

# Block 1 - Assignment 1

Tejashree R Mastamardi

Assignment 1: Spam Classification with nearest neighbors

```
library(readxl)
spambase <- read_excel("spambase.xlsx")
View(spambase)
```

Question 1.1 Import the data into R and divide it into training and test sets (50%/50%)

```
n=dim(spambase)[1]
set.seed(12345)
id=sample(1:n, floor(n*0.5))
train=spambase[id,]
test=spambase[-id,]
```

Question 1.2 Use logistic regression (functions glm(), predict()) to classify the training and test data by the classification principle  $\hat{Y} = 1$  if  $p(Y=1|X) > 0.5$ , otherwise  $\hat{Y} = 0$  and report the confusion matrices (use table()) and the misclassification rates for training and test data. Analyse the obtained results.

```
GLM <- function(p){
  glm_model <- glm(Spam~., family = binomial(link="logit"), data = train)
  glm_prob <- predict(glm_model, type = "response")
  glm_pred <- ifelse(glm_prob > p, "Spam", "Not Spam")
  tab <- table(train$Spam, glm_pred)
  MSE <- (1 - (sum(diag(tab))/sum(tab)))*100
  tab1 <- table(test$Spam, glm_pred)
  MSE1 <- (1 - (sum(diag(tab1))/sum(tab1)))*100
  return(list("Confmat-train" = tab, "MSEtrain" = MSE,
             "Confmat-test" = tab1, "MSEtest" = MSE1))
}
GLM(0.5)
```

```
## $'Confmat-train'
##      glm_pred
##      Not Spam Spam
## 0      804  127
## 1       93  346
##
## $MSEtrain
## [1] 16.05839
##
## $'Confmat-test'
##      glm_pred
##      Not Spam Spam
```

```
##    0      616  335
##    1      281  138
##
## $MSEtest
## [1] 44.9635
```

Question 1.3 Use logistic regression (functions `glm()`, `predict()`) to classify the training and test data by the classification principle  $\hat{Y} = 1$  if  $p(Y=1|X) > 0.9$ , otherwise  $\hat{Y} = 0$  and report the confusion matrices (use `table()`) and the misclassification rates for training and test data. Analyse the obtained results.

GLM(0.9)

```
## $'Confmat-train'
##      glm_pred
##      Not Spam Spam
##    0      928    3
##    1      378   61
##
## $MSEtrain
## [1] 27.81022
##
## $'Confmat-test'
##      glm_pred
##      Not Spam Spam
##    0      910   41
##    1      396   23
##
## $MSEtest
## [1] 31.89781
```

Question 1.4 Use standard classifier `kkn()` with  $K=30$  from package `kkn`, report the misclassification rates for the training and test data and compare the results with step 2.

```
#library(kkn)
kkn_model <- function(a, b,X){
  model1 <- kkn(Spam~., a, b, k=X)
  fitted_model <- predict(model1)
  kkn_train <- ifelse(fitted_model > 0.5, "Spam", "Not Spam")
  tab1 <- table(train$Spam, kkn_train)
  MSE1 <- (1 - (sum(diag(tab1))/sum(tab1)))*100
  tab2 = table(test$Spam, kkn_train)
  MSE2 = (1 - (sum(diag(tab2))/sum(tab2)))*100
  return(list("confmat_train"=tab1, "MSEtrain"=MSE1,
             "confmat_test"=tab2,"MSEtest"=MSE2))
}

kkn_model(train, train, X=30)
```

```
## $confmat_train
##      kkn_train
##      Not Spam Spam
##    0      779  152
```

```
##      1      77 362
##
## $MSEtrain
## [1] 16.71533
##
## $confmat_test
##      knnn_train
##      Not Spam Spam
##      0      580 371
##      1      276 143
##
## $MSEtest
## [1] 47.22628
```

```
knnc_model(train, test, X=30)
```

```
## $confmat_train
##      knnn_train
##      Not Spam Spam
##      0      589 342
##      1      293 146
##
## $MSEtrain
## [1] 46.35036
##
## $confmat_test
##      knnn_train
##      Not Spam Spam
##      0      702 249
##      1      180 239
##
## $MSEtest
## [1] 31.31387
```

