DS502 - HW4
Vandana Anand
Kratika Agrawal

Question 1:

a) **iii.** The lasso, relative to least squares, is Less flexible and hence will give improved prediction accuracy when its increase in bias is less than its decrease in variance.
This is because Lasso, relative to least squares, is less flexible and thus avoids overfitting the model. It means, when least squares yield very high variance, lasso can yield a small variance by slightly increasing the bias and thus avoiding overfitting and producing better prediction accuracy.

b) **iii.** Ridge regression methods also exhibit the similar behaviour as Lasso relative to least squares. Ridge regression also is less flexible relative to least squares and when its increase in bias is slightly less than the increase in variance, ridge regression can yield more accurate predictions.

c) **ii.** Non-linear methods are more flexible relative to least squares and hence non-linear methods generally have relatively higher variance and smaller bias. So, with its increase in variance less than its decrease in bias, results in a much stable model that gives an improved prediction accuracy.

Question 2:

a)

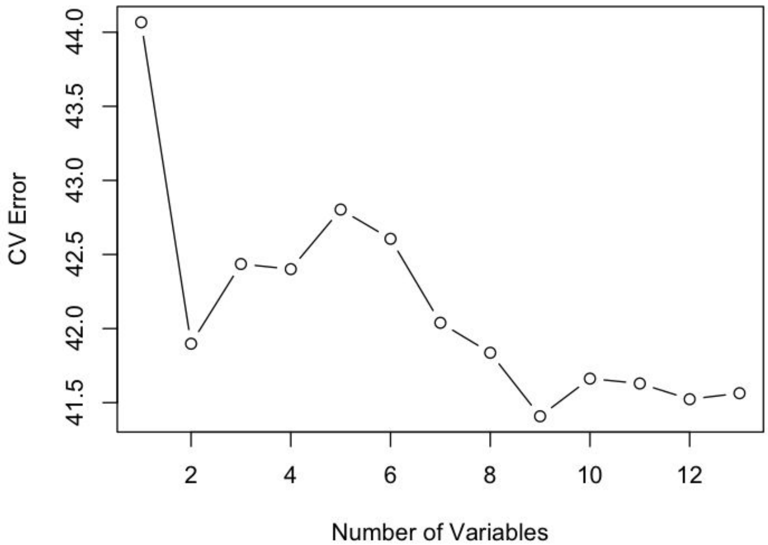
```
#Best Subset selection

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
folds=sample(1:k,nrow(Boston), replace=TRUE)
cv.errors=matrix(NA,k,13, dimnames=list(NULL,paste(1:13)))

for(i in 1:k){
    best.fit=regsubsets(crim~.,data=Boston[folds != i, ],nvmax=13)
    for(j in 1:13){
        pred=predict(best.fit,Boston[folds == i, ], id=j)
        cv.errors[i,j]=mean((Boston$crim[folds == i] -pred)^2)
    }
}

mean.cv.errors = apply(cv.errors,2,mean)
plot(mean.cv.errors, type="b", xlab= "Number of Variables", ylab="CV Error")
mean.cv.errors[which.min(mean.cv.errors)]
```

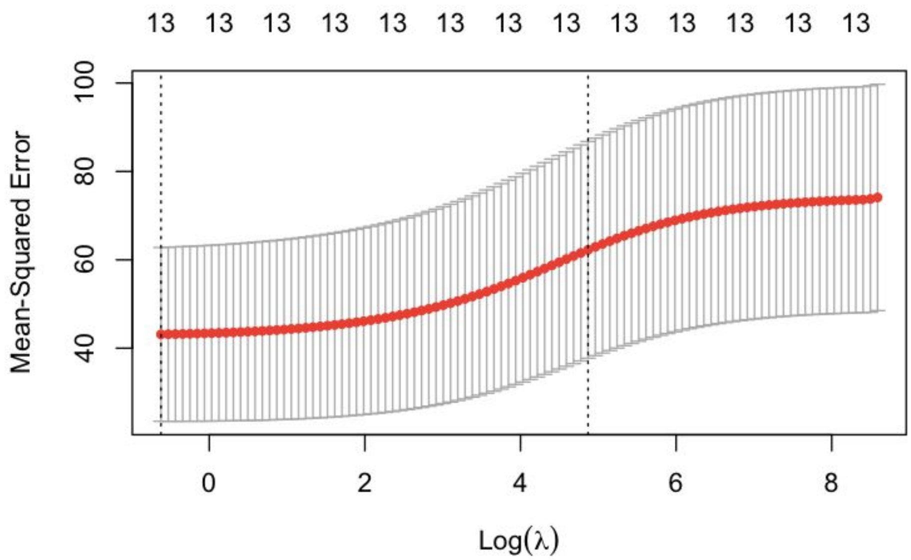


Cross Validation chooses a 9-variables model and the CV estimate for the test MSE is 41.40812.

```
#Ridge Regression
x=model.matrix(crim~.,Boston)[,-1]
y=Boston$crim
cv.ridge = cv.glmnet(x,y,alpha=0,type.measure="mse")
plot(cv.ridge)
cv.ridge
```

```
Measure: Mean-Squared Error

    Lambda Measure    SE Nonzero
min   0.54   43.13 19.66      13
1se 130.08   62.25 24.52      13
```

