Assignment 2: Lasso and GLM

Víctor Villegas, Roger Llorenç, Luis Sierra

2024-02-27

Loading the necessary libraries and preparing the datasets. This time, we will be using a modified version of the Boston dataset, as well as a biological dataset from two files, concerned on survival time dependent on genes.

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-8
```

```
load("boston.Rdata")
express <- read.csv("journal.pbio.0020108.sd012.CSV",header=FALSE)
surv <- read.csv("journal.pbio.0020108.sd013.CSV",header=FALSE)

death <- (surv[,2]==1)
log.surv <- log(surv[death,1]+.05)
expr <- as.matrix(t(express[,death]))
```

1. Lasso for the Boston Housing data

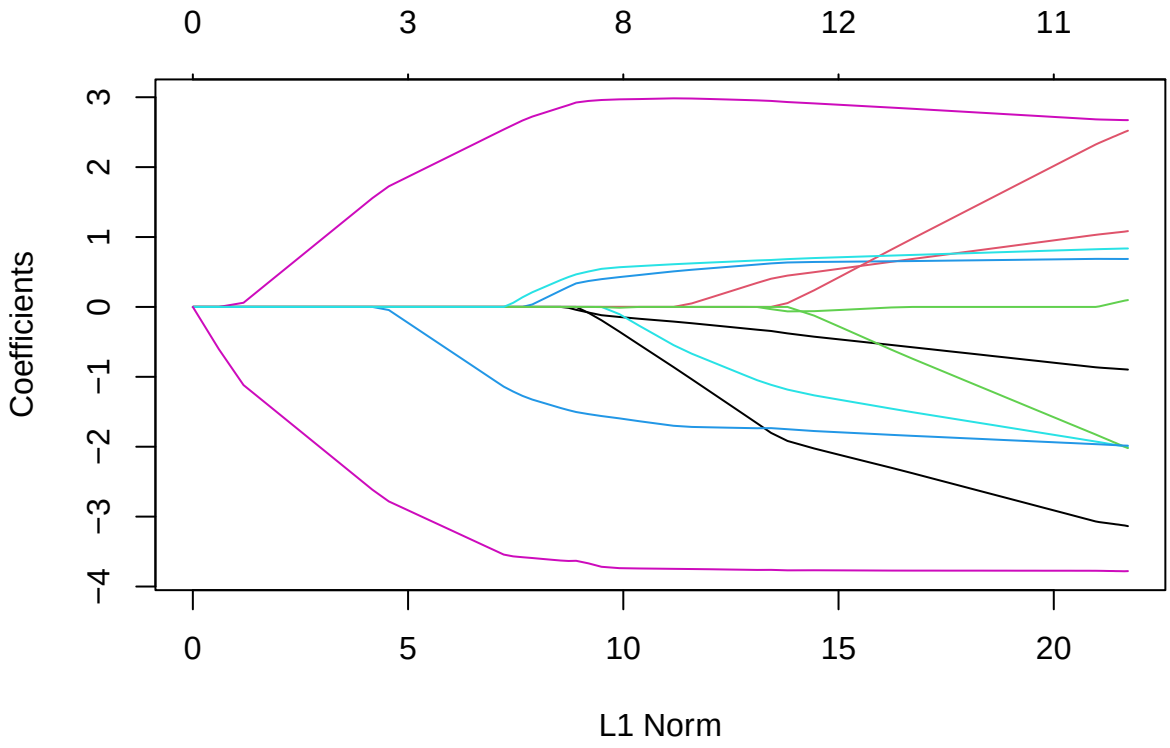
```
# anyNA(boston.c)
boston.c$CHAS <- as.numeric(boston.c$CHAS)
Y <- scale(boston.c$CMEDV, center=TRUE, scale=FALSE)
X <- scale(as.matrix(boston.c[,8:20]), center=TRUE, scale=TRUE) # Exclude response variable

