High Diminsional Statistics-Sheet 3-Exercise 4

Hamed Vaheb

13/04/2022

The goal of an experimental research is to analyse the link between the value of the specific prostate antigen and some covariates in subjects undergoing prostatectomy surgery. The prostate antigen and some covariates in subjects undergoing prostatectomy surgery. The prostate dataset can be found at this link

Prepare

First of all we import the dataset:

```r
url <- "https://hastie.su.domains/ElemStatLearn/datasets/prostate.data"
df <- read.table(url, sep = '\t', header = TRUE)
df %>% head(10)
```

```
##     X     lcavol  lweight age       lbph svi       lcp gleason pgg45       lpsa
## 1   1 -0.5798185 2.769459  50 -1.3862944   0 -1.386294       6     0 -0.4307829
## 2   2 -0.9942523 3.319626  58 -1.3862944   0 -1.386294       6     0 -0.1625189
## 3   3 -0.5108256 2.691243  74 -1.3862944   0 -1.386294       7    20 -0.1625189
## 4   4 -1.2039728 3.282789  58 -1.3862944   0 -1.386294       6     0 -0.1625189
## 5   5  0.7514161 3.432373  62 -1.3862944   0 -1.386294       6     0  0.3715636
## 6   6 -1.0498221 3.228826  50 -1.3862944   0 -1.386294       6     0  0.7654678
## 7   7  0.7371641 3.473518  64  0.6151856   0 -1.386294       6     0  0.7654678
## 8   8  0.6931472 3.539509  58  1.5368672   0 -1.386294       6     0  0.8544153
## 9   9 -0.7765288 3.539509  47 -1.3862944   0 -1.386294       6     0  1.0473190
## 10 10  0.2231436 3.244544  63 -1.3862944   0 -1.386294       6     0  1.0473190
##    train
## 1   TRUE
## 2   TRUE
## 3   TRUE
## 4   TRUE
## 5   TRUE
## 6   TRUE
## 7  FALSE
## 8   TRUE
## 9  FALSE
## 10 FALSE
```

Question 1

Build the regression model for the variable prostate antigen (lpsa)

```
##
## Call:
## lm(formula = df$lpsa ~ ., data = features)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -1.76644 -0.35510 -0.00328  0.38087  1.55770
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.181561   1.320568   0.137  0.89096
## lcavol       0.564341   0.087833   6.425 6.55e-09 ***
## lweight      0.622020   0.200897   3.096  0.00263 **
## age         -0.021248   0.011084  -1.917  0.05848 .
## lbph         0.096713   0.057913   1.670  0.09848 .
## svi          0.761673   0.241176   3.158  0.00218 **
## lcp         -0.106051   0.089868  -1.180  0.24115
## gleason      0.049228   0.155341   0.317  0.75207
## pgg45        0.004458   0.004365   1.021  0.31000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6995 on 88 degrees of freedom
## Multiple R-squared:  0.6634, Adjusted R-squared:  0.6328
## F-statistic: 21.68 on 8 and 88 DF,  p-value: < 2.2e-16
```

The estimators of b_0 and $b_j \in \{1, ..., 8\}$ are respectively:

```
coef(model_lm)
```

```
##  (Intercept)       lcavol      lweight          age         lbph          svi
##  0.181560845  0.564341280  0.622019788 -0.021248185  0.096712522  0.761673402
##          lcp      gleason        pgg45
## -0.106050939  0.049227934  0.004457512
```

Question 2

Build the regression model with L1-constraint on the parameters. Estimate then the co- efficients and plot them.

```
#model_lasso <- l1ce(df$lpsa ~ . , data = features)
model_lasso <- glmnet(features, df$lpsa, alpha = 1)
model_lasso
```