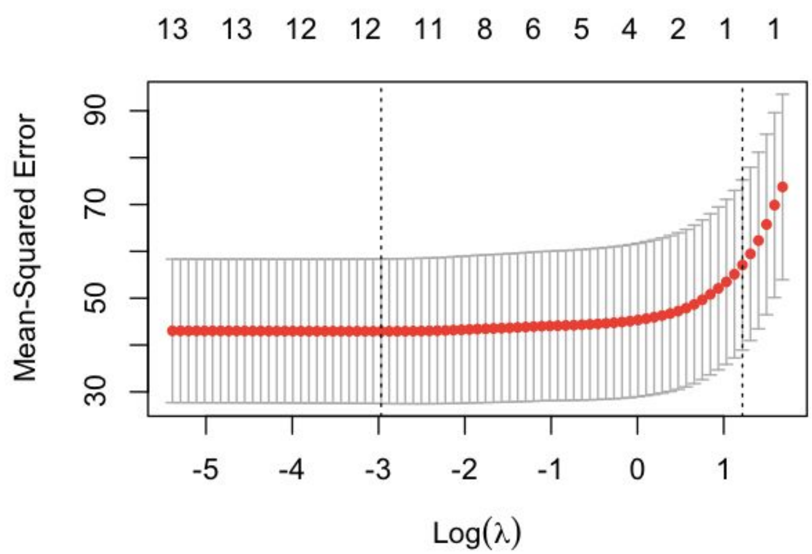


Cross validation chooses lambda to equal 0.54 and the CV estimate for the test MSE is 43.13.

```
#Lasso
cv.lasso = cv.glmnet(x,y,alpha=1,type.measure="mse")
plot(cv.lasso)
cv.lasso
```

```
Measure: Mean-Squared Error

    Lambda Measure    SE Nonzero
min  0.051   42.94 15.45      11
1se  3.376   57.09 18.17       1
```



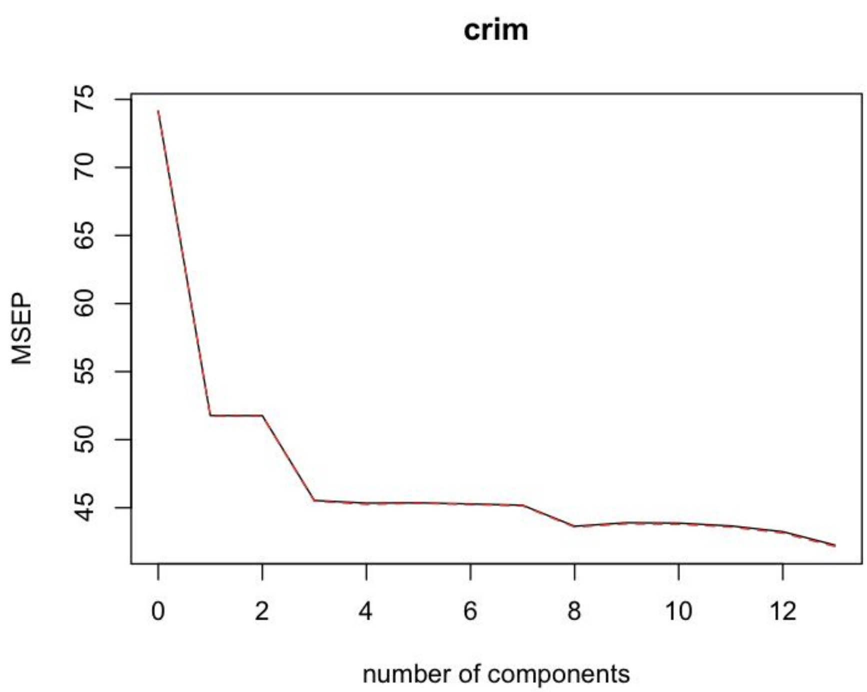
Cross validation chooses lambda to equal 0.051 and the CV estimate for the test MSE is 42.94.

```
#PCR
pcr.fit=pcr(crim~., data=Boston, scale=TRUE, validation="CV")
summary(pcr.fit)
validationplot(pcr.fit,val.type="MSEP")
```

```
> summary(pcr.fit)
Data:     X dimension: 506 13
     Y dimension: 506 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  7 comps  8 comps  9 comps
CV            8.61    7.195    7.195    6.748    6.734    6.735    6.729    6.722    6.606    6.626
adjCV         8.61    7.193    7.192    6.744    6.727    6.732    6.726    6.718    6.601    6.620
       10 comps  11 comps  12 comps  13 comps
CV        6.623     6.608     6.576     6.501
adjCV     6.617     6.602     6.568     6.493

TRAINING: % variance explained
      1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps  9 comps  10 comps
X       47.70    60.36    69.67    76.45    82.99    88.00    91.14    93.45    95.40     97.04
crim    30.69    30.87    39.27    39.61    39.61    39.86    40.14    42.47    42.55     42.78
      11 comps  12 comps  13 comps
X        98.46     99.52     100.0
crim     43.04     44.13      45.4
```



Cross validation chooses M to equal 14, so there is no dimension reduction. The CV estimate for the test MSE is 45.4.

b) Using the cross validation errors from the models in part (a), best subset selection seems to work the best with the lowest error of 41.40812. Next is ridge regression and lasso, which have similar test MSEs that are 43.13 and 42.94, respectively. PCR has the highest test MSE at 45.4.

c) No, best subset selection uses a 9-variable model so not all predictors are used.

