Stat_101C_HW3

Junhyuk Jang

4/30/2017

SID: 004 728 134 DIS: 2A

```r
# install.packages("ISLR")
# install.packages("boot",dep = TRUE)
# install.packages("resample")
library("ISLR")
library("ggplot2")
library("boot")
```

```
## Warning: package 'boot' was built under R version 3.2.5
```

```r
library("resample")
require("boot")
attach(Carseats)
# Q1
# (a)
df <- Carseats
head(df)
```

```
##   Sales CompPrice Income Advertising Population Price ShelveLoc Age
## 1  9.50       138     73          11        276   120       Bad  42
## 2 11.22       111     48          16        260    83      Good  65
## 3 10.06       113     35          10        269    80    Medium  59
## 4  7.40       117    100           4        466    97    Medium  55
## 5  4.15       141     64           3        340   128       Bad  38
## 6 10.81       124    113          13        501    72       Bad  78
##   Education Urban  US
## 1        17   Yes Yes
## 2        10   Yes Yes
## 3        12   Yes Yes
## 4        14   Yes Yes
## 5        13   Yes  No
## 6        16    No Yes
```

```r
summary(df)
```

```
##      Sales          CompPrice       Income        Advertising
##  Min.   : 0.000   Min.   : 77   Min.   : 21.00   Min.   : 0.000
##  1st Qu.: 5.390   1st Qu.:115   1st Qu.: 42.75   1st Qu.: 0.000
##  Median : 7.490   Median :125   Median : 69.00   Median : 5.000
##  Mean   : 7.496   Mean   :125   Mean   : 68.66   Mean   : 6.635
##  3rd Qu.: 9.320   3rd Qu.:135   3rd Qu.: 91.00   3rd Qu.:12.000
##  Max.   :16.270   Max.   :175   Max.   :120.00   Max.   :29.000
##    Population        Price        ShelveLoc        Age
##  Min.   : 10.0   Min.   : 24.0   Bad   : 96   Min.   :25.00
```

```