Building the model

```
##
## Call:  glmnet(x = features, y = df$lpsa, alpha = 1)
##
##    Df  %Dev  Lambda
## 1   0  0.00 0.84340
## 2   1  9.16 0.76850
## 3   1 16.76 0.70020
## 4   1 23.07 0.63800
## 5   1 28.32 0.58130
## 6   1 32.67 0.52970
## 7   1 36.28 0.48260
## 8   1 39.28 0.43980
## 9   2 42.21 0.40070
## 10  2 44.90 0.36510
```

```
## 11  3 48.01 0.33270
## 12  3 50.66 0.30310
## 13  3 52.85 0.27620
## 14  3 54.68 0.25160
## 15  3 56.19 0.22930
## 16  3 57.45 0.20890
## 17  3 58.49 0.19040
## 18  3 59.36 0.17350
## 19  3 60.08 0.15800
## 20  3 60.67 0.14400
## 21  4 61.23 0.13120
## 22  5 61.75 0.11960
## 23  5 62.24 0.10890
## 24  5 62.64 0.09926
## 25  5 62.98 0.09044
## 26  5 63.25 0.08240
## 27  5 63.49 0.07508
## 28  5 63.68 0.06841
## 29  6 63.89 0.06234
## 30  6 64.21 0.05680
## 31  6 64.48 0.05175
## 32  6 64.70 0.04715
## 33  6 64.88 0.04297
## 34  6 65.03 0.03915
## 35  7 65.16 0.03567
## 36  7 65.27 0.03250
## 37  7 65.36 0.02961
## 38  7 65.44 0.02698
## 39  7 65.50 0.02459
## 40  7 65.55 0.02240
## 41  8 65.67 0.02041
## 42  8 65.78 0.01860
## 43  8 65.88 0.01695
## 44  8 65.95 0.01544
## 45  8 66.02 0.01407
## 46  8 66.07 0.01282
## 47  8 66.12 0.01168
## 48  8 66.16 0.01064
## 49  8 66.19 0.00970
## 50  8 66.21 0.00884
## 51  8 66.23 0.00805
## 52  8 66.25 0.00734
## 53  8 66.27 0.00668
## 54  8 66.28 0.00609
## 55  8 66.29 0.00555
## 56  8 66.30 0.00506
## 57  8 66.30 0.00461
## 58  8 66.31 0.00420
## 59  8 66.32 0.00382
## 60  8 66.32 0.00348
## 61  8 66.32 0.00317
## 62  8 66.33 0.00289
## 63  8 66.33 0.00264
## 64  8 66.33 0.00240
```

```
## 65  8 66.33 0.00219
## 66  8 66.33 0.00199
## 67  8 66.33 0.00182
## 68  8 66.33 0.00166
## 69  8 66.34 0.00151
## 70  8 66.34 0.00138
```

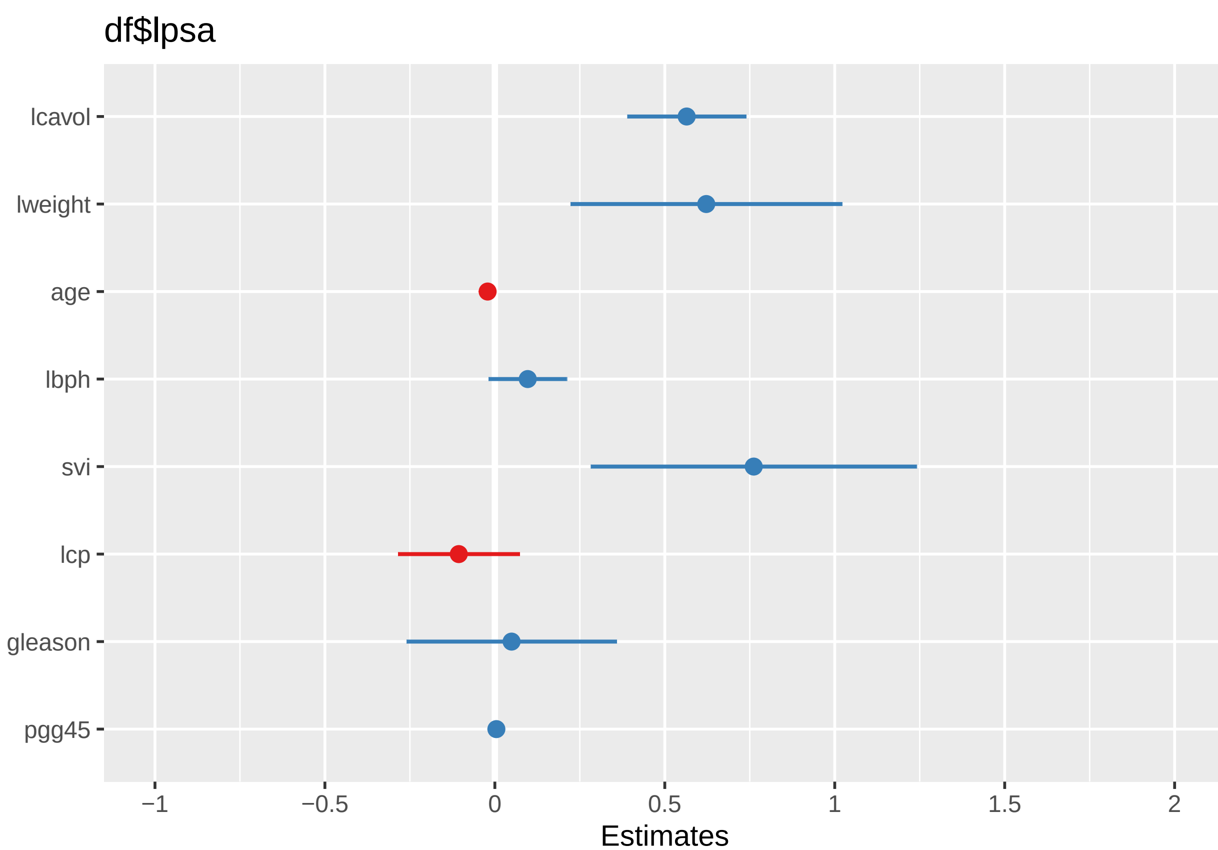
```
plot_model(model_lm)
```

Plotting the estimations of the coefficients for both linear model and lasso model

