

Recitation Exercises:**1.1 Chapter 3 Answers:****Exercise 1:**

	Coefficient	Std. error	t-statistic	p-value
Intercept	2.939	0.3119	9.42	<0.0001
TV	0.046	0.0014	32.81	<0.0001
Radio	0.189	0.0086	21.89	<0.0001
newspaper	-0.001	0.0059	-0.18	0.8599

Null Hypothesis refers to No relationship between X and Y. Mathematically it can be represented as:

$$H_0: \beta_1 = 0$$

Thus, from the above table,

- The null hypothesis for "TV": In the presence of radio ads and newspaper ads, TV ads have no effect on sales.
- The null hypothesis for "radio": In the presence of TV and newspaper ads, radio ads have no effect on sales.
- The null hypothesis for "newspaper": In the presence of TV and radio ads, newspaper ads have no effect on sales.
- However, because of the low p-values for TV and radio, the null hypotheses are rejected.
- On the other hand, the high p-value of newspaper suggests that the null hypothesis holds true for newspaper.

Exercise 3:

a. Given,

$X_1 = \text{GPA}$

$X_4 = \text{Interaction between GPA and IQ}$

$X_2 = \text{IQ}$

$X_5 = \text{Interaction bet}^n \text{ GPA \& Gender}$

$X_3 = \text{Gender}$

$$\widehat{\beta}_0 = 50, \widehat{\beta}_1 = 20, \widehat{\beta}_2 = 0.07, \widehat{\beta}_3 = 35, \widehat{\beta}_4 = 0.01, \widehat{\beta}_5 = -10$$

Therefore, we know: $\hat{Y} = \widehat{\beta}_0 + \widehat{\beta}_1 X_1 + \widehat{\beta}_2 X_2 + \widehat{\beta}_3 X_3 + \widehat{\beta}_4 X_4 + \widehat{\beta}_5 X_5$

$$Y = 50 + 20(\text{GPA}) + 0.07(\text{IQ}) + 35(\text{Gender}) + 0.01(\text{GPA} \times \text{IQ}) - 10(\text{GPA} \times \text{Gender})$$

Also, we know in Gender, Male = 0 and Female = 1,

$$\text{Males: } Y = 50 + 20(\text{GPA}) + 0.07(\text{IQ}) + 35(0) + 0.01(\text{GPA} \times \text{IQ}) - 10(\text{GPA} \times 0)$$

$$Y = 50 + 20(\text{GPA}) + 0.07(\text{IQ}) + 0.01(\text{GPA} \times \text{IQ})$$

$$\text{Females: } Y = 50 + 20(\text{GPA}) + 0.07(\text{IQ}) + 35(1) + 0.01(\text{GPA} \times \text{IQ}) - 10(\text{GPA} \times 1)$$

$$Y = 85 + 10(\text{GPA}) + 0.07(\text{IQ}) + 0.01(\text{GPA} \times \text{IQ})$$

On solving above two equations we $\text{GPA} = 3.5$,

Therefore, Males earn more than Females if and only if GPA is greater than 3.5 (High GPA)

Therefore, **iii** is correct.

- b. Given IQ = 110, GPA = 4.0, To find Salary of female =?

From the equation of females derived from the above question we have,

$$\text{Females: } Y = 85 + 10(\text{GPA}) + 0.07(\text{IQ}) + 0.01(\text{GPA} \times \text{IQ})$$

$$\text{Therefore, } Y = 85 + 10(4.0) + 0.07(110) + 0.01(4.0 \times 110)$$

$$\text{Salary of female} = 137.1 = \$137100/-$$

- c. **False.** To examine if the GPA/IQ has an impact on the quality of the model we need to test the hypothesis $H_0: \widehat{\beta}_4 = 0$ and look at the p-value associated with the t-statistic to draw a conclusion.

Exercise 4:

- Without knowing more details about the training data, it is difficult to know which training RSS is lower between linear or cubic. However, as the true relationship between X and Y is linear, it can be assumed that the least squares line to be close to the true regression line, and consequently the RSS for the linear regression may be lower than for the cubic regression.
- The test RSS depends on test data, thus to draw any conclusion, enough information isn't available. However, it can be assumed that polynomial regression will have a higher test RSS as the overfit from training would have more error than the linear regression.
- Due to the high flexibility, Polynomial regression has lower train RSS as compared to the linear fit. So, the more flexible model will closer follow points and reduce train RSS.
- The information available is not sufficient enough to tell which test RSS would be lower for either regression as it is not clear that what level of flexibility will fit data better. If it is closer to linear, the linear regression test RSS would be lower and if it closer to cubic then the cubic regression test RSS could be lower.

