Case 2

Summary

Part 1

- Through stepwise regression, only `Company size` is a significant feature with a p-value of 0.000919 and a beta of 1.4651.
- This stepwise model has a good fit, as it has an adjusted R^2 of 0.7299 and a p-value of 0.0001541.
- Thsis stepwise model generates the following function

$$\text{Size of Purchase (\$1,000s)} = 128.7 + 1.4651 \times \text{Company Size (\$millions sales)} - 41.07 \times \text{Similar Products}$$

Part 2

- There seems to be no linear relationship between the dependent variable (`Average sales`) and independent variable (`Hours worked per week` and `Number of Customers`)
- Plotting `Average sales` against `Hours worked per week` is heteroscedatic, therefore ruling out the use of OLS
- However oddly, using OLS, we could get a (almost) good fit of p-value of 0.002616 and Adjusted R-squared of 0.9786 with the following equation (with S as the `Average sales`, C as the `Number of Customers`, H as the `Hours worked per week`). But this equation is kind of useless as the RESET test would show these parameters to be invalid.

$$\begin{aligned} S =& 4.804(10^3) \\ &+2.586(10^{-1})C - 4.999(10^{-6})C^3 \\ &-3.863(10^2)H + 1.028(10^1)H^2 - 9.065(10^{-2})H^3 \\ &+6.025(10^{-19})e^H \end{aligned}$$

Part 3

- There is a quadratic relationship between `Sales ($ million)` and `Number of Employees`.
- The quadratic model would follow the following model

$$\text{Average Sales} = -93.21 + 1.4554 \times \text{No. of employees} + -0.0040 \times \text{No. of employees}^2$$

- This quadratic model would be have an Adjusted R-squared of 0.7535 and p-value: 0.003084

Works

Importing various libraries

```
library(tidyverse)
library(caret)
library(leaps)
library(MASS)
```

Part 1

Import the data set and then generate `stepwise regression` model

```
data_part1 <- read.csv('csv/part1.csv', header = TRUE)
# Fit the full model
full.model_part1 <- lm(Size.of.Purchase...1.000s. ~ ., data=data_part1)
# Stepwise regression model
step.model_part1 <- stepAIC(full.model_part1, direction = "both", trace = FALSE)
summary(step.model_part1)
```

```
##
## Call:
## lm(formula = Size.of.Purchase...1.000s. ~ Company.Size...millions.sales. +
##     Similar.Products, data = data_part1)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -92.89 -64.97 -20.14  60.35 182.32
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    128.6713    62.7241   2.051 0.062718 .
## Company.Size...millions.sales.   1.4651     0.3356   4.366 0.000919 ***
## Similar.Products               -41.0732    19.7615  -2.078 0.059787 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 90.47 on 12 degrees of freedom
## Multiple R-squared:  0.7685, Adjusted R-squared:  0.7299
## F-statistic: 19.91 on 2 and 12 DF,  p-value: 0.0001541
```

