STAT 435 HW6

Chongyi Xu

May 17, 2018

Question 1

First this problem, you will analyze a data set of your choice.

(a) Describe the data in words.

The dataset I will use is `New York Stock Exchange` Dataset that I found at kaggle. The dataset is extraced from annual SEC 10k fillings. In order to have $n \approx p$, I will first reduce the size of original dataset(1781 observations) to 100 observations.

```
dat <- read.csv('fundamentals.csv', header=T)[1:100,]
colnames(dat)
```

```
##  [1] "Ticker.Symbol"
##  [2] "Period.Ending"
##  [3] "Accounts.Payable"
##  [4] "Accounts.Receivable"
##  [5] "Add.l.income.expense.items"
##  [6] "After.Tax.ROE"
##  [7] "Capital.Expenditures"
##  [8] "Capital.Surplus"
##  [9] "Cash.Ratio"
## [10] "Cash.and.Cash.Equivalents"
## [11] "Changes.in.Inventories"
## [12] "Common.Stocks"
## [13] "Cost.of.Revenue"
## [14] "Current.Ratio"
## [15] "Deferred.Asset.Charges"
## [16] "Deferred.Liability.Charges"
## [17] "Depreciation"
## [18] "Earnings.Before.Interest.and.Tax"
## [19] "Earnings.Before.Tax"
## [20] "Effect.of.Exchange.Rate"
## [21] "Equity.Earnings.Loss.Unconsolidated.Subsidiary"
## [22] "Fixed.Assets"
## [23] "Goodwill"
## [24] "Gross.Margin"
## [25] "Gross.Profit"
## [26] "Income.Tax"
## [27] "Intangible.Assets"
## [28] "Interest.Expense"
## [29] "Inventory"
## [30] "Investments"
## [31] "Liabilities"
## [32] "Long.Term.Debt"
## [33] "Long.Term.Investments"
## [34] "Minority.Interest"
## [35] "Misc..Stocks"
## [36] "Net.Borrowings"
```

```
## [37] "Net.Cash.Flow"
## [38] "Net.Cash.Flow.Operating"
## [39] "Net.Cash.Flows.Financing"
## [40] "Net.Cash.Flows.Investing"
## [41] "Net.Income"
## [42] "Net.Income.Adjustments"
## [43] "Net.Income.Applicable.to.Common.Shareholders"
## [44] "Net.Income.Cont..Operations"
## [45] "Net.Receivables"
## [46] "Non.Recurring.Items"
## [47] "Operating.Income"
## [48] "Operating.Margin"
## [49] "Other.Assets"
## [50] "Other.Current.Assets"
## [51] "Other.Current.Liabilities"
## [52] "Other.Equity"
## [53] "Other.Financing.Activities"
## [54] "Other.Investing.Activities"
## [55] "Other.Liabilities"
## [56] "Other.Operating.Activities"
## [57] "Other.Operating.Items"
## [58] "Pre.Tax.Margin"
## [59] "Pre.Tax.ROE"
## [60] "Profit.Margin"
## [61] "Quick.Ratio"
## [62] "Research.and.Development"
## [63] "Retained.Earnings"
## [64] "Sale.and.Purchase.of.Stock"
## [65] "Sales..General.and.Admin."
## [66] "Short.Term.Debt...Current.Portion.of.Long.Term.Debt"
## [67] "Short.Term.Investments"
## [68] "Total.Assets"
## [69] "Total.Current.Assets"
## [70] "Total.Current.Liabilities"
## [71] "Total.Equity"
## [72] "Total.Liabilities"
## [73] "Total.Liabilities...Equity"
## [74] "Total.Revenue"
## [75] "Treasury.Stock"
## [76] "For.Year"
## [77] "Earnings.Per.Share"
## [78] "Estimated.Shares.Outstanding"
```

Among the features we have above, we could remove some features that we are not really interested in, such as `Period Ending`, the ending period of the trade, and `For Year`, the year of the trade happened.

```
dat$For.Year <- NULL
dat$Period.Ending <- NULL
dat$Ticker.Symbol <- NULL
```