Question 1.5 Repeat step 4 for  $K=1$  and compare the results with step 4. What effect does the decrease of  $K$  lead to and why?

```
knnc_model(train, train, X=1)
```

```
## $confmat_train
##      knnn_train
##      Not Spam Spam
##      0      931  0
##      1       0 439
##
## $MSEtrain
## [1] 0
##
## $confmat_test
##      knnn_train
##      Not Spam Spam
##      0      633 318
##      1      298 121
```

```
##
## $MSEtest
## [1] 44.9635
```

```
kknnc_model(train, test, X=1)
```

```
## $confmat_train
##      knnc_train
##      Not Spam Spam
##    0      560 371
##    1      269 170
##
## $MSEtrain
## [1] 46.71533
##
## $confmat_test
##      knnc_train
##      Not Spam Spam
##    0      644 307
##    1      185 234
##
## $MSEtest
## [1] 35.91241
```

#### Assignment 4: Linear Regression and Regularization

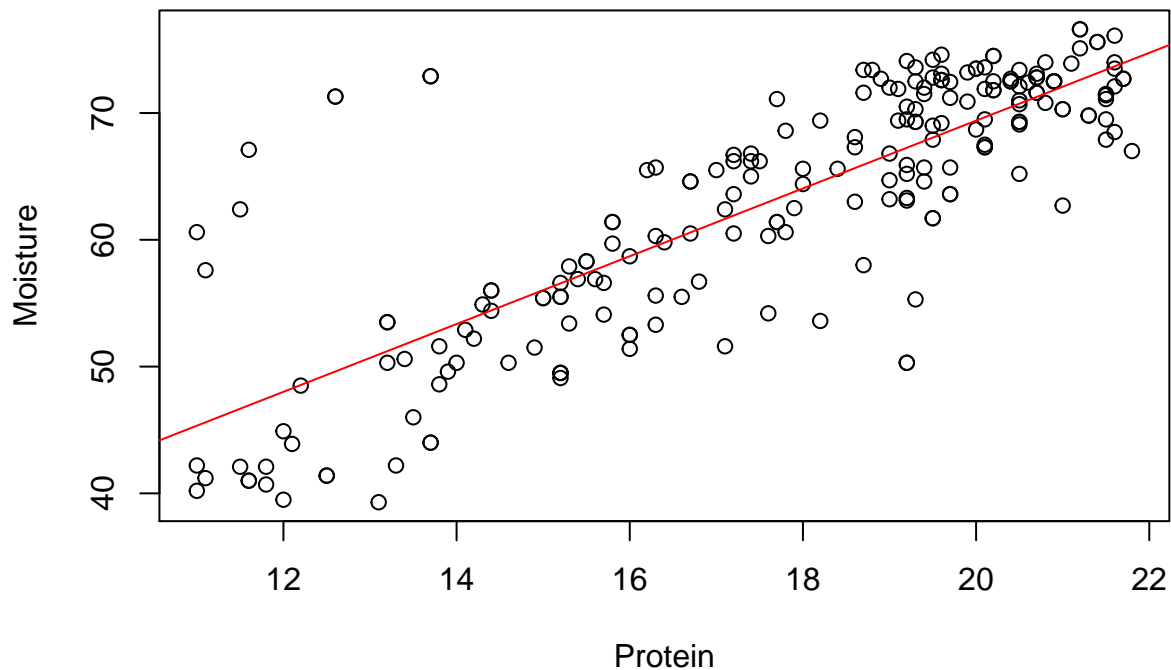
```
library(readxl)
tecator <- read_excel("tecator.xlsx")
#View(tecator)
```

Question 4.1 Import data to R and create a plot of Moisture versus Protein. Do you think that these data are described well by a linear model?

```
#Plot of Moisture vs protein
plot(Moisture~Protein, data = tecator)

#do you think that these data are described well by a linear model
Model <- lm(Moisture~Protein, data = tecator)

plot(Moisture~Protein, data = tecator)
abline(Model, col = "red")
```

