Statistical Learning: Chapter 3 Applied Exercise

```
library(dplyr)
library(tidyverse)
library(ggplot2)
library(ggthemes)
```

8. This question involves the use of simple linear regression on the Auto data set.

```
setwd(dir = "~/Desktop/Statistical Learning/dataset")
## read the data set :
Auto  = read.csv("Auto.csv",
                 stringsAsFactors = FALSE,
                 na.strings = "?")
str(Auto)
```

```
## 'data.frame':    397 obs. of  9 variables:
##  $ mpg         : num  18 15 18 16 17 15 14 14 14 15 ...
##  $ cylinders   : int  8 8 8 8 8 8 8 8 8 8 ...
##  $ displacement: num  307 350 318 304 302 429 454 440 455 390 ...
##  $ horsepower  : int  130 165 150 150 140 198 220 215 225 190 ...
##  $ weight      : int  3504 3693 3436 3433 3449 4341 4354 4312 4425 3850 ...
##  $ acceleration: num  12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
##  $ year        : int  70 70 70 70 70 70 70 70 70 70 ...
##  $ origin      : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ name        : chr  "chevrolet chevelle malibu" "buick skylark 320" "plymouth satellite" "amc rebel
```

```
Auto = Auto[complete.cases(Auto),]
dim(Auto)
```

```
## [1] 392   9
```

Part Use the `lm()` to peform the simple linear regression with `mpg` response and horsepower as the predictor

```
str(Auto)
```

```
## 'data.frame':    392 obs. of  9 variables:
##  $ mpg         : num  18 15 18 16 17 15 14 14 14 15 ...
##  $ cylinders   : int  8 8 8 8 8 8 8 8 8 8 ...
##  $ displacement: num  307 350 318 304 302 429 454 440 455 390 ...
##  $ horsepower  : int  130 165 150 150 140 198 220 215 225 190 ...
##  $ weight      : int  3504 3693 3436 3433 3449 4341 4354 4312 4425 3850 ...
##  $ acceleration: num  12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
##  $ year        : int  70 70 70 70 70 70 70 70 70 70 ...
##  $ origin      : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ name        : chr  "chevrolet chevelle malibu" "buick skylark 320" "plymouth satellite" "amc rebel
```

```
lm.fit  = lm(formula = 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
```

- There is a relationship between the predictor and the response.

```
cor(x = Auto$horsepower, y = Auto$mpg, method = c("pearson"))
```

```
## [1] -0.7784268
```

- The relationship is quite negatively strong.
- The predicted `mpg` associated with a `horsepower` of 98. The associated 95% confidence and prediction intervals.

```
## 95% Confidence interval:
predict(object = lm.fit, data.frame(horsepower = c(98)),
        interval = "confidence")
```

```
##        fit      lwr      upr
## 1 24.46708 23.97308 24.96108
```

```
## 95% prediction interval:
predict(object = lm.fit, data.frame(horsepower = c(98)),
        interval = "prediction")
```