Question 3:

$Y = 1 + x - 2(x-1)^2 I(x \geq 1) + \varepsilon$ Equation

$Y = 1 + -2 - 2(-2-1)^2 \cdot 0 + \varepsilon$ Insert -2

$= -1 \rightarrow (-2, -1)$

$Y = 1 - 1 - 2(-1-1)^2 \cdot 0 + \varepsilon$ Insert -1

$= 0 \rightarrow (-1, 0)$

$Y = 1 + 0 - 2(0-1)^2 \cdot 0 + \varepsilon$ Insert 0

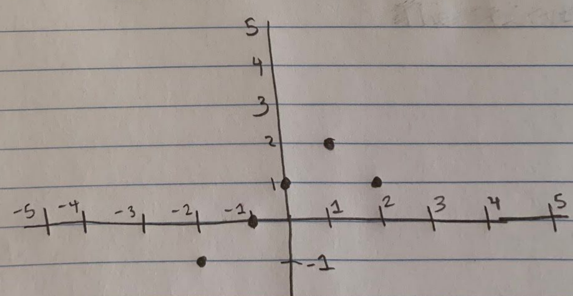
$= 1 \rightarrow (0, 1)$

$Y = 1 + 1 - 2(1-1)^2 \cdot 1 + \varepsilon$ Insert 1

$= 2 \rightarrow (1, 2)$

$Y = 1 + 2 - 2(2-1)^2 \cdot 1 + \varepsilon$ Insert 2

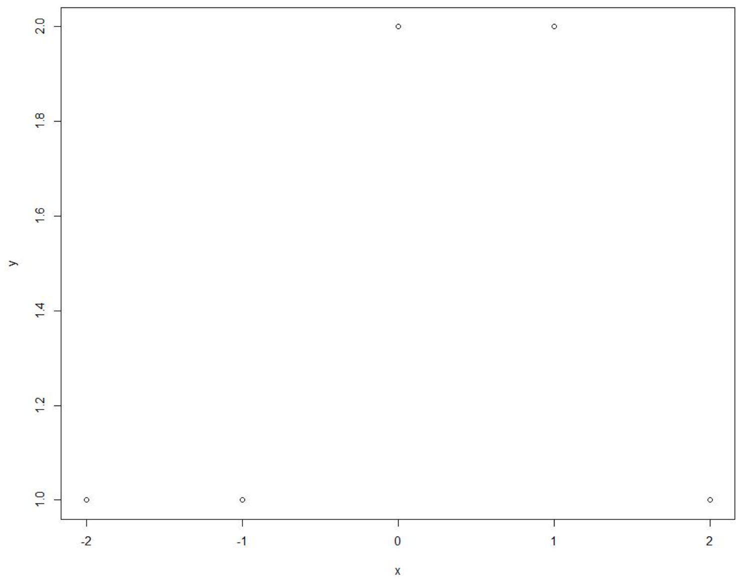
$= 1 \rightarrow (2, 1)$



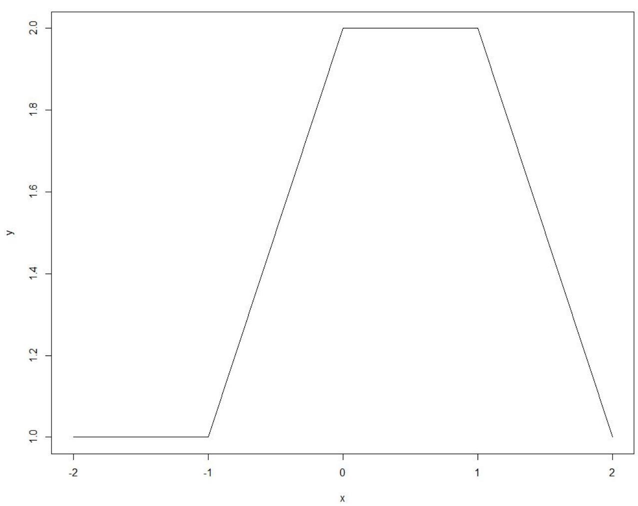
The curve is linear from x= -2 to 1, Y=1+x, and quadratic from x = 1 to 2, Y = 1+x-2(x-1)^2.

Question 4:

```
> x = -2:2
> beta0 = 1
> beta1 = 1
> beta2 = 3
>
> y = (beta0) + beta1*(I(0<=x & x<=2)-(x-1)*I(1<=x & x<=2)) + beta2*((x-3)*I(3<=x & x<=4)+I(4<x & x<=5))
> plot(x,y)
```



Joining the points: `> plot(x,y,type='l')`, gives us this plot:



Thus, we can clearly see that
y is constant at 1 for x = -2 to -1
y is constant at 2 for x = 0 to 1
y is linear with y = x+2 for x = -1 to 0
y is linear with y = 3-x for x = 1 to 2

Question 5:

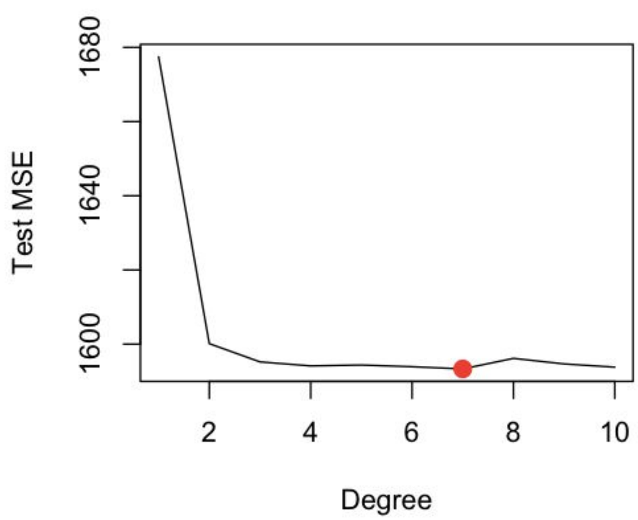
a) K-fold cross validation using K = 10:

```r
deltaVar = rep(NA, 10)
for (i in 1:10){
    fit = glm(wage~poly(age,i), data=Wage)
    deltaVar[i] = cv.glm(Wage, fit, K=10)$delta[1]
}

plot(1:10, deltaVar, xlab="Degree", ylab="Test MSE", type="l")
d.min=which.min(deltaVar)
points(which.min(deltaVar), deltaVar[which.min(deltaVar)], col="red", cex=2,
 pch=20)


fit1 = lm(wage~age, data=Wage)
fit2 = lm(wage~poly(age, 2), data=Wage)
fit3 = lm(wage~poly(age, 3), data=Wage)
fit4 = lm(wage~poly(age, 4), data=Wage)
fit5 = lm(wage~poly(age, 5), data=Wage)
anova(fit1,fit2,fit3,fit4,fit5)


plot(wage~age, data=Wage, col="black")
ageRange=range(Wage$age)
ages=seq(from=ageRange[1], to=ageRange[2])
fit=lm(wage~poly(age,3), data=Wage)
prediction=predict(fit, newdata=list(age=ages))
lines(ages, prediction, col="red", lwd=2)
```

