SS2864B Assignment 2

Ali Al-Musawi

01/02/2020

Question 1

In this question, we create a vector and a dataframe. First, create a vector that holds the elements $-20^2, -19^2, ..., 19^2, 20^2$. Next, create a dataframe that holds the table of values of $y = x^2$. Then, save the vector to a file named "squares"and the dataframe to a file named "quadratic". Finally, read in the saved objects and check if they are equvalent to the previously created objects.

```
vector <- I((-20:20)^2)
df <- data.frame(-20:20, vector)
names(df) <- c("x", "x^2")
write(vector, file = "squares")
write.table(df, "quadratic")
vector1 <- scan("squares")
df1 <- read.table("quadratic")
vector1 == vector && df == df1
```

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

Running this code, we get "TRUE", which means they are equivalent.

Question 2

In this question, we find out for which elements of the sequences $a_n = \{2^1, 2^2, ..., 2^{15}\}$ and $b_n = \{1^3, 2^3, ..., 15^3\}$ we have that $a_i > b_i$.

```
powers2 <- 2^(1:15)
cubics <- (1:15)^3
n <- length(powers2)
(1:n)[powers2 > cubics]
```

```
## [1]  1 10 11 12 13 14 15
```

Running this code, we find out the $a_i > b_i, \forall i \in \{1, 10, 11, 12, 13, 14, 15\}$.

Question 3

In this question, we examine the differeces of the R functions **save** and **dump**.

```
x <- 1:10
y <- as.factor(rep(c("M", "F"), 5))
z <- data.frame(x, y)
dump("x", "x.R")
dump("y", "y.R")
dump("z", "z.R")
save("x", file= "x.RData")
save("y", file= "y.RData")
save("z", file= "z.RData")
xx <- x
yy <- y
zz <- z
rm(list = c("x", "y", "z"))
source("x.R")
source("y.R")
source("z.R")
(x == xx) && (y == yy) && (z == zz)
```

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

```
rm(list = c("x", "y", "z"))
load("x.RData")
load("y.RData")
load("z.RData")
(x == xx) && (y == yy) && (z == zz)
```

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

Running this code and examining its results, we conclude with the following: both functions allow us to save objects created here. When importing the saved objects, there is no need to assign the saved objects to a new variable as they keep the original name, and we note that they are equivalent to the original objects. The function **dump** saves R objects in text (script) format, and to retrieve the object, a call to **source** is needed. On the other hand, the function **save** saves R objects in binary format, and to retrieve the object, a call to the function **load** is required. By convention, R binary objects are saved into **.RData** extension files, whereas R scripts are saved into **.R** extension files.

