1. Statistical learning methods

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.

An overfitting will be caused by flexible model as the size of sample is small. To avoid this, we must use an inflexible model as there will be a s small reduction in bias and bigger inflation in mean. Apart from this the computational cost will be higher in a flexible method rather than an inflexible method.

(b) The sample size n is extremely large, and the number of predictors p is small.

As the sample size is large there is less chances of overfit and it will tend to reduce bias. Thus we need to use a flexible model. This flexible model allows to extract extra information as there is large sample size. As the number of predictors are less the computational cost is less.

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

A flexible model will perform better in general because it'll be necessary to use a flexible model to find the non-linear effect. By using this method we can reduce error as the method will be capable for capturing non-liner relationships.

(d) The standard deviation of the error terms, i.e. σ = sd(ε), is extremely high.

In this case we need to use an inflexible method because if the standard deviation of error is high, the variance of error will me square times higher than that. Thus ,higher the variance of error the sample will consist of more noise between the relation of response and predictor.

---

2. Bayes' rule

Given a dataset including 20 observations (S_1, ..., S_20) about the temperature (i.e. hot or cool) for playing golf (i.e. yes or no), you are required to use the Bayes' rule to calculate by hand the probability of playing golf according to the temperature, i.e. P(Play Golf | Temperature).

No. of Days	Temperature	Play Golf
S_1	Cool	Yes
S_2	Hot	No
S_3	Hot	Yes
S_4	Hot	No
S_5	Cool	Yes
S_6	Cool	Yes
S_7	Hot	No
S_8	Cool	Yes
S_9	Hot	No
S_10	Hot	Yes
S_11	Hot	No
S_12	Hot	No
S_13	Hot	Yes
S_14	Cool	No
S_15	Hot	No
S_16	Hot	No
S_17	Cool	Yes
S_18	Cool	No
S_19	Cool	No
S_20	Hot	No

The Data is of 20 records

- P(E = Cool) = 8/20 = 0.4
- P(E=Hot)= 12/20 = 0.6

- P(H=Yes) = 8/20= 0.4
- P(H=No) = 12/20= 0.6

- P(E= Cool | H= Yes) = 5/8 = 0.625
- P(E= Hot | H = Yes) = 3/8 = 0.375
- P(E=Cool | H= No) = 3/12 = 0.25
- P(E=Hot | H= No) = 9/12 = 0.75

Now Using Bayes' Rule

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

True Positive:

$$P(H = Yes \mid E = Cool) = \frac{P(E= Cool \mid H= Yes) \cdot P(H= Yes)}{P(E= Cool)} = \frac{0.625 \times 0.4}{0.4} = \mathbf{0.625}$$

False Negative:

P (H = Yes | E = Hot) = P(E= Hot | H= Yes). P(H= Yes) = 0.375 x 0.4 = **0.25**
P(E= Hot) 0.6

False Positive:
P (H = No | E = Cool) = P(E= Cool | H= No). P(H= No) = 0.25 x 0.6 = **0.375**
P(E= Cool) 0.4

True Negative:
P (H = No | E = Hot) = P(E= Hot | H= No). P(H= No) = 0.75 x 0.6 = **0.75**
P(E= Hot) 0.6

---

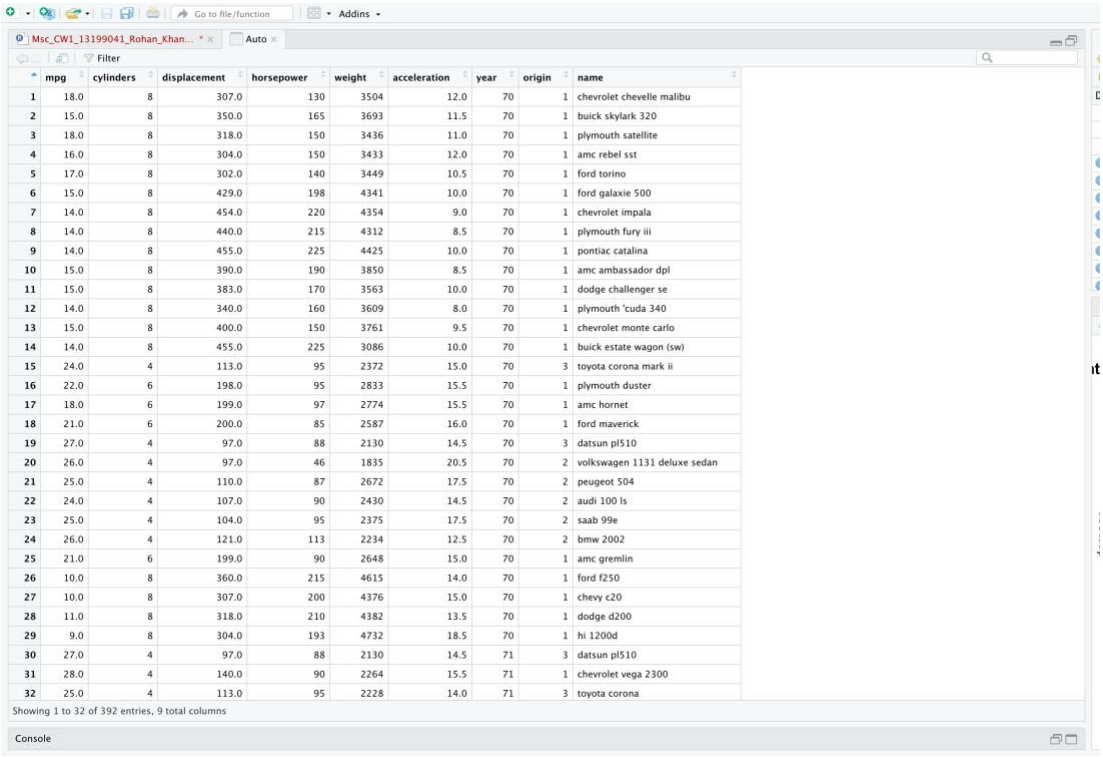
3)This question involves the Auto dataset included in the "ISLR" package.

(a) Which of the predictors are quantitative, and which are qualitative?

(i) The following code were perform to load the packages in library

```
install.packages("ISLR")
library(ISLR)
install.packages("Auto")
View(Auto)
```

The following result was obtained:



Msc_CW1_13199041_Rohan_Khan... | Auto

Filter

	mpg	cylinders	displacement	horsepower	weight	acceleration	year	origin	name
1	18.0	8	307.0	130	3504	12.0	70	1	chevrolet chevelle malibu
2	15.0	8	350.0	165	3693	11.5	70	1	buick skylark 320
3	18.0	8	318.0	150	3436	11.0	70	1	plymouth satellite
4	16.0	8	304.0	150	3433	12.0	70	1	amc rebel sst
5	17.0	8	302.0	140	3449	10.5	70	1	ford torino
6	15.0	8	429.0	198	4341	10.0	70	1	ford galaxie 500
7	14.0	8	454.0	220	4354	9.0	70	1	chevrolet impala
8	14.0	8	440.0	215	4312	8.5	70	1	plymouth fury iii
9	14.0	8	455.0	225	4425	10.0	70	1	pontiac catalina
10	15.0	8	390.0	190	3850	8.5	70	1	amc ambassador dpl
11	15.0	8	383.0	170	3563	10.0	70	1	dodge challenger se
12	14.0	8	340.0	160	3609	8.0	70	1	plymouth 'cuda 340
13	15.0	8	400.0	150	3761	9.5	70	1	chevrolet monte carlo
14	14.0	8	455.0	225	3086	10.0	70	1	buick estate wagon (sw)
15	24.0	4	113.0	95	2372	15.0	70	3	toyota corona mark ii
16	22.0	6	198.0	95	2833	15.5	70	1	plymouth duster
17	18.0	6	199.0	97	2774	15.5	70	1	amc hornet
18	21.0	6	200.0	85	2587	16.0	70	1	ford maverick
19	27.0	4	97.0	88	2130	14.5	70	3	datsun pl510
20	26.0	4	97.0	46	1835	20.5	70	2	volkswagen 1131 deluxe sedan
21	25.0	4	110.0	87	2672	17.5	70	2	peugeot 504
22	24.0	4	107.0	90	2430	14.5	70	2	audi 100 ls
23	25.0	4	104.0	95	2375	17.5	70	2	saab 99e
24	26.0	4	121.0	113	2234	12.5	70	2	bmw 2002
25	21.0	6	199.0	90	2648	15.0	70	1	amc gremlin
26	10.0	8	360.0	215	4615	14.0	70	1	ford f250
27	10.0	8	307.0	200	4376	15.0	70	1	chevy c20
28	11.0	8	318.0	210	4382	13.5	70	1	dodge d200
29	9.0	8	304.0	193	4732	18.5	70	1	hi 1200d
30	27.0	4	97.0	88	2130	14.5	71	3	datsun pl510
31	28.0	4	140.0	90	2264	15.5	71	1	chevrolet vega 2300
32	25.0	4	113.0	95	2228	14.0	71	3	toyota corona

Showing 1 to 32 of 392 entries, 9 total columns

Console

(ii) Then we use the str() command to identify the features of data set
Code:
str(Auto)

The following result was obtained:

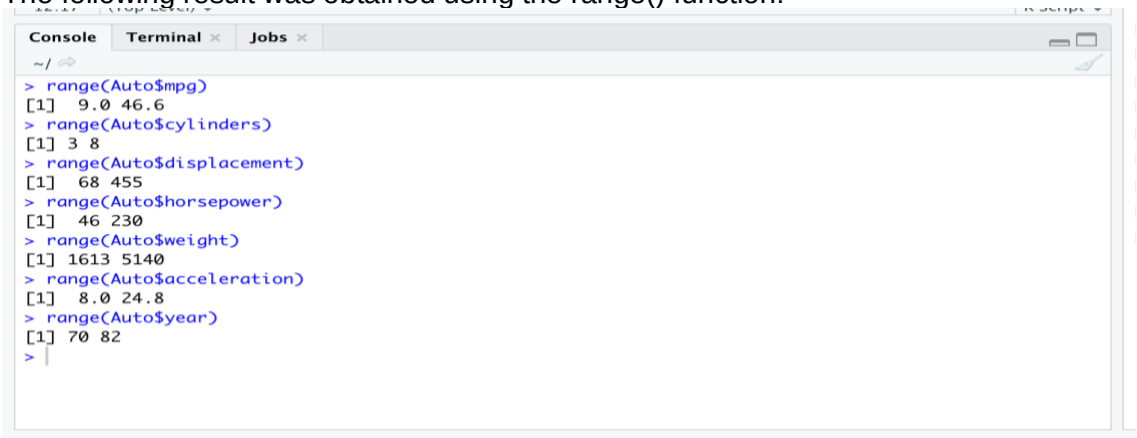
Thus form the data set we can observe that the **Quantitative Predictors** are **"mpg", "cylinders", "displacement", "horsepower", "weight", "acceleration", "year"** and the **Qualitative Predictors** are **"origin" (as 1 <- American; 2 <- European, 3 <- Japanese)** and **"name"**

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

(i) The following code were perform to find the range for each quantitative predictor:

range(Auto$mpg)
range(Auto$cylinders)
range(Auto$displacement)
range(Auto$horsepower)
range(Auto$weight)
range(Auto$acceleration)
range(Auto$year)

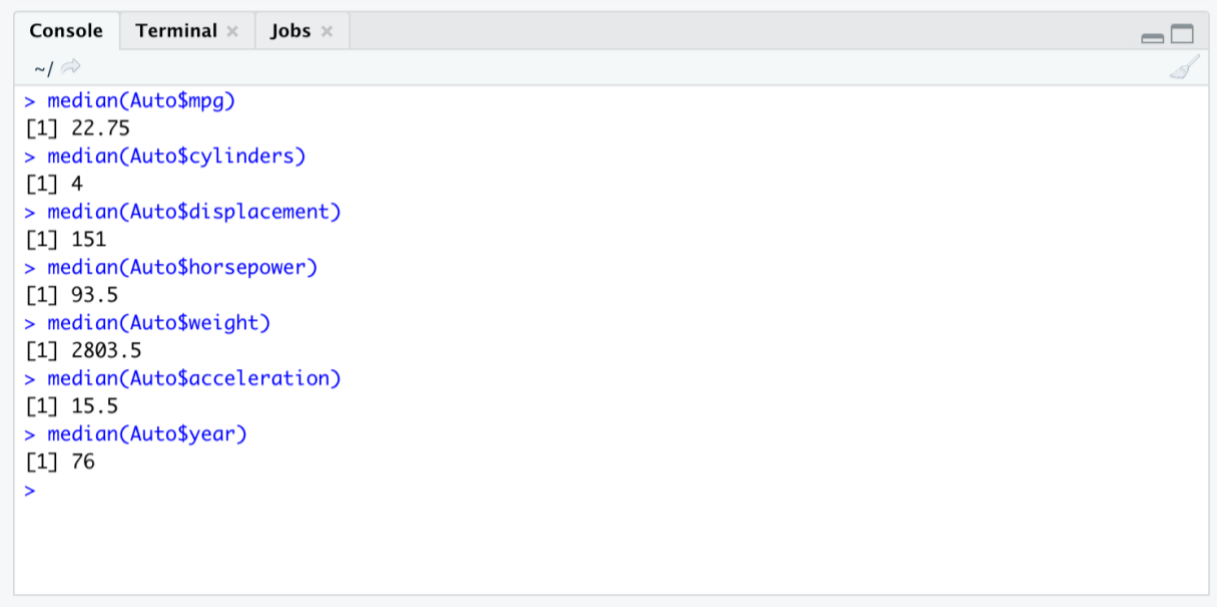
The following result was obtained using the range() function:



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

(i) The following codes were performed to find out median of each quantitative predictor:

