Session-2, Linear Regression: Examples and Exercises

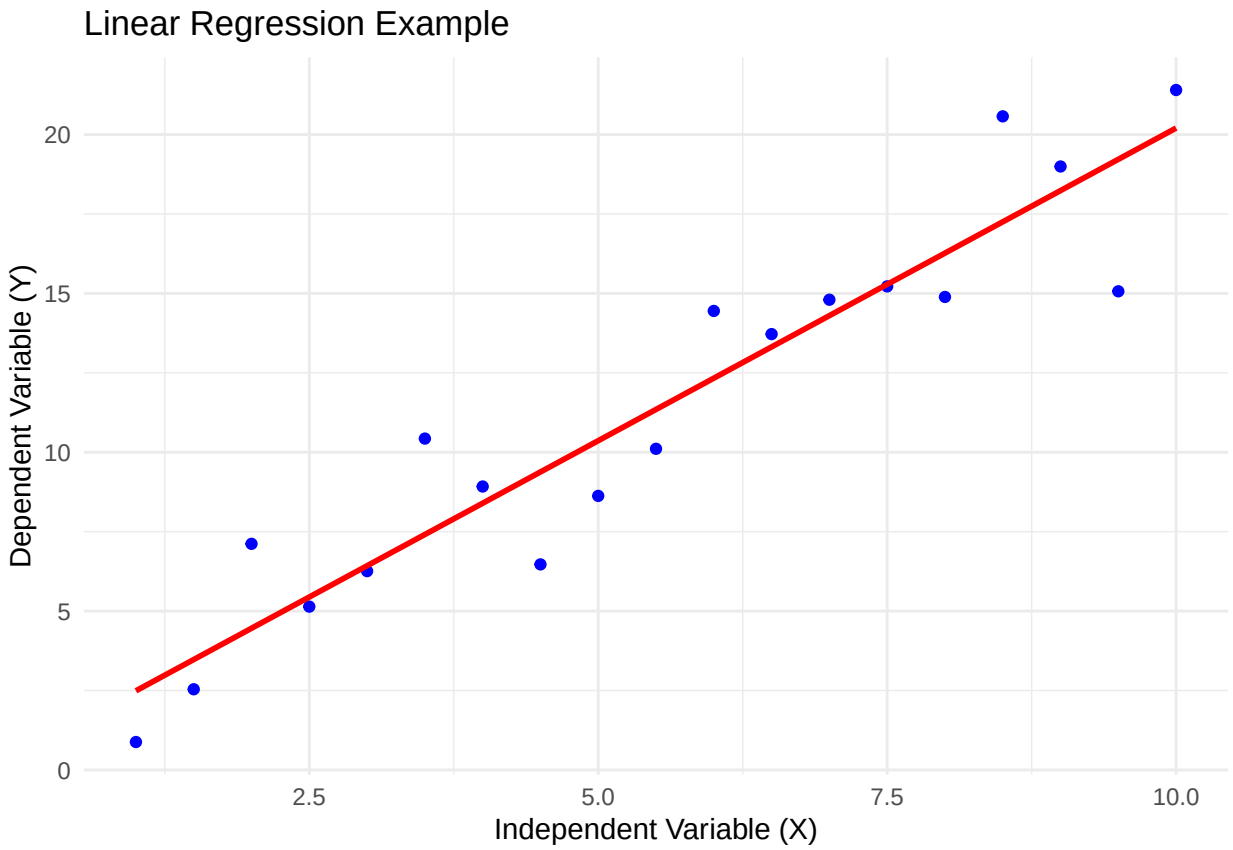
Ali Jaddoa, BSc, MSc, PhD Computer Science, email: Ali.Jaddoa@Canterbury.ac.uk

08/09/2023

1. Linear Regression: Introduction

Linear Regression example

```r
# Load required library
library(ggplot2)

set.seed(123) # Setting a seed for reproducibility
X <- seq(1, 10, 0.5) # Independent variable (X) with values from 1 to 10
Y <- 2 * X + rnorm(length(X), mean = 0, sd = 2) # Dependent variable (Y) with noise

data <- data.frame(X, Y)

plot <- ggplot(data, aes(x = X, y = Y)) +
        geom_point(color = "blue") +
        geom_smooth(method = "lm", se = FALSE, color = "red") +
        labs(title = "Linear Regression Example",
             x = "Independent Variable (X)",
             y = "Dependent Variable (Y)") +
        theme_minimal()

# Display the plot
print(plot)
```