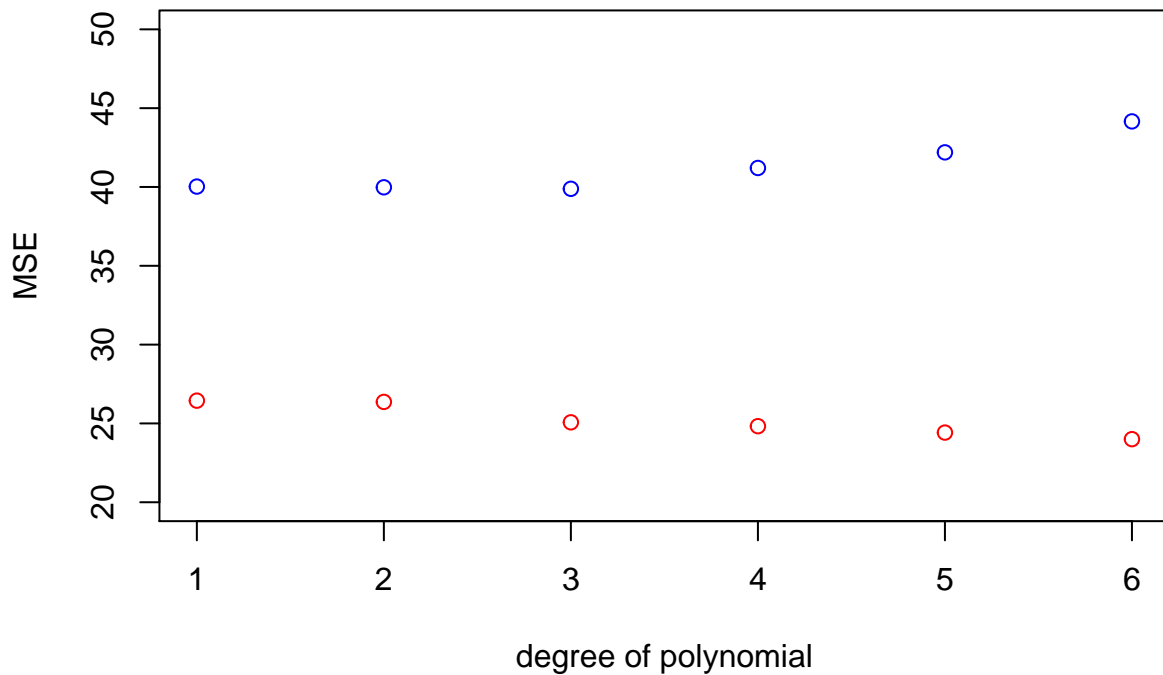


Yes, the data is well described by a linear model

Question 4.3 Divide the data into training and validation sets( 50%/50%) and fit models  $M_i, i=1 \dots 6$ . For each model, record the training and the validation MSE and present a plot showing how training and validation MSE depend on  $i$  (write some R code to make this plot). Which model is best according to the plot? How do the MSE values change and why? Interpret this picture in terms of bias-variance tradeoff.

```
n=dim(tecator)[1]
set.seed(12345)
id=sample(1:n, floor(n*0.5))
train=tecator[id,]
valid=tecator[-id,]

ModelPlot <- function(){
  training_MSE <- rep(0,6)
  validation_MSE <- rep(0,6)
  for (i in 1:6) {
    model <- lm(Moisture~poly(Protein,i), data = train)
    fitted_train <- predict(model, train)
    fitted_valid <- predict(model, valid)
    training_MSE[i] <- mean((train$Moisture - fitted_train)^2) #mean((Y-Y_hat)^2)
    validation_MSE[i] <- mean((valid$Moisture - fitted_valid)^2)
  }
  plot(seq(1:6), training_MSE, col = "red", ylim = c(20,50), ylab = "MSE", xlab = "degree of polynomial",
       points(seq(1:6), validation_MSE, col = "blue"))
}
ModelPlot()
```



Question 4.4 Perform variable selection of a linear model in which Fat is response and Channel1-Channel100 are predictors by using stepAIC. Comment on how many variables were selected.

```
library(mltools)
library(MASS)
new_data <- as.data.frame(as.matrix(tecator[,2:102]))

#Perform variable selection of a linear model in which Fat is response and Channel1-Channel100 are pred
myAIC <- lm(Fat ~., new_data)
step <- stepAIC(myAIC, direction = "both", trace = FALSE)

#Comment on how many variables were selected
selected_variables <- length(step$coefficients) - 1 #removing the intercept term
selected_variables
```

```
## [1] 63
```