p <- dim(X)[1]
n <- dim(X)[2]
```

After scaling the variables, we use *glmnet* on the Boston dataset to find the Lasso analysis. Note that the cross-validation fit has a probabilistic component, so we declare a seed.

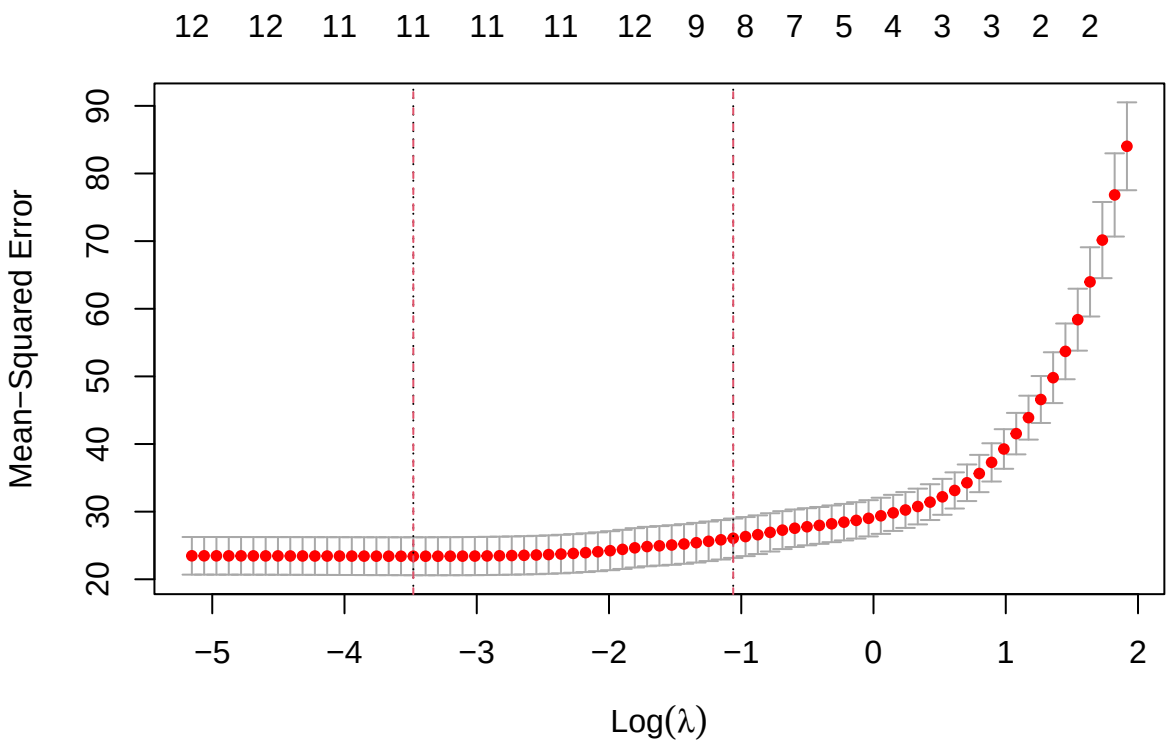
```
set.seed(1234) # To ensure replicability
fit_boston <- glmnet(X, Y)
cv_fit_boston <- cv.glmnet(X, Y)
```

In the following plot we show the path of the coefficients as we decrease the value of the shrinkage parameter λ.

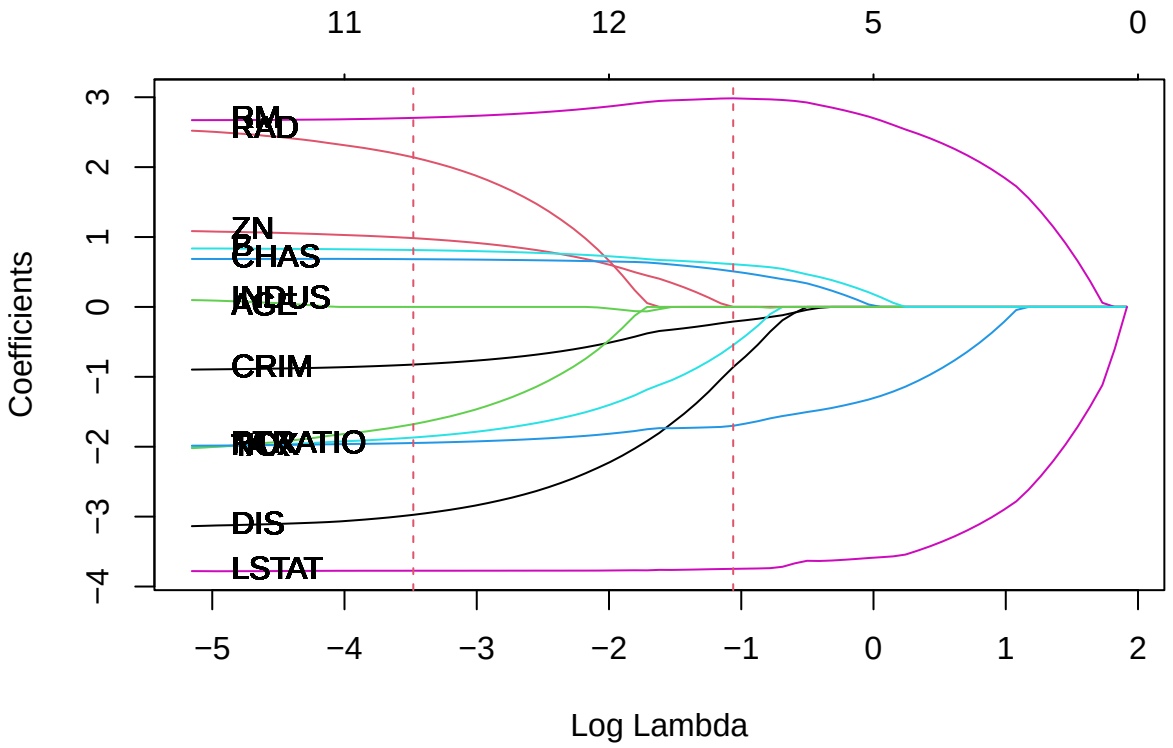
```
plot(fit_boston)
```