For this problem, I will use `Earnings Per Share` as my response Y. And study for how other features are related to earning. And note that there could be some missing values in the dataset. I first use `is.na()` to identify missing observations.

```r
sum(is.na(dat))
```

```
## [1] 92
```

Then we should use `na.omit()` to remove all observations that has missing values.

```r
dat <- na.omit(dat)
sum(is.na(dat))
```

```
## [1] 0
```

And now we would like to know our n and p.

```r
print(paste('n=', dim(dat)[1]))
```

```
## [1] "n= 69"
```

```r
print(paste('p=', dim(dat)[2]))
```

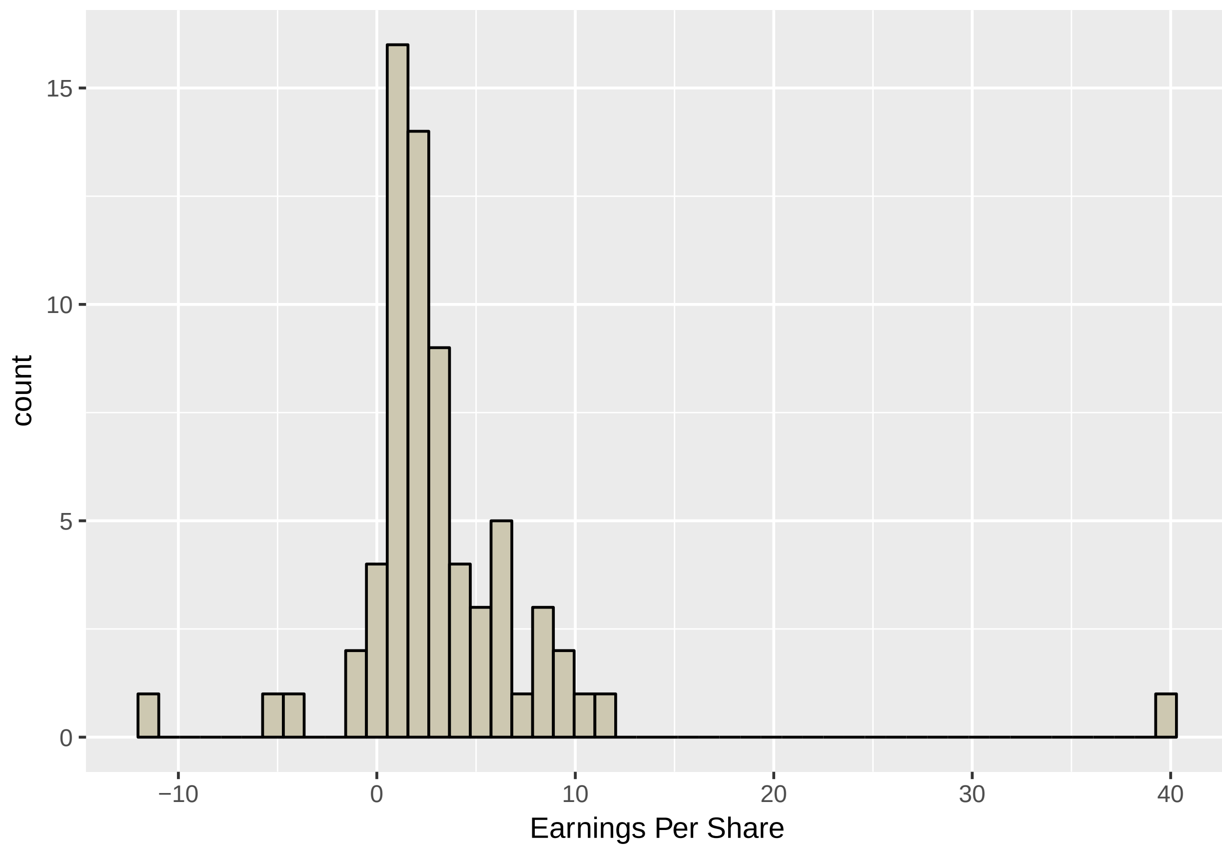
```
## [1] "p= 75"
```

Now we would like to have a look at the distribution of the response that we are interested in.

```r
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.4.4
```

```r
ggplot() + geom_histogram(aes(dat$Earnings.Per.Share),
                          bins=50,
                          color="black",
                          fill="cornsilk3") +
  xlab('Earnings Per Share')
```



From the plot we can see that there are only a few outliers and most are concentrated. This is a good news since some models could be robust some would be not.

