Assignment 3

Ling Fei Zhang, 260985358

2022-11-06

```
library(readxl)
library(here)
library(matlib)
library(formatR)
library(gridExtra)
library(tidyverse)
knitr::opts_chunk$set(tidy.opts=list(width.cutoff=60),
                      tidy=TRUE,
                      echo=TRUE,
                      comment=NA,
                      message=FALSE,
                      warning=FALSE)
```

Question 1

```
salary <- read_excel("salary.xlsx")
x1 <- salary$SPENDING/1000
y <- salary$SALARY
fit.salary <- lm(y ~ x1)
summary(fit.salary)
```

```

Call:
lm(formula = y ~ x1)

Residuals:
    Min      1Q  Median      3Q     Max 
-3848.0 -1844.6  -217.5  1660.0  5529.3 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  12129.4     1197.4   10.13 1.31e-13 ***
x1            3307.6      311.7   10.61 2.71e-14 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2325 on 49 degrees of freedom
Multiple R-squared:  0.6968,	Adjusted R-squared:  0.6906 
F-statistic: 112.6 on 1 and 49 DF,  p-value: 2.707e-14
```

part a)

```
m <- cbind(1, x1)
t1 <- t(m) %*% m
t2 <- t(m) %*% y
betas <- solve(t1, t2)
betas  #betas[2] = slope and betas[1] = intercept
```

```
        [,1]
   12129.371
x1  3307.585
```

part b)

```
# first one
yhat <- m %*% betas
res <- y - yhat
sum(res)
```

```
[1] -1.67347e-10
```

```
# second one
x_mean <- mean(x1)
sum(res * (x1 - x_mean))
```

```
[1] 2.215698e-09
```

```
# third one
sum(res * yhat)
```

```
[1] 3.29758e-06
```

Note that there are small floating point computational calculation errors, hence the non-zero results.

part c)

We are looking for $ESE(\hat{\beta}_0)$. From the table, we have $ESE(\hat{\beta}_0) = \frac{\hat{\beta}_0}{t_0} = \frac{12129.4}{10.13}$

```
12129.4/10.13
```

```
[1] 1197.374
```

Then, using the data directly, we have that

```
Sxx <- sum((x1 - mean(x1))^2)
SSres <- sum(res^2)
MSres <- SSres/(length(y) - 2)
ese_beta0 <- sqrt(MSres * (1/length(y) + mean(x1)^2/Sxx))
ese_beta0
```

```
[1] 1197.351
```

Note that there is a small difference between the two calculated Estimated Standard Error, but it is simply due to computational rounding of floating points. However, apart from the small rounding imprecision, we can see that the results we obtained is the same as the ESE calculated in the summary.

part d)

We are looking for $\hat{\sigma}$.

```
sigma <- sqrt(MSres)
sigma
```

```
[1] 2324.779
```

part e)

Here, we set $\alpha = 0.05$. Our alternative hypothesis test is

$$H_1 : \beta_1 \neq 0$$

The decision rule is that we do not reject $H_1 : \beta_1 \neq 0$ if the corresponding p-value $\leq \alpha$. We can calculate the p-value as follows:

```
ese_b1 <- sqrt(sigma^2/Sxx)
t <- betas[2]/ese_b1
t
```

```
[1] 10.61129
```

```
# p-value
p <- pt(-abs(t), length(y) - 2)
2 * p  #two tails
```

```
[1] 2.706871e-14
```

As we can see, the calculated p-value of β_1 is the same as the one given in the summary, which is $2.71e - 14$, much smaller than α. We therefore would **not** reject $H_1 : \beta_1 \neq 0$. As a result, we would say that there **is** a linear association between SALARY and SPENDING.

Question 2

part a)

Below, I calculate the confidence interval with $\alpha = 0.01$. First, we can say that the confidence interval serves to measure the overall quality of the regression line. We can interpret the confidence interval for β_1 as follows: If we were to repeatedly sample data from *salary*, and construct a 99% confidence interval, then 99% of those intervals will contain the true value of β_1.

```
confint(fit.salary, level = 0.99)
```

```
                0.5 %    99.5 %
(Intercept) 8920.528 15338.214
x1          2472.233  4142.937
```

Clearly, we can see that the value 3500 is contained within the confidence interval for β_1.

The hypothesis tests are as follows:

$$H_0 : \beta_1 = 3500$$
$$H_1 : \beta_1 \neq 3500$$

We can also calculate the t-statistic and p-value as follows:

```
# t statistic
t <- (betas[2] - 3500)/ese_b1
t
```

```
[1] -0.6172998
```

```
# p-value
p <- pt(-abs(t), length(y) - 2)
2 * p  #two tails
```

```
[1] 0.5398951
```

Since our significance level is $\alpha = 0.01$, we can see that our p-value for β_1 is much bigger than α. As a result, this means that $H_0 : \beta_1 = 3500$ **is not** rejected and $H_1 : \beta_1 \neq 3500$ **is** rejected. Together, this means that there is a linear association between SALARY and SPENDING with $\alpha = 0.01$.

part b)

```
SSR <- sum((yhat - mean(y))^2)
p <- 2
Fstat <- (SSR/(p - 1))/(SSres/(length(y) - p))
Fstat
```

```
[1] 112.5995
```

```
summary(fit.salary)$fstatistic
```

```
   value    numdf    dendf
112.5995   1.0000  49.0000
```

As we can see, the F-statistic I calculated is the same as the one from the summary. We can also see from the summary that we are dealing with a F distribution with 1 DF on the numerator, and 49 DF on the denominator. The p-value can be calculated as follows:

```
p_val <- pf(Fstat, p - 1, length(y) - p, lower.tail = FALSE)
p_val
```

```
[1] 2.706871e-14
```

This gives us a p-value of $2.707e-14$, with a corresponding $\alpha = 0.05$. Our null and alternative hypothesis are as follows:

$$H_0 : \beta_1 = 0$$
$$H_1 : \beta_1 \neq 0$$

The decision criteria is as follows: if p-value $< \alpha = 0.05$, then the null hypothesis is rejected. At the same time, the alternative hypothesis is not be rejected. This is in fact the case as p-value $= 2.707e-14 < \alpha$. This indicates that there is a linear association between SALARY and SPENDING. This is not surprising, as the F-test is the same as the t-test in simple linear regression from part e) of Q1.

part c)

```r
SST1 <- sum((y - mean(y))^2)
SST2 <- SSres + SSR
SST1
```

```
[1] 873380265
```

```r
SST2
```

```
[1] 873380265
```

We have thus verified the sum of squares decomposition numerically.

part d)

We simply want to predict the value of y when x = 5 ($5000).

```r
ynew <- betas[1] + betas[2] * 5
ynew
```

```
[1] 28667.3
```

part e)