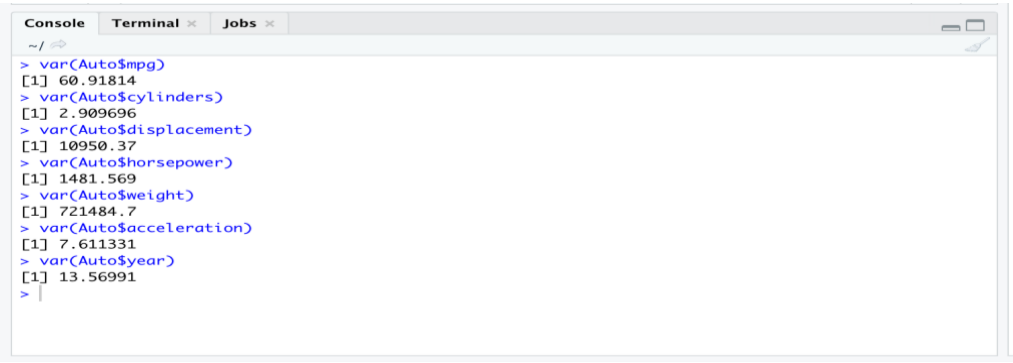
median(Auto$mpg)
median(Auto$cylinders)
median(Auto$displacement)
median(Auto$horsepower)
median(Auto$weight)
median(Auto$acceleration)
median(Auto$year)
The following result was obtained using the median() function:



(ii) The following codes were performed to find out variance of each quantitative predictor

var(Auto$mpg)
var(Auto$cylinders)
var(Auto$displacement)
var(Auto$horsepower)
var(Auto$weight)
var(Auto$acceleration)
var(Auto$year)

The following result was obtained using the var() function:

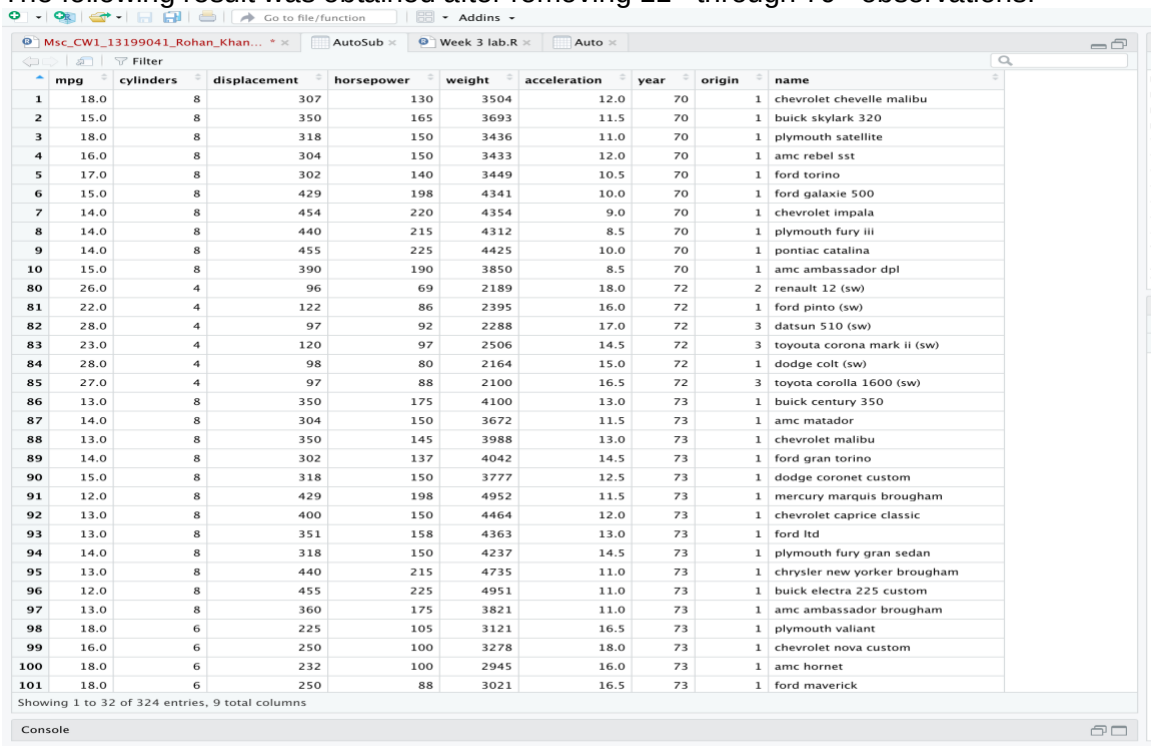


(d) Now remove the 11th through 79th observations (inclusive) in the dataset. What is the range, median, and variance of each predictor in the subset of the data that remains?

(i) The following codes were performed to remove 11th through 79th observations:

```
AutoSub <- Auto[-c(11:78),]
View(AutoSub)
```

The following result was obtained after removing 11th through 79th observations:

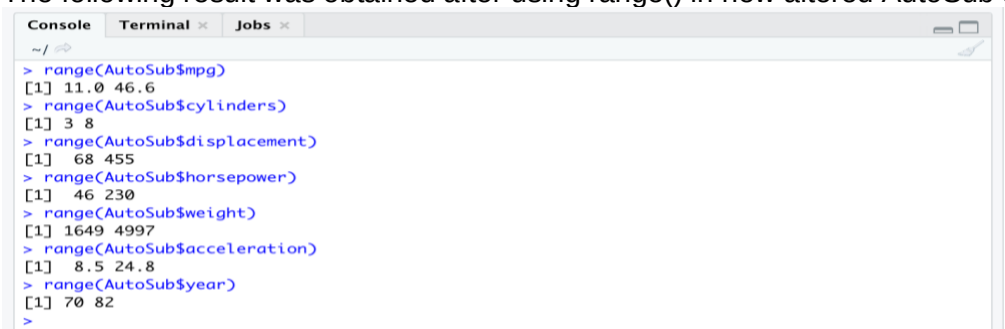


	mpg	cylinders	displacement	horsepower	weight	acceleration	year	origin	name
1	18.0	8	307	130	3504	12.0	70	1	chevrolet chevelle malibu
2	15.0	8	350	165	3693	11.5	70	1	buick skylark 320
3	18.0	8	318	150	3436	11.0	70	1	plymouth satellite
4	16.0	8	304	150	3433	12.0	70	1	amc rebel sst
5	17.0	8	302	140	3449	10.5	70	1	ford torino
6	15.0	8	429	198	4341	10.0	70	1	ford galaxie 500
7	14.0	8	454	220	4354	9.0	70	1	chevrolet impala
8	14.0	8	440	215	4312	8.5	70	1	plymouth fury iii
9	14.0	8	455	225	4425	10.0	70	1	pontiac catalina
10	15.0	8	390	190	3850	8.5	70	1	amc ambassador dpl
80	26.0	4	96	69	2189	18.0	72	2	renault 12 (sw)
81	22.0	4	122	86	2395	16.0	72	1	ford pinto (sw)
82	28.0	4	97	92	2288	17.0	72	3	datsun 510 (sw)
83	23.0	4	120	97	2506	14.5	72	3	toyouta corona mark ii (sw)
84	28.0	4	98	80	2164	15.0	72	1	dodge colt (sw)
85	27.0	4	97	88	2100	16.5	72	3	toyota corolla 1600 (sw)
86	13.0	8	350	175	4100	13.0	73	1	buick century 350
87	14.0	8	304	150	3672	11.5	73	1	amc matador
88	13.0	8	350	145	3988	13.0	73	1	chevrolet malibu
89	14.0	8	302	137	4042	14.5	73	1	ford gran torino
90	15.0	8	318	150	3777	12.5	73	1	dodge coronet custom
91	12.0	8	429	198	4952	11.5	73	1	mercury marquis brougham
92	13.0	8	400	150	4464	12.0	73	1	chevrolet caprice classic
93	13.0	8	351	158	4363	13.0	73	1	ford ltd
94	14.0	8	318	150	4237	14.5	73	1	plymouth fury gran sedan
95	13.0	8	440	215	4735	11.0	73	1	chrysler new yorker brougham
96	12.0	8	455	225	4951	11.0	73	1	buick electra 225 custom
97	13.0	8	360	175	3821	11.0	73	1	amc ambassador brougham
98	18.0	6	225	105	3121	16.5	73	1	plymouth valiant
99	16.0	6	250	100	3278	18.0	73	1	chevrolet nova custom
100	18.0	6	232	100	2945	16.0	73	1	amc hornet
101	18.0	6	250	88	3021	16.5	73	1	ford maverick

(ii) The following codes were performed to find out range of each quantitative predictor

```
range(AutoSub$mpg)
range(AutoSub$cylinders)
range(AutoSub$displacement)
range(AutoSub$horsepower)
range(AutoSub$weight)
range(AutoSub$acceleration)
range(AutoSub$year)
```

The following result was obtained after using range() in new altered AutoSub dataset:

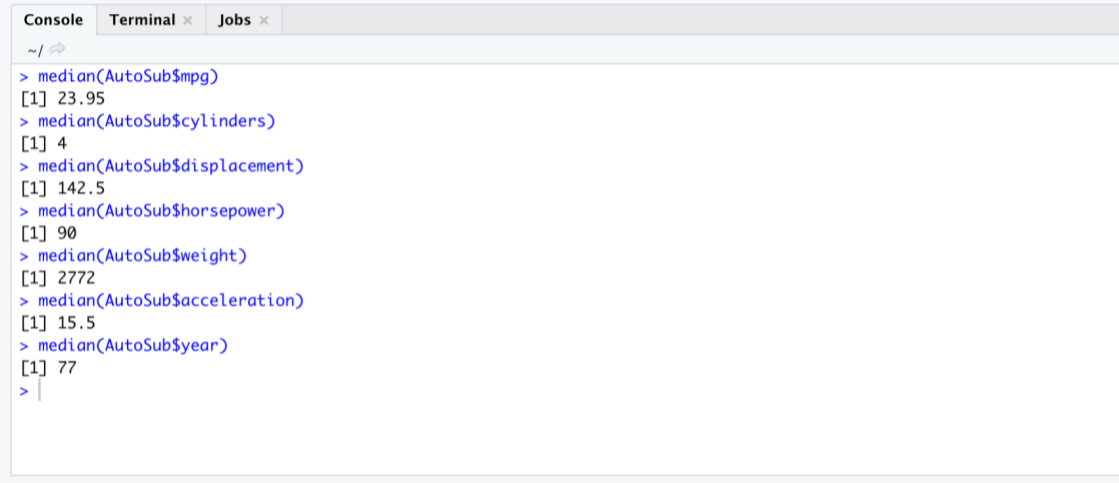


```
> range(AutoSub$mpg)
[1] 11.0 46.6
> range(AutoSub$cylinders)
[1] 3 8
> range(AutoSub$displacement)
[1]  68 455
> range(AutoSub$horsepower)
[1]  46 230
> range(AutoSub$weight)
[1] 1649 4997
> range(AutoSub$acceleration)
[1]  8.5 24.8
> range(AutoSub$year)
[1] 70 82
>
```

(iii) The following codes were performed to find out median of each quantitative predictor

```
median(AutoSub$mpg)
median(AutoSub$cylinders)
median(AutoSub$displacement)
median(AutoSub$horsepower)
median(AutoSub$weight)
median(AutoSub$acceleration)
median(AutoSub$year)
```

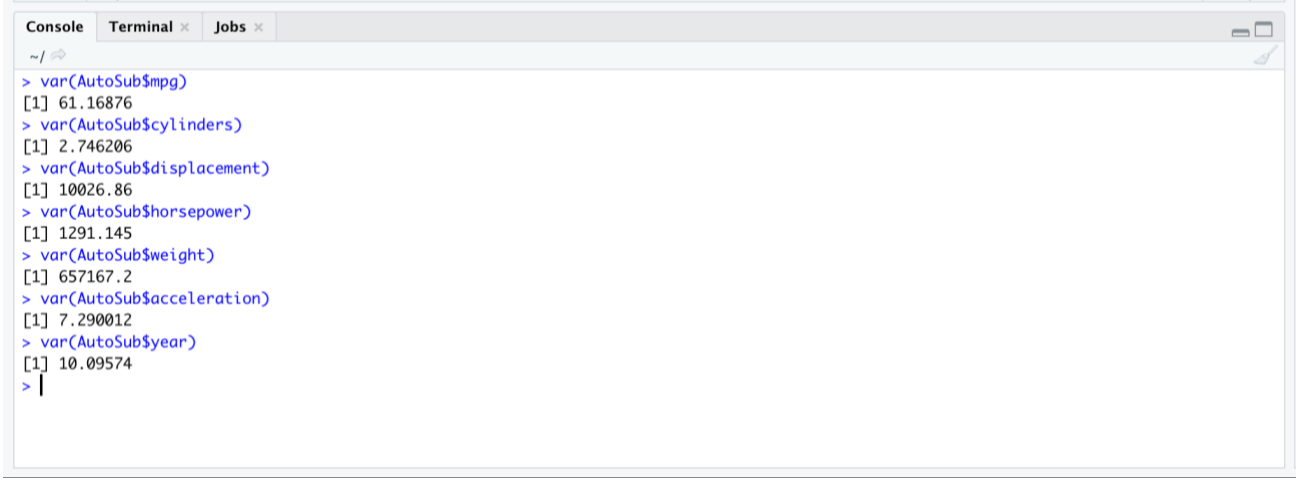
The following result was obtained after using median() in new altered AutoSub dataset:



(iv) The following codes were performed to find out variance of each quantitative predictor

```
var(AutoSub$mpg)
var(AutoSub$cylinders)
var(AutoSub$displacement)
var(AutoSub$horsepower)
var(AutoSub$weight)
var(AutoSub$acceleration)
var(AutoSub$year)
```

The following result was obtained after using var() in new altered AutoSub dataset:

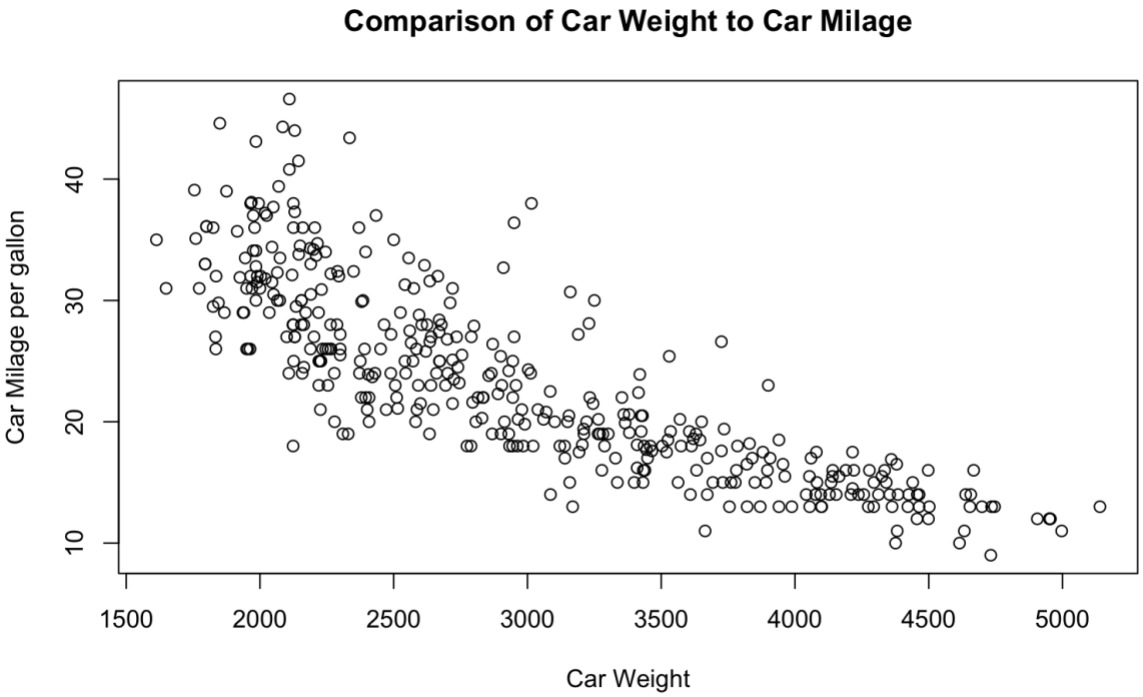


(e) Using the full data set, investigate the relationship between individual predictors with the target response gas mileage (mpg) graphically. Comment on your findings.

(i) We will compare weight of cars with the milage of cars

Code:
Auto.mpg1 <- plot(Auto$weight, Auto$mpg, main = "Comparison of Car Weight to Car Milage", xlab = "Car Weight", ylab = "Car Milage per gallon")

The following result was obtained

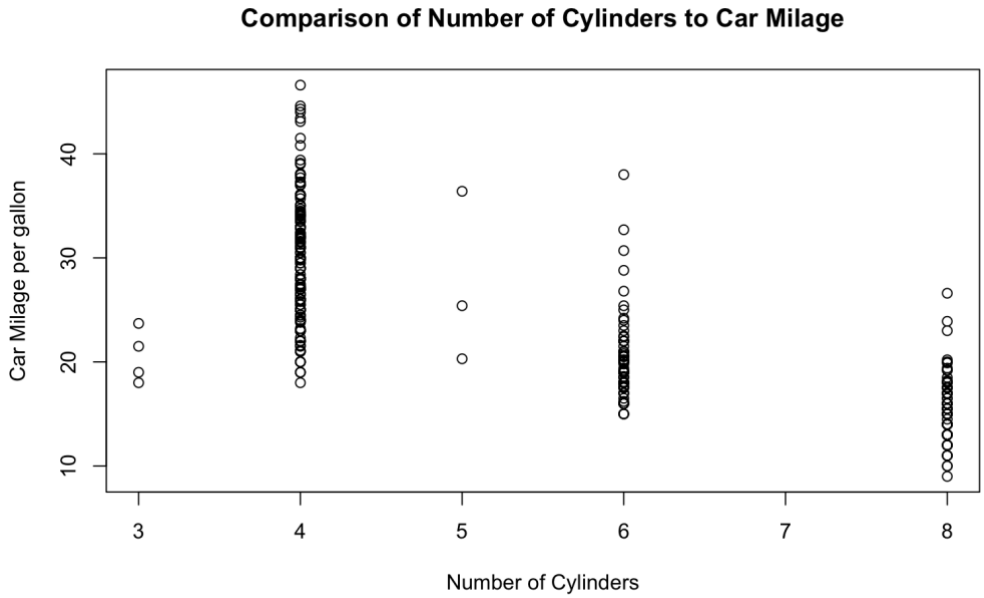


From the above graph we can note that, as the weight of the cars increases the milage per gallon decreases. We can note that cars weight between 1500-2500 lbs. give better milage than the cars weight between 4000 to 5000 lbs.

(ii) We will compare the number of cylinders with the milage of cars

Code:
Auto.mpg2 <- plot(Auto$cylinders, Auto$mpg, main = "Comparison of Number of Cylinders to Car Milage", xlab = "Number of Cylinders", ylab = "Car Milage per gallon")

The following result was obtained



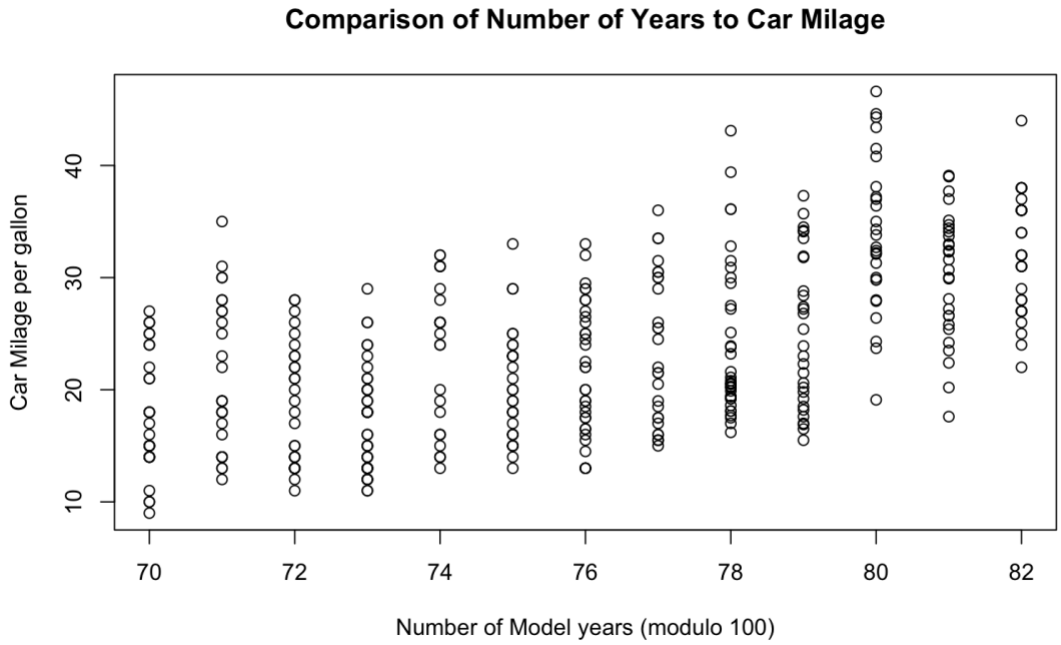
From the above graph we can note that, as the number of cylinders in the cars increases the milage per gallon decreases. We can note that 4 cylinder cars give better milage than a 8 cylinder cars.

(iii) We will compare the number of years with the milage of cars

Code:

```
Auto.mpg3 <- plot(Auto$year, Auto$mpg, main = "Comparison of Number of Years to Car Milage",xlab = "Number of Model years (modulo 100)", ylab = "Car Milage per gallon")
```

The following result was obtained

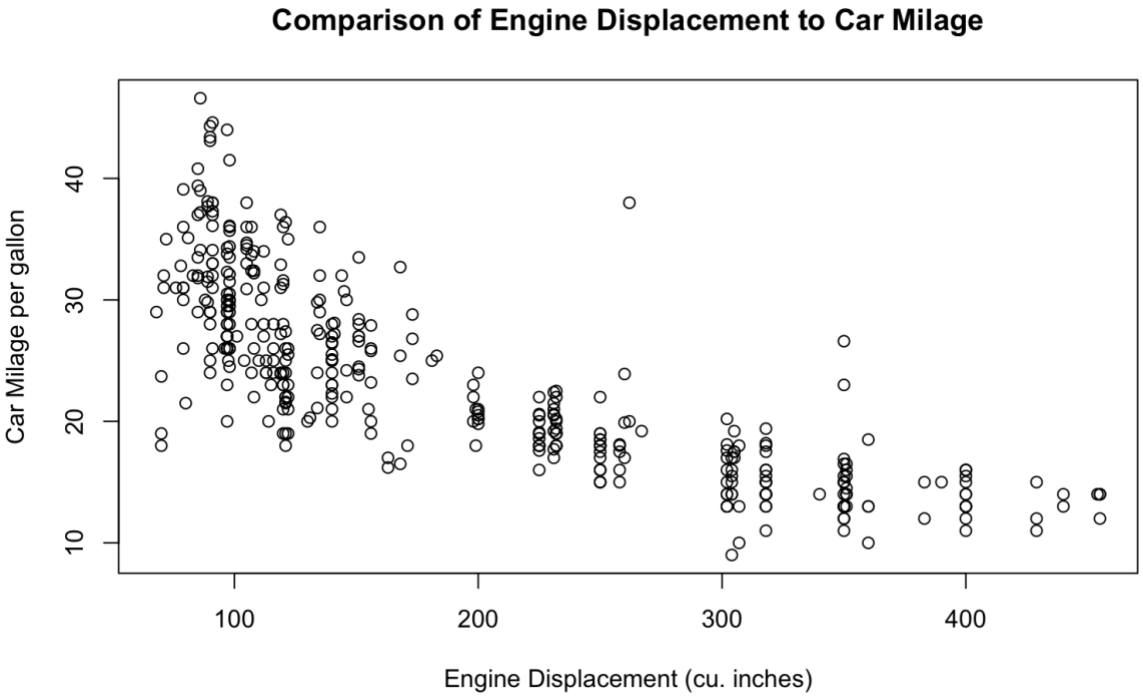


From the above graph we can note that, as the age of cars increases the milage per gallon also increases.

(iv) We will compare the Engine Displacement with the milage of cars

Code:
```
Auto.mpg4 <- plot(Auto$displacement, Auto$mpg, main = "Comparison of Engine Displacement to Car Milage", xlab = "Engine Displacement (cu. inches)", ylab = "Car Milage per gallon")
```

The following result was obtained

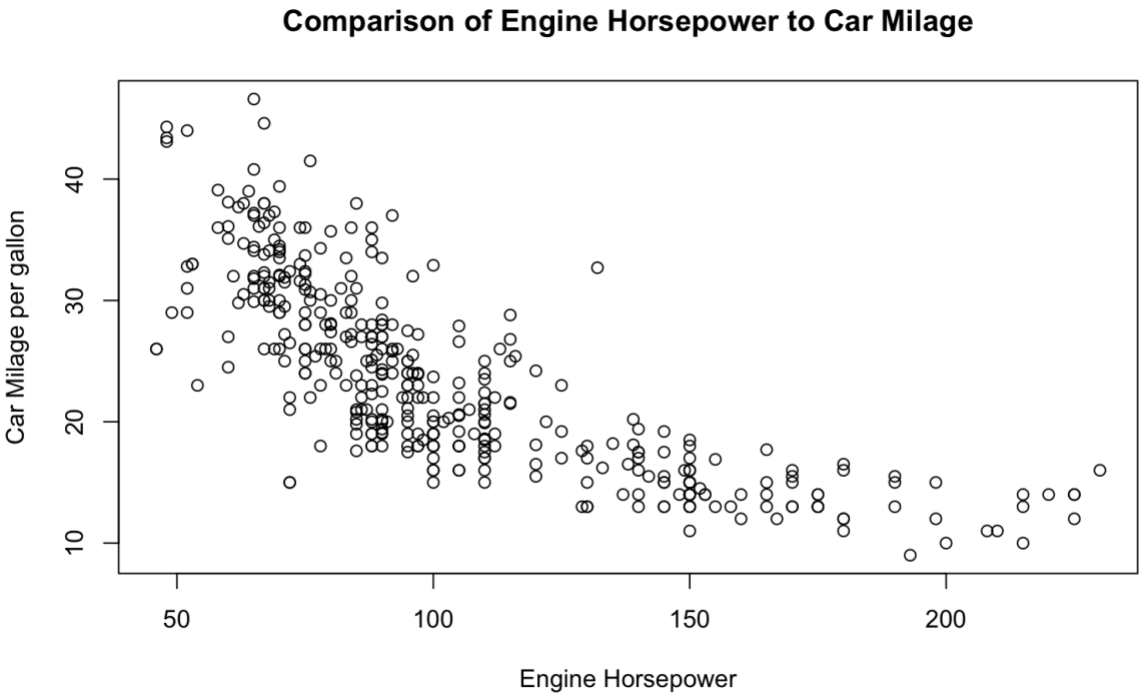


From the above graph we can note that, as the engine displacement in the cars increases the milage per gallon decreases. We can note that cars with 100 cu. inches gives better milage that cars above 300 cu. inches.

