ORF525, Assignment 2, Problem 5

Igor Silin

(a)

Let's load the data and do all necessary steps:

```
macro = read.csv("macro-transformed.csv", header=T)
macro = macro[14:719,]                                      # Jan 1960 to Oct 2018
macro[ , "sasdate"] = list(NULL)                            # remove the feature "sasdate"

features_na = colnames(macro)[colSums(is.na(macro)) > 0]    # features with missing entries
print(features_na)
```

```
## [1] "ACOGNO"   "ANDENOx"  "TWEXMMTH" "UMCSENTx" "VXOCLSx"
```

```
macro[ , features_na] = list(NULL)                          # remove features with missing entries
```

The printed above features have missing values.

(b)

```
n = dim(macro)[1]
p = dim(macro)[2] - 1

Y <- macro$UNRATE
macro[ , "UNRATE"] = list(NULL)

X = as.matrix(macro)
```

Let's standardize the data. We need to do that because one of the questions asks to find the most important variables. To be able to compare magnitudes of coefficients we need to standardize the predictors. Actually, the algorithms perform standardization themselves, but then they transform the obtained coefficients back to the original scale, so it is dishonest to compare them in this case. So, we have to standardize the predictors manually to be able to track the corresponding coefficients relative to each other.

```
meanX = rep(apply(X, 2 , mean))
stdX = rep(apply(X, 2, sd))

X = X - matrix(meanX, ncol=dim(X)[2], nrow=dim(X)[1], byrow=T)
X = X/matrix(stdX, ncol=dim(X)[2], nrow=dim(X)[1], byrow=T)
```

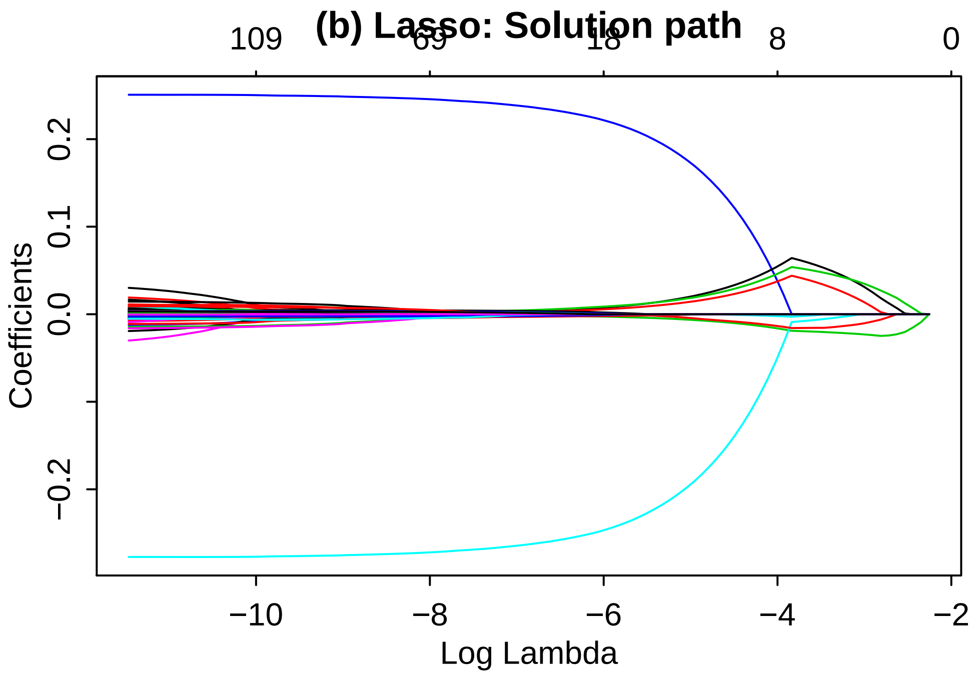
```
#install.packages("glmnet")
#install.packages("ncvreg")
set.seed(525)
library("glmnet")
```

```
## Loading required package: Matrix

## Loading required package: foreach

## Loaded glmnet 2.0-16
```