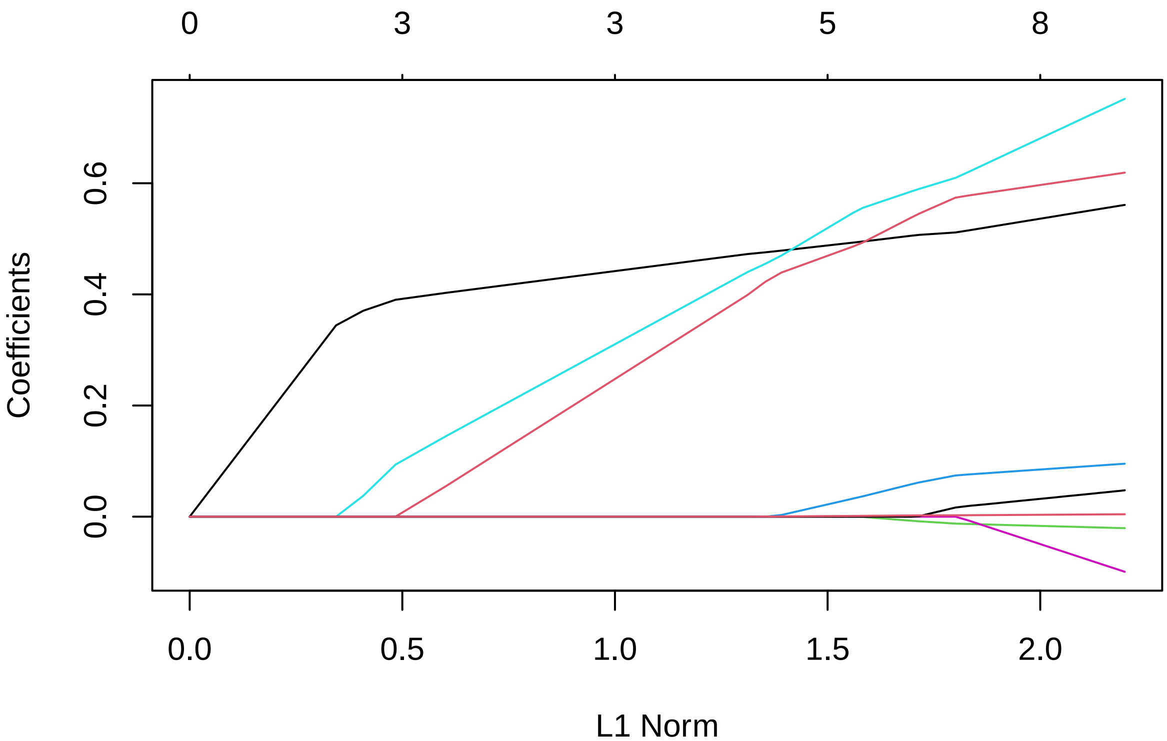
```
## Warning: Using `$` in model formulas can produce unexpected results. Specify your model
##   using the `data` argument instead.
##   Try: lpsa ~ lcavol + lweight + age + lbph +
##   svi + lcp + gleason + pgg45, data =

## Warning: Using `$` in model formulas can produce unexpected results. Specify your model
##   using the `data` argument instead.
##   Try: lpsa ~ lcavol + lweight + age + lbph +
##   svi + lcp + gleason + pgg45, data =

## Warning: Using `$` in model formulas can produce unexpected results. Specify your model
##   using the `data` argument instead.
##   Try: lpsa ~ lcavol + lweight + age + lbph +
##   svi + lcp + gleason + pgg45, data =
```



```
plot(model_lasso)
```



