FE590. Assignment #2.

2022-03-12

I pledge on my honor that I have not given or received any unauthorized assistance on this assignment/examination. I further pledge that I have not copied any material from a book, article, the Internet or any other source except where I have expressly cited the source.

By filling out the following fields, you are signing this pledge. No assignment will get credit without being pledged.

Name:Zongqi Shen

CWID:10479206

Date:02/28/2022

Instructions

In this assignment, you should use R markdown to answer the questions below. Simply type your R code into embedded chunks as shown above. When you have completed the assignment, knit the document into a PDF file, and upload both the .pdf and .Rmd files to Canvas.

```
CWID = 10479206 #Place here your Campus wide ID number, this will personalize
#your results, but still maintain the reproduceable nature of using seeds.
#If you ever need to reset the seed in this assignment, use this as your seed
#Papers that use -1 as this CWID variable will earn 0's so make sure you change
#this value before you submit your work.
personal = CWID %% 10000
set.seed(personal)#You can reset the seed at any time in your code,
#but please always set it to this seed.
```

Question 1

Create a .csv file consisting of daily adjusted close prices for 10 different stocks and 2 ETF's. You should have at least two years of data for every asset and ETF and should include the date for your data to make sure that you are including everything appropriately. After creating the file, put it in your working directory (or move your working directory to where its stored). Read the data into R.

```
#library("quantmod")
#All_ETF <- c("AMZN","NKE","VALE","NOK","AAL","CCL","TGT","DQ","BAC","IWS","IWN")
#data_merged <- get(getSymbols("AMD", src="yahoo",from = '2019-01-01'))[ ,6]
#for (i in All_ETF){
#  single_ETF <- get(getSymbols(i,from ="2019-01-01"))[,6]
#  data_merged <- merge(data_merged, single_ETF)
#}
#write.csv(data_merged,"FA590_hw2dataset.csv", row.names = FALSE)
stock <- read.csv("~/Documents/Stevens_second_semester/FA590/Homework/Homework2/FA590_hw2dataset.csv")
stock <- na.omit(stock)
```

1. List the names of the variables in the data set.

```
colnames(stock)
```

```
##  [1] "AMD.Adjusted"  "AMZN.Adjusted" "NKE.Adjusted"  "VALE.Adjusted"
##  [5] "NOK.Adjusted"  "AAL.Adjusted"  "CCL.Adjusted"  "TGT.Adjusted"
##  [9] "DQ.Adjusted"   "BAC.Adjusted"  "IWS.Adjusted"  "IWN.Adjusted"
```

2. As the date will be unimportant, remove that field from your data frame

```
#I select dataframe format, when i download the data, so i don't have date in my data
```

3. What is the range of each quantitative variable? Answer this question using the range() function with the sapply() function e.g., sapply(cars, range). Print a simple table of the ranges of the variables. The rows should correspond to the variables. The first column should be the lowest value of the corresponding variable, and the second column should be the maximum value of the variable. The columns should be suitably labeled.

```
stock.range <- sapply(stock, range)
stock.range <- t(stock.range)
colnames(stock.range) <- c("min", "max")
stock.range
```

```
##                       min        max
## AMD.Adjusted    17.049999  161.91000
## AMZN.Adjusted 1500.280029 3731.40991
## NKE.Adjusted    61.914204  177.19153
## VALE.Adjusted    5.356024   20.76196
## NOK.Adjusted     2.420000    6.55000
## AAL.Adjusted     9.040000   36.44143
## CCL.Adjusted     7.970000   56.31021
## TGT.Adjusted    61.140236  265.24051
## DQ.Adjusted      4.660000  124.13000
## BAC.Adjusted    17.385441   49.38000
## IWS.Adjusted    52.292778  123.59000
## IWN.Adjusted    69.230812  175.90698
```

4. What is the mean and standard deviation of each variable? Create a simple table of the means and standard deviations.

```
stock.mean <- sapply(stock, mean)
stock.sd <- sapply(stock, sd)
mandv <- rbind(stock.mean, stock.sd)
mandv <- t(mandv)
colnames(mandv) <- c("mean", "standard_deviation")
mandv
```

```
##                       mean standard_deviation
## AMD.Adjusted    68.518271         35.2789871
## AMZN.Adjusted 2629.674911        709.3674865
## NKE.Adjusted   115.030171         30.8587397
## VALE.Adjusted   11.994084          3.7430119
## NOK.Adjusted     4.786584          0.9090762
## AAL.Adjusted    21.823386          7.3526098
## CCL.Adjusted    30.394273         13.8344083
## TGT.Adjusted   149.982120         61.8274418
## DQ.Adjusted     33.800622         28.8863149
## BAC.Adjusted    31.909306          7.9390241
## IWS.Adjusted    93.440042         16.8098525
## IWN.Adjusted   127.341176         26.8439167
```