(v) We will compare the Engine Horsepower with the milage of cars

Code:
```
Auto.mpg5 <- plot(Auto$horsepower, Auto$mpg, main = "Comparison of Engine Horsepower to Car Milage", xlab = "Engine Horsepower", ylab = "Car Milage per gallon")
```

The following result was obtained



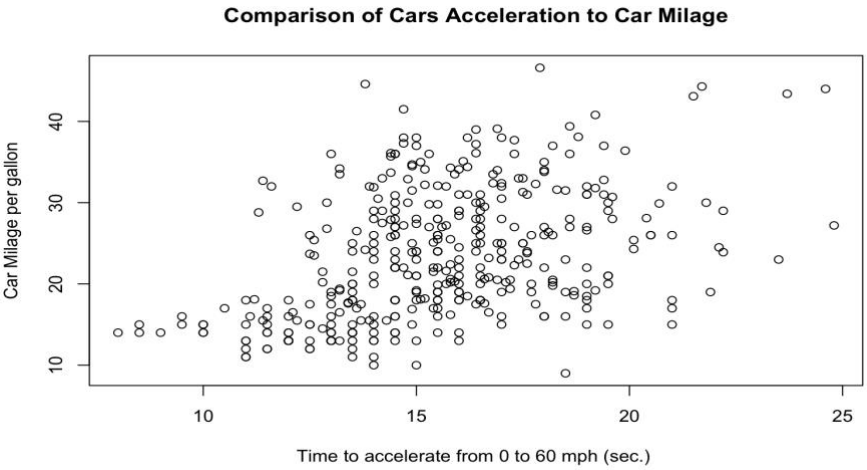
From the above graph we can note that, as the engine horsepower in the cars increases the milage per gallon decreases. We can note that between 50 to 100 horsepower the cars give a better milage compared to cars above 150 horsepower.

(vi) We will compare the Car Acceleration (time taken from 0-60mph in seconds) with the milage of cars

Code:

```
Auto.mpg6 <- plot(Auto$acceleration, Auto$mpg, main = "Comparison of Cars Acceleration to Car Milage", xlab = "Time to accelerate from 0 to 60 mph (sec.) ", ylab = "Car Milage per gallon")
```

The following result was obtained



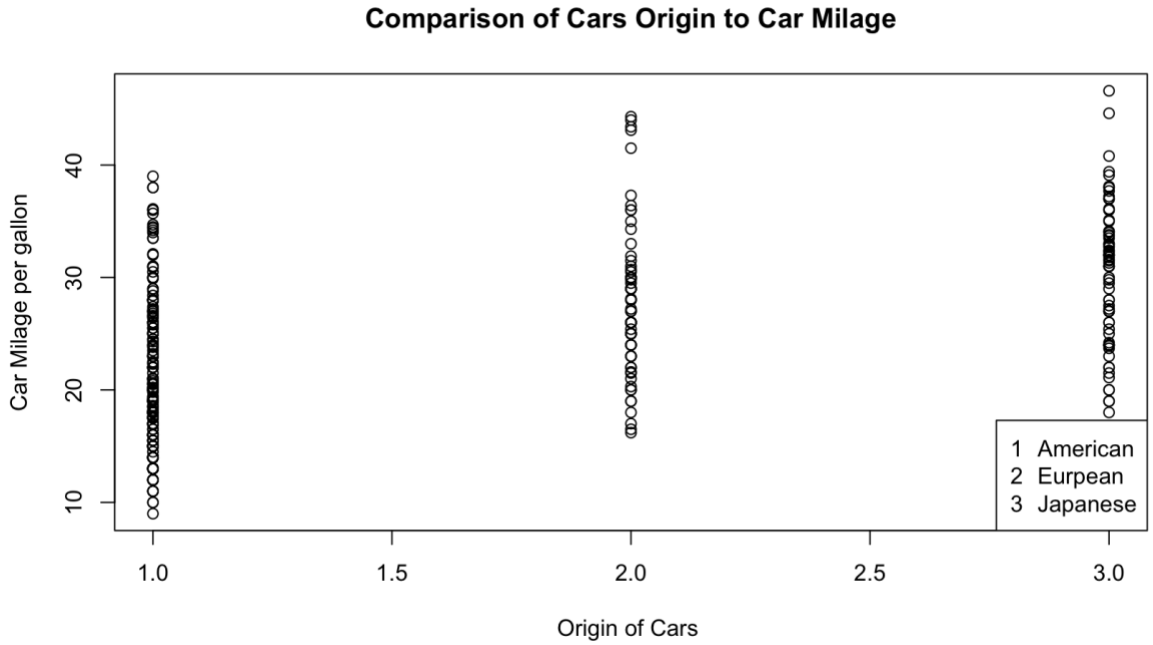
From the above graph we can note that, as the more the time take to accelerate from 0 to 60 mph in the cars the milage per gallon increases. Note that the time above 15 sec. to accelerate give more milage than time taken below 10 sec.

(vii) We will compare the Car Origin with the milage of cars

Code:

```
Auto.mpg7 <- plot(Auto$origin,Auto$mpg, main = "Comparison of Cars Origin to Car Milage",xlab = "Origine of Cars ", ylab = "Car Milage per gallon")
legend("bottomright",pch =c("1","2","3"), col = c("black", "black", "black"),legend = c("American", "Eurpean", "Japanese"))
```

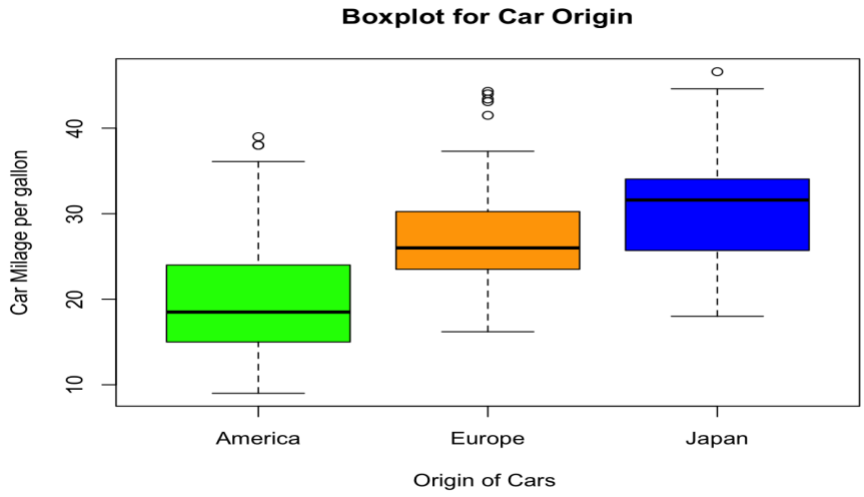
The following result was obtained



Another way for analysing this is by using Boxplot:
Code:

```
boxplot(Auto$mpg~Auto$origin, col= c("green", "orange", "blue"), names= c("America","Europe","Japan"), xlab = "Origin of Cars ", ylab = "Car Milage per gallon", main = "Boxplot for Car Origin")
```

The following result was obtained



From the above graphs we can note that, on an average Japanese cars give better milage than European and American cars.

(f) Suppose that we wish to predict 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.

Yes, other vectors might be useful, but they are not mentioned in this Auto dataset. For an example, maintenance of car plays a key role in giving better milage and also weather and road traffic condition plays an important role in cars milage. Such missing factors can change the demographics of entire dataset.

---

4. Linear regression (24% | 24%)

This question involves the use of simple linear regression on the Auto dataset.

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

We will first perform simple linear regression using lm() function with response as mpg and predictor as Acceleration (time taken from 0-60mph in seconds) :

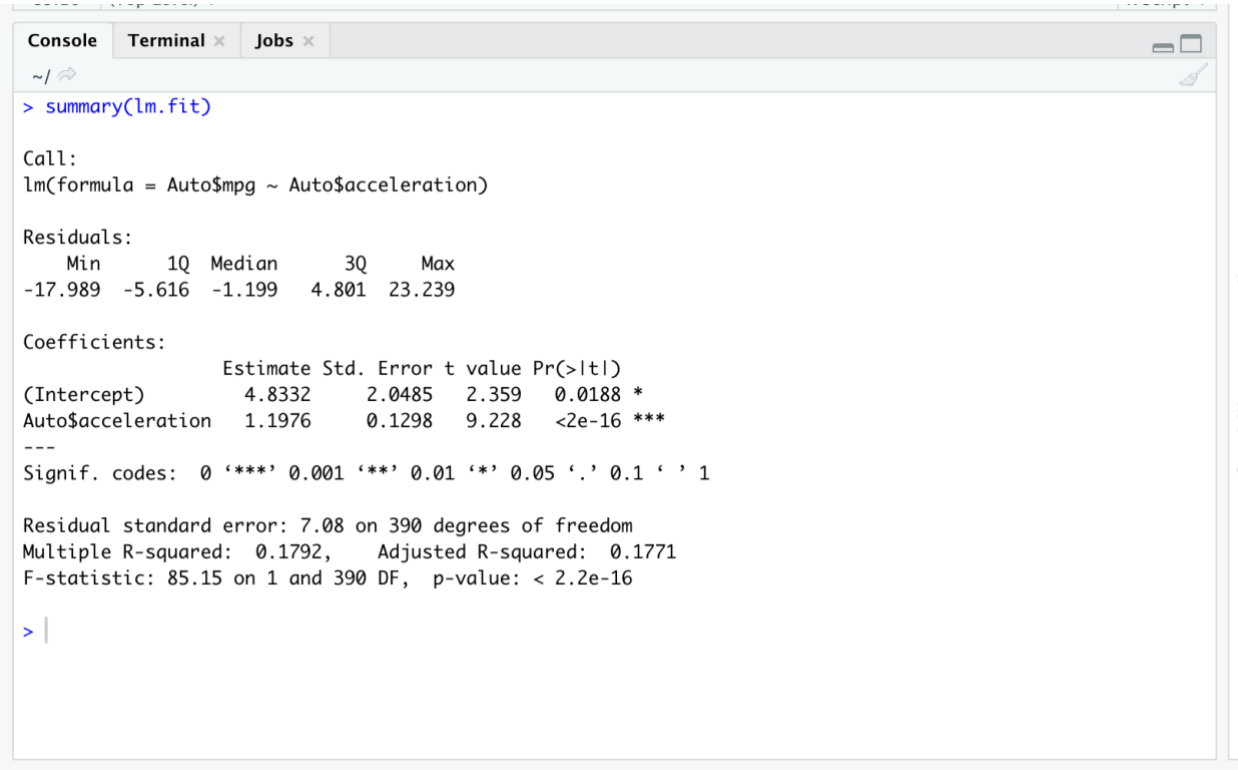
Code:

```
## this code is continued in sequence to Q 3
lm.fit <- lm(Auto$mpg~Auto$acceleration)
lm.fit
```

The following result was obtained

Now we will use the summary() function and obtain the output:

Code:
summary(lm.fit)
The following result was obtained



In the Residual the value of Min and Max are not equidistant from Median, but the 1Q and 3Q are almost equidistant. The correlation is positive as the values are increasing together. As the Std. Error of intercept is not Zero we can say that there is a linear relationship between X & Y, as the T-value of acceleration is high this signifies that x & y are related. X has significant linear relation with target respond. As the P-value of intercept is low proves that it has strong evidence against null hypotheses. The RSE is High and R-Squared is low, so the quality of model fitting is not good as the relation between adjusted R squared is implies that it is only 17.71% between predictor and the response.

i. Is there a relationship between the predictor and the response?

There is no relationship between predictor and the response as the P-value do not match with each other.

ii. How strong is the relationship between the predictor and the response?

The value of Adjusted R^2 indicates that there is approximately 17.71% of variation in mpg response variable is due to the predictor variable of acceleration.

iii. Is the relationship between the predictor and the response positive or negative?

The relationship between the predictor and the response is positive.

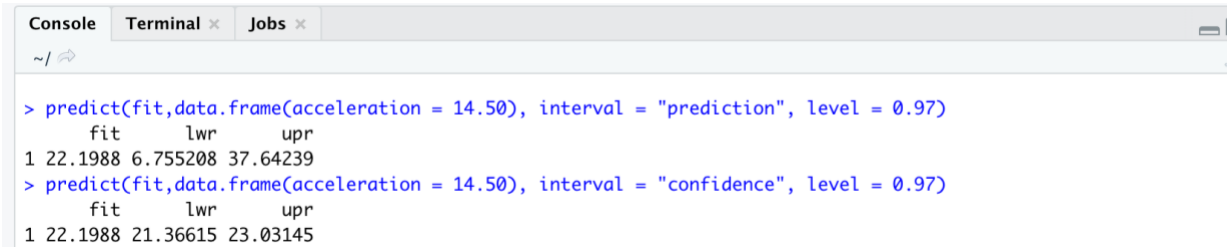
iv. What is the predicted mpg associated with an acceleration of 14.50? What are the associated 97% confidence and prediction intervals?

We will first predict mpg with acceleration of 14.50 with associated 97% confidence interval

Code:

```
predict(lm.fit,data.frame(acceleration = 14.50), interval = "prediction", level = 0.97)
predict(lm.fit,data.frame(acceleration = 14.50), interval = "confidence", level = 0.97)
```

The following results was obtained



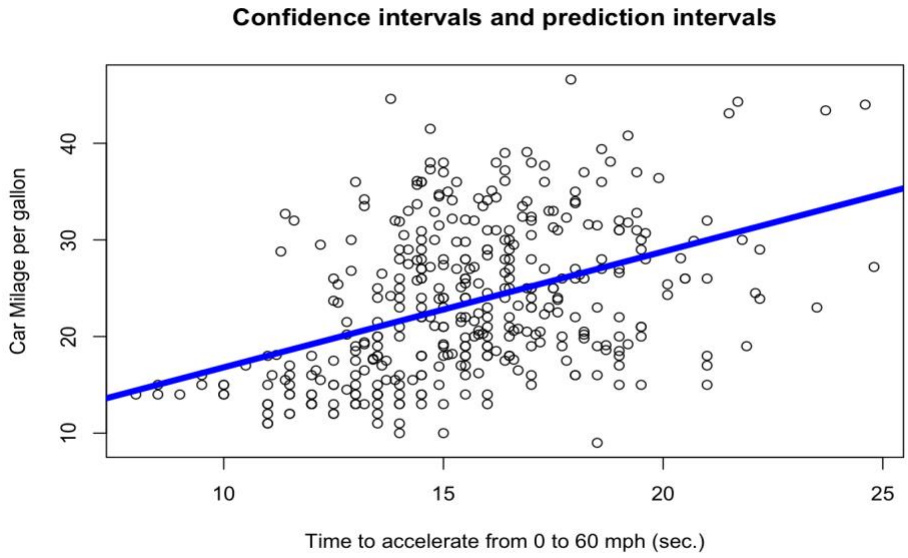
(b) Plot the response and the predictor. Use the abline() function to display the least squares regression line.

