Problem 5

Candidate 10029

03 juni, 2022

5)

```r
id <- "1HM1ytt-x9QkTHQu7bMvhBJSJWihzpZJ2" # google file ID
d.heart <- read.csv(sprintf("https://docs.google.com/uc?id=%s&export=download", id))
d.heart$HeartDisease <- as.factor(d.heart$HeartDisease)




# 70% of the sample size for training set
training_set_size <- floor(0.70 * nrow(d.heart))

set.seed(4268)
train_ind <- sample(seq_len(nrow(d.heart)), size = training_set_size)

train <- d.heart[train_ind, ]
test <- d.heart[-train_ind, ]

head(d.heart)
```

```
##   HeartDisease   BMI Smoking AlcoholDrinking Stroke PhysicalHealth MentalHealth
## 1           No 38.41      No              No     No              0           30
## 2           No 29.99      No              No     No             30           15
## 3           No 26.63      No              No     No              0            0
## 4           No 16.60      No              No     No              0           30
## 5           No 34.70      No              No     No             25            5
## 6           No 21.29      No              No     No              0            5
##   DiffWalking    Sex AgeCategory     Race PhysicalActivity GenHealth SleepTime
## 1          No Female       30-34 Hispanic               No Excellent         3
## 2         Yes   Male       65-69    White               No      Fair         5
## 3          No Female       45-49    Black               No Excellent         7
## 4          No Female       75-79    White               No      Good         9
## 5          No Female       75-79    White              Yes      Fair         7
## 6          No Female       18-24    White              Yes Very good         7
##   Asthma KidneyDisease SkinCancer
## 1     No            No         No
## 2     No            No        Yes
## 3     No            No         No
## 4     No            No         No
## 5     No            No        Yes
## 6     No            No         No
```

```r
str(d.heart)
```

```
## 'data.frame':    20000 obs. of  17 variables:
##  $ HeartDisease    : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ BMI             : num  38.4 30 26.6 16.6 34.7 ...
##  $ Smoking         : chr  "No" "No" "No" "No" ...
##  $ AlcoholDrinking : chr  "No" "No" "No" "No" ...
##  $ Stroke          : chr  "No" "No" "No" "No" ...
##  $ PhysicalHealth  : int  0 30 0 0 25 0 0 0 0 0 ...
##  $ MentalHealth    : int  30 15 0 30 5 5 0 1 0 0 ...
##  $ DiffWalking     : chr  "No" "Yes" "No" "No" ...
##  $ Sex             : chr  "Female" "Male" "Female" "Female" ...
##  $ AgeCategory     : chr  "30-34" "65-69" "45-49" "75-79" ...
##  $ Race            : chr  "Hispanic" "White" "Black" "White" ...
##  $ PhysicalActivity: chr  "No" "No" "No" "No" ...
##  $ GenHealth       : chr  "Excellent" "Fair" "Excellent" "Good" ...
##  $ SleepTime       : int  3 5 7 9 7 7 8 8 8 7 ...
##  $ Asthma          : chr  "No" "No" "No" "No" ...
##  $ KidneyDisease   : chr  "No" "No" "No" "No" ...
##  $ SkinCancer      : chr  "No" "Yes" "No" "No" ...
```

```r
names(d.heart)
```

```
##  [1] "HeartDisease"     "BMI"              "Smoking"          "AlcoholDrinking"
##  [5] "Stroke"           "PhysicalHealth"   "MentalHealth"     "DiffWalking"
##  [9] "Sex"              "AgeCategory"      "Race"             "PhysicalActivity"
## [13] "GenHealth"        "SleepTime"        "Asthma"           "KidneyDisease"
## [17] "SkinCancer"
```

