Take Home Final Exam

STAT 32950

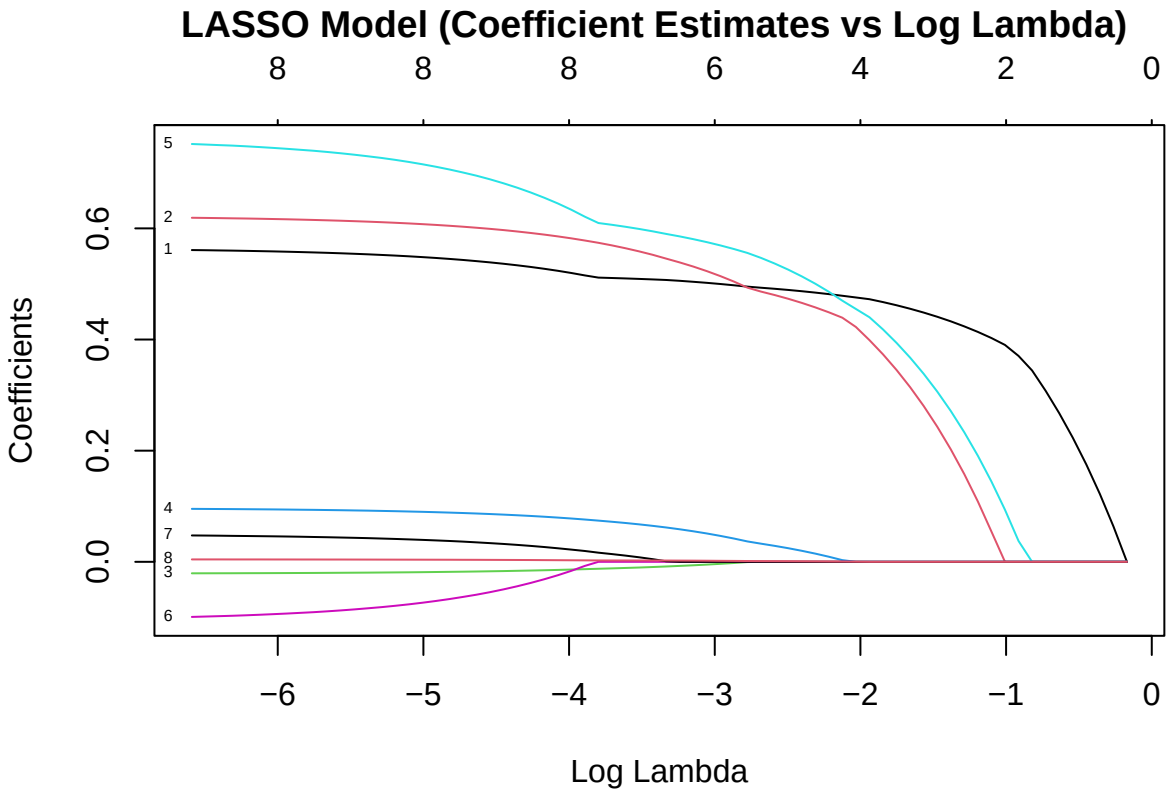
Ki Hyun

Due: 13:00 (CT) 2023-05-24

Problem 1.

(a)

Fitting the LASSO regression model on the prostate cancer data with `log-psa` as the response variable using the `glmnet` function in `R` yields the plot below for coefficients of the eight independent variables.



We may also use cross-validation function (`cv.glmnet`) in `R` to find the optimal λ values.

The λ value which minimizes test error is approximately 0.002. Moreover, the largest λ with error within one standard error of the minimum is approximately 0.19.

The fitted model with estimated parameters when λ is set as λ_{1se} is shown below, where ϵ_i is the error term for observation i.