1.2 Chapter 4 Answers:

Exercise 4:

- a. Given,

If $x \in [0.05, 0.95]$ then the observations are in the interval $[x-0.05, x+0.05]$.

Length = 0.1 (a fraction of 10%).

If $x < 0.05$, then the observations in the interval $[0, x+0.05]$ are used, which represents a fraction of $(100x+5)$ %; by a similar argument it is concluded that if $x > 0.95$, then the fraction of observations used is $(105-100x)$ %. To calculate the average fraction, we will use the following expression:

$$\int_{0.05}^{0.95} 10dx + \int_{0.00}^{0.05} (100x + 5)dx + \int_{0.95}^1 (105 - 100x)dx = 9 + 0.375 + 0.375 = 9.75$$

The fraction of available observations used to make the prediction is **9.75%**.

- b. If it is assumed that X_1 and X_2 to be independent, the fraction of available observations we will use to make the prediction is $9.75\% \times 9.75\% = \mathbf{0.950625\%}$.

c. Like the explanation in (a) and (b), it can be concluded that the fraction of available observations used to make the prediction is $9.75\%^{100} \simeq 0\%$

d. Like the explanation in (a) to (c), the fraction of available observations we will use to make the prediction is $(9.75\%)^p$ with p the number of features, hence $p \rightarrow \infty, \lim_{p \rightarrow \infty} (9.75\%)^p = 0$

e.

For $p=1$, we have $l=0.1$,

For $p=2$, we have $l=0.11/2$

For $p=100$, we have $l=0.11/100$.

Exercise 6:

We know for Logistic Regression with multiple variables,

$$p(X) = \frac{e^{\beta_0 + X_1\beta_1 + X_2\beta_2}}{1 + e^{\beta_0 + X_1\beta_1 + X_2\beta_2}}$$

Given: $\beta_0 = -6$, $\beta_1 = 0.05$, $\beta_2 = 1$, $e = 2.71828$

a. $X_1 = 40\text{hrs}$, $X_2 = 3.5\text{ GPA}$

$$p(X) = \frac{2.72^{-6+(40 \times 0.05)+(3.5 \times 1)}}{1 + 2.72^{-6+(40 \times 0.05)+(3.5 \times 1)}} = \mathbf{0.3774 = 33.74\%}$$

b. $p(X) = 0.5 = 50\%$, $X_2 = 3.5\text{ GPA}$, $X_1 = ?$

$$0.5 = \frac{2.72^{-6+(X_1 \times 0.05)+(3.5 \times 1)}}{1 + 2.72^{-6+(X_1 \times 0.05)+(3.5 \times 1)}} = \mathbf{50\text{ hrs}}$$

Exercise 7:

Given:

$\Pi_{\text{yes}} = 80\% = 0.8$, $\Pi_{\text{no}} = 20\% = 0.2$, $\mu_{\text{yes}} = 10$, $\mu_{\text{no}} = 0$, $\sigma^2 = 36$

To Find:

$p_{\text{yes}}(4) = \Pr(Y=\text{yes} \mid X = 4) = ?$

Solution:

We know from Bayes formula,

$$p_k(x) = \Pr(Y = k \mid X = x) = \frac{\pi_k \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2}\left(\frac{x-\mu_k}{\sigma}\right)^2}}{\sum_{l=1}^K \pi_l \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2}\left(\frac{x-\mu_l}{\sigma}\right)^2}}$$

$$p_{\text{yes}}(4) = \Pr(Y = \text{yes} \mid X = 4) = \frac{\pi_{\text{yes}} \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2}\left(\frac{4-\mu_{\text{yes}}}{\sigma}\right)^2}}{\pi_{\text{yes}} \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2}\left(\frac{4-\mu_{\text{yes}}}{\sigma}\right)^2} + \pi_{\text{no}} \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2}\left(\frac{4-\mu_{\text{no}}}{\sigma}\right)^2}}$$

$$= \frac{0.8 \times \frac{1}{\sqrt{2\pi} \times 6} e^{-\frac{1}{2}\left(\frac{4-10}{6}\right)^2}}{\left(0.8 \times \frac{1}{\sqrt{2\pi} \times 6} e^{-\frac{1}{2}\left(\frac{4-10}{6}\right)^2}\right) + \left(0.2 \times \frac{1}{\sqrt{2\pi} \times 6} e^{-\frac{1}{2}\left(\frac{4-0}{6}\right)^2}\right)} = 0.752 = 75.2\%$$