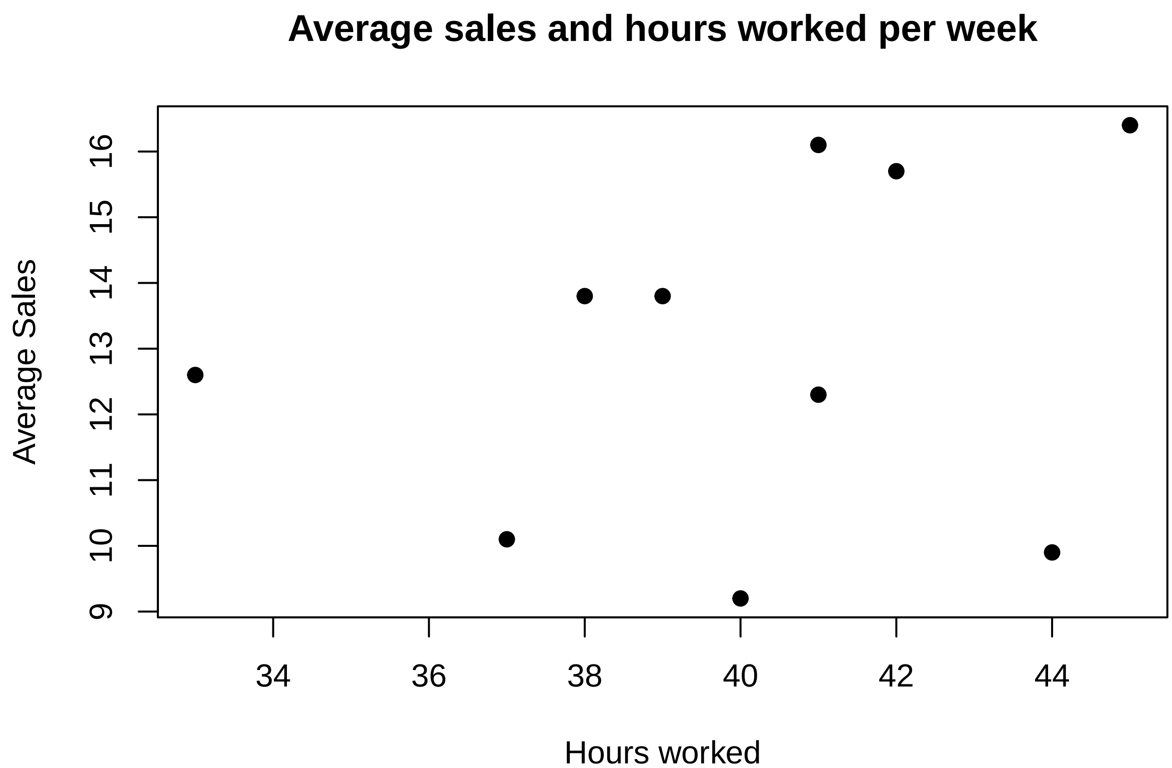
Part 2

Get the data and divide up the columns of data

```
data_part2 <- read.csv('csv/part2.csv', header = TRUE)
# split the data into various columns
avg_sales <- data_part2$Average.Sales....million.
hours_work <- data_part2$Hours.Worked.per.Week
no_customer <- data_part2$Number.of.Customers
```

Plot the `Average Sales ($ million)` against `Hours Worked per Week`

```
library(lmtest)
library(skedastic)
plot(hours_work, avg_sales,
     main = "Average sales and hours worked per week",
     ylab = "Average Sales",
     xlab = "Hours worked",
     pch=19)
```




```
sales_hour_model <- lm(data_part2$Average.Sales....million. ~
                       data_part2$Hours.Worked.per.Week)
```

We see that the data maybe **heteroscedatic** and we would use the Breusch–Pagan test and White Test to test whether they are actually heteroscedatic.

```
print(bptest(sales_hour_model,
      data_part2$Average.Sales....million.~data_part2$Hours.Worked.per.Week,
      data=data_part2))
```

```
##
##  studentized Breusch-Pagan test
```