a)

```r
log.fit<-glm(HeartDisease~BMI + Smoking + AlcoholDrinking + Sex + AgeCategory + Smoking*Sex + AlcoholDri

summary(log.fit)
```

```
##
## Call:
## glm(formula = HeartDisease ~ BMI + Smoking + AlcoholDrinking +
##     Sex + AgeCategory + Smoking * Sex + AlcoholDrinking * Sex,
##     family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6768  -0.4691  -0.3061  -0.1491   3.5543
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)             -7.035768   0.523698 -13.435  < 2e-16 ***
## BMI                      0.042950   0.004948   8.681  < 2e-16 ***
## SmokingYes               0.574912   0.097360   5.905 3.53e-09 ***
## AlcoholDrinkingYes      -0.314465   0.234249  -1.342  0.17945
```

```
## SexMale                     0.563997   0.097182   5.803 6.49e-09 ***
## AgeCategory25-29           -0.766922   0.868092  -0.883  0.37699
## AgeCategory30-34            1.161927   0.565991   2.053  0.04008 *
## AgeCategory35-39            0.995359   0.570593   1.744  0.08108 .
## AgeCategory40-44            1.669477   0.537230   3.108  0.00189 **
## AgeCategory45-49            1.651726   0.537050   3.076  0.00210 **
## AgeCategory50-54            2.409280   0.517575   4.655 3.24e-06 ***
## AgeCategory55-59            2.626340   0.513470   5.115 3.14e-07 ***
## AgeCategory60-64            2.880835   0.510277   5.646 1.65e-08 ***
## AgeCategory65-69            3.034768   0.509109   5.961 2.51e-09 ***
## AgeCategory70-74            3.532333   0.507697   6.958 3.46e-12 ***
## AgeCategory75-79            3.788152   0.508763   7.446 9.64e-14 ***
## AgeCategory80 or older      4.185637   0.507548   8.247  < 2e-16 ***
## SmokingYes:SexMale          0.141130   0.130081   1.085  0.27795
## AlcoholDrinkingYes:SexMale  0.001067   0.300323   0.004  0.99717
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 8265.8  on 13999  degrees of freedom
## Residual deviance: 7024.4  on 13981  degrees of freedom
## AIC: 7062.4
##
## Number of Fisher Scoring iterations: 8
```

```
ncol(train)
```

```
## [1] 17
```

```
str(train)
```

```
## 'data.frame':    14000 obs. of  17 variables:
##  $ HeartDisease    : Factor w/ 2 levels "No","Yes": 2 1 1 2 1 1 1 1 1 1 ...
##  $ BMI             : num  29.4 23.9 23.9 29.6 21.9 ...
##  $ Smoking         : chr  "Yes" "No" "No" "Yes" ...
##  $ AlcoholDrinking : chr  "No" "No" "No" "No" ...
##  $ Stroke          : chr  "No" "No" "No" "No" ...
##  $ PhysicalHealth  : int  0 0 0 0 0 0 10 0 0 30 ...
##  $ MentalHealth    : int  0 0 0 0 2 0 10 0 0 30 ...
##  $ DiffWalking     : chr  "No" "No" "No" "No" ...
##  $ Sex             : chr  "Male" "Male" "Female" "Male" ...
##  $ AgeCategory     : chr  "60-64" "55-59" "65-69" "65-69" ...
##  $ Race            : chr  "White" "White" "White" "White" ...
##  $ PhysicalActivity: chr  "Yes" "Yes" "Yes" "Yes" ...
##  $ GenHealth       : chr  "Very good" "Excellent" "Excellent" "Good" ...
##  $ SleepTime       : int  8 8 6 8 7 8 8 7 6 4 ...
##  $ Asthma          : chr  "No" "No" "No" "No" ...
##  $ KidneyDisease   : chr  "No" "No" "No" "No" ...
##  $ SkinCancer      : chr  "No" "No" "No" "No" ...
```

```r
str(train$AgeCategory)
```

```
##  chr [1:14000] "60-64" "55-59" "65-69" "65-69" "30-34" "50-54" "75-79" ...
```

We see from the string of train that there are 12 classes, 3 integer classes and 1 numerical class in this dataset and the response is also a class. We also see from the summary of the fit that there are 13 regression parameters estimated in the AgeCategory, which leads to twelve dummy variables.

b)

1) As we use dummy variables for categories, we can say that the model for our male who does not drink or smoke and is 20 years old, we have no coefficients for the age category, the drinking category or the smoking category. All these dummy variables would be 0 for him. We only need to consider the coefficients of BMI and SexMale.