$$\mathbf{p_{yes} (4) = 75.2\%}$$

Exercise 9:

a. Given: $\frac{p(x)}{1-p(x)} = 0.37$ To find: $p(x) = ?$

$$p(x) = 0.37[1 - p(x)], \text{ thus } \mathbf{p(x) = 0.27 = 27\%}$$

b. Given: $p(x) = 16\% = 0.16$, To find: $\frac{p(x)}{1-p(x)} = ?$

$$\frac{0.16}{1-0.16} = \frac{0.16}{0.84} = \frac{4}{21} \text{ (4 people who defaulted vs 21 people who did not)}$$

Practicum Problems:

Problem 1:

#Problem 1

```
library(MASS)
library(ggplot2)
bostonData = data.frame(Boston)
attach(Boston)

linearModel1 = lm(medv~lstat, data = bostonData)
summary(linearModel1)

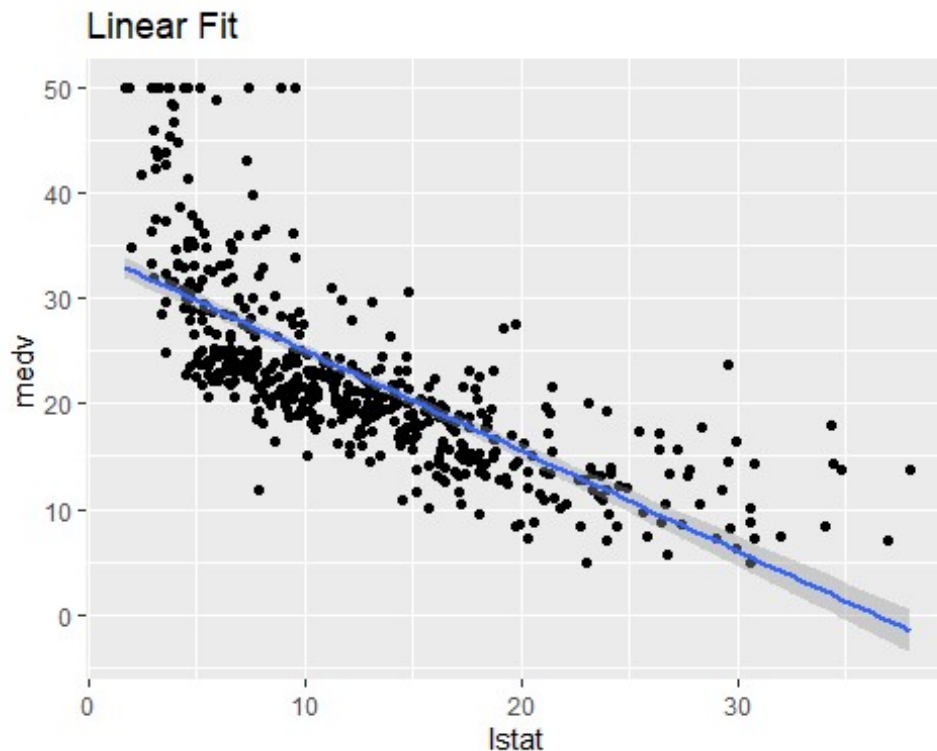
##
## Call:
## lm(formula = medv ~ lstat, data = bostonData)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.168  -3.990  -1.318   2.034  24.500
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  34.55384    0.56263   61.41  <2e-16 ***
## lstat       -0.95005    0.03873  -24.53  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.216 on 504 degrees of freedom
## Multiple R-squared:  0.5441, Adjusted R-squared:  0.5432
## F-statistic: 601.6 on 1 and 504 DF, p-value: < 2.2e-16

coef(linearModel1)

## (Intercept)      lstat
##  34.5538409  -0.9500494
```

For Linear Model we have $R^2 = 0.5441$

```
ggplot(bostonData, aes(lstat, medv)) + geom_point() + stat_smooth(method = lm, se =
TRUE) + ggtitle("Linear Fit")
```



