# MSc_CW1_13173510_D_Monzon.word.doc

Delvin Monzon

09/11/2019

1) For each of parts (a) through (d), indicate whether we would generally expect the performance of a flexible statistical learning method to be better or worse than an inflexible method. Justify your answer.

(a) The number of predictors p is extremely large, and the number of observations n is small.

Works well with an inflexible model because n observations are low. Flexible models requires a lot of observations for an accurate function and would over fit the model due to low observations.

(b) The sample size n is extremely large, and the number of predictors (salary:job, education, etc) p is small.

A flexibile (Non parametric) would perform better due to the large sample size thus would fit the data better.

(c) The relationship between the predictors and response is highly non-linear.

Flexible model would be expected to perform better because a non flexible model assumes the model initialy e.g. linear. Flexible model has high degrees of freedom, more flexible relationship with x and y.

(d) The standard deviation of the error terms, i.e. $\sigma = sd(\varepsilon)$, is extremely high.

Works well with non-flexible model because flexible models are prone to over fit based on high errors and noise. It would perform well on training data but accuracy would be low in test data. As a result, variance would increase.

2) Bayes' Golf prediction P(Golf|Temperature) P(H|E) = P(E|H)*P(H)/P(E) Golf yes = 10/20 = 0.5 P(H) Golf no = 10/20 = 0.5 P(H) Hot = 12/20 = 0.6 P(E) Cold = 8/20 = 0.4 P(E)

1)P(Temp = hot|Golf = yes)= #5/10= 0.5 2)P(Tempp = hot|Golf = no)= #7/10= 0.7 3)P(Tempp = cold|Golf = yes)= #5/10= 0.5 4)P(Temp = cold| Golf = no)= #3/10 = 0.3

If hot then golf = no If cold then golf = yes

P(H|E) = P(E|H)*P(H)/P(E) P(Y|hot) = 1)P(golf = yes|Temp = hot)= (0.5*0.5)/0.6 = 0.4166667

#5/12 = 0.4166667 (5 = yes hot golf, 12 = hot) 2)P(golf = no|Temp = hot)= (0.7*0.5)/0.6= 0.5833333

#7/12 = 0.583333 (7 = no hot golf, 12 = hot) 3)P(golf = yes|Temp = cold)= (0.5*0.5)/0.4= 0.625

#5/8 = 0.625 (5 = yes cold golf, 8 = cold)

4)P(golf = no|Temp = cold)= (0.3*0.5)/0.4= 0.375

#3/8 = 0.375 (3 = no cold golf, 8 = cold)

3) This exercise involves the Auto data set studied in the class. (12% | 22%)

(a) Which of the predictors are quantitative, and which are qualitative? Quantitative: mpg, cylinders, displacement, horsepower, weight, acceleration, year, origin Qualitative: name

However, visually we could treat cylinders, origin and year as qualitative data using as.factor().

```
library(ISLR)
head(Auto)

##   mpg cylinders displacement horsepower weight acceleration year origin
## 1  18         8          307        130   3504         12.0   70      1
## 2  15         8          350        165   3693         11.5   70      1
## 3  18         8          318        150   3436         11.0   70      1
## 4  16         8          304        150   3433         12.0   70      1
## 5  17         8          302        140   3449         10.5   70      1
## 6  15         8          429        198   4341         10.0   70      1
##                        name
## 1 chevrolet chevelle malibu
## 2         buick skylark 320
## 3        plymouth satellite
## 4             amc rebel sst
## 5               ford torino
## 6          ford galaxie 500
```

#(b) What is the range of each quantitative predictor? #You can answer this using the range() function.

```
dfauto <- subset(Auto, select=c(1:8))

apply(dfauto, 2, range)

##       mpg cylinders displacement horsepower weight acceleration year origi
n
## [1,]  9.0         3           68         46   1613          8.0   70
1
## [2,] 46.6         8          455        230   5140         24.8   82
3
```

#(c) What is the median and variance of each quantitative predictor?

```
apply(dfauto, 2, median)
```

```
##          mpg    cylinders displacement   horsepower       weight accelerat
ion
##        22.75         4.00       151.00        93.50      2803.50        15
.50
##         year       origin
##        76.00         1.00
```

```
apply(dfauto, 2, var)
```

```
##          mpg    cylinders displacement   horsepower       weight accelerat
ion
## 6.091814e+01 2.909696e+00 1.095037e+04 1.481569e+03 7.214847e+05 7.611331e
+00
##         year       origin
## 1.356991e+01 6.488595e-01
```

#(d) Now remove the 11th through 79th observations (inclusive) in the dataset. #What is the range, median, and variance of #each predictor in the subset of the data that remains?

```
dfauto2 = dfauto[-c(11:79),]


apply(dfauto2, 2, range)
```

```
##       mpg cylinders displacement horsepower weight acceleration year origi
n
## [1,] 11.0         3           68         46   1649          8.5   70
1
## [2,] 46.6         8          455        230   4997         24.8   82
3
```

```
apply(dfauto2, 2, median)
```

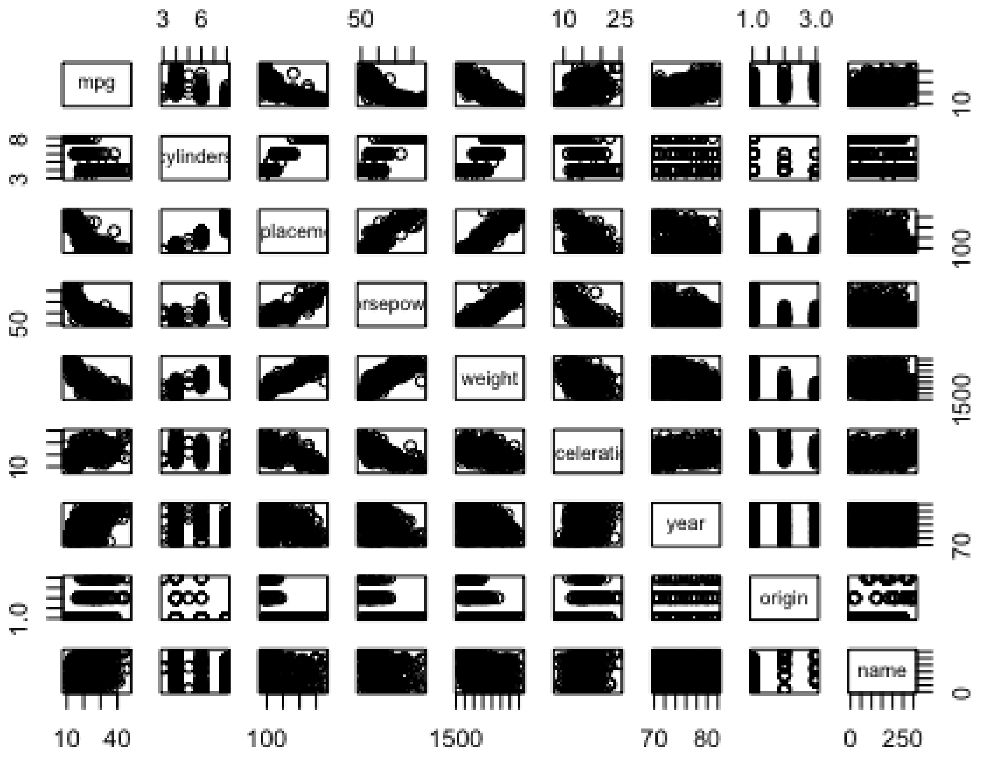
```
##          mpg    cylinders displacement   horsepower       weight accelerat
ion
##         23.9          4.0        144.0         90.0       2789.0         1
5.5
##         year       origin
##         77.0          1.0
```

```
apply(dfauto2, 2, var)
```