```
plot(cv_fit_boston)
abline(v=log(cv_fit_boston$lambda.min),col=2,lty=2)
abline(v=log(cv_fit_boston$lambda.1se),col=2,lty=2)
```




```
plot(fit_boston, xvar = "lambda")
abline(v = log(cv_fit_boston$lambda.min), col = 2, lty = 2)
abline(v = log(cv_fit_boston$lambda.1se), col = 2, lty = 2)
text(0*(1:p)+(-5),fit_boston$beta[ ,77],colnames(X),pos=4)
```



The minimum λ and the coefficients sorted by impact and their values are shown below:

```
cv_fit_boston$lambda.min
```

```
## [1] 0.03081795
```

```
coefs_min_boston <- coef(cv_fit_boston, s = "lambda.min")

print("The coefficients sorted by impact for lambda_{min} are: ")
```

```
## [1] "The coefficients sorted by impact for lambda_{min} are: "
```

```
coefs_min_boston_aux = coefs_min_boston[-1,]
coefs_min_boston_aux[order(abs(coefs_min_boston_aux))]
```

```
##      INDUS        AGE       CHAS          B       CRIM         ZN        TAX
##  0.0000000  0.0000000  0.6829374  0.8139427 -0.8243413  0.9840467 -1.6779859
##        NOX    PTRATIO        RAD         RM        DIS      LSTAT
## -1.8684319 -1.9470681  2.1404354  2.7043694 -2.9743762 -3.7739886
```

From these observations, *"AGE"* (proportion of owner-occupied units built prior to 1940) and *"INDUS"* (proportion of non-retail business acres per town) seem to be the least relevant predictors to determine the median value of a home.

The most impactful predictors seem to be *"LSTAT"* (% of lower status of the population), *"DIS"* (weighted distances to five Boston employment centers) and *"RM"* (average number of rooms per dwelling).

The 1-SE, or one standard error λ, and the coefficients sorted by impact and their value are shown below:

```
cv_fit_boston$lambda.1se
```

```
## [1] 0.3461855
```

```
coefs_reg_boston <- coef(cv_fit_boston, s = "lambda.1se")

print("The coefficients sorted by impact for lambda_{se} are: ")
```

```
## [1] "The coefficients sorted by impact for lambda_{se} are: "
```

```
coefs_reg_boston_aux = coefs_reg_boston[-1,]
coefs_reg_boston_aux[order(abs(coefs_reg_boston_aux))]
```