5. Using the regsubsets function in the leaps library, regress one of your ETF's on the remaining assets.

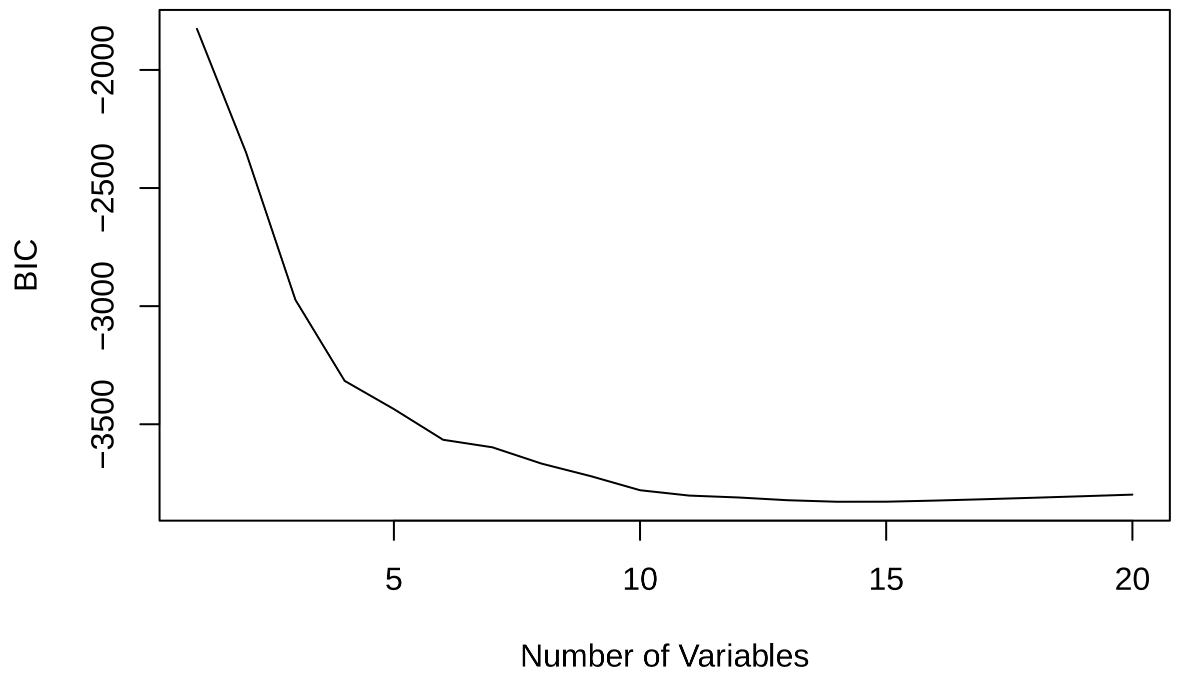
```r
library(leaps)
attach(stock)
newdf <- stock[-11]
colnames(newdf) <- c("AMD","AMZN","NKE","VALE","NOK","AAL","CCL","TGT","DQ","BAC","IWN")
m <- regsubsets(IWN.Adjusted~., data = newdf)
```

a. Print a table showing what variables would be selected using best subset selection for all predictors up to order 2 (i.e. asset and asset^2). Determine the optimal model using BIC and output the model, including its coeffecients.

```r
dewdf.re <- newdf
srstock = NULL
for (i in 1:10){
  dewdf.re <- cbind(dewdf.re,dewdf.re[i]^2)
}
#here the stock with * is stock^2
names(dewdf.re)[12:21] <- c("AMD*","AMZN*","NKE*","VALE*","NOK*","AAL*","CCL*","TGT*","DQ*","BAC*")
#write.csv(dewdf.re,"poly.data.csv", row.names = FALSE)
m <- regsubsets(IWN~., data = dewdf.re,really.big=T,nvmax=20)
summary_m <- summary(m)
min_m = which.min(summary_m$bic)
min_m
```

```
## [1] 14
```

