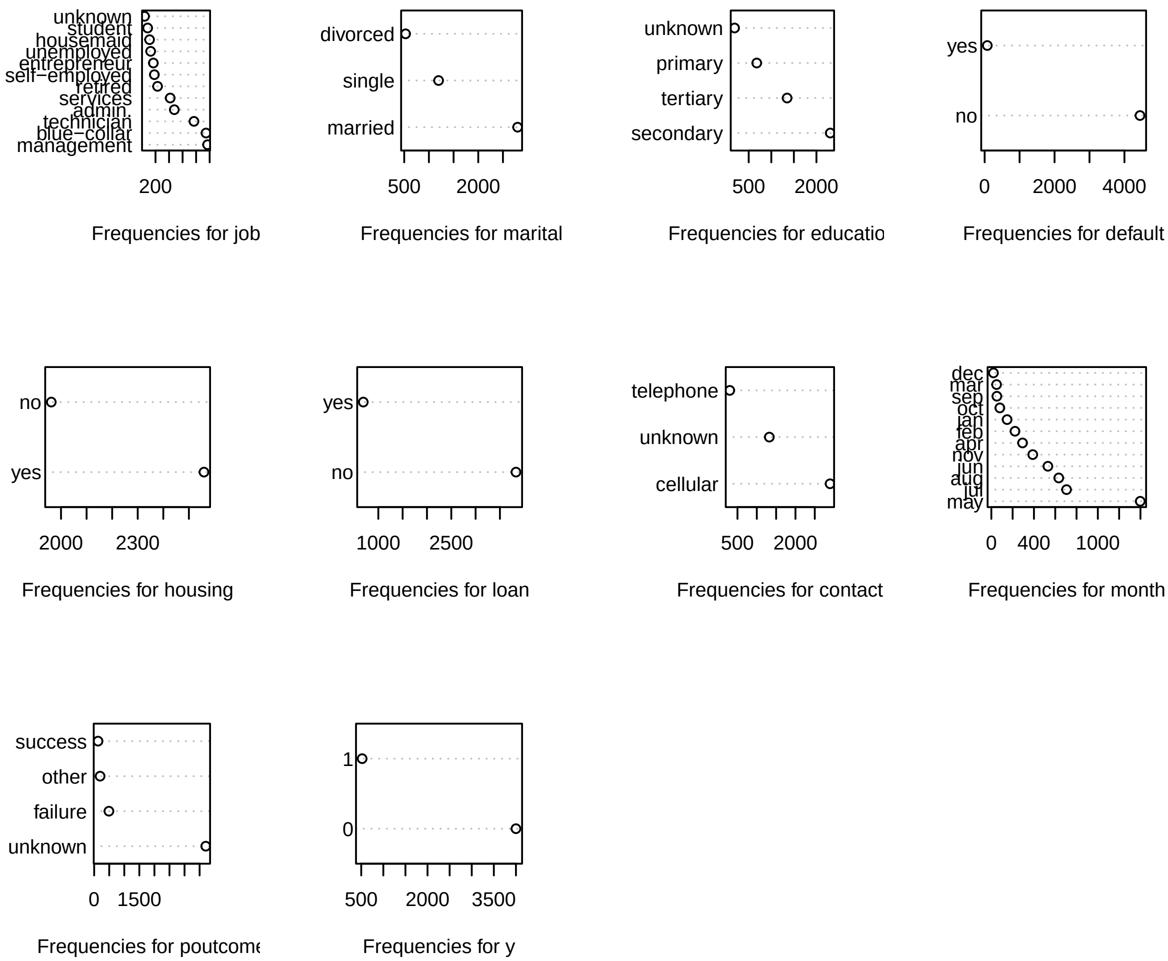

Figure 3: Frequencies of the categorical variables

Question 1(a)

For the training set, 3000 observations were selected randomly.

```
set.seed(12223236)
train <- sample(1:nrow(bank), 3000)
test <- (1:nrow(bank))[-train]
```

and then the logistic regression was applied using the function glm() with family = “binomial”.

```
model.lr <- glm(y~., data=bank, subset=train, family="binomial")
```