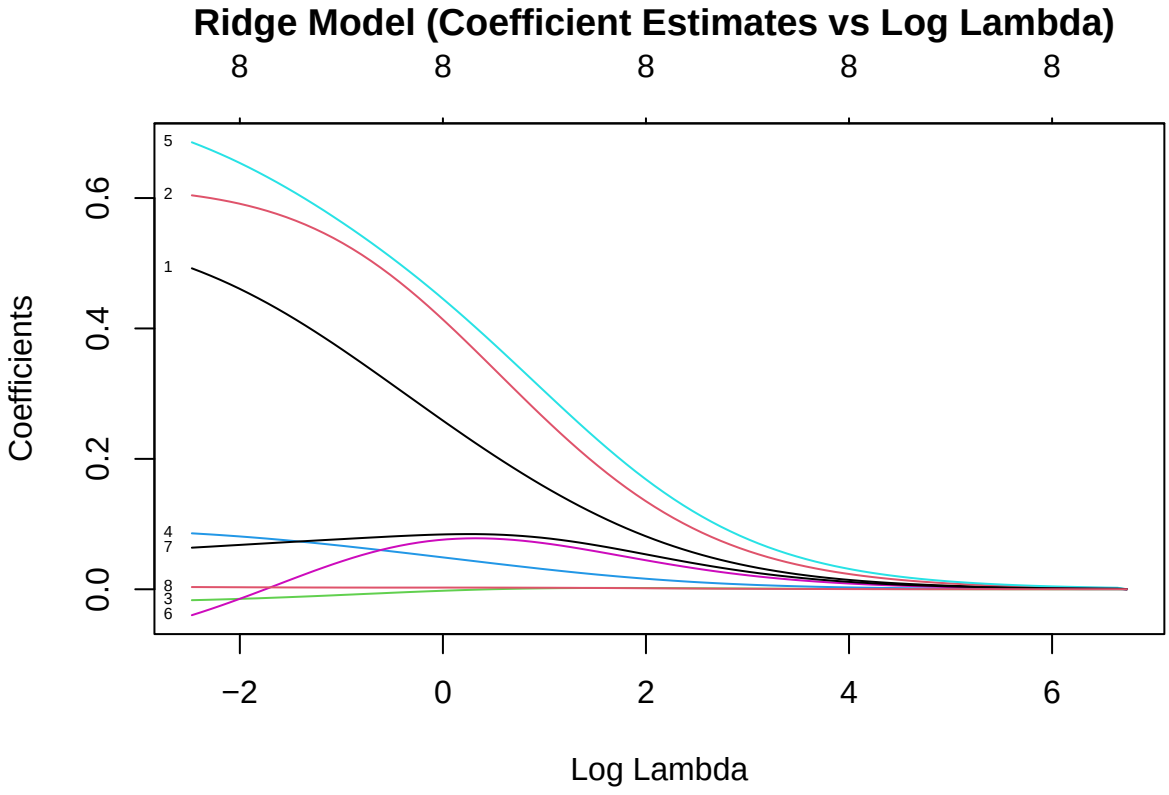
$$lpsa_i = 0.6435469 + (0.4553889)lcavol_i + (0.3142829)lweight_i + (0.367427)svi_i + \epsilon_i$$

(Answer: Running the code twice would yield different results since the `cv.glmnet` uses a random method for cross-validation. Without a random seed present, it would yield different results every time the code runs.)

(b)

Similar analysis could be done on Ridge regression by setting `alpha` to 0 in the `glmnet` function.

The fitted Ridge regression model on the prostate cancer data with `log-psa` as the response variable yields the plot below for coefficients of the eight independent variables.



We may also use cross-validation function (`cv.glmnet`) in `R` to find the optimal λ values.

The λ value which minimizes test error is approximately 0.084. Moreover, the largest λ with error within one standard error of the minimum is approximately 0.863.