There is some evidence for non-linearity in the relationship between **lstat** and **medv**. This will be discussed ahead.

```
confint(linearModel1)

##              2.5 %      97.5 %
## (Intercept) 33.448457 35.6592247
## lstat      -1.026148 -0.8739505

predict(linearModel1, data.frame(lstat=c(5,10,15))), interval="confidence")

##      fit      lwr      upr
## 1 29.80359 29.00741 30.59978
## 2 25.05335 24.47413 25.63256
## 3 20.30310 19.73159 20.87461

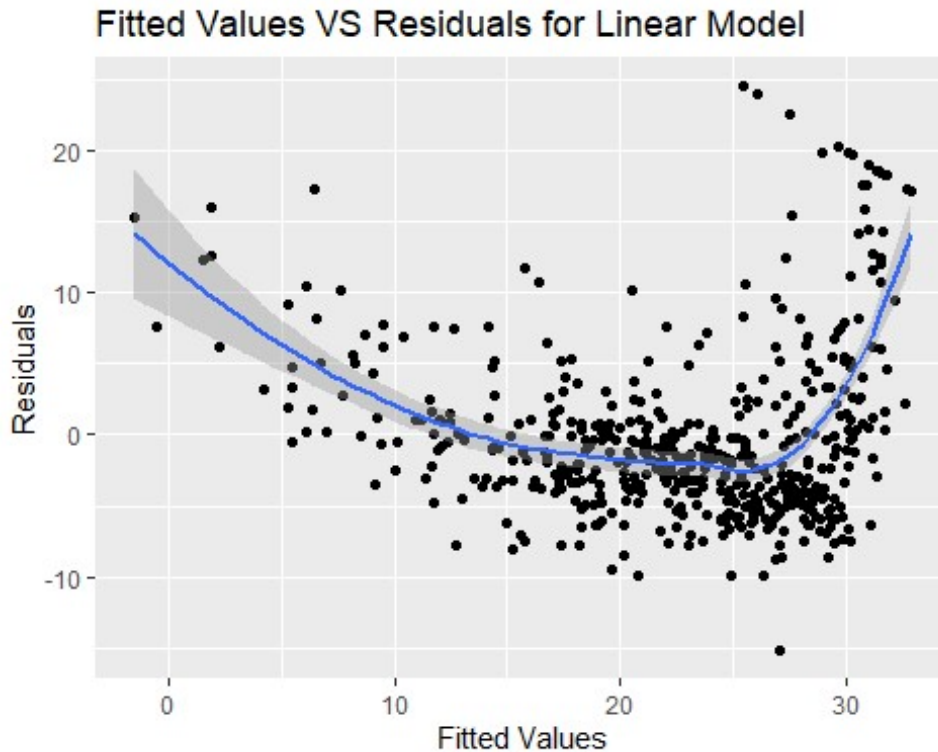
predict(linearModel1, data.frame(lstat=c(5,10,15))), interval="prediction")

##      fit      lwr      upr
## 1 29.80359 17.565675 42.04151
## 2 25.05335 12.827626 37.27907
## 3 20.30310  8.077742 32.52846
```

The Confidence Level and Prediction Level of the results are not same, for example, the 95% confidence interval associated with a **lstat** value of 10 is (24.47, 25.63), and the 95% prediction interval is (12.828, 37.28). Thus, the confidence and prediction intervals are centered around the same point (a predicted value of 25.05 for **medv** when **lstat** equals 10), but the latter are substantially wider.

```
ggplot(linearModel1, aes(x = linearModel1$fitted.values, y = linearModel1$residuals)) +
geom_point() + stat_smooth(se = TRUE) + labs(x = "Fitted Values", y = "Residuals") +
ggtitle("Fitted Values VS Residuals for Linear Model")

## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



```
cat("Modifying the model to include lstat^2")

## Modifying the model to include lstat^2

linearModel2=lm(medv~lstat + I(lstat^2))
summary(linearModel2)

##
## Call:
## lm(formula = medv ~ lstat + I(lstat^2))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.2834  -3.8313  -0.5295   2.3095  25.4148
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  42.862007   0.872084   49.15  <2e-16 ***
## lstat        -2.332821   0.123803  -18.84  <2e-16 ***
## I(lstat^2)    0.043547   0.003745   11.63  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 5.524 on 503 degrees of freedom
## Multiple R-squared: 0.6407, Adjusted R-squared: 0.6393
## F-statistic: 448.5 on 2 and 503 DF, p-value: < 2.2e-16

coef(linearModel2)