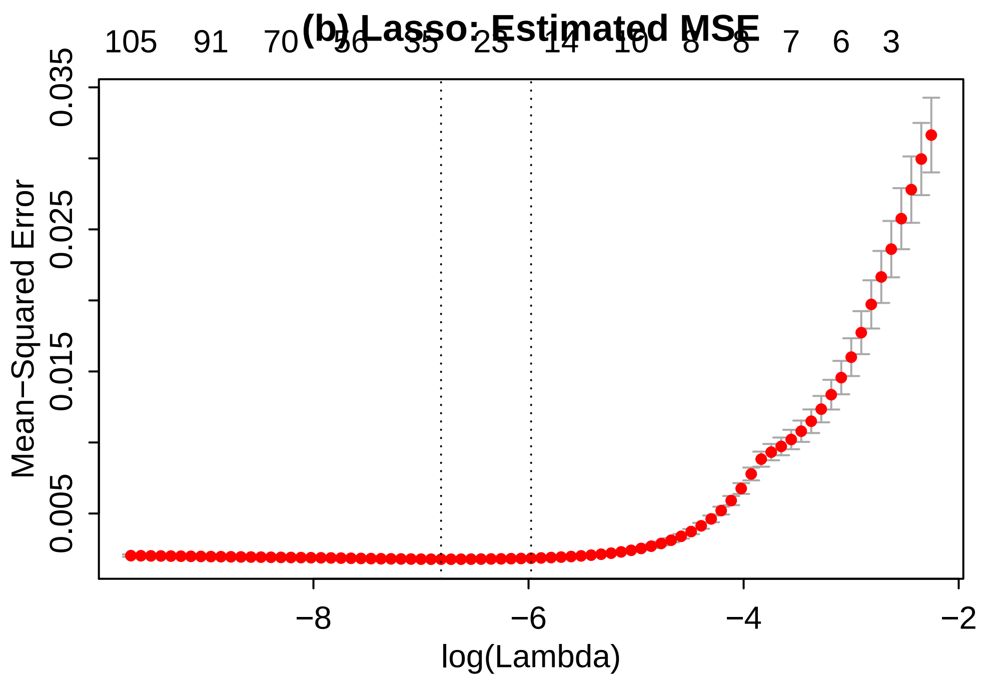
```r
library("ncvreg")
```

First, we consider Lasso. Let's plot the regularization paths as well as the mean squared errors estimated by 10-fold cross-validation:

```r
LASSO <- glmnet(X, Y, alpha = 1)
LASSOcv <- cv.glmnet(X, Y, alpha = 1, nfolds = 10)

par(mfrow = c(2,1), mex=0.5)
par(fig=c(0,10,2,10)/10)
par(mai=c(0.5,1.5,0.5,0.5))
plot(LASSO, xvar=c("lambda"));
title('(b) Lasso: Solution path', line=2);
```



```r
plot(LASSOcv)
title('(b) Lasso: Estimated MSE', line=2)
```



The best model is described by λ that gives the smallest mean-squared error.

```r
sprintf("(b) Lasso: Best lambda = %f", LASSOcv$lambda.min)
```

```
## [1] "(b) Lasso: Best lambda = 0.001099"
```

Now let's compute the in-sample R^2:

```r
Y.pred = predict(LASSOcv, newx=X, s="lambda.min")

R2 = 1 - sum((Y - Y.pred)^2)/sum((Y - mean(Y))^2)
sprintf("(b) Lasso: In-sample R^2 = %f", R2)
```

```
## [1] "(b) Lasso: In-sample R^2 = 0.949020"
```

As we see from the regularization path, there are two most influential variables. Let's find them: for λ, for which the difference in coefficients is significant, in particular, it works for the best fitted by the algorithm, we extract the coefficients, take its absolute values, sort them and output 5 largest:

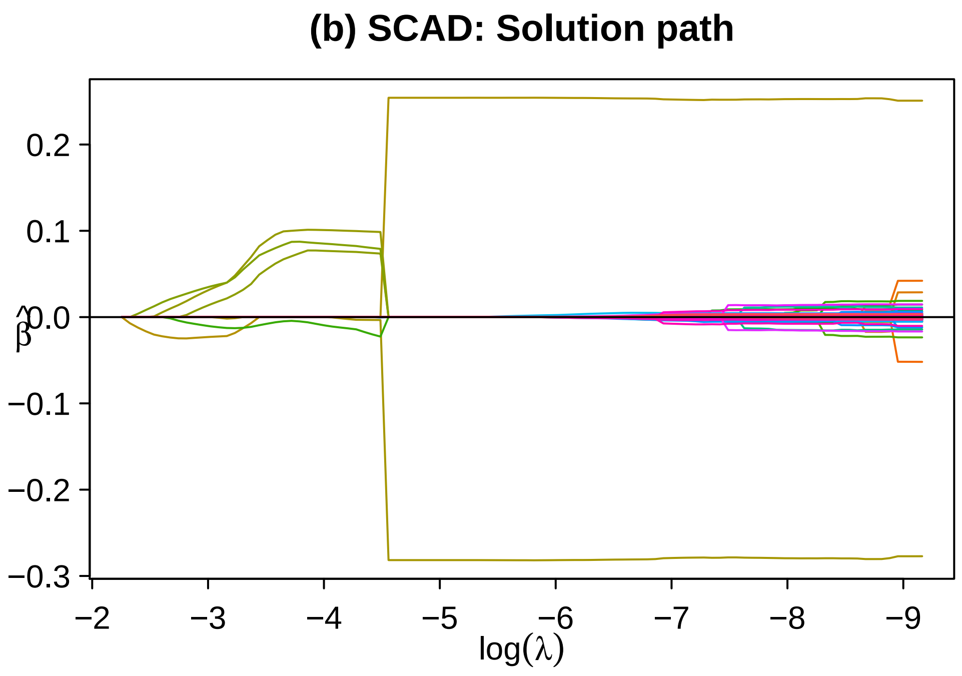
```r
print(sort((abs(coef(LASSOcv, s=LASSOcv$lambda.min)))[,1], decreasing=TRUE)[1:5])
```

```
##      CE160V      CLF160V     UEMP15OV      UEMPLT5     BUSLOANS
## 0.262281366 0.236350466 0.004765816 0.004034141 0.003581622
```

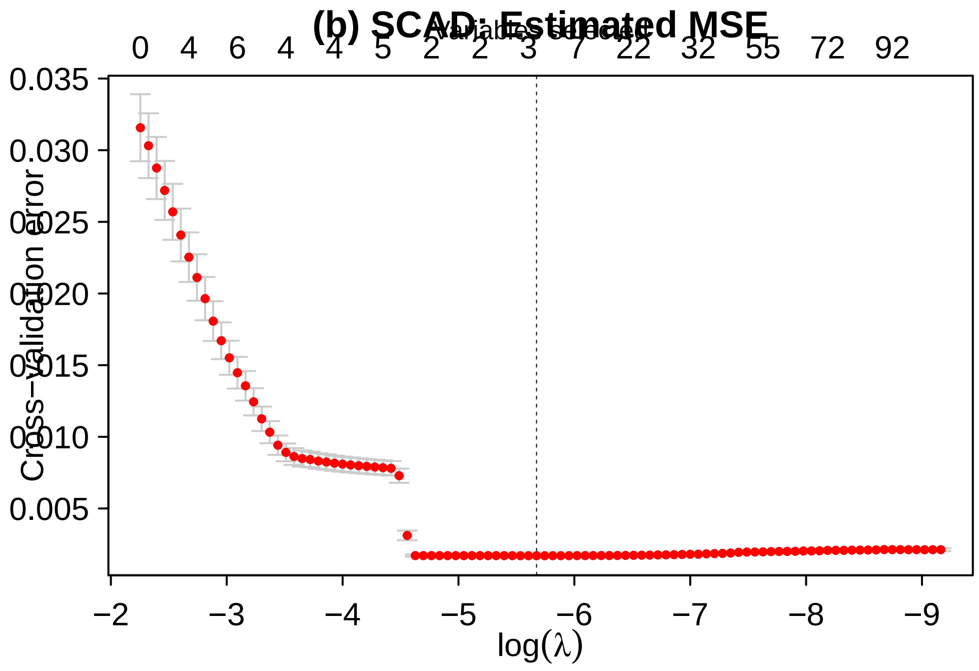
So, this confirms the regularization path: two coefficients, corresponding to "CE160V" and "CLF160V", has absolute values significantly larger than all the other.