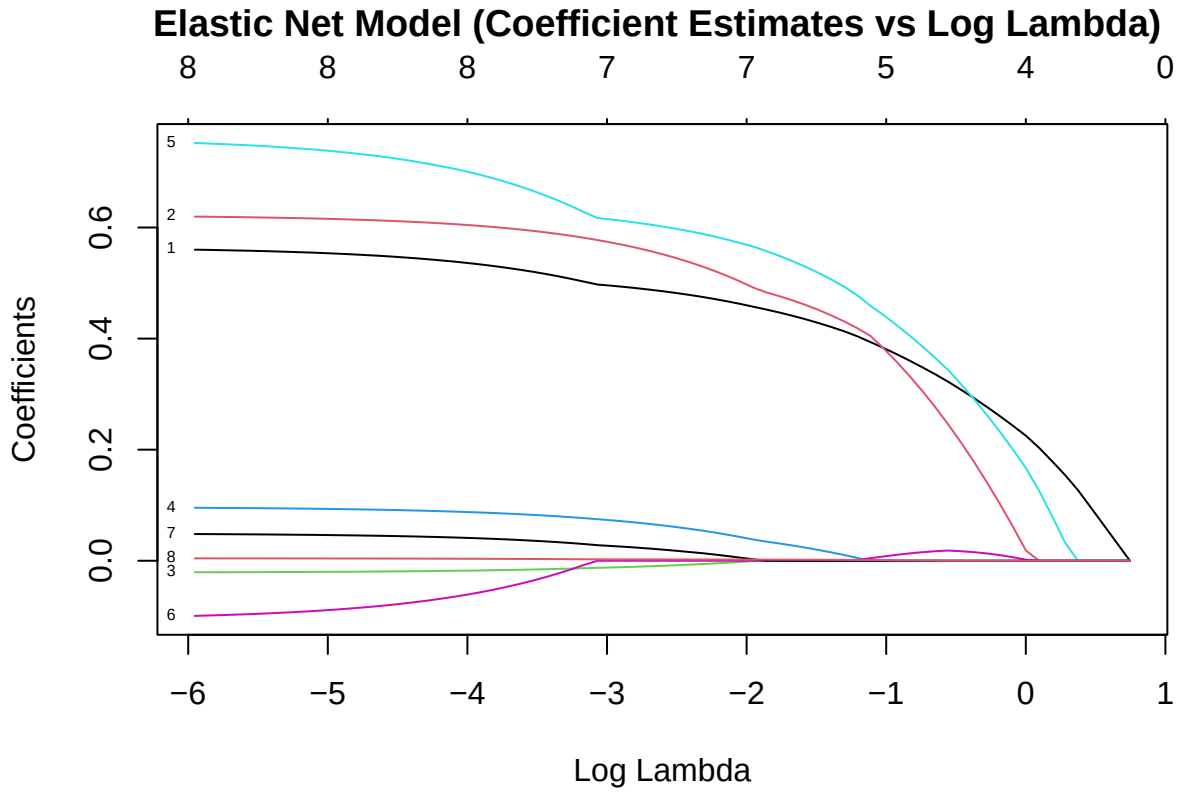
The fitted model with estimated parameters when λ is set as λ_{1se} is shown below, where ϵ_i is the error term for observation i.

$$\begin{aligned} lpsa_i =& 0.0092417 + (0.2748682)lcavol_i + (0.4344449)lweight_i + (-0.0031012)age_i + \\ & (0.0516365)lbph_i + (0.4645032)svi_i + (0.0737319)lcp_i + (0.0834232)gleason_i + \\ & (0.0026132)pgg45_i + \epsilon_i \end{aligned}$$

(c)

Similar analysis could be done on Elastic Net regression by setting `alpha` to 0.4 in the `glmnet` function.

The fitted Elastic Net regression model on the prostate cancer data with `log-psa` as the response variable yields the plot below for coefficients of the eight independent variables.



We may also use cross-validation function (`cv.glmnet`) in `R` to find the optimal λ values.

The λ value which minimizes test error is approximately 0.006. Moreover, the largest λ with error within one standard error of the minimum is approximately 0.328.

The fitted model with estimated parameters when λ is set as λ_{1se} is shown below, where ϵ_i is the error term for observation i.

$$\begin{aligned} lpsa_i =& 0.3445204 + (0.3939412)lcavol_i + (0.4052253)lweight_i + (0.4594009)svi_i + \\ & (0.0045756)lcp_i + (0.001348)pgg45_i + \epsilon_i \end{aligned}$$

(d)

i)

