Christian Caburian
220133475

STAT330: Assignment 2

Question 1.

We will now derive the probability that a given observation is part of a bootstrap sample. Suppose that we obtain a bootstrap sample from a set of n observations.

(a) What is the probability that the first bootstrap observation is *not* the jth observation from the original sample? Justify your answer.

The probability that the first bootstrap observation is the jth observation is 1/n. That is, out of n possible outcomes, only 1 of these outcomes will result in the jth observation being first. Therefore, out of n possible outcomes, all other independent outcomes apart from that one outcome, (n - 1), correspond to the event that the first bootstrap observation is not the jth observation. These other outcomes are independent and can therefore be summed to n - 1 out of all n outcomes, resulting in:

$(n - 1) / n = 1 - 1/n$

(b) What is the probability that the second bootstrap observation is *not* the jth observation from the original sample?

Given that obtaining a bootstrap sample involves sampling with replacement, all successive bootstrap observations have an equal probability of not being the jth observation from the original sample. This is because after the first observation, the state of the sample pool is exactly as it was before the first observation, due to the observation being replaced after the first observation. Therefore, the probability of the second bootstrap observation will be the same as the first: $1 - 1/n$.

(c) Argue that the probability that the jth observation is *not* in the bootstrap sample is $(1 - 1/n)^n$.

The bootstrap technique can only be used given that the assumption that each observation is independent holds. Therefore, the product rule of probability can be applied to the probability of individual observations (found in the previous question) over all n observations:

$$(1 - 1/n)_1 * (1 - 1/n)_2 * \ldots * (1 - 1/n)_n = (1 - 1/n)^n$$

Christian Caburian
220133475

(d) When $n = 5$, what is the probability that the jth observation is in the bootstrap sample?

P(is not in the sample) = $(1 - 1/5)^5 = 0.8^5 = 0.32768$ or 32.768%.
P(is in the sample) = 1 - P(is not in the sample) = 0.67232 or 67.232%

(e) When $n = 100$, what is the probability that the jth observation is in the bootstrap sample?

P(is not in the sample) = $(1 - 1/100)^{100} = 0.99^{100} = 0.36603$ or 36.603%.
P(is in the sample) = 1 - P(is not in the sample) = 0.63397 or 63.397%

(f) When $n = 10,000$, what is the probability that the jth observation is in the bootstrap sample?

P(is not in the sample) = $(1 - 1/10000)^{10000} = 0.9999^{10000} = 0.36786$ or 36.768%.
P(is in the sample) = 1 - P(is not in the sample) = 0.63232 or 63.232%

(g) Create a plot that displays, for each integer value of n from 1 to $100,000$, the probability that the jth observation is in the bootstrap sample. Comment on what you observe.

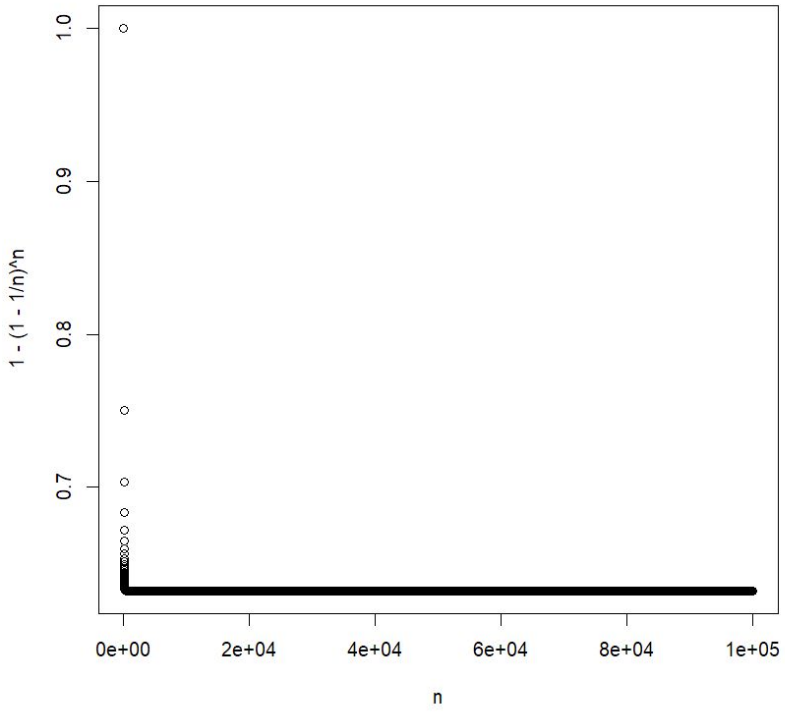

Figure 1: The plot of the probability that the jth observation is in the bootstrap sample.