The estimated variance of $\hat{Y}^{new}$ is

$$\begin{aligned} Var_{Y|X,X_1^{new}}(\hat{Y}^{new}|X, X_1^{new}) &= X_1^{new} Var_{Y|X}(\hat{\beta}|X)(X_1^{new})^T \\ &= X_1^{new}(\sigma^2(X^TX)^{-1})(X_1^{new})^T \qquad \text{From Handout 7} \end{aligned}$$

Using the plug in principle, we can replace σ by $\hat{\sigma}$.

```r
x1_new <- t(matrix(c(1, 5)))
var = x1_new %*% (sigma^2 * inv(t(m) %*% m)) %*% t(x1_new)
sqrt(var)  #looking for squared root of estimated variance
```

```
         [,1]
[1,] 520.6061
```

part f)

First, we have obtained from part d) that $\hat{y}^{new} = 28667.3$. We are thus looking for the lower and upper bound of both the confidence interval and the prediction interval. Note also that we have $\alpha = 0.05$.

```r
# confidence interval
ese_error1 <- sqrt(MSres * (1/length(y) + (5 - mean(x1))^2/Sxx))
t_value1 <- qt(0.05/2, df = length(y) - 2, lower.tail = FALSE)
conf_interval_lower <- ynew - t_value1 * ese_error1
conf_interval_upper <- ynew + t_value1 * ese_error1
conf_interval_lower
```

```
[1] 27621.1
```

```r
conf_interval_upper
```

```
[1] 29713.49
```

```r
# confidence interval from R
predict(fit.salary, newdata = data.frame(x1 = 5), interval = "confidence")
```

```
      fit     lwr      upr
1 28667.3 27621.1 29713.49
```

```r
# prediction interval
ese_error2 <- sqrt(MSres * (1 + 1/length(y) + (5 - mean(x1))^2/Sxx))
t_value2 <- qt(0.05/2, df = length(y) - 2, lower.tail = FALSE)
pred_interval_lower <- ynew - t_value2 * ese_error2
pred_interval_upper <- ynew + t_value2 * ese_error2
pred_interval_lower
```

```
[1] 23879.77
```

```r
pred_interval_upper
```

```
[1] 33454.82
```

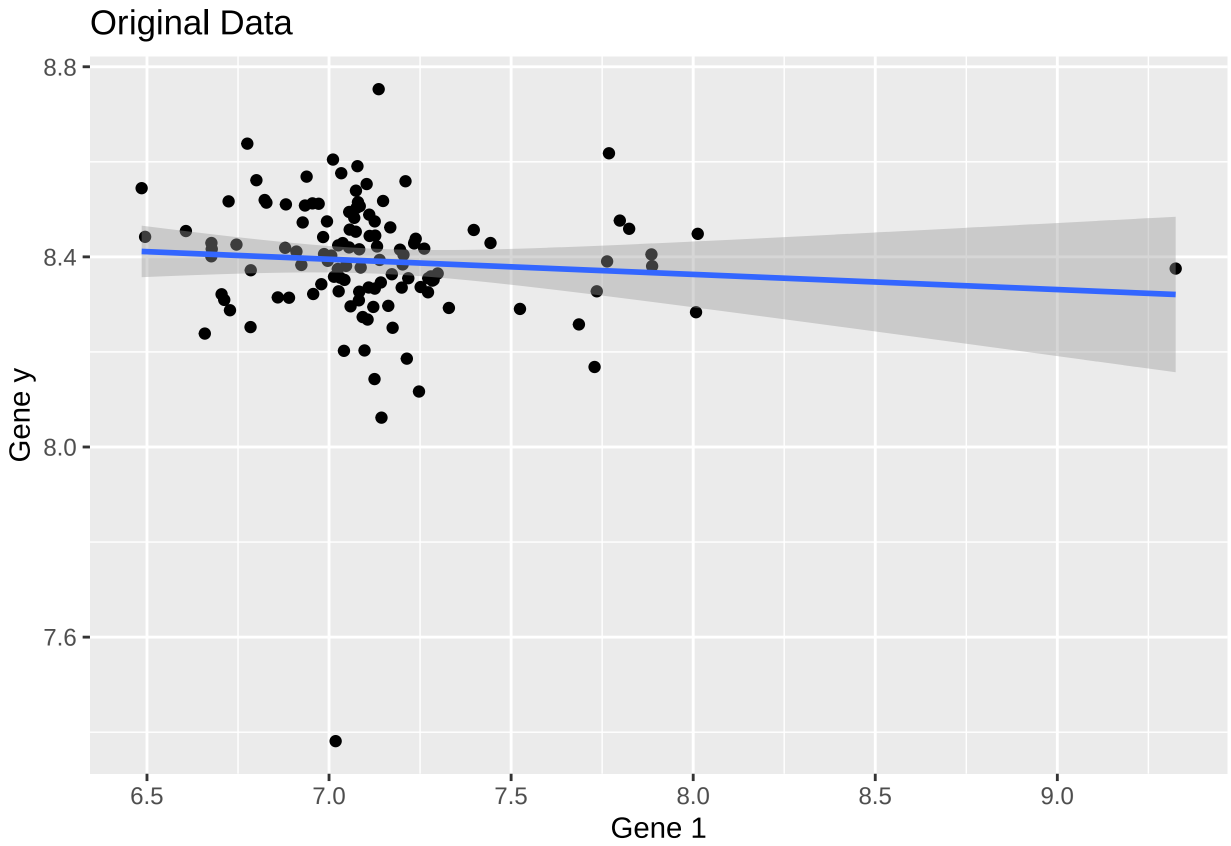
```r
# prediction interval from R
predict(fit.salary, newdata = data.frame(x1 = 5), interval = "prediction")
```

```
      fit      lwr      upr
1 28667.3 23879.77 33454.82
```

Question 3

part a)

```
genetic_data <- read_excel("genetic_data2.xlsx")
y <- unlist(genetic_data[, 1])
g1 <- unlist(genetic_data[, 2])

genetic_data %>%
    ggplot(aes(x = g1, y = y)) + geom_point() + geom_smooth(method = lm) +
    labs(x = "Gene 1", y = "Gene y", title = "Original Data")
```



Let T_1 be the observed value of the t-test statistic. We first compute T_1 on the original dataset under the assumption of the null hypothesis

$$H_0 : \beta_1 = 0.$$

We would get

$$\begin{aligned} T_1 &= \frac{\hat{\beta}_1 - 0}{ese(\hat{\beta}_1)} \\ &= \frac{\hat{\beta}_1}{ese(\hat{\beta}_1)} \\ &= -0.8584019 \end{aligned}$$

```
# Extract t-stat for beta1
s <- summary(lm(y ~ g1))
s$coef
```

```
               Estimate Std. Error    t value     Pr(>|t|)
(Intercept)  8.61830682 0.26531352 32.4834819 1.087158e-60
g1          -0.03189849 0.03716032 -0.8584019 3.924108e-01
```

```r
tstat_beta1 <- s$coef[2, 3]
tstat_beta1
```

```
[1] -0.8584019
```

