Project 10

Contents

Problem 29

```
library(caret)
library(glmnet)
library(pROC)
```

```
load(file='yeastStorey.rda')
```

```
print(paste("Number of samples (N):", nrow(data)))
```

```
## [1] "Number of samples (N): 112"
```

```
print(paste("Number of features (p):", ncol(data)))
```

```
## [1] "Number of features (p): 232"
```

Splitting data into training and testing subsets

```
set.seed(42)
trainIndex <- createDataPartition(data$Marker, p=0.7, list=FALSE, times=1)
trainData <- data[trainIndex,]
testData <- data[-trainIndex,]
```

Cross-validation of elastic-net model

```
# Preparing data for cv.glmnet
x <- trainData[, !(names(trainData) %in% c("Marker"))]
x <- as.matrix(x)
y <- trainData$Marker
```

```
# Executing 10-fold CV for each value of alpha
foldid <- sample(1:10, size=length(y), replace=TRUE)
alphas <- seq(0, 1, by=0.1)

elasticNetCVAlpha <- function(alpha) {
  cv.glmnet(x, y, family="binomial", alpha=alpha, nfolds=10, foldid=foldid)
}
```

```
resultsCV <- lapply(alphas, elasticNetCVAlpha)
```

```
# Finding the optimal alpha
minMeanCVMIdx <- 1
minMeanCVM <- mean(resultsCV[[1]]$cvm)
for(i in 1:length(alphas)) {
  if(minMeanCVM > mean(resultsCV[[i]]$cvm)) {
    minMeanCVM <- mean(resultsCV[[i]]$cvm)
    minMeanCVMIdx <- i
  }
  # Reporting mean of mean cross-validated error of each alpha
  print(paste0("alpha=", alphas[i], "; error=", mean(resultsCV[[i]]$cvm)))
}
```

```
## [1] "alpha=0; error=1.41826065852264"
## [1] "alpha=0.1; error=1.207662476829"
## [1] "alpha=0.2; error=1.07852495274017"
## [1] "alpha=0.3; error=0.978145211748884"
## [1] "alpha=0.4; error=0.892525261344348"
## [1] "alpha=0.5; error=0.81568741249565"
## [1] "alpha=0.6; error=0.745849929007219"
## [1] "alpha=0.7; error=0.678769142955298"
## [1] "alpha=0.8; error=0.60340112911262"
## [1] "alpha=0.9; error=0.519596711095501"
## [1] "alpha=1; error=0.430489062157447"
```

Finding optimal alpha

α with which mean of mean cross-validated error is the smallest: $\alpha = 1$. This α will be considered as optimal.

```
print(paste("Min. mean of mean cross-validated error:", minMeanCVM))
```

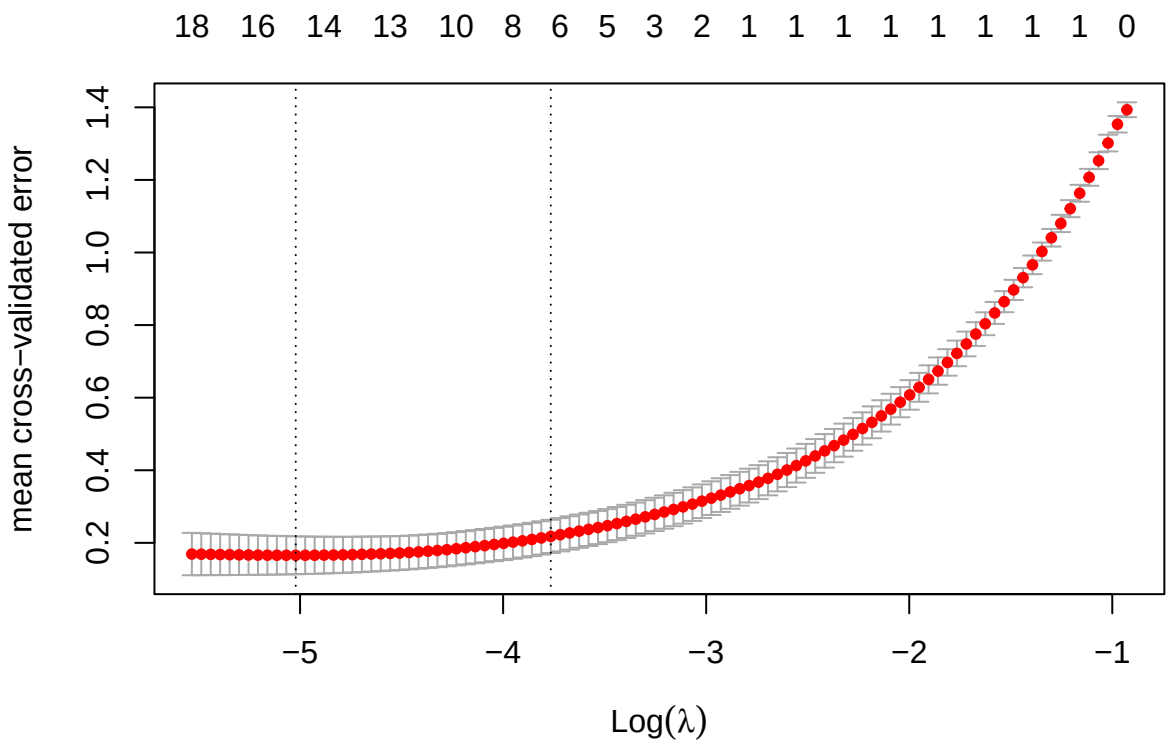
```
## [1] "Min. mean of mean cross-validated error: 0.430489062157447"
```

```
optimalAlphaIdx <- minMeanCVMIdx
optimalAlpha <- alphas[optimalAlphaIdx]
```

