Assignment 1

Kunal Sharma

Importing Important Packages

```
library(ISLR)
library(dplyr)
library(ggplot2)
```

Loading the Carseats dataset with only 3 variables. Filtering the data into two subsets based on the shelve location(GOOD or BAD).

```
SafeBabies <- Carseats %>% select("Sales", "Price", "ShelveLoc")
Good_shevles <- filter(SafeBabies, ShelveLoc == "Good")
Bad_shevles <- filter(SafeBabies, ShelveLoc == "Bad")
```

#Building a Linear Regression model to predict the sales of the carseat for both good as well as bad shelve location individually.

```
#Linear Model for GOOD Shelve location
Lm_Good <- lm(Sales ~ Price, data = Good_shevles)
summary(Lm_Good)

##
## Call:
## lm(formula = Sales ~ Price, data = Good_shevles)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -3.721 -1.351 -0.098  1.483  4.353
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 17.968864   0.988008  18.187  < 2e-16 ***
## Price       -0.065785   0.008199  -8.023 5.85e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.888 on 83 degrees of freedom
## Multiple R-squared:  0.4368, Adjusted R-squared:   0.43
## F-statistic: 64.37 on 1 and 83 DF,  p-value: 5.848e-12

#Linear Model for BAD Shelve location
Lm_Bad <- lm(Sales ~ Price, data = Bad_shevles)
summary(Lm_Bad)

##
## Call:
```

```
## lm(formula = Sales ~ Price, data = Bad_shevles)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.4622 -1.0617 -0.2014  1.2050  4.6412
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 11.832984   0.990317  11.949  < 2e-16 ***
## Price       -0.055220   0.008486  -6.507  3.7e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.967 on 94 degrees of freedom
## Multiple R-squared:  0.3105, Adjusted R-squared:  0.3032
## F-statistic: 42.34 on 1 and 94 DF,  p-value: 3.702e-09
```

#Equation: Total Profit = Sales * (Selling Price - Production Cost)

Futher simplication of equation we got the below formula,

Optimal Selling Price = Predicted Sales based on Production Cost / (2 * Estimated Price Coefficient)

Assuming the Production Cost of a CarSeat is $55.0, below is the predicted optimal Selling Price of the Carseat.

optimal cost for shelve location Good

```
Productioncost<-55
paste("The optimal price for a good shelf position", ((-
Lm_Good$coefficients[[2]] *Productioncost) + (Lm_Good$coefficients[[1]]))/(-2
* Lm_Good$coefficients[[2]]))

## [1] "The optimal price for a good shelf position 164.07312564386"
```

optimal cost for shelve location bad

```
paste("The optimal price for a bad shelf position", ((-
Lm_Bad$coefficients[[2]] *Productioncost) + (Lm_Bad$coefficients[[1]]))/(-2 *
Lm_Bad$coefficients[[2]]))

## [1] "The optimal price for a bad shelf position 134.643464696399"
```

Note: The negative sign is inserted in the denominator to nullify the negation, as the sign of the LM shows Price's negative correlation with the intercept (Sales).

Here the variation in Production Cost from $40 to $85 the Selling Price also varies as below.

```
Good_Optimal_price_Range <- (predict(Lm_Good, data.frame(Price = c(40:85))))
/ (-2*Lm_Good$coefficients[2])
Bad_Optimal_Price_Range <- (predict(Lm_Bad, data.frame(Price = c(40:85)))) /
(-2*Lm_Bad$coefficients[2])
```

Selling Price for Good and Bad Shelve Locations over Production Costs of $40-$85

```
Price_Range<- cbind.data.frame(Production_Cost = c(40:85), Selling_Price_Good
= Good_Optimal_price_Range, Selling_Price_Bad = Bad_Optimal_Price_Range)
Price_Range
```

```
##    Production_Cost Selling_Price_Good Selling_Price_Bad
## 1               40          116.57313          87.14346
## 2               41          116.07313          86.64346
## 3               42          115.57313          86.14346
## 4               43          115.07313          85.64346
## 5               44          114.57313          85.14346
## 6               45          114.07313          84.64346
## 7               46          113.57313          84.14346
## 8               47          113.07313          83.64346
## 9               48          112.57313          83.14346
## 10              49          112.07313          82.64346
## 11              50          111.57313          82.14346
## 12              51          111.07313          81.64346
## 13              52          110.57313          81.14346
## 14              53          110.07313          80.64346
## 15              54          109.57313          80.14346
## 16              55          109.07313          79.64346
## 17              56          108.57313          79.14346
## 18              57          108.07313          78.64346
## 19              58          107.57313          78.14346
## 20              59          107.07313          77.64346
## 21              60          106.57313          77.14346
## 22              61          106.07313          76.64346
## 23              62          105.57313          76.14346
## 24              63          105.07313          75.64346
## 25              64          104.57313          75.14346
## 26              65          104.07313          74.64346
## 27              66          103.57313          74.14346
## 28              67          103.07313          73.64346
## 29              68          102.57313          73.14346
## 30              69          102.07313          72.64346
## 31              70          101.57313          72.14346
## 32              71          101.07313          71.64346
## 33              72          100.57313          71.14346
## 34              73          100.07313          70.64346
## 35              74           99.57313          70.14346
## 36              75           99.07313          69.64346
## 37              76           98.57313          69.14346
## 38              77           98.07313          68.64346
## 39              78           97.57313          68.14346
## 40              79           97.07313          67.64346
## 41              80           96.57313          67.14346
## 42              81           96.07313          66.64346
## 43              82           95.57313          66.14346
```

```
## 44                83           95.07313          65.64346
## 45                84           94.57313          65.14346
## 46                85           94.07313          64.64346
```

The variations of Price for both good and bad shelve locations is represented graphically below.

```
ggplot(Price_Range, aes(Production_Cost, Price_Range)) + geom_line(aes(y =
Good_Optimal_price_Range, col = "Good Shelve")) + geom_line(aes(y =
Bad_Optimal_Price_Range, col = "Bad Shelve"))
```