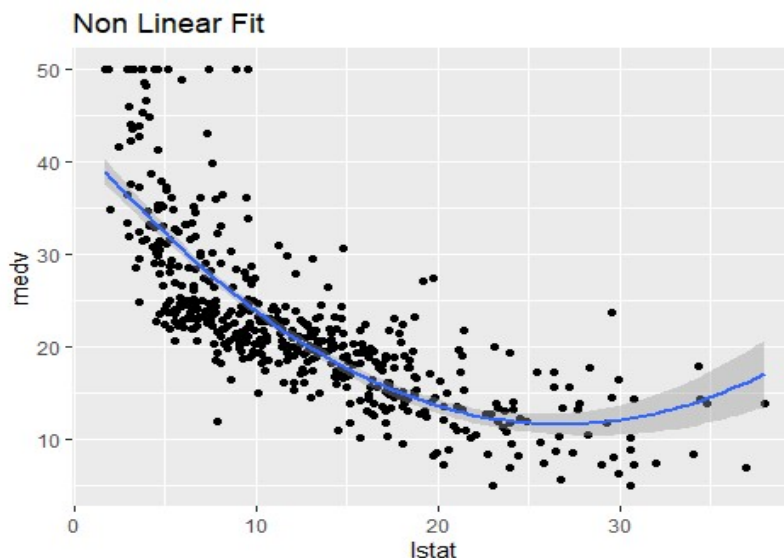
## (Intercept)      lstat      I(lstat^2)
## 42.86200733 -2.33282110  0.04354689

anova(linearModel1,linearModel2)

## Analysis of Variance Table
##
## Model 1: medv ~ lstat
## Model 2: medv ~ lstat + I(lstat^2)
##   Res.Df    RSS Df Sum of Sq    F    Pr(>F)
## 1      504 19472
## 2      503 15347   1    4125.1 135.2 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

For Non-Linear Model we have $R^2 = 0.6407$. The near-zero p-value associated with the quadratic term suggests that it leads to an improved model. We use the `anova()` function to further quantify the extent to which the quadratic fit is superior to the linear fit. The `anova()` function performs a hypothesis test comparing the two models. The null hypothesis is that the two models fit the data equally well, and the alternative hypothesis is that the full model is superior. Here the F-statistic is 135 and the associated p-value is virtually zero. This provides very clear evidence that the model containing the predictors `lstat` and `lstat2` is far superior to the model that only contains the predictor `lstat`. This is not surprising, since earlier we saw evidence for non-linearity in the relationship between `medv` and `lstat`.¹

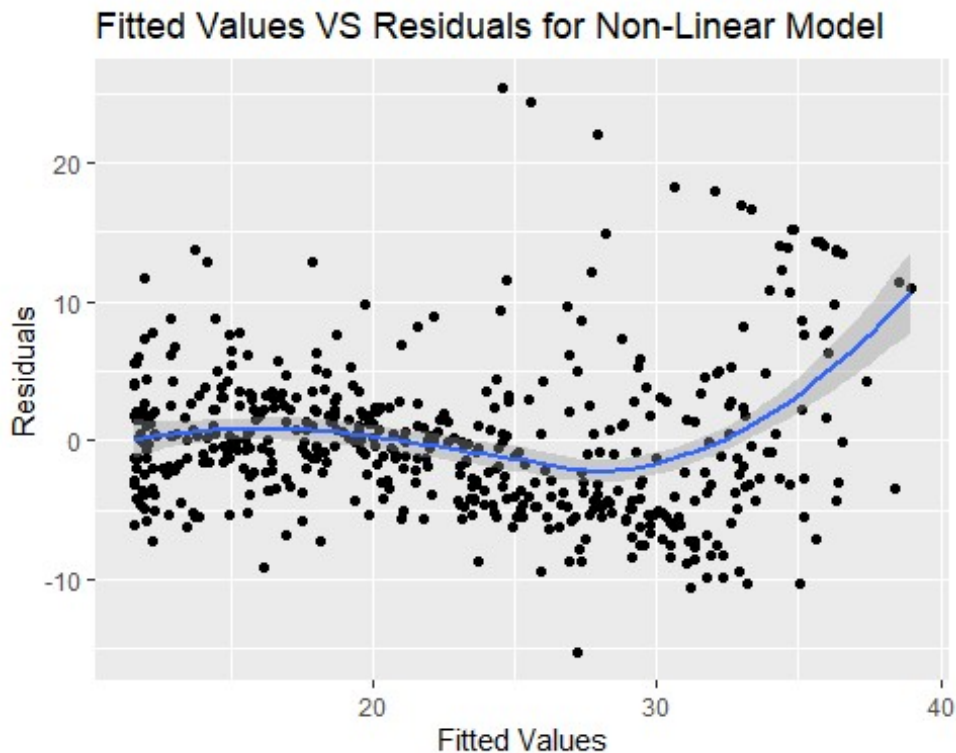
```
ggplot(bostonData,aes(x= lstat,y = medv)) + geom_point() + stat_smooth(method = "lm",
formula = y ~ x + I(x^2), se = TRUE) + ggtitle("Non Linear Fit")
```