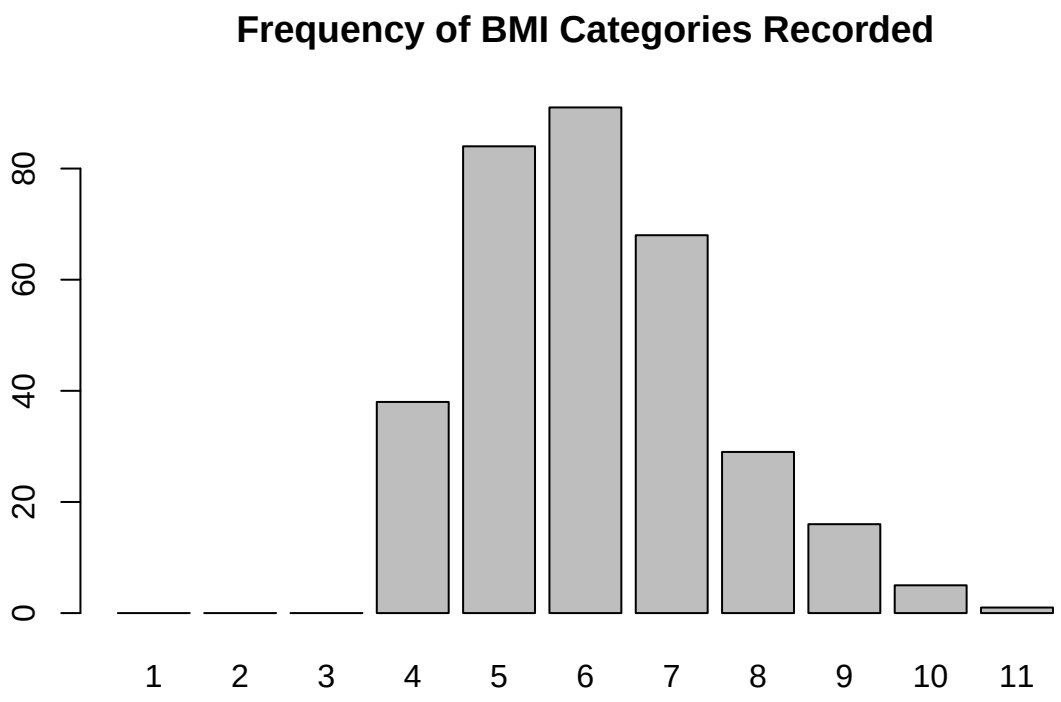
Question 4

In this question, we categorize BMI data according to the table in this link: Wikipedia: BMI. We first categorize the data into 11 intervals, find out about the frequency of records belonging to each interval, then make a barplot of our findings.

```
library(MASS)
data(Pima.te)
fac1 <- cut(x = Pima.te$bmi, breaks = c(0, 15, 16, 18.5, 25, 30, 35, 40, 45, 50, 60, Inf))
levels(fac1)[1] <- "Very severely underweight"
levels(fac1)[2] <- "Severely underweight"
levels(fac1)[3] <- "Underweight"
levels(fac1)[4] <- "Normal (healthy weight)"
levels(fac1)[5] <- "Overweight"
levels(fac1)[6] <- "Obese Class I (Moderately obese)"
levels(fac1)[7] <- "Obese Class II (Severely obese)"
levels(fac1)[8] <- "Obese Class III (Very severely obese)"
levels(fac1)[9] <- "Obese Class IV (Morbidly obese)"
levels(fac1)[10] <- "Obese Class V (Super obese)"
levels(fac1)[11] <- "Obese Class VI (Hyper obese)"
table(fac1)
```

```
## fac1
##             Very severely underweight
##                                     0
##                  Severely underweight
##                                     0
##                            Underweight
##                                     0
##               Normal (healthy weight)
##                                    38
##                             Overweight
##                                    84
##      Obese Class I (Moderately obese)
##                                    91
##       Obese Class II (Severely obese)
##                                    68
## Obese Class III (Very severely obese)
##                                    29
##       Obese Class IV (Morbidly obese)
##                                    16
##           Obese Class V (Super obese)
##                                     5
##          Obese Class VI (Hyper obese)
##                                     1
```

```
levels(fac1) <- 1:11
plot(fac1, main = "Frequency of BMI Categories Recorded")
```

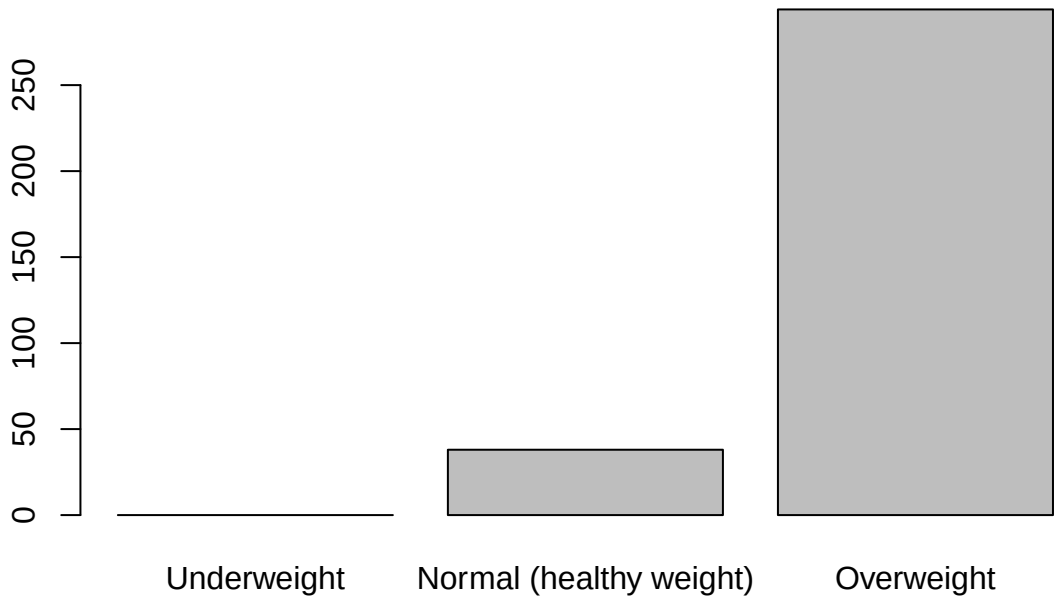


Next, we repeat the same process, but instead we restrict it to 3 intervals.

```
fac2 <- cut(x = Pima.te$bmi, breaks = c(0, 18.5, 25, Inf))
levels(fac2)[1] <- "Underweight"
levels(fac2)[2] <- "Normal (healthy weight)"
levels(fac2)[3] <- "Overweight"
table(fac2)
```

```
## fac2
##             Underweight Normal (healthy weight)              Overweight
##                       0                      38                     294
```