The inference table from the logistic regression is presented below.

```r
summary(model.lr)
```

```
##
## Call:
## glm(formula = y ~ ., family = "binomial", data = bank, subset = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.8816  -0.3749  -0.2594  -0.1715   3.1071
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -2.628e+00  7.584e-01  -3.465 0.000531 ***
## age                -1.496e-02  8.906e-03  -1.679 0.093073 .
## jobblue-collar     -3.490e-01  2.928e-01  -1.192 0.233392
## jobentrepreneur    -2.120e-01  4.575e-01  -0.463 0.643087
## jobhousemaid       -2.642e-01  5.292e-01  -0.499 0.617634
## jobmanagement       4.801e-02  2.947e-01   0.163 0.870600
## jobretired          1.036e+00  3.789e-01   2.736 0.006227 **
## jobself-employed   -7.305e-02  4.425e-01  -0.165 0.868891
## jobservices        -1.562e-01  3.432e-01  -0.455 0.649144
## jobstudent         -1.050e-01  5.158e-01  -0.204 0.838702
## jobtechnician      -2.083e-01  2.849e-01  -0.731 0.464821
## jobunemployed      -3.854e-01  4.800e-01  -0.803 0.422084
## jobunknown          2.576e-01  7.337e-01   0.351 0.725528
## maritalmarried     -3.192e-01  2.207e-01  -1.446 0.148178
## maritalsingle      -2.659e-01  2.556e-01  -1.040 0.298111
## educationsecondary  5.432e-02  2.446e-01   0.222 0.824232
## educationtertiary   3.099e-01  2.908e-01   1.066 0.286602
## educationunknown   -3.649e-01  4.364e-01  -0.836 0.403137
## defaultyes          8.351e-01  4.775e-01   1.749 0.080344 .
## balance            -1.391e-05  2.206e-05  -0.631 0.528234
## housingyes         -2.820e-01  1.693e-01  -1.666 0.095753 .
## loanyes            -5.517e-01  2.362e-01  -2.336 0.019491 *
## contacttelephone   -4.348e-02  2.878e-01  -0.151 0.879912
## contactunknown     -1.478e+00  2.729e-01  -5.416 6.08e-08 ***
## day                 2.543e-02  1.018e-02   2.498 0.012501 *
## monthaug           -4.526e-01  3.127e-01  -1.447 0.147808
## monthdec            3.599e-01  8.080e-01   0.445 0.656001
## monthfeb            5.487e-01  3.609e-01   1.520 0.128423
## monthjan           -1.120e+00  5.071e-01  -2.209 0.027166 *
## monthjul           -6.184e-01  3.105e-01  -1.992 0.046419 *
## monthjun            7.356e-01  3.685e-01   1.997 0.045873 *
## monthmar            1.860e+00  4.800e-01   3.874 0.000107 ***
## monthmay           -1.549e-01  2.904e-01  -0.533 0.593858
## monthnov           -7.129e-01  3.377e-01  -2.111 0.034792 *
## monthoct            1.720e+00  4.240e-01   4.057 4.96e-05 ***
## monthsep           -2.645e-01  5.887e-01  -0.449 0.653255
## duration            3.991e-03  2.380e-04  16.771  < 2e-16 ***
## campaign           -4.569e-02  3.137e-02  -1.457 0.145249
## pdays              -1.367e-04  1.342e-03  -0.102 0.918876
## previous           -1.059e-02  5.103e-02  -0.208 0.835546
## poutcomeother       7.237e-01  3.518e-01   2.057 0.039691 *
```

```
## poutcomesuccess     2.769e+00  3.444e-01   8.040 8.97e-16 ***
## poutcomeunknown     1.226e-01  4.103e-01   0.299 0.765012
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2132.9  on 2999  degrees of freedom
## Residual deviance: 1447.9  on 2957  degrees of freedom
## AIC: 1533.9
##
## Number of Fisher Scoring iterations: 6
```

Based on the summary table, it is observed that the syntax of the coefficients is identical to linear models (function lm() in R). However, these results are based on the maximum likelihood estimations. On the bottom of the summary table, the number of Fisher scoring iterations are presented, which are 6. This indicates that 6 iterations was needed from the final calculations of the coefficients starting from zero.