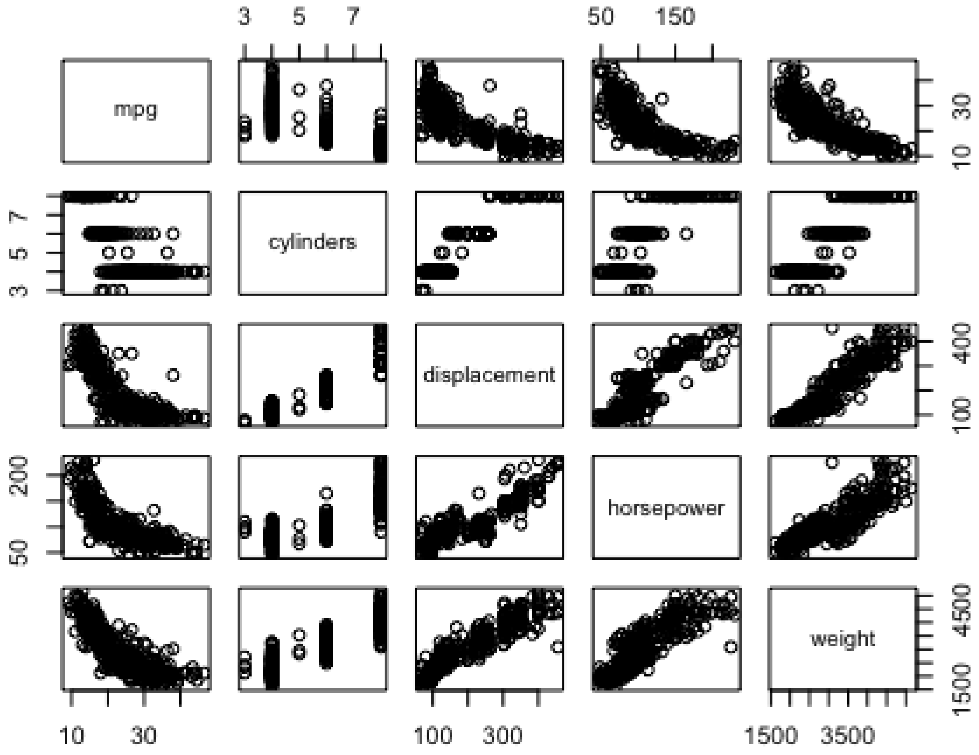
```
##          mpg    cylinders displacement   horsepower       weight accelerat
ion
## 6.135039e+01 2.748938e+00 1.003229e+04 1.291977e+03 6.574970e+05 7.296234e
+00
##         year       origin
## 1.004873e+01 6.803261e-01
```

#(e) Using the full data set, investigate the predictors graphically, #using scatterplots or other tools of your choice. #Create some plots highlighting the relationships among the predictors. #Comment on your findings. As you can see from the below pairs() plot, there are many factors that do not show a correlation based on the scatter of the plots. Below that, I have chosen plots, I believe,that have a significant correlation with each other.

```
pairs(Auto)
```

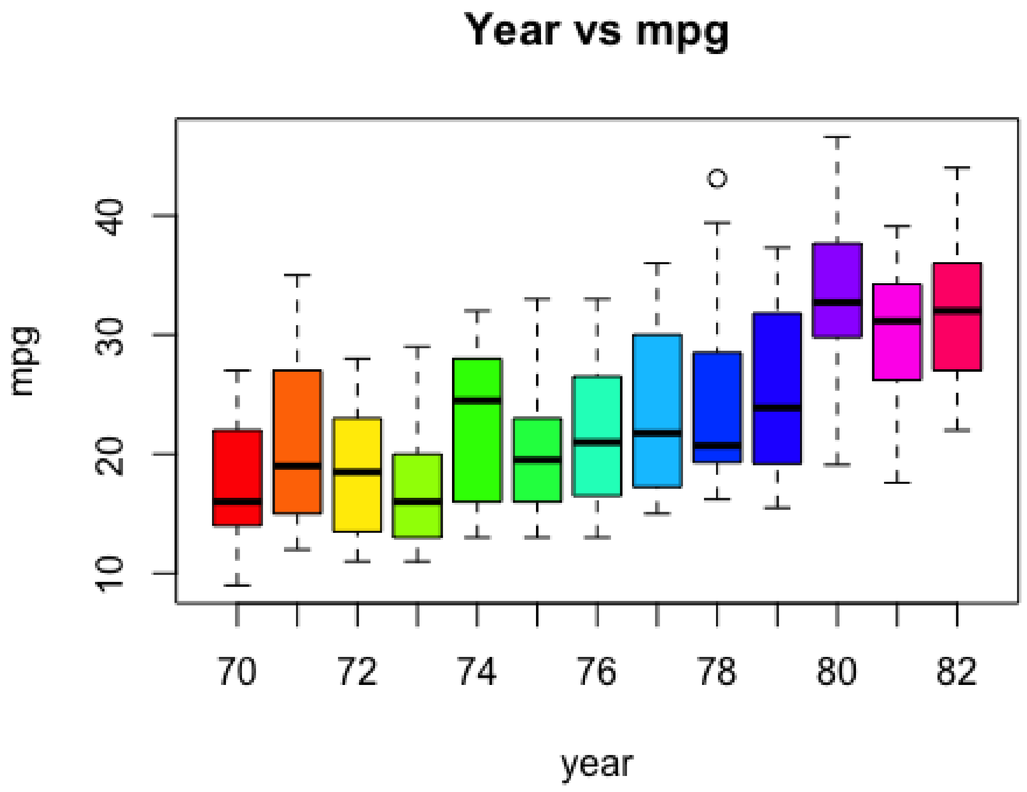


```
cylinders = as.factor(Auto$cylinders)
origin = as.factor(Auto$origin)
year = as.factor(Auto$year)

pairs(~mpg +cylinders + displacement + horsepower + weight, Auto)
```