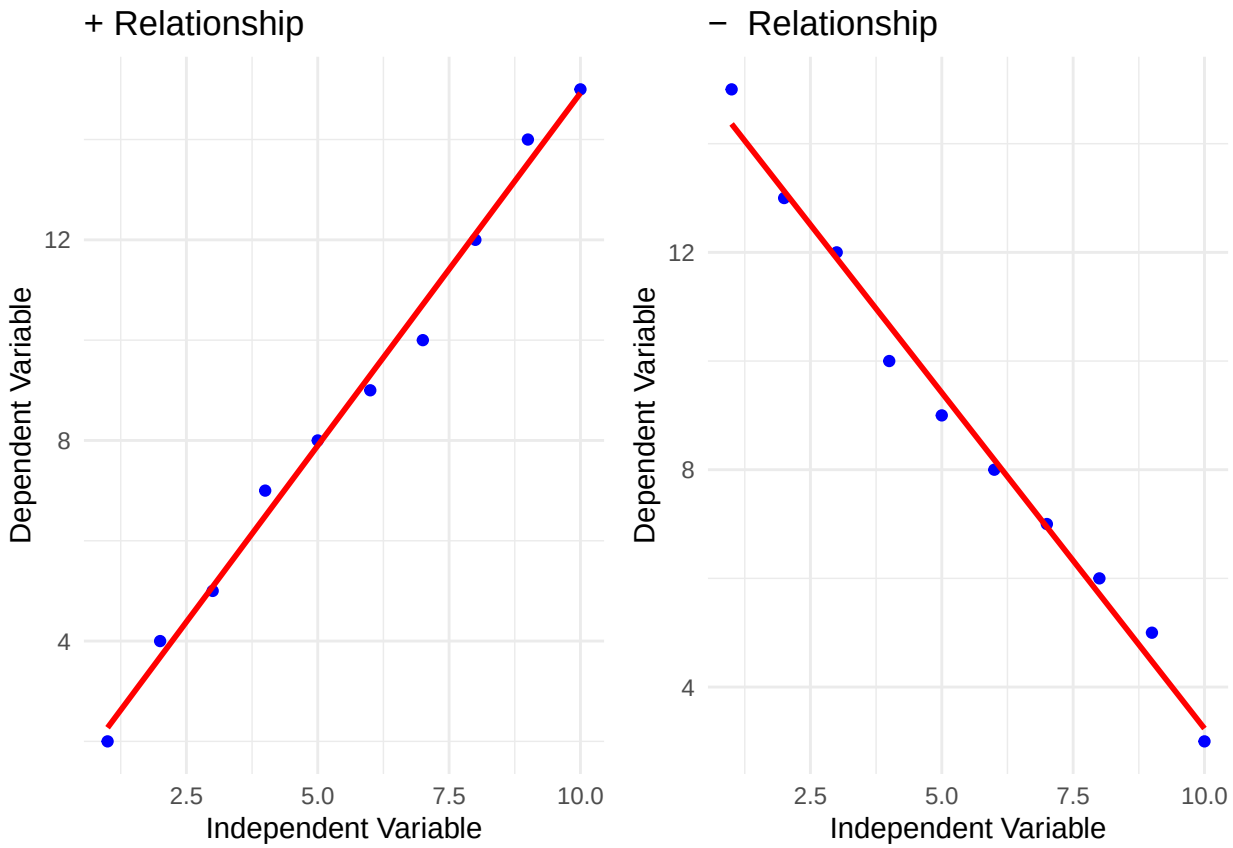


The goal is to find the best-fitting line that minimizes the overall distance between the line and the data points.

- The relationship can be
  - Positive (positive correlation between X and Y) or
  - Negative (negative correlation between X and Y)

```r
# Step 1: Install and load necessary packages
library(ggplot2)

# Step 2: Create or import your datasets for positive and negative relationships
# Replace 'independent_var_pos'/'independent_var_neg' and 'dependent_var_pos'/'dependent_var_neg'
#with your actual variable names.
independent_var_pos <- c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
dependent_var_pos <- c(2, 4, 5, 7, 8, 9, 10, 12, 14, 15)

independent_var_neg <- c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
dependent_var_neg <- c(15, 13, 12, 10, 9, 8, 7, 6, 5, 3)

# Step 3: Perform linear regression for positive and negative relationships
lm_model_pos <- lm(dependent_var_pos ~ independent_var_pos)
lm_model_neg <- lm(dependent_var_neg ~ independent_var_neg)

# Step 4: Create the plots for positive and negative relationships
# Plot for positive relationship
plot_pos <- ggplot(data = data.frame(x = independent_var_pos, y = dependent_var_pos), aes(x = x, y = y)
```

```
  geom_point(color = "blue") +                    # Plot the data points
  geom_smooth(method = "lm", se = FALSE, color = "red") +  # Add the best fit line
  labs(title = "+ Relationship",
       x = "Independent Variable",
       y = "Dependent Variable") +
  theme_minimal()

# Plot for negative relationship
plot_neg <- ggplot(data = data.frame(x = independent_var_neg, y = dependent_var_neg), aes(x = x, y = y))
  geom_point(color = "blue") +                    # Plot the data points
  geom_smooth(method = "lm", se = FALSE, color = "red") +  # Add the best fit line
  labs(title = "-  Relationship",
       x = "Independent Variable",
       y = "Dependent Variable") +
  theme_minimal()

# Display both plots side by side
library(gridExtra)
grid.arrange(plot_pos, plot_neg, ncol = 2)
```



2. Simple linear regression – model building

- Simple linear regression estimates how much Y will change when X changes by a certain amount. With the correlation coefficient, the variables X and Y are inter- changeable. With regression, we are trying to predict the Y variable from X using a linear relationship (i.e., a line):

$$Y = \beta_0 + \beta_1 X + \epsilon$$

β_0	intercept (coefficient)	The predicted value of Y when $X = 0$.
β_1	Slope (coefficient)	How much we expect Y to change as X increases
Y	Target	Response or dependent variable
X	Feature victor	Predictor or independent variable
ϵ	Error term	Accounts for the variability/noise not explained by the regression model.