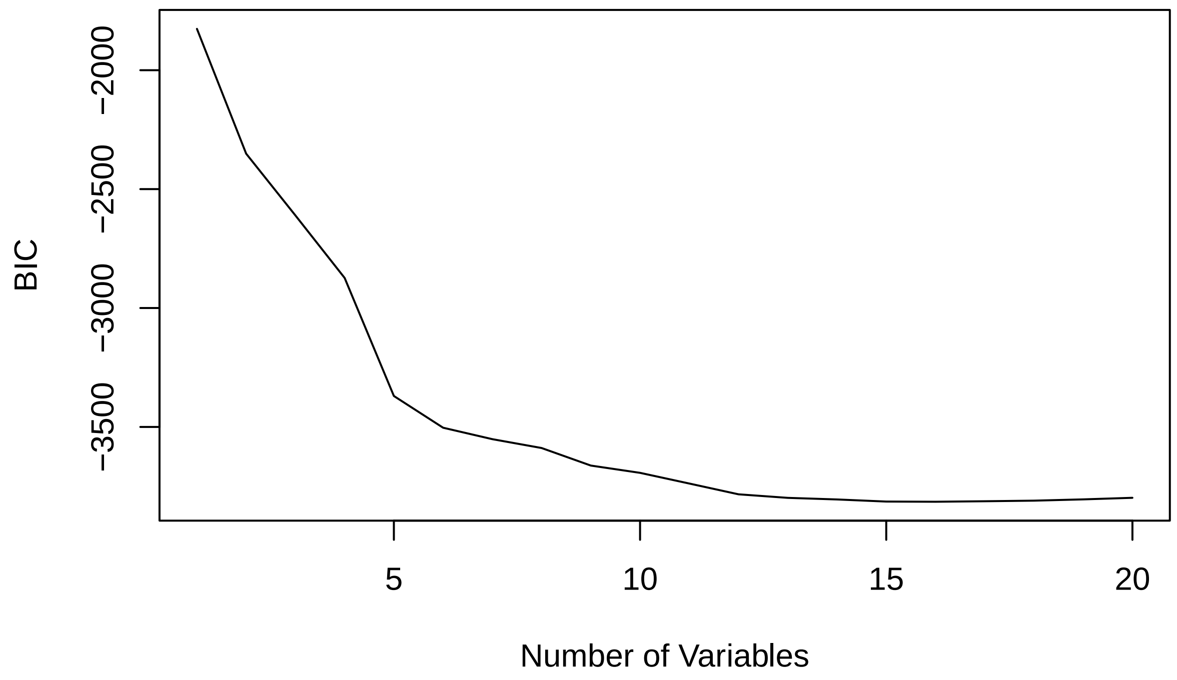
```r
plot(summary_m$bic, xlab = "Number of Variables", ylab = "BIC", type = "l")
```



```
#choose "AMD","AMZN","CCL","TGT","DQ","BAC","AMD*","AMZN*","VALE*","NOK*","AAL*","CCL*","DQ*","BAC*"
```

b. Print a table showing what variables would be selected using forward subset selection for all predictors up to order 2 (i.e. asset and asset^2). Determine the optimal model using BIC and output the model, including its coeffecients.

```
a=regsubsets(IWN~., data = dewdf.re,method="forward",nvmax=20)
summary_a <- summary(a)
#as.data.frame(t(summary(a)$which)[,9])
bic_min = which.min(summary_a$bic)
plot(summary_a$bic, xlab = "Number of Variables", ylab = "BIC", type = "l")
```