There appears to be some limit that is approached as n increases. From the calculation below, the limit is found to be 0.632. That is, as the number of observations approaches infinity, the probability that that the jth observation is in the bootstrap sample approaches an asymptote of 0.63212 or 63.212% in the negative direction.

$$\lim_{n\to\infty}\left(1-\left(1-\frac{1}{n}\right)^n\right)=1-\frac{1}{e} \quad \text{(Decimal: } 0.63212\ldots\text{)}$$

Given the previous probability (0.63397) calculated for n = 100 is already very close to the limit, there appears to be almost no decrease in probability as the number of observations increases from around n = 100.

STAT330: Assignment 2

(h) We will now investigate numerically the probability that a bootstrap sample of size $n = 100$ contains the jth observation. Here $j = 4$. We repeatedly create bootstrap samples, and each time we record whether or not the fourth observation is contained in the bootstrap sample.

```
> store=rep(NA, 10000)
> for(i in 1:10000){
    store[i]=sum(sample(1:100, rep=TRUE)==4)>0
}
> mean(store)
```

Comment on the results obtained.

The result found was 0.6408 (with set.seed(1)) which appears to agree very closely with the calculation of the probability when n = 100, i.e. 0.63397, and also with the calculated limit being 0.63212.

Christian Caburian
220133475

Question 2.

In Sections 5.3.2 and 5.3.3, we saw that the `cv.glm()` function can be used in order to compute the LOOCV test error estimate. Alternatively, one could compute those quantities using just the `glm()` and `predict.glm()` functions, and a for loop. You will now take this approach in order to compute the LOOCV error for a simple logistic regression model on the `Weekly` data set. Recall that in the context of classification problems, the LOOCV error is given in (5.4).

(a) Fit a logistic regression model that predicts `Direction` using `Lag1` and `Lag2`.

```
glm.fit1 <- glm(Direction ~ Lag1 + Lag2, data = Weekly, family = binomial)
summary(glm.fit1)
```

```
Call:
glm(formula = Direction ~ Lag1 + Lag2, family = binomial, data = Weekly)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.623  -1.261   1.001   1.083   1.506

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.22122    0.06147   3.599 0.000319 ***
Lag1        -0.03872    0.02622  -1.477 0.139672
Lag2         0.06025    0.02655   2.270 0.023232 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1496.2  on 1088  degrees of freedom
Residual deviance: 1488.2  on 1086  degrees of freedom
AIC: 1494.2

Number of Fisher Scoring iterations: 4
```

Table 1: Summary table for the logistic regression model predicting Direction using Lag1 and Lag2. The model uses all observations.

STAT330: Assignment 2

(b) Fit a logistic regression model that predicts `Direction` using `Lag1` and `Lag2` *using all but the first observation.*

```
nMinusn1 <- Weekly[-1, ]
glm.fit2 <- glm(Direction ~ Lag1 + Lag2, data = nMinusn1, family = binomial)
summary(glm.fit2)
```

```
Call:
glm(formula = Direction ~ Lag1 + Lag2, family = binomial, data =
nMinusn1)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6258  -1.2617   0.9999   1.0819   1.5071

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.22324    0.06150   3.630 0.000283 ***
Lag1        -0.03843    0.02622  -1.466 0.142683
Lag2         0.06085    0.02656   2.291 0.021971 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1494.6  on 1087  degrees of freedom
Residual deviance: 1486.5  on 1085  degrees of freedom
AIC: 1492.5

Number of Fisher Scoring iterations: 4
```

Table 2: Summary table for the logistic regression model predicting Direction using Lag1 and Lag2. The model uses all but the first observation.

STAT330: Assignment 2

(c) Use the model from (b) to predict the direction of the first observation. You can do this by predicting that the first observation will go up if $P(\texttt{Direction="Up"}|\texttt{Lag1}, \texttt{Lag2}) > 0.5$. Was this observation correctly classified?

```
glm.probs <- predict(glm.fit2, newdata = Weekly[1, ], type = "response")
glm.probs > 0.5 #> TRUE (Up)
Weekly[1,]$Direction #> "Down"
```

Using the model from b), Direction was predicted to be “Up”. However, the first observation for Direction was recorded as “Down”. Therefore, the observation was incorrectly classified.

(d) Write a for loop from $i = 1$ to $i = n$, where n is the number of observations in the data set, that performs each of the following steps:

i. Fit a logistic regression model using all but the ith observation to predict `Direction` using `Lag1` and `Lag2`.

ii. Compute the posterior probability of the market moving up for the ith observation.

iii. Use the posterior probability for the ith observation in order to predict whether or not the market moves up.

iv. Determine whether or not an error was made in predicting the direction for the ith observation. If an error was made, then indicate this as a 1, and otherwise indicate it as a 0.

```
n <- dim(Weekly)[1]
errors <- rep(0, n)
for(i in 1:n)
{
   glm.fit <- glm(Direction ~ Lag1 + Lag2, data = Weekly[-i, ], family = binomial)
   glm.probs <- predict(glm.fit, newdata = Weekly[i, ], type = "response")
   errors[i] <- as.integer((glm.probs > 0.5) != (Weekly[i, ]$Direction == "Up"))
}
```

STAT330: Assignment 2

(e) Take the average of the n numbers obtained in (d)iv in order to obtain the LOOCV estimate for the test error. Comment on the results.

```
numErrors <- sum(errors)
numErrors #> 490
LOOCVTestErrorEstimate <- mean(errors)
LOOCVTestErrorEstimate #> 44.99541% error rate
```

The error rate of 45% means that we would be better off guessing the direction of the market randomly. Given the data and implied application, the model performs poorly and should not be proposed for practical use.