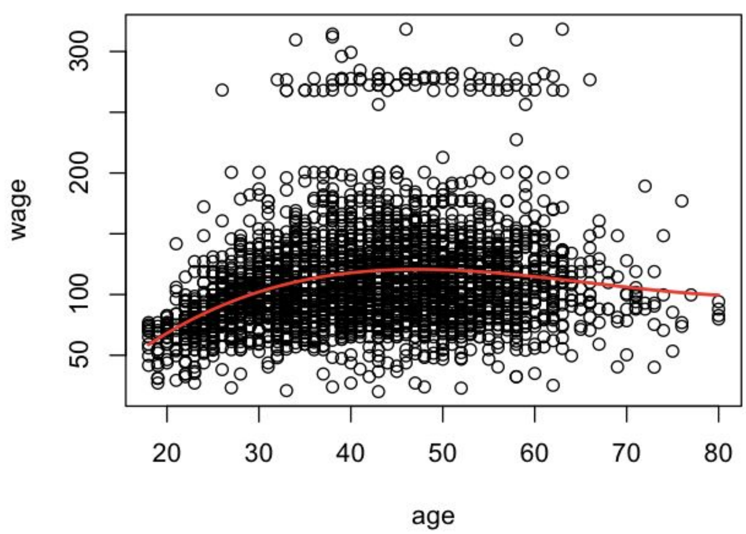


From the graph above, we can see that a degree of 7 is the optimal degree for the polynomial. Using ANOVA, as shown below, we can test the null hypothesis:

```
Analysis of Variance Table

Model 1: wage ~ age
Model 2: wage ~ poly(age, 2)
Model 3: wage ~ poly(age, 3)
Model 4: wage ~ poly(age, 4)
Model 5: wage ~ poly(age, 5)
  Res.Df     RSS Df Sum of Sq        F    Pr(>F)
1   2998 5022216
2   2997 4793430  1    228786 143.5931 < 2.2e-16 ***
3   2996 4777674  1     15756   9.8888  0.001679 **
4   2995 4771604  1      6070   3.8098  0.051046 .
5   2994 4770322  1      1283   0.8050  0.369682
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

The p values tell us that a cubic or quartic appears to fit the data reasonably, but lower or higher order models do not work. So we can use the following graph:

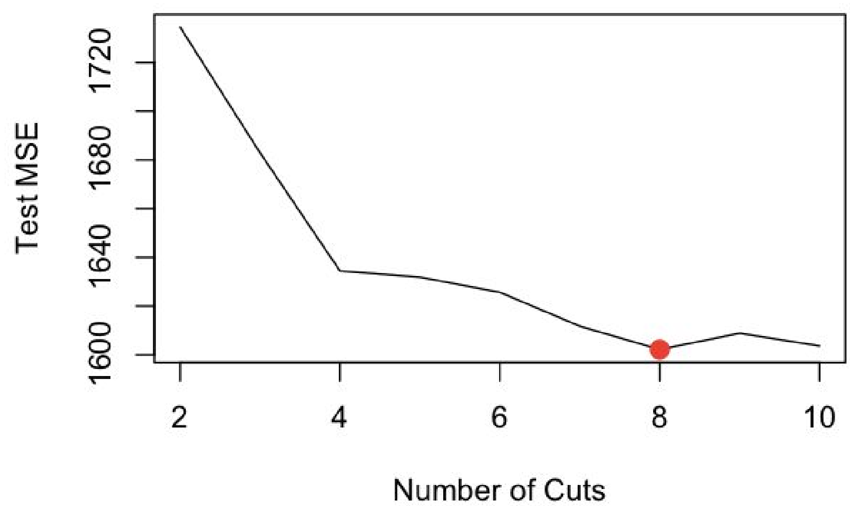


b) K-fold cross validation using K = 10:

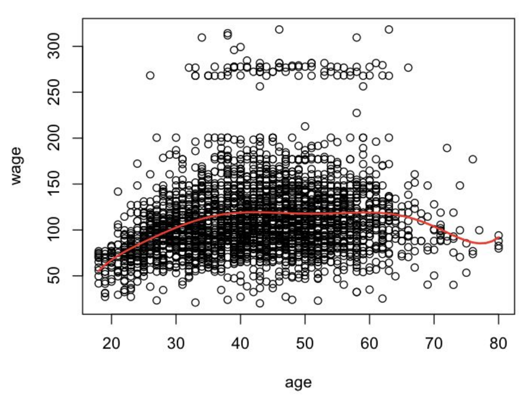
```r
deltaVar2 = rep(NA, 10)
for (i in 2:10){
    Wage$ageCutoff=cut(Wage$age, i)
    fit = glm(wage~ageCutoff, data=Wage)
    deltaVar2[i] = cv.glm(Wage, fit, K=10)$delta[1]
}

plot(2:10, deltaVar2[-1], xlab="Number of Cuts", ylab="Test MSE", type="l")
d.min=which.min(deltaVar2)
points(which.min(deltaVar2), deltaVar2[which.min(deltaVar2)], col="red",
 cex=2, pch=20)