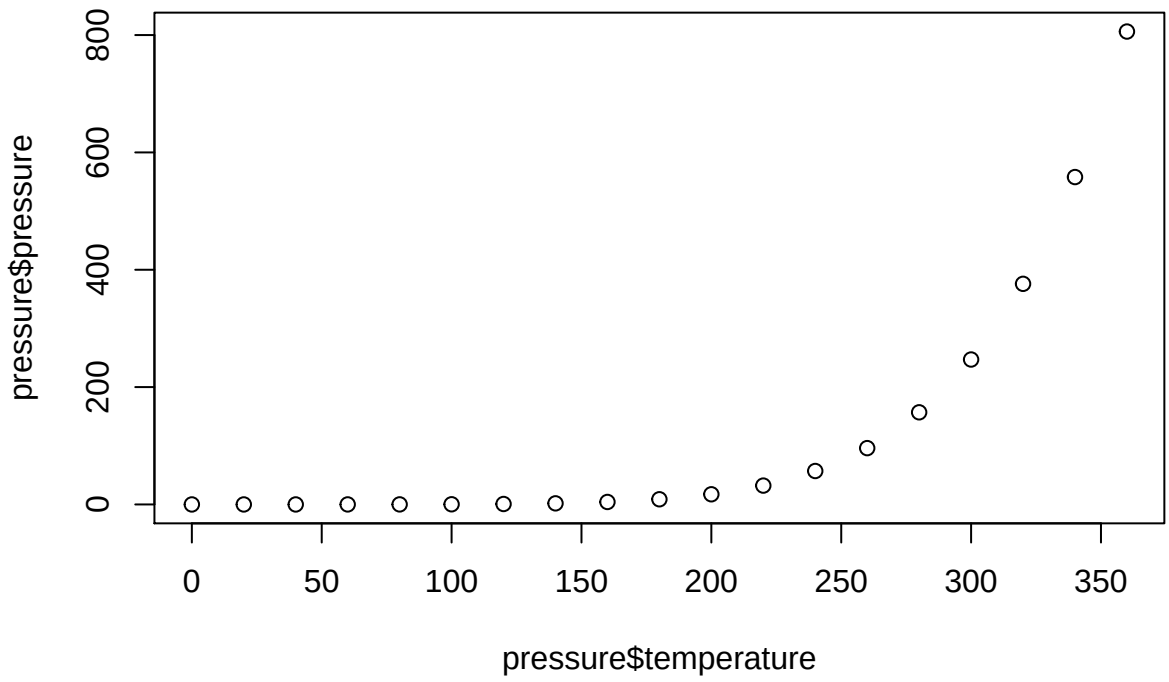
```
plot(fac2, main = "Frequency of BMI Categories Recorded")
```


Frequency of BMI Categories Recorded
0
50
100
150
200
250
Underweight
Normal (healthy weight)
Overweight