- The simple linear regression for individual observation looks like this:

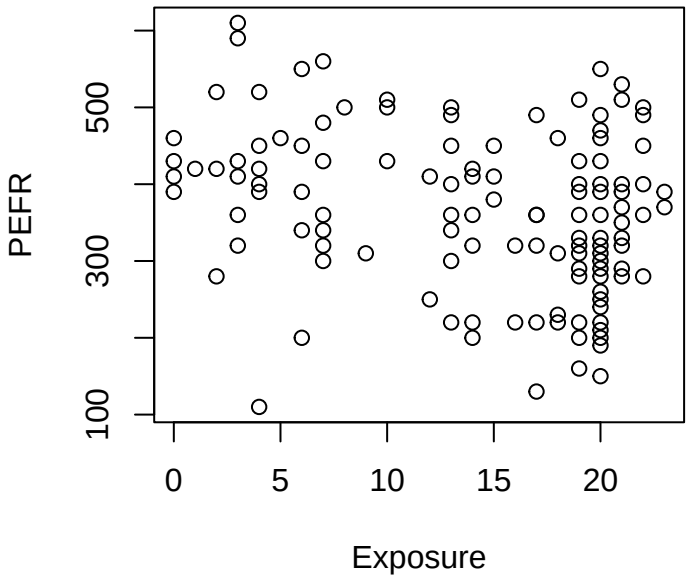
$$Y_i = \beta_0 + \beta_1 X_i, i = 1, .., n$$

2.1 Example-1

A Dataset "LungDisease.csv" consist of number of years a worker was exposed to cotton dust (Exposure) and a measure of lung capacity (PEFR or "peak expiratory flow rate"). How is PEFR related to Exposure?

1. Let's create scatterplot to display the data points. What can you tell?

```
lung <- read.csv('LungDisease.csv')
plot(lung$Exposure, lung$PEFR, xlab="Exposure", ylab="PEFR")
```



$$Y = \beta_0 + \beta_1 X$$

$$PEFR = \beta_0 + \beta_1 Exposure$$

2. The relationship does not seem to be perfectly linear. This is due to the fact, that the response variable is affected by other unknown and/or random processes. Therefore, in some case error (ϵ)will be added to the equation, like so:

$$PEFR = \beta_0 + \beta_1 Exposure + \epsilon$$

3. Use help function to find what the function `lm` does?

```
#help(lm)
```

4. The **lm** function in R can be used to fit a linear regression:

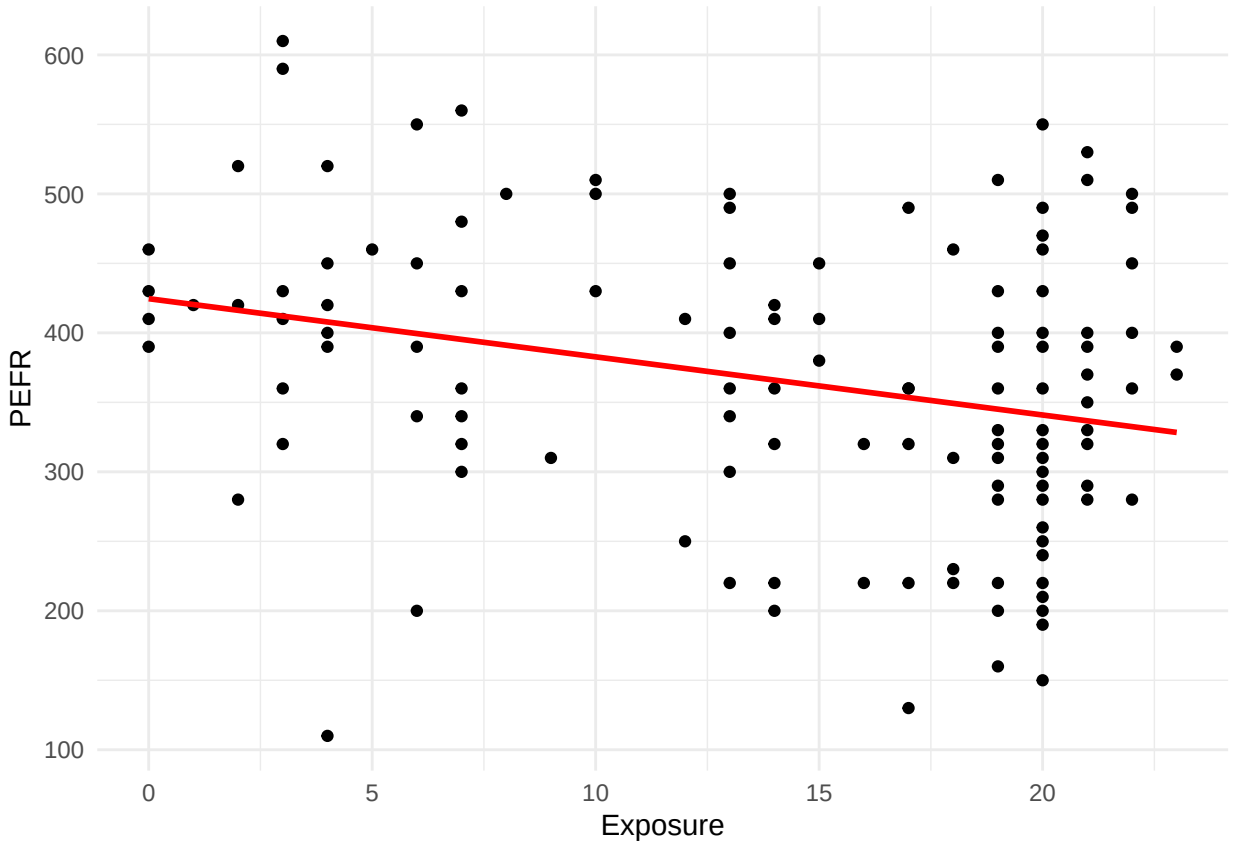
```
lung <- read.csv('LungDisease.csv')
model <- lm(PEFR ~ Exposure, data=lung)
model
```

```
##
## Call:
## lm(formula = PEFR ~ Exposure, data = lung)
##
## Coefficients:
## (Intercept)     Exposure
##     424.583       -4.185
```