##  1st Qu.:139.0   1st Qu.:100.0   Good  : 85   1st Qu.:39.75
##  Median :272.0   Median :117.0   Medium:219   Median :54.50
##  Mean   :264.8   Mean   :115.8                Mean   :53.32
##  3rd Qu.:398.5   3rd Qu.:131.0                3rd Qu.:66.00
##  Max.   :509.0   Max.   :191.0                Max.   :80.00
##    Education    Urban       US
##  Min.   :10.0   No :118   No :142
##  1st Qu.:12.0   Yes:282   Yes:258
##  Median :14.0
##  Mean   :13.9
##  3rd Qu.:16.0
##  Max.   :18.0
```

```r
dim(df)
```

```
## [1] 400  11
```

```r
summary(df$Sales)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   5.390   7.490   7.496   9.320  16.270
```

```r
summary(df$CompPrice)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      77     115     125     125     135     175
```

```r
summary(df$Income)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   21.00   42.75   69.00   68.66   91.00  120.00
```

```r
summary(df$Advertising)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   0.000   5.000   6.635  12.000  29.000
```

```r
summary(df$Population)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    10.0   139.0   272.0   264.8   398.5   509.0
```

```r
summary(df$Price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    24.0   100.0   117.0   115.8   131.0   191.0
```

```r
summary(df$ShelveLoc)
```

```
##    Bad   Good Medium
##     96     85    219
```

```r
summary(df$Age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   25.00   39.75   54.50   53.32   66.00   80.00
```

```r
summary(df$Education)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    10.0    12.0    14.0    13.9    16.0    18.0
```

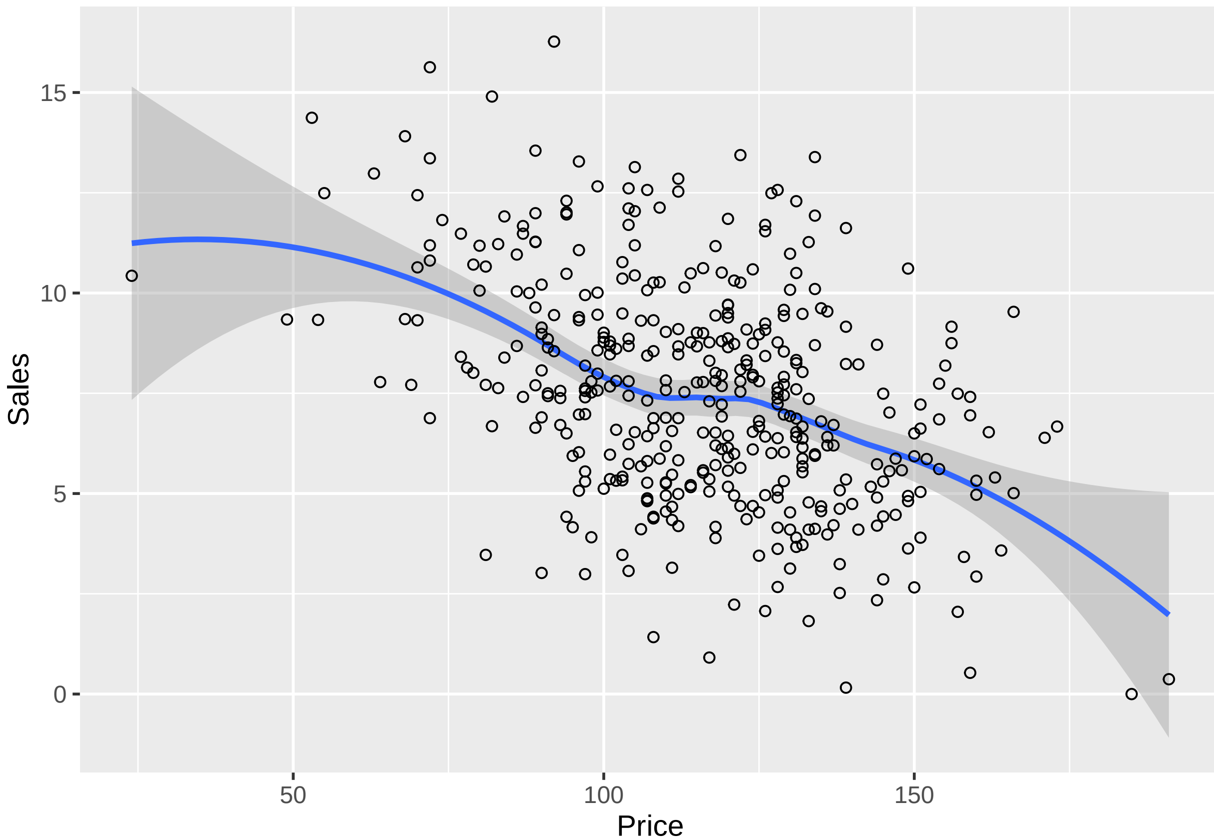
```r
summary(df$Urban)
```

```
##  No Yes
## 118 282
```

```r
summary(df$US)
```

```
##  No Yes
## 142 258
```

```r
# (b)
ggplot(df,aes(x = Price,y = Sales)) + geom_smooth() +
        geom_point(shape = 1)
```