STAT330: Assignment 2

Question 3.

We will now try to predict per capita crime rate in the `Boston` data set.

(a) Try out some of the regression methods explored in this chapter, such as best subset selection, the lasso, ridge regression, and PCR. Present and discuss results for the approaches that you consider.

Best Subset Selection

First, best subset selection was used with k = 10 fold cross validation, to determine the number of variables that fits the model with the lowest mean CV error. In figure 2, it appears that this method reports the 9 variable model having the lowest mean CV error of 43.47287 (see Appendix for R code). The variables selected are listed in Table 3.

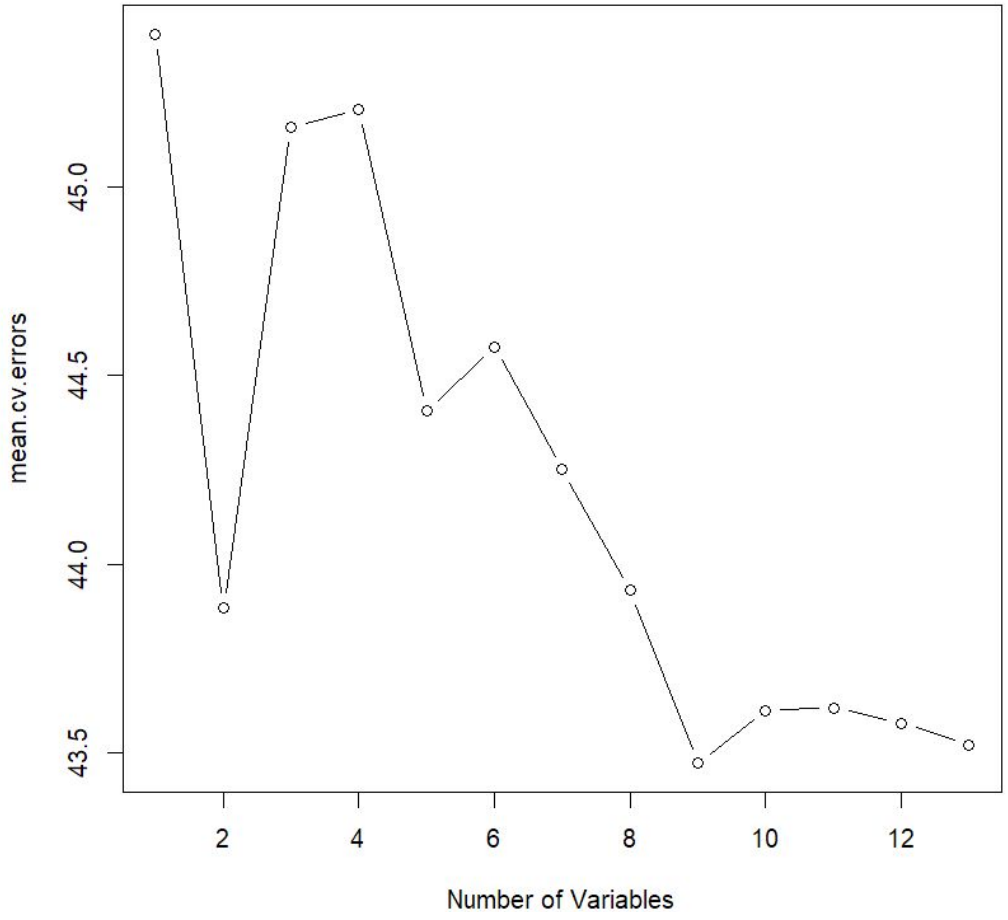

Figure 2: Plot of p-variable models with corresponding mean CV errors found using best subset selection.

(Intercept)	zn	indus	nox	dis
21.056533195	0.045345047	-0.090954491	-11.595481360	-1.087406625
rad	ptratio	black	lstat	medv
0.547936841	-0.274318819	-0.009357563	0.096806735	-0.191928059

Table 3: The coefficient estimates in the model obtained using the best subset selection method.

Ridge Regression

Cross validation was applied with a ridge regression model to select the tuning parameter. In figure 3, the plot of log lambdas appears to show that the minimum lambda corresponds to the value of 0.79 (found in R). This lambda was then used to fit a ridge regression model to find the test MSE of 38.36587. This is a substantial improvement over best subset selection but suffers the disadvantage of requiring all variables in the model. At this point, the higher test MSE for the 9 variable model obtained using best subset selection may not be enough of a downside to discard it for this model with its added complexity.