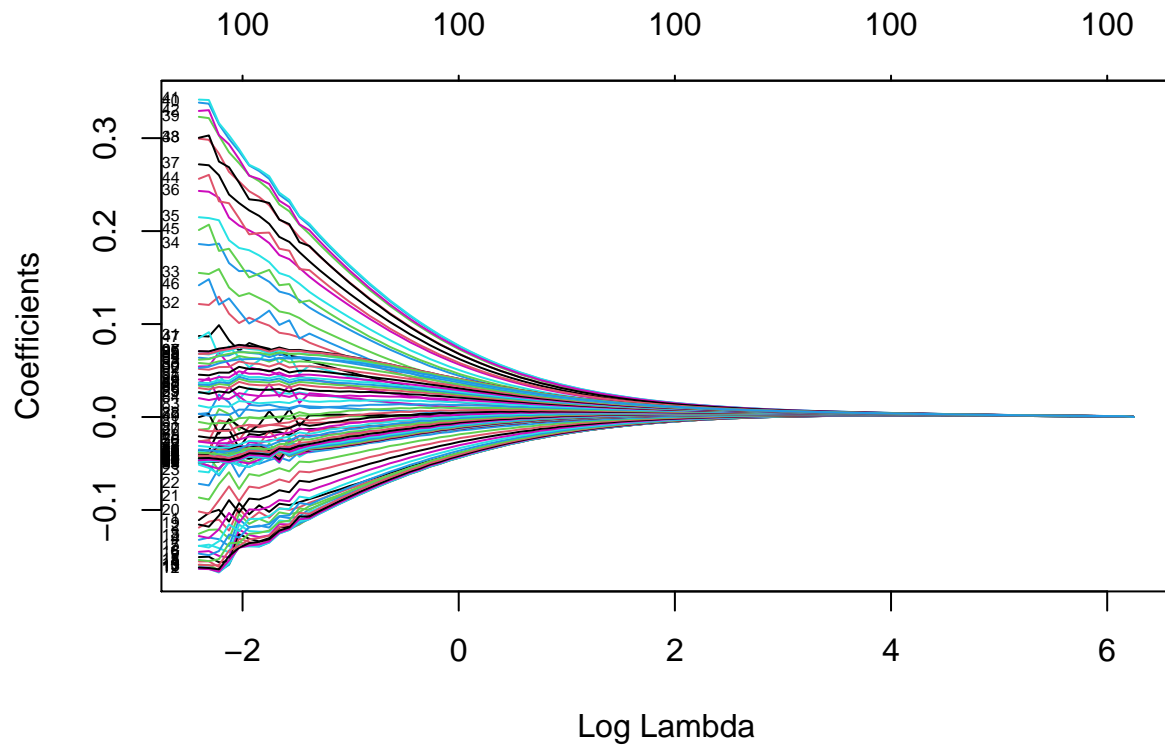
Question 4.5 Fit a Ridge regression model with the same predictor and response variables. Present a plot showing how model coefficients depend on the log of the penalty factor lambda and report how the coefficients change with lambda.

```
library(readxl)
library(glmnet)

covariates <- scale(new_data[,1:100])
```

```