```
# What did you notice?
# As price increases the sale decreaces monotonically.

# (c)  & (d)
#Basic confidence interval
my.mean <- function(data,indices){
  d=data[indices]
  mean(d)
}
my.median <- function(data,indices){
  d=data[indices]
  median(d)
}
(out.bs.mean <- boot(data = df$Sales,statistic = my.mean, R = 1000))
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = df$Sales, statistic = my.mean, R = 1000)
##
##
## Bootstrap Statistics :
##     original     bias    std. error
## t1* 7.496325 0.0016441   0.1390179
```

```r
(out.bs.median <- boot(data = df$Sales,statistic = my.median,R = 1000))
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = df$Sales, statistic = my.median, R = 1000)
##
##
## Bootstrap Statistics :
##     original     bias    std. error
## t1*     7.49 -0.039975   0.1751424
```

```r
(se.mean <- sd(out.bs.mean$t))
```

```
## [1] 0.1390179
```

```r
(se.median <- sd(out.bs.median$t))
```

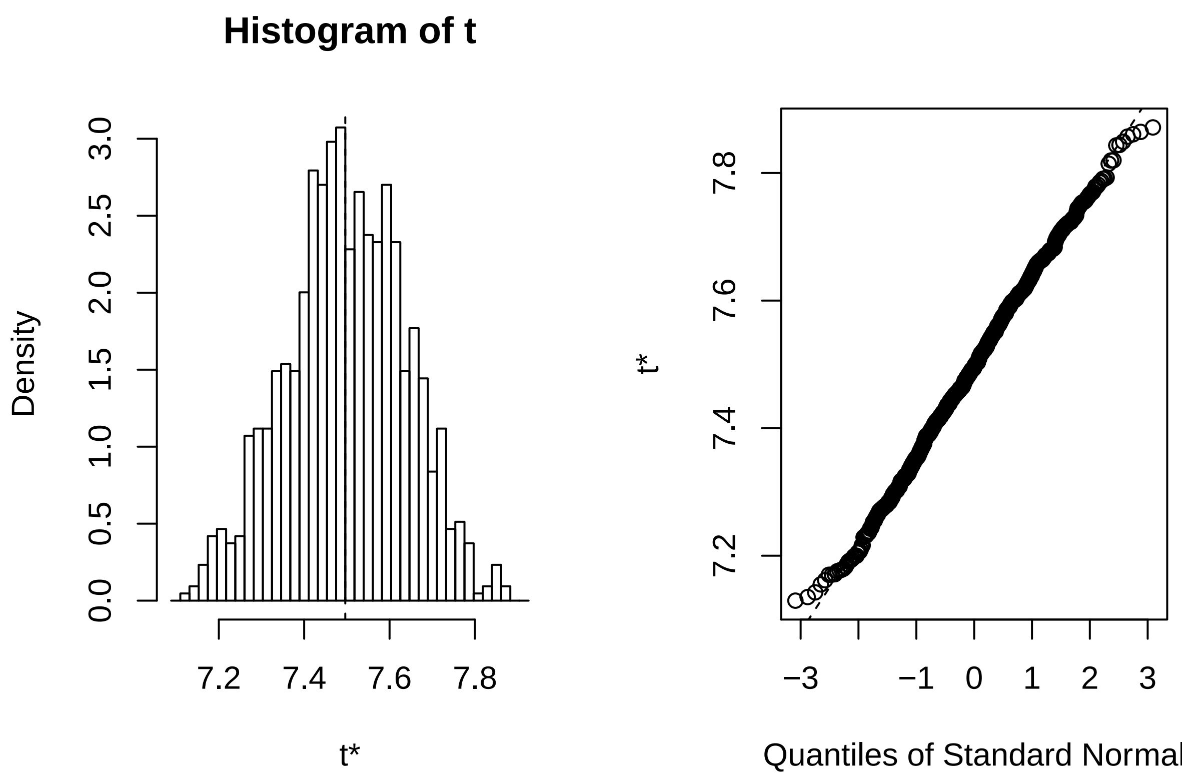
```
## [1] 0.1751424
```

```r
# CI mean & plot
boot.ci(out.bs.mean)
```

```
## Warning in boot.ci(out.bs.mean): bootstrap variances needed for studentized
## intervals
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = out.bs.mean)
##
## Intervals :
## Level      Normal              Basic
## 95%   ( 7.222,  7.767 )   ( 7.231,  7.782 )
##
## Level     Percentile            BCa
## 95%   ( 7.210,  7.762 )   ( 7.216,  7.764 )
## Calculations and Intervals on Original Scale
```

```r
plot(out.bs.mean) # normally distributed.