Now we do the same for SCAD:

```r
SCAD = ncvreg(X, Y, penalty="SCAD")
SCADcv = cv.ncvreg(X, Y, penalty="SCAD", nfolds=10)

par(mfrow = c(2,1), mex=0.5)
par(fig=c(0,10,2,10)/10)
par(mai=c(0.5,1.5,0.5,0.5))
plot(SCAD, log.l=TRUE, shade=FALSE);
title('(b) SCAD: Solution path', line=2);
```



```
plot(SCADcv, log.l=TRUE)
title('(b) SCAD: Estimated MSE', line=2)
```



The best model is described by λ that gives the smallest mean-squared error.

```
sprintf("(b) SCAD: Best lambda = %f", SCADcv$lambda.min)
```

```
## [1] "(b) SCAD: Best lambda = 0.003435"
```

Now let's compute the in-sample R^2:

```
Y.pred = predict(SCAD, X=X, lambda=SCADcv$lambda.min)
```

```
R2 = 1 - sum((Y - Y.pred)^2)/sum((Y - mean(Y))^2)
sprintf("(b) SCAD: In-sample R^2 = %f", R2)
```

```
## [1] "(b) SCAD: In-sample R^2 = 0.946786"
```

As we see from the regularization path, there are two most influential variables. Let's find them: for small λ, in particular, for the last fitted by the algorithm, we extract the coefficients, take its absolute values, sort them and output 5 largest:

```
print(sort(abs(SCAD$beta[, 100]), decreasing=TRUE)[1:5])
```

```
##     CE16OV    CLF16OV     INDPRO    IPFPNSS      IPMAT
## 0.27716365 0.25081645 0.05186192 0.04209846 0.02871298
```

So, this confirms the regularization path: two coefficients, corresponding to "CE16OV" and "CLF16OV", has absolute values significantly larger than all the other.

(c)

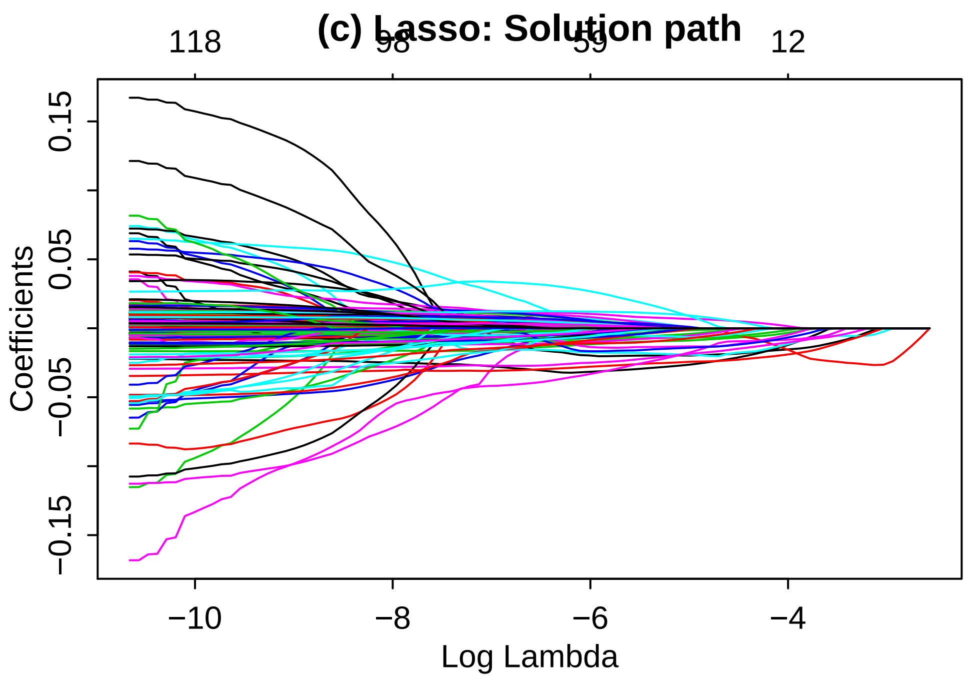
Let's shift the data for purpuses of this part:

```
X = X[1:(n-1), ]
Y = Y[2:n]
n = n-1
```