Plotting mean cross-validated error

Cross-validated error function is binomial deviance. The plot of $log(\lambda)$ versus mean cross-validated error is done using results retrieved with $\alpha = 1$.

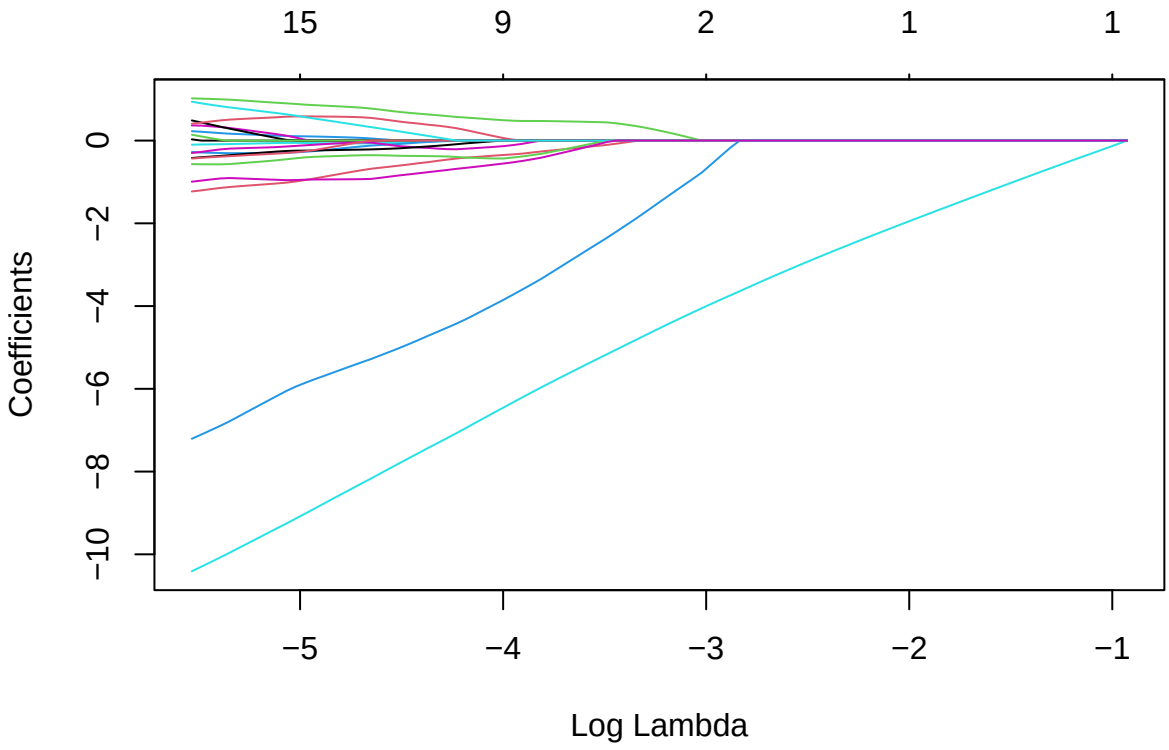
```
plot(resultsCV[[optimalAlphaIdx]], ylab="mean cross-validated error")
```



Plotting trace curve of coefficients

The plot of $log(\lambda)$ versus coefficients is done using results retrieved with $\alpha = 1$.

```
plot(resultsCV[[optimalAlphaIdx]]$glmnet.fit, "lambda")
```



Picking optimal lambda

```
optimalLambdaIdx <- which.min(resultsCV[[optimalAlphaIdx]]$cvm)
optimalLambda <- resultsCV[[optimalAlphaIdx]]$lambda[[optimalLambdaIdx]]
```

Optimal $\lambda = 0.0065961$.