```

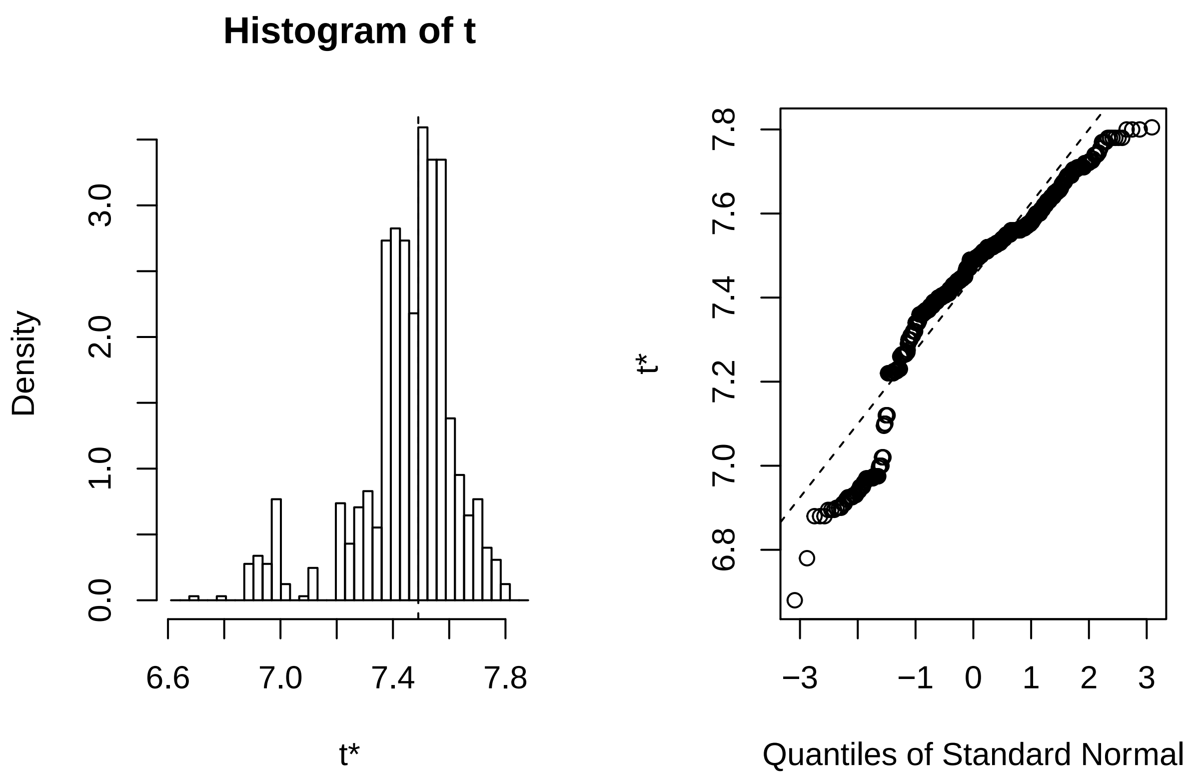



```r
# CI median & plot
boot.ci(out.bs.median)
```

```
## Warning in boot.ci(out.bs.median): bootstrap variances needed for
## studentized intervals
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = out.bs.median)
##
## Intervals :
## Level      Normal              Basic
## 95%   ( 7.187,  7.873 )   ( 7.260,  8.030 )
##
## Level     Percentile            BCa
## 95%   ( 6.950,  7.720 )   ( 6.929,  7.710 )
## Calculations and Intervals on Original Scale
```

```r
plot(out.bs.median) # not normally distributed
```




```
# Q2
# (A)
set.seed(77)
train <- sample(400,280)
training_mse<-c()
MSE_training <- function(y,x){
   for(i in 1:9){
    lm.fit<-lm(formula=y~poly(x,i,raw=T),data=df,subset = train)
  training_mse[i]<- mean((y-predict(lm.fit,df))[train]^2)
  }
  return(training_mse)
}
MSE_training(Sales,Price)
```

```
## [1] 6.471708 6.459979 6.453792 6.272033 6.260994 6.241209 6.176282 6.176111
## [9] 6.176103
```

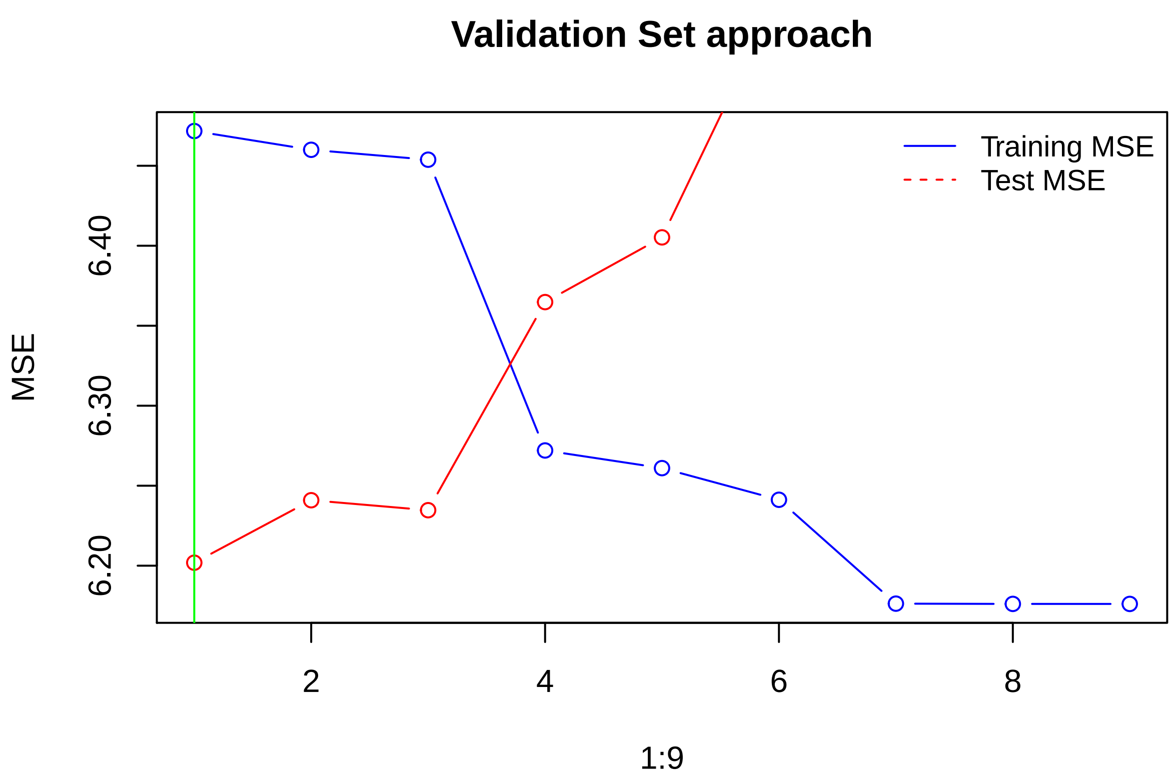
```
# (B)
set.seed(77)
testing_mse<-c()
MSE_testing <- function(y,x){
   for(i in 1:9){
    lm.fit<-lm(formula=y~poly(x,i,raw=T),data=df,subset = train)
  testing_mse[i]<- mean((y-predict(lm.fit,df))[-train]^2)
  }
  return(testing_mse)
}
MSE_testing(Sales,Price)
```

```