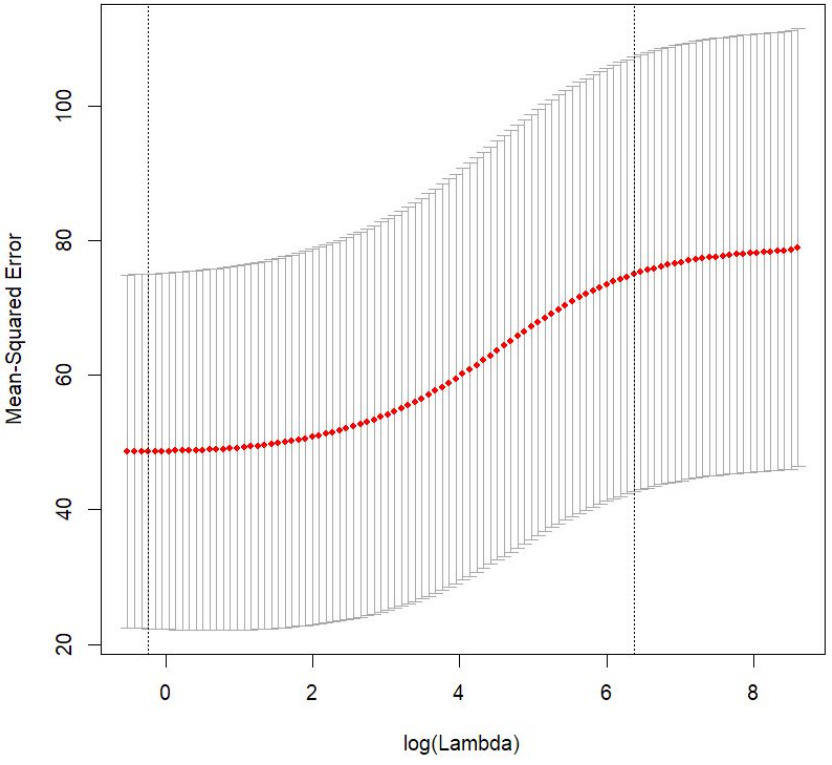

Figure 3: Plot of log lambdas with corresponding MSE found using ridge regression.

STAT330: Assignment 2

Lasso

Cross validation was also applied in a lasso model to select the tuning parameter. In figure 3, the plot of log lambdas appears to show that the minimum lambda corresponds to the value of 0.1 (found in R).

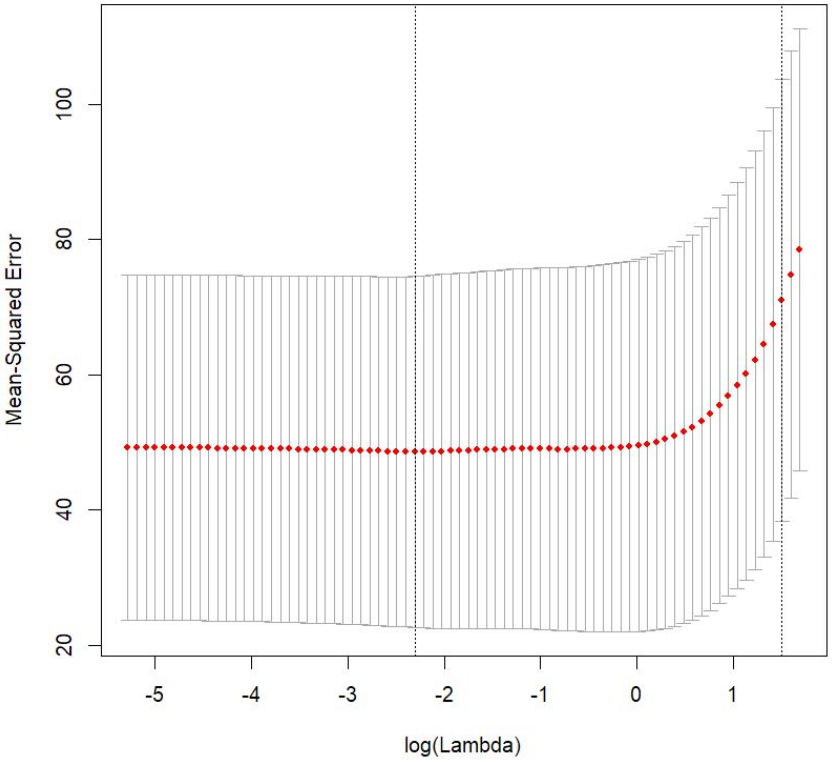

Figure 4: Plot of log lambdas with corresponding MSE found using the lasso method.