response <- scale(new_data[,101])

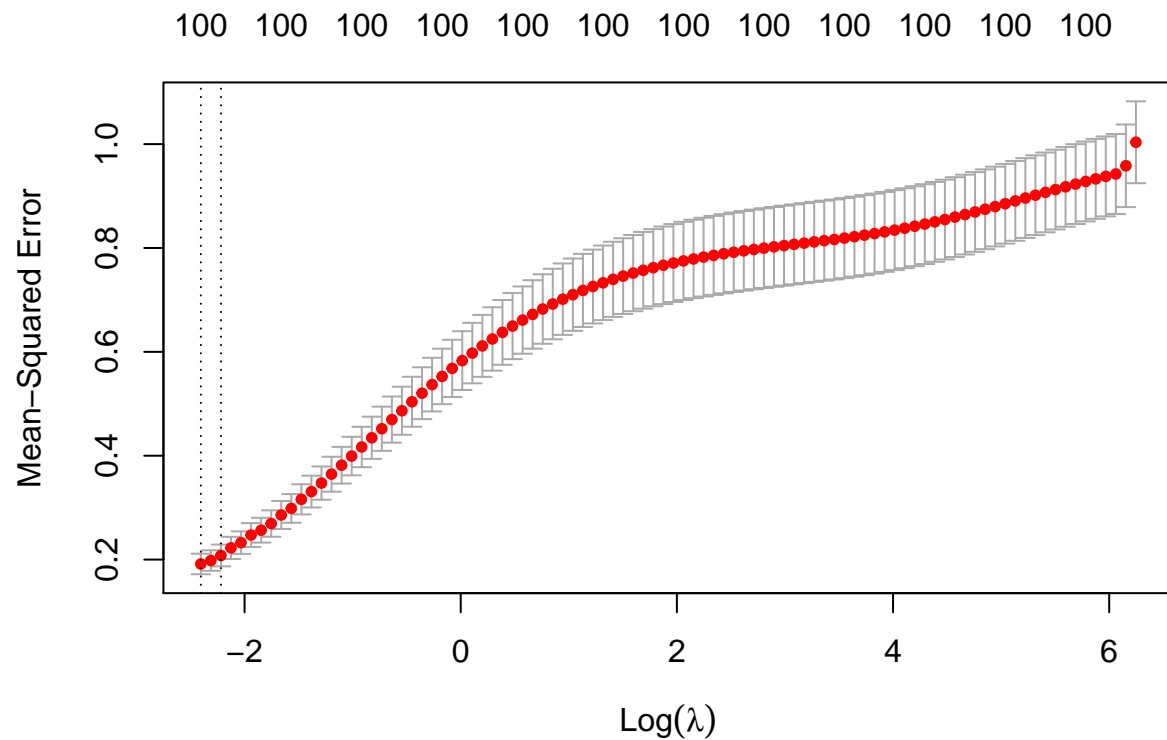
model <- glmnet(as.matrix(covariates), response, alpha=0, family = "gaussian")
plot(model, xvar = "lambda", label = TRUE)
```



```
model=cv.glmnet(as.matrix(covariates),response, alpha=0,family="gaussian")#Lambda value is selected using
model$lambda.min
```

```
## [1] 0.09030492
```

```
plot(model)
```