Next, we randomly permute the data. We will do this 1000 times and compute T_1 again using the permuted data. Let $T_1^1, ..., T_1^{1000}$ denote the resulting permuted t-statistic values.

```r
permutation_test <- function(output, input, nrep = 1000) {
    tstat_vec <- rep(0, nrep)
    for (i in 1:nrep) {
        # permute y
        y_permuted <- sample(output)

        # refit the regression
        m_permuted <- lm(y_permuted ~ input)

        # extract regression result
        results <- summary(m_permuted)

        # extract the t-statistic for b1
        tstat_vec[i] <- results$coef[2, 3]
    }
    return(tstat_vec)
}

tstat_vec <- permutation_test(y, g1)
```

Finally, we compute the approximated *p-value* as follows:

$$p - value = \frac{1}{1000}\sum_{j=1}^{1000} I(|T_1^j| > |T_1|)$$

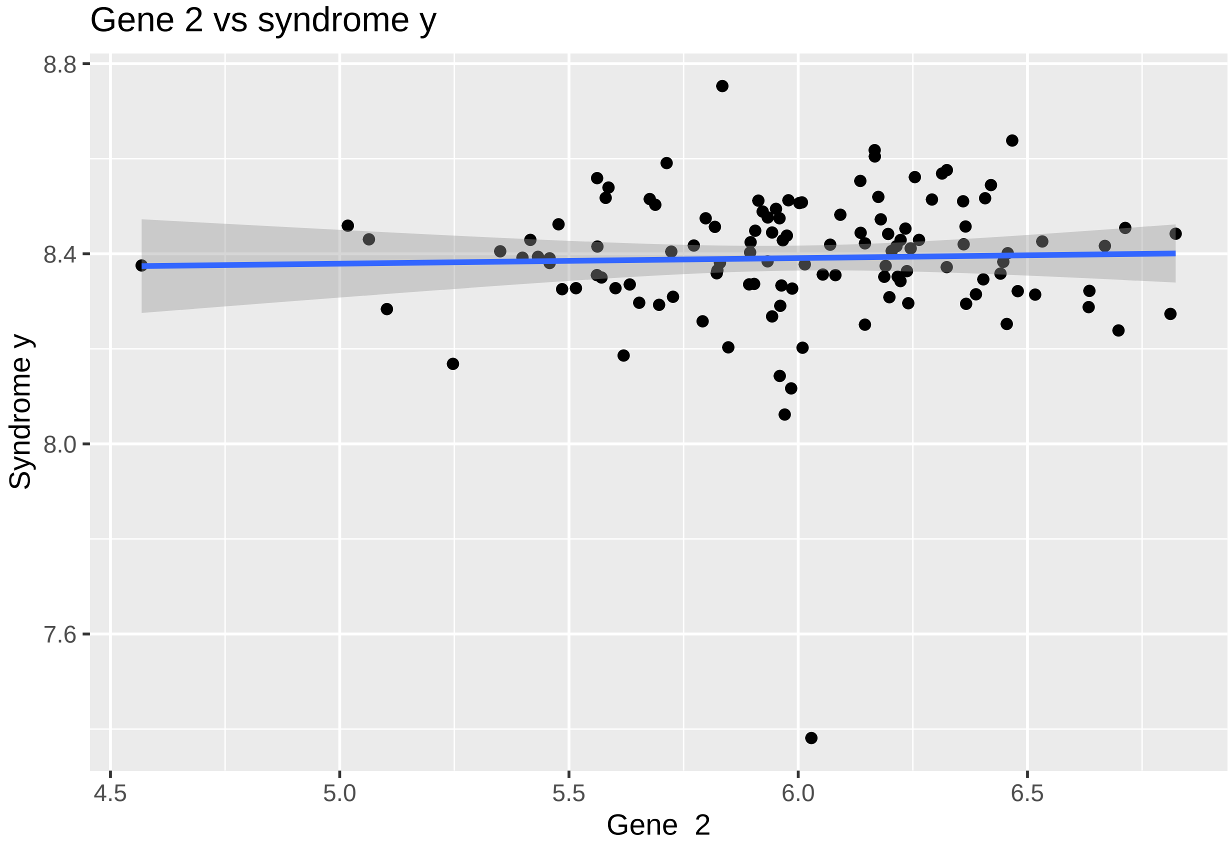
```r
p_value = sum(tstat_vec > abs(tstat_beta1) | tstat_vec < -abs(tstat_beta1))/1000
p_value
```