¹ Referred from Introduction to Statistical Learning, Chapter 3: Linear Regression
Chirag Khandhar

```
ggplot(linearModel2, aes(x = linearModel2$fitted.values, y = linearModel2$residuals)) +
  geom_point() + stat_smooth(se = TRUE) + labs(x = "Fitted Values", y = "Residuals") +
  ggtitle("Fitted Values VS Residuals for Non-Linear Model")

## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



Problem 2:

#Problem 2

```
library(readr)
library(data.table)
library(corrplot)

## corrplot 0.84 loaded

library(caret)

## Loading required package: lattice

library(pROC)

## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##   cov, smooth, var

abaloneURL = "https://archive.ics.uci.edu/ml/machine-learning-
databases/abalone/abalone.data"
```

```

abaloneData = fread(abaloneURL, header = FALSE)
abaloneHeader = c("Sex", "Length", "Diameter", "Height", "Whole weight", "Shucked
weight", "Viscera weight", "Shell weight", "Rings")
colnames(abaloneData) = abaloneHeader

#Direct use of attribute names in code
attach(abaloneData)

#Returns the indices when the following condition is true in which()
condition = which(abaloneData$Sex!="I")

#All the Rows with Sex = I are removed
abaloneData2 = abaloneData[condition]

#As the response variable need to be numeric, converting categorical Sex to a factor
abaloneData2$Sex = factor(abaloneData2$Sex)

#Creating 80-20 Training Testing Split, createDataPartition() returns the indices
trainIndex = createDataPartition(y = abaloneData2$Sex, p = 0.8, list = FALSE)

#Training data
trainData = abaloneData2[trainIndex,]

#Testing data (note the minus sign)
testData = abaloneData2[-trainIndex,]

#Predicting Sex using glm
model <- glm(Sex~.,family=binomial,data=trainData)

#Summary of our model
summary(model)

##
## Call:
## glm(formula = Sex ~ ., family = binomial, data = trainData)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -1.8384  -1.2046   0.8933   1.1180   1.5171
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    2.658156    0.520680   5.105 3.31e-07 ***
## Length        -1.848333    2.281360  -0.810  0.41783
## Diameter      -3.853427    2.721468  -1.416  0.15679
## Height        -4.420852    2.734666  -1.617  0.10597
## `Whole weight` -0.215849    0.811310  -0.266  0.79020
## `Shucked weight` 3.236048    0.987393   3.277  0.00105 **
## `Viscera weight` -1.817911    1.390371  -1.308  0.19104
## `Shell weight`  0.510163    1.256973   0.406  0.68484
## Rings          0.001187    0.018163   0.065  0.94788
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

```
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 3131.7 on 2268 degrees of freedom
## Residual deviance: 3076.9 on 2260 degrees of freedom
## AIC: 3094.9
##
## Number of Fisher Scoring iterations: 4

#Coefficients of our model
coef(model)

## (Intercept) Length Diameter Height
## 2.658155529 -1.848332717 -3.853426988 -4.420852357
## `Whole weight` `Shucked weight` `Viscera weight` `Shell weight`
## -0.215848702 3.236048107 -1.817911418 0.510162507
## Rings
## 0.001187393

#Confidence Interval
confint(model)

## Waiting for profiling to be done...

## 2.5 % 97.5 %
## (Intercept) 1.65049888 3.69293678
## Length -6.32406005 2.62518325
## Diameter -9.20237103 1.47432874
## Height -9.80206425 0.92880324
## `Whole weight` -1.81610782 1.37641368
## `Shucked weight` 1.30885174 5.18755135
## `Viscera weight` -4.55065032 0.90672173
## `Shell weight` -1.95863205 2.98052017
## Rings -0.03443305 0.03681874
```

From the above observations for confidence interval, we can see that, all the predictors, except the “Shucked Weight” contain 0 within their confidence interval range which is in line with the basic assumption of Null Hypothesis which says, No relationship between X and Y. Mathematically: $H_0: \beta_1 = 0$. This can further be supported with their respective p-values observed above. The only attribute with a substantially low p-value is “Shucked Weight” with a p-value of **0.00105** and all the other attributes have a high p-value. Thus, we can conclude that, there is no relationship between the predictors and the response variable (except the “Shucked Weight”) and Null Hypothesis holds true for all the predictors except the “Shucked Weight”. Hence Shucked Weight is the only predictor which is relevant.