Question 5

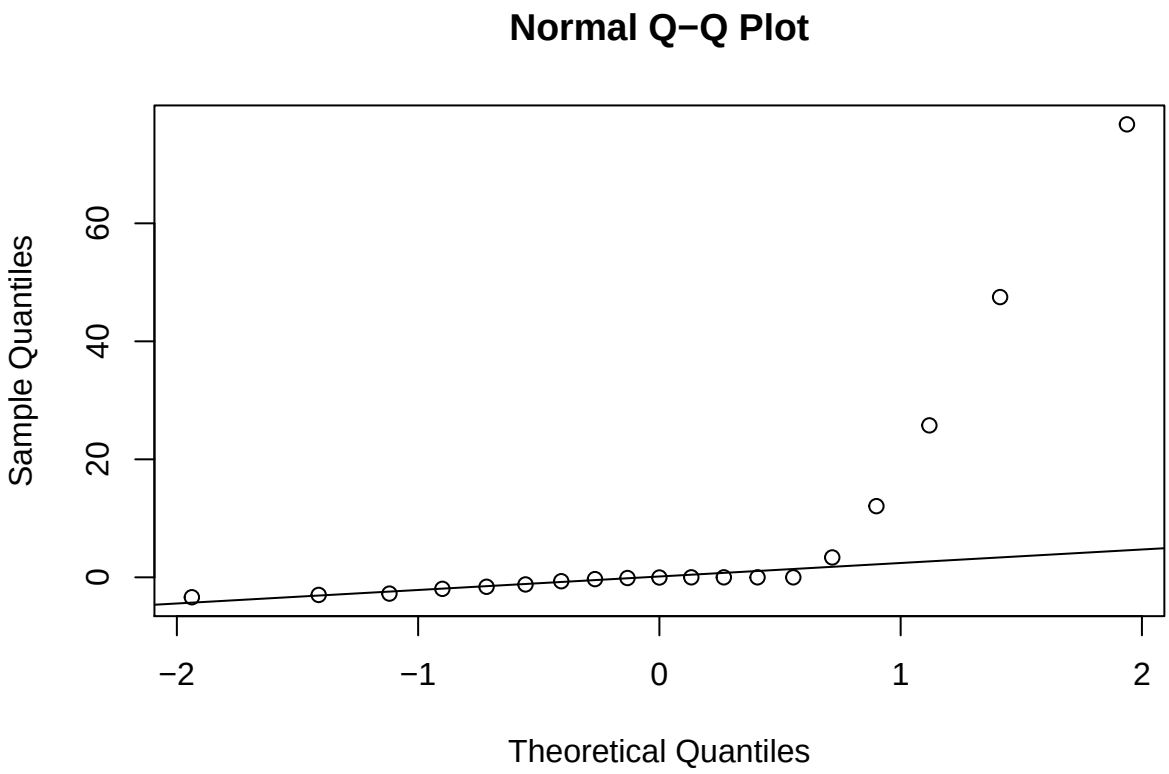
In this question, we examine the relationship between *temperature* and *pressure* in the **pressure** dataset. First, we produce a scatterplot of the variables.

```
data("pressure")
plot(x = pressure$temperature, y = pressure$pressure)
```



Based on this graph, we conclude the relationship between the two variables is non-linear. Next, we assume that the relationship between *pressure* P and *temperature* T is given by $P = (0.168 + 0.007T)^{\frac{20}{3}}$. We compute the residuals of this function and the observed values, and produce a normal QQ plot of the residuals.

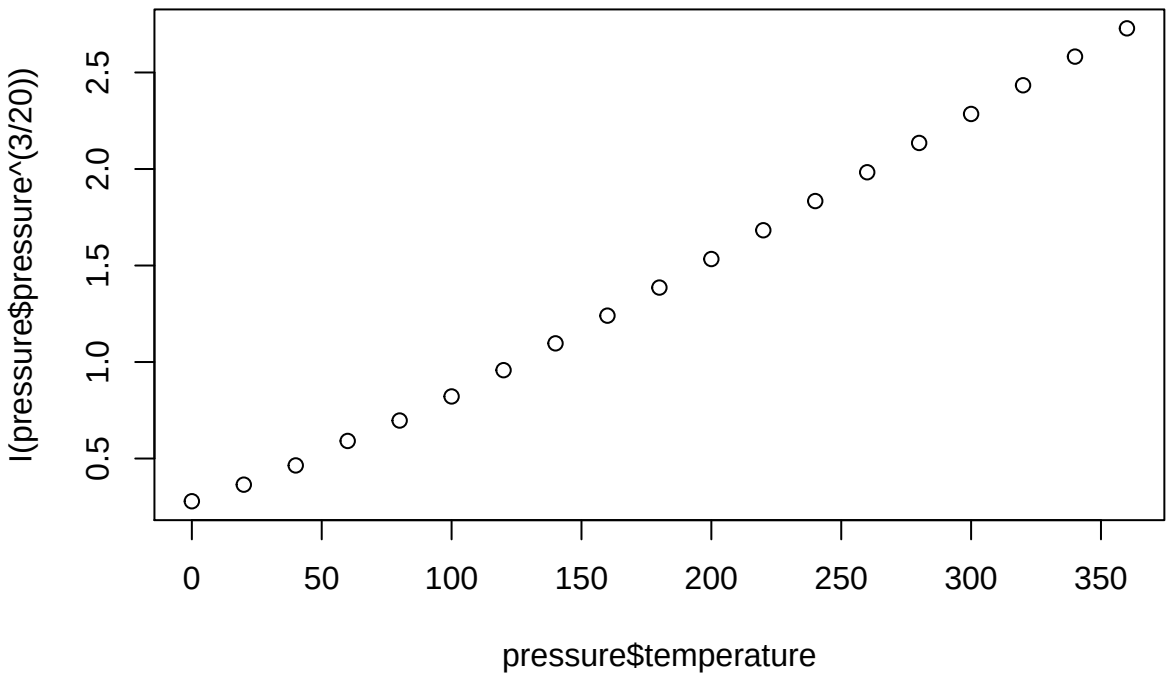
```
residuals <- with(pressure, pressure-(0.168+0.007*temperature)^(20/3))
qqnorm(residuals)
qqline(residuals)
```



Based on the plot, we see that the residuals are skewed and not normally distributed.