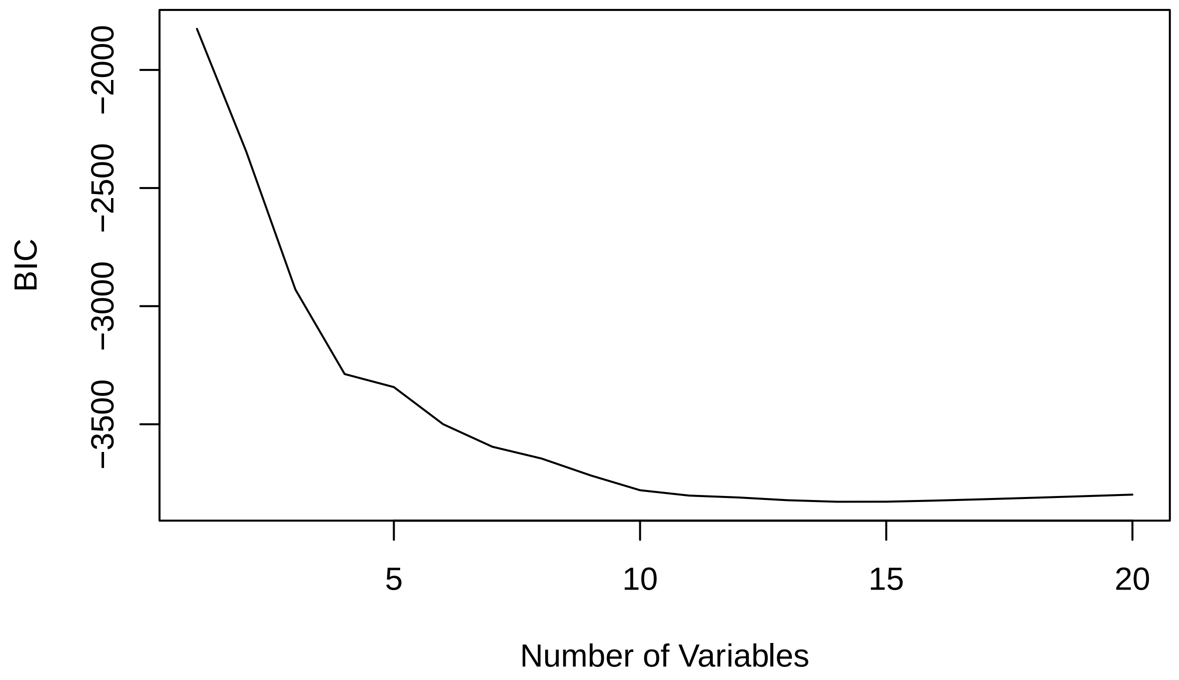
```
bic_min = which.min(summary_a$bic)
bic_min
```

```
## [1] 16
```

```
#choose "AMZN","NKE","VALE","NOK","AAL","CCL","TGT","DQ","BAC","AMD*","AMZN*","VALE*","NOK*","CCL*","DQ
```

c. Print a table showing what variables would be selected using backward subset selection for all predictors up to order 2 (i.e. asset and asset^2). Determine the optimal model using adjusted R^2 and output the model, including its coeffecients.

```
a=regsubsets(IWN~., data = dewdf.re,method="backward",nvmax=20)
summary_a <- summary(a)
#as.data.frame(t(summary(a)$which)[,9])
bic_min = which.min(summary_a$bic)
plot(summary_a$bic, xlab = "Number of Variables", ylab = "BIC", type = "l")
```



```
bic_min = which.min(summary_a$bic)
bic_min
```

```
## [1] 14
```

```
#choose "AMD","AMZN","CCL","TGT","DQ","BAC","IWN","AMD*","AMZN*","VALE*","NOK*","AAL*","CCL*","DQ*","BA
```

Question 2

Using the data set that you loaded for the first problem, choose the other ETF, and create a data frame consisting of simple lagged returns going up to 10 days back. Create another field in this data frame that looks to the direction of the ETF moving one day into the future, this direction should be listed as a factor, not a number.

```
stock.q2 <- stock[-12]
IWS=stock.q2$IWS.Adjusted
length(IWS)
```

```
## [1] 798
```

```
n <- length(IWS)
LogLag1=log(IWS[10:795]/IWS[9:794])
```

```
LogLag2=log(IWS[10:795]/IWS[8:793])
LogLag3=log(IWS[10:795]/IWS[7:792])
LogLag4=log(IWS[10:795]/IWS[6:791])
LogLag5=log(IWS[10:795]/IWS[5:790])
LogLag6=log(IWS[10:795]/IWS[4:789])
LogLag7=log(IWS[10:795]/IWS[3:788])
LogLag8=log(IWS[10:795]/IWS[2:787])
LogLag9=log(IWS[10:795]/IWS[1:786])
stock.q2.direction <- cbind(LogLag1,LogLag2,LogLag3,LogLag4,LogLag5,LogLag6,LogLag7,LogLag8,LogLag9)
stock.q2.direction <- as.data.frame(stock.q2.direction)
#Direction=sign()
IWSlog <- sign(log(IWS[-1]/IWS[-n]))
IWS[IWSlog==1]="Up"
IWS[IWSlog!=1]="Down"
IWS <- IWS[11:796]
IWS <- factor(IWS)
stock.q2.direction <- cbind(stock.q2.direction, IWS)
```