```
##         ZN      INDUS        AGE        RAD        TAX       CRIM       CHAS
##  0.0000000  0.0000000  0.0000000  0.0000000  0.0000000 -0.2094351  0.5085467
##        NOX          B        DIS    PTRATIO         RM      LSTAT
## -0.5463353  0.6108086 -0.8612478 -1.7004343  2.9839100 -3.7471328
```

From these observations, *"AGE"* and *"INDUS"* are still zero and *"RAD"* (index of accessibility to radial highways) and *"TAX"* (full-value property-tax rate per $10,000) join the zero-valued coefficients.

The most impactful predictors are still the same (*"LSTAT"*, *"DIS"* and *"RM"*) with the addition of *"PTRATIO"* (pupil-teacher ratio by town).

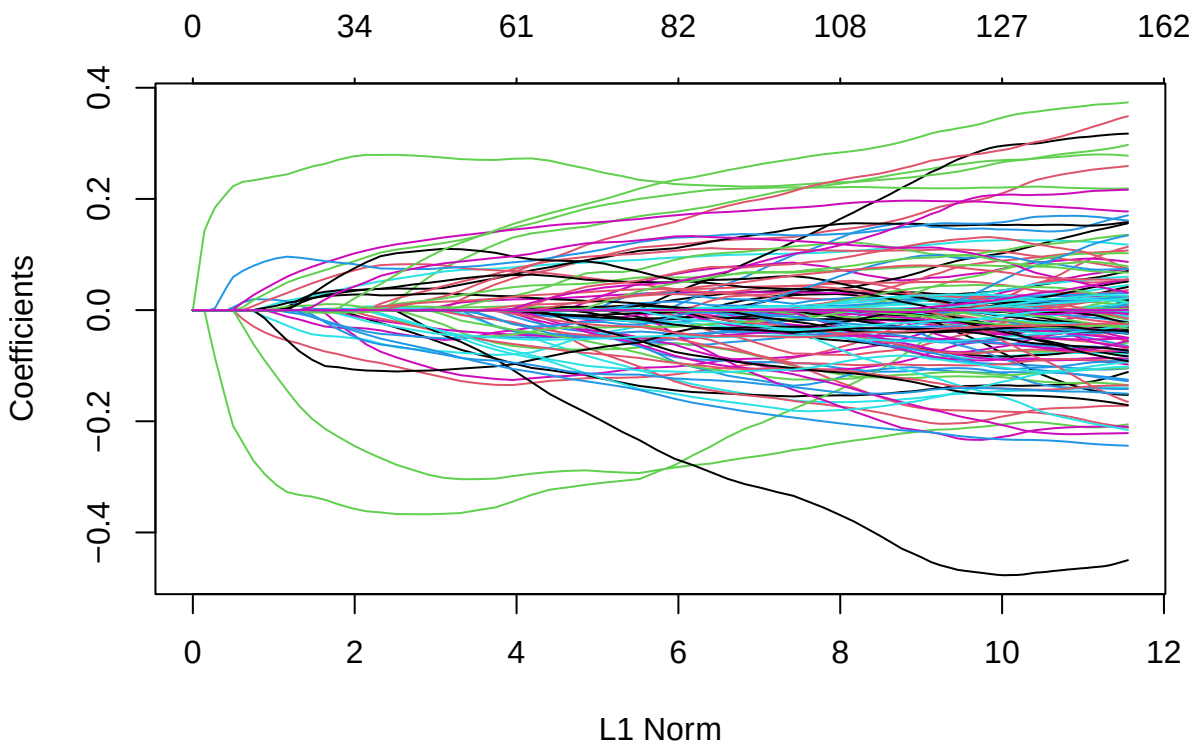
2. A regression model with $p >> n$:

Using *glmnet* we shall fit, using Lasso with different shrinkage parameters, the logarithm of the survival time against a large set of gene expressions.

2.1. In the following two figures we show the optimal value of λ