This lambda was then used to fit a lasso model to find the test MSE of 38.3096. Not only is this is a substantial improvement over best subset selection but also slightly improves the ridge regression result. Additionally, this method has performed variable selection, reducing the number of variables to 12, as seen in Table 4.

```
14 x 1 sparse Matrix of class "dgCMatrix"
                       1
(Intercept)  9.262700913
zn           0.031356409
indus       -0.051023135
chas        -0.512648901
nox         -3.755451657
rm           0.041320041
age          .
dis         -0.600700390
rad          0.494793892
tax          .
ptratio     -0.107509984
black       -0.007556396
lstat        0.118431941
medv        -0.126165598
```

Table 4: The coefficient estimates in the model obtained using the lasso method.

The advantages offered separately by best subset selection and ridge regression are provided simultaneously by the model obtained by lasso. Although it does not reduce as many variables as the model obtained by best subset selection, the complexity added by 3 variables can be argued to be negligible given some context.

STAT330: Assignment 2

Principal Components Regression

The PCR method was applied with CV which resulted in selecting the M = 13 component model as it had the lowest CV RMSEP of 7.00 (seen in Figure 5 and Table 5). This 13 component model was then validated with a test MSE of 39.27592. Given that it only improves in test MSE over the best subset selection as well as performing no variable selection, this model cannot compete with the model obtained using the lasso method.