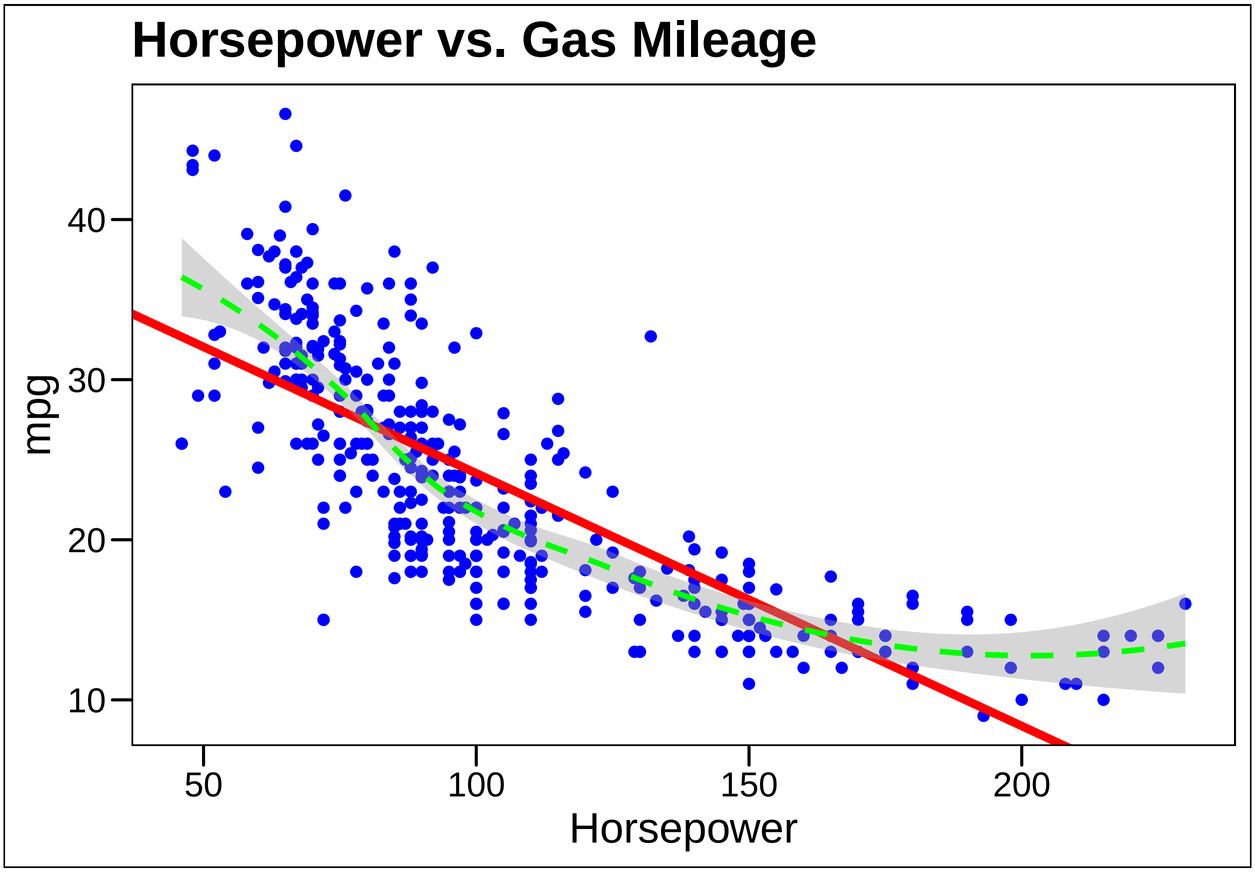
```
##        fit     lwr      upr
## 1 24.46708 14.8094 34.12476
```

- So the predicted `mpg` is 24.46708 associated with `horsepower` of 98.
- The 95% confidence interval is $(23.97308, 24.96108)$. The 95% prediction interval is $(14.8094, 34.12476)$.