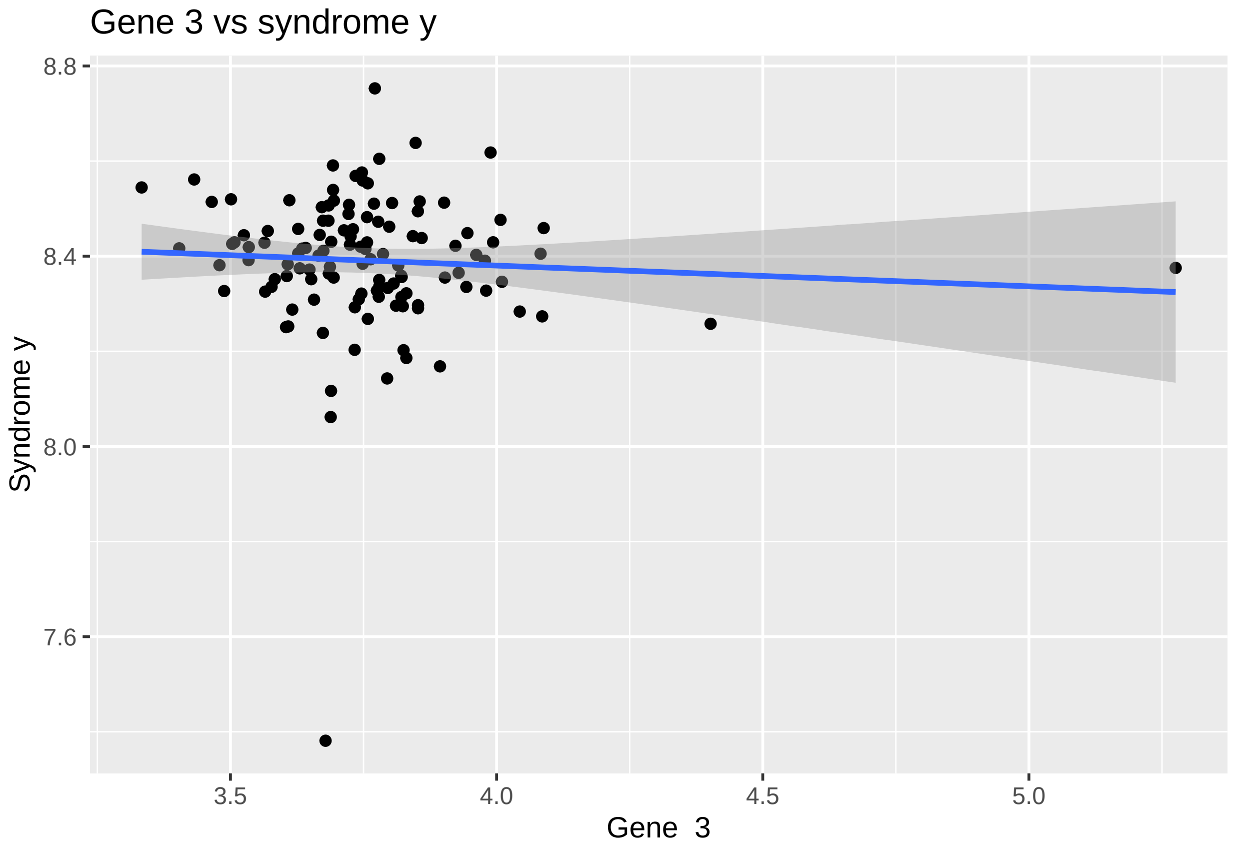
```
[1] 0.316
```

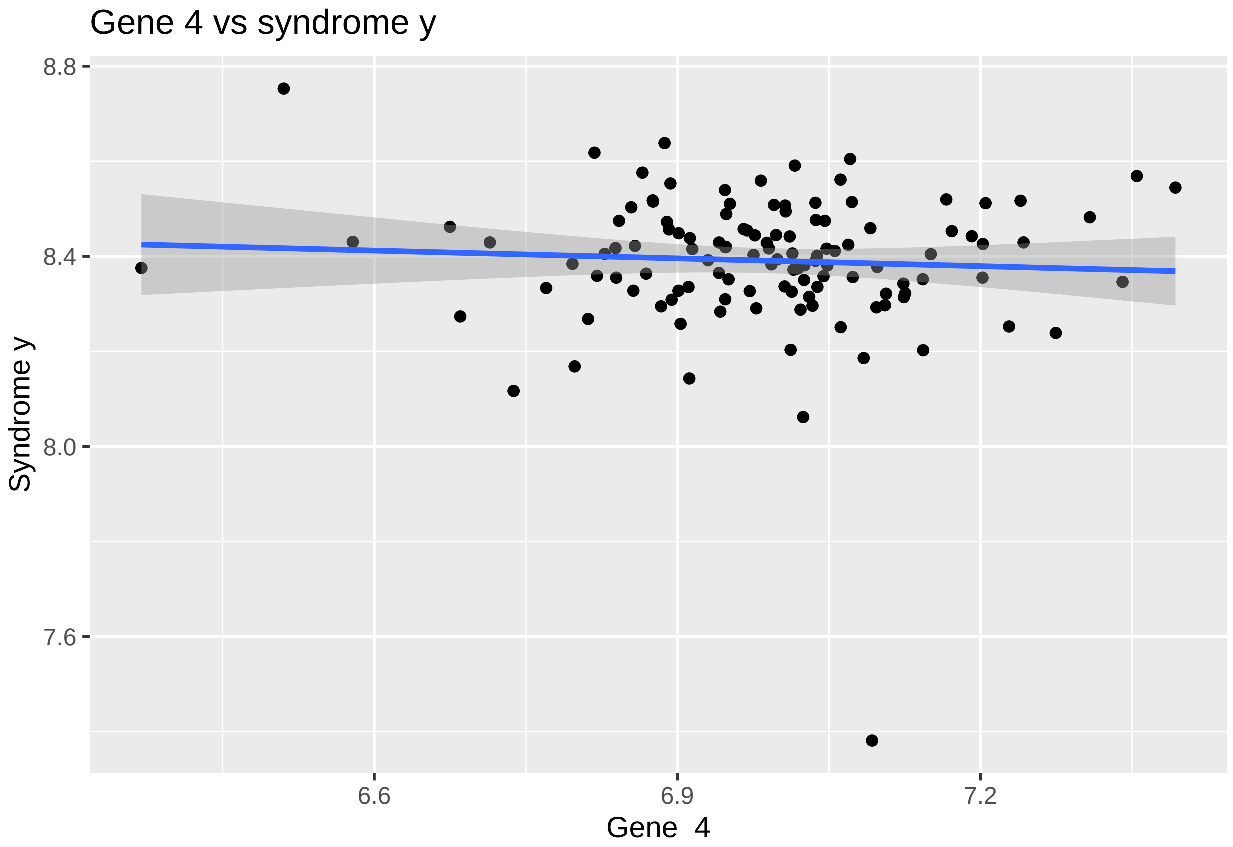
First note that our sampled p-value is really close to the p-value shown in the summary. Next, Suppose we use the conventional $\alpha = 0.05$, then we clearly see that *p-value* $> \alpha$, which means that we **do not reject** our null hypothesis. As a result, we conclude that *g1* is not significantly associated with y.

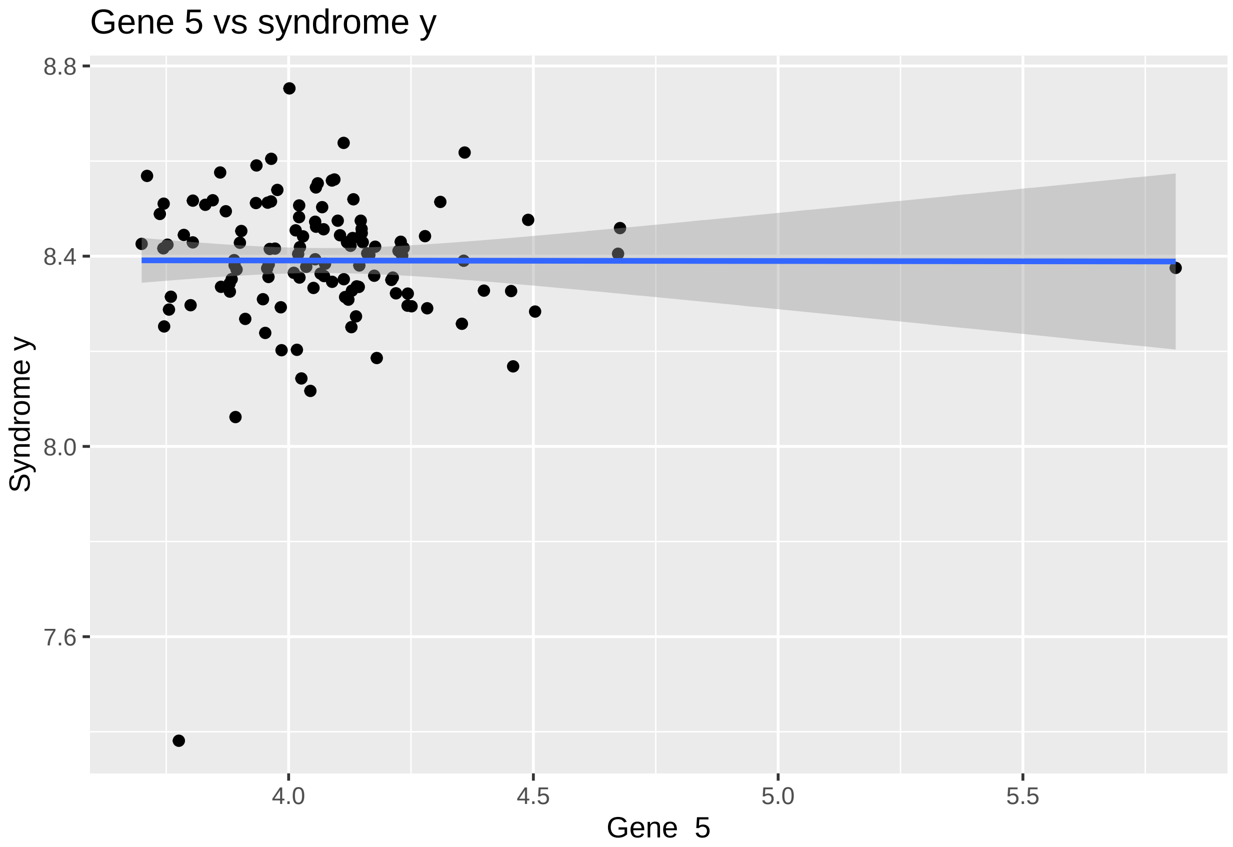
part b)