```
fit <- glmnet(expr, log.surv)
```

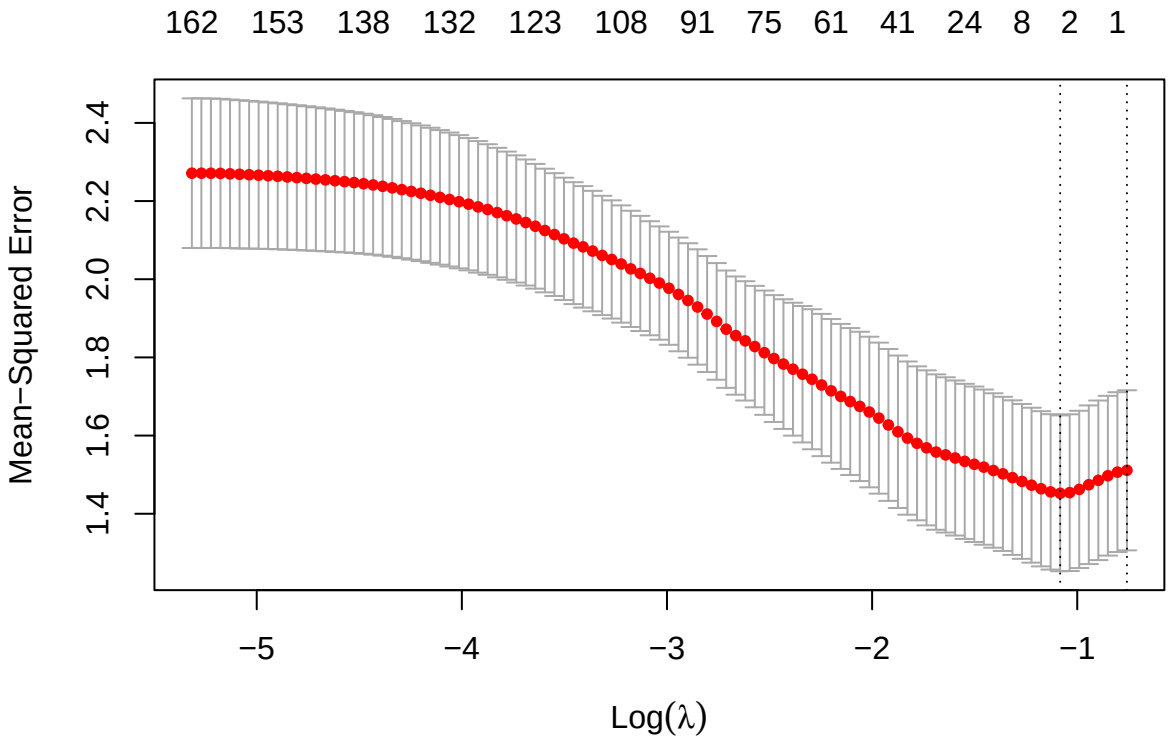
```
plot(fit)
```



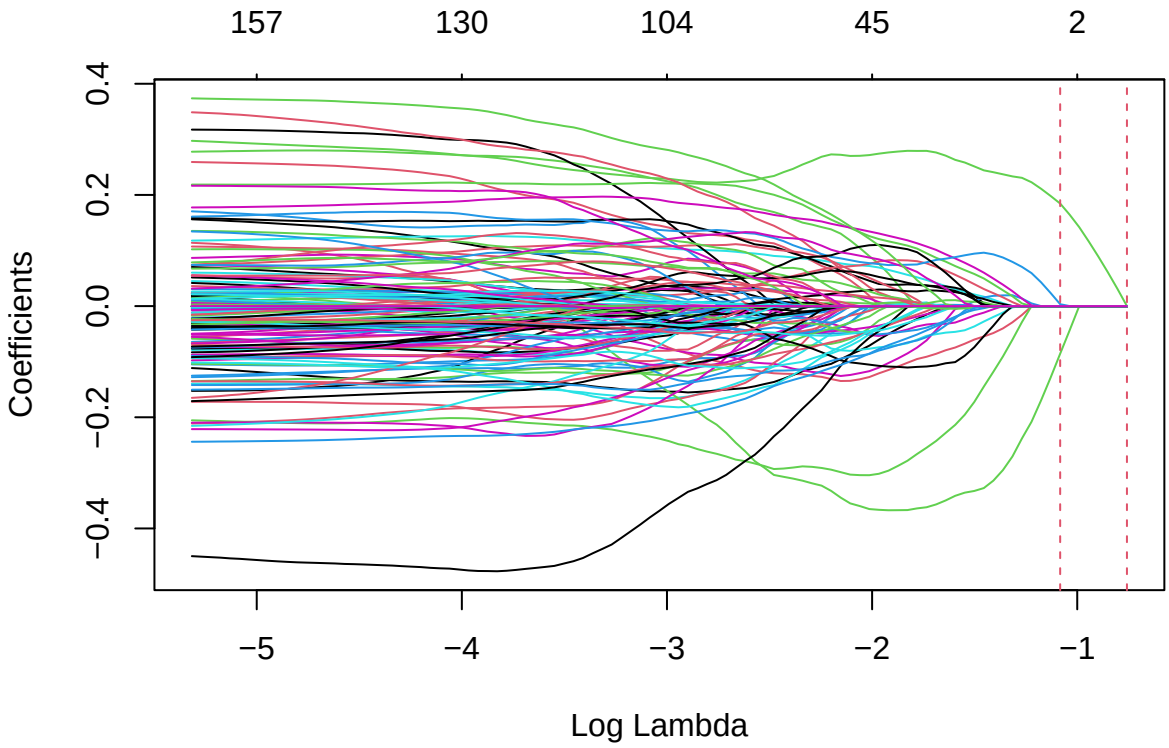
Selecting the best shrinkage parameter λ by cross validation, again defining a seed to ensure replicability.