#Predict [By setting the parameter type='response', R will output probabilities in the form of $P(y=1|X)$]

```
probs = predict(model, testData, type = "response")
```

#Using a 50% cut-off factor i.e probabilities > 0.5 are Males and rest are Females

```
resultSet = ifelse(probs > 0.5, "M", "F")
```

```
resultSet2 = factor(resultSet)
```

#Creating a confusion matrix

```
confusionMatrix(resultSet2, testData$Sex)
```

```
## Confusion Matrix and Statistics
```

```
##
```

```
##           Reference
```

```
## Prediction   F     M
```

```
##           F  96   86
```

```
##           M 165  219
```

```
##
```

```
##           Accuracy : 0.5565
```

```
##           95% CI : (0.5145, 0.598)
```

```
##           No Information Rate : 0.5389
```

```
##           P-Value [Acc > NIR] : 0.2117
```

```
##
```

```
##           Kappa : 0.0878
```

```
##
```

```
##           McNemar's Test P-Value : 8.509e-07
```

```
##
```

```
##           Sensitivity : 0.3678
```

```
##           Specificity : 0.7180
```

```
##           Pos Pred Value : 0.5275
```

```
##           Neg Pred Value : 0.5703
```

```
##           Prevalence : 0.4611
```

```
##           Detection Rate : 0.1696
```

```
##           Detection Prevalence : 0.3216
```

```
##           Balanced Accuracy : 0.5429
```

```
##
```

```
##           'Positive' Class : F
```

```
##
```

#Plotting the ROC Curve

```
library(ROCR)
```

```
## Loading required package: gplots
```

```
##
```

```
## Attaching package: 'gplots'
```

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

```
##
```

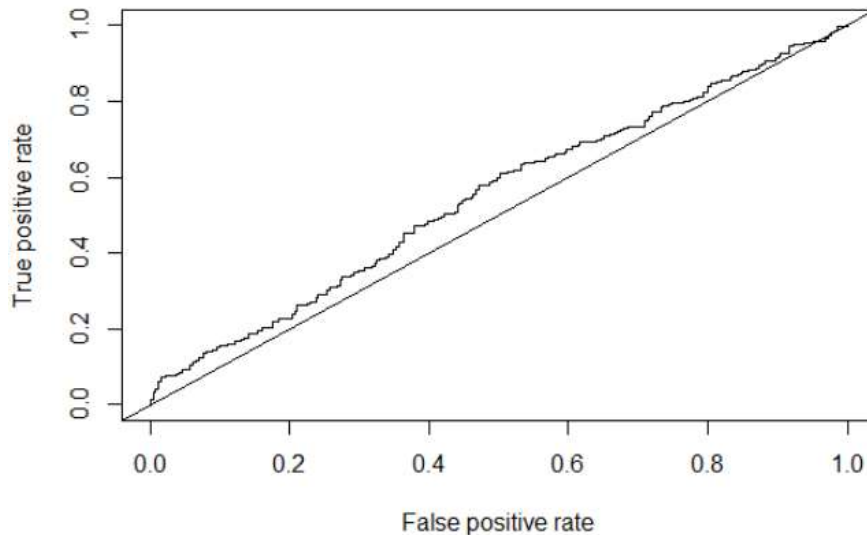
```
##           lowess
```

```
roc.pred = prediction(probs, testData$Sex)
```

```
roc.perf = performance(roc.pred, measure = "tpr", x.measure = "fpr")
```

```
plot(roc.perf)
```

```
abline(0,1)
```



```
auc.perf = performance(roc.pred, measure = "auc")
cat("Area Under the Curve: ")
```

```
## Area Under the Curve:
```

```
auc.perf@y.values
```

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

```
## [1] 0.5777714
```

```
#Plotting the correlations between the predictors
```

```
cm = cor(abaloneData2[, -1])
```

```
corrplot(cm, method = "number")
```



From the above plot, we can clearly see that, for all the predictors there exists a positive linear relationship. More precisely, only the **Rings** predictor has a weak uphill (positive) relationship while the rest of the others have a strong uphill (positive) relationship.