(b) Split the data into a training set and a test set. What are the values of n and p on the training set?

```
set.seed(435)
train.index <- sample(nrow(dat),50)
train <- dat[train.index,]
test <- dat[-train.index,]
test.eps <- test$Earnings.Per.Share
print(paste('n_train=', dim(train)[1]))
```

```
## [1] "n_train= 50"
```

```
print(paste('p_train=', dim(train)[2]))
```

```
## [1] "p_train= 75"
```

(c) Fit a linear model using least squares on training set. And report the test error.

```
lm.model <- lm(data=train, Earnings.Per.Share~.)
lm.pred <- predict(lm.model, newdata=test)
print(paste('test MSE=',mean((test.eps-lm.pred)^2)))
```

```
## [1] "test MSE= 5015.91333521625"
```

(d) Fit a ridge model on the training set, with λ chosen by cross-validation. Report the test error.

```
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 3.4.4
```

```
## Loading required package: Matrix
## Loading required package: foreach
## Warning: package 'foreach' was built under R version 3.4.4
## Loaded glmnet 2.0-16
```

```
library(glmnetUtils)
```

```
##
## Attaching package: 'glmnetUtils'
## The following objects are masked from 'package:glmnet':
##
##     cv.glmnet, glmnet
```

```
grid <- 10^seq(10,-2,length=100)
ridge.model <- glmnet(Earnings.Per.Share ~ ., data=train, lambda=grid, alpha=0)
ridge.cv <- cv.glmnet(Earnings.Per.Share ~ ., data=train, alpha=0)
ridge.pred <- predict(ridge.model, s=ridge.cv$lambda.min,newdata=test)
print(paste('test MSE=', mean((test.eps-ridge.pred)^2)))
```

```
## [1] "test MSE= 73.9607557157376"
```

(e) Fit a lasso model on the training set, which *lambda* chosen by CV. Report the test error, along with the number of non-zero coefficient estimates.

```
lasso.model <- glmnet(Earnings.Per.Share ~ ., data=train, lambda=grid, alpha=1)
lasso.cv <- cv.glmnet(Earnings.Per.Share ~ ., data=train, alpha=1)
lasso.pred <- predict(lasso.model, s=lasso.cv$lambda.min, newdata=test)
print(paste('test MSE=', mean((test.eps-lasso.pred)^2)))
```

```
## [1] "test MSE= 80.7309096231647"
```

```
out <- glmnet(Earnings.Per.Share ~ ., data=dat, alpha=1, lambda=grid)
lasso.coef <- predict(out, type="coefficients", s=lasso.cv$lambda.min, newdata=dat)
print(paste('The number of non-zero coefficient estimates is ', length(lasso.coef[lasso.coef!=0])))
```