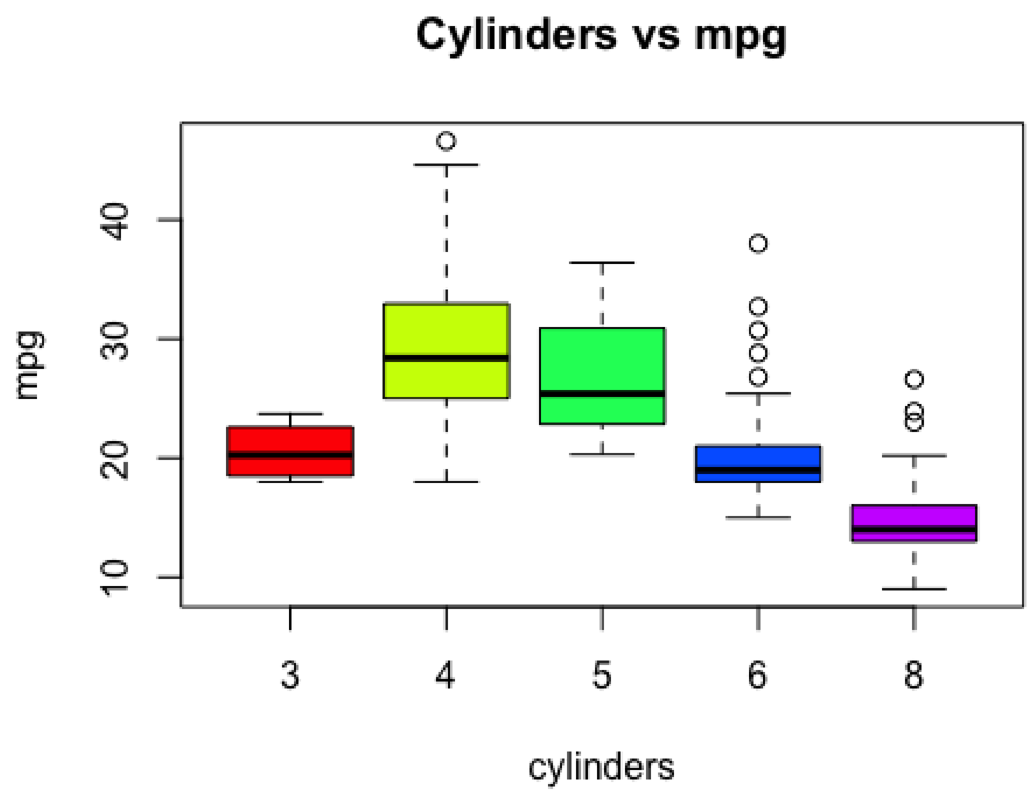


```
cor_data = Auto[,c(1:8)]
cor_data2 = cor(cor_data)
round(cor_data2,2)

##                mpg cylinders displacement horsepower weight acceleration 
year
## mpg           1.00     -0.78        -0.81      -0.78  -0.83         0.42 
0.58
## cylinders    -0.78      1.00         0.95       0.84   0.90        -0.50 -
0.35
## displacement -0.81      0.95         1.00       0.90   0.93        -0.54 -
0.37
## horsepower   -0.78      0.84         0.90       1.00   0.86        -0.69 -
0.42
## weight       -0.83      0.90         0.93       0.86   1.00        -0.42 -
0.31
## acceleration  0.42     -0.50        -0.54      -0.69  -0.42         1.00 
0.29
## year          0.58     -0.35        -0.37      -0.42  -0.31         0.29 
1.00
## origin        0.57     -0.57        -0.61      -0.46  -0.59         0.21 
0.18
##              origin
## mpg            0.57
## cylinders     -0.57
## displacement  -0.61
## horsepower    -0.46
## weight        -0.59
## acceleration   0.21
## year           0.18
## origin         1.00
```

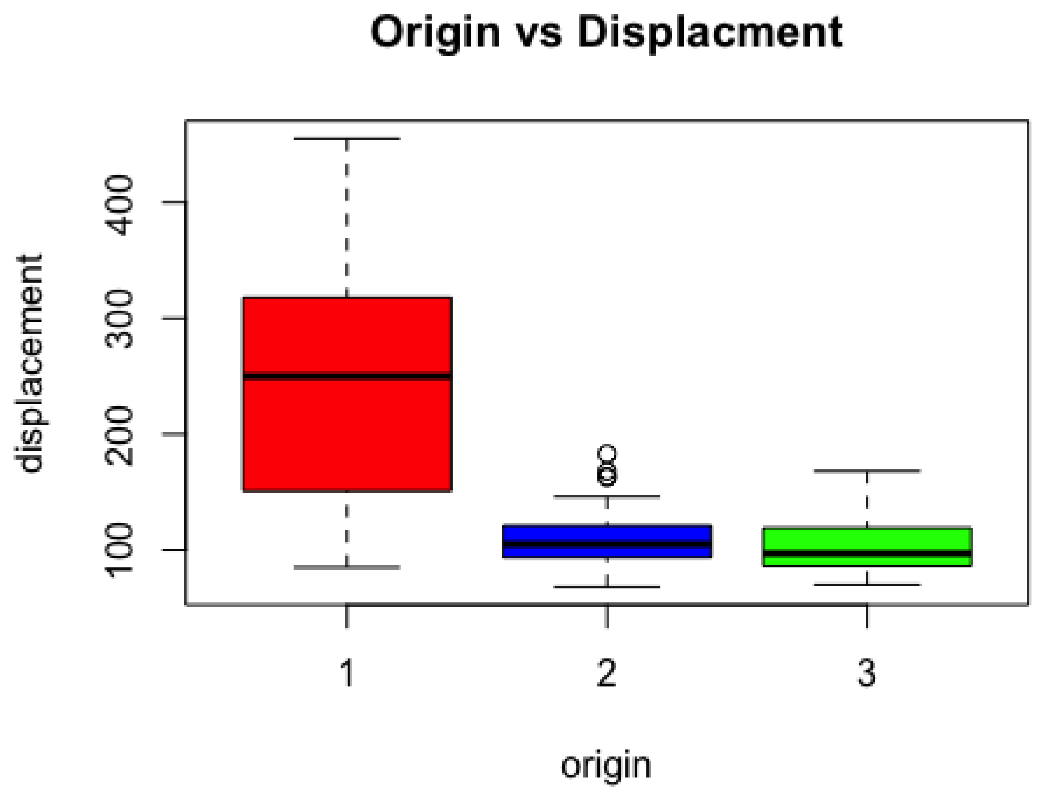
```
plot(year, Auto$mpg, xlab = 'year',ylab = 'mpg', col = rainbow(length(unique(
year))), main = 'Year vs mpg')
```



```
plot(cylinders, Auto$mpg, xlab = 'cylinders',ylab = 'mpg', col = rainbow(leng
th(unique(cylinders))), main = 'Cylinders vs mpg')
```



```
plot(origin, Auto$displacement,xlab = 'origin',ylab = 'displacement', col = c
('red','blue','green'),main = 'Origin vs Displacment')
```



I have also treated cylinders, origin and year as quanlitative data.