Fitting model on the whole training set

```
trainedModel <- glmnet(x, y, family="binomial", alpha=optimalAlpha, lambda=optimalLambda)
trainedModel
```

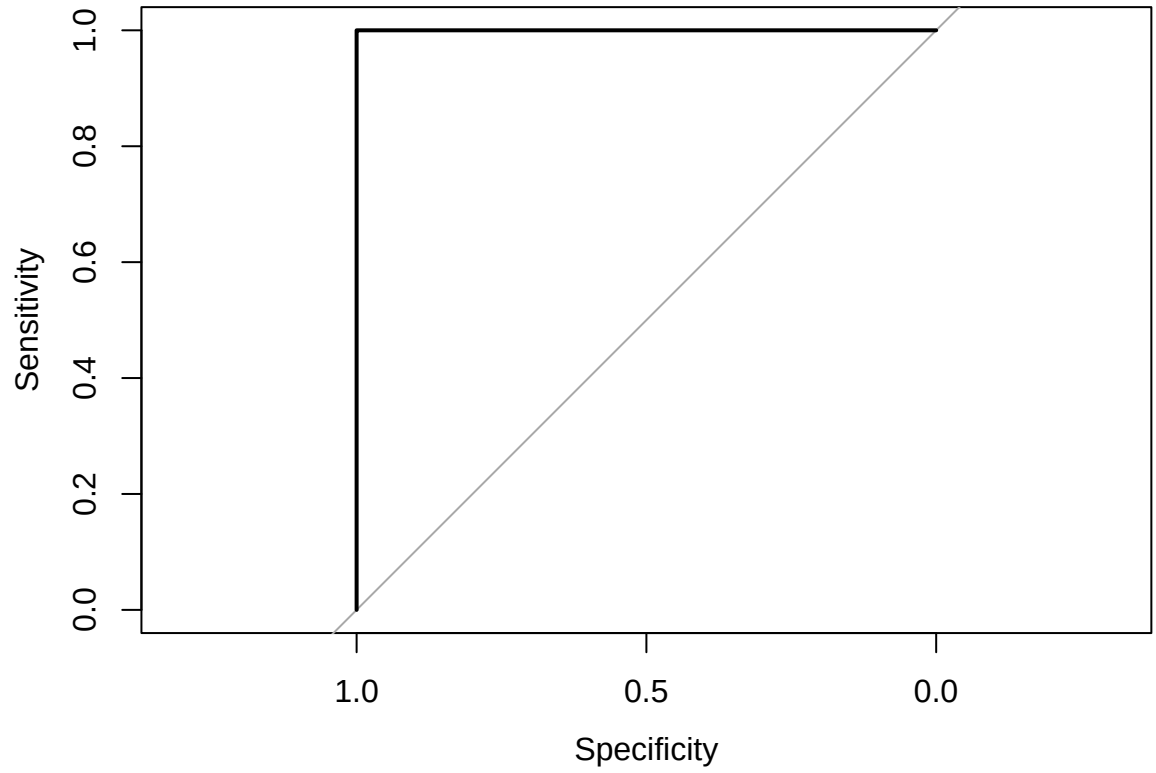
```
##
## Call:  glmnet(x = x, y = y, family = "binomial", alpha = optimalAlpha,      lambda = optimalLambda)
##
##   Df  %Dev   Lambda
## 1 15 96.72 0.006596
```

Testing model

```
# Preparing data for inference using fitted glmnet
xTest <- testData[, !(names(testData) %in% c("Marker"))]
xTest <- as.matrix(xTest)
yTest <- testData$Marker
```

```r
# Making predictions and evaluating performance
predictions <- predict(trainedModel, newx=xTest)
resultsTest <- assess.glmnet(predictions, newy=yTest, family="binomial")

roc(yTest, predictions, plot=TRUE)
```




```
##
## Call:
## roc.default(response = yTest, predictor = predictions, plot = TRUE)
##
## Data: predictions in 17 controls (yTest 0) < 16 cases (yTest 1).
## Area under the curve: 1
```

```r
library(rmarkdown)
render("project10.Rmd", pdf_document(TRUE), "Indilewitsch_Toidze_Houhamdi_Pudziuvelyte_Project10.pdf")
```