The code performed were

Code:

```
attach(Auto)
plot(acceleration,mpg, ylab = "Car Milage per gallon",xlab = "Time to accelerate from 0 to 60 mph (sec.)", main = "Confidence intervals and prediction intervals", )
abline(lm.fit,lwd=5,col="blue")
```

The following results was obtained

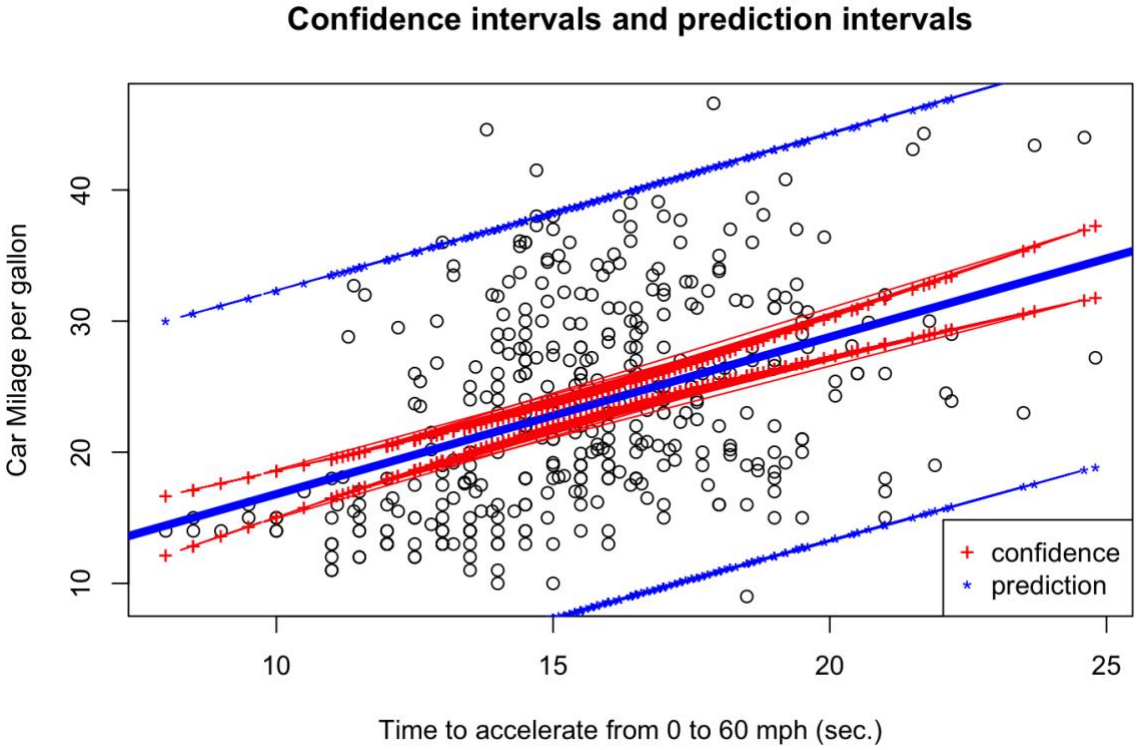


(c) Plot the 97% confidence interval and prediction interval in the same plot as (b) using different colours and legends.

The codes used were :

```
p_pred <- predict(lm.fit,data.frame(acceleration = 14.50), interval = "prediction", level = 0.97)
p_conf <- predict(lm.fit,data.frame(acceleration = 14.50), interval = "confidence", level = 0.97)
attach(Auto)
plot(acceleration,mpg, ylab = "Car Milage per gallon",xlab = "Time to accelerate from 0 to 60 mph (sec.)", main = "Confidence intervals and prediction intervals", )
abline(lm.fit,lwd=5,col="blue")
lines(Auto$acceleration, p_conf[,"lwr"], col="red", type="b", pch="+")
lines(Auto$acceleration, p_conf[,"upr"], col="red", type="b", pch="+")
lines(Auto$acceleration, p_pred[,"upr"], col="blue", type="b", pch="*")
lines(Auto$acceleration, p_pred[,"lwr"], col="blue", type="b", pch="*")
legend("bottomright",pch=c("+","*"), col=c("red","blue"), legend = c("confidence","prediction"))
```

The following results was obtained



---

5. Logistic regression and cross validation

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 predictors (features), such as temperature, humidity, light, CO_2 and humidity ratio, to predict the office room occupancy. The provided training dataset consists of 2,000 observations, whilst the testing dataset consists of 300 observations.

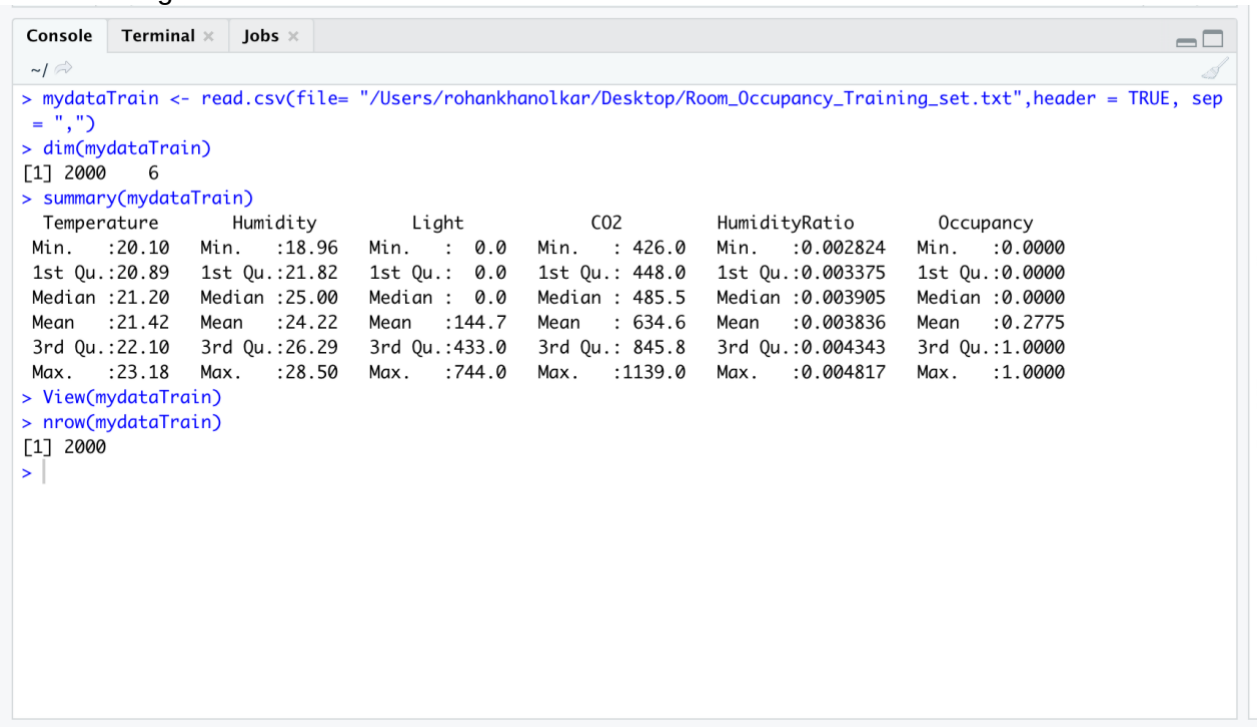
(a) Load the training and testing datasets from corresponding files, and display the statistics about different predictors in the training dataset.

The codes used were loading training data:

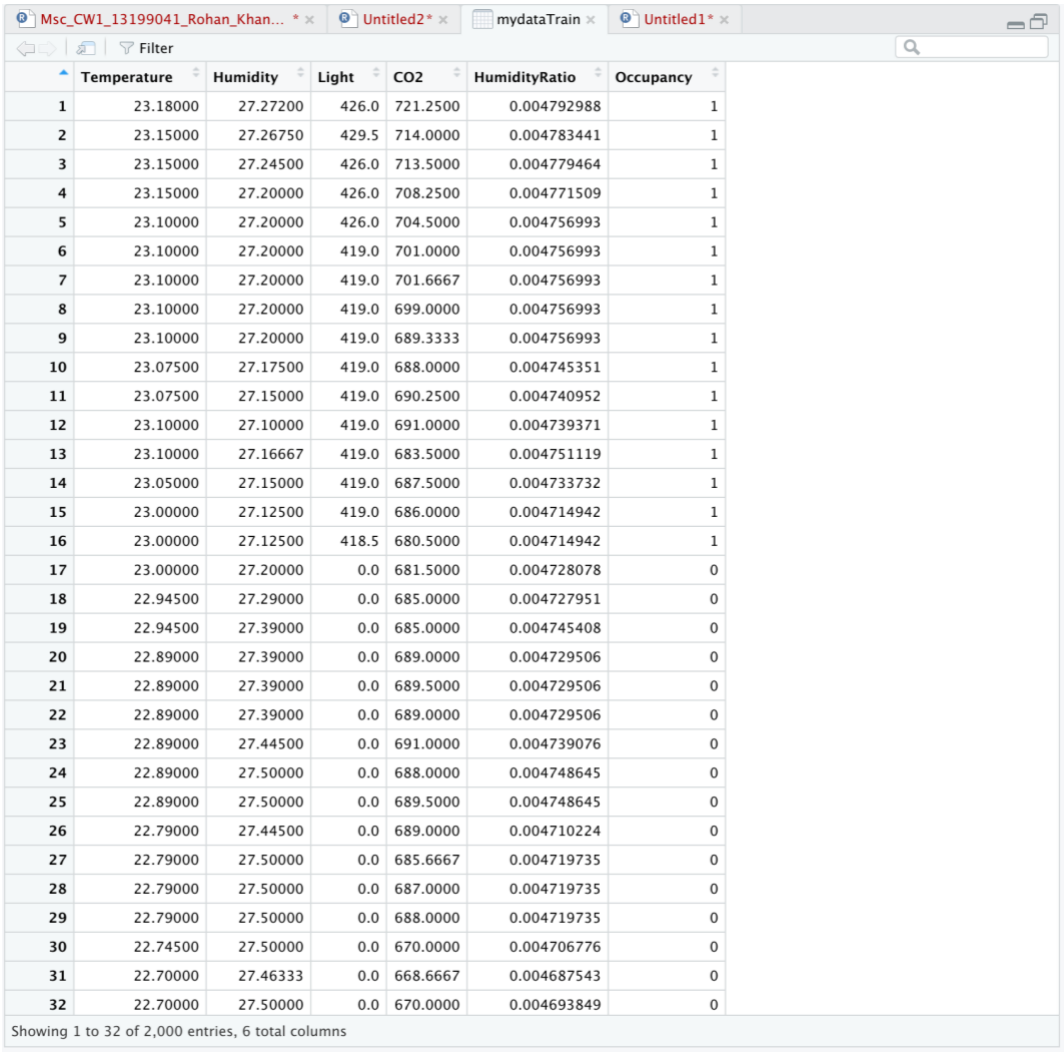
```
## File located on the computer desktop.

mydataTrain <- read.csv(file=
"/Users/rohankhanolkar/Desktop/Room_Occupancy_Training_set.txt",header = TRUE, sep =
",")
```

dim(mydataTrain)
summary(mydataTrain)
nrow(mydataTrain)
View(mydataTrain)

The following results was obtained:



for *View(mydataTrain)* on next page



Msc_CW1_13199041_Rohan_Khan... * | Untitled2* | mydataTrain | Untitled1*

Filter

	Temperature	Humidity	Light	CO2	HumidityRatio	Occupancy
1	23.18000	27.27200	426.0	721.2500	0.004792988	1
2	23.15000	27.26750	429.5	714.0000	0.004783441	1
3	23.15000	27.24500	426.0	713.5000	0.004779464	1
4	23.15000	27.20000	426.0	708.2500	0.004771509	1
5	23.10000	27.20000	426.0	704.5000	0.004756993	1
6	23.10000	27.20000	419.0	701.0000	0.004756993	1
7	23.10000	27.20000	419.0	701.6667	0.004756993	1
8	23.10000	27.20000	419.0	699.0000	0.004756993	1
9	23.10000	27.20000	419.0	689.3333	0.004756993	1
10	23.07500	27.17500	419.0	688.0000	0.004745351	1
11	23.07500	27.15000	419.0	690.2500	0.004740952	1
12	23.10000	27.10000	419.0	691.0000	0.004739371	1
13	23.10000	27.16667	419.0	683.5000	0.004751119	1
14	23.05000	27.15000	419.0	687.5000	0.004733732	1
15	23.00000	27.12500	419.0	686.0000	0.004714942	1
16	23.00000	27.12500	418.5	680.5000	0.004714942	1
17	23.00000	27.20000	0.0	681.5000	0.004728078	0
18	22.94500	27.29000	0.0	685.0000	0.004727951	0
19	22.94500	27.39000	0.0	685.0000	0.004745408	0
20	22.89000	27.39000	0.0	689.0000	0.004729506	0
21	22.89000	27.39000	0.0	689.5000	0.004729506	0
22	22.89000	27.39000	0.0	689.0000	0.004729506	0
23	22.89000	27.44500	0.0	691.0000	0.004739076	0
24	22.89000	27.50000	0.0	688.0000	0.004748645	0
25	22.89000	27.50000	0.0	689.5000	0.004748645	0
26	22.79000	27.44500	0.0	689.0000	0.004710224	0
27	22.79000	27.50000	0.0	685.6667	0.004719735	0
28	22.79000	27.50000	0.0	687.0000	0.004719735	0
29	22.79000	27.50000	0.0	688.0000	0.004719735	0
30	22.74500	27.50000	0.0	670.0000	0.004706776	0
31	22.70000	27.46333	0.0	668.6667	0.004687543	0
32	22.70000	27.50000	0.0	670.0000	0.004693849	0