Mpg vs cylinders shows a general negative correlation the more cylinders a car has the less mpg there is. The plot of '3' cylinders vs mpg does not follow this trend as it has less average mpg than 4 and 5 cylinders.

As seen in the correlation matrix and graph plots, many variables have high positive and negative correlations.

This is: Mpg - highly negatively correlated with displacement,weight and horsepower. Although positively correlated with year.

Displacement - highly negative correlation with mpg, highly positive with cylinders, horsepower and weight.

Horsepower - highly positively with cylinders, displacement, weight, negative correlation with mpg.

Weight - highly positive with cylinders, displacement, horsepower, negative with mpg.

Cylinders - highly negative with mpg, highly positive with displacement, horsepower and weight.

#(f) Suppose that we wish to predict gas mileage (mpg) on the basis of the other variables. Do your plots suggest that any of the other variables might be useful in predicting mpg? Justify your answer.

In predicting mpg, other variables that are highly correlated with it are based on plots and correlation: weight, displacement, horsepower and cylinders in that order. All are highly negatively correlated. Year also showed a moderate positive correlation with mpg and could be used.

As a result we can use these to predict the mpg of a vehicle. Due to the variables being in different scales, the correlation coefficient was the best way to see and measure any relation between variables.

#4(a) Use the lm() function to perform a simple linear regression with mpg as the response and horsepower as the predictor. Use the summary() function to print the results. Comment on the output. For example:

```
library(ISLR)

lm.fit = lm(mpg~horsepower, data = Auto)
summary(lm.fit)

##
## Call:
## lm(formula = mpg ~ horsepower, data = Auto)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.5710  -3.2592  -0.3435   2.7630  16.9240
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 39.935861   0.717499   55.66   <2e-16 ***
## horsepower  -0.157845   0.006446  -24.49   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.906 on 390 degrees of freedom
## Multiple R-squared:  0.6059, Adjusted R-squared:  0.6049
## F-statistic: 599.7 on 1 and 390 DF,  p-value: < 2.2e-16

cor(Auto$mpg,Auto$horsepower)

## [1] -0.7784268
```

#i. Is there a relationship between the predictor and the response? Initially, a high t-value of 55 which indicates the null hypothesis (that mpg and horsepower have no relationship) is untrue and translates to a high p-value.

A p-value of <2.2e-16 is well below the rejection of <0.05 to disprove the null hypothesis. Therefore, this shows that there is a very strong relationship between mpg and horsepower.

RSE = 4.906 4.906/(46.6-9)*100 = 13% off each. Each response is expected to deviate around 4.906 mpg from predictor.

Std. error of Beta 1 is very small (0.006) which indicates that the true value of beta 1 is as far away from 0. A value far from 0 indicates a relationship between mpg and horsepower.

#ii. How strong is the relationship between the predictor and the response? A correlation coefficient of -0.7784268 indicates a highly negative correlation between mpg and horsepower. A high t-value of 55 also indicates a strong relationship.

As this is a linear model, R2 = cor(X,Y)^2. The adjusted Rsquared value is 0.6049 (60.49%) which indicates that the linear model is a good model and has a strong relationship between predictor and response, accounting for roughly 60% of the variance. However, because we are only using one variable against mpg we can use the 0.6059 (60.59%) R-squared value.

A value closer to 1 would indicate a perfect fit, so perhaps a polynomial model would suit it better, such as a quadratic model.

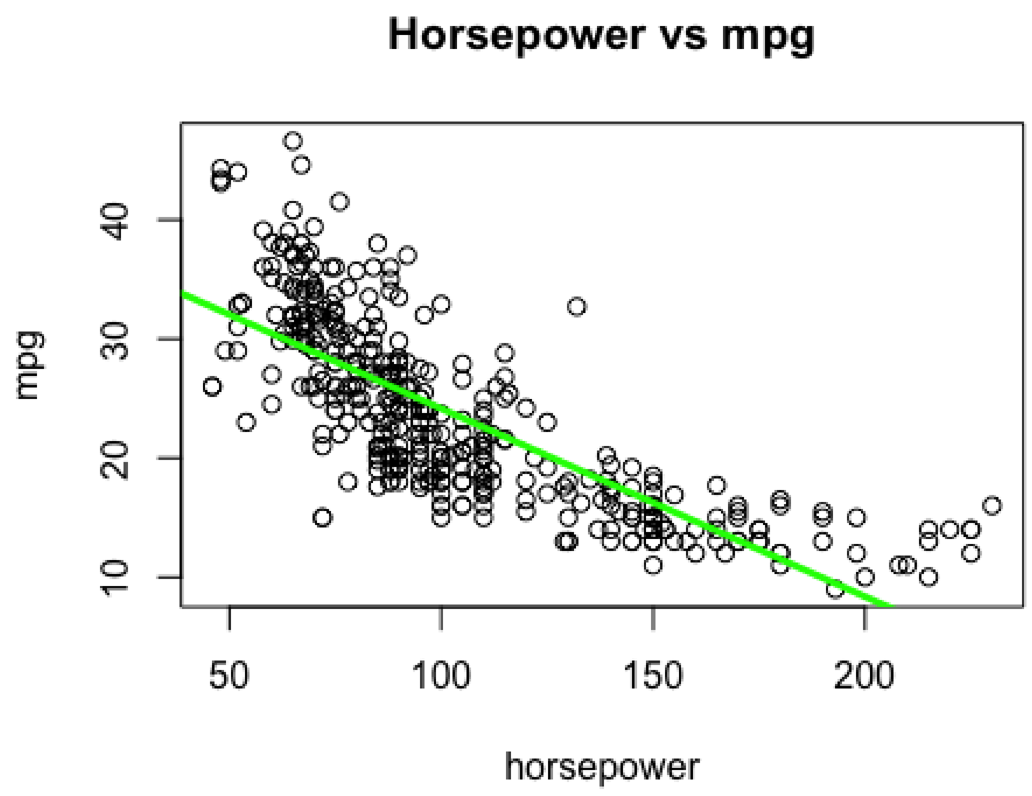
#iii. Is the relationship between the predictor and the response positive or negative? The relationship is negative because as the mpg increases the hosepower decreases. The horsepower has an intercept of -0.157845 indicating a negative relationship, which is also supported by the highly negative correlation coefficient of -0.7784268.