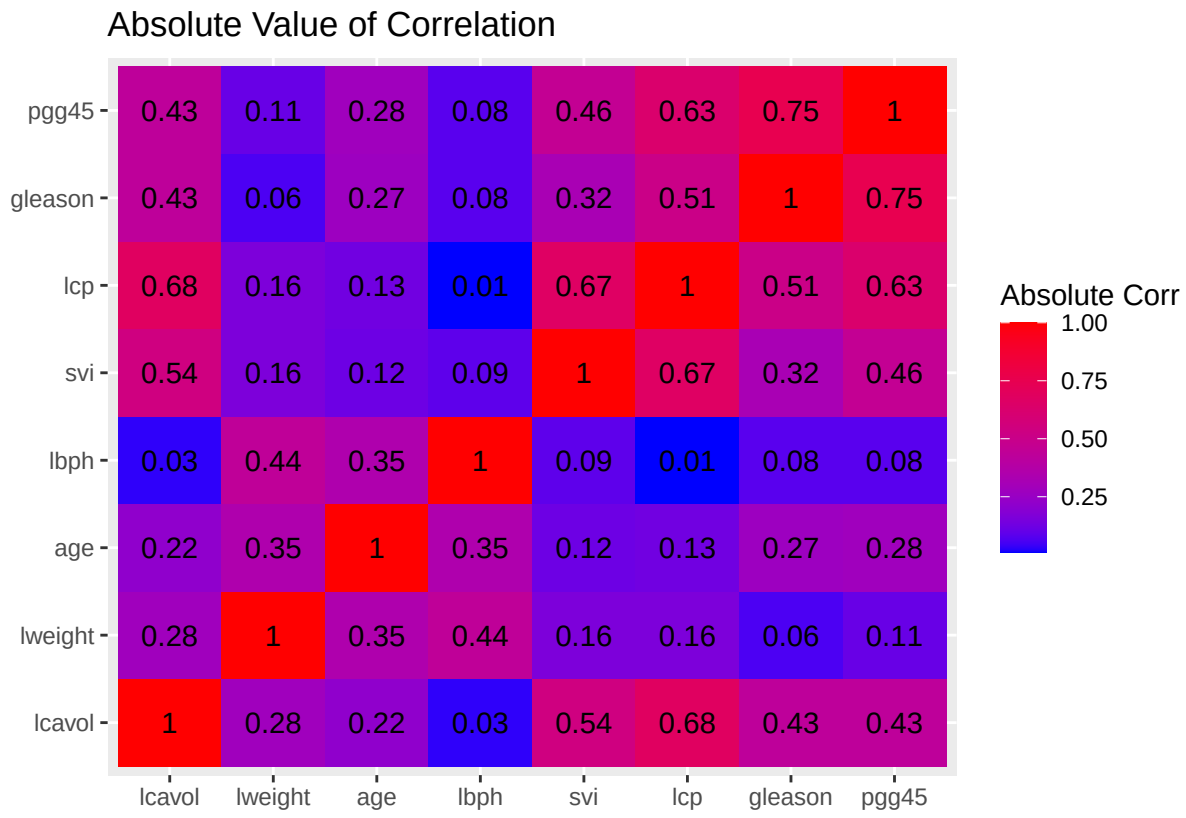
The Ridge regression results in no variable selection. However, the coefficients of each explanatory variable are smaller than the Least-square coefficients as the Ridge regression is constrained under l_2 penalty.

The optimized model under LASSO and Elastic Net methods do undergo variable selection with the involvement of l_1 penalty.

The LASSO regression selected 3 variables: lcavol, lweight, and svi.

The Elastic Net regression (with alpha set as 0.4) selected 5 variables: lcavol, lweight, svi, lcp, and pgg45.

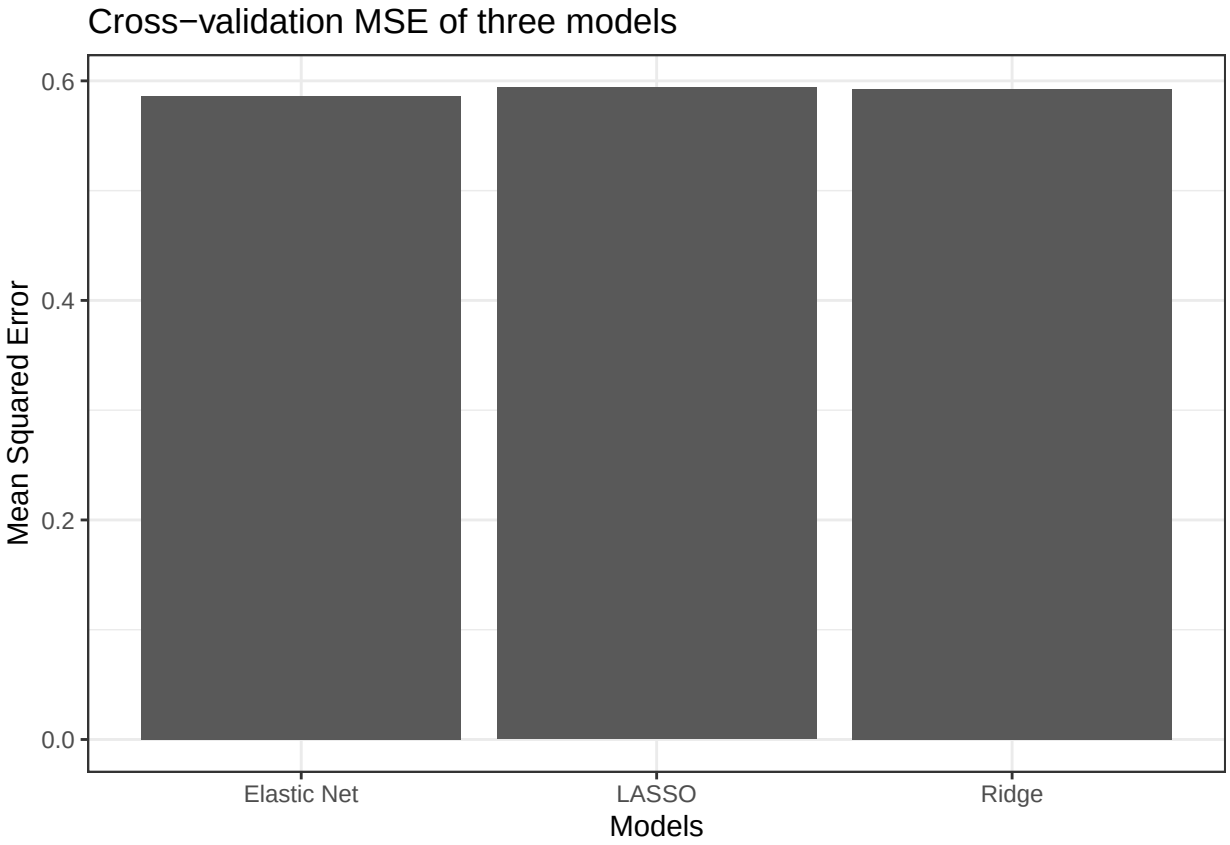
We may plot the correlation heat map of the explanatory variables to see what these variable selection means in terms of correlation among the variables.