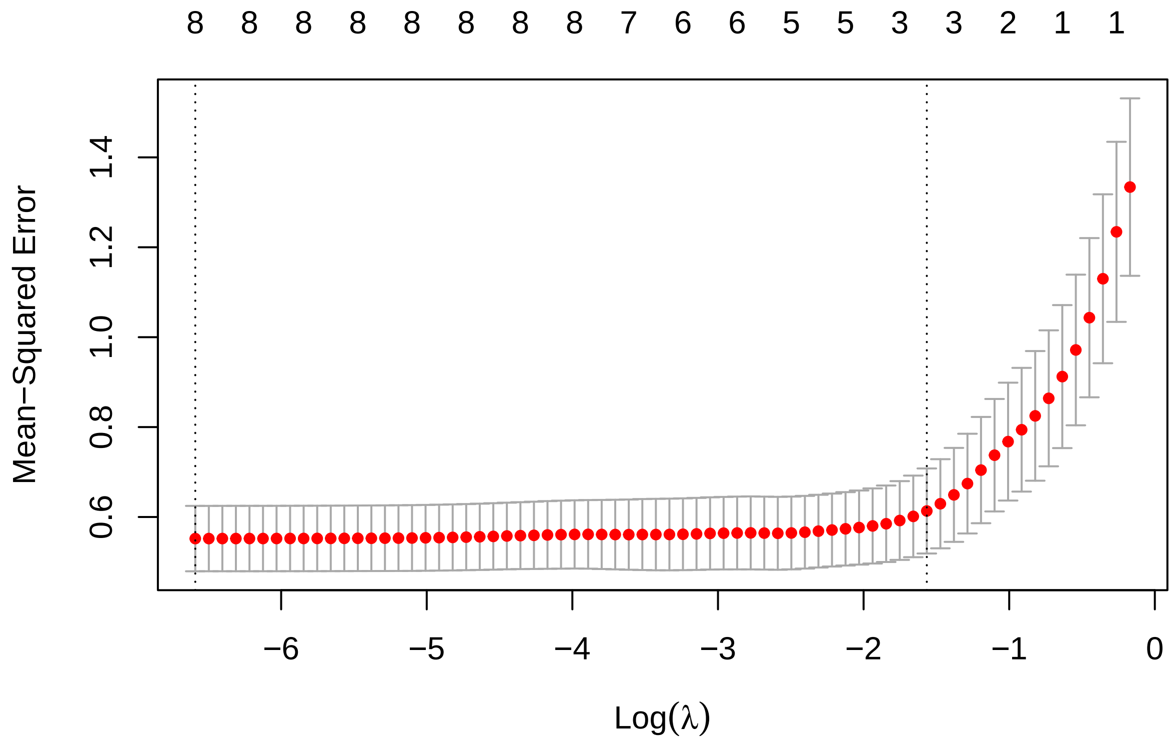
For $j \in 1, ..., 8$ ith curve corresponds to jth variable. It shows the path of b_j against the ℓ_1-norm of the whole coefficient vector b as λ varies. The axis above indicates the number of nonzero coefficients at the current λ.

Question 3:

Report two values for λ : "lambda.min" and "lambda.1se", where "lambda.min" is the λ at which the smallest mean squared error (MSE) is achieved and "lambda.1se" is the largest λ at which the MSE is within one standard error of the smallest MSE (default). Report the number of nonzero coefficients for the selected values of λ and the corresponding estimated coefficients.

Perform k-fold cross-validation to find optimal lambda value

```
X <- data.matrix(features)
y <- df$lpsa
cv_lasso <- cv.glmnet(X, y, alpha = 1)
plot(cv_lasso)
```




```
lambda_min = cv_lasso$lambda.min
lambda_1se = cv_lasso$lambda.1se
print(paste( "lambda.min = ",lambda_min))
```

```
## [1] "lambda.min =  0.00137457657144743"
```

```
print(paste( "lambda.1se = ",lambda_1se))
```

```
## [1] "lambda.1se =  0.208923416531039"
```

```
#lasso_model_min <- glmnet(features, y, alpha  = 1,lambda = #lambda_min)
#obain number of non-zero coefficients
#lasso_model_min$beta
#lasso_model_se <- glmnet(features, y=y, alpha = 1, lambda #=lambda_1se)
#obain number of non-zero coefficients
#lasso_model_se$beta
#predict(lasso_model_min,type="coef")
```

```
#coef.exact <- coef(model_lasso, s = c(lambda_min, lambda_1se), exact = TRUE)
#predict(model_lasso, newx = X, s = c(lambda_min, lambda_1se))
#coef.apprx <- coef(model_lasso, s = c(lambda_min, lambda_1se), exact = FALSE, x=X, y=y)
#coef.apprx[which(coef.apprx != 0)]
#coef.exact[which(coef.exact != 0)]

coeffs <- predict(model_lasso, s = c(lambda_min, lambda_1se), type="coef")

coeffs_s1 = coeffs[,1]
coeffs_s2 = coeffs[,2]


n1 <- coeffs_s1[which(coeffs_s1 != 0)] %>% length()
n2 <- coeffs_s2[which(coeffs_s2 != 0)] %>% length()
```

```
print(paste( "Number of non-zero coefficients for model with  lambdal.min = ",n1))
```

```
## [1] "Number of non-zero coefficients for model with  lambdal.min =  9"
```

```
print(paste( "Number of non-zero coefficients for model with  lambda.1se = ",n2))
```

```
## [1] "Number of non-zero coefficients for model with  lambda.1se =  4"
```