5. The intercept, or β_0, is 424.583 and can be interpreted as the predicted PEFR for a worker with zero years exposure. The regression coefficient, or β_1 can be interpreted as follows: for each additional year that a worker is exposed to cotton dust, the worker's PEFR measurement is reduced by ˇ4.185.

6. Let's find the regression line.

```
lung <- read.csv('LungDisease.csv')
model <- lm(PEFR ~ Exposure, data=lung)
library(ggplot2)
ggplot(data = lung, aes(x = Exposure, y = PEFR)) +
  geom_point(color = "black") +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  labs(
       x = "Exposure",
       y = "PEFR") +
  theme_minimal()
```



2.2 Example-2

Build a linear regression model that fits on the `airquality_data` where the response variable is ozone concentration and the explanatory variable is the speed of wind using `lm` function in R.

1. Building the formula

$$ozone = \beta_0 + \beta_1 wind$$

2. building the model in R, and get the summary for the built model.

```
# First specify the dataset
airquality_data <- read.csv('airquality_data.csv')
simple_model <- lm(formula=ozone ~ wind, data=airquality_data)
# Get a summary of the model
summary(simple_model)
```

```
##
## Call:
## lm(formula = ozone ~ wind, data = airquality_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -48.351 -16.648  -3.276  13.866  97.514
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  85.1881     5.9303  14.365  < 2e-16 ***
```

```
## wind         -4.3243     0.5617  -7.699 1.67e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 24.4 on 151 degrees of freedom
## Multiple R-squared:  0.2819, Adjusted R-squared:  0.2771
## F-statistic: 59.28 on 1 and 151 DF,  p-value: 1.667e-12
```

The summary gives us a range of diagnostic information about the model we've fitted:

1. **Call:** This is an R feature that shows what function and parameters were used to create the model.
2. **Residuals:** difference of the observed value and the predicted value.
3. **Estimate:** coefficient estimates for the intercept and explanatory variables.
4. **Std Error:** standard errors (i.e. an estimate of the uncertainty) of the coefficient estimates.
5. **t value**: t-statistic for the t-test comparing whether the coefficient is different to 0.
6. **Pr(>|t|):** p-value for the t statistics, giving the significance of coefficient.
7. **Residual standard error:** an expression of the variation of the observations around the regression line.
8. **Multiple R-squared/Adj. R-squared:** The proportion of the variance in the observed values explained by the model. The Adj. R-squared takes into account the number of variables in the model.
9. **F-statistic, p-value:** Model fit info (allow you to compare different models to assess the best one)

3. p-value helps determine significance of the relationship between $wind$ and $ozone$.
   - Note: The smaller the p-value, the stronger, the more significant the relationship is considered to be, also the stronger he evidence against the null hypothesis
   - p-value=$1.667e-12$ suggests that changes in the explanatory variable (`wind`) are associated with changes in the response variable (`ozone`).
   - We can also see that R2=0.282 which means that the speed of wind explains 28.2% of the variance in average ozone concentration.
4. Regression coefficients:
   - $\beta_0 = 85.2$, $\beta_1 = -4.32$
   - We can re-write the regression model as follows:

$$\hat{ozone} = 85.2 + (-4.32) * wind$$

   - **Intercept interpretation:** The model predicted mean ozone concentration is 85.2 parts in billion when the speed of the wind is 0 miles per hours.