Further investigating the summary table, the model quality indices can be seen, which are the Null and Residual deviance and the AIC score. The deviances are the contribution and more specifically the negative contribution of each observation to the log-likelihood function. The null Deviance is the deviance for a model with the intercept only. Thus, a model where none of the explanatory variables explain the response variable. The Residual Deviance corresponds to the full model. Finally, the AIC metric is used for model comparison.

The most important thing on the summary table is the coefficients. Similar to linear models, each coefficient has its own p value. This indicates the variable significance and insignificance in case of the p value being smaller or higher than 0.05, respectively. The difference from the linear models is that the t value was replaced be the z value.

It is worth mentioning that there are plenty of categorical variables. In R for each categorical variable is recorded into a set of separate binary variables and thus each binary variable is treated as an explanatory.

Question 1(b)

The rest of the data set was used as the test set. The predict() function was used in order to predict the group label pf the remaining test observations. By default the predict function returns the predictions in the scale of the linear predictor and thus the decision boundary is zero. However, the type = "response" was used in the predict function to obtain the probabilities from the linear predictor. This time the decision boundary is 0.5. Next, the misclassification error was calculated for each group separately.

```r
pred <- predict(model.lr,bank[test,])

prob <- predict(model.lr,bank[test,], type="response")
predicted.classes <- ifelse(prob > 0.5, 1, 0)
actual.classes <- bank[test,]$y
TAB <- table(actual.classes,predicted.classes)
TAB
```

```
##               predicted.classes
## actual.classes    0    1
##              0 1315   28
##              1  130   48
```

```r
error.no <- 1 - TAB[1,1]/(TAB[1,2]+TAB[2,1]+TAB[1,1])
error.no
```

```
## [1] 0.1072641
```

```r
error.yes <- 1 - TAB[2,2]/(TAB[1,2]+TAB[2,1]+TAB[2,2])
error.yes
```

```
## [1] 0.7669903
```

It is observed that the misclassification error is very different for each group and that is due to the fact that the data are imbalance.

Question 1(c)

In order to improve the misclassification error in the minority class, a good method is by assigning weights in each observation based on the class that it belongs. Thus, the weights will depend mainly from the class of the observation. Basically, the weight will be small if the observation belongs to majority class and high if the observation belongs to the minority class.The entire idea of the weights is to penalize the minority class for misclassifying itself by increasing class weight while simultaneously decreasing class weight for the majority class. The formula for the weight assignments is: wj=n_samples / (n_classes * n_samplesj), where wj is the weight for each class(j signifies the class), n_samples the total number of samples or rows in the dataset, n_classes the total number of unique classes in the target, n_samplesj is the total number of rows of the respective class. Thus the final weights are for the two classes:

```r
w0 <- length(train)/(2*table(bank[train,]$y)[1]) #weight for the 0 or class "no"
w1 <- length(train)/(2*table(bank[train,]$y)[2]) #weight for the 1 or class "yes"
```

Then, the weights were assigned in each observation and logistic regression model was created.

```r
w <- ifelse(bank[train,]$y == 1, w1, w0)
model.lr.weight <- glm(y~., data=bank[train,], family="binomial",
                       weight = w)
```

The resulting missclassification errors are:

```r
prob <- predict(model.lr.weight,bank[test,], type="response")
predicted.classes <- ifelse(prob > 0.5, 1, 0)
actual.classes <- bank[test,]$y
TAB.weight <- table(actual.classes,predicted.classes)
TAB.weight
```

```
##               predicted.classes
## actual.classes    0    1
##              0 1140  203
##              1   37  141
```

```r
error.no.weight <- 1 - TAB.weight[1,1]/(TAB.weight[1,2]+TAB.weight[2,1]+TAB.weight[1,1])
error.no.weight
```

```
## [1] 0.173913
```

```
error.yes.weight <- 1 - TAB.weight[2,2]/(TAB.weight[1,2]+TAB.weight[2,1]+TAB.weight[2,2])
error.yes.weight
```

```
## [1] 0.6299213
```

Compared to the previous errors without the weights, it is observed that the error for the minority class was improved. However the error for the majority class was slightly increased.