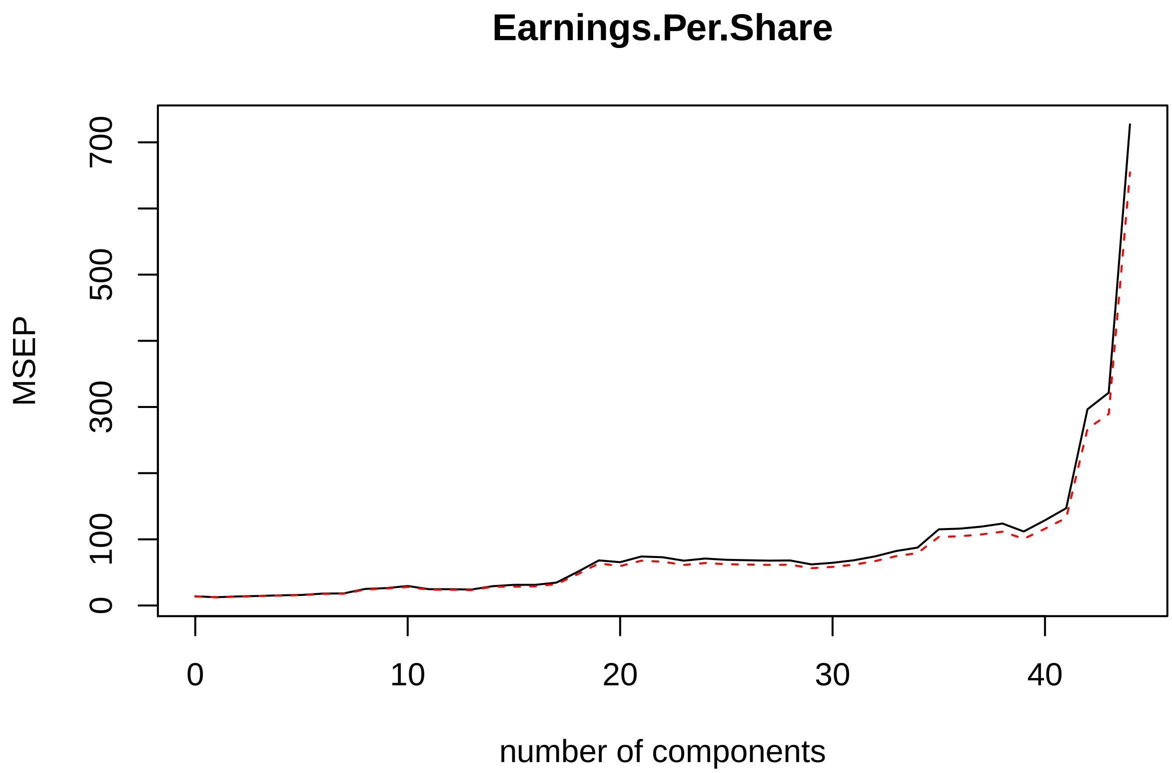
```
## <sparse>[ <logic> ] : .M.sub.i.logical() maybe inefficient
## [1] "The number of non-zero coefficient estimates is  4"
```

(f) Fit a PCR model on the training set, with M chosen by CV. Report the test error, along with the value of M selected by CV.

```
library(pls)
```

```
## Warning: package 'pls' was built under R version 3.4.4
##
## Attaching package: 'pls'
## The following object is masked from 'package:stats':
##
##     loadings
```

```
set.seed(435)
pcr.model <- pcr(Earnings.Per.Share ~ ., data=train, scale=TRUE, validation="CV")
validationplot(pcr.model, val.type="MSEP")
```