plot(wage~age, data=Wage, col="black")
ageRange2=range(Wage$age)
ages2=seq(from=ageRange2[1], to=ageRange2[2])
fit=glm(wage~poly(age,8), data=Wage)
prediction=predict(fit, data.frame(age=ages2))
lines(ages2, prediction, col="red", lwd=2)
```



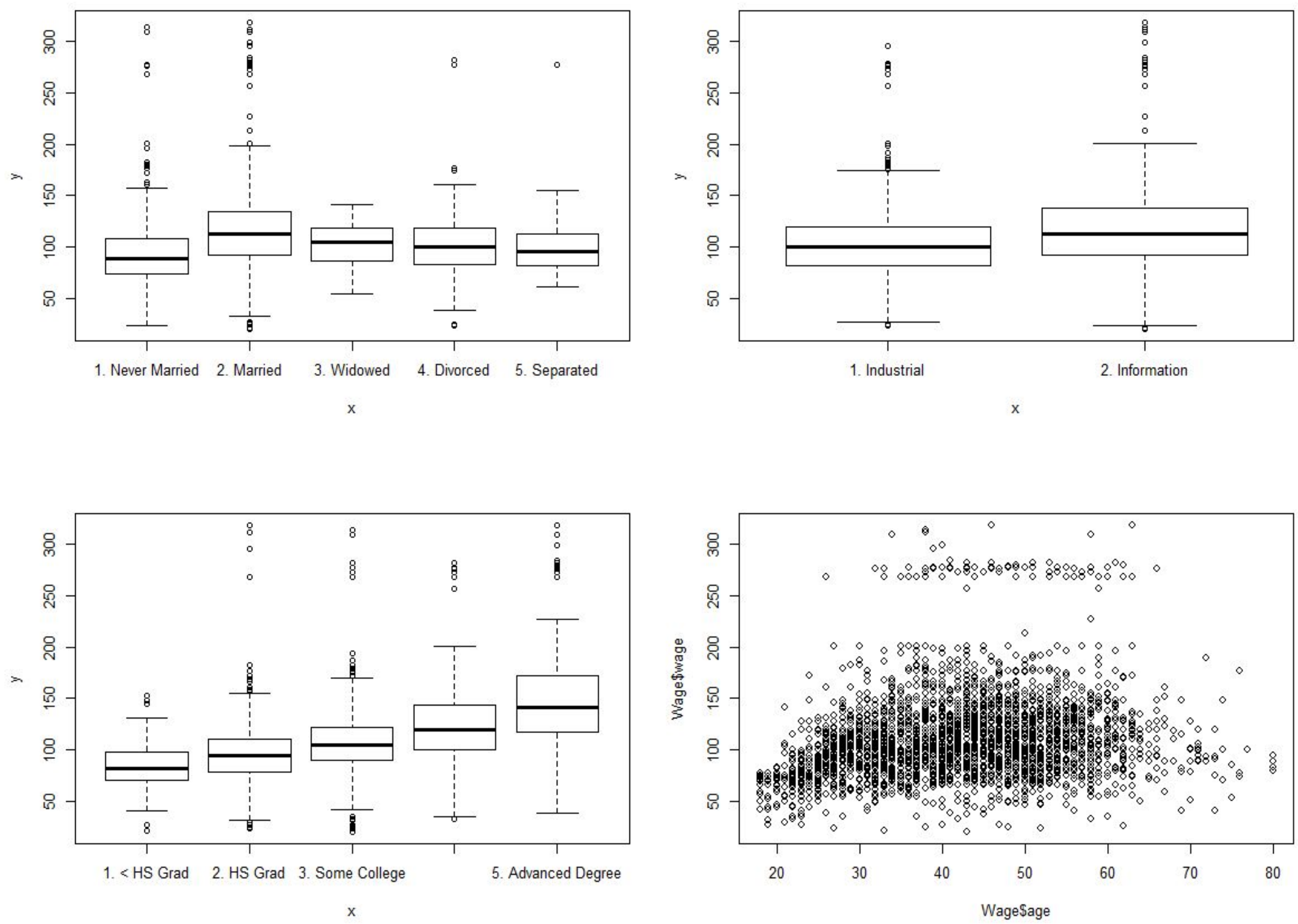
The error is lowest when the number of cuts is 8. We can fit the entire data with a step function using these 8 cuts:



Question 6

```
> library(ISLR)
>
> summary(Wage$age)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  18.00   33.75   42.00   42.41   51.00   80.00 
> summary(Wage$maritl)
1. Never Married       2. Married       3. Widowed      4. Divorced     5. Separated 
             648             2074               19              204               55 
> summary(Wage$jobclass)
 1. Industrial 2. Information 
          1544           1456 
> summary(Wage$race)
1. White 2. Black 3. Asian 4. Other 
    2480      293      190       37 
> summary(Wage$education)
      1. < HS Grad         2. HS Grad    3. Some College    4. College Grad 5. Advanced Degree 
               268                971                650                685                426 
> summary(Wage$health)
     1. <=Good 2. >=Very Good 
           858           2142 