Problem 3:

```
library(data.table) #Data Import
library(e1071)      #Naive Bayes

#Setting up the URL for data import
mushroomURL = "https://archive.ics.uci.edu/ml/machine-learning-
databases/mushroom/agaricus-lepiota.data"
mushroomData = fread(mushroomURL,header=FALSE)

#Adding the headers
mushroomHeader = c("Class","cap-shape","cap-surface","cap-color","bruises","odor","gill-
attachment","gill-spacing","gill-size","gill-color","stalk-shape","stalk-root","stalk-
surface-above-ring","stalk-surface-below-ring","stalk-color-above-ring","stalk-color-
below-ring","veil-type","veil-color","ring-number","ring-type","spore-print-
color","population","habitat")
colnames(mushroomData) = mushroomHeader

#Class Distribution
table(mushroomData$Class)

##
##      e      p
## 4208 3916

#Converting Class Attribute to a factor
mushroomData$Class = factor(mushroomData$Class)

#Dimensions
dim(mushroomData)

## [1] 8124   23

#Structure
str(mushroomData)

## Classes 'data.table' and 'data.frame':  8124 obs. of  23 variables:
## $ Class          : Factor w/ 2 levels "e","p": 2 1 1 2 1 1 1 1 2 1 ...
## $ cap-shape      : chr  "x" "x" "b" "x" ...
## $ cap-surface    : chr  "s" "s" "s" "y" ...
## $ cap-color      : chr  "n" "y" "w" "w" ...
## $ bruises        : chr  "t" "t" "t" "t" ...
## $ odor           : chr  "p" "a" "l" "p" ...
## $ gill-attachment : chr  "f" "f" "f" "f" ...
## $ gill-spacing    : chr  "c" "c" "c" "c" ...
## $ gill-size       : chr  "n" "b" "b" "n" ...
## $ gill-color      : chr  "k" "k" "n" "n" ...
## $ stalk-shape     : chr  "e" "e" "e" "e" ...
## $ stalk-root      : chr  "e" "c" "c" "e" ...
## $ stalk-surface-above-ring: chr  "s" "s" "s" "s" ...
## $ stalk-surface-below-ring: chr  "s" "s" "s" "s" ...
## $ stalk-color-above-ring : chr  "w" "w" "w" "w" ...
## $ stalk-color-below-ring : chr  "w" "w" "w" "w" ...
## $ veil-type       : chr  "p" "p" "p" "p" ...
```

```
## $ veil-color      : chr  "w" "w" "w" "w" ...
## $ ring-number     : chr  "o" "o" "o" "o" ...
## $ ring-type       : chr  "p" "p" "p" "p" ...
## $ spore-print-color : chr  "k" "n" "n" "k" ...
## $ population      : chr  "s" "n" "n" "s" ...
## $ habitat         : chr  "u" "g" "m" "u" ...
## - attr(*, ".internal.selfref")=<externalptr>
```

#Finding the number of missing values

```
cat("Number of missing values = ",sum(mushroomData=="?"))
```

```
## Number of missing values = 2480
```

As we have enough observations after excluding the ones with missing value, there is no harm even if we remove the observations having missing values.

#New dataset with removed missing values

```
mushroomData2 = mushroomData[mushroomData$`stalk-root`!="?"]
```

#Creating a split

```
trainSize = floor(0.80*nrow(mushroomData2))
trainIndex = sample(nrow(mushroomData2), size = trainSize)
trainData = mushroomData2[trainIndex,]
testData = mushroomData2[-trainIndex,]
```

#Naive Bayes

```
model = naiveBayes(trainData[, -1], trainData$Class)
```

#Prediction on Testing Data

```
testPred = predict(model, testData[, -1])
```

#Prediction on Training Data

```
trainPred = predict(model, trainData[, -1])
```

#Accuracy of Testing Model

```
cat("Accuracy of Testing Model: ", mean(testPred == testData$Class)*100, "%")
```

```
## Accuracy of Testing Model: 95.30558 %
```

#Accuracy of Training Model

```
cat("Accuracy of Training Model: ", mean(trainPred == trainData$Class)*100, "%")
```

```
## Accuracy of Training Model: 95.76966 %
```

#Confusion Matrix

```
table(testPred, testData$Class)
```

```
##
## testPred  e  p
##          e 689 50
##          p  3 387
```

From the above the confusion matrix we have the following:

TP = 689, FP = 50, FN = 3 and TN = 387