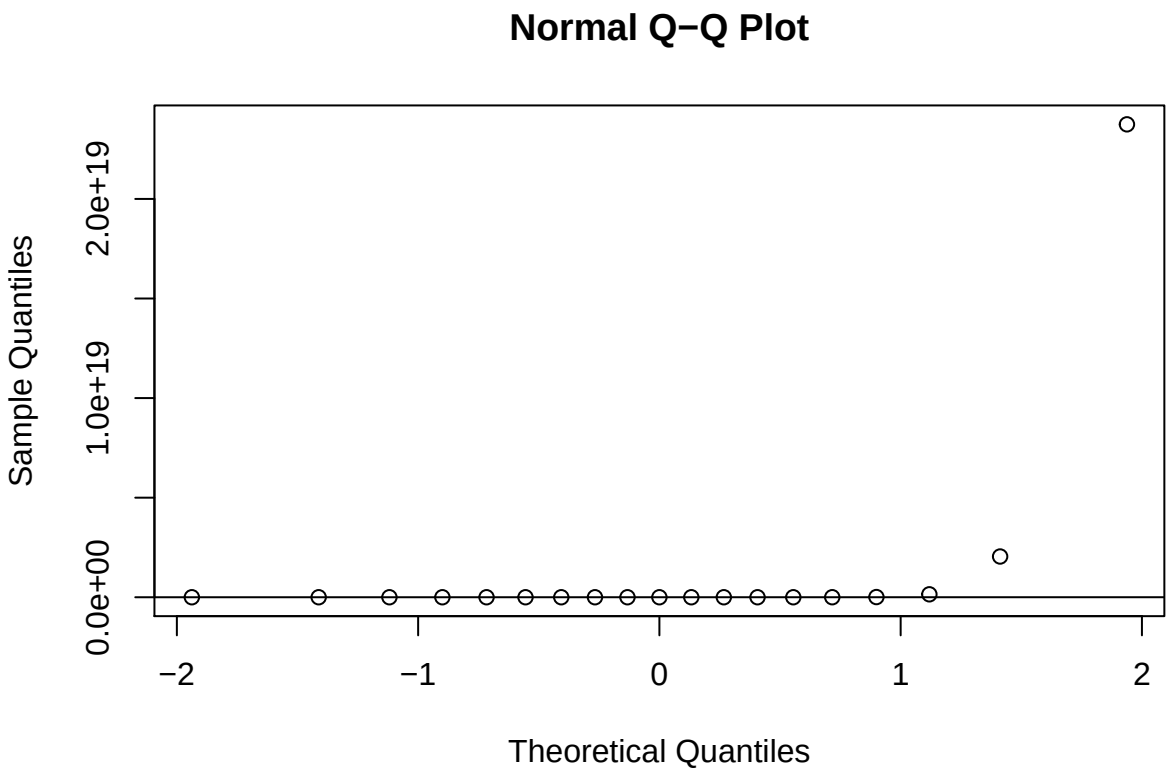
Now, we repeat the process above, but instead we transform P to $P^{\frac{3}{20}}$. The following is a scatterplot of the data.

```
plot(x = pressure$temperature, y = I(pressure$pressure^(3/20)))
```



The relationship here appears to be linear. Now, we compute residuals associated with the transformed pressure to examine their distribution:

```
residuals <- with(pressure, pressure^(20/3)-(0.168+0.007*temperature)^(20/3))
qqnorm(residuals)
qqline(residuals)
```