1. Split your data into a training set and a testing set and run LDA on the direction using all 10 different lags on the training set. How accurate is your model?

```
library(MASS)
train.num <- 1:round(length(IWS)-(length(IWS)/3))
stock.q2.train <- stock.q2.direction[1:round(length(IWS)-(length(IWS)/3)), ]
stock.q2.train <- as.data.frame(stock.q2.train)
stock.q2.test <- stock.q2.direction[-train.num,]
stock.q2.test <- as.data.frame(stock.q2.test)
lda.fit <- lda(IWS~.,data = stock.q2.train)
lda.pred <- predict(lda.fit,stock.q2.test)
lda.class <- lda.pred$class
table(lda.class, stock.q2.test$IWS)
```

```
##
## lda.class Down  Up
##      Down    1   2
##      Up    119 140
```

```
mean(lda.class == stock.q2.test$IWS)
```

```
## [1] 0.5381679
```

2. Create code to determine the estimate for the expected test error using K=5 cross validation. Do this by actually splitting the date into five pieces and give the average of the test error, not just by using a command from a package (such as cv.glm).

```
library(pROC)
len <- length(IWS)
K=5
folds=seq(1:K)
member=sample(folds,len,replace=T)
cv.error=rep(0,K)
for (i in 1:K)
{
  train=stock.q2.direction[member!=i,]
  test=stock.q2.direction[member==i,]
  temp.mod=lda(IWS~.,data=train)
  temp.pre <- predict(temp.mod,test)
  real.temp <- ifelse(test[,10] == "Up", 1, 0)
  cv.error[i]=mean((as.numeric(real.temp) - (temp.pre)$posterior[,2])^2)
}
cv.error
```

```
## [1] 0.2459871 0.2658220 0.2536181 0.2455344 0.2438355
```

```
mean(cv.error)
```

```
## [1] 0.2509594
```

3. Determine the LOOCV estimate of the expected test error of your model. How do your answers to each part of this question compare? Do you see any noticable differences between your answer? Why do you think that is?

```
K=len

cv.error=rep(0,K)
for (i in 1:K)
{
  temp.mod=lda(IWS~.,data=stock.q2.direction[-i,])
  temp.pre <- predict(temp.mod,stock.q2.direction[i,])
  real.temp <- ifelse(stock.q2.direction[,10] == "Up", 1, 0)
  cv.error[i]=(as.numeric(real.temp[i]) - (temp.pre)$posterior[,2])^2

}
#cv.error
mean(cv.error)
```

```
## [1] 0.250529
```

Question 3

This question should be answered using the Weekly data set, which is part of the ISLR package. This data contains 1,089 weekly returns for 21 years, from the beginning of 1990 to the end of 2010.

1. What does the data represent?

```
library(ISLR)
?Weekly
```

2. Use the full data set to perform a logistic regression with Direction as the response and the five lag variables plus Volume as predictors. Use the summary function to print the results. Do any of the predictors appear to be statistically significant? If so, which ones?

```
attach(Weekly)
glm_model <- glm(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, family=binomial)
summary(glm_model)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
##     Volume, family = binomial)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6949  -1.2565   0.9913   1.0849   1.4579
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.26686    0.08593   3.106   0.0019 **
## Lag1        -0.04127    0.02641  -1.563   0.1181
## Lag2         0.05844    0.02686   2.175   0.0296 *
## Lag3        -0.01606    0.02666  -0.602   0.5469
## Lag4        -0.02779    0.02646  -1.050   0.2937
## Lag5        -0.01447    0.02638  -0.549   0.5833
## Volume      -0.02274    0.03690  -0.616   0.5377
## ---
## 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: 1486.4  on 1082  degrees of freedom
## AIC: 1500.4
##
## Number of Fisher Scoring iterations: 4
```

3. Fit a logistic regression model using a training data period from 1990 to 2008, using the predictors from the previous problem that you determined were statistically significant. Test your model on the held out data (that is, the data from 2009 and 2010) and express its accuracy.

```r
#From previous problem we can use Lag2
Weekly.train <- subset(Weekly, Year <= 2008)
Weekly.test <- subset(Weekly, Year > 2008)
mod <- glm(Direction ~ Lag2, data = Weekly.train, family = binomial)
glm_pro <- predict(mod, Weekly.test, type ="response")
glm.pred <- ifelse(glm_pro > 0.5, "Up", "Down")
table(glm.pred, Weekly.test$Direction)
```

```
##
## glm.pred Down Up
##     Down    9  5
##     Up     34 56
```

```r
glm.acc <- mean(glm.pred == Weekly.test$Direction)
glm.acc
```

```
## [1] 0.625
```