     Slope interpretation If the speed of wind increase by 1 mile per hour, we predict the daily average ozone concentration decreases by 4.32 parts per billion.
   - To better understand this, consider the daily reading of mean ozone concentration for two different values of speed of the wind. The first is when the wind speed is 7 miles per hours and the second is when the wind speed is 8 miles per hours. Now let's compute daily mean ozone concentration using the estimated regression coefficients.

```r
wind_speed <- c(7,8)
# Compute ozone concentration when wind speed is 7 mph and 8mph
85.2-4.32*wind_speed
```

```
## [1] 54.96 50.64
```

2.3 Exercise-1

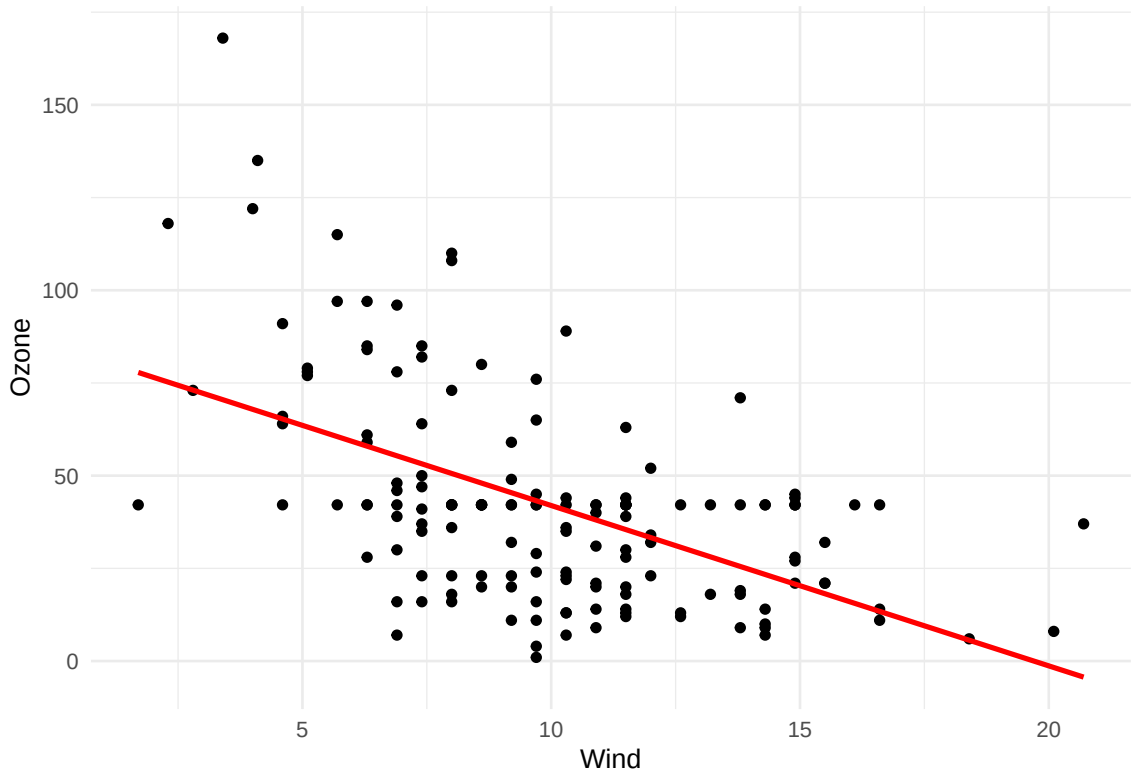
1. Build a linear regression model on the `airquality_data` where the response variable is ozone concentration and the explanatory variable is the temperature.
2. Interpret your estimated regression parameters.
3. Rely on the previously built model above.

The solution can be located in Section 11, which is situated at the end of this document.

3. Predicting using the regression model

- Once we have created a linear regression model, we can use the `predict()` function to predict the response variable for new values given explanatory variables.

```r
AirData <- read.csv('airquality_data.csv')
model <- lm(ozone ~ wind, data=AirData)
library(ggplot2)
ggplot(data = AirData, aes(x = wind, y = ozone)) +
  geom_point(color = "black") +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  labs(
       x = "Wind",
       y = "Ozone") +
  theme_minimal()
```




```
    new_wind <- data.frame(wind = c(23, 26, 30))


    # Use the predict function on new values
    # And also get confidence intervals for these
    predict(simple_model, newdata = new_wind, interval = "confidence")
```

```
    ##         fit       lwr         upr
    ## 1 -14.26965 -29.25870   0.7193894
    ## 2 -27.24240 -45.46673  -9.0180785
    ## 3 -44.53941 -67.11995 -21.9588611
```

```
# Create some new data
new_wind <- data.frame(wind = c(23, 26, 30))


# Use the predict function on new values
# And also get confidence intervals for these
predict(simple_model, newdata = new_wind, interval = "confidence")
```

```
##         fit       lwr         upr
## 1 -14.26965 -29.25870   0.7193894
## 2 -27.24240 -45.46673  -9.0180785
## 3 -44.53941 -67.11995 -21.9588611
```

- The `fit` column is the predicted value, with `lwr` and `upr` showing the confidence intervals.

When the wind speed is 23 mph, the average ozone concentration is -14.26965 parts per billion - an unfeasible value.

3.1 Exercise-2

Predict the ozone concentration when the temperature is 48,52 and 55 degrees Fahrenheit using the linear model created in the last exercise (2.3-exercise)

The solution can be located in Section 11, which is situated at the end of this document.