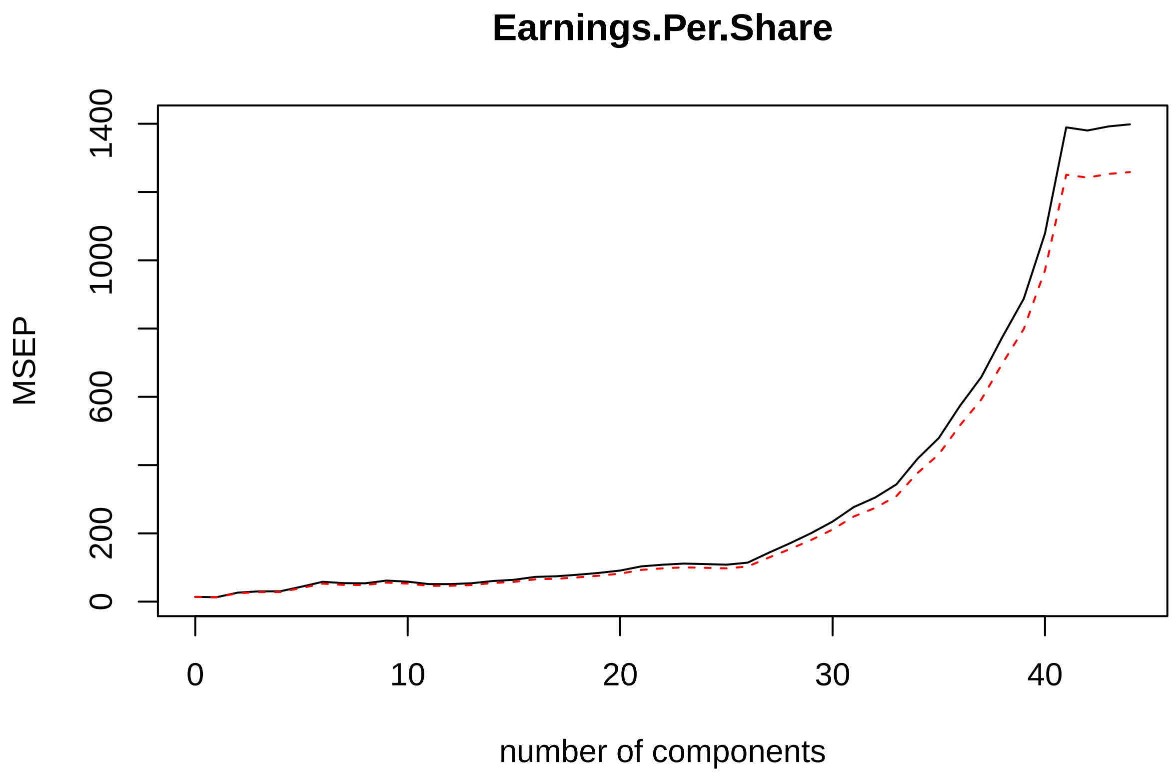
We find that the lowest CV error occurs when $M = 16$ component is used.

```
pcr.pred <- predict(pcr.model, test, ncomp=16)
print(paste('test MSE=', mean((test.eps-pcr.pred)^2)))
```

```
## [1] "test MSE= 43.7011653471072"
```

(g) Fit a partial least squares model on the training set, with M chonse by CV. Report the error along with the value of M.

```
set.seed(1)
pls.model <- plsr(Earnings.Per.Share ~ ., data=train, scale=TRUE, validation="CV")
validationplot(pls.model, val.type="MSEP")
```



We could find that the cross validation error is lowest at $M = 28$.

```
pls.pred <- predict(pls.model, test, ncomp=28)
print(paste('test MSE=', mean((test.eps-pls.pred)^2)))
```

```
## [1] "test MSE= 13.0109919502939"
```

(h) Comment on the result obtained.

As a result, the best model I got to the data set I used is partial least squares model, which only gives a test MSE of 13.01. This value is extremely small comparing to the MSE I got in the linear regression model that gives me a MSE of 5015.91. However, other models are also good that only have their MSE under 100. Among all models, I prefer partial least square model since it gives the best result.