#iv. What is the predicted mpg associated with a horsepower of 89? #What are the associated 99% confidence and prediction intervals?

```
#Predicted mpg is 25.88768.
predict(lm.fit,data.frame(horsepower=89),level = 0.99, interval="confidence",
data = Auto)

##        fit      lwr      upr
## 1 25.88768 25.19633 26.57903

predict(lm.fit,data.frame(horsepower=89),level = 0.99, interval="prediction",
data = Auto)

##        fit      lwr      upr
## 1 25.88768 13.17035 38.60501
```

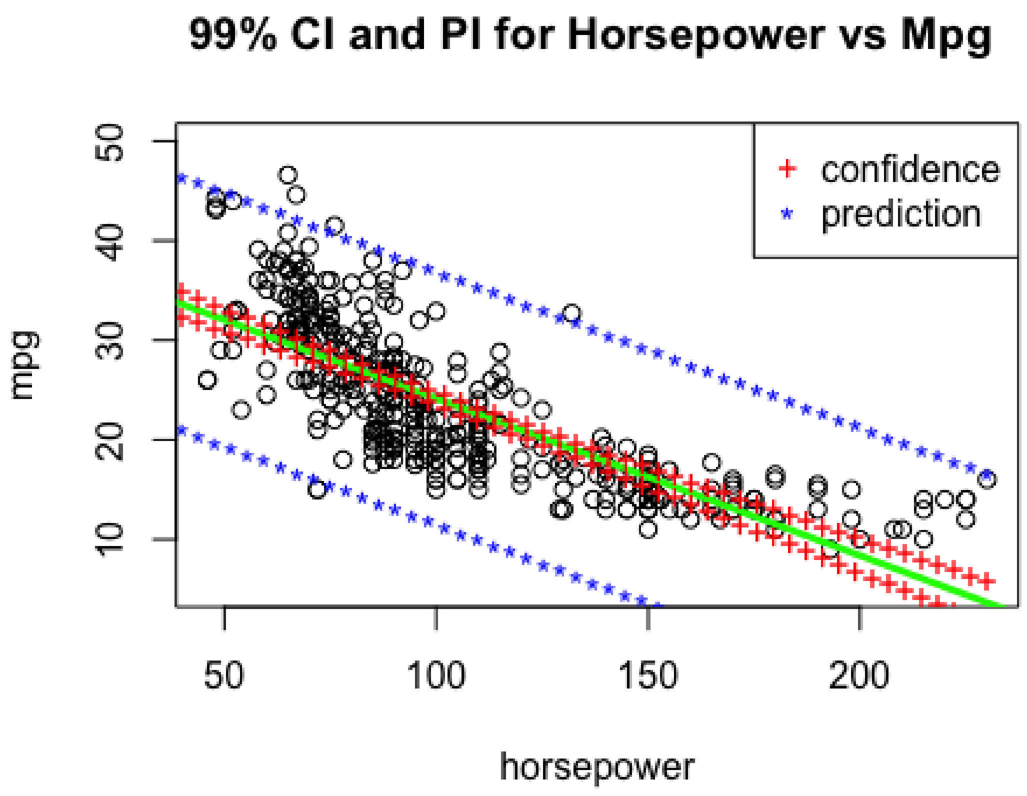
#(b) Plot the response and the predictor. Use the abline() function to display the least squares regression line.

```
plot(Auto$horsepower, Auto$mpg,
     xlab = 'horsepower', ylab = 'mpg',
     main = 'Horsepower vs mpg')
abline(lm.fit, lwd=3, col='green')
```



#(c) Plot the 99% confidence interval and prediction interval in the same plot as (b) using different colours and legends.

```
nd  <- data.frame(horsepower=seq(40,230,length=50))

plot(Auto$horsepower, Auto$mpg,
     xlab="horsepower", ylab = "mpg",
     main = "99% CI and PI for Horsepower vs Mpg ",
     ylim = c(5,50)
    )
abline(lm.fit)
abline(lm.fit, lwd=3, col='green')

p_conf <- predict(lm.fit,nd, level = 0.99, interval="confidence")
p_pred <- predict(lm.fit, nd, level = 0.99, interval="prediction")
lines(nd$horsepower, p_conf[,"lwr"], col="red", type="b", pch="+")
lines(nd$horsepower, p_conf[,"upr"], col="red", type="b", pch="+")
lines(nd$horsepower, p_pred[,"upr"], col="blue", type="b", pch="*")
lines(nd$horsepower, p_pred[,"lwr"], col="blue", type="b", pch="*")
legend("topright",
       pch=c("+","*"),
       col=c("red","blue"),
       legend = c("confidence","prediction"))
```



#Q5 Logistic regresion #A recent study has shown that the accurate prediction of the office room occupancy leads to potential energy savings of 30%. #In this question, you are required to build logistic regression models by using different environmental measurements #as features, such as temperature, humidity, light, CO2 and humidity ratio, to predict the office room occupancy. #The provided training dataset consists of 2,000 samples, whilst the testing dataset consists of 300 samples.

#(a) Load the training and testing datasets from corresponding files, and display the statistics about different features in the training dataset.

```
mydata.train = read.table("Training_set.txt", header = T, sep = ',')
mydata.test = read.table("Testing_set.txt", header = T, sep = ',')

summary(mydata.train)

##   Temperature       Humidity         Light             CO2
##  Min.   :20.10   Min.   :18.96   Min.   :  0.0   Min.   : 426.0
##  1st Qu.:20.89   1st Qu.:21.82   1st Qu.:  0.0   1st Qu.: 448.0
##  Median :21.20   Median :25.00   Median :  0.0   Median : 485.5
##  Mean   :21.42   Mean   :24.22   Mean   :144.7   Mean   : 634.6
##  3rd Qu.:22.10   3rd Qu.:26.29   3rd Qu.:433.0   3rd Qu.: 845.8
##  Max.   :23.18   Max.   :28.50   Max.   :744.0   Max.   :1139.0
##  HumidityRatio        Occupancy
##  Min.   :0.002824   Min.   :0.0000
##  1st Qu.:0.003375   1st Qu.:0.0000
##  Median :0.003905   Median :0.0000
##  Mean   :0.003836   Mean   :0.2775
##  3rd Qu.:0.004343   3rd Qu.:1.0000
##  Max.   :0.004817   Max.   :1.0000
```

```
summary(mydata.test)
```

```
##   Temperature       Humidity         Light             CO2
##  Min.   :20.39   Min.   :21.86   Min.   :  0.00   Min.   : 484.7
##  1st Qu.:20.65   1st Qu.:24.43   1st Qu.:  0.00   1st Qu.: 560.5
##  Median :21.32   Median :26.32   Median : 44.67   Median : 592.8
##  Mean   :21.68   Mean   :26.66   Mean   :207.41   Mean   : 657.1
##  3rd Qu.:22.11   3rd Qu.:28.79   3rd Qu.:429.00   3rd Qu.: 735.9
##  Max.   :24.39   Max.   :36.00   Max.   :756.50   Max.   :1595.7
##  HumidityRatio        Occupancy
##  Min.   :0.003275   Min.   :0.0
##  1st Qu.:0.003959   1st Qu.:0.0
##  Median :0.004335   Median :0.0
##  Mean   :0.004276   Mean   :0.2
##  3rd Qu.:0.004562   3rd Qu.:0.0
##  Max.   :0.005670   Max.   :1.0
```