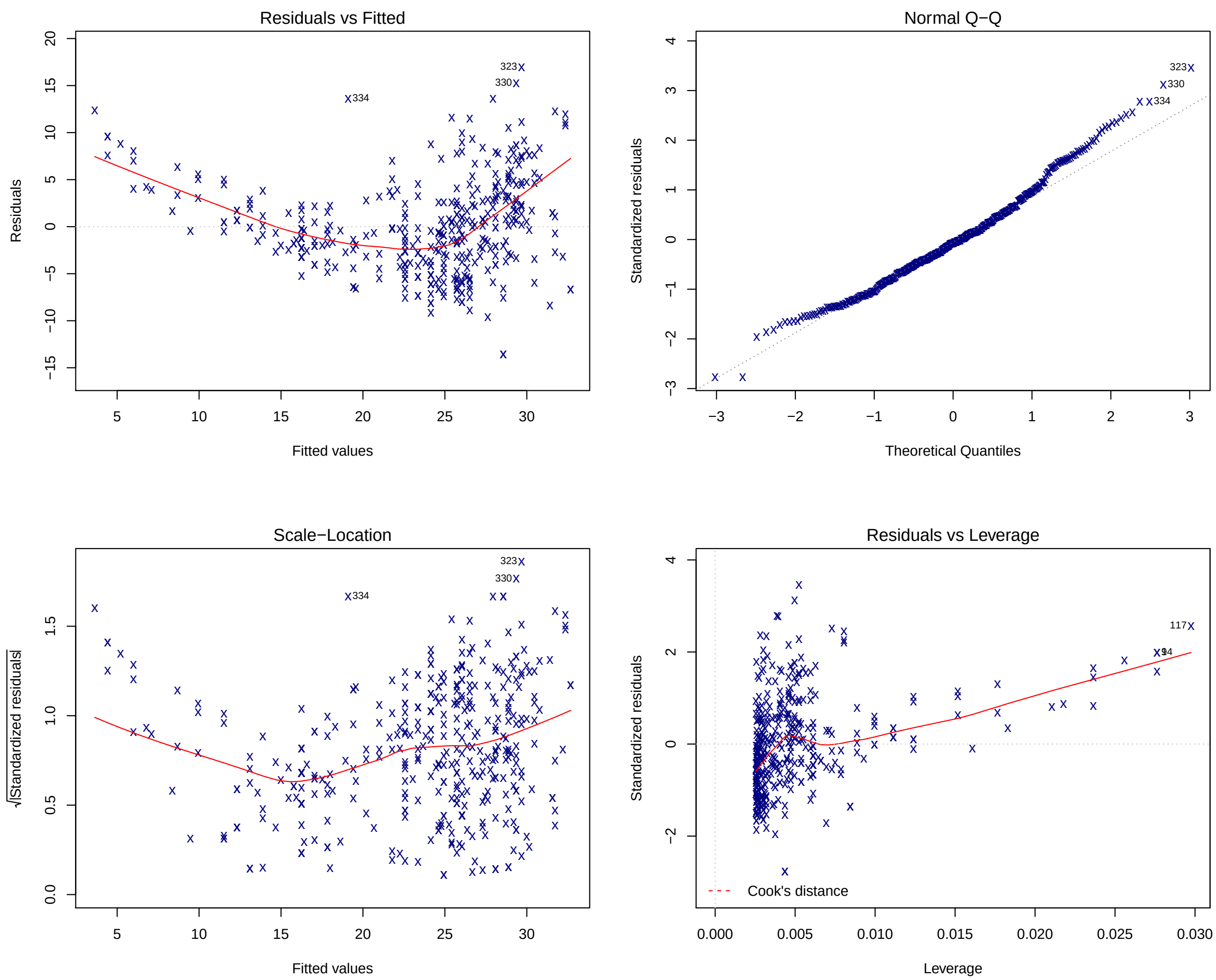
Part(b)

```
intercept.lmfit = as.numeric(lm.fit$coefficients[1])
slope.lmfit = as.numeric(lm.fit$coefficients[2])

scatterplot = ggplot(data = Auto, aes(x = horsepower, y = mpg))+
  geom_point(color = "blue") +
  xlab(label = "Horsepower") +
  ylab(label = "mpg") +
  ggtitle(label = "Horsepower vs. Gas Mileage") +
  theme_base() + geom_abline(slope = slope.lmfit,
                             intercept = intercept.lmfit,
                             color = "red",
                             size = 1.5) +
```

```r
  geom_smooth(color = "green",
              size = 1,
              linetype = "dashed",
              method = 'loess')

print(scatterplot)
```




```r
par(mfrow = c(2,2))

plot(lm.fit, pch  = "x", col = "navy")
```



- There is a problem of non-constance variance assumption. The residual vs. fitted values suggest the heteroscadascity of variance.