```
set.seed(1234) # To ensure replicability
cvfit <- cv.glmnet(expr, log.surv)
```

```
plot(cvfit)
```




```
plot(fit, xvar = "lambda")
abline(v = log(cvfit$lambda.min), col = 2, lty = 2)
abline(v = log(cvfit$lambda.1se), col = 2, lty = 2)
```



Results for λ_{min}:

```
cvfit$lambda.min
```

```
## [1] 0.3384861
```

```
coefs_min <- coef(cvfit, s = "lambda.min")

print("The coefficients sorted by impact for lambda_{min} are: ")
```

```
## [1] "The coefficients sorted by impact for lambda_{min} are: "
```

```
coefs_min_aux = coefs_min[-1,];
sorted_coefs_min = coefs_min_aux[order(abs(coefs_min_aux))]
sorted_coefs_min[sorted_coefs_min != 0]
```

```
##        V5352        V2252        V3787
##  0.003404018 -0.085604533  0.184507554
```

Results for λ_{1se}:

```
cvfit$lambda.1se
```

```
## [1] 0.4687648
```

```
coefs_reg <- coef(cvfit, s = "lambda.1se")

print("The coefficients sorted by impact for lambda_{min} are: ")
```

```
## [1] "The coefficients sorted by impact for lambda_{min} are: "
```

```
coefs_reg_aux = coefs_reg[-1,];
sorted_coefs_reg = coefs_reg_aux[order(abs(coefs_reg_aux))]
sorted_coefs_reg[sorted_coefs_reg != 0]; print("That is, all coefficients are zero.")
```

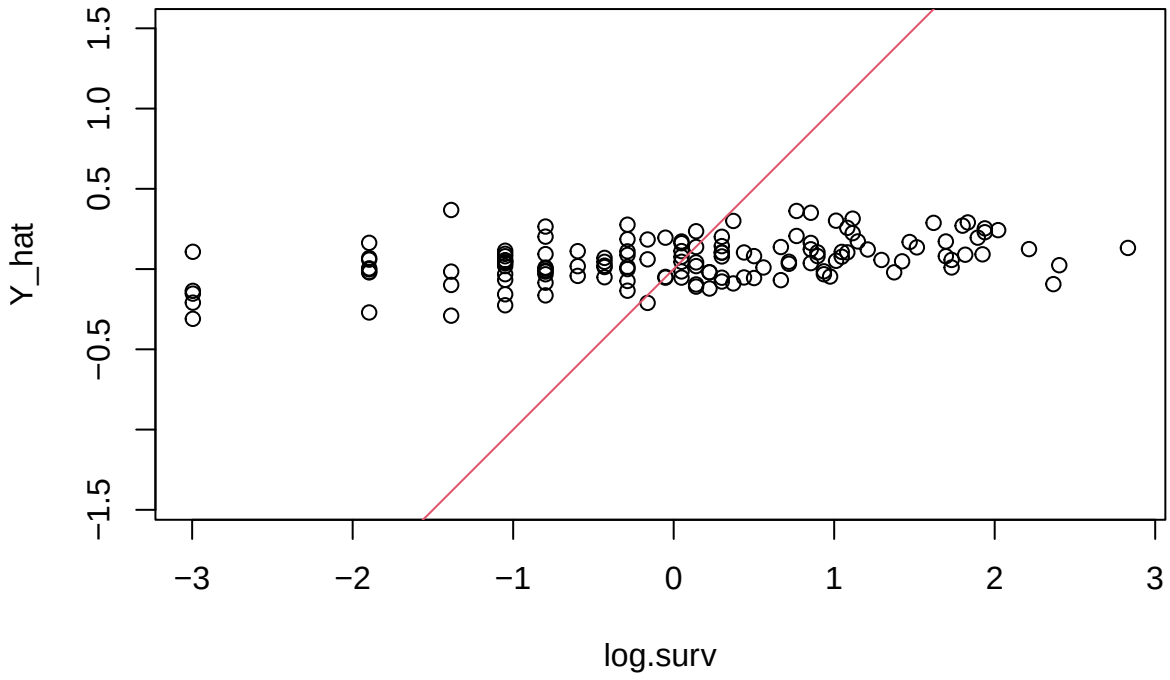
```
## named numeric(0)
```

```
## [1] "That is, all coefficients are zero."
```

Only three coefficients are different from zero for the minimum PMSE, and note that for the 1-SE values, the coefficients are all zero and we are left with the null model (always predict the mean average).

2.2. Computing fitted values with Lasso parameters.