Question 1(d)

In this part of the exercise, the stepwise variable selection was used with the function step() in order to simplify the model from the question 1(c).

```
model.lr.step <-  step(model.lr.weight,direction="both")
```

```
## Start:  AIC=3471.62
## y ~ age + job + marital + education + default + balance + housing +
##     loan + contact + day + month + duration + campaign + pdays +
##     previous + poutcome
##
##             Df Deviance    AIC
## - marital    2   2465.3 3468.3
## - previous   1   2464.6 3469.7
## - pdays      1   2464.6 3469.7
## - balance    1   2464.7 3469.7
## <none>           2464.6 3471.6
## - default    1   2467.5 3472.6
## - education  3   2471.8 3472.9
## - day        1   2469.6 3474.6
## - housing    1   2470.5 3475.5
## - age        1   2473.2 3478.2
## - campaign   1   2477.3 3482.4
## - job       11   2512.4 3497.4
## - loan       1   2498.6 3503.6
## - contact    2   2529.4 3532.5
## - poutcome   3   2598.7 3599.8
## - month     11   2655.4 3640.4
## - duration   1   3497.7 4502.7
##
## Step:  AIC=3469.03
## y ~ age + job + education + default + balance + housing + loan +
##     contact + day + month + duration + campaign + pdays + previous +
##     poutcome
##
##             Df Deviance    AIC
## - pdays      1   2465.3 3467.1
## - previous   1   2465.3 3467.1
## - balance    1   2465.3 3467.1
## <none>           2465.3 3469.0
## - default    1   2468.3 3470.1
```

```
## - education  3   2472.7 3470.5
## - day        1   2470.2 3472.0
## + marital    2   2464.6 3472.4
## - housing    1   2472.0 3473.7
## - age        1   2476.3 3478.1
## - campaign   1   2477.9 3479.7
## - job       11   2515.0 3496.8
## - loan       1   2499.3 3501.1
## - contact    2   2530.5 3530.2
## - poutcome   3   2598.8 3596.5
## - month     11   2657.5 3639.3
## - duration   1   3515.0 4516.7
##
## Step:  AIC=3467.06
## y ~ age + job + education + default + balance + housing + loan +
##     contact + day + month + duration + campaign + previous +
##     poutcome
##
##             Df Deviance    AIC
## - balance    1   2465.4 3465.1
## - previous   1   2465.4 3465.1
## <none>           2465.3 3467.1
## - default    1   2468.4 3468.1
## - education  3   2472.8 3468.5
## + pdays      1   2465.3 3469.0
## - day        1   2470.3 3470.0
## + marital    2   2464.6 3470.4
## - housing    1   2472.0 3471.7
## - age        1   2476.3 3476.1
## - campaign   1   2478.0 3477.7
## - job       11   2515.2 3494.9
## - loan       1   2499.3 3499.0
## - contact    2   2530.9 3528.6
## - poutcome   3   2608.4 3604.2
## - month     11   2657.5 3637.3
## - duration   1   3515.0 4514.7
##
## Step:  AIC=3464.85
## y ~ age + job + education + default + housing + loan + contact +
##     day + month + duration + campaign + previous + poutcome
##
##             Df Deviance    AIC
## - previous   1   2465.5 3462.9
## <none>           2465.4 3464.9
## - default    1   2468.4 3465.9
## - education  3   2472.8 3466.3
## + balance    1   2465.3 3466.8
## + pdays      1   2465.3 3466.8
## - day        1   2470.3 3467.7
## + marital    2   2464.7 3468.2
## - housing    1   2472.0 3469.5
## - age        1   2476.3 3473.8
## - campaign   1   2478.0 3475.5
## - job       11   2515.4 3492.8
```

```
## - loan       1   2499.8 3497.2
## - contact    2   2530.9 3526.4
## - poutcome   3   2608.5 3602.0
## - month     11   2658.0 3635.5
## - duration   1   3518.7 4516.1
##
## Step:  AIC=3462.72
## y ~ age + job + education + default + housing + loan + contact +
##     day + month + duration + campaign + poutcome
##
##             Df Deviance    AIC
## <none>           2465.5 3462.7
## - default    1   2468.5 3463.8
## - education  3   2472.9 3464.1
## + previous   1   2465.4 3464.6
## + balance    1   2465.4 3464.6
## + pdays      1   2465.4 3464.6
## - day        1   2470.4 3465.6
## + marital    2   2464.8 3466.0
## - housing    1   2472.1 3467.3
## - age        1   2476.4 3471.6
## - campaign   1   2478.2 3473.4
## - job       11   2515.5 3490.7
## - loan       1   2500.0 3495.3
## - contact    2   2531.0 3524.2
## - month     11   2658.0 3633.2
## - poutcome   3   2643.4 3634.6
## - duration   1   3519.4 4514.6
```