11. To begin we generate the predictor x and a response y as follows:

```
set.seed(1)
x = rnorm(n = 100)
y = 2*x + rnorm(100)
```

Part (a)

```
### without interception:
lm.fit = lm(y ~ x + 0)

summary(lm.fit)
```

```
##
## Call:
## lm(formula = y ~ x + 0)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -1.9154 -0.6472 -0.1771  0.5056  2.3109
##
## Coefficients:
##   Estimate Std. Error t value Pr(>|t|)
## x   1.9939     0.1065   18.73   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9586 on 99 degrees of freedom
## Multiple R-squared:  0.7798, Adjusted R-squared:  0.7776
## F-statistic: 350.7 on 1 and 99 DF,  p-value: < 2.2e-16
```

- The coefficient estimate $\hat{\beta} = 1.9939$, the standard error is 0.1065, t-value is 18.73 and p-value is extremely small as it is less than 2×10^{-16}.

Part (b)

- We perfom the regression of x onto y without an intercept, and report the estimated coefficient, SE, t-statistic and p-values.

```
lm.fit2 = lm(x ~ y + 0)
summary(lm.fit2)
```

```
##
## Call:
## lm(formula = x ~ y + 0)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.8699 -0.2368  0.1030  0.2858  0.8938
##
## Coefficients:
##   Estimate Std. Error t value Pr(>|t|)
## y  0.39111    0.02089   18.73   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4246 on 99 degrees of freedom
## Multiple R-squared:  0.7798, Adjusted R-squared:  0.7776
## F-statistic: 350.7 on 1 and 99 DF,  p-value: < 2.2e-16
```

- The estimated coefficient is 0.39111, $SE = 0.02089$, $t - value = 18.73$ and the p-value is extremely small less than 2×10^{-16}.

Exercise 13: Simulating the data

Part (a)

- Using `rnorm()` create the vector `x` containing 100 observations from a Normal(0,1) distribution. This represents feature X.

```
set.seed(1)
X = rnorm(n = 100, mean = 0, sd = 1)
```

Part (b)

- Create the vector `eps` containing 100 observations from a Normal(0,0.25) with mean zero and variance 0.25.

```r
set.seed(1)
eps = rnorm(n = 100,
            mean = 0,
            sd = sqrt(0.25))
```

Part (c)

- Generate the vector `y` according to the model:

$$Y = -1 + 0.5X + \epsilon$$

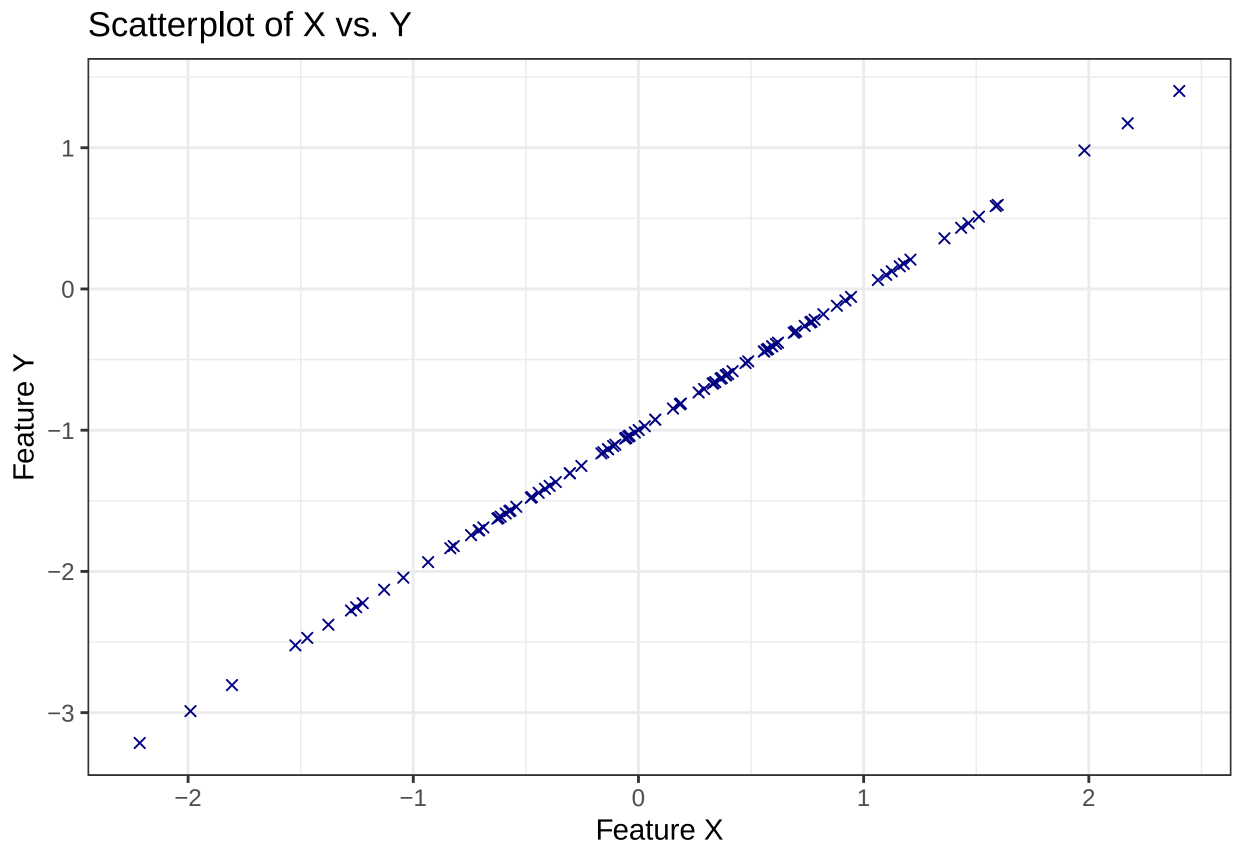
```r
set.seed(1)
Y = -1 + 0.5*X + eps
Dataset = cbind(X, Y, eps)
Dataset = as.data.frame(Dataset)
```

- The length of vector `y` is 100. The value of β_0 is -1, and β_1 is 0.5.

Part (d)

- The scatterplot displays the relationship between `x` and `y`.

```r
plot1 = ggplot(data = Dataset, aes(x = X, y = Y)) +
  geom_point(pch = 4, color = "navy") +
  ggtitle(label = "Scatterplot of X vs. Y") +
  xlab(label = "Feature X") +
  ylab(label = "Feature Y") +
  theme_bw()

print(plot1)
```



Part (e)