```
Y_hat <- predict(cvfit, newx = expr, s = "lambda.min")
plot(log.surv, Y_hat, asp=1)
abline(a=0,b=1,col=2)
```



From this plot we can see how, despite there being some correlation, it is indeed rather weak.

2.3. OLS with non-zero coefficients set S_0 from Lasso fit

```
S_0 <- coefs_min@i # S_0[1] is intercept
ols_model <- lm(log.surv ~ expr[,S_0[2]] + expr[,S_0[3]] + expr[,S_0[4]])
coefs_ols <- ols_model$coefficients
coefs_ols
```

```
##    (Intercept) expr[, S_0[2]] expr[, S_0[3]] expr[, S_0[4]]
```

```
##      0.06620014     -1.03092012      0.48356293      0.44666762
```

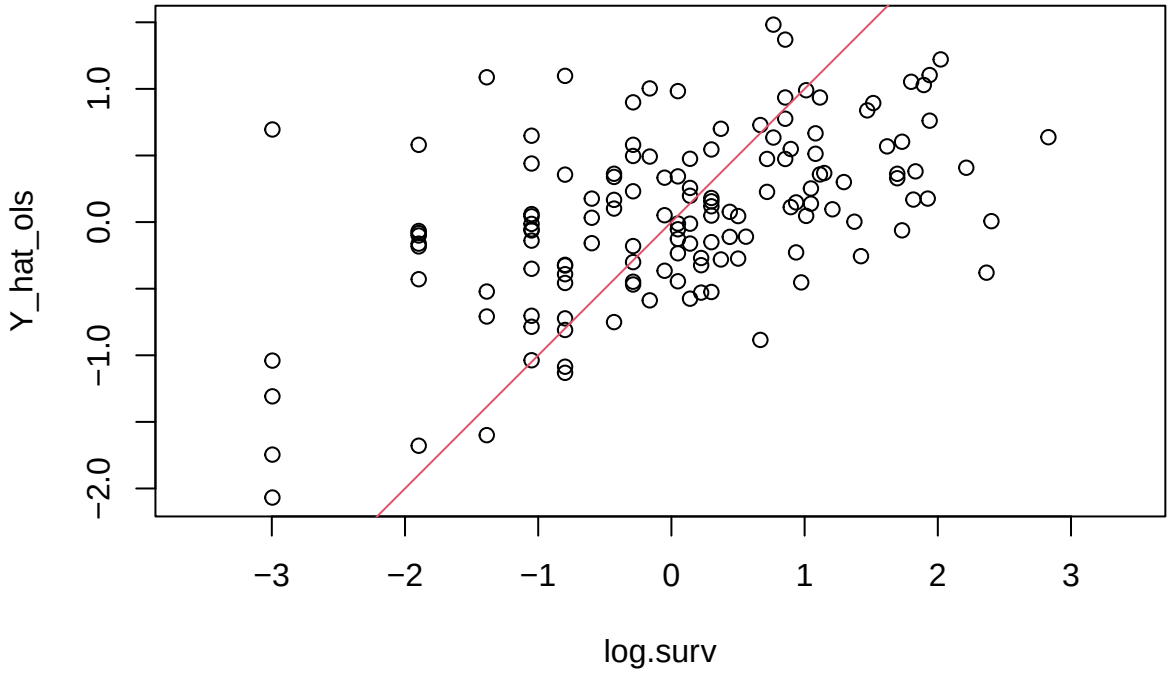
```
coefs_min@x
```