Showing 1 to 32 of 2,000 entries, 6 total columns

The codes used were loading testing data:

```
mydataTest <- read.csv(file=
"/Users/rohankhanolkar/Desktop/Room_Occupancy_Testing_set.txt",header = TRUE, sep =
",")

dim(mydataTest)
summary(mydataTest)
nrow(mydataTest)
View(mydataTest)
```

The following results was obtained:

```
> mydataTest <- read.csv(file= "/Users/rohankhanolkar/Desktop/Room_Occupancy_Testing_set.txt",header = TRUE, sep =
 ",")
> dim(mydataTest)
[1] 300   6
> summary(mydataTest)
  Temperature       Humidity         Light             CO2         HumidityRatio        Occupancy
 Min.   :20.39   Min.   :21.86   Min.   :  0.00   Min.   : 484.7   Min.   :0.003275   Min.   :0.0
 1st Qu.:20.65   1st Qu.:24.43   1st Qu.:  0.00   1st Qu.: 560.5   1st Qu.:0.003959   1st Qu.:0.0
 Median :21.32   Median :26.32   Median : 44.67   Median : 592.8   Median :0.004335   Median :0.0
 Mean   :21.68   Mean   :26.66   Mean   :207.41   Mean   : 657.1   Mean   :0.004276   Mean   :0.2
 3rd Qu.:22.11   3rd Qu.:28.79   3rd Qu.:429.00   3rd Qu.: 735.9   3rd Qu.:0.004562   3rd Qu.:0.0
 Max.   :24.39   Max.   :36.00   Max.   :756.50   Max.   :1595.7   Max.   :0.005670   Max.   :1.0
> nrow(mydataTest)
[1] 300
> View(mydataTest)
>
```

for *View(mydataTrain)*

Msc_CW1_13199041_Rohan_Khan... * | Untitled2* | mydataTest | Untitled1*

Filter

	Temperature	Humidity	Light	CO2	HumidityRatio	Occupancy
1	21.89000	31.55000	436.50000	1047.0000	0.005129660	0
2	21.89000	31.36000	434.00000	1031.0000	0.005098515	0
3	21.89000	31.12500	432.75000	977.5000	0.005059998	0
4	21.70000	28.50000	279.33333	585.0000	0.004576247	1
5	20.60000	21.86500	454.00000	652.5000	0.003274764	0
6	20.60000	22.20000	442.75000	681.7500	0.003325206	0
7	20.60000	22.26000	444.00000	702.3333	0.003334241	0
8	20.63333	22.26000	444.00000	707.0000	0.003341137	0
9	23.67500	22.74500	289.00000	795.7500	0.004114101	1
10	24.20000	23.02250	497.00000	722.0000	0.004299052	0
11	24.39000	23.39250	236.50000	852.5000	0.004419017	1
12	24.20000	23.70000	630.00000	734.0000	0.004426464	0
13	24.20000	23.74500	690.50000	729.0000	0.004434928	0
14	23.70000	24.43000	87.00000	599.6667	0.004427757	1
15	22.39000	26.00000	191.50000	534.5000	0.004352666	1
16	21.89000	27.89000	279.33333	603.6667	0.004530253	1
17	21.29000	25.34500	328.50000	519.0000	0.003964875	1
18	21.29000	25.49667	499.33333	525.0000	0.003988753	0
19	21.29000	25.46333	510.33333	528.6667	0.003983505	0
20	22.65000	24.89750	726.75000	585.0000	0.004233654	0
21	22.66667	24.99667	737.00000	594.0000	0.004254963	0
22	22.70000	24.97250	723.50000	598.0000	0.004259487	0
23	22.70000	25.00000	707.00000	595.5000	0.004264210	0
24	22.72250	25.17500	720.50000	604.5000	0.004300171	0
25	22.72250	25.04750	714.50000	613.0000	0.004278243	0
26	23.76750	25.67500	654.00000	547.0000	0.004674187	0
27	23.76750	25.74750	668.50000	545.0000	0.004687486	0
28	23.70000	25.89000	680.33333	564.0000	0.004694368	0
29	23.72250	25.97250	664.75000	567.7500	0.004715873	0
30	23.70000	25.89000	675.25000	569.5000	0.004694368	0
31	23.74500	26.00000	669.00000	571.5000	0.004727350	0
32	23.70000	26.07500	667.25000	564.7500	0.004728167	0

Showing 1 to 32 of 300 entries, 6 total columns

(b) Conduct a 10-fold cross validation to evaluate the predictive accuracy of a logistic regression model that uses Temperature as the only predictor. Report the average accuracy and AUROC value obtained over the 10-fold cross validation. Set the value of random seed as "100" when generating fold indices. Consider the predictive label equals to 1, if the predictive probability is greater than 0.5.

(i) First we load the Library and then we estimate the logistic regression model using glm() function and then create the summary :

Codes:

```
library(caret)
library(ROCR)
glm.fit <- glm(Occupancy ~ Temperature, data = mydataTrain, family = binomial)
summary(glm.fit)
```

The following results was obtained:

```
> glm.fit <- glm(Occupancy ~ Temperature, data = mydataTrain, family = binomial)
> summary(glm.fit)

Call:
glm(formula = Occupancy ~ Temperature, family = binomial, data = mydataTrain)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.2180  -0.4862  -0.3368   0.6739   2.5871

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -54.4413     2.4166  -22.53   <2e-16 ***
Temperature   2.4701     0.1107   22.32   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2362.3  on 1999  degrees of freedom
Residual deviance: 1548.5  on 1998  degrees of freedom
AIC: 1552.5

Number of Fisher Scoring iterations: 5
```

(ii) Now we use model to predict the training dataset and store the results to a vector **:**

Code:

```
glm.probs <- predict(glm.fit, mydataTrain, type = "response")
head(glm.probs)
```

The following results was obtained:

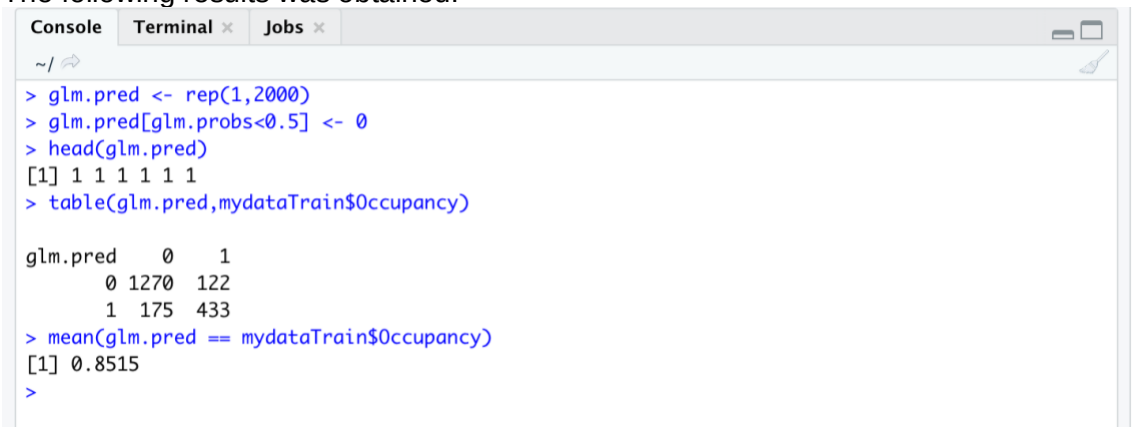
```
> glm.probs <- predict(glm.fit, mydataTrain, type = "response")
> head(glm.probs)
        1         2         3         4         5         6
0.9434899 0.9394067 0.9394067 0.9394067 0.9319832 0.9319832
>
```

(iii) Now we create another vector to show predictive label equals to 1, if the predictive probability is greater than 0.5

Codes:

```
glm.pred <- rep(1,2000)
glm.pred[glm.probs<0.5] <- 0
head(glm.pred)
table(glm.pred,mydataTrain$Occupancy)
mean(glm.pred == mydataTrain$Occupancy)
```

The following results was obtained:

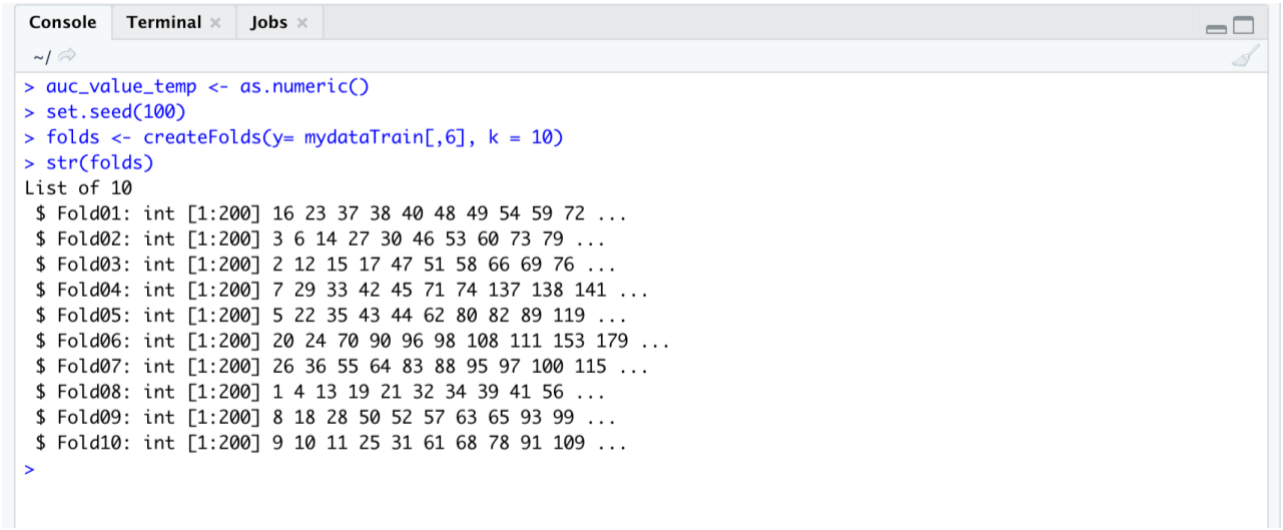


(iv) Now we conduct a 10-fold cross validation to evaluate the predictive accuracy of a logistic regression model that uses Temperature as the only predictor and set.seed(100):

Code:

```
auc_value_temp <- as.numeric()
set.seed(100)
folds <- createFolds(y= mydataTrain[,6], k = 10)
str(folds)
```

The following results was obtained:



(v) Now we conduct 10 fold cross validation find the average accuracy and AUROC value.

Code:

```
for (i in 1:10) {
  fold_cv_test <- mydataTrain[folds[[i]],]
  fold_cv_train <- mydataTrain[-folds[[i]],]
  trained_model_Temperature <- glm(Occupancy ~ Temperature, data = fold_cv_train, family
= binomial)
  pred_prob_Temperature <- predict(trained_model_Temperature, fold_cv_test, type =
"response")
  pr_Temperature <- prediction(pred_prob_Temperature, fold_cv_test$Occupancy)
  auroc_Temperature <- performance(pr_Temperature, measure = "auc")
  auroc_Temperature <- auroc_Temperature@y.values[[1]]
print(auroc_Temperature)
auc_value_temp <- append(auc_value_temp,auroc_Temperature)
}
print(auc_value_temp)
print(mean(auc_value_temp))
```

The following results was obtained:

```
> for (i in 1:10) {
+   fold_cv_test <- mydataTrain[folds[[i]],]
+   fold_cv_train <- mydataTrain[-folds[[i]],]
+   trained_model_Temperature <- glm(Occupancy ~ Temperature, data = fold_cv_train, family = binomial)
+   pred_prob_Temperature <- predict(trained_model_Temperature, fold_cv_test, type = "response")
+   pr_Temperature <- prediction(pred_prob_Temperature, fold_cv_test$Occupancy)
+   auroc_Temperature <- performance(pr_Temperature, measure = "auc")
+   auroc_Temperature <- auroc_Temperature@y.values[[1]]
+   print(auroc_Temperature)
+   auc_value_temp <- append(auc_value_temp,auroc_Temperature)
+ }
[1] 0.8438
[1] 0.9140359
[1] 0.8586085
[1] 0.8805031
[1] 0.863101
[1] 0.8930484
[1] 0.867897
[1] 0.8278063
[1] 0.8541019
[1] 0.8904075
> print(auc_value_temp)
 [1] 0.8438000 0.9140359 0.8586085 0.8805031 0.8631010 0.8930484 0.8678970 0.8278063 0.8541019 0.8904075
> print(mean(auc_value_temp))
[1] 0.869331
```