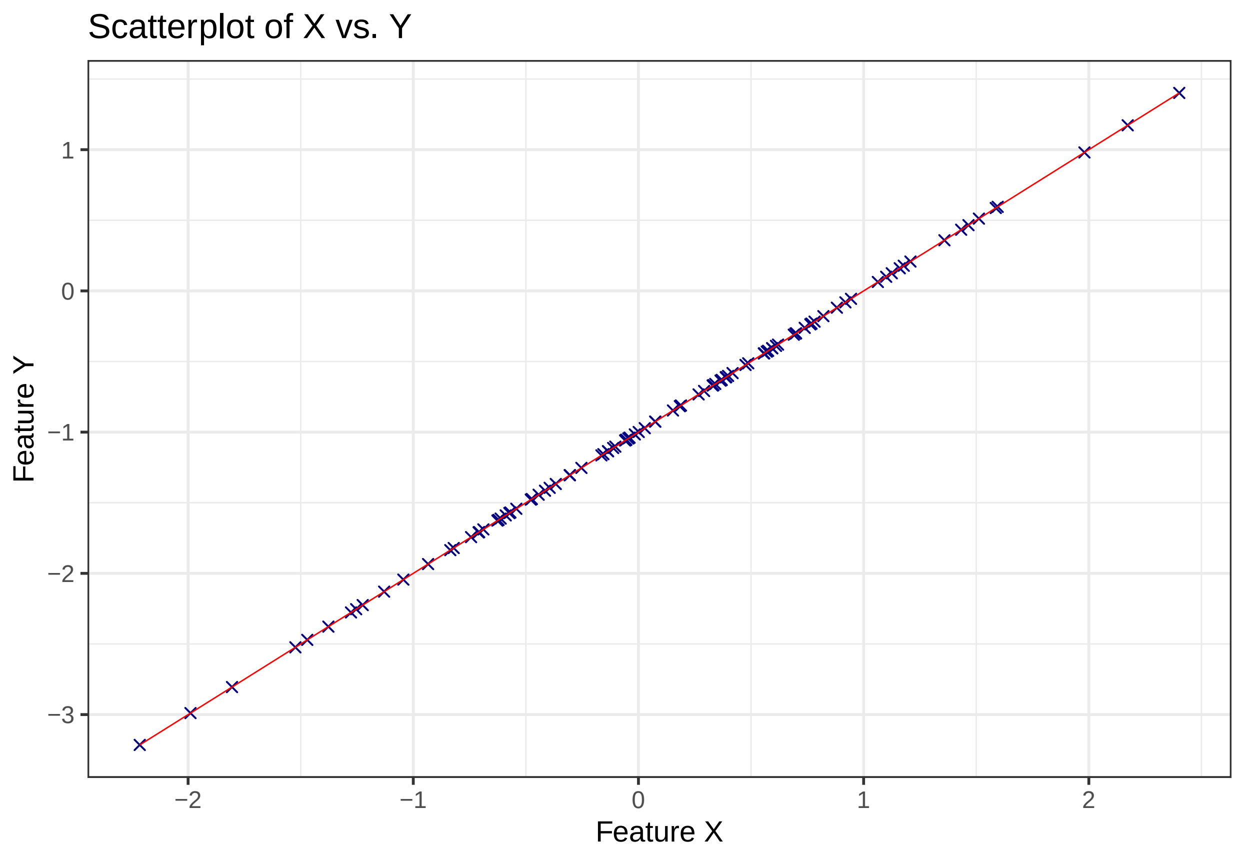
- We fit a least squares linear model to predict y using x:

```
lm.fit = lm(Y ~ X , data = Dataset)
summary(lm.fit)
```

```
##
## Call:
## lm(formula = Y ~ X, data = Dataset)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -3.230e-16 -1.043e-16 -2.620e-17  5.079e-17  2.322e-15
##
## Coefficients:
##               Estimate Std. Error    t value Pr(>|t|)
## (Intercept) -1.000e+00  2.757e-17 -3.628e+16   <2e-16 ***
## X            1.000e+00  3.062e-17  3.266e+16   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.736e-16 on 98 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 1.067e+33 on 1 and 98 DF,  p-value: < 2.2e-16
```

Part (f)

```r
plot2 = ggplot(data = Dataset, aes(x = X, y = Y)) +
  geom_point(pch = 4, color = "navy") +
  ggtitle(label = "Scatterplot of X vs. Y") +
  xlab(label = "Feature X") +
  ylab(label = "Feature Y") +
  theme_bw() +
  geom_smooth(method = "lm", color = "red",
              size = 0.25, lty = 1)

print(plot2)
```



Part (g)

- We fit a polynomial regression predicting y using x and x^2:

```r
poly.fit = lm(Y ~ X + I(X^2), data = Dataset)
summary(poly.fit)
```

```
## Warning in summary.lm(poly.fit): essentially perfect fit: summary may be
## unreliable

##
## Call:
## lm(formula = Y ~ X + I(X^2), data = Dataset)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -3.091e-16 -1.063e-16 -2.840e-17  5.297e-17  2.321e-15
```

```
##
## Coefficients:
##               Estimate Std. Error    t value Pr(>|t|)
## (Intercept) -1.000e+00  3.377e-17 -2.961e+16   <2e-16 ***
## X            1.000e+00  3.100e-17  3.226e+16   <2e-16 ***
## I(X^2)      -3.115e-18  2.433e-17 -1.280e-01    0.898
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.75e-16 on 97 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 5.28e+32 on 2 and 97 DF,  p-value: < 2.2e-16
```

- There is no evidence that the quadratic term improves the model fit because the p-value associated with quadratic term is large as $p = 0.898$.