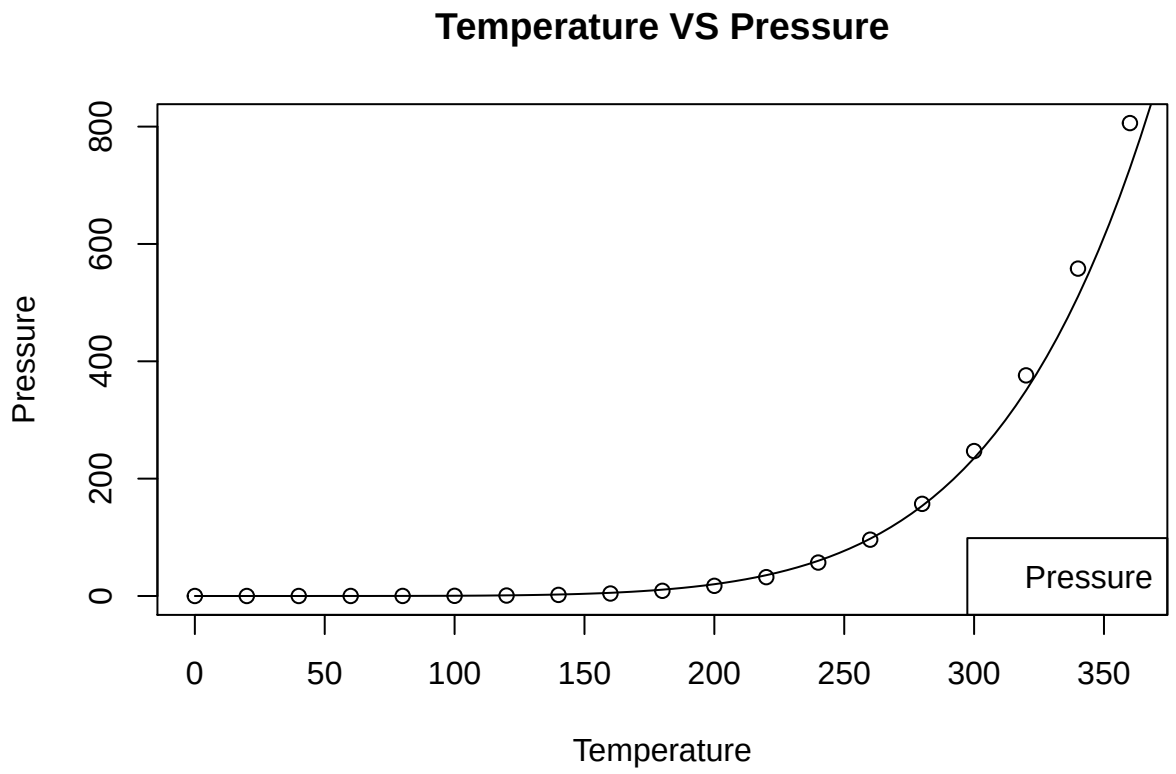


Based on this graph, we see that the residuals are slightly skewed.

Question 6

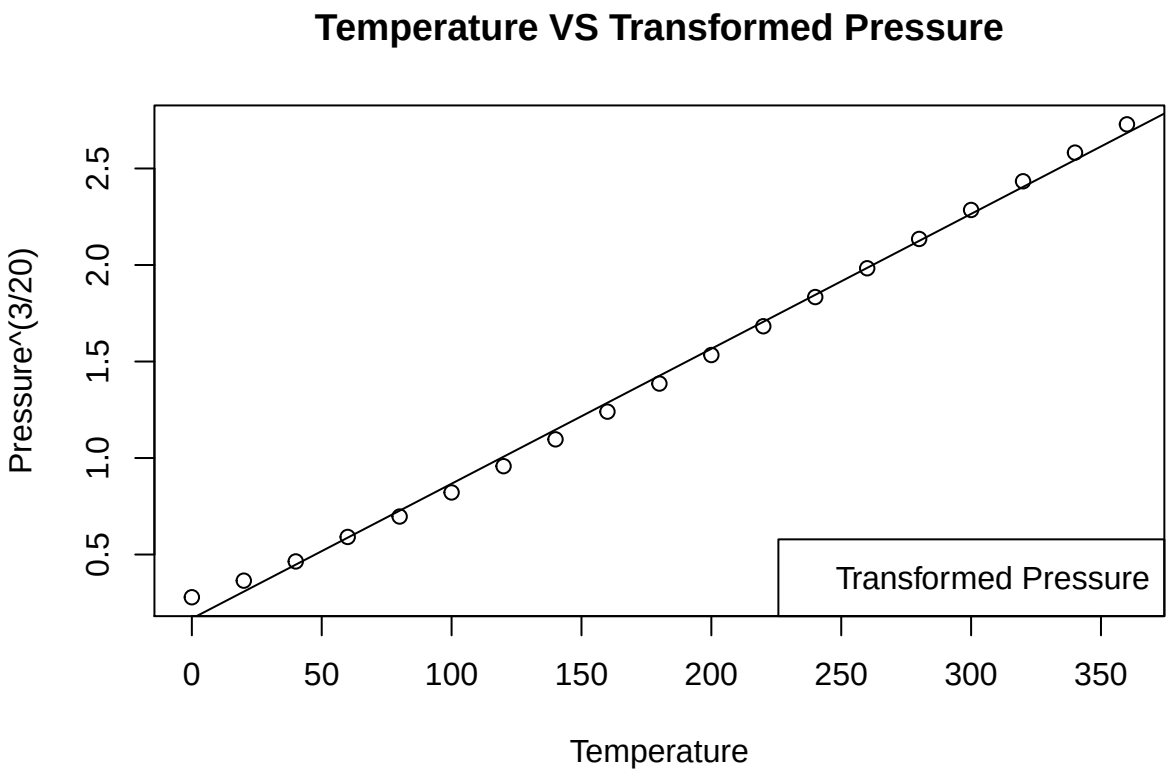
In this question, we continue working with the previous dataset. We fit a curve to P as a function of T:

```
plot(x = pressure$temperature, y = pressure$pressure,
     xlab = "Temperature", ylab = "Pressure",
     main = "Temperature VS Pressure")
legend(x = "bottomright", legend = c("Pressure"), xpd = T)
curve((0.168+0.007*x)^(20/3), from = 0, to = 400, add = TRUE)
```