#(b) Build a logistic regression model by only using the Temperature feature to predict the room occupancy. Display the confusion matrix and the predictive accuracy obtained on the testing dataset.

```
glm.temp.train <- glm(Occupancy~Temperature,
                      data=mydata.train,
                      family = binomial
                     )

temp.prob <- predict(glm.temp.train, mydata.test, type = "response")
temp.prob[1:10]
```

```
##          1          2          3          4          5          6
7
## 0.40824223 0.40824223 0.40824223 0.30141854 0.02771593 0.02771593 0.027715
93
##          8          9         10
## 0.03002312 0.98267039 0.99520154
```

```
nrow(mydata.test)
```

```
## [1] 300
```

```
temp.pred <- rep(1,300)
temp.pred[temp.prob<0.5] = 0

table(temp.pred, mydata.test$Occupancy)
```

```
##
## temp.pred   0   1
##         0 182  39
##         1  58  21

#TP and TN rate
mean(temp.pred == mydata.test$Occupancy)

## [1] 0.6766667

#FP and FN rate
mean(temp.pred != mydata.test$Occupancy)

## [1] 0.3233333
```

#(c) Build a logistic regression model by only using the Humidity feature to predict the room occupancy. #Display the confusion matrix and the predictive accuracy obtained on the testing dataset.

```
glm.hum.train <- glm(Occupancy~Humidity,
                      data=mydata.train,
                      family = binomial
                    )
hum.prob <- predict(glm.hum.train, mydata.test, type = "response")
hum.prob[1:10]

##          1          2          3          4          5          6
7
## 0.95624118 0.95101246 0.94372443 0.76573687 0.04979711 0.06065172 0.062816
41
##          8          9         10
## 0.06281641 0.08313123 0.09729204

hum.pred <- rep(1,300)
hum.pred[hum.prob<0.5] = 0

table(hum.pred, mydata.test$Occupancy)

##
## hum.pred   0   1
##        0 133  22
##        1 107  38

#TP and TN rate
mean(hum.pred == mydata.test$Occupancy)

## [1] 0.57

#FP and FN rate
mean(hum.pred != mydata.test$Occupancy)

## [1] 0.43
```

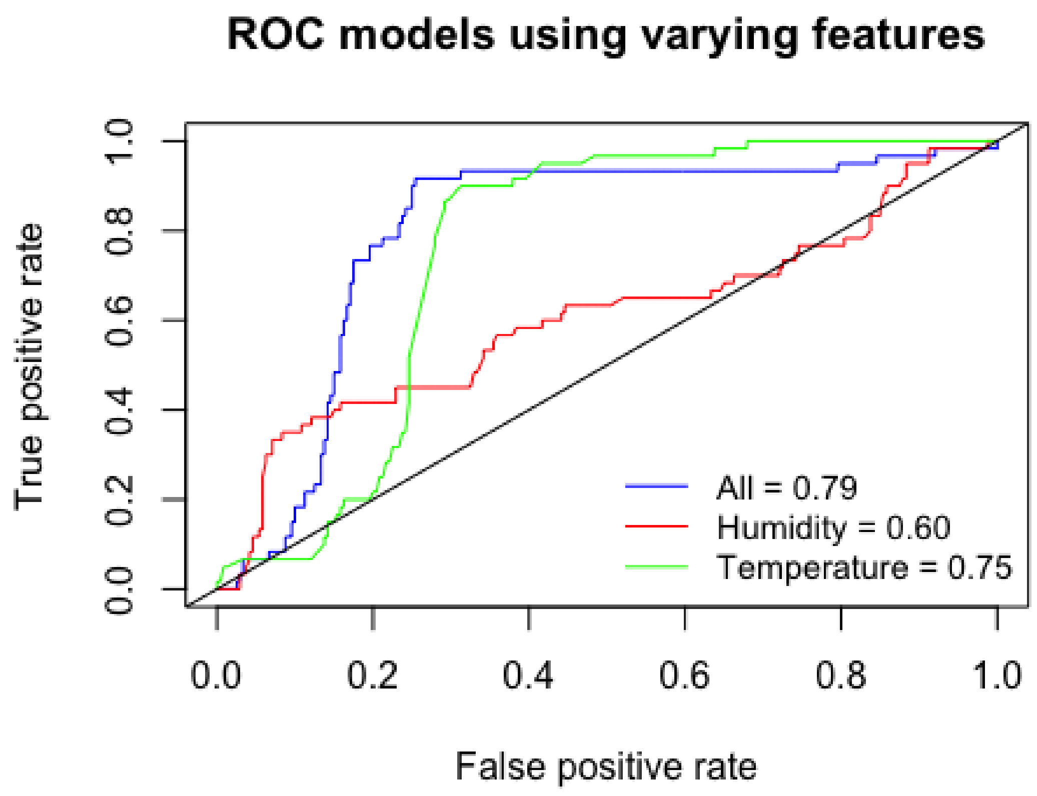
#(d) Build a logistic regression model by using all features to predict the room occupancy. #Display the confusion matrix and the predictive accuracy obtained on the testing dataset.

```
glm.train <- glm(Occupancy~Humidity+Temperature+Light+CO2+HumidityRatio,
                      data=mydata.train,
                      family = binomial
                )

all.prob <- predict(glm.train, mydata.test, type = "response")
all.prob[1:10]

##           1           2           3           4           5           6
## 0.987364969 0.985196328 0.979438941 0.385985442 0.930870345 0.935976282
##           7           8           9          10
## 0.943582743 0.941501391 0.009555105 0.045356242

all.pred <- rep(1,300)
all.pred[all.prob<0.5] = 0

table(all.pred, mydata.test$Occupancy)

##
## all.pred   0   1
##        0 184  13
##        1  56  47

#TP and TN rate
mean(all.pred == mydata.test$Occupancy)

## [1] 0.77

#FP and FN rate
mean(all.pred != mydata.test$Occupancy)

## [1] 0.23
```

#(e) Compare the predictive performance of three different models by drawing ROC curves and calculating the AUROC values. #Discuss the comparison results.

```
library(gplots)

##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess

library(ROCR)


pr_trained_model1 <- prediction(all.prob, mydata.test$Occupancy)
```

```
auroc_trained_model1 <- performance(pr_trained_model1, measure = "auc")
auroc_trained_model_value1 <- auroc_trained_model1@y.values[[1]]
print(paste("The AUROC value of the trained model using all features is", aur
oc_trained_model_value1,"."))

## [1] "The AUROC value of the trained model using all features is 0.79777777
7777777 ."

pr_trained_model2 <- prediction(hum.prob, mydata.test$Occupancy)
auroc_trained_model2 <- performance(pr_trained_model2, measure = "auc")
auroc_trained_model_value2 <- auroc_trained_model2@y.values[[1]]
print(paste("The AUROC value of the trained model using humidity is", auroc_t
rained_model_value2,"."))

## [1] "The AUROC value of the trained model using humidity is 0.603819444444
444 ."

pr_trained_model3 <- prediction(temp.prob, mydata.test$Occupancy)
auroc_trained_model3 <- performance(pr_trained_model3, measure = "auc")
auroc_trained_model_value3 <- auroc_trained_model3@y.values[[1]]
print(paste("The AUROC value of the trained model using temperature is", auro
c_trained_model_value3,"."))

## [1] "The AUROC value of the trained model using temperature is 0.752743055
555556 ."

prf_trained_model1 <- performance(pr_trained_model1, measure = "tpr", x.measu
re = "fpr")
prf_trained_model2 <- performance(pr_trained_model2, measure = "tpr", x.measu
re = "fpr")
prf_trained_model3 <- performance(pr_trained_model3, measure = "tpr", x.measu
re = "fpr")


plot(prf_trained_model1, col= 'Blue', main = "ROC models using varying featur
es")
plot(prf_trained_model2, col= 'Red', add = TRUE)
plot(prf_trained_model3, col= 'Green', add = TRUE)
#plot(prf_trained_model, colorize = T) you can try the colored version
legend("bottomright",
       legend = c('All = 0.79','Humidity = 0.60','Temperature = 0.75'),
                lty=1, cex=0.9, bty="n", col = c("Blue","Red","Green")
       )
abline(a = 0, b = 1)
```