## [1] 6.201862 6.240919 6.234695 6.364783 6.405235 6.557303 6.490667 6.498849
## [9] 6.507095
```

```r
(mse_test <- MSE_testing(Sales,Price))
```

```
## [1] 6.201862 6.240919 6.234695 6.364783 6.405235 6.557303 6.490667 6.498849
## [9] 6.507095
```

```r
# (C)
plot(1:9,MSE_training(Sales,Price),type = "b",col = "blue",ylab = "MSE",
     main = "Validation Set approach")
points(1:9,mse_test,col = "red",type = "b")
legend("topright", legend=c("Training MSE","Test MSE"),
       col=c("blue", "red"), lty=1:2, cex=0.88,
       box.lty=0)
abline(v = which.min(MSE_testing(Sales,Price)),col = "green")
```

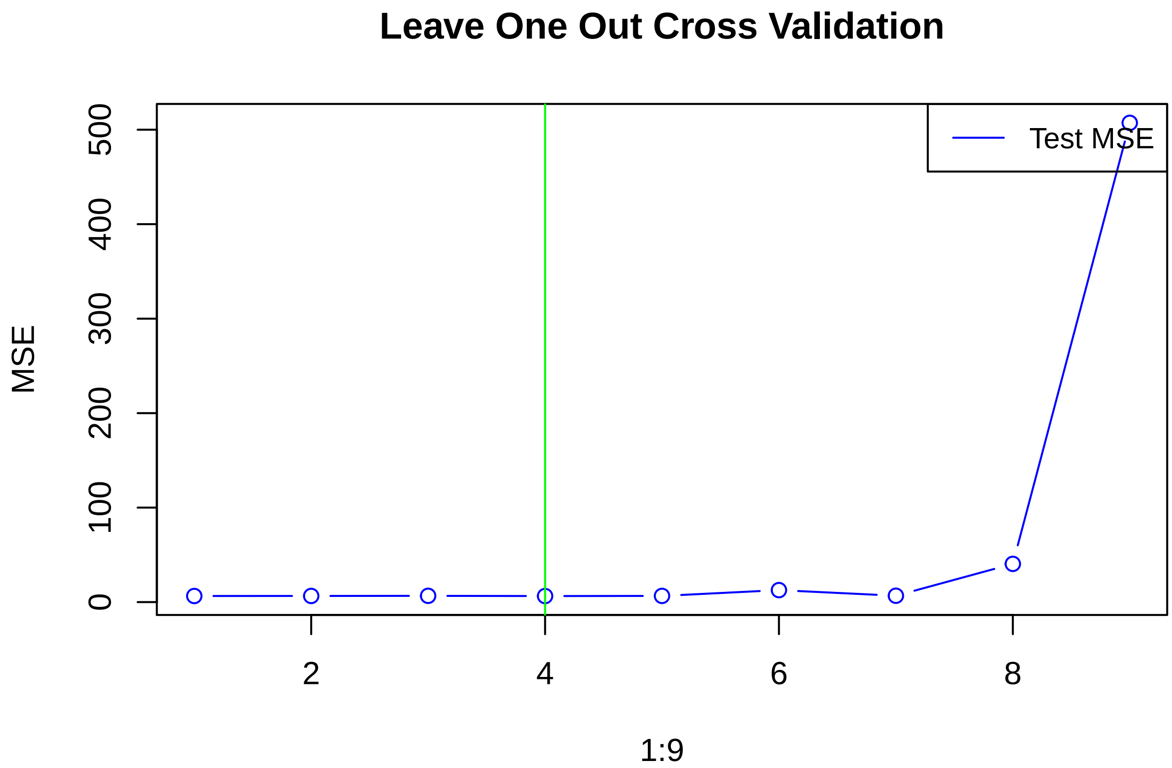



```r
# INTERPRETATION
# Based on Test MSE, polynomial degree of 1 minimizes the MSE.

# Q3
cv.error1 <- rep(0,9)
for (i in 1:9) {
       glm <- glm(Sales ~ poly(Price,i),data = df)
       cv.error1[i] <- cv.glm(df,glm)$delta[1]
}
cv.error1
```

```
## [1]   6.444262   6.483740   6.623883   6.386824   6.533110  12.680164
## [7]   6.712122  40.436400 507.238051
```

```
# plot & interpretation
plot(1:9,cv.error1,type = "b",col = "blue",ylab = "MSE",
     main = "Leave One Out Cross Validation")
legend("topright", legend="Test MSE",
       col= "blue", lty=1:1, cex=0.88,
       box.lty=1)
abline(v = which.min(cv.error1),col = "green")
```




```