4. Multiple linear regression

A multiple linear regression model is a generalisation of the simple linear regression model. In the multiple linear regression model, there is one response variable and more than one explanatory variable. The model will contain the intercept $\beta_0$and more than one coefficient, denoted as β_0 ,β_1 ,. . . ,$\beta_k$where k is the number of explanatory variables in the model. The mathematical equation of a multiple linear regression model is shown below:

$$Y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + ... + \beta_k X_{ik} + \epsilon_i$$

$$ozone = \beta_0 + \beta_1 wind + \beta_2 temp + \epsilon$$

```
# First specify your formula then the dataset
airquality_data<-read.csv('airquality_data.csv')
multiple_model <- lm(formula=ozone ~ wind + temp, data=airquality_data)

# Get a summary of the model
summary(multiple_model)
```

```
##
## Call:
## lm(formula = ozone ~ wind + temp, data = airquality_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -38.550 -13.998  -4.306  10.530 104.458
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -41.2159    19.3090  -2.135   0.0344 *
## wind         -2.5986     0.5543  -4.688 6.14e-06 ***
## temp          1.4024     0.2063   6.798 2.35e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 21.4 on 150 degrees of freedom
## Multiple R-squared:  0.451,  Adjusted R-squared:  0.4437
## F-statistic: 61.62 on 2 and 150 DF,  p-value: < 2.2e-16
```

$$\hat{ozone} = -41.2 - 2.6 * wind_1 + 1.4 * temp$$

4.1 Exercise-3

Using airquality_data data, fit a multiple linear regression model with explanatory variables wind, temp and solar_r.

The solution can be located in Section 11, which is situated at the end of this document.

5. Fitted values and residuals

Before describing regression assumptions and regression diagnostics, two key concepts in regression analysis need to be explained:

1. Fitted values
2. Residuals

The fitted (or predicted) values are the y-values that you would expect for the given x-values according to the built regression model (or visually, the best-fitting straight regression line). The fitted value is denoted as in our simple linear regression example, the fitted (predicted) ozone concentration is

$$\hat{ozone} = 85.2 + (-4.32) * wind$$

The estimated random error is called the residuals and it is the difference of the observed value $Y_i$and the fitted (predicted) value $\hat{Y_i}$. So, the residual is denoted as:

$$\hat{e_i} = Y_i - \hat{Y_i}$$

```
library(magrittr) # needs to be run every time you start R and want to use %>%
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:gridExtra':
##
##     combine
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

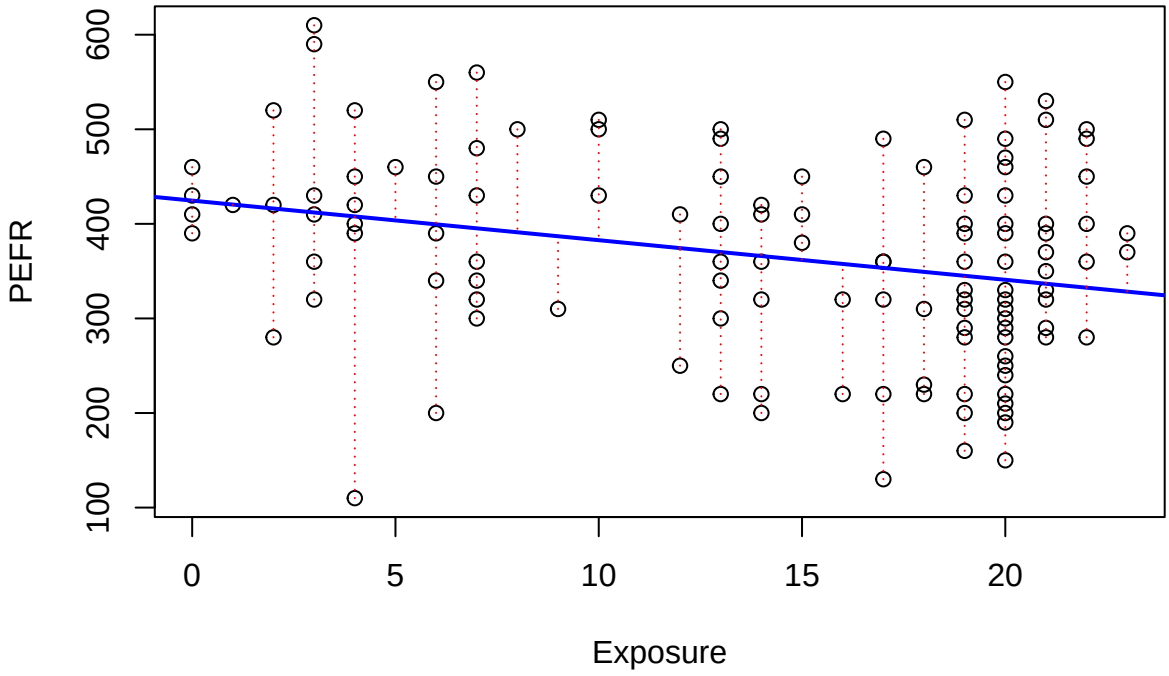
```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
lung <- read.csv('LungDisease.csv')
model <- lm(PEFR ~ Exposure, data=lung)
fitted <- predict(model)
resid <- residuals(model)

lung1 <- lung %>%
  mutate(Fitted=fitted,
         positive = PEFR>Fitted) %>%
  group_by(Exposure, positive) %>%
  summarize(PEFR_max = max(PEFR),
            PEFR_min = min(PEFR),
```