The larger area under a ROC curve indicates better accuracy, therefore using 'All' features in a model obtained the best accuracy out of the three models(0.79,0.75 and 0.6 respectively). 'All' had the higher TPR. The single feature model of 'Temperature' had significantly better accuracy then Humidity. It also was very close to 'All' suggesting it had a high correlation to Occupancy.

To improve the 'All' model I would remove some irrelevant features based on their correlation with occupancy which should increase the overall accuracy of the model.

#Q6 We are trying to learn regression parameters for a dataset which we know was generated from a polynomial of a certain degree, but we do not know what this degree is. # Assume the data was actually generated from a polynomial of degree 3 with some added noise, that is y=β0+β1x+β2x2+β3x3+ε, ε~N(0,1) # For training we have 400 (x, y)-pairs and for testing we are using an additional set of 400 (x, y)-pairs. Since we do not know the degree of the polynomial we learn two models from the data. # • Model A learns parameters for a polynomial of degree 2, and # • Model B learns parameters for a polynomial of degree 4. # (a) Which of these two models is likely to fit the test data better? Justify your answer.

I believe Model A (trained on polynomial 2) would fit the test data better because it is simpler than B, regarding degrees of polynomial, so will less likely over react/over fit to noise and errors in the training data. As a result, it may have better predictive performance than model B. However, Model A will have higher bias but less variance than B.

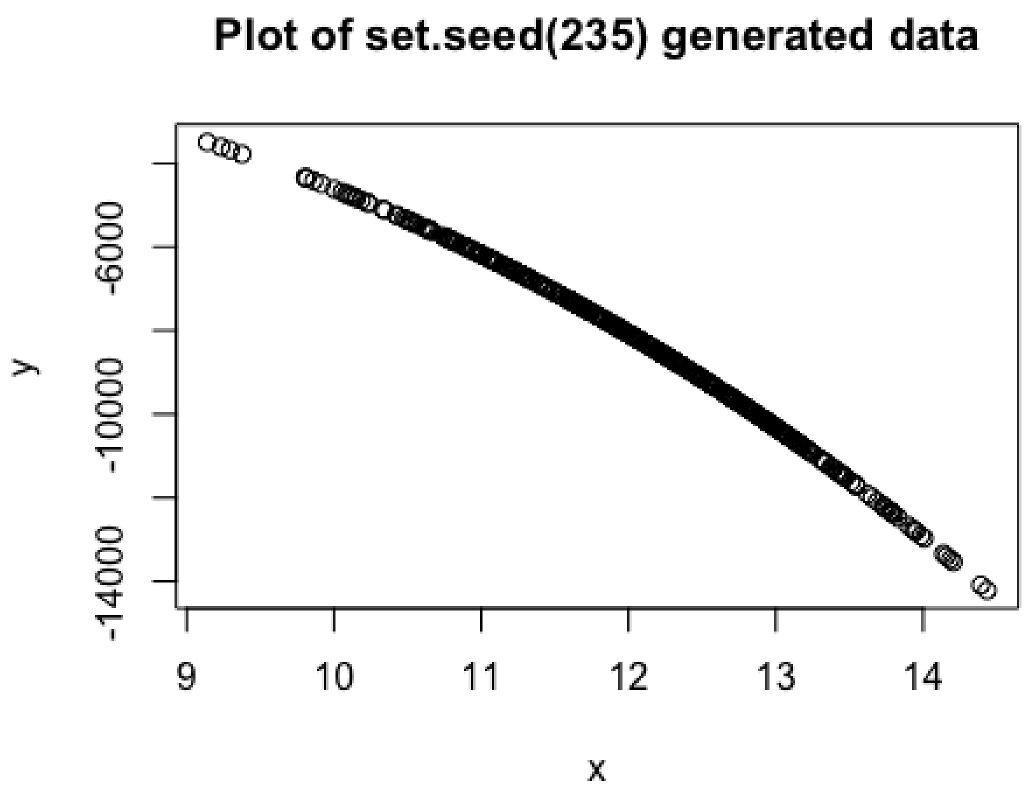
Although model B was trained on a higher polynomial degree (4) and is a more flexible model that can potentially provide more accurate estimates, it may require a lot more than 400 data points. Also, it could lead to overfitting (the random errors and noisey data) when trained and may not perform as well on the test data. This would also mean Model B has a higher variance but less bias than model A.

## We will now perform cross-validation on this simulated data set. (b) Generate the simulated data set as follows:

```
set.seed(235)
x = 12 + rnorm(400)
y = 1 - x + 4*x^2 - 5*x^3 + rnorm(400)
d1 = data.frame(x,y)
```

## Create a scatterplot of X against Y. Comment on what you find.

```
set.seed(235)
p1 = plot(x,y,main = 'Plot of set.seed(235) generated data')
```



The plot shows a negative correlation between x and y.

## (c) Set the seed to be 34, and then compute the LOOCV and 10-fold CV errors that result from fitting the following four models using least squares:

## i. Y =β0 +β1X+β2X2 +ε

## Note you may find it helpful to use the data.frame() function to create a single data set containing both X and Y.

```
library(boot)

#   i. Y =β0 +β1X+β2X2 +ε
set.seed(34)
glm.fit1 <- glm(y~poly(x,2), data = d1)

#LOOCV
cv.err1 <- cv.glm(d1,glm.fit1)
cv.err1$delta

## [1] 100.2671 100.2571

#K-fold
cv.err2 <- cv.glm(d1,glm.fit1, K=10)
cv.err2$delta

## [1] 105.8857 105.1349

# ii.  Y =β0 +β1X+β2X2 +β3X3 +β4X4 +ε.
glm.fit2 <- glm(y~poly(x,4))

#LOOV
cv.err3 <- cv.glm(d1,glm.fit2)
cv.err3$delta

## [1] 1.017966 1.017931

#K-Fold
cv.err4 <- cv.glm(d1,glm.fit2, K=10) #8 works better
cv.err4$delta

## [1] 1.023266 1.021521

summary(glm.fit1)
```

```
##
## Call:
## glm(formula = y ~ poly(x, 2), data = d1)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -48.414   -6.713    1.202    6.156   63.558
##
## Coefficients:
##               Estimate Std. Error  t value Pr(>|t|)
## (Intercept) -8.406e+03  4.829e-01 -17407.5   <2e-16 ***
## poly(x, 2)1 -4.207e+04  9.658e+00  -4355.6   <2e-16 ***
## poly(x, 2)2 -4.659e+03  9.658e+00   -482.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 93.27952)
##
##     Null deviance: 1791391934  on 399  degrees of freedom
## Residual deviance:      37032  on 397  degrees of freedom
## AIC: 2954.4
##
## Number of Fisher Scoring iterations: 2
```

```
summary(glm.fit2)
```

```
##
## Call:
## glm(formula = y ~ poly(x, 4))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9184  -0.5951   0.0207   0.6436   3.3437
##
## Coefficients:
##               Estimate Std. Error    t value Pr(>|t|)
## (Intercept) -8.406e+03  5.008e-02 -1.678e+05   <2e-16 ***
## poly(x, 4)1 -4.207e+04  1.002e+00 -4.200e+04   <2e-16 ***
## poly(x, 4)2 -4.659e+03  1.002e+00 -4.651e+03   <2e-16 ***
## poly(x, 4)3 -1.914e+02  1.002e+00 -1.911e+02   <2e-16 ***
## poly(x, 4)4  2.331e-03  1.002e+00  2.000e-03    0.998
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 1.003333)
##
##     Null deviance: 1.7914e+09  on 399  degrees of freedom
## Residual deviance: 3.9632e+02  on 395  degrees of freedom
## AIC: 1143.5
```

```
##
## Number of Fisher Scoring iterations: 2
```

## (d) Repeat (c) using random seed 68, and report your results. Are your results the same as what you got in (c)? Why?

```
#   i. Y =β0 +β1X+β2X2 +ε
set.seed(68)
glm.fit1 <- glm(y~poly(x,2), data = d1)

#LOOCV
cv.err1 <- cv.glm(d1,glm.fit1)
cv.err1$delta

## [1] 100.2671 100.2571

#K-fold
cv.err2 <- cv.glm(d1,glm.fit1, K=10)
cv.err2$delta

## [1] 97.40727 97.15072

# ii. Y =β0 +β1X+β2X2 +β3X3 +β4X4 +ε.
glm.fit2 <- glm(y~poly(x,4))

#LOOV
cv.err3 <- cv.glm(d1,glm.fit2)
cv.err3$delta

## [1] 1.017966 1.017931

#K-Fold
cv.err4 <- cv.glm(d1,glm.fit2, K=10) #8 works better
cv.err4$delta

## [1] 1.020981 1.019355
```

Set.seed(34) Model 1 (LOOCV) = 100.2571 Model 1 (10-Fold) = 103.3234 Model 2(LOOCV) = 1.017931 Model 2(10-Fold) = 1.021031

Set.seed(68) Model 1 (LOOCV) = 100.2571 Model 1 (10-Fold) = 96.63373 Model 2 (LOOCV) = 1.017931 Model 2 (10-Fold) = 1.017374

The results for LOOCV remain the same in both seed 34 and 68. The is because no matter how you split the data, LOOCV always works by iterating through/takes into consideration every single data point in the data frame and only splits 1 observation at a time. As a result the MSE calculated is the same.

However, for K-fold the results change from seed 34 to 68. This is because the split folds/parts created by the method is random for each seed and a different seed would split the data differently every time. As a result, the MSE calculated is different for each part thus a slightly different averaged MSE would be calculated.

## (e) Which of the models in (c) had the smallest LOOCV and 10-fold CV error? Is this what you expected? Explain your answer.

Model 2 (polynomial degree 4) produced the lowest MSE in both LOOCV and 10-Fold CV. This was unexpected as (seen in my answer to 6a) I initially thought it would over fit the training data and as a result may not perform well on test data. It seems 400 data points was enough to train it sufficiently and its flexibility was better suited than model 1.

## (f) Comment on the coefficient estimates and the statistical significance of the coefficient estimates that results from fitting the preferred model in (a).

The preffered model in (a) (quadratic model, polynomial 2) had a much higher MSE/CV error than anticipated, so most data points have a lot of error from the true values, for both LOOCV and 10-Fold. This indicates a very poor fit. The model most likely underfitted the data and was not flexible enough to predict accurately.

The t-value is much smaller than the quartic model which translates to a much higher p-value and thus a more likely to accept the null hypotheis. The coefficients are also all negative which indicates negative relationships between the response and predictor. Std. Error was also much higher than the other model, further indicating a poorer fit.

#(g) Fit a cubic model and compute its LOOCV error, 10-fold CV error under seed 34, #and comment on the coefficient estimates and the statistical significance of the coefficient estimates. #Compare the LOOCV and 10-fold CV error with the preferred model in (a).

```
set.seed(34)
glm.fit5 <- glm(y~poly(x,3))


#LOOV
cv.err9 <- cv.glm(d1,glm.fit5)
m3L = cv.err9$delta
m3L

## [1] 1.010868 1.010843

#K-Fold
cv.err10 <- cv.glm(d1,glm.fit5, K=10) #8 works better
m3K = cv.err10$delta
m3K

## [1] 1.011995 1.010844
```

Cubic model: LOOCV = 1.010843 10-Fold = 1.009027

```
summary(glm.fit5)

##
## Call:
## glm(formula = y ~ poly(x, 3))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9184  -0.5950   0.0207   0.6436   3.3436
##
## Coefficients:
##                Estimate Std. Error   t value Pr(>|t|)
## (Intercept) -8.406e+03  5.002e-02 -168056.7   <2e-16 ***
## poly(x, 3)1 -4.207e+04  1.000e+00  -42050.4   <2e-16 ***
## poly(x, 3)2 -4.659e+03  1.000e+00   -4657.2   <2e-16 ***
## poly(x, 3)3 -1.914e+02  1.000e+00    -191.3   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 1.0008)
##
##     Null deviance: 1.7914e+09  on 399  degrees of freedom
## Residual deviance: 3.9632e+02  on 396  degrees of freedom
## AIC: 1141.5
##
## Number of Fisher Scoring iterations: 2
```

The cubic model has a much lower MSE/CV error, so most data points have less error from the true values, for both LOOCV and 10-Fold than my initial preferred model in (a) and model 2. This indicates that the cubic model is a much better fit, as expected, seeing as the data was generated from this model.

The t-value is also much larger for the cubic model than model 1 and 2 which translates to a much lower p-value and thus a more decisive rejection of the null hypotheis. The coefficients are also all negative which indicates negative relationships between the response and predictor. Std. Error was also much lower for the cubic model than the other two.