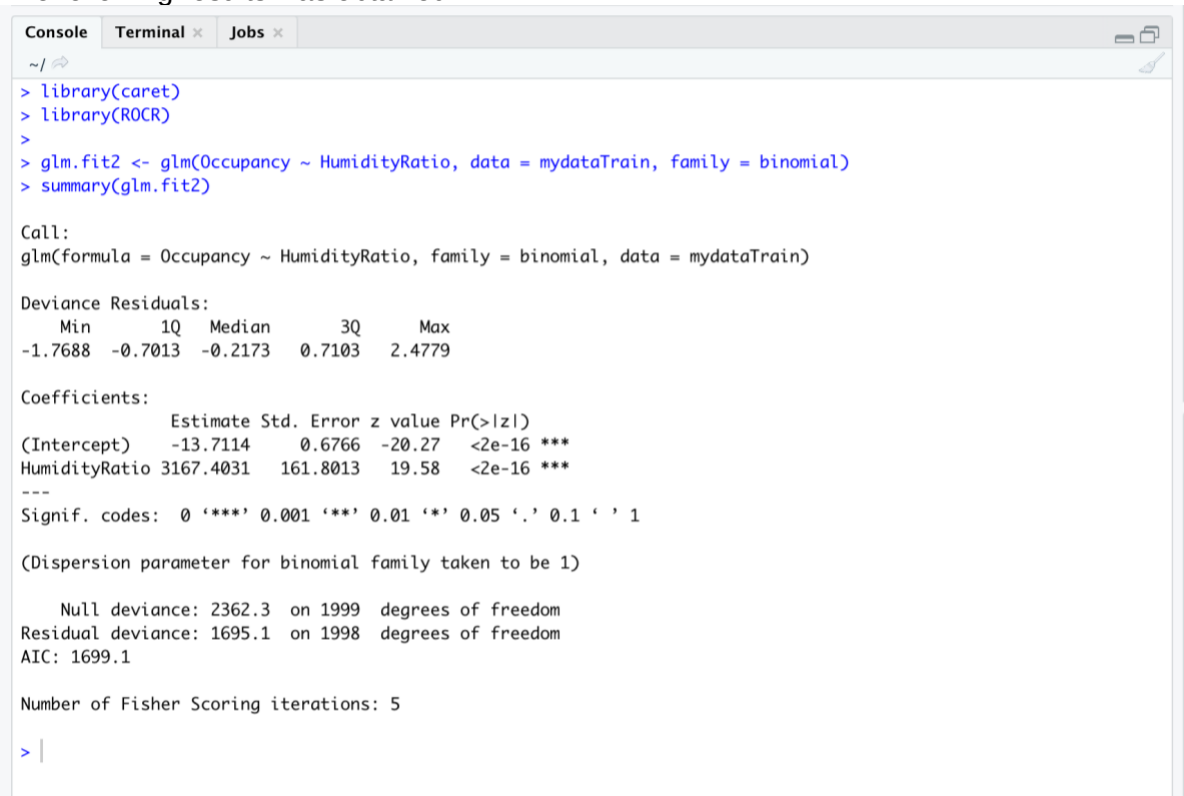
(c) Conduct a 10-fold cross validation to evaluate the predictive accuracy of a logistic regression model that uses HumidityRatio as the only predictor. Report the average accuracy and AUROC value obtained over the 10-fold cross validation. Set the value of random seed as "100" when generating fold indices. Consider the predictive label equals to 1, if the predictive probability is greater than 0.5.

(i) First we load the Library and then we estimate the logistic regression model using glm() function and then create the summary :

Code:

```
library(caret)
library(ROCR)

glm.fit2 <- glm(Occupancy ~ HumidityRatio, data = mydataTrain, family = binomial)
summary(glm.fit2)
```

The following results was obtained:



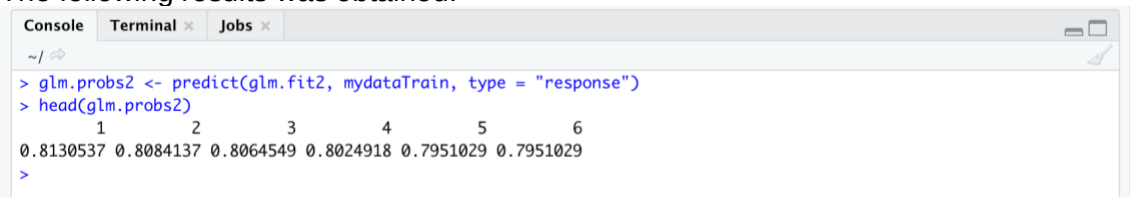
```
Console  Terminal  Jobs
~/
> library(caret)
> library(ROCR)
>
> glm.fit2 <- glm(Occupancy ~ HumidityRatio, data = mydataTrain, family = binomial)
> summary(glm.fit2)

Call:
glm(formula = Occupancy ~ HumidityRatio, family = binomial, data = mydataTrain)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.7688  -0.7013  -0.2173   0.7103   2.4779

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)   -13.7114     0.6766  -20.27   <2e-16 ***
HumidityRatio 3167.4031   161.8013   19.58   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2362.3  on 1999  degrees of freedom
Residual deviance: 1695.1  on 1998  degrees of freedom
AIC: 1699.1

Number of Fisher Scoring iterations: 5

>
```

(ii) Now we use model to predict the training dataset and store the results to a vector **:**

Code:

```
glm.probs2 <- predict(glm.fit2, mydataTrain, type = "response")
head(glm.probs2)
```

The following results was obtained:



```
Console  Terminal  Jobs
~/
> glm.probs2 <- predict(glm.fit2, mydataTrain, type = "response")
> head(glm.probs2)
        1         2         3         4         5         6
0.8130537 0.8084137 0.8064549 0.8024918 0.7951029 0.7951029
>
```

(iii) Now we create another vector to show predictive label equals to 1, if the predictive probability is greater than 0.5

Codes:

```
glm.pred2 <- rep(1,2000)
glm.pred2[glm.probs2<0.5] <- 0
head(glm.pred2)
table(glm.pred2,mydataTrain$Occupancy)
mean(glm.pred2 == mydataTrain$Occupancy)
```

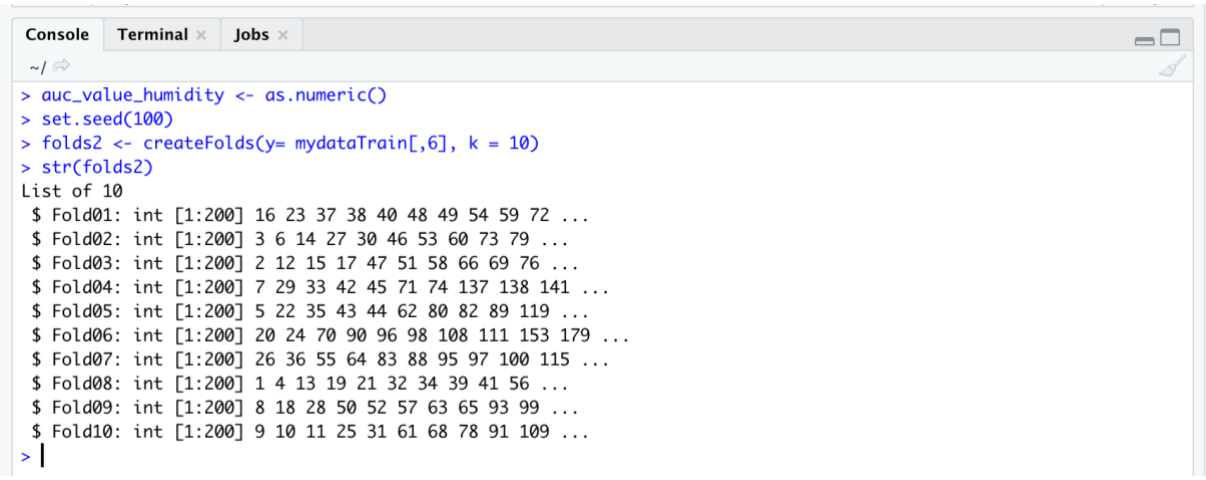
The following results was obtained:

(iv) Now we conduct a 10-fold cross validation to evaluate the predictive accuracy of a logistic regression model that uses HumidityRatio as the only predictor and set.seed(100):

Code:

```
auc_value_humidity <- as.numeric()
set.seed(100)
folds2 <- createFolds(y= mydataTrain[,6], k = 10)
str(folds2)
```

The following results was obtained:



(v) Now we conduct 10 fold cross validation find the average accuracy and AUROC value.

Code:

```
for (I in 1:10) {
  fold_cv_test2 <- mydataTrain[folds[[i]],]
  fold_cv_train2 <- mydataTrain[-folds[[i]],]
  trained_model_HumidityRatio <- glm(Occupancy ~ HumidityRatio, data = fold_cv_train2,
family = binomial)
  pred_prob_HumidityRatio <- predict(trained_model_HumidityRatio, fold_cv_test2, type =
“response”)
  pr_HumidityRatio <- prediction(pred_prob_HumidityRatio, fold_cv_test2$Occupancy)
  auroc_HumidityRatio <- performance(pr_HumidityRatio, measure = “auc”)
  auroc_HumidityRatio <- auroc_HumidityRatio@y.values[[1]]
  print(auroc_HumidityRatio)
  auc_value_humidity <- append(auc_value_humidity,auroc_HumidityRatio)
}
print(auc_value_humidity)
print(mean(auc_value_humidity))
```

The following results was obtained:

```
Console  Terminal  Jobs
~/
> for (i in 1:10) {
+   fold_cv_test2 <- mydataTrain[folds2[[i]],]
+   fold_cv_train2 <- mydataTrain[-folds2[[i]],]
+   trained_model_HumidityRatio <- glm(Occupancy ~ HumidityRatio, data = fold_cv_train2, family = binomial)
+   pred_prob_HumidityRatio <- predict(trained_model_HumidityRatio, fold_cv_test2, type = "response")
+   pr_HumidityRatio <- prediction(pred_prob_HumidityRatio, fold_cv_test2$Occupancy)
+   auroc_HumidityRatio <- performance(pr_HumidityRatio, measure = "auc")
+   auroc_HumidityRatio <- auroc_HumidityRatio@y.values[[1]]
+   print(auroc_HumidityRatio)
+   auc_value_humidity <- append(auc_value_humidity,auroc_HumidityRatio)
+ }
[1] 0.8185333
[1] 0.8711753
[1] 0.8289218
[1] 0.845206
[1] 0.840578
[1] 0.8843305
[1] 0.8283147
[1] 0.7820108
[1] 0.8255845
[1] 0.8594984
> print(auc_value_humidity)
 [1] 0.8185333 0.8711753 0.8289218 0.8452060 0.8405780 0.8843305 0.8283147 0.7820108 0.8255845 0.8594984
> print(mean(auc_value_humidity))
[1] 0.8384153
```

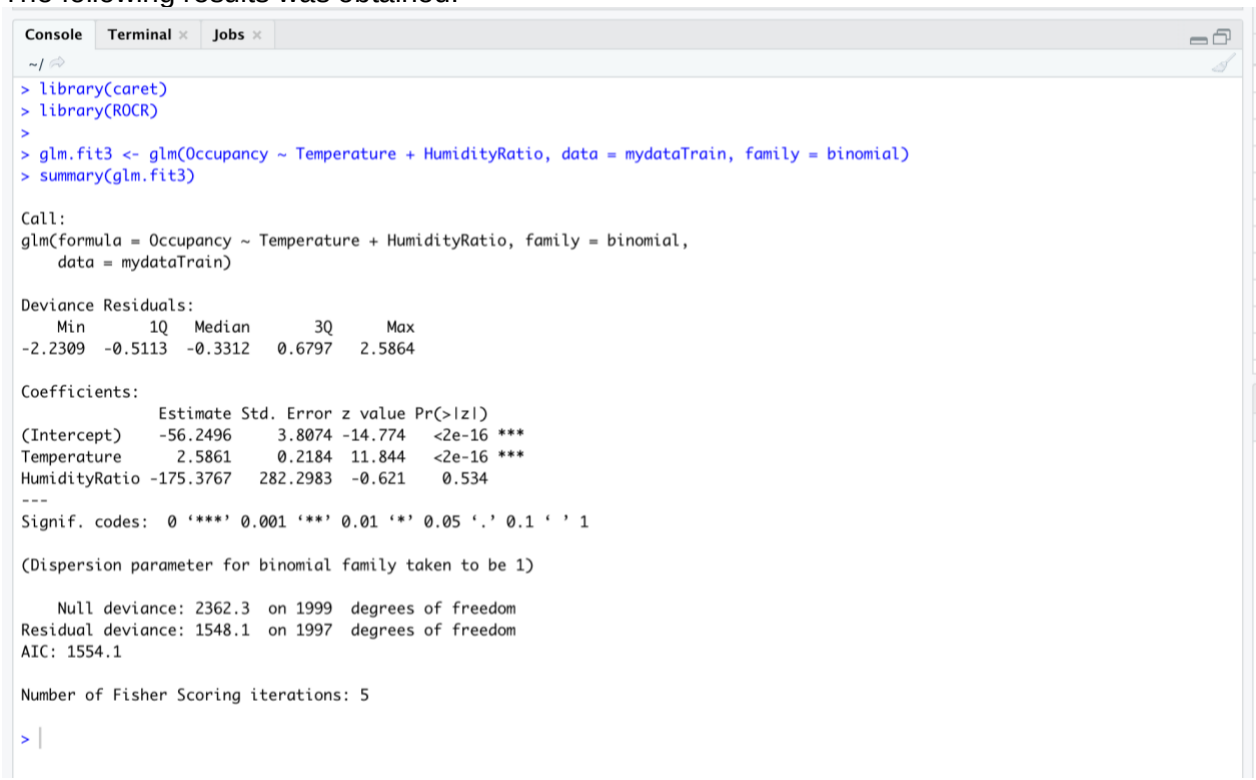
(d) Conduct a 10-fold cross validation to evaluate the predictive accuracy of a logistic regression model that uses Temperature and HumidityRatio in the training dataset. Report the average accuracy and AUROC value obtained over the 10-fold cross validation. Set the value of random seed as "100" when generating fold indices. Consider the predictive label equals to 1, if the predictive probability is greater than 0.5.