Question 2

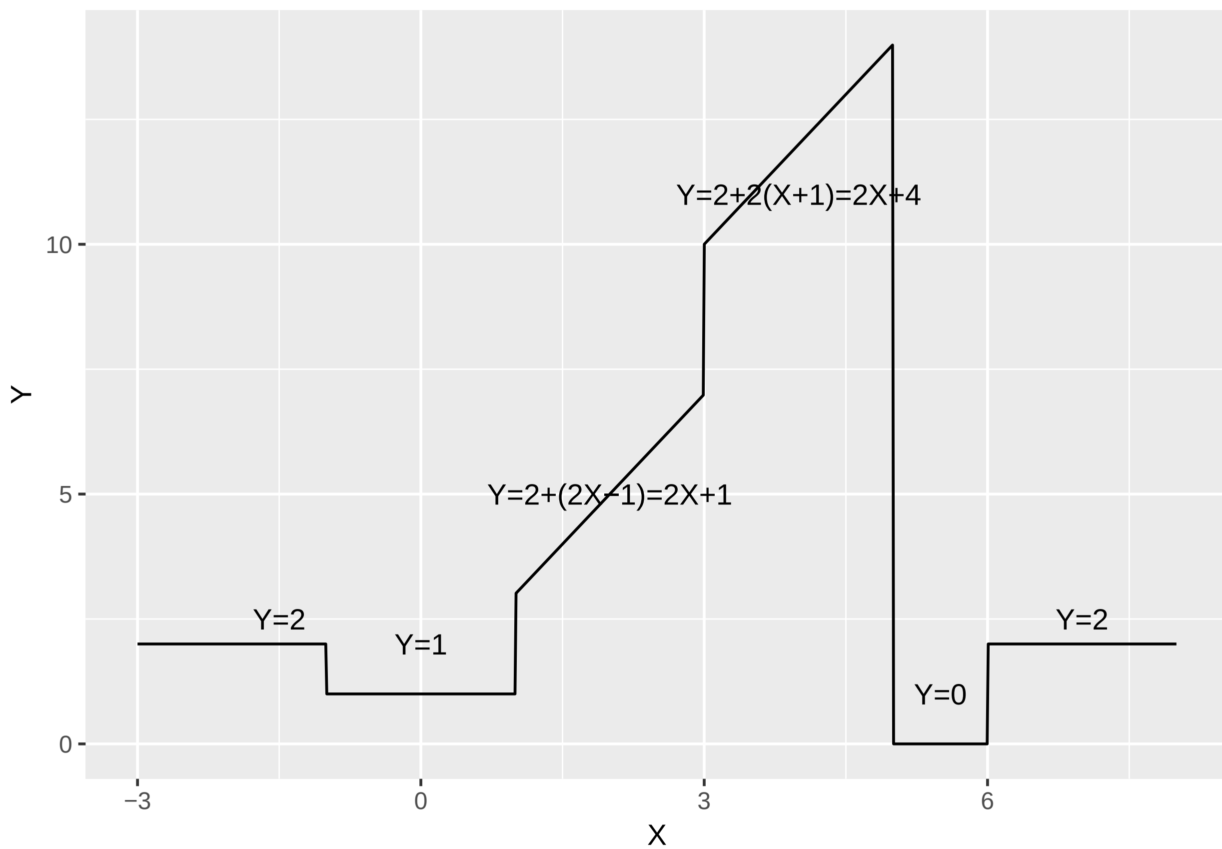
```
b1 <- function(x) {
  b1 <- 0
  if (x > -1 && x <= 1) {
    b1 <- b1 + 1
  }
  if (x > 1 && x <= 3) {
    b1 <- b1 - (2 * x - 1)
  }
  return(b1)
}
```

```r
b2 <- function(x) {
  b2 <- 0
  if (x > 3 && x <= 5) {
    b2 <- b2 + (x + 1)
  }
  if (x > 5 && x <= 6) {
    b2 <- b2 - 1
  }
  return(b2)
}
```

Sketch the estimated curve between $X = -3$ and $X = 8$.

```r
beta0 <- 2
beta1 <- -1
beta2 <- 2
X <- seq(-3, 8, length=1000)
Y <- rep(NA, 1000)
for (i in 1:1000) {
  Y[i] <- beta0 + beta1*b1(X[i]) + beta2*b2(X[i])
}
```

```r
ggplot() + geom_line(aes(X,Y)) +
  annotate("text", x=-1.5, y=2.5, label="Y=2") +
  annotate("text", x=0, y=2, label="Y=1") +
  annotate("text", x=2, y=5, label="Y=2+(2X-1)=2X+1") +
  annotate("text", x=4, y=11, label="Y=2+2(X+1)=2X+4") +
  annotate("text", x=5.5, y=1, label="Y=0") +
  annotate("text", x=7, y=2.5, label="Y=2")
```


Y=2+2(X+1)=2X+4
Y=2+(2X-1)=2X+1
Y=2
Y=1
Y=0
Y=2
10
5
0
Y
−3
0
3
6
X