4. Repeat Part 3 using LDA.

```r
lda.model <- lda(Direction ~ Lag2, data = Weekly.train)
lda.pred <- predict(lda.model, Weekly.test)
lda.result <- lda.pred$class
table(lda.result, Weekly.test$Direction)
```

```
##
## lda.result Down Up
##       Down    9  5
##       Up     34 56
```

```r
lda.acc <- mean(lda.result == Weekly.test$Direction)
lda.acc
```

```
## [1] 0.625
```

```r
#confusionMatrix(table(lda.result, Weekly.test$Direction), positive = "Up")
```

5. Repeat Part 3 using QDA.

```r
qda.model <- qda(Direction ~ Lag2, data = Weekly.train)
qda.pred <- predict(qda.model, Weekly.test)
qda.result <- qda.pred$class
table(qda.result, Weekly.test$Direction)
```

```
##
## qda.result Down Up
##       Down    0  0
##       Up     43 61
```

```r
qda.acc <- mean(qda.result == Weekly.test$Direction)
qda.acc
```

```
## [1] 0.5865385
```

6. Repeat Part 3 using KNN with K = 1, 2, 3.

```r
library (class)
direction <- Weekly.train$Direction
Train.knn <- as.matrix(Weekly.train[, 3])
Test.knn <- as.matrix(Weekly.test[, 3])
#k = 1
knn.preda <- knn(Train.knn, Test.knn, direction, k = 1)
table(knn.preda, Weekly.test$Direction)
```

```
##
## knn.preda Down Up
##      Down   21 30
##      Up     22 31
```

```r
knna.acc <- mean(knn.preda == Weekly.test$Direction)
knna.acc
```

```
## [1] 0.5
```

```r
#k = 2
knn.predb <- knn(Train.knn, Test.knn, direction, k = 2)
table(knn.predb, Weekly.test$Direction)
```

```
##
## knn.predb Down Up
##      Down   19 28
##      Up     24 33
```

```r
knnb.acc <- mean(knn.predb == Weekly.test$Direction)
knnb.acc
```

```
## [1] 0.5
```

```r
#k = 3
knn.predc <- knn(Train.knn, Test.knn, direction, k = 3)
table(knn.predc, Weekly.test$Direction)
```

```
##
## knn.predc Down Up
##      Down   16 19
##      Up     27 42
```

```r
knnc.acc <- mean(knn.predc == Weekly.test$Direction)
knnc.acc
```

```
## [1] 0.5576923
```

7. Which of these methods in Parts 3, 4, 5, and 6 appears to provide the best results on this data?

#LDA and glm model have the best result

```r
best_results <- data.frame(glm.acc, lda.acc, qda.acc, knna.acc, knnb.acc, knnc.acc)
best_results
```

```
##   glm.acc lda.acc   qda.acc knna.acc knnb.acc  knnc.acc
## 1   0.625   0.625 0.5865385      0.5      0.5 0.5576923
```