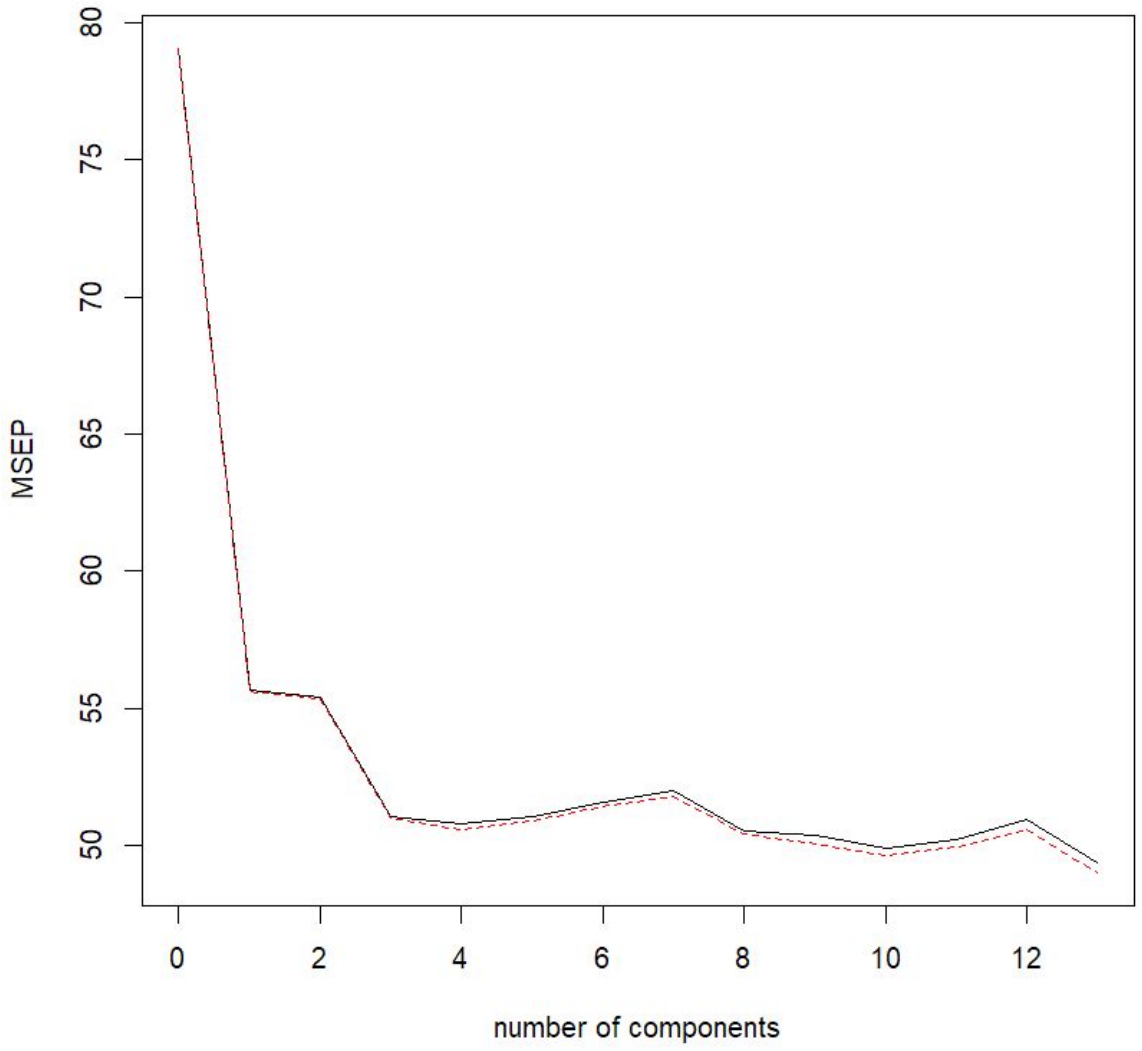

Figure 5: The plot of MSEP's corresponding to the number of components found using PCR.

```
Data:   X dimension: 253 13
        Y dimension: 253 1
Fit method: svdpc
Number of components considered: 13

VALIDATION: RMSEP
Cross-validated using 10 random segments.
       (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
CV           8.892    7.459    7.444    7.146    7.128    7.143    7.181
adjCV        8.892    7.456    7.440    7.140    7.113    7.136    7.170
       7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
CV       7.209    7.108    7.097     7.065     7.086     7.137     7.025
adjCV    7.196    7.099    7.075     7.045     7.066     7.112     7.000

TRAINING: % variance explained
      1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8
comps
X       49.04    60.72    69.75    76.49    83.02    88.40    91.73
93.77
crim    30.39    30.93    36.63    37.31    37.35    37.98    38.85
39.94
      9 comps  10 comps  11 comps  12 comps  13 comps
X       95.73     97.36     98.62     99.57    100.00
crim    41.89     42.73     42.73     43.55     45.48
```

Table 5: Summary table for the PCR, listing the CV RMSEP corresponding to models of M components.

STAT330: Assignment 2

Partial Least Squares

The PLS method was applied with CV which resulted in selecting the M = 10 component model as it had the lowest CV RMSEP of 6.99 (seen in Figure 6 and Table 6). Although it is not the lowest by much, and can be see that from M = 9 components onwards, there is almost no change in CV error. This 10 component model was then validated with a test MSE of 39.26028. This improves only slightly ahead of the PCR method with the same disadvantages and is therefore is not in consideration for proposal.