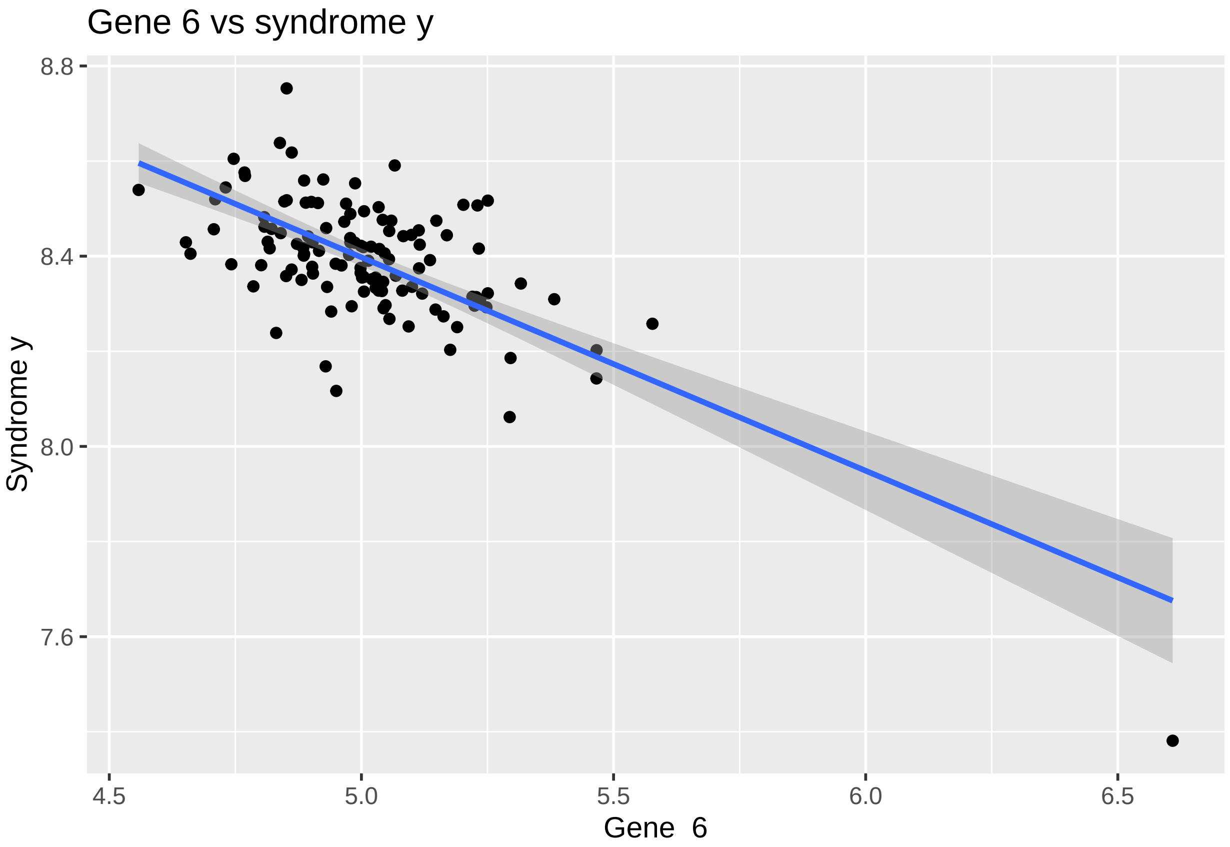
```r
p_values <- c()
for (i in 1:9) {
    # extracts g2, ..., g10
    x <- unlist(genetic_data[, i + 2])
    p_values[i] <- summary(lm(y ~ x))$coef[2, 4]  #append p-value

    # plot of gene vs syndrome
    print(genetic_data %>%
        ggplot(aes(x = unlist(genetic_data[, i + 2]), y = y)) +
        geom_point() + geom_smooth(method = lm) + labs(x = paste("Gene ",
        i + 1), y = "Syndrome y", title = paste("Gene", i + 1,
        "vs syndrome y")))

}
```