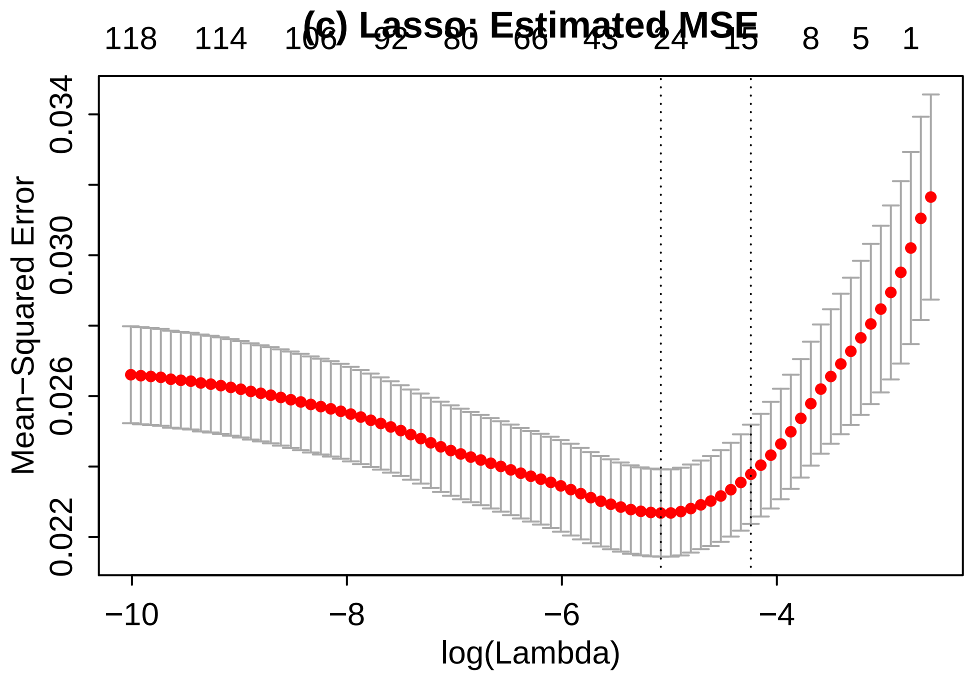
Then we perform absolutely the same analysis as in (b).

Let's plot the regularization paths as well as the mean squared errors estimated by 10-fold cross-validation:

```
LASSO <- glmnet(X, Y, alpha = 1) #Lasso fit for solution path
LASSOcv <- cv.glmnet(X, Y, alpha = 1, nfolds = 10)

par(mfrow = c(2,1), mex=0.5)
par(fig=c(0,10,2,10)/10)
par(mai=c(0.5,1.5,0.5,0.5))
plot(LASSO, xvar=c("lambda"));
title('(c) Lasso: Solution path', line=2);
```



```r
plot(LASSOcv)
title('(c) Lasso: Estimated MSE', line=2)
```



The best model is described by λ that gives the smallest mean-squared error.

```r
sprintf("(c) Lasso: Best lambda = %f", LASSOcv$lambda.min)
```

```
## [1] "(c) Lasso: Best lambda = 0.006224"
```

Now let's compute the in-sample R^2:

```r
Y.pred = predict(LASSOcv, newx=X, s="lambda.min")
```

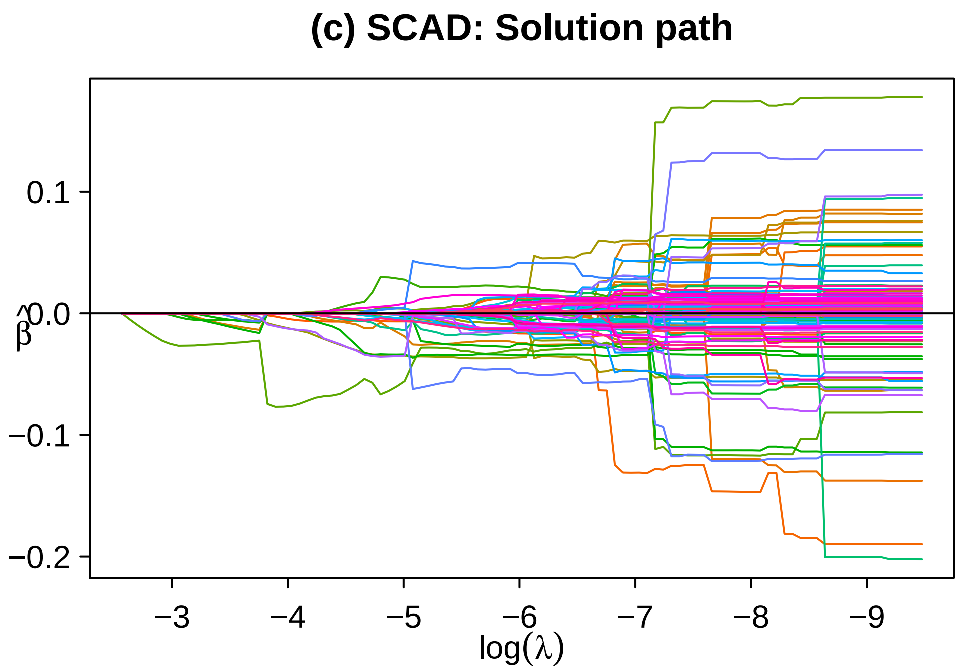
```r
R2 = 1 - sum((Y - Y.pred)^2)/sum((Y - mean(Y))^2)
sprintf("(c) Lasso: In-sample R^2 = %f", R2)
```

```
## [1] "(c) Lasso: In-sample R^2 = 0.345932"
```