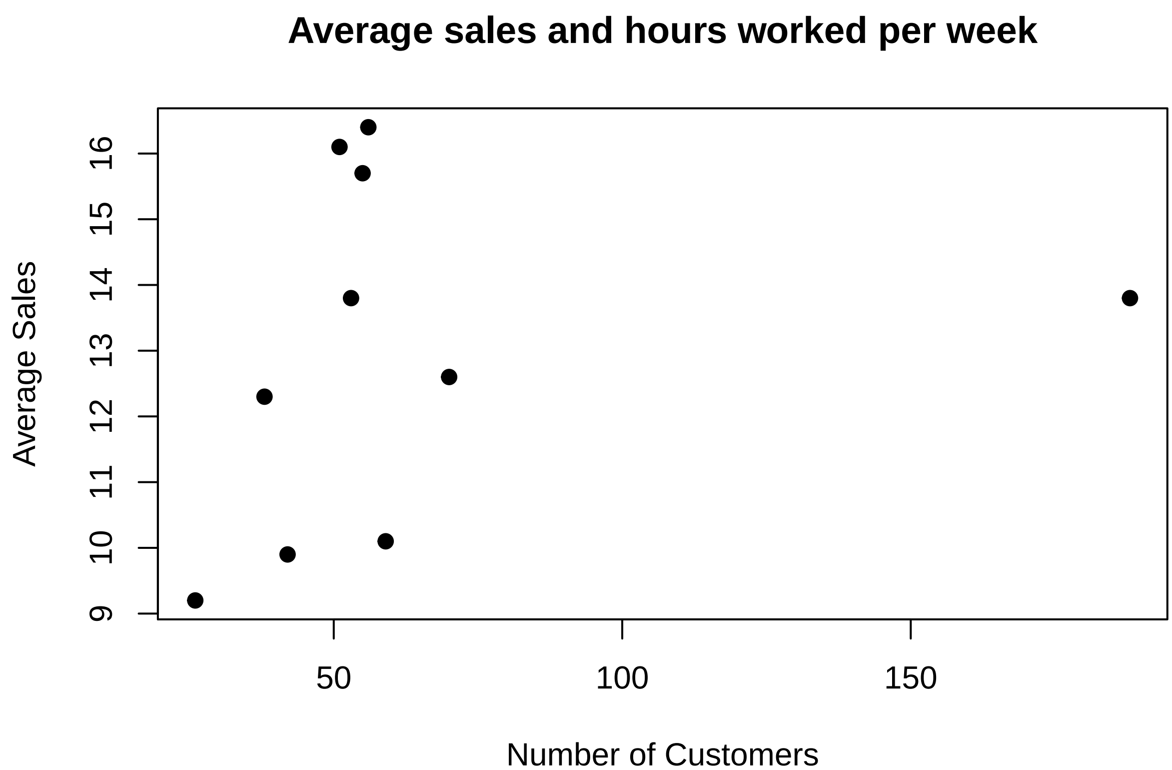
```
##
## data:  sales_hour_model
## BP = 2.4507, df = 1, p-value = 0.1175
```

```r
print(white_lm(sales_hour_model))
```

```
## # A tibble: 1 x 5
##   statistic p.value parameter method       alternative
##       <dbl>   <dbl>     <dbl> <chr>        <chr>
## 1      2.45   0.293         2 White's Test greater
```

From the two test, we see that the data is **heteroscedatic**. Therefore, any method of regression using OLS directly on this set of data would be unfavoured. Then we plot the `Average Sales ($ million)` against `Number of Customers`

```r
plot(no_customer, avg_sales,
     main = "Average sales and hours worked per week",
     ylab = "Average Sales",
     xlab = "Number of Customers",
     pch=19)
```



We see that the data has a possible outlier at (188, 13.8). There seems to be a weak relationship between the `Average Sales ($ million)` and `Number of Customers`. Then, we run a stepwise regression (however unwillingly, as it is unfavourable to run an OLS model with data that is heteroscedatic)

```r
full.model_part2 <- lm(avg_sales ~ poly(no_customer,3,raw=TRUE) + poly(hours_work,3,raw=TRUE)
                     + exp(hours_work))
step.model_part2 <- stepAIC(full.model_part2, direction = "both", trace = FALSE)
summary(step.model_part2)
```

```
##
## Call:
```

```
## lm(formula = avg_sales ~ poly(no_customer, 3, raw = TRUE) + poly(hours_work,
##     3, raw = TRUE) + exp(hours_work))
##
## Residuals:
##          1          2          3          4          5          6          7
## -0.0384251  0.4226218 -0.2495736  0.1277513 -0.3860165  0.0903110  0.1339930
##          8          9         10
## -0.1148592  0.0139471  0.0002503
##
## Coefficients:
##                                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)                         4.788e+03  4.899e+02   9.775  0.01030 *
## poly(no_customer, 3, raw = TRUE)1  2.356e-01  2.879e-01   0.818  0.49924
## poly(no_customer, 3, raw = TRUE)2  3.654e-04  4.544e-03   0.080  0.94323
## poly(no_customer, 3, raw = TRUE)3 -6.346e-06  1.676e-05  -0.379  0.74138
## poly(hours_work, 3, raw = TRUE)1  -3.850e+02  3.891e+01  -9.896  0.01006 *
## poly(hours_work, 3, raw = TRUE)2   1.025e+01  1.023e+00  10.018  0.00982 **
## poly(hours_work, 3, raw = TRUE)3  -9.038e-02  8.947e-03 -10.102  0.00966 **
## exp(hours_work)                     5.997e-19  7.512e-20   7.982  0.01533 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4729 on 2 degrees of freedom
## Multiple R-squared:  0.9929, Adjusted R-squared:  0.968
## F-statistic: 39.87 on 7 and 2 DF,  p-value: 0.02468
```