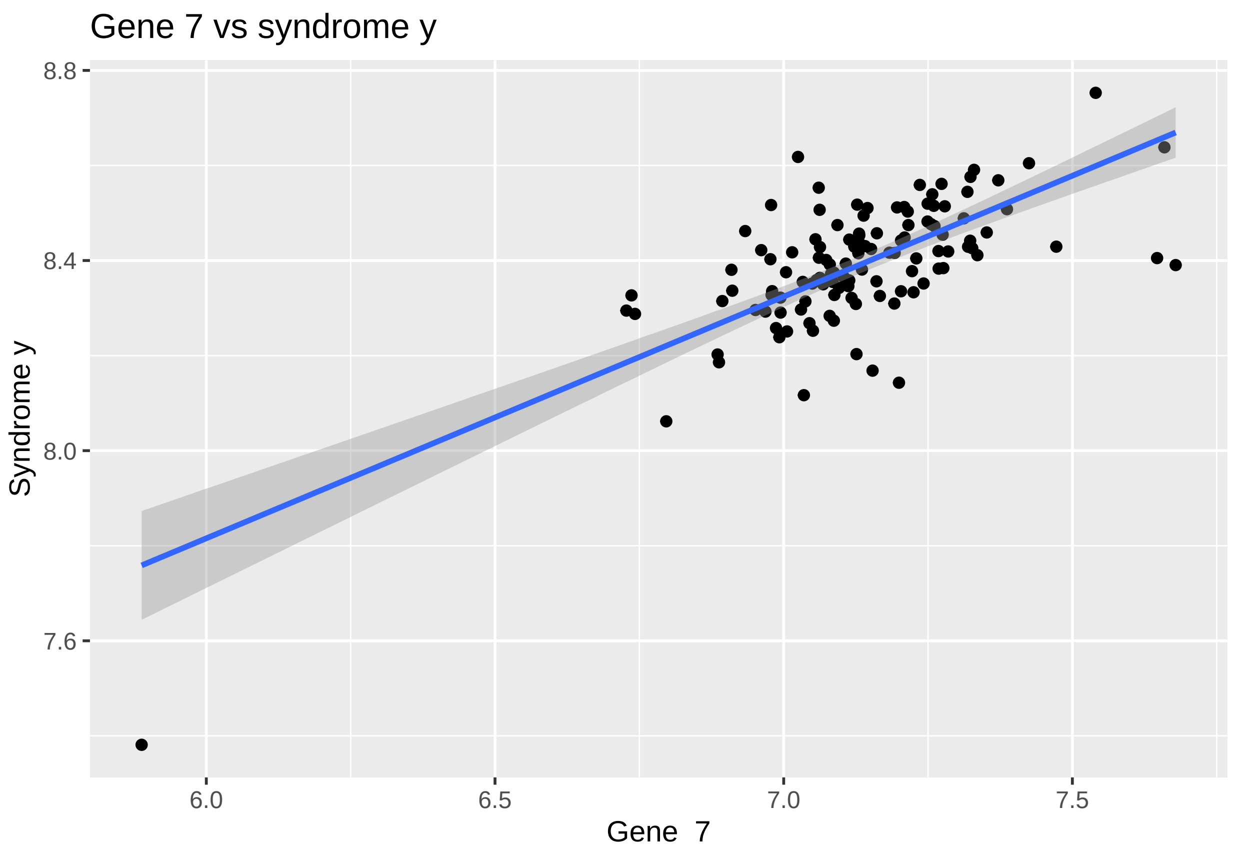


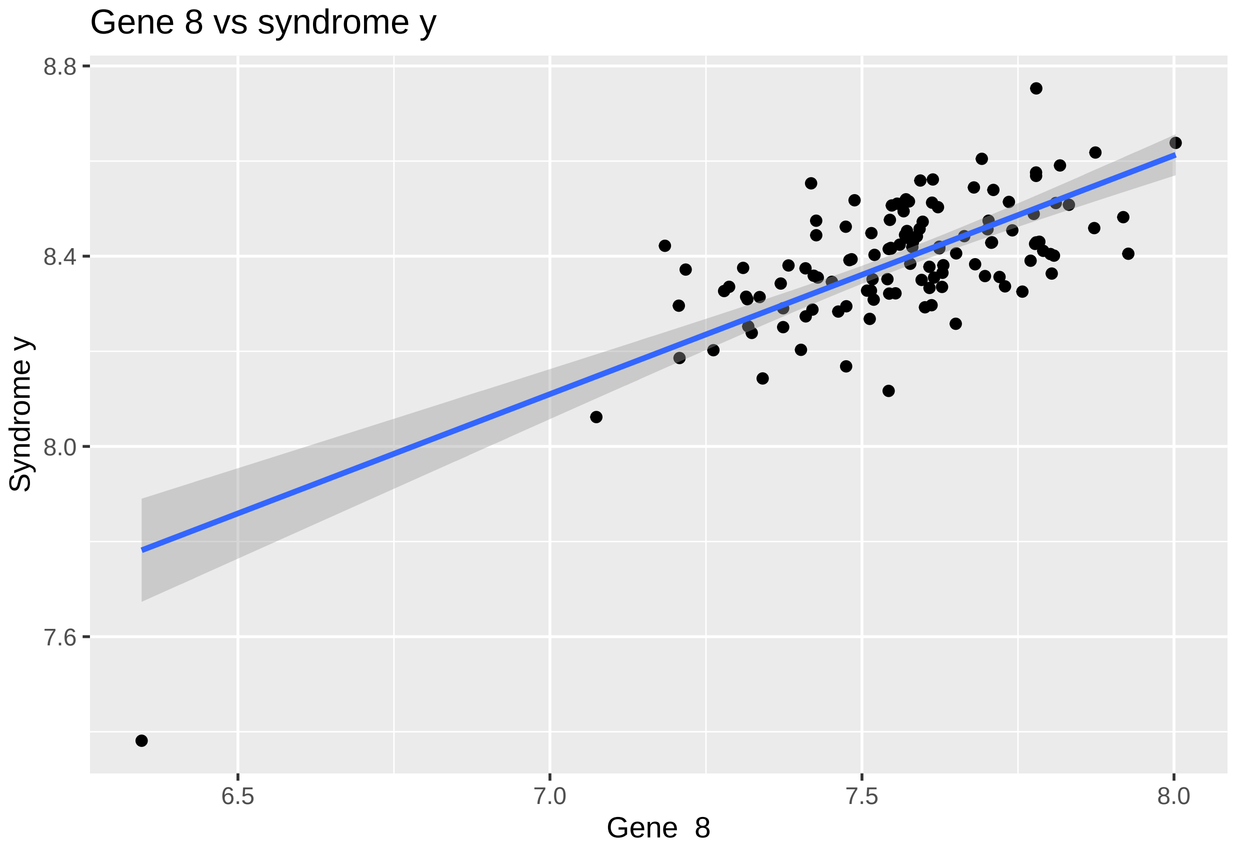

Gene 3 vs syndrome y
8.8
8.4
8.0
7.6
Syndrome y
3.5
4.0
4.5
5.0
Gene 3

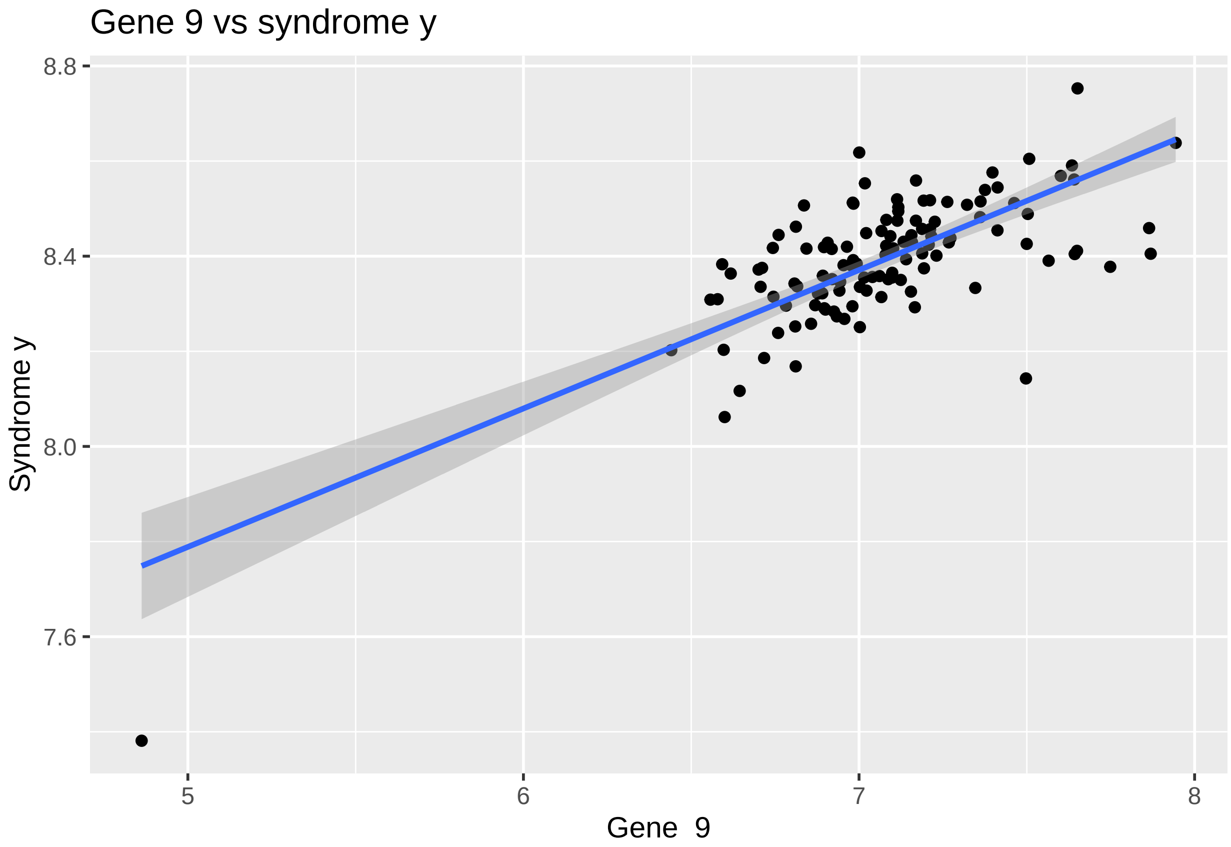

Gene 4 vs syndrome y
8.8
8.4
8.0
7.6
Syndrome y
6.6
6.9
7.2
Gene 4

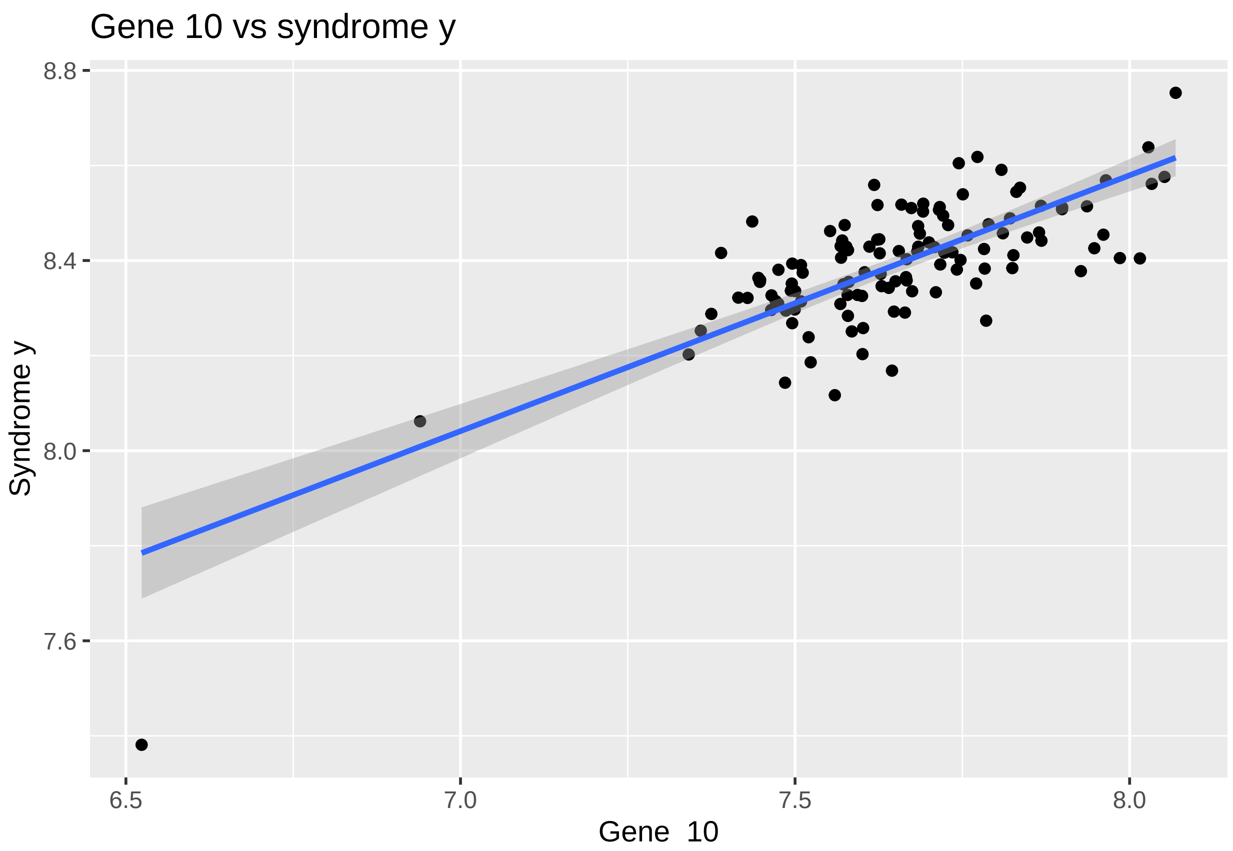

Gene 5 vs syndrome y
8.8
8.4
8.0
7.6
Syndrome y
4.0
4.5
5.0
5.5
Gene 5


Gene 6 vs syndrome y
8.8
8.4
8.0
7.6
Syndrome y
4.5
5.0
5.5
6.0
6.5
Gene 6


Gene 7 vs syndrome y
Syndrome y
8.8
8.4
8.0
7.6
Gene 7
6.0
6.5
7.0
7.5


Gene 8 vs syndrome y
8.8
8.4
8.0
7.6
Syndrome y
6.5
7.0
7.5
8.0
Gene 8


Gene 9 vs syndrome y
8.8
8.4
8.0
7.6
Syndrome y
5
6
7
8
Gene 9



```
p_values
```

```
[1] 7.266319e-01 4.885012e-01 5.167613e-01 9.828657e-01 1.681927e-19
[6] 5.297975e-20 1.767854e-20 4.825228e-21 7.919171e-24
```

Quick note that *p_values[i]* represents the p_value for the (i + 1) gene. For example, *p-values[1]* represents the p_value for g_2.

part c)

We can say that the gene with the smallest p-value will have the most significant association to the Bardet-Biedl syndrome y and that the largest p-value will have the least significant association. From part b), we can see that genes 6 to 10 have a very small p-value, and that gene 1 to 5 (gene 1 was observed in part a)) have a significant large p-value. We can also see from the graphs above that there are clear linear associations from genes 6 to 10, whereas the fitted lines from genes 1 to 5 seem to be horizontal.