Hence the regression model would be

$$P(Y=1|X) = p_i = \frac{e^{-6.46+0.04x_{i1}+0.065x_{i2}}}{1+e^{-6.46+0.04x_{i1}+0.065x_{i2}}}$$

Where I have rounded of the coefficients down to two decimals. found from the summary of the logistic regression fit. 2)

```r
log.fit2<-glm(HeartDisease~BMI + Smoking + AlcoholDrinking + Sex + AgeCategory + Smoking*Sex , data=trai

anova(log.fit, log.fit2, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: HeartDisease ~ BMI + Smoking + AlcoholDrinking + Sex + AgeCategory +
##     Smoking * Sex + AlcoholDrinking * Sex
## Model 2: HeartDisease ~ BMI + Smoking + AlcoholDrinking + Sex + AgeCategory +
##     Smoking * Sex
##   Resid. Df Resid. Dev Df    Deviance Pr(>Chi)
## 1     13981     7024.4
## 2     13982     7024.4 -1 -1.2623e-05   0.9972
```

We fit one model without the interaction term and see that from the F-test of this, the p-value is to high to say that there is evidence that the effect of drinking alcohol differs between male and female.

3) I think the purpose was inference. Even though the logistic regression is a predictor of 0/1, we compare the effects of different predictors and how they effect the risk of getting heart disease. We can thus find the most impacting predictors and thus we can inform people if they are at a risk of getting heart disease, and how one can avoid the risk. As the logistic regression is parametric, we can tell which parameters are significant. We could have used a non-parametric approach if we just wanted to predict if a person was going to get heart disease.

c)

```
lda.fit = lda(HeartDisease ~., data = train, type="response")

lda_prob<-predict(lda.fit, newdata=test)$class

lda_pred<-predict(lda.fit, newdata=test)$posterior

conf<-table(lda_prob, test$HeartDisease)

conf
```

```
##
## lda_prob   No  Yes
##      No  5311  417
##      Yes  161  111
```

```
testerr.lda<-(conf[2,1]+conf[1,2])/(sum(conf))
testerr.lda
```

```
## [1] 0.09633333
```

```
qda.fit<-qda(HeartDisease ~., data = train, type="response")

qda_prob<-predict(qda.fit, newdata=test)$class
qda_pred<-predict(qda.fit, newdata=test)$posterior

confqda<-table(qda_prob, test$HeartDisease)

testerr.qda<-(confqda[2,1]+confqda[1,2])/(sum(confqda))
testerr.qda
```

```
## [1] 0.321
```

We see that the test error rate for LDA is 0.0985 and for QDA it is 0.3002

3)

```
lda.roc = roc(response=test$HeartDisease, predictor=lda_pred[,2], direction="<")
qda.roc = roc(response=test$HeartDisease, predictor=qda_pred[,2], direction="<")

auc(lda.roc)
```

```
## Area under the curve: 0.8186
```

```
auc(qda.roc)
```

```
## Area under the curve: 0.7971
```

We see that the AUC for the LDA is 0.8187 and for the QDA it is 0.8029

4) KNN does not work well with large datasets, and we have many predictors the dimension is high. Then we would experience the curse of dimensionality, since in such high dimensions, points tend to lie very far from eachother.