From the correlation heat-map, it appears that explanatory variables with high correlation were not selected.

ii)

The MSE evaluated by the cross-validations for the three models in (a), (b), and (c) are represented below in the bar graph



For the optimal models, LASSO has the lowest MSE followed by Ridge and then Elastic Net method.

Appendix (Problem 1 Code):

```
# packages for Question 1
library(MASS)
library(glmnet)
library(reshape2)
library(ggplot2)
library(dplyr)

# reading the data
Prostate = read.table("pcancer.dat", header = T)
# response variable "log-psa"
Y = as.numeric(Prostate[,9])
# regressors from the data set
X = as.matrix(Prostate[, 1:8])

# setting the seed
set.seed(5881)

# (a)

# Fitting the LASSO using the "glmnet" function
LASSO_fit = glmnet(X, Y, alpha = 1)
par(mfrow = c(1, 1))
plot(LASSO_fit, label = TRUE, xvar = "lambda")
title(main = "LASSO Model (Coefficient Estimates vs Log Lambda) \n")

LASSO_cvfit = cv.glmnet(X, Y, alpha = 1)

LASSO_betas <- coef(LASSO_cvfit, s = "lambda.1se")
LASSO_regressors <- rownames(LASSO_betas)[LASSO_betas[,1]^2 > 0]
LASSO_coefs <- as.numeric(LASSO_betas[,1][LASSO_betas[,1]^2 > 0])

# (b)

# Fitting the Ridge using the "glmnet" function
Ridge_fit = glmnet(X, Y, alpha = 0)
par(mfrow = c(1, 1))
plot(Ridge_fit, label = TRUE, xvar = "lambda")
title(main = "Ridge Model (Coefficient Estimates vs Log Lambda) \n")

Ridge_cvfit = cv.glmnet(X, Y, alpha = 0)

Ridge_betas <- coef(Ridge_cvfit, s = "lambda.1se")
Ridge_regressors <- rownames(Ridge_betas)[Ridge_betas[,1]^2 > 0]
Ridge_coefs <- as.numeric(Ridge_betas[,1][Ridge_betas[,1]^2 > 0])

# (c)

# Fitting the Elastic Net using the "glmnet" function
EN_fit = glmnet(X, Y, alpha = 0.4)
par(mfrow = c(1, 1))
plot(EN_fit, label = TRUE, xvar = "lambda")
```

```r
title(main = "Elastic Net Model (Coefficient Estimates vs Log Lambda) \n")

EN_cvfit = cv.glmnet(X, Y, alpha = 0.4)

EN_betas <- coef(EN_cvfit, s = "lambda.1se")
EN_regressors <- rownames(EN_betas)[EN_betas[,1]^2 > 0]
EN_coefs <- as.numeric(EN_betas[,1][EN_betas[,1]^2 > 0])

# (d)

# correlation heat-map plot
ggplot(data = melt(abs(round(cor(X), 2))),
       aes(x=Var1, y=Var2, fill = value)) +
  geom_tile() +
  scale_fill_gradient(low = "blue", high = "red") +
  geom_text(aes(Var2, Var1, label = value),
            color = "black", size = 4) +
  labs(title = "Absolute Value of Correlation",
       x = "", y = "", fill = "Absolute Corr")

tibble(MSE = c(LASSO_cvfit$cvm[LASSO_cvfit$lambda == LASSO_cvfit$lambda.1se],
               Ridge_cvfit$cvm[Ridge_cvfit$lambda == Ridge_cvfit$lambda.1se],
               EN_cvfit$cvm[EN_cvfit$lambda == EN_cvfit$lambda.1se]),
       models = c("LASSO", "Ridge", "Elastic Net")) %>%
  ggplot(aes(x = models, y = MSE)) +
  geom_col() +
  labs(title = "Cross-validation MSE of three models",
       x = "Models", y = "Mean Squared Error") +
  theme_bw()
```