From this, we could see there are various insignificant variables, but oddly, it seems that the `poly()` function has caused the `stepAIC()` function unable to take away values such as Hours of work^2 in the process, therefore we tried to run it with various arrays generated individually.

```
no_customer_2 <- no_customer ^ 2
no_customer_3 <- no_customer ^ 3
hours_work_2 <- hours_work ^ 2
hours_work_3 <- hours_work ^ 3
exp_hours <- exp(hours_work)
full.model_part2 <- lm(avg_sales ~
                         no_customer + no_customer_2 + no_customer_3 +
                         hours_work + hours_work_2 + hours_work_3 +
                         exp_hours)
step.model_part2 <- stepAIC(full.model_part2, direction = "both", trace = FALSE)
summary(step.model_part2)
```

```
##
## Call:
## lm(formula = avg_sales ~ no_customer + no_customer_3 + hours_work +
##     hours_work_2 + hours_work_3 + exp_hours)
##
## Residuals:
##          1          2          3          4          5          6          7
## -3.660e-02  4.182e-01 -2.281e-01  1.170e-01 -4.081e-01  1.074e-01  1.272e-01
##          8          9         10
## -1.106e-01  1.347e-02  9.755e-05
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)    4.804e+03  3.669e+02   13.10 0.000962 ***
## no_customer    2.586e-01  1.684e-02   15.36 0.000599 ***
## no_customer_3 -4.999e-06  3.630e-07  -13.77 0.000828 ***
## hours_work    -3.863e+02  2.925e+01  -13.21 0.000938 ***
## hours_work_2   1.028e+01  7.734e-01   13.29 0.000920 ***
## hours_work_3  -9.065e-02  6.788e-03  -13.35 0.000908 ***
## exp_hours      6.025e-19  5.396e-20   11.17 0.001539 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3867 on 3 degrees of freedom
## Multiple R-squared:  0.9929, Adjusted R-squared:  0.9786
## F-statistic: 69.54 on 6 and 3 DF,  p-value: 0.002616
```

It worked! At least at the face of it, with an Adjusted R-squared of 0.9786 and p-value of 0.002616. Also, the model has none insignificant values, but still we would run a Ramsey RESET test and VIF

```r
library(car)
print(vif(step.model_part2))
```

```
##   no_customer no_customer_3    hours_work  hours_work_2  hours_work_3
##  3.502133e+01  3.354134e+01  6.291257e+05  2.709004e+06  7.360596e+05
##     exp_hours
##  2.195300e+01
```

All the variables with relation to hours of work, has a significantly higher value than the rest of the variables.

```r
print(resettest(step.model_part2))
```

```
##
##  RESET test
##
## data:  step.model_part2
## RESET = 0.5947, df1 = 2, df2 = 1, p-value = 0.6758
```