d) For this task I will chose to build a random forest. This is because it utilizes bagging which is very powerful and also I will make the number of predictors to consider at each split the square root of the total number of predictors, $\sqrt{p} = \sqrt{16} = 4$, as this has been shown to give good results for classification. As the number of trees is not a tuning parameter here, I will check the OOB error and then I can pick a sufficiently large number of trees.

```r
ncol(train)
```

```
## [1] 17
```

```r
round(sqrt(ncol(train)-1))
```

```
## [1] 4
```

```r
rf.fit<-randomForest(HeartDisease~., data=train, mtry=round(sqrt(ncol(train)-1)), ntree=500, importance=

rf.fit$err.rate[, 1] #Check OOB error
```

```
##   [1] 0.12679945 0.12446454 0.12194888 0.11757172 0.11655030 0.11071292
##   [7] 0.11007740 0.10890436 0.10506063 0.10511876 0.10091347 0.09953355
##  [13] 0.09886803 0.09723812 0.09698877 0.09565342 0.09554095 0.09424795
##  [19] 0.09517005 0.09323426 0.09350000 0.09371429 0.09214286 0.09292857
##  [25] 0.09214286 0.09221429 0.09192857 0.09171429 0.09078571 0.09107143
##  [31] 0.09135714 0.09085714 0.09107143 0.09021429 0.09035714 0.08914286
##  [37] 0.09014286 0.08992857 0.08942857 0.08900000 0.08992857 0.08928571
##  [43] 0.08907143 0.08900000 0.08885714 0.08878571 0.08842857 0.08921429
##  [49] 0.08907143 0.08907143 0.08871429 0.08835714 0.08792857 0.08892857
##  [55] 0.08871429 0.08921429 0.08942857 0.08900000 0.08857143 0.08857143
##  [61] 0.08842857 0.08857143 0.08850000 0.08807143 0.08800000 0.08828571
##  [67] 0.08807143 0.08800000 0.08807143 0.08821429 0.08757143 0.08750000
##  [73] 0.08742857 0.08764286 0.08735714 0.08721429 0.08728571 0.08735714
##  [79] 0.08728571 0.08692857 0.08714286 0.08785714 0.08735714 0.08735714
##  [85] 0.08757143 0.08771429 0.08771429 0.08750000 0.08728571 0.08750000
##  [91] 0.08757143 0.08764286 0.08735714 0.08721429 0.08771429 0.08721429
##  [97] 0.08735714 0.08721429 0.08692857 0.08700000 0.08707143 0.08721429
## [103] 0.08721429 0.08728571 0.08692857 0.08671429 0.08664286 0.08700000
## [109] 0.08707143 0.08707143 0.08700000 0.08700000 0.08671429 0.08685714
## [115] 0.08700000 0.08642857 0.08657143 0.08728571 0.08685714 0.08671429
## [121] 0.08678571 0.08700000 0.08692857 0.08721429 0.08692857 0.08742857
## [127] 0.08714286 0.08771429 0.08707143 0.08707143 0.08714286 0.08764286
## [133] 0.08778571 0.08764286 0.08785714 0.08757143 0.08764286 0.08721429
## [139] 0.08707143 0.08707143 0.08721429 0.08700000 0.08692857 0.08735714
## [145] 0.08757143 0.08778571 0.08771429 0.08735714 0.08764286 0.08764286
## [151] 0.08778571 0.08785714 0.08807143 0.08757143 0.08785714 0.08771429
## [157] 0.08771429 0.08778571 0.08771429 0.08821429 0.08800000 0.08800000
## [163] 0.08800000 0.08792857 0.08821429 0.08821429 0.08792857 0.08785714
## [169] 0.08785714 0.08814286 0.08814286 0.08807143 0.08800000 0.08807143
## [175] 0.08835714 0.08814286 0.08821429 0.08828571 0.08807143 0.08785714
```

```
## [181] 0.08792857 0.08792857 0.08778571 0.08785714 0.08785714 0.08735714
## [187] 0.08764286 0.08764286 0.08771429 0.08742857 0.08750000 0.08742857
## [193] 0.08742857 0.08764286 0.08742857 0.08771429 0.08742857 0.08750000
## [199] 0.08764286 0.08778571 0.08750000 0.08742857 0.08721429 0.08714286
## [205] 0.08742857 0.08742857 0.08700000 0.08728571 0.08728571 0.08707143
## [211] 0.08721429 0.08721429 0.08700000 0.08707143 0.08735714 0.08707143
## [217] 0.08721429 0.08764286 0.08757143 0.08785714 0.08771429 0.08764286
## [223] 0.08757143 0.08750000 0.08721429 0.08742857 0.08750000 0.08750000
## [229] 0.08707143 0.08728571 0.08742857 0.08750000 