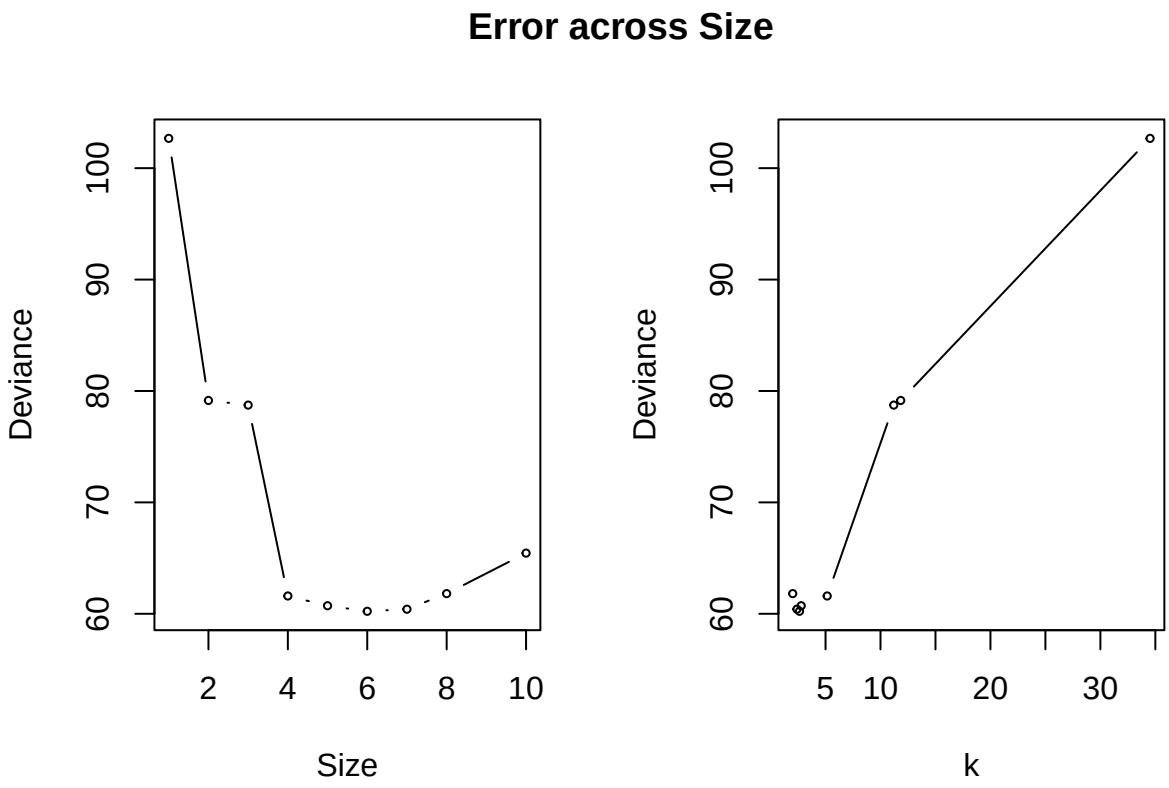
Problem 2.

(a)

In order to optimize the regression tree model over the training data, we will need to select an appropriate tree size.