```
## [1]  0.055671743 -0.085604533  0.184507554  0.003404018
```

Only three coefficients were non-zero after the cross-validation, those corresponding to genes 2252, 3787 and 5352.

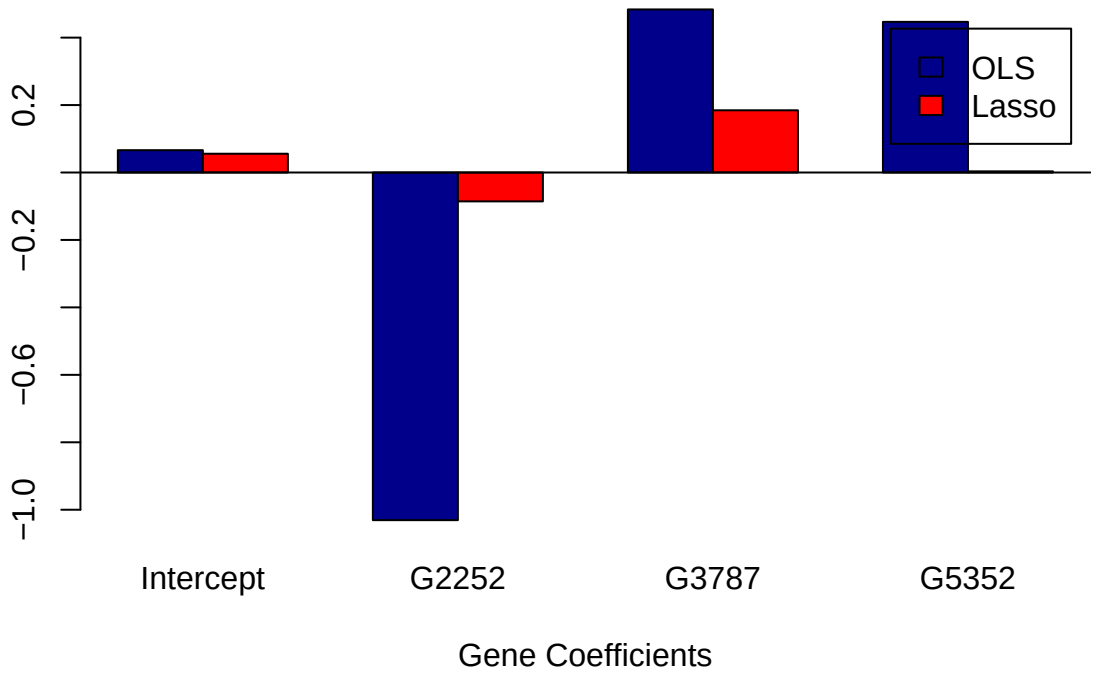
What happens now with the predictions? They are, optically, somewhat better to those produced by the Lasso model, and indeed we can see a tendency, albeit still a weak one, between the predictions and the sample data.

```
Y_hat_ols <- predict(ols_model, newx = expr)
plot(log.surv, Y_hat_ols, asp=1)
abline(a=0,b=1,col=2)
```



2.4. Comparing fitted Lasso and OLS coefficients:

```
coef_names <- c("Intercept", "G2252", "G3787", "G5352")
names(coefs_ols) <- coef_names
barplot(rbind(coefs_ols, coefs_min@x), xlab='Gene Coefficients', col=c("darkblue","red"), legend = c("O
abline(h=0)
```



We can see how the coefficients for the OLS model are larger than for Lasso, which is to be expected since there is no shrinkage involved. In fact the coefficient for the *5352* gene was almost 0 in the Lasso model (0.0034) but now in the OLS it has a statistically significant amount, with a value of 0.447. In the following chunck we can see a breakdown of the OLS model, where we can see how all predictive variables are statistically significant, but the adjusted R^2 is around 0.25, indicating that this model explains a low amount of the variance in the dataset.

```
summary(ols_model)
```

```
##
## Call:
## lm(formula = log.surv ~ expr[, S_0[2]] + expr[, S_0[3]] + expr[,
##     S_0[4]])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.6908 -0.6717  0.1249  0.7036  2.7453
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     0.06620    0.09023   0.734 0.464411
## expr[, S_0[2]] -1.03092    0.26005  -3.964 0.000119 ***
## expr[, S_0[3]]  0.48356    0.14159   3.415 0.000844 ***
## expr[, S_0[4]]  0.44667    0.15197   2.939 0.003877 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 1.058 on 134 degrees of freedom
## Multiple R-squared:   0.27,  Adjusted R-squared:  0.2536
## F-statistic: 16.52 on 3 and 134 DF,  p-value: 3.429e-09
```

In the following plot we show a comparison between the OLS fitted values and the Lasso fitted values, where we can see a nice tendency. We can also see how the Lasso predicts lower amounts for the points then the OLS, which is completely expected given the shrinkage of the coefficients involved.

```
plot(Y_hat_ols, Y_hat, asp=1)
abline(a=0,b=1,col=2)
```