0.08742857 0.08771429
## [235] 0.08778571 0.08778571 0.08764286 0.08757143 0.08750000 0.08750000
## [241] 0.08757143 0.08735714 0.08764286 0.08742857 0.08771429 0.08764286
## [247] 0.08750000 0.08742857 0.08750000 0.08764286 0.08742857 0.08764286
## [253] 0.08757143 0.08742857 0.08750000 0.08771429 0.08750000 0.08750000
## [259] 0.08721429 0.08735714 0.08742857 0.08721429 0.08714286 0.08714286
## [265] 0.08707143 0.08721429 0.08707143 0.08735714 0.08714286 0.08742857
## [271] 0.08735714 0.08742857 0.08721429 0.08735714 0.08721429 0.08707143
## [277] 0.08728571 0.08714286 0.08735714 0.08750000 0.08742857 0.08728571
## [283] 0.08742857 0.08714286 0.08700000 0.08714286 0.08714286 0.08735714
## [289] 0.08728571 0.08728571 0.08750000 0.08764286 0.08757143 0.08750000
## [295] 0.08771429 0.08785714 0.08785714 0.08785714 0.08757143 0.08750000
## [301] 0.08764286 0.08771429 0.08750000 0.08757143 0.08778571 0.08742857
## [307] 0.08778571 0.08764286 0.08757143 0.08764286 0.08757143 0.08778571
## [313] 0.08778571 0.08771429 0.08771429 0.08757143 0.08757143 0.08785714
## [319] 0.08771429 0.08771429 0.08742857 0.08750000 0.08750000 0.08750000
## [325] 0.08764286 0.08742857 0.08742857 0.08742857 0.08764286 0.08750000
## [331] 0.08742857 0.08764286 0.08771429 0.08750000 0.08771429 0.08757143
## [337] 0.08757143 0.08771429 0.08764286 0.08757143 0.08778571 0.08764286
## [343] 0.08785714 0.08764286 0.08764286 0.08757143 0.08778571 0.08764286
## [349] 0.08771429 0.08771429 0.08764286 0.08757143 0.08757143 0.08757143
## [355] 0.08757143 0.08764286 0.08764286 0.08778571 0.08764286 0.08764286
## [361] 0.08778571 0.08757143 0.08764286 0.08764286 0.08750000 0.08764286
## [367] 0.08757143 0.08750000 0.08742857 0.08728571 0.08742857 0.08757143
## [373] 0.08764286 0.08771429 0.08771429 0.08757143 0.08778571 0.08764286
## [379] 0.08778571 0.08771429 0.08785714 0.08778571 0.08771429 0.08792857
## [385] 0.08764286 0.08785714 0.08807143 0.08807143 0.08800000 0.08800000
## [391] 0.08792857 0.08792857 0.08785714 0.08750000 0.08764286 0.08750000
## [397] 0.08757143 0.08792857 0.08792857 0.08792857 0.08771429 0.08792857
## [403] 0.08792857 0.08771429 0.08764286 0.08750000 0.08771429 0.08785714
## [409] 0.08785714 0.08792857 0.08778571 0.08792857 0.08778571 0.08792857
## [415] 0.08764286 0.08771429 0.08757143 0.08778571 0.08785714 0.08785714
## [421] 0.08785714 0.08800000 0.08785714 0.08792857 0.08785714 0.08792857
## [427] 0.08785714 0.08778571 0.08771429 0.08785714 0.08778571 0.08771429
## [433] 0.08778571 0.08778571 0.08785714 0.08778571 0.08771429 0.08778571
## [439] 0.08771429 0.08764286 0.08764286 0.08750000 0.08757143 0.08764286
## [445] 0.08764286 0.08757143 0.08764286 0.08757143 0.08764286 0.08757143
## [451] 0.08742857 0.08742857 0.08742857 0.08757143 0.08742857 0.08750000
## [457] 0.08750000 0.08757143 0.08757143 0.08735714 0.08735714 0.08721429
## [463] 0.08728571 0.08750000 0.08735714 0.08728571 0.08750000 0.08728571
## [469] 0.08728571 0.08742857 0.08742857 0.08742857 0.08750000 0.08742857
## [475] 0.08742857 0.08742857 0.08750000 0.08750000 0.08757143 0.08750000
## [481] 0.08742857 0.08757143 0.08750000 0.08750000 0.08757143 0.08750000
## [487] 0.08750000 0.08764286 0.08764286 0.08757143 0.08764286 0.08757143
## [493] 0.08764286 0.08771429 0.08785714 0.08750000 0.08757143 0.08757143
## [499] 0.08757143 0.08764286
```

```r
rf.pred<-predict(rf.fit, newdata=test, type="class")

table(rf.pred, test$HeartDisease)
```

```
##
## rf.pred   No  Yes
##     No  5436  496
##     Yes   36   32
```

```r
confrf<-table(rf.pred, test$HeartDisease)

testerr.rf<-(confrf[2,1]+confrf[1,2])/(sum(confrf))
testerr.rf
```