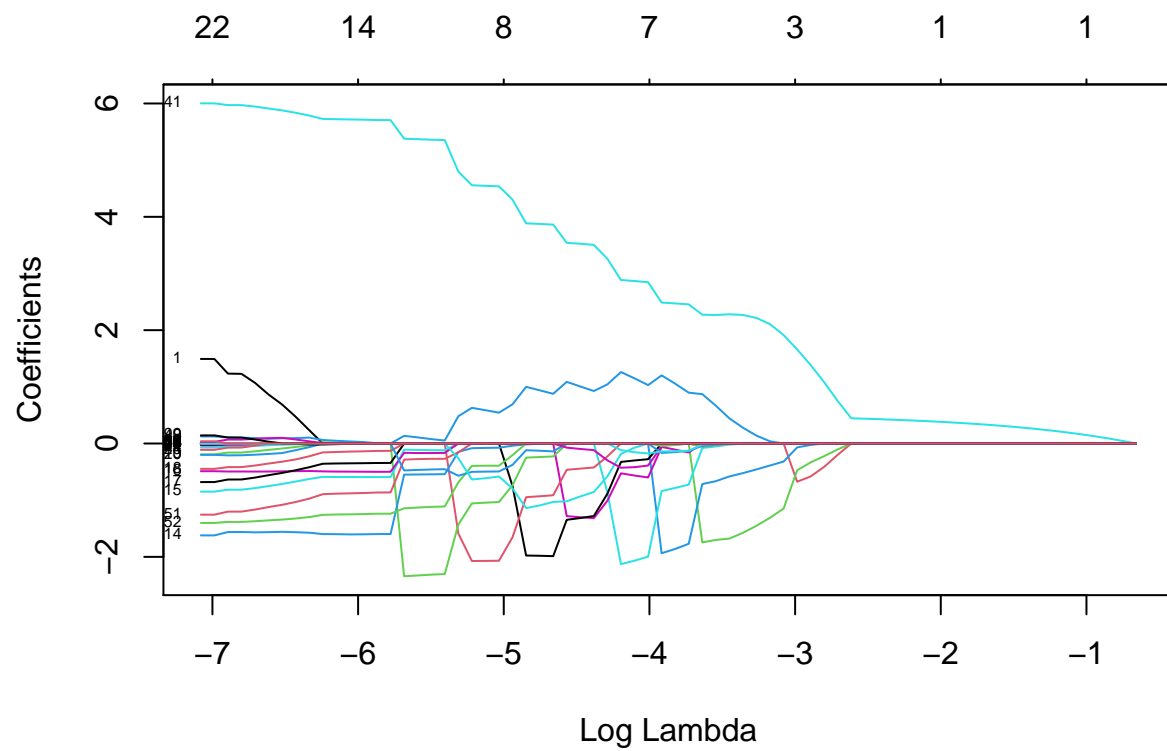


Question 4.6 Fit a LASSO regression model with the same predictor and response variables. Present a plot showing how model coefficients depend on the log of the penalty factor lambda and report how the coefficients change with lambda.

```
covariates <- scale(new_data[,1:100])
response <- scale(new_data[,101])

model <- glmnet(as.matrix(covariates), response, alpha=1, family = "gaussian")
plot(model, xvar = "lambda", label = TRUE)
```

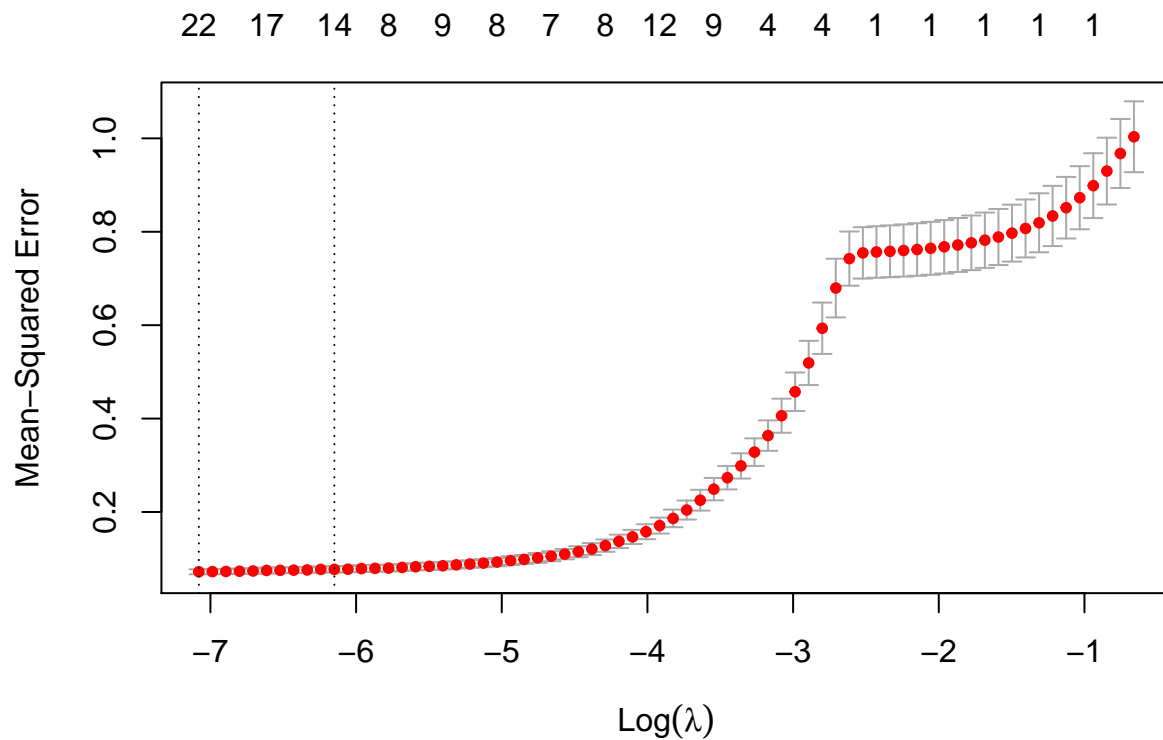




```
model=cv.glmnet(as.matrix(covariates),response, alpha=1,family="gaussian")
model$lambda.min
```

```
## [1] 0.0008421867
```

```
plot(model)
```