Question 4

Write a function that works in R to gives you the parameters from a linear regression on a data set of n predictors. You can assume all the predictors and the prediction is numeric. Include in the output the standard error of your variables. You cannot use the lm command in this function or any of the other built in regression models.

```r
simple.linear.coef <- function(x, y) {
  n <- length(x)
  errorx <- sum(x^2) - sum(x)^2 / n
  errory <- sum(y^2) - sum(y)^2 / n
  errorxy <- sum(x * y) - (sum(x) * sum(y)) / n
  b1 <- errorxy / errorx
  b0 <- mean(y) - b1 * mean(x)
  sse <- errorx - errorxy^2 / errorx
  mse <- sse / (n - 2)
  b1.err <- sqrt(mse) / sqrt(errorx)
  b0.err <- sqrt(mse) / sqrt(n) * sqrt(1 + (mean(x)^2 / (sum((x - mean(x))^2) / n)))
  b0.t <- (b0 - 0) / b0.err
  b1.t <- (b1 - 0) / b1.err
  b0.p <- 2 * pt(b0.t, df = n - 2)
  b1.p <- 2 * pt(b1.t, df = n - 2, lower.tail = FALSE)
  r2 <- (errorx - sse) /errorx
  adj_r2 <- r2 - (1 - r2) * ((2 - 1) / (length(y) - 2))

  rsquare <- paste('Multiple R-squared: ', round(r2, 4), ', Adjusted R-squared: ', round(adj_r2, 4))

  coeffs <- data.frame(cbind(c(b0, b1), c(b0.err, b1.err), c(b0.t, b1.t), c(b0.p, b1.p)))
  colnames(coeffs) <- c('Estimate', 'Std. Error', 't value', 'Pr(>|t|)')
  rownames(coeffs) <- c('Intercept', 'x1')

  fitted <- x * b1 + b0

  msr <- sum((fitted - mean(y))^2) / 1
  mse2 <- sum((y - fitted)^2) / (length(y) - 2)
  f <- msr / mse2
  p <- pf(f, 1, length(y) - 2, lower.tail = FALSE)

  f.stat <- paste('F-statistic', round(f, 2), n - 2, 'p-value', p)
  resd <- y - fitted
  resdi <- paste('Residual standard error', round(sqrt(mse2), 2))
  regres <- list('Coefficients'=coeffs, resdi, rsquare, f.stat)

  return(regres)
}
simple.linear.coef(Weekly$Lag4, Weekly$Lag2)
```

```
## $Coefficients
##             Estimate Std. Error  t value   Pr(>|t|)
## Intercept 0.14257679 0.07157108 1.992101 1.95339017
## x1        0.05830672 0.03027929 1.925630 0.05441118
##
## [[2]]
## [1] "Residual standard error 2.35"
##
## [[3]]
## [1] "Multiple R-squared:  0.0034 , Adjusted R-squared:  0.0025"
##
## [[4]]
## [1] "F-statistic 3.72 1087 p-value 0.0541017537038782"
```

Compare the output of your function to that of the lm command in R.

```
model_check<- lm(Lag4 ~ Lag2, data = Weekly)
summary(model_check)
```

```
##
## Call:
## lm(formula = Lag4 ~ Lag2, data = Weekly)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -17.9356  -1.2845   0.1091   1.2743  11.8646
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.13699    0.07158   1.914   0.0559 .
## Lag2         0.05846    0.03032   1.928   0.0541 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.357 on 1087 degrees of freedom
## Multiple R-squared:  0.003408,   Adjusted R-squared:  0.002492
## F-statistic: 3.718 on 1 and 1087 DF,  p-value: 0.0541
```

Question 5

As you have probably seen in this homework, just simply looking at the close prices and trying to run models on the variables is not terribly interesting. You've begun to see what types of techniques we will be studying in this class. Here is an exerpt from the final project/homework:

"In this assignment, you will be required to find a set of data to run regression on. This data set should be financial in nature, and of a type that will work with the models we have discussed this semester (hint: we didn't look at time series) You may not use any of the data sets in the ISLR package that we have been looking at all semester. Your data set that you choose should have both qualitative and quantitative variables. (or has variables that you can transform)

Provide a description of the data below, where you obtained it, what the variable names are and what it is describing."

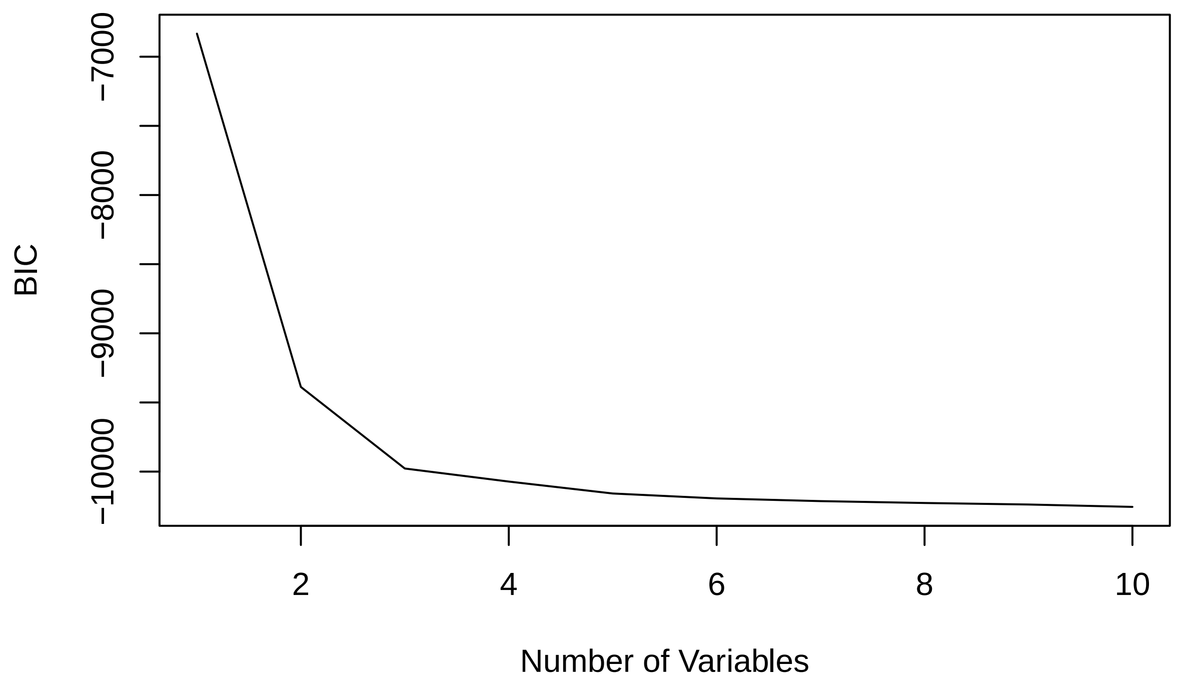
You don't have to actually create the data set at this time, but what sort of problem are you looking to solve? What data set would you need to answer this question? Please provide what you are looking into and how you could approach the problem below.