Based on that model, the classification error was calculated.

```
prob <- predict(model.lr.step,bank[test,], type="response")
predicted.classes <- ifelse(prob > 0.5, 1, 0)
actual.classes <- bank[test,]$y
TAB.step <- table(actual.classes,predicted.classes)
TAB.step
```

```
##                 predicted.classes
## actual.classes    0    1
##               0 1137  206
##               1   38  140
```

```
error.no.step<- 1 - TAB.step[1,1]/(TAB.step[1,2]+TAB.step[2,1]+TAB.step[1,1])
error.no.step
```

```
## [1] 0.1766836
```

```
error.yes.step <- 1 - TAB.step[2,2]/(TAB.step[1,2]+TAB.step[2,1]+TAB.step[2,2])
error.yes.step
```

```
## [1] 0.6354167
```

This stepwise method did not lead to improvement of the classification error. Actually, both errors for the two classes were slightly increased.

Question 2(a)

The data set data(Khan) was used from the package ISLR. The target class of the data set consists of 4 groups and all the data set contains 2308 genes, which are the features. The data were split into train, with 63 subjects, and test, with 20 subjects.

```
library(ISLR)
```

```
data(Khan)

khan_train = data.frame(x = Khan$xtrain, y = as.factor(Khan$ytrain))
khan_test = data.frame(x = Khan$xtest, y = as.factor(Khan$ytest))
```

Linear and Quadratic Discriminant Analysis (LDA and QDA) will not work in the Khan data set because the number of variables is much higher than the number of observations. Therefore, we are not able to calculate the inverse covariance matrix. However, Regularized Discriminant Analysis (RDA) uses a combination of LDA and QDa regarding the calculation of the covariance matrix. RDA shrinks the separate covariances of QDA toward a common covariance as in LDA using a parameter alpha or lambda. For lambda equals to zero the covariance matrix is the same as the QDA and for lambda equals to one the covariance matrix is the same as the LDA. In R, there is a second parameter called gamma, which pushes the elements of the covariance matrix into a diagonal matrix using the trace of the covariance matrix and the identity. Therefore, the final covariance matrix is a diagonal matrix and as a result the inverse covariance matrix can be easily calculated. However, using this method, we lose information because the trace is used (the sum of the diagonal elements) for the calculation of the matrix.

Question 2(b)

From the package glmnet, the function cv.glmnet was used with the argument family="multinomial".

```
library(glmnet)
```

```
res.cv <-  cv.glmnet(Khan$xtrain,as.factor(Khan$ytrain),family="multinomial")
```