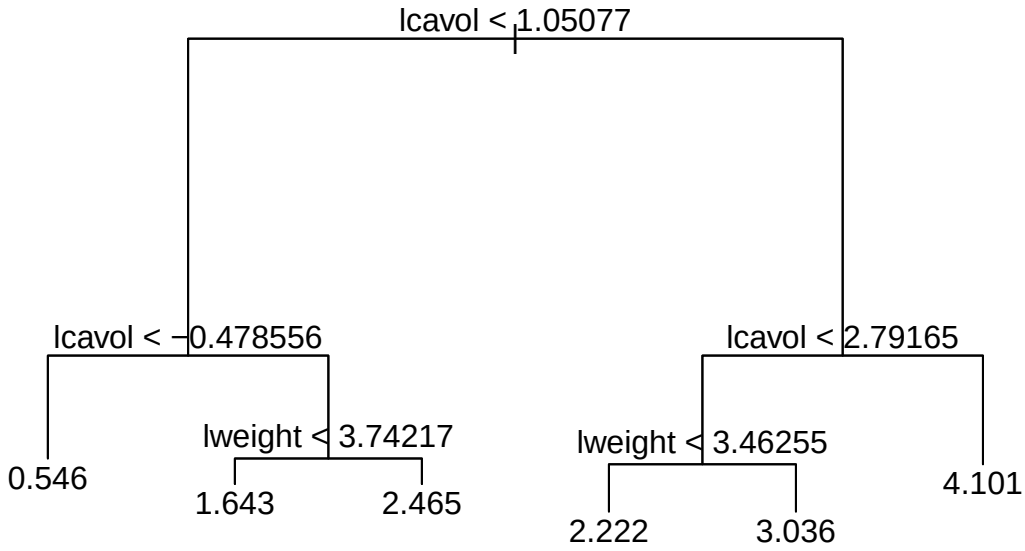
Instead of creating a separate validation set for optimization, I will conduct cross-validation using `cv.tree` function in `R` over the training data. Moreover, the deviance, instead of misclassification, will be used as the decision rule since we are dealing with a regression tree model.

The plot below shows how the deviance error changes across different sizes of regression trees.



Based on the cross-validation results, it appears that a regression tree of size 6 would be the best. A pruning method using `prune.tree` method in `R` can be conducted to fit the optimal regression tree model.

The final tree model can be plotted as below.



(b)

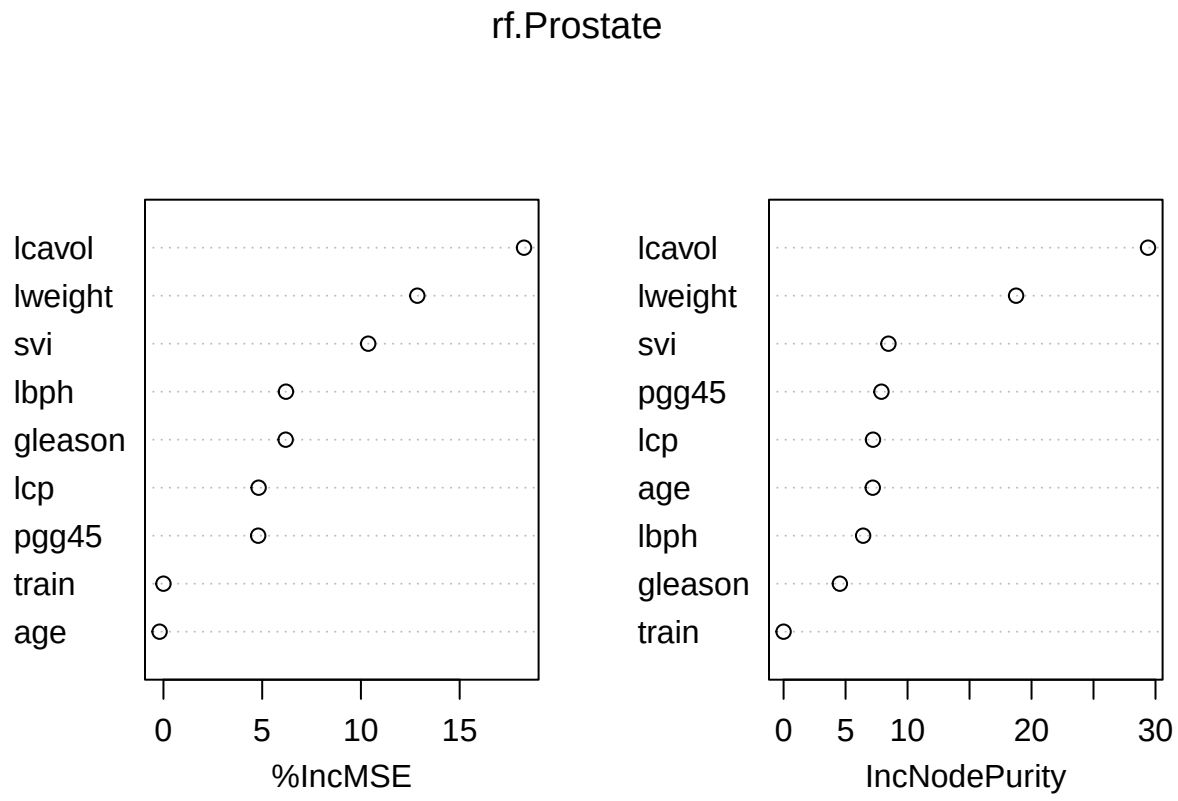
Random Forest model can also be fitted to the data. We first need to choose an appropriate number of predictors at each node (m). For a Random Forest method m of $\sqrt{p}/2$, $\sqrt{p}$, or $2\sqrt{p}$ are suggested.Since our data involves 9 explanatory variables, $m = 3$ will be chosen.