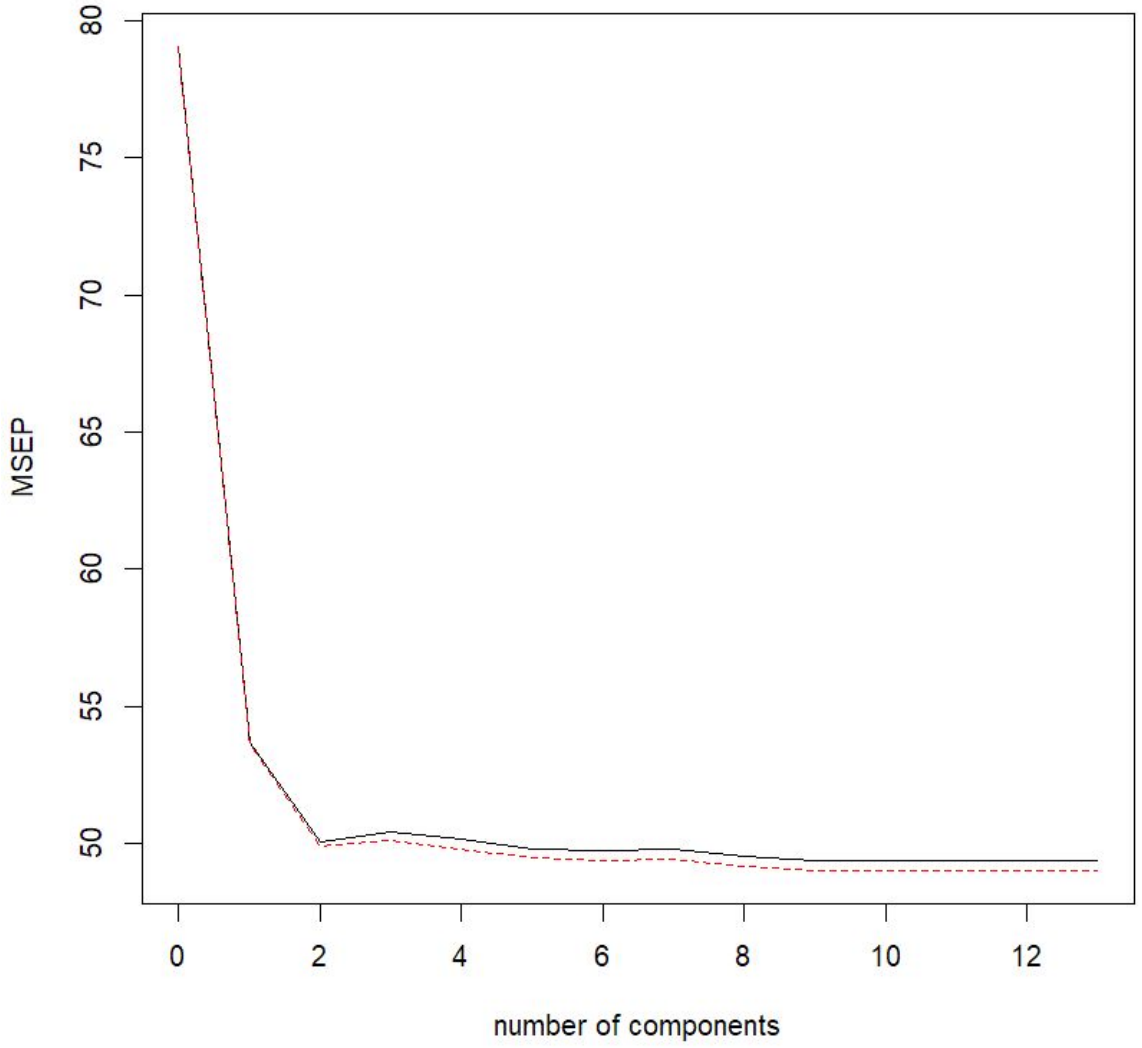

Figure 6: The plot of MSEP's corresponding to the number of components found using PLS.

```
Data:   X dimension: 253 13
        Y dimension: 253 1
Fit method: kernelpls
Number of components considered: 13

VALIDATION: RMSEP
Cross-validated using 10 random segments.
       (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
CV           8.892    7.327    7.074    7.101    7.081    7.057    7.051
adjCV        8.892    7.323    7.064    7.077    7.056    7.033    7.025
       7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
CV       7.056    7.037    7.026     7.025     7.025     7.025     7.025
adjCV    7.028    7.010    7.001     6.999     7.000     7.000     7.000

TRAINING: % variance explained
      1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8
comps
X       48.70    57.72    63.00    70.98    76.61    80.06    83.65
87.40
crim    33.37    40.66    43.35    44.26    44.72    45.15    45.37
45.44
      9 comps  10 comps  11 comps  12 comps  13 comps
X       88.87     94.22     96.76     98.71    100.00
crim    45.47     45.48     45.48     45.48     45.48
```

Table 6: Summary table for the PLS, listing the CV RMSEP corresponding to models of M components.

(b) Propose a model (or set of models) that seem to perform well on this data set, and justify your answer. Make sure that you are evaluating model performance using validation set error, cross-validation, or some other reasonable alternative, as opposed to using training error.

```
testMSEs #> 43.47287 38.36587 38.30960 39.27592 39.26028
```

According to the corresponding test MSEs listed in “testMSEs”, the lowest value is the third, which corresponds to the third model which was obtained using the lasso method. The second lowest test MSE comes from the model obtained by ridge regression which consists of all variables. Considering this, the lasso model is chosen as the best and is proposed because it has performed well on the validation data set while reducing the number of variables for better interpretability.

(c) Does your chosen model involve all of the features in the data set? Why or why not?

The chosen model does not involve all of the features in the data set, as seen in Table 4, where it has discarded the variables “age” and “tax”. The reason that the lasso method has performed variable selection is because it has found a value for its tuning parameter where a region of constant RSS intersects with the polytope shaped constraint region where the coefficient for tax and age equal 0.

STAT330: Assignment 2

Appendix: R Code.

Available at https://github.com/ianacaburian/Statistical_Learning

Question 1.

```r
# g)
# Figure 1
n <- 1:100000
plot(n, 1 - (1 - 1/n)^n)


# h)
store <- rep(NA, 10000)
set.seed(1)
for (i in 1:10000)
{
   store[i] <- sum(sample(1:100, rep = TRUE) == 4) > 0
}
mean(store) #> 0.6408
```

STAT330: Assignment 2

Question 2.

```
library(ISLR)
library(boot)
summary(Weekly)
set.seed(1)
attach(Weekly)
#_____________________________________________________________________________
# a)
glm.fit1 <- glm(Direction ~ Lag1 + Lag2, data = Weekly, family = binomial)
summary(glm.fit1)

# b)
nMinusn1 <- Weekly[-1, ]
glm.fit2 <- glm(Direction ~ Lag1 + Lag2, data = nMinusn1, family = binomial)
summary(glm.fit2)

# c)
glm.probs <- predict(glm.fit2, newdata = Weekly[1, ], type = "response")
glm.probs > 0.5 #> TRUE (Up)
Weekly[1,]$Direction #> "Down"
   # Predicted incorrectly

# d)
n <- dim(Weekly)[1]
errors <- rep(0, n)
for(i in 1:n)
{
   glm.fit <- glm(Direction ~ Lag1 + Lag2, data = Weekly[-i, ], family = binomial)
   glm.probs <- predict(glm.fit, newdata = Weekly[i, ], type = "response")
   errors[i] <- as.integer((glm.probs > 0.5) != (Weekly[i, ]$Direction == "Up"))
}

# e)
numErrors <- sum(errors)
numErrors #> 490
LOOCVTestErrorEstimate <- mean(errors)
LOOCVTestErrorEstimate #> 44.99541% error rate
```