```
            Fitted = first(Fitted),
            .groups='keep') %>%
  ungroup() %>%
  mutate(PEFR = ifelse(positive, PEFR_max, PEFR_min)) %>%
  arrange(Exposure)

plot(lung$Exposure, lung$PEFR, xlab="Exposure", ylab="PEFR")
abline(a=model$coefficients[1], b=model$coefficients[2], col="blue", lwd=2)
segments(lung1$Exposure, lung1$PEFR, lung1$Exposure, lung1$Fitted, col="red", lty=3)
```



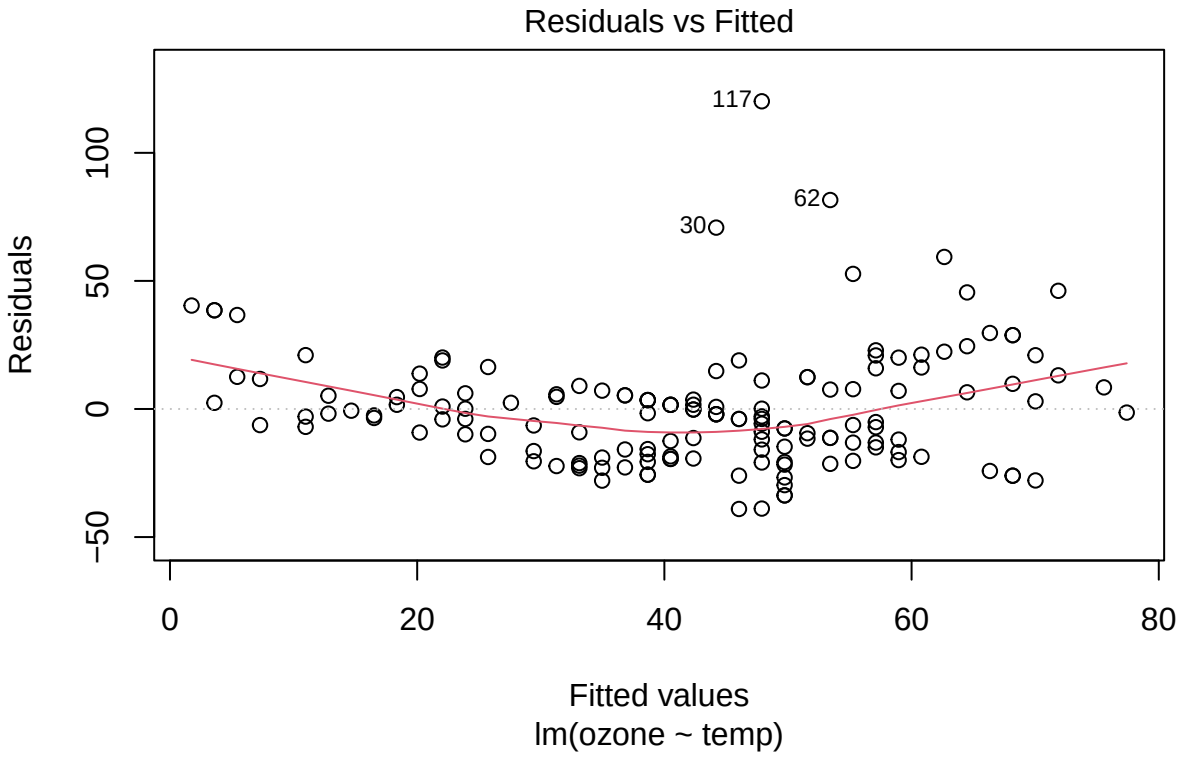
6. Regression Assumptions

- Linear relationship: The relationship between X and Y is linear.
- Independence: The observations are independent of each other.
- Homoscedasticity: The variability of the residuals is constant across all values of X.
- Normality: The residuals are normally distributed.

Residual Diagnostics Plots

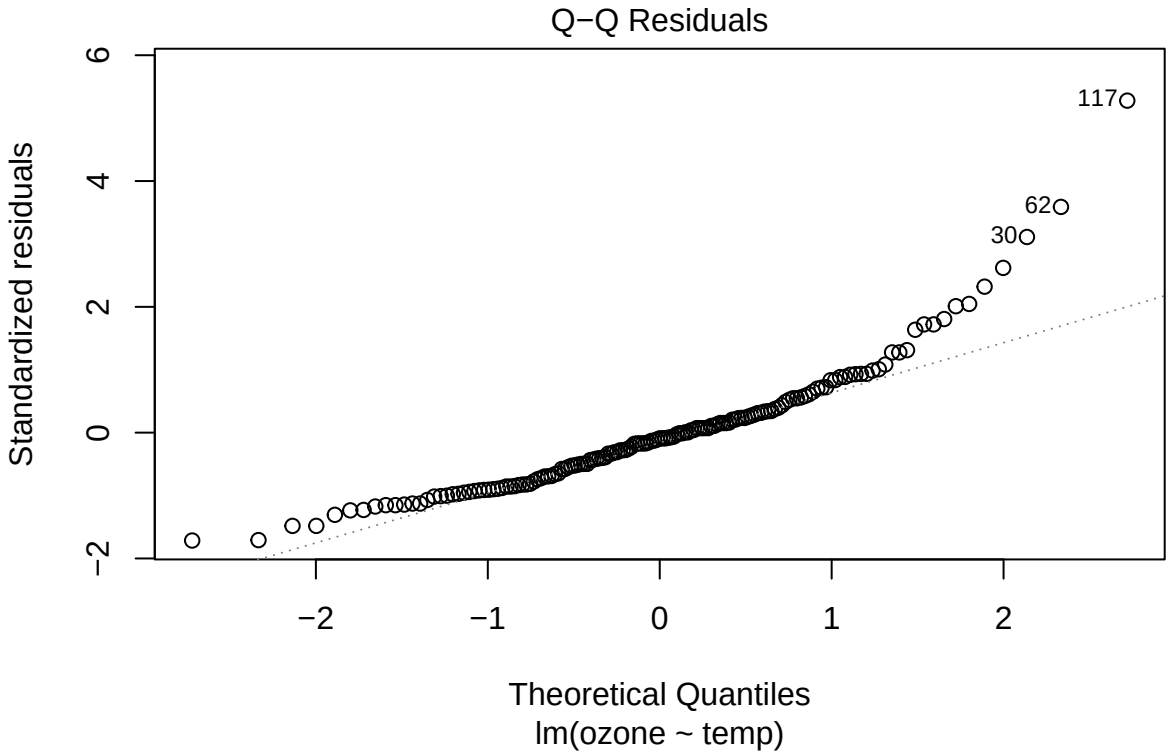
1. **Linearity of the data**