Below, the outcome of the function cv.glmnet is presented.

```
plot(res.cv)
```

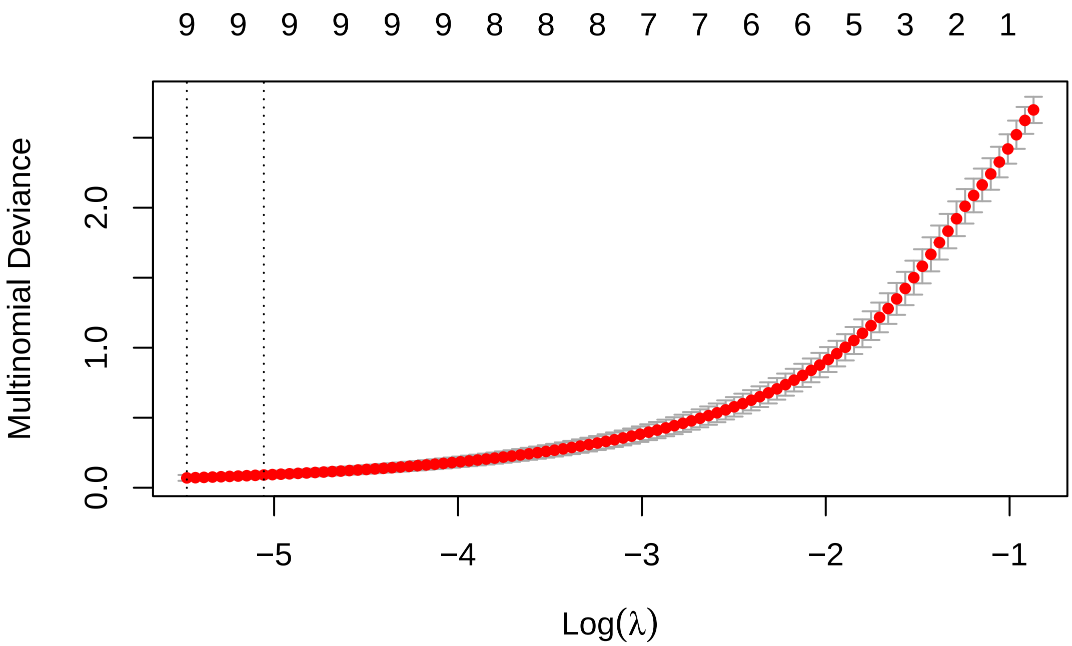

Figure 4: Cross Validation outcome

According to the plot, by using only 9 features we obtain the minimum multinomial deviance. Thus the loss function that we try to minimize is the multinomial deviance which is the negative multinomial log-likelihood loss function for multi-class classification with n classes mutually exclusive classes.

The parameter on the y axis could change to the misclassification error by including type.measure = "class" into the cv.glmnet function. Thus, the result would be:

```
res.cv.error <-  cv.glmnet(Khan$xtrain,as.factor(Khan$ytrain),
                           family="multinomial",type.measure = "class")
```

```
plot(res.cv.error)
```

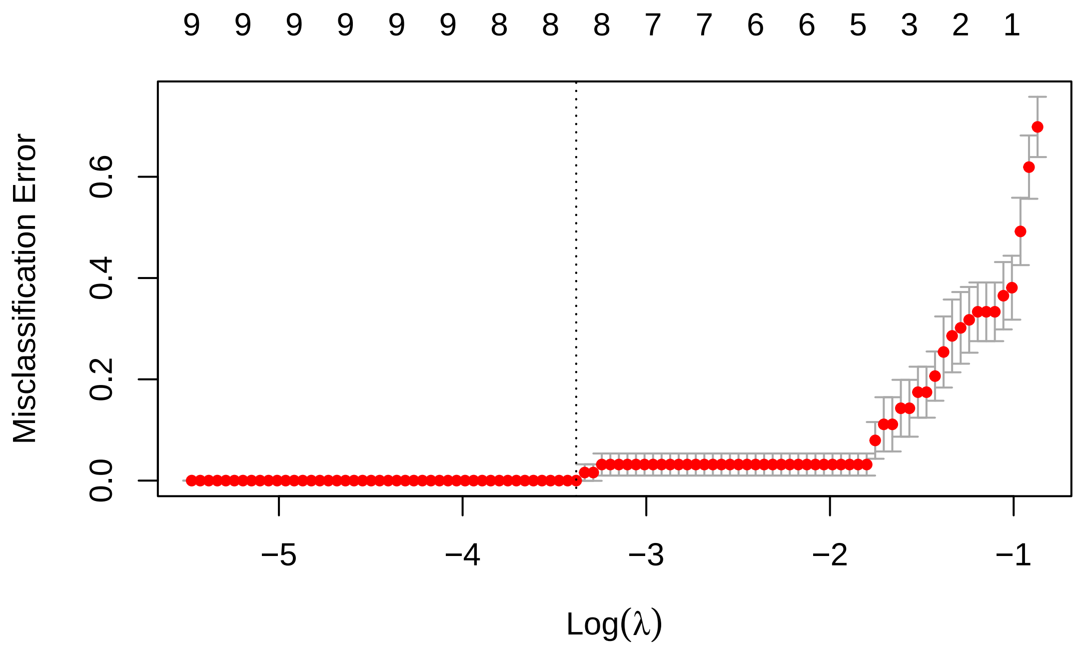

Figure 5: Cross Validation outcome

It is clear that the number of non zero variables has been changed from 9 to 8.