STAT330: Assignment 2

Question 3.

```
library(MASS)
library(leaps)
library(glmnet)
library(ISLR)
library(pls)
attach(Boston)
sum(is.na(Boston)) #> 0

x.all <- model.matrix(crim ~ ., data = Boston)
y.all <- Boston$crim
testMSEs <- rep(0, 5)

# a)

# Best Subset Selection

predict.regsubsets <- function(object, newdata, id, ...)
{
   form <- as.formula(object$call[[2]])
   mat <- model.matrix(form, newdata)
   coefi <- coef(object, id = id)
   xvars <- names(coefi)
   mat[, xvars]%*%coefi
}

k <- 10
p <- ncol(Boston) - 1
set.seed(1)
folds = sample(rep(1:k, length = nrow(Boston)))
   # Text book uses replace = TRUE here, however it also stresses that folds
   # must be the same size and non-overlapping. Therefore, the default
   # replace = FALSE is used.

cv.errors <- matrix(NA, k, p, dimnames = list(NULL, paste(1:p)))
```

```
for (j in 1:k)
{
   best.fit <- regsubsets(crim ~ ., data = Boston[folds != j,], nvmax = p)
   for (i in 1:p)
   {
       pred <- predict(best.fit, Boston[folds == j,], id = i)
       cv.errors[j, i] <- mean((y.all[folds == j] - pred)^2)
   }
}

mean.cv.errors <- apply(cv.errors, 2, mean)
best.model <- which.min(mean.cv.errors) #> 9
par(mfrow = c(1, 1))
plot(mean.cv.errors, type = 'b', xlab = "Number of Variables")
   # We see that CV selects a 9-var model.
coef(best.fit, id = 9)
   # Variables: zn, indus, nox, dis, rad, ptratio, black, lstat, medv.

testMSEs[1] <- mean.cv.errors[best.model]
testMSEs[1] #> 43.47287
#_____________________________________________________________________________
```

STAT330: Assignment 2

```r
# Ridge Regression with CV selected tuning parameter.

x <- model.matrix(crim ~ ., Boston)[, -1]
y <- Boston$crim
set.seed(1)
train <- sample(1:nrow(x), nrow(x) / 2)
test <- (-train)
y.test <- y[test]

# Select tuning parameter via CV
set.seed(1)
cv.out <- cv.glmnet(x[train, ], y[train], alpha = 0)
plot(cv.out)
bestRidgeLambda <- cv.out$lambda.min
bestRidgeLambda #> 0.7908625

# Test the selected model and report its test error.
grid = 10^seq(10, -2, length = 100)
ridge.mod <- glmnet(x[train, ], y[train], alpha = 0, lambda = grid, thresh = 1e-12)
ridge.pred <- predict(ridge.mod, s = bestRidgeLambda, newx = x[test, ])
testMSEs[2] <- mean((ridge.pred - y.test)^2)
testMSEs[2] #> 38.36587
#_____________________________________________________________________________
```

STAT330: Assignment 2

```
# Lasso with CV selected tuning parameter.

lasso.mod <- glmnet(x[train,], y[train], alpha = 1, lambda = grid)
plot(lasso.mod)
set.seed(1)
cv.out <- cv.glmnet(x[train,], y[train], alpha = 1)
plot(cv.out)

bestLassoLambda <- cv.out$lambda.min
bestLassoLambda #> 0.09979553
lasso.pred <- predict(lasso.mod, s = bestLassoLambda, newx = x[test,])
testMSEs[3] <- mean((lasso.pred - y.test)^2)
testMSEs[3] #> 38.3096

out <- glmnet(x, y, alpha = 1, lambda = grid)
lasso.coef <- predict(out, type = "coefficients", s = bestLassoLambda)
lasso.coef #> 12 non-zero coefficients.
#_____________________________________________________________________________

# PCR

set.seed(1)
pcr.fit <- pcr(crim ~ ., data = Boston, subset = train,
               scale = TRUE, validation = "CV")
validationplot(pcr.fit, val.type = "MSEP")
   # 13 components appear to have the lowest MSEP.
summary(pcr.fit)
   # The 13 component model has CV RMSEP of 7.025 - the lowest.

M <- 13
pcr.pred <- predict(pcr.fit, x[test,], ncomp = M)
testMSEs[4] <- mean((pcr.pred - y.test) ^2)
testMSEs[4] #> 39.27592
#_____________________________________________________________________________
```

STAT330: Assignment 2

```
# PLS

set.seed(1)
pls.fit <- plsr(crim ~ ., data = Boston, subset = train, scale = TRUE, validation =
"CV")
validationplot(pls.fit, val.type = "MSEP")
   # From 9 components onwards, RMSEP (at its lowest) appears to be
   # indistinguishable.
summary(pls.fit)
   # The lowest adjCV RMSEP of 6.99 comes from the 10 component model.
   # However, like the plot, the CV is reported to be almost equal from
   # 9 to 13 components.

M <- 10
pls.pred <- predict(pls.fit, x[test,], ncomp = M)
testMSEs[5] <- mean((pls.pred - y.test)^2)
testMSEs[5] #> 39.26028
#_____________________________________________________________________________

# b)

testMSEs #> 43.47287 38.36587 38.30960 39.27592 39.26028
#_____________________________________________________________________________
```