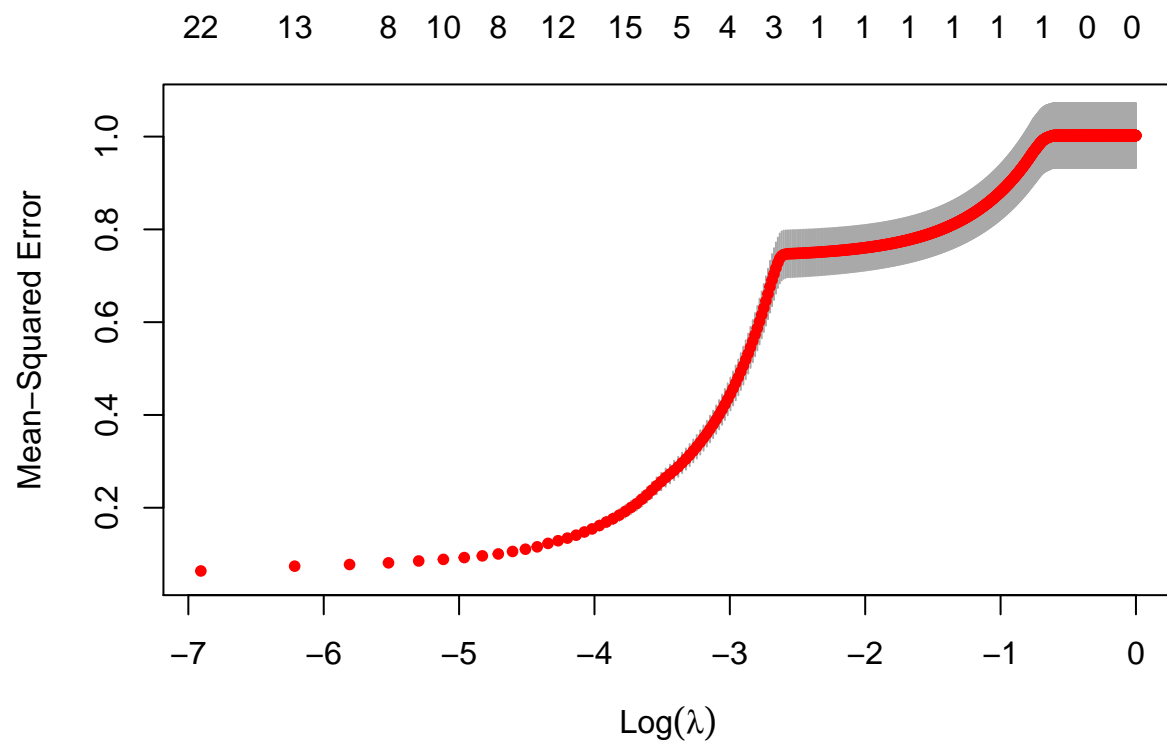


Question 4.7 Use cross-validation to find the optimal LASSO model, report the optimal lambda and how many variables were chosen by the model and make conclusions.

```
scaled_data <- scale(tecator[,2:102])

covariates <- scaled_data[,1:100]
response <- scaled_data[,101]

model_cv <- cv.glmnet(as.matrix(covariates), response, alpha = 1, family = "gaussian", lambda = seq(0,1
plot(model_cv, xvar="lambda", label=TRUE)
```



```
#optimal Lambda
```

```
opt_lambda <- model_cv$lambda.min
```

```
opt_lambda
```

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

```
cat(paste("number of variables chosen = ",length(coef(model_cv,s="lambda.min"))-1))
```

```
## number of variables chosen = 100
```