> par(mfrow = c(2,2))
> plot(Wage$maritl, Wage$wage)
> plot(Wage$jobclass, Wage$wage)
> plot(Wage$education, Wage$wage)
> plot(Wage$age, Wage$wage)
```



Comparing the relation of Wage with Marital Status, Job Class, Education & Age.

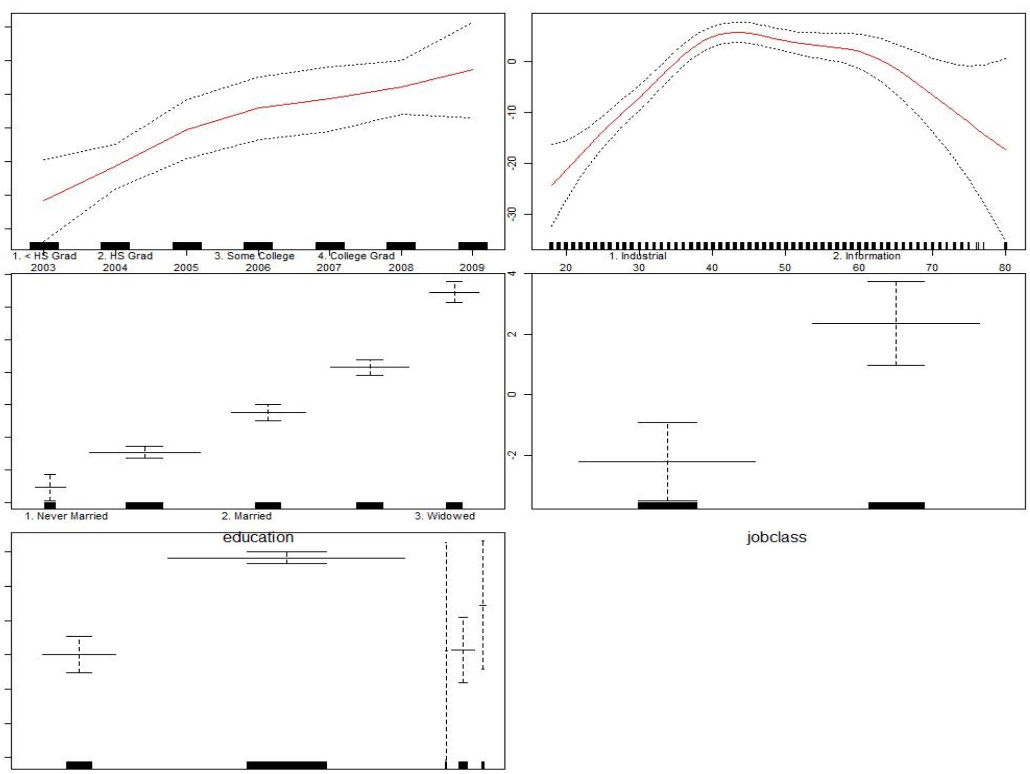
- We can see that Wages for Married couples are the highest among all, while those for ‘Never Married’ persons are the lowest.
- Also, the wage for ‘Information’ Job Class people is more than the other.
- Wages increase as the level of Education increases.
- Wage first increases on average with age, and then settles.

We will use GAM to predict wages using natural spline functions of all these features: year, age, educated, job class, and marital status:

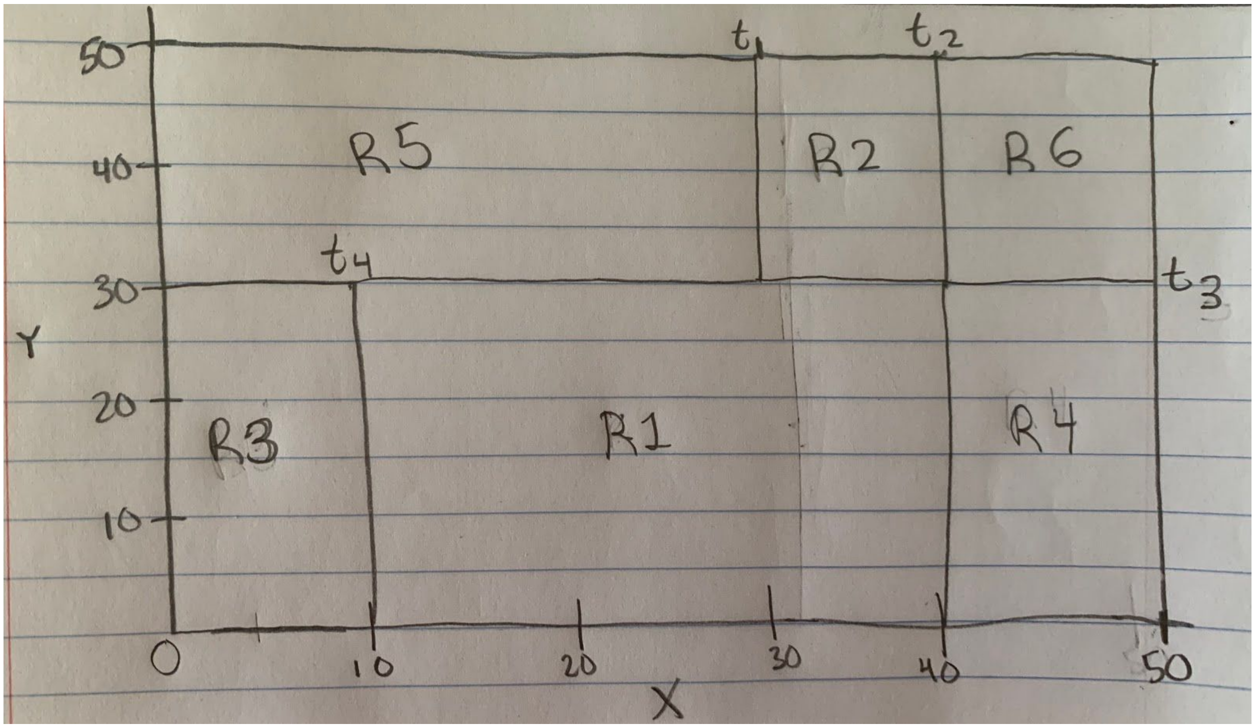
```
Analysis of Deviance Table

Model 1: wage ~ lo(year, span = 0.8) + s(age, 5) + education
Model 2: wage ~ lo(year, span = 0.8) + s(age, 5) + education + jobclass
Model 3: wage ~ lo(year, span = 0.8) + s(age, 5) + education + maritl
Model 4: wage ~ lo(year, span = 0.8) + s(age, 5) + education + jobclass +
    maritl
  Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
1    2987.5    3691875
2    2986.5    3679635  1    12240 0.0014149 **
3    2983.5    3597679  3    81956 1.033e-14 ***
4    2982.5    3583733  1    13946 0.0006573 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Here model 4 performs better than all which means that predicting wages using education, jobClass, Marital Status, Age yields better results. Thus plotting the graph for it:



Question 7



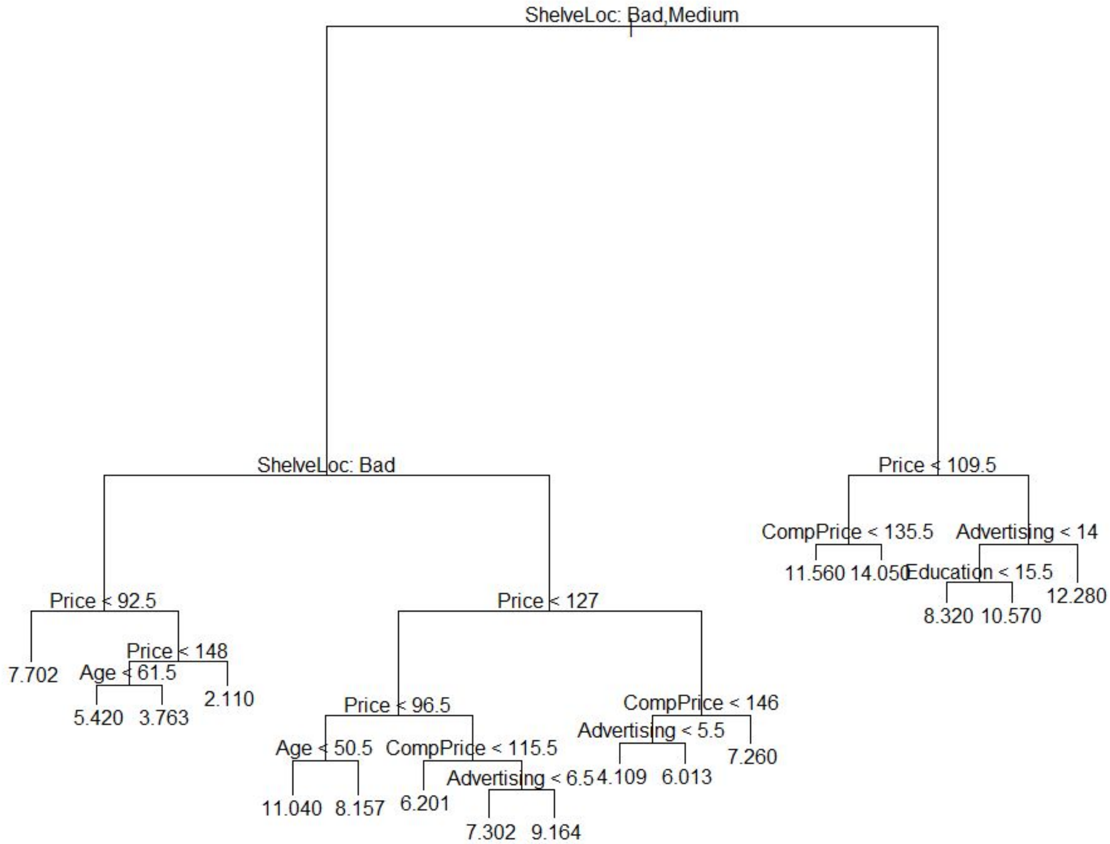
Question 8

b)

```
> #answer b
> library('tree')
> fit.tree = tree(Sales~., data=train)
> summary(fit.tree)

Regression tree:
tree(formula = Sales ~ ., data = train)
Variables actually used in tree construction:
[1] "ShelveLoc"   "Price"       "Age"         "CompPrice"   "Advertising" "Education"
Number of terminal nodes:  17
Residual mean deviance:  2.261 = 594.5 / 263
Distribution of residuals:
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-4.23200 -0.93760 -0.02135  0.00000  0.95750  4.21000

> plot(fit.tree)
> text(fit.tree,pretty=0)
```

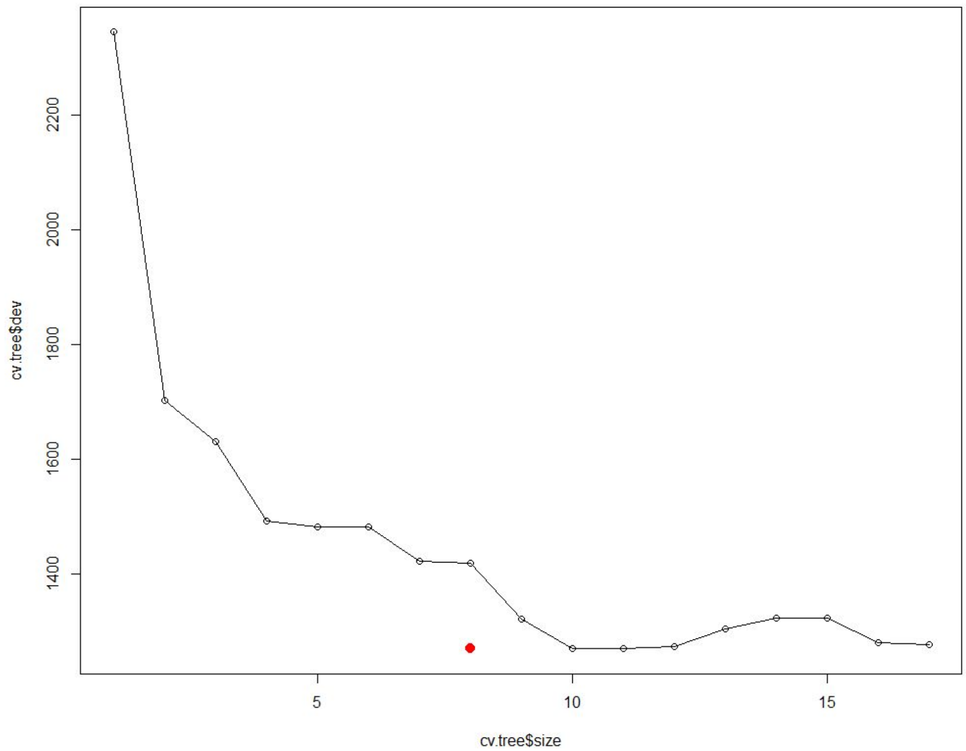



```
> pred.tree = predict(fit.tree, test)
> err.tree = mean((test$Sales - pred.tree)^2)
> err.tree
[1] 4.462848
```