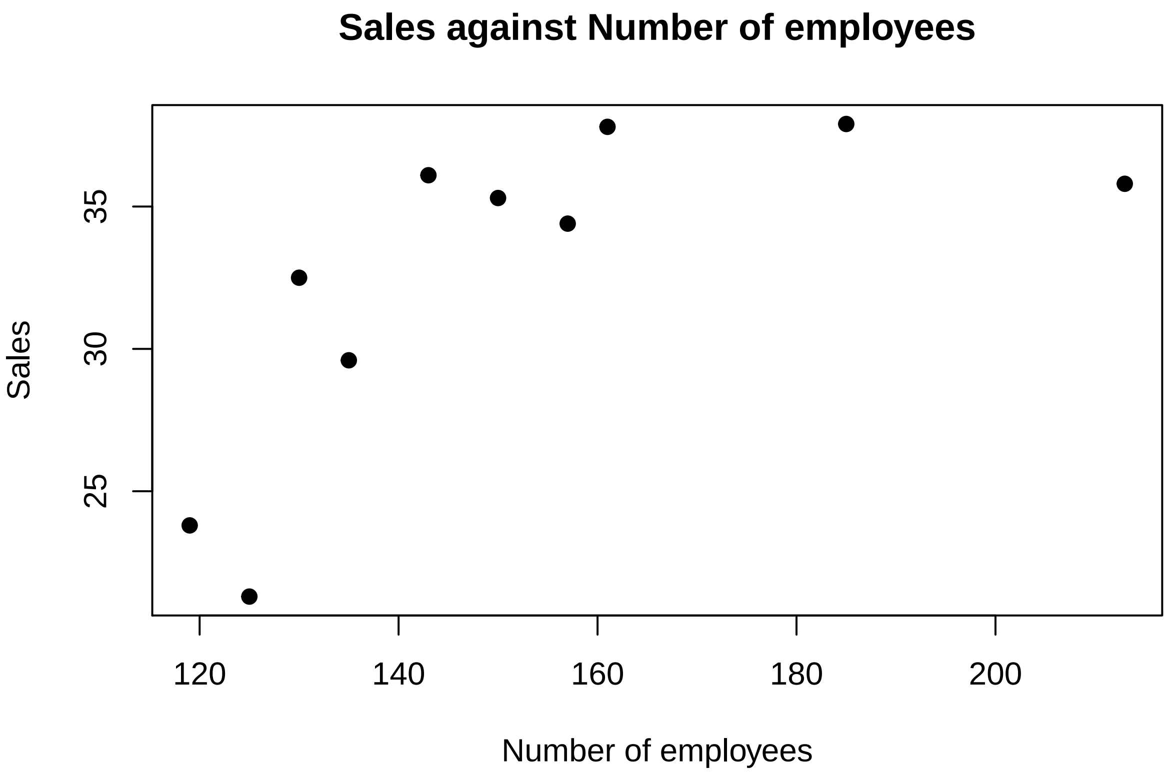
Our worst fears. The RESET test p-value would indicate that the model has misspecified in the sense that the data generating process might be better approximated by a polynomial or another non-linear functional form. To this point, I am just confused, maybe I wrote the wrong code?

Part 3

Get the data and plot the scatter plot

```
data_part3 <- read.csv('csv/part3.csv', header = TRUE)

plot(data_part3$Number.of.Employees,
     data_part3$Sales.............million.,
     main = "Sales against Number of employees",
     ylab = "Sales",
     xlab = "Number of employees", pch=19)
```



There seems to be a non-linear relationship between `Sales ($ million)` and `Number of Employees`. Therefore we try to do a non-linear regression for it. This looks like a quadratic relationship.

```
sales <- data_part3$Sales.............million.
no_employees <- data_part3$Number.of.Employees


full.model_part3 <- lm(sales ~ poly(no_employees,2,raw=TRUE))
summary(full.model_part3)
```

```
##
## Call:
## lm(formula = sales ~ poly(no_employees, 2, raw = TRUE))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.8762 -1.0446  0.3484  0.5760  4.1501
##
## Coefficients:
```

```
##                                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)                       -93.211581  30.250946  -3.081  0.01778 *
## poly(no_employees, 2, raw = TRUE)1   1.455446   0.376271   3.868  0.00615 **
## poly(no_employees, 2, raw = TRUE)2  -0.004003   0.001138  -3.518  0.00975 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.876 on 7 degrees of freedom
## Multiple R-squared:  0.8083, Adjusted R-squared:  0.7535
## F-statistic: 14.76 on 2 and 7 DF,  p-value: 0.003084
```

We would also draw out the final graph for it.

```
#generate range of 50 numbers starting from 30 and ending at 160
xx <- seq(110,220, length=50)
plot(no_employees, sales, pch=19,
     xlab="Number of Employees",
     ylab="Sales",
     ylim=c(20,40))
lines(xx, predict(full.model_part3, data.frame(no_employees=xx)), col="red")
```