Question 2(c)

The function coef() was used in order to check which variables contribute to the model. For the group 1 the variables that contribute are:

```
coef <- coef(res.cv,s="lambda.1se")
which(coef$`1`!=0)
```

```
##  [1]    1    2  124  590  837  847 1067 1388 1428 2023 2199
```

For the group 2 the variables that contribute are:

```
which(coef$`2`!=0)
```

```
## [1]    1  247  546 1320 1390 1955 2051
```

For the group 3 the variables that contribute are:

```
which(coef$`3`!=0)
```

```
## [1]    1  256  576  696  743  843  880 1765 1777
```

For the group 4 the variables that contribute are:

```
which(coef$`4`!=0)
```

```
##  [1]    1  175  510  555  911 1004 1056 1106 1208 1724 1956 2047
```

Note that the first variable, which appears to all the groups, is the intercept.

Question 2(d)

The variable 124 from the group 1 was chosen. In the below figure the variable 124 is plotted against the response.

```
plot(khan_train$x.124,khan_train$y, xlab = "Variable 124", ylab = "Response")
```

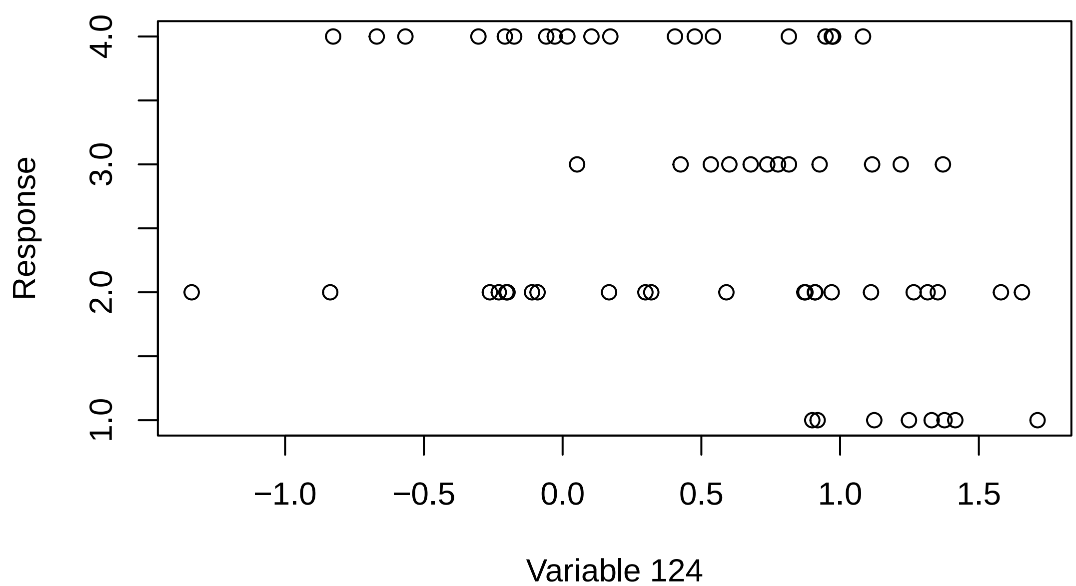

Figure 6: Relevant variable against the response

Question 2(e)

Finally, the trained model was used in order to predict the group membership of the test data. In the predict function, the type = "class" was used to obtain the classes of the test data.

```
pred <- as.numeric(predict(res.cv,newx=Khan$xtest,s="lambda.1se",type="class"))
actual.classes <- Khan$ytest
TAB <- table(actual.classes,pred)
TAB
```

```
##                pred
## actual.classes 1 2 3 4
##              1 3 0 0 0
##              2 0 6 0 0
##              3 0 0 6 0
##              4 0 0 0 5
```

```r
class.error<- 1-sum(diag(TAB))/sum(TAB)
class.error
```

```
## [1] 0
```

According to the confusion table and the misclassification error, all the observations were classified correctly.