Test MSE using a regression tree comes out to be: 4.462848

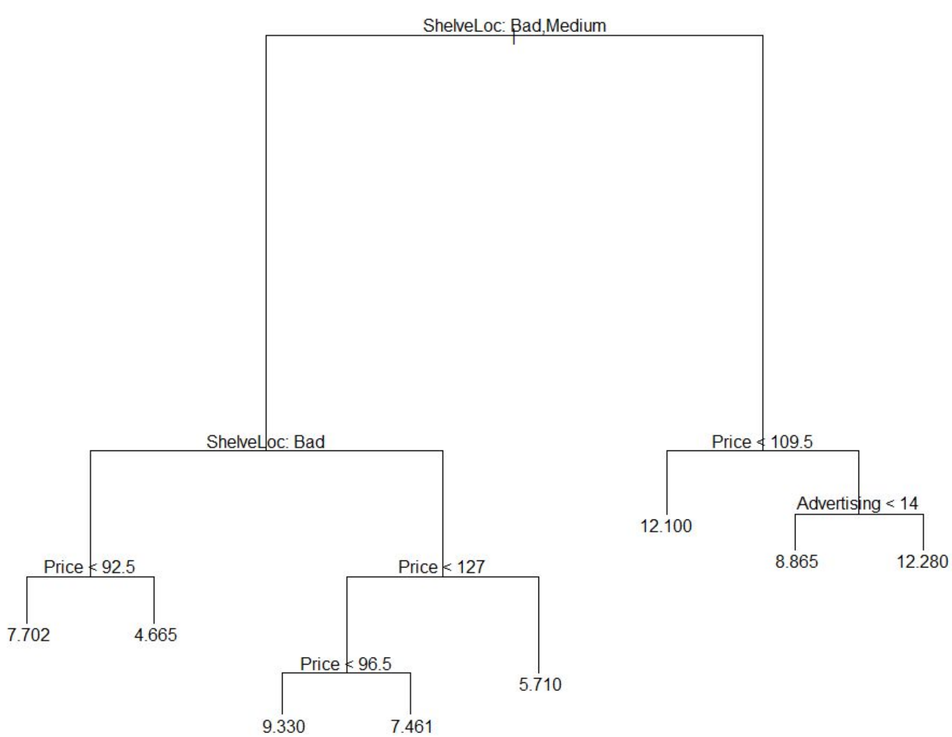
c)

```
> cv.tree = cv.tree(fit.tree)
> min.tree=which.min(cv.tree$dev)
> plot(cv.tree$size,cv.tree$dev,type='o')
> points(min.tree,cv.tree$dev[min.tree],col="red",cex=2,pch=20)
> min.tree
[1] 8
```



Using cross-validation, we determine that the optimal level of tree complexity is at 8. Thus, pruning the tree at 8:

```
> tree.prune=prune.tree(fit.tree,best=min.tree)
> plot(tree.prune)
> text(tree.prune,pretty=0)
```



```
> pred.prune=predict(tree.prune,newdata=test)
> err.prune = mean((test$Sales-pred.prune)^2)
> err.prune
[1] 4.595431
```

Test MSE is still approximately equal to what it was without tree pruning.

d) Using Bagging Approach

```
> fit.bag=randomForest(Sales~.,data=train,mtry=10,ntree=50,importance=T)
> pred.bag=predict(fit.bag,test)
> err.bag = mean((test$Sales-pred.bag)^2)
> err.bag
[1] 2.640623
```

The test MSE using the Bagging approach has reduced to 2.550465, which is a great improvement.

```
> importance(fit.bag)
              %IncMSE IncNodePurity
CompPrice  10.3974720     212.02613
Income      3.1174483     104.39811
Advertising 6.1454821     145.77314
Population -0.1012083      75.74027
Price      27.3947886     593.98440
ShelveLoc  25.6457444     902.54740
Age         6.4562175     163.25445
Education   3.0383080      57.82708
Urban      -0.7833393      11.04658
US          0.9710942      10.20255
```

The most important features are Price and SelveLoc.

e) Using Random Forest Approach

```
> fit.rf=randomForest(Sales~.,data=train,mtry=5,ntree=20,importance=T)
> pred.rf=predict(fit.rf,test)
> err.rf = mean((test$Sales-pred.rf)^2)
> err.rf
[1] 2.839471
```

Test MSE has reduced to 2.839471 using the Random Forest Approach.

```
> importance(fit.rf)
              %IncMSE IncNodePurity
CompPrice   3.4142618     185.48466
Income      2.2467896     119.02254
Advertising 3.2678845     198.23532
Population -0.7908272     106.21685
Price       9.1403308     611.59222
ShelveLoc  11.3303589     762.72732
Age         1.0311442     219.20300
Education  -0.1581187      67.99278
Urban       0.5609455      10.02120
US          2.1119816      15.49195
```

The most important features are ShelveLoc and Price.

It denotes that using 10 variables, the test error $\approx \sqrt{10}$, i.e. $\sqrt{m}$