(i) First we load the Library and then we estimate the logistic regression model using glm() function and then create the summary :

Code:

```
library(caret)
library(ROCR)
glm.fit3 <- glm(Occupancy ~ Temperature + HumidityRatio, data = mydataTrain, family = binomial)
summary(glm.fit3)
```

The following results was obtained:



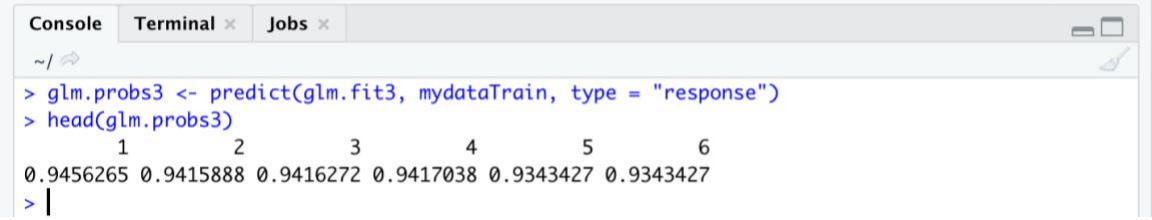
```
Console  Terminal  Jobs
~/
> library(caret)
> library(ROCR)
>
> glm.fit3 <- glm(Occupancy ~ Temperature + HumidityRatio, data = mydataTrain, family = binomial)
> summary(glm.fit3)

Call:
glm(formula = Occupancy ~ Temperature + HumidityRatio, family = binomial,
    data = mydataTrain)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.2309  -0.5113  -0.3312   0.6797   2.5864

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)    -56.2496     3.8074 -14.774   <2e-16 ***
Temperature      2.5861     0.2184  11.844   <2e-16 ***
HumidityRatio -175.3767   282.2983  -0.621    0.534
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2362.3  on 1999  degrees of freedom
Residual deviance: 1548.1  on 1997  degrees of freedom
AIC: 1554.1

Number of Fisher Scoring iterations: 5

>
```

(ii) Now we use model to predict the training dataset and store the results to a vector **:**

Code:

```
glm.probs3 <- predict(glm.fit3, mydataTrain, type = "response")
head(glm.probs3)
```

The following results was obtained:



(iii) Now we create another vector to show predictive label equals to 1, if the predictive probability is greater than 0.5

Codes:

```
glm.pred3 <- rep(1,2000)
glm.pred3 <- rep(1,2000)
glm.pred3[glm.probs3<0.5] <- 0
head(glm.pred3)
table(glm.pred3,mydataTrain$Occupancy)
mean(glm.pred3 == mydataTrain$Occupancy)
```

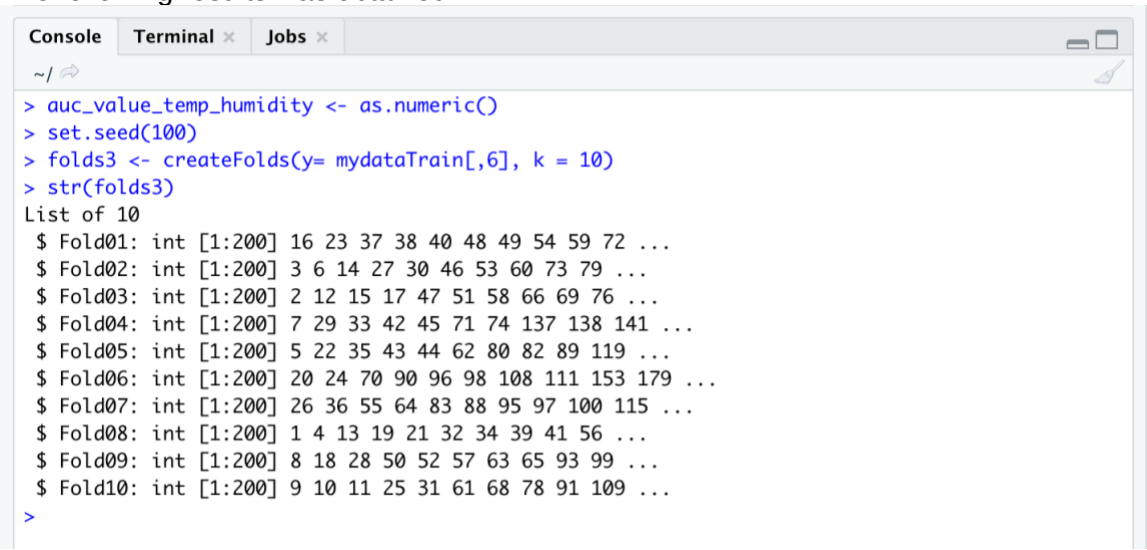
The following results was obtained:

iv) Now we conduct a 10-fold cross validation to evaluate the predictive accuracy of a logistic regression model that uses Temperature + HumidityRatio as the only predictor and set.seed(100):

Code:

```
auc_value_temp_humidity <- as.numeric()
set.seed(100)
folds3 <- createFolds(y= mydataTrain[,6], k = 10)
str(folds3)
```

The following results was obtained:



(v) Now we conduct 10 fold cross validation find the average accuracy and AUROC value.

Code:

```
for (i in 1:10) {
  fold_cv_test3 <- mydataTrain[folds3[[i]],]
  fold_cv_train3 <- mydataTrain[-folds3[[i]],]
  trained_model_Temp_Humidity <- glm(Occupancy ~ Temperature + HumidityRatio, data =
fold_cv_train3, family = binomial)
  pred_prob_Temp_Humidity <- predict(trained_model_Temp_Humidity, fold_cv_test3, type
= "response")
  pr_Temp_Humidity <- prediction(pred_prob_Temp_Humidity, fold_cv_test3$Occupancy)
  auroc_Temp_Humidity <- performance(pr_Temp_Humidity, measure = "auc")
  auroc_Temp_Humidity <- auroc_Temp_Humidity@y.values[[1]]
  print(auroc_Temp_Humidity)
  auc_value_temp_humidity <- append(auc_value_temp_humidity,auroc_Temp_Humidity)
}
print(auc_value_temp_humidity)
print(mean(auc_value_temp_humidity))
```

The following results was obtained:

```
Console  Terminal  Jobs
~/
> for (i in 1:10) {
+   fold_cv_test3 <- mydataTrain[folds3[[i]],]
+   fold_cv_train3 <- mydataTrain[-folds3[[i]],]
+   trained_model_Temp_Humidity <- glm(Occupancy ~ Temperature + HumidityRatio, data = fold_cv_
train3, family = binomial)
+   pred_prob_Temp_Humidity <- predict(trained_model_Temp_Humidity, fold_cv_test3, type = "resp
onse")
+   pr_Temp_Humidity <- prediction(pred_prob_Temp_Humidity, fold_cv_test3$Occupancy)
+   auroc_Temp_Humidity <- performance(pr_Temp_Humidity, measure = "auc")
+   auroc_Temp_Humidity <- auroc_Temp_Humidity@y.values[[1]]
+   print(auroc_Temp_Humidity)
+   auc_value_temp_humidity <- append(auc_value_temp_humidity,auroc_Temp_Humidity)
+ }
[1] 0.8465333
[1] 0.9158572
[1] 0.8617047
[1] 0.8825568
[1] 0.8642545
[1] 0.8954986
[1] 0.8706897
[1] 0.8312278
[1] 0.8570753
[1] 0.8932288
> print(auc_value_temp_humidity)
 [1] 0.8465333 0.9158572 0.8617047 0.8825568 0.8642545 0.8954986 0.8706897 0.8312278
 [9] 0.8570753 0.8932288
> print(mean(auc_value_temp_humidity))
[1] 0.8718627
>
```

(e) Compare the performance of those three different models on predicting the testing dataset. Draw ROC curves for all individual models and calculating the corresponding AUROC values. Discuss the comparison results obtained by the 10-fold cross validation and the hold-out testing.

(i) We will first rain different classifiers using the entire training dataset. So we will train Model_1 that uses Temperature as the only feature, then we will train Model_2 that uses HumidityRatio as the only feature and finally will train Model_3 that uses both Temperature and HumidityRatio as the features.

Code:

```
trained_model_1 <- glm(Occupancy ~ Temperature, data = mydataTrain, family = binomial)
trained_model_2 <- glm(Occupancy ~ HumidityRatio, data = mydataTrain, family = binomial)
trained_model_3 <- glm(Occupancy ~ Temperature + HumidityRatio, data = mydataTrain,
family = binomial)
```

(ii) We will now make predictions on testing instances by using different models.

Code:

```
prediction_trained_model_1 <- predict(trained_model_1, mydataTest, type='response')
prediction_trained_model_2 <- predict(trained_model_2, mydataTest, type='response')
prediction_trained_model_3 <- predict(trained_model_3, mydataTest, type='response')

pr_trained_model_1 <- prediction(prediction_trained_model_1, mydataTest$Occupancy)
pr_trained_model_2 <- prediction(prediction_trained_model_2, mydataTest$Occupancy)
pr_trained_model_3 <- prediction(prediction_trained_model_3, mydataTest$Occupancy)
```

(iii) Now we calculate the AUROC values obtained by different models.

Code:

```
auroc_trained_model_1 <- performance(pr_trained_model_1, measure = "auc")
auroc_trained_model_2 <- performance(pr_trained_model_2, measure = "auc")
auroc_trained_model_3 <- performance(pr_trained_model_3, measure = "auc")
auroc_trained_model_value_1 <- auroc_trained_model_1@y.values[[1]]
auroc_trained_model_value_2 <- auroc_trained_model_2@y.values[[1]]
auroc_trained_model_value_3 <- auroc_trained_model_3@y.values[[1]]
auroc_trained_model_value_1
auroc_trained_model_value_2
auroc_trained_model_value_3
```

(iv) We will calculate TPR (True Positive Rate) and FPR (False Positive Rate) values obtained by different models.

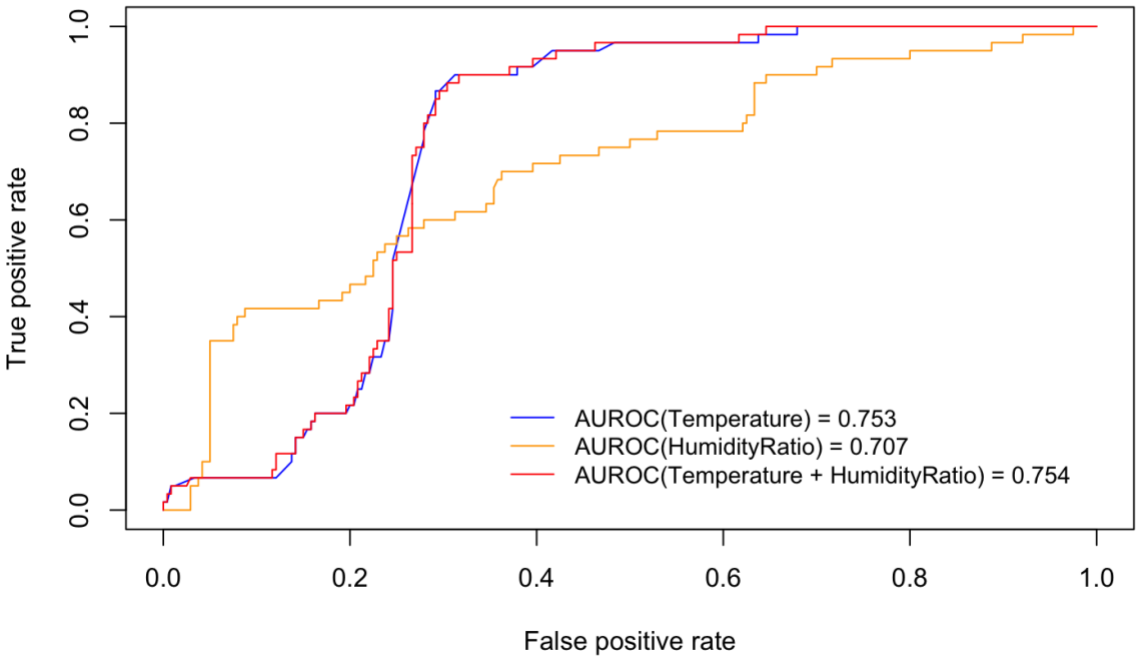
Code:

```
prf_trained_model_1 <- performance(pr_trained_model_1, measure = "tpr", x.measure =
"fpr")
prf_trained_model_2 <- performance(pr_trained_model_2, measure = "tpr", x.measure =
"fpr")
prf_trained_model_3 <- performance(pr_trained_model_3, measure = "tpr", x.measure =
"fpr")
```

(V) Now we will Draw ROC curves for different models.

```
plot(prf_trained_model_1, col="blue")
plot(prf_trained_model_2, col="orange", add=TRUE)
plot(prf_trained_model_3, col="red", add=TRUE)
legend(0.35,0.25, c(text=sprintf('AUROC(Temperature) =
%s',round(auroc_trained_model_value_1,digits=3)),
                    text=sprintf('AUROC(HumidityRatio) =
%s',round(auroc_trained_model_value_2, digits=3)),
              text=sprintf('AUROC(Temperature + HumidityRatio) =
             %s',round(auroc_trained_model_value_3, digits = 3))),
            lty=1, cex=0.9, col = c("blue", "orange", "red"), bty="n", y.intersp=1, inset=c(0.1,-0.15))
```

The following results was obtained:



The Temperature + HumidityRatio has better accuracy amongst the three models as it has larger area under ROC curve (AUROC). You can notice the deviation towards True positive. This means that there is a relation between the Temperature and Humidity of rooms promotes increase in occupancy. The 10-K fold method is better for testing data/ models like Temperature + HumidityRatio to get accurate results.