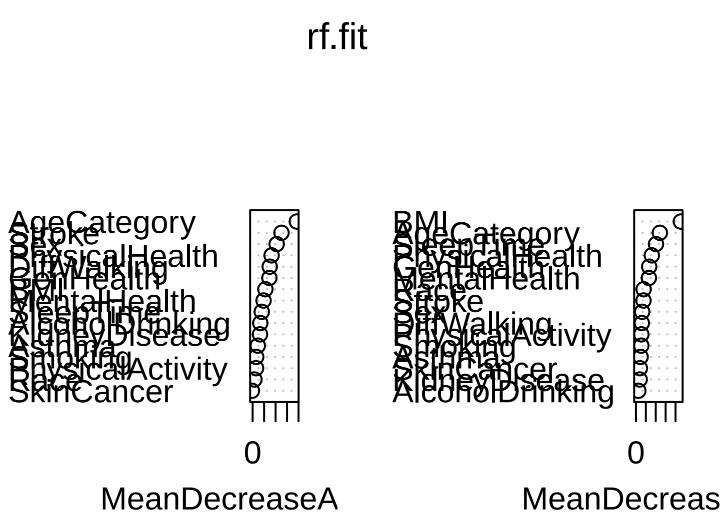
```
## [1] 0.08866667
```

We can see that the OOB error rate is steady from about 50 trees an onward, so 500 trees is more than enough, but we do not need to reduce it.

As the test error of the random forest is 0.0917 we have succeded in producing a classification tool with lower test error than the LDA and the QDA.

2)

```r
varImpPlot(rf.fit)
```



The variance importance plot show that for measuring node impurity, i.e. the Gini index, BMI, AgeCategory and SleepTime are the three most important factors.

```r
B = 1000
estimator = rep(NA, B)
n=length(d.heart$BMI)
```

```r
for (b in 1:B) {
    thisboot = sample(x = d.heart$BMI, size = n, replace = TRUE)
    estimator[b] = mean(thisboot) - median(thisboot)
}
sd(estimator)
```

```
## [1] 0.03168301
```