Exercise 14: This problem involves on `collinearity` problem:

Part (a)

```r
set.seed(1)

x1 = runif(n = 100)
x2 = 0.5*x1 + rnorm(100)/10
y = 2 + 2*x1 + 0.3*x2 + rnorm(100)

DataSet = cbind(x1,x2,y)
DataSet = as.data.frame(DataSet)
```

- The form of the linear model:

$$Y = 2 + 2X_1 + 0.3X_2 + \epsilon$$

The regression coefficients:

$$\beta_0 = 2; \quad \beta_{1,1} = 2; \quad \beta_{1,2} = 0.3$$

Part(b)

- The correlation between `x1` and `x2`:

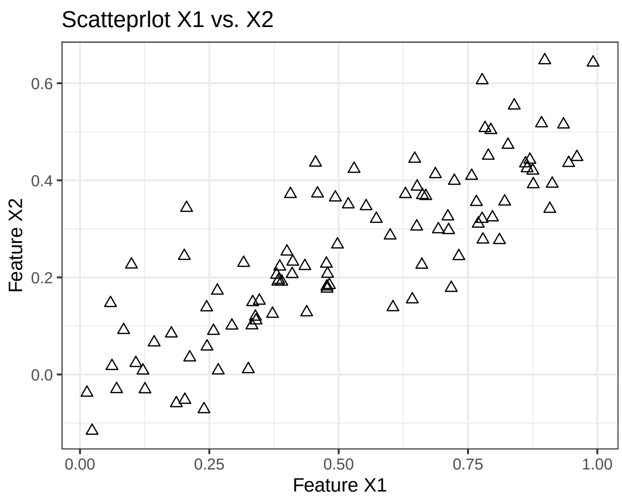
```r
Cor.X1.X2 = cor(x = DataSet$x1, y = DataSet$x2)

print(list(
  Cor.X1.X2 = Cor.X1.X2
))
```