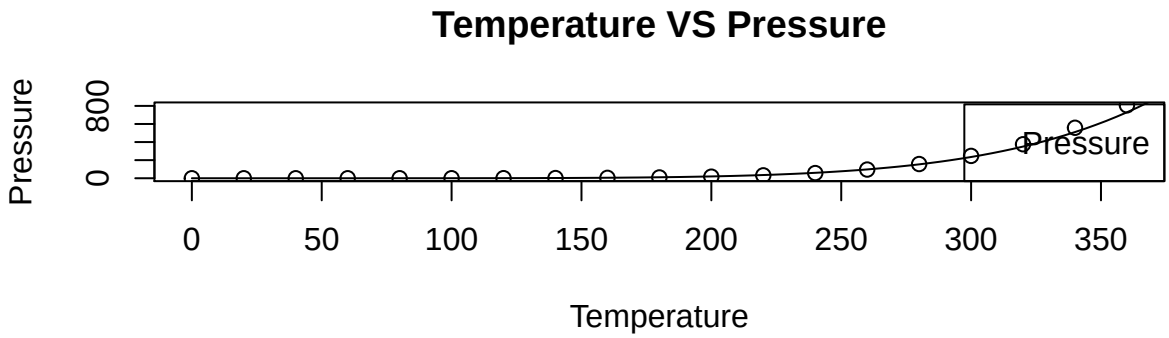
Next, we fit a line to the transformed pressure $P^{\frac{3}{20}}$ as a function of T:

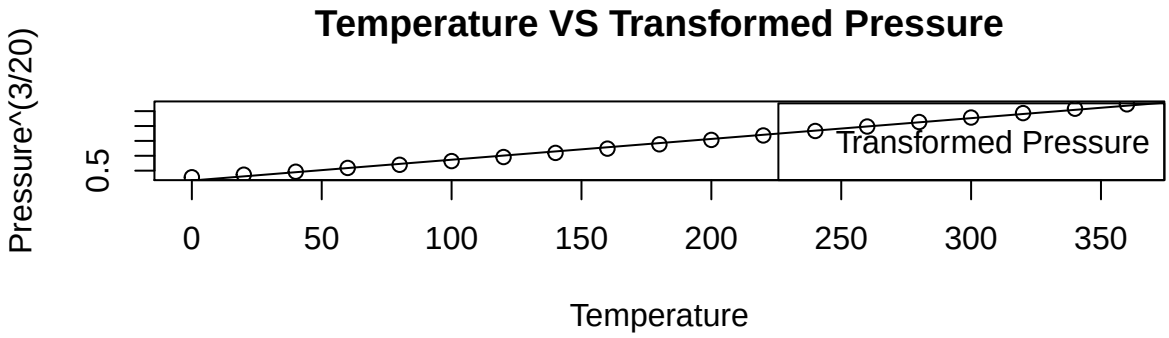
```
plot(x = pressure$temperature, y = I(pressure$pressure^(3/20)),
     xlab = "Temperature", ylab = "Pressure^(3/20)",
      main = "Temperature VS Transformed Pressure")
legend(x = "bottomright", legend = c("Transformed Pressure"), xpd = T)
fit <- lm(I(pressure^(3/20))~temperature, data = pressure)
abline(fit)
```



Now, we produce a 2×1 plot of the two fits above:

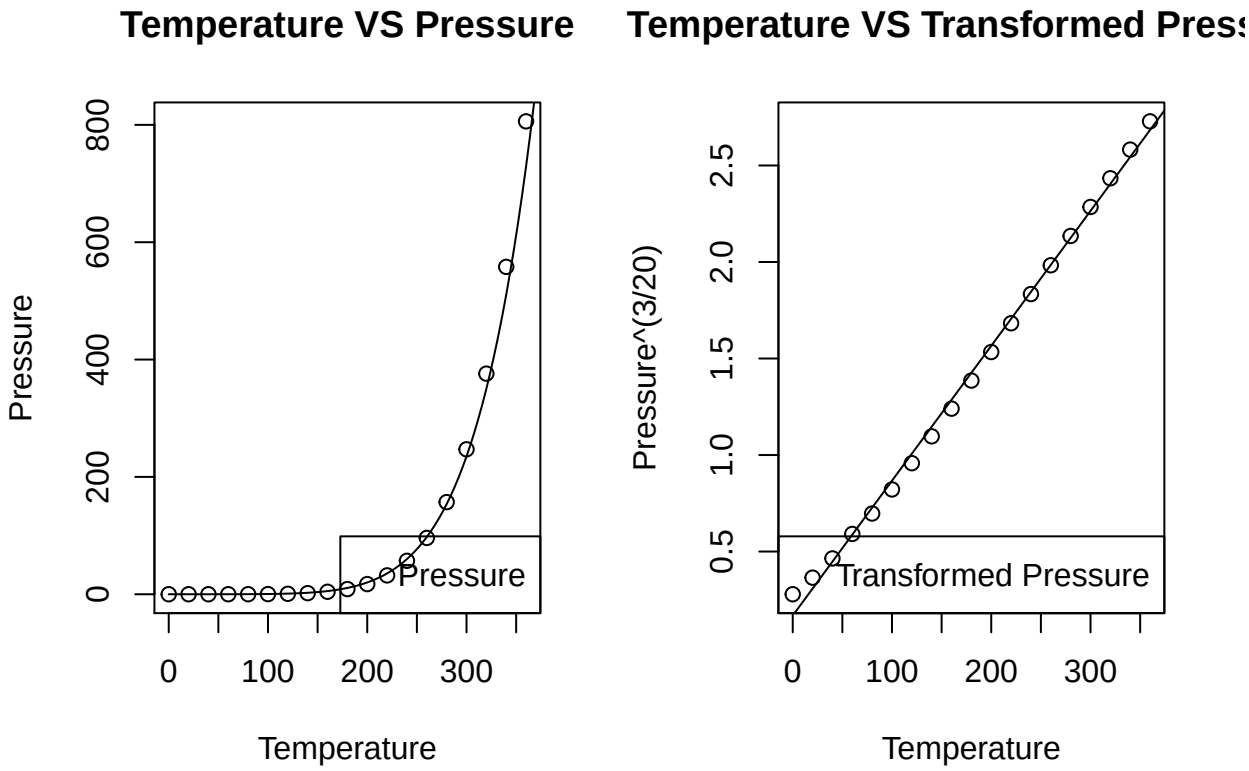
```
par(mfrow=c(2,1))
plot(x = pressure$temperature, y = pressure$pressure,
     xlab = "Temperature", ylab = "Pressure",
     main = "Temperature VS Pressure")
curve((0.168+0.007*x)^(20/3), from = 0, to = 400, add = TRUE)
legend(x = "bottomright", legend = c("Pressure"), xpd = T)
plot(x = pressure$temperature, y = I(pressure$pressure^(3/20)),
     xlab = "Temperature", ylab = "Pressure^(3/20)",
     main = "Temperature VS Transformed Pressure")
abline(fit)
legend(x = "bottomright", legend = c("Transformed Pressure"), xpd = T)
```





Finally, we produce a 1×2 plot of the two fits above:

```
par(mfrow=c(1,2))
plot(x = pressure$temperature, y = pressure$pressure,
     xlab = "Temperature", ylab = "Pressure",
     main = "Temperature VS Pressure")
curve((0.168+0.007*x)^(20/3), from = 0, to = 400, add = TRUE)
legend(x = "bottomright", legend = c("Pressure"), xpd = T)
plot(x = pressure$temperature, y = I(pressure$pressure^(3/20)),
     xlab = "Temperature", ylab = "Pressure^(3/20)",
     main = "Temperature VS Transformed Pressure")
abline(fit)
legend(x = "bottomright", legend = c("Transformed Pressure"), xpd = T)
```



Please note that the points run through the legend, but this problem is due to knitting with R markdown. If the code is used in an R script, this problem does not occur. Please review the accompanying R script for the source code.