# INTERPRETATION
# Based on Test MSE, polynomial degree of 4 minimizes the MSE.

# Q4
# (a)
set.seed(77)
# split k = 10
a <- split(sample(1:400),f=rep(1:10,400))
```

```
## Warning in split.default(sample(1:400), f = rep(1:10, 400)): data length is
## not a multiple of split variable
```

```
a1 <- a[[1]]
a2 <- a[[2]]
head(df[a1,])
```

```
##     Sales CompPrice Income Advertising Population Price ShelveLoc Age
```

```
## 117  5.08       135     75           0        202   128    Medium  80
## 341  7.50       140     29           0        105    91       Bad  43
## 1    9.50       138     73          11        276   120       Bad  42
## 268  5.83       134     82           7        473   112       Bad  51
## 144  0.53       122     88           7         36   159       Bad  28
## 194 13.28       139     70           7         71    96      Good  61
##     Education Urban  US
## 117        10    No  No
## 341        16   Yes  No
## 1          17   Yes Yes
## 268        12    No Yes
## 144        17   Yes Yes
## 194        10   Yes Yes
```

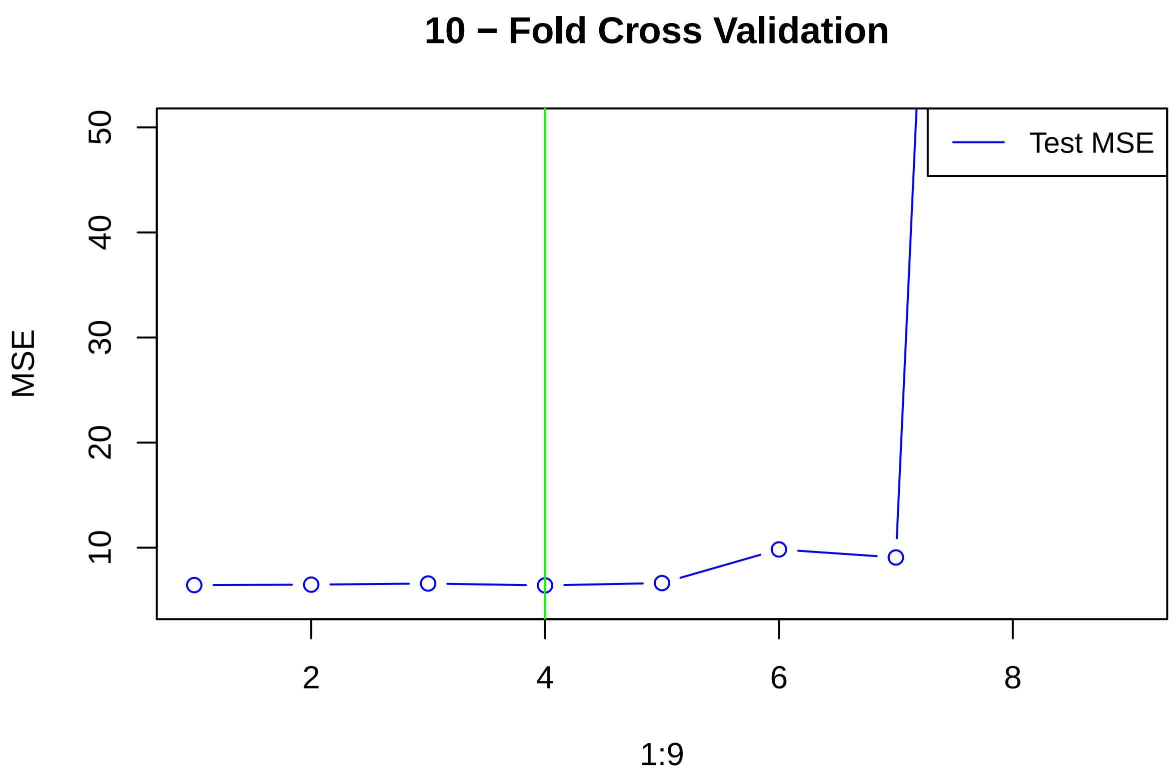
```
head(df[a2,])
```

```