Next, let's put all the p-values into a single vector in order to rank them.

```
p_values_all <- append(p_value, p_values, after = 1)
p_values_all
```

```
 [1] 3.160000e-01 7.266319e-01 4.885012e-01 5.167613e-01 9.828657e-01
 [6] 1.681927e-19 5.297975e-20 1.767854e-20 4.825228e-21 7.919171e-24
```

```
ranked_p_values <- order(p_values_all)
ranked_p_values
```

```
 [1] 10  9  8  7  6  1  3  4  2  5
```

The vector returned by *ranked_p_values* is the ranking in decreasing order of the significance of gene i in respect to y. The value returned by *ranked_p_values[i]* represents the specific gene. For example, *ranked_p_values[1]* = 10 implies that gene 10 has the most significant association with the symptom.

Question 4

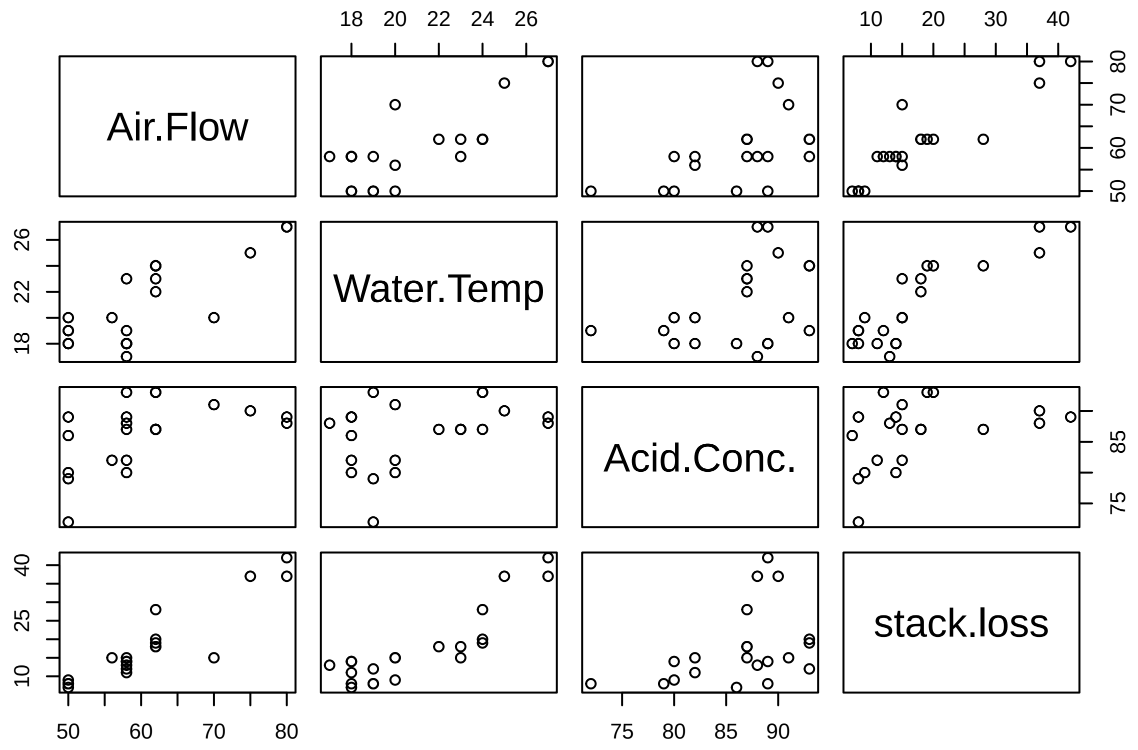
```
data(stackloss)
names(stackloss)
```

```
[1] "Air.Flow"   "Water.Temp" "Acid.Conc." "stack.loss"
```

```
# help(stackloss)
```

part a)

```
pairs(stackloss)
```



part b)

```
fit.stackloss <- lm(stack.loss ~ ., data = stackloss)
summary(fit.stackloss)
```

```

Call:
lm(formula = stack.loss ~ ., data = stackloss)

Residuals:
    Min      1Q  Median      3Q     Max 
-7.2377 -1.7117 -0.4551  2.3614  5.6978 
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -39.9197    11.8960  -3.356  0.00375 ** 
Air.Flow      0.7156     0.1349   5.307  5.8e-05 ***
Water.Temp    1.2953     0.3680   3.520  0.00263 ** 
Acid.Conc.   -0.1521     0.1563  -0.973  0.34405    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.243 on 17 degrees of freedom
Multiple R-squared:  0.9136,	Adjusted R-squared:  0.8983 
F-statistic:  59.9 on 3 and 17 DF,  p-value: 3.016e-09
```

From the summary, we can see that the multiple regression model can be described as

$$\begin{aligned} Y &= \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \epsilon \\ Y &= -39.9197 + 0.7156 X_1 + 1.2953 X_2 - 0.1521 X_3 + \epsilon \end{aligned}$$

Where X_1 represents the Air Flow, X_2 the Water Temperature and X_3 the Acid.

part c)

A 90% confidence interval for the coefficients of the linear regression model can be constructed as follows:

```
confint(fit.stackloss, level = 0.9)
```

```
                   5 %        95 %
(Intercept) -60.6140306 -19.2253183
Air.Flow      0.4810400   0.9502404
Water.Temp    0.6550686   1.9355036
Acid.Conc.   -0.4240127   0.1197676
```

part d)

We can use the pre-defined function *predict* from R

```
predict(fit.stackloss, newdata = data.frame(Air.Flow = 58, Water.Temp = 20,
    Acid.Conc. = 86), interval = "prediction", level = 0.99)
```

```
       fit      lwr      upr
1 14.41064 4.759959 24.06133
```

part e)

Let $\alpha = 0.10$ and defined the null hypothesis as

$$H_0 : \beta_3 = 0$$

Recall from the summary the coefficients of fit.stackloss

```
summary(fit.stackloss)$coef
```

```
               Estimate Std. Error    t value     Pr(>|t|)
(Intercept) -39.9196744 11.8959969 -3.3557234 3.750307e-03
Air.Flow      0.7156402  0.1348582  5.3066130 5.799025e-05
Water.Temp    1.2952861  0.3680243  3.5195672 2.630054e-03
Acid.Conc.   -0.1521225  0.1562940 -0.9733098 3.440461e-01
```

We can see that the p-value for Acid is 3.440461e-01 $> \alpha = 0.10$. This indicates that we **do not** reject the null hypothesis H_0. Thus, we cannot conclude that there is a linear association between the Acid and the Stackloss.