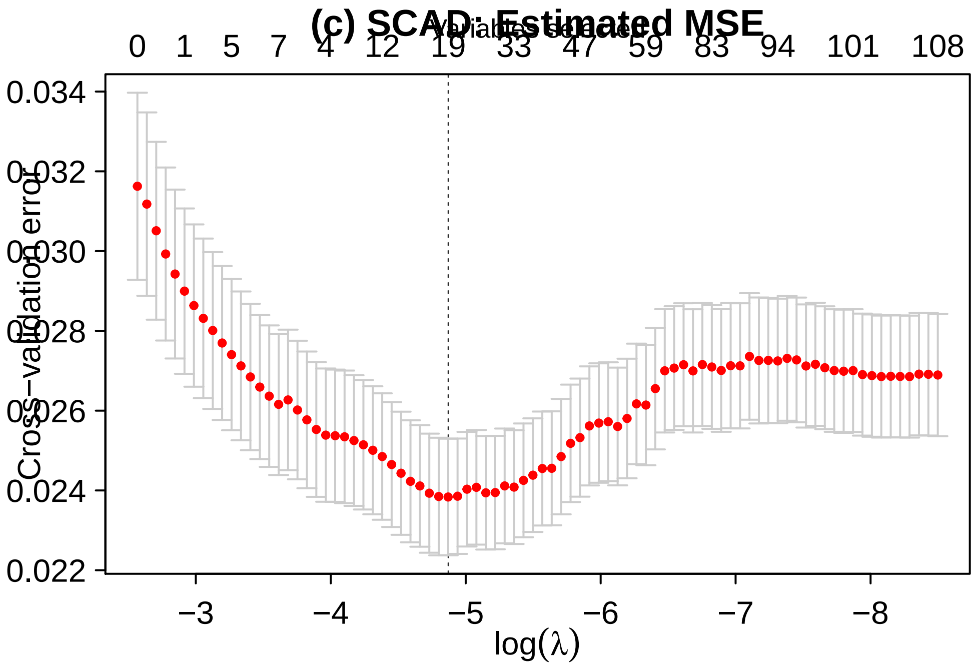
As we see from the regularization path, there is no just two most important variables; actually, a lot of them are of comparable importance.

Now we do the same for SCAD:

```r
SCAD = ncvreg(X, Y, penalty="SCAD", nfolds=10)
SCADcv = cv.ncvreg(X, Y, penalty="SCAD", nfolds=10)

par(mfrow = c(2,1), mex=0.5)
par(fig=c(0,10,2,10)/10)
par(mai=c(0.5,1.5,0.5,0.5))
plot(SCAD, log.l=TRUE, shade=FALSE);
title('(c) SCAD: Solution path', line=2);
```



```r
plot(SCADcv, log.l=TRUE)
title('(c) SCAD: Estimated MSE', line=2)
```



The best model is described by λ that gives the smallest mean-squared error.

```
sprintf("(c) SCAD: Best lambda = %f", SCADcv$lambda.min)
```

```
## [1] "(c) SCAD: Best lambda = 0.007674"
```

Now let's compute the in-sample R^2:

```
Y.pred = predict(SCAD, X=X, lambda=SCADcv$lambda.min)

R2 = 1 - sum((Y - Y.pred)^2)/sum((Y - mean(Y))^2)
sprintf("(c) SCAD: In-sample R^2 = %f", R2)
```

```
## [1] "(c) SCAD: In-sample R^2 = 0.329534"
```

As we see from the regularization path, there is no just two most important variables; actually, a lot of them are of comparable importance.