##     Sales CompPrice Income Advertising Population Price ShelveLoc Age
## 287  7.53       117    118          11        429   113    Medium  67
## 385 12.85       123     37          15        348   112      Good  28
## 393  4.53       129     42          13        315   130       Bad  34
## 232  8.09       132     69           0        123   122    Medium  27
## 59   5.42       103     93          15        188   103       Bad  74
## 394  5.57       109     51          10         26   120    Medium  30
##     Education Urban  US
## 287        18    No Yes
## 385        12   Yes Yes
## 393        13   Yes Yes
## 232        11    No  No
## 59         16   Yes Yes
## 394        17    No Yes
```

```
# (b)
set.seed(77)
cv.error.10 <- NULL
for (i in 1:9) {
        glm <- glm(Sales ~ poly(Price,i),data = df)
        cv.error.10[i] <- cv.glm(df,glm,K = 10)$delta[1]
}
cv.error.10
```

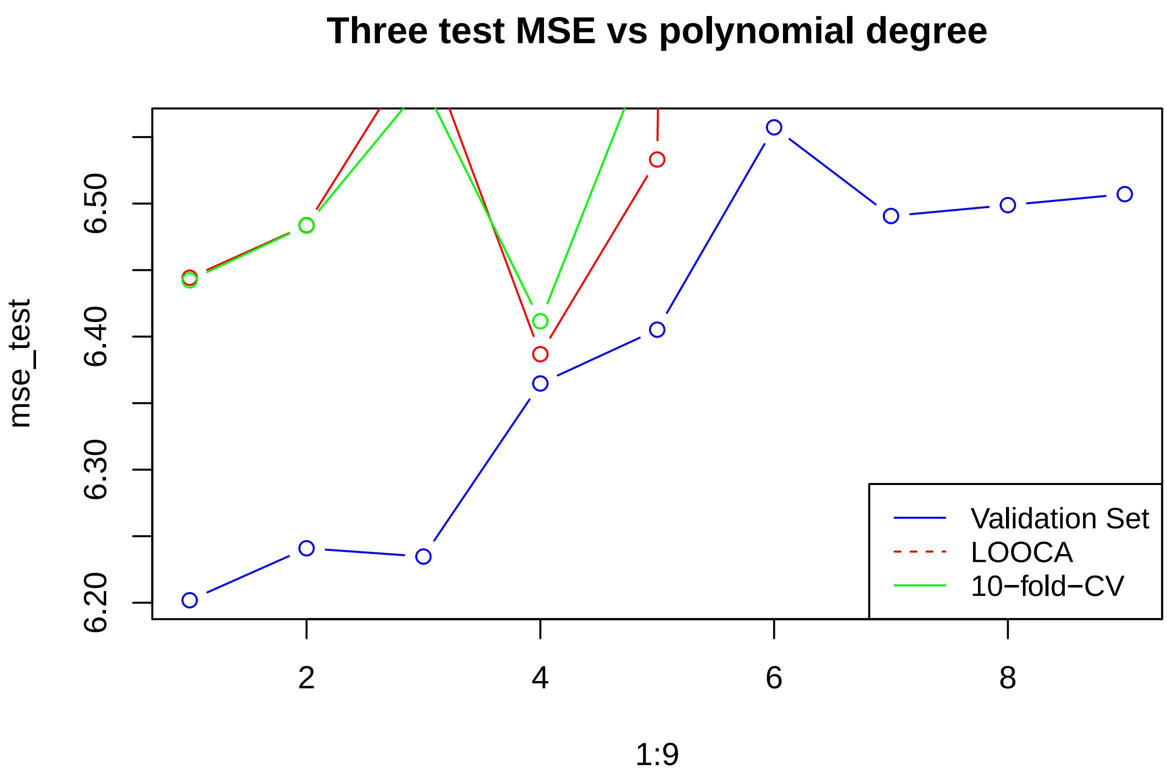
```
## [1]   6.442357   6.483433   6.589914   6.411529   6.633696   9.833213
## [7]   9.070080 250.948685 378.487707
```

```
plot(1:9,cv.error.10,type = "b",col = "blue",ylab = "MSE",
     main = "10 - Fold Cross Validation ",ylim = c(5,50))
legend("topright", legend="Test MSE",
       col= "blue", lty=1:1, cex=0.88,
       box.lty=1)
abline(v = which.min(cv.error.10),col = "green")
```




```r
# INTERPRETATION WHO IS THE BEST?
# Based on Test MSE, polynomial degree of 4 minimizes the MSE.

# Q5
plot(1:9,mse_test,type = "b",col = "blue",main = "Three test MSE vs polynomial degree")
points(1:9,cv.error1,col = "red",type = "b")
points(1:9,cv.error.10,col = "green",type = "b")
legend("bottomright", legend=c("Validation Set","LOOCA","10-fold-CV"),
       col=c("blue", "red","Green"), lty=1:2, cex=0.88,
       box.lty=1)
```




```
# INTERPRETATION
# Based on the plot three test Mse vs polynomial degree,
# we can see polynomial degree of 1 and 4 are outperforming than the other
# polynomial degrees.If I have to choose one polynomial degree to fit my model,
# I will choose degree of 4 derived from 10-fold cross validation because
# leave on out cross validation method is averaging the output of n fitted model
# ,hence, outputs are highly correlated each. In other words, LOOCV have higher
# variance than 10-fold CV. In case of the validation set approach, it has two
# crucial drawbacks. Firstly, error rate can be highly variate depending on
# which observations are included in the training set and which observations are
# included in the testing set. Secondly, it has higher risk to overestimate testing
# error because we split our data into training and testing which implies that less
# observations are used to make our fitted model.
# For these reasons,I believe making model with polynomial degree of 4 would give us
# the best prediction model.
```