I want to explore such the relationship between interest rate of the loan and applicant's records (inq.last.6mths…) . I get this data set from kaggle. The goal is to fit models in this data

```r
library(gdata)
loan_data <- read.csv("~/Documents/Stevens_second_semester/FA590/Homework/Homework2/loan_data.csv")
head(loan_data)
```

```
##   credit.policy            purpose int.rate installment log.annual.inc   dti
## 1             1 debt_consolidation   0.1189      829.10       11.35041 19.48
## 2             1        credit_card   0.1071      228.22       11.08214 14.29
## 3             1 debt_consolidation   0.1357      366.86       10.37349 11.63
## 4             1 debt_consolidation   0.1008      162.34       11.35041  8.10
## 5             1        credit_card   0.1426      102.92       11.29973 14.97
## 6             1        credit_card   0.0788      125.13       11.90497 16.98
##   fico days.with.cr.line revol.bal revol.util inq.last.6mths delinq.2yrs
## 1  737          5639.958     28854       52.1              0           0
## 2  707          2760.000     33623       76.7              0           0
## 3  682          4710.000      3511       25.6              1           0
## 4  712          2699.958     33667       73.2              1           0
## 5  667          4066.000      4740       39.5               0            1
## 6  727          6120.042     50807       51.0               0            0
##   pub.rec not.fully.paid
## 1       0              0
## 2       0              0
## 3       0              0
## 4       0              0
## 5       0              0
## 6       0              0
```

```
a=regsubsets(int.rate~., data = loan_data,method="forward",nvmax=10)
summary_a <- summary(a)
#as.data.frame(t(summary(a)$which)[,9])
bic_min = which.min(summary_a$bic)
plot(summary_a$bic, xlab = "Number of Variables", ylab = "BIC", type = "l")
```



```
bic_min = which.min(summary_a$bic)
bic_min
```

```
## [1] 10
```

credit.policy: 1 if the customer meets the credit underwriting criteria of LendingClub.com, and 0 otherwise.

purpose: The purpose of the loan (takes values “creditcard”, “debtconsolidation”, “educational”, “major-purchase”, “smallbusiness”, and “all_other”).

int.rate: The interest rate of the loan, as a proportion (a rate of 11% would be stored as 0.11). Borrowers judged by LendingClub.com to be more risky are assigned higher interest rates.

installment: The monthly installments owed by the borrower if the loan is funded. log.annual.inc: The natural log of the self-reported annual income of the borrower.

dti: The debt-to-income ratio of the borrower (amount of debt divided by annual income). fico: The FICO credit score of the borrower.

days.with.cr.line: The number of days the borrower has had a credit line.

revol.bal: The borrower’s revolving balance (amount unpaid at the end of the credit card billing cycle).

revol.util: The borrower’s revolving line utilization rate (the amount of the credit line used relative to total credit available).

inq.last.6mths: The borrower’s number of inquiries by creditors in the last 6 months.

delinq.2yrs: The number of times the borrower had been 30+ days past due on a payment in the past 2 years.

pub.rec: The borrower’s number of derogatory public records (bankruptcy filings, tax liens, or judgments).