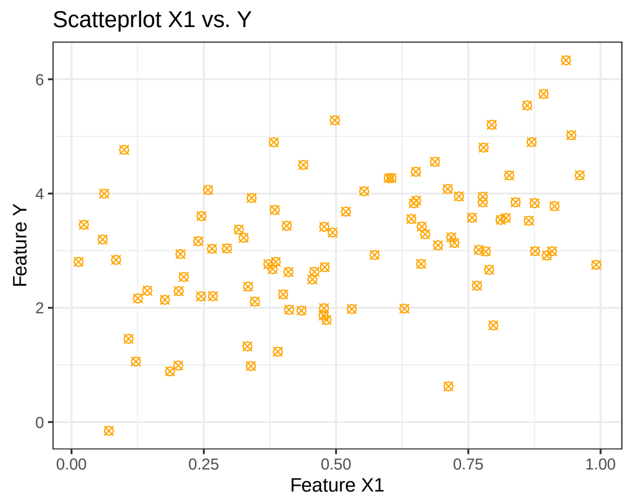
```
## $Cor.X1.X2
## [1] 0.8351212
```

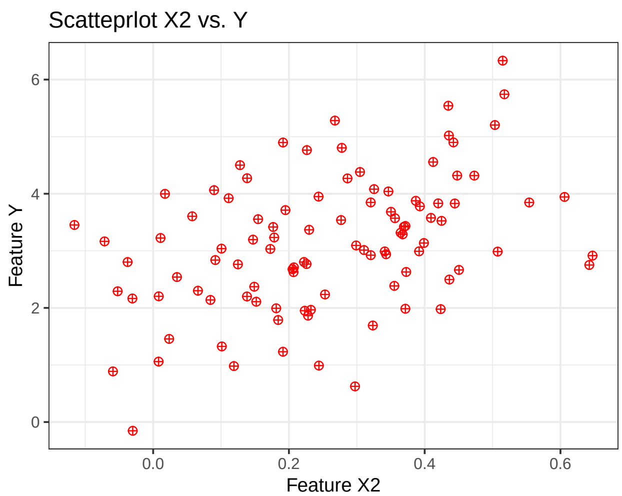
- The scatterplot between the variables:

```r
library(gridExtra)


Plot1 = ggplot(data = DataSet,
```

```
               aes(x = x1, y = x2)) +
  geom_point(pch = 2, color = "black", size = 2) +
  ggtitle(label = "Scatteprlot X1 vs. X2") +
  xlab(label = "Feature X1") +
  ylab(label = "Feature X2") +
  theme_bw()


Plot2 = ggplot(data = DataSet,
               aes(x = x1, y = y)) +
  geom_point(pch = 13, color = "orange", size = 2) +
  ggtitle(label = "Scatteprlot X1 vs. Y") +
  xlab(label = "Feature X1") +
  ylab(label = "Feature Y") +
  theme_bw()


Plot3 = ggplot(data = DataSet,
               aes(x = x2, y = y)) +
  geom_point(pch = 10, color = "red", size = 2) +
  ggtitle(label = "Scatteprlot X2 vs. Y") +
  xlab(label = "Feature X2") +
  ylab(label = "Feature Y") +
  theme_bw()

grid.arrange(Plot1,
             Plot2,
             Plot3,ncol = 2)
```







Part (c)

- We fit a least squares regression to predict y using x1 and x2:

```r
lm.fit = lm(y ~ x1 + x2, data = DataSet)

summary(lm.fit)
```

```
##
## Call:
## lm(formula = y ~ x1 + x2, data = DataSet)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.8311 -0.7273 -0.0537  0.6338  2.3359
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.1305     0.2319   9.188 7.61e-15 ***
## x1            1.4396     0.7212   1.996   0.0487 *
## x2            1.0097     1.1337   0.891   0.3754
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 1.056 on 97 degrees of freedom
## Multiple R-squared:  0.2088, Adjusted R-squared:  0.1925
## F-statistic:  12.8 on 2 and 97 DF,  p-value: 1.164e-05
```

- The estimated coefficients are:
- $\hat{\beta}_0 = 2.1305$.
- $\hat{\beta}_1 = 1.4396$.
- $\hat{\beta}_2 = 1.0097$
- We can reject the null hypothesis for $\beta_1 = 0$ because the p-value is 0.0487 which shows there is some evidence to accept the alternative hypothesis. We cannot reject the null hypothesis for $\beta_2 = 0$ because the p-value is 0.3754 which is a large p value.

Part (d)

- We fit a linear regression to predict y using x1 only

```
lm.fit2 = lm(y ~ x1, data = DataSet)

summary(lm.fit2)
```

```
##
## Call:
## lm(formula = y ~ x1, data = DataSet)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.89495 -0.66874 -0.07785  0.59221  2.45560
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.1124     0.2307   9.155 8.27e-15 ***
## x1            1.9759     0.3963   4.986 2.66e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.055 on 98 degrees of freedom
## Multiple R-squared:  0.2024, Adjusted R-squared:  0.1942
## F-statistic: 24.86 on 1 and 98 DF,  p-value: 2.661e-06
```

- We have strong evidence to reject the null hypothesis for $\beta_1 = 0$ because the p-value is very small.

Part (e)

- We fit a linear regression to predict y using x2 only

```
lm.fit3 = lm(y ~ x2, data = DataSet)

summary(lm.fit3)
```

```
##
## Call:
## lm(formula = y ~ x2, data = DataSet)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -2.62687 -0.75156 -0.03598  0.72383  2.44890
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.3899     0.1949   12.26  < 2e-16 ***
## x2            2.8996     0.6330    4.58 1.37e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.072 on 98 degrees of freedom
## Multiple R-squared:  0.1763, Adjusted R-squared:  0.1679
## F-statistic: 20.98 on 1 and 98 DF,  p-value: 1.366e-05
```

- We have strong evidence to reject the null hypothesis for $\beta_2 = 0$ because the p-value is very small.