```
airData <- read.csv('airquality_data.csv')
Airmodel <- lm(ozone ~ temp, data=airData)
plot(Airmodel,1)
```



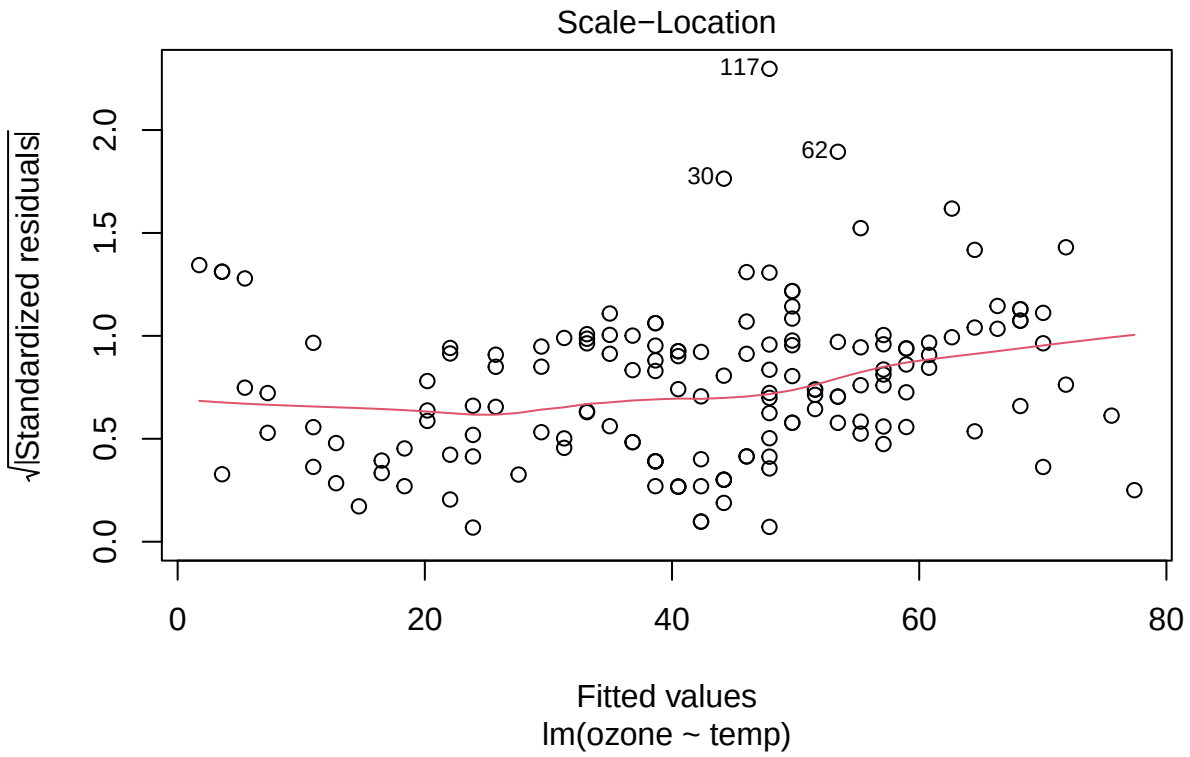
2. **Normality of residual**

```
airData <- read.csv('airquality_data.csv')
Airmodel <- lm(ozone ~ temp, data=airData)
plot(Airmodel,2)
```