Moreover, the number of trees to be considered at each node will be set as the default value of 500.

The Out-of-Bag observations from the Random Forest may be used to evaluate the variable importance as below:

```
##            %IncMSE IncNodePurity
## lcavol  18.2586848     29.397046
## lweight 12.8558420     18.755693
## age     -0.1962015      7.194619
## lbph     6.2033099      6.418149
## svi     10.3685808      8.459087
## lcp      4.8185829      7.213307
## gleason  6.1893274      4.539435
## pgg45    4.7970874      7.892949
## train    0.0000000      0.000000
```

The plot for variable importance is shown as below.



(c)

We may compare the results of a single tree and a forest based on their performance against the test set. We would have to compare the **mean Residual Squared Sum** to decide which model is better in predicting an out-of-sample observation (i.e., the comparison is not regarding how well each model fits the in-sample data).

The mean RSS for a regression tree on the test data is approximately 0.322

The mean RSS for a Random Forest model on the test data is approximately 0.278

Based on the above test error statistics, the **Random Forest model** seems to perform better.

Appendix (Problem 2 Code):

```
# packages for Question 2
library(MASS)
library(tree)
library(randomForest)

# setting the seed
set.seed(5881)

# randomly sampled one-third to test
test = sample(1:nrow(Prostate), size = nrow(Prostate)/3)
# the remaining two-thirds to train
train = 1:nrow(Prostate)[-test]

# (a)

# creating the initial tree
tree.Prostate = tree(lpsa ~ ., data = Prostate, subset = train)
# cross-validation on the regression tree
cv.Prostate = cv.tree(tree.Prostate, method = "deviance")

# plotting deviance across different tree sizes
par(mfrow = c(1, 2))
plot(cv.Prostate$size, cv.Prostate$dev, type = "b", cex = .5,
     xlab = "Size", ylab = "Deviance")
plot(cv.Prostate$k, cv.Prostate$dev, type = "b", cex = .5,
     xlab = "k", ylab = "Deviance")
title(main = "Error across Size", outer = TRUE, line = -2)

# best tree based on cross-validation on the training set
prune.Prostate = prune.tree(tree.Prostate, best = 6)

# plotting the regression tree
plot(prune.Prostate); text(prune.Prostate);

# (b)

# random forest model
rf.Prostate = randomForest(lpsa ~ ., data = Prostate,
                           subset = train, mtry = 3, importance = TRUE)

# variable importance table
importance(rf.Prostate)

# variable importance plot
varImpPlot(rf.Prostate)

# (c)

# test response variable data
Prostate.test = Prostate[test, 9]
```

```r
# test error (mean RSS) for the regression tree
yhat_tree = predict(tree.Prostate, newdata = Prostate[test,])
RSS_tree = (Prostate.test - yhat_tree)^2

# test error (mean RSS) for the Random Forest Model
yhat_rf = predict(rf.Prostate, newdata = Prostate[test,])
RSS_rf = (Prostate.test - yhat_rf)^2
```

Problem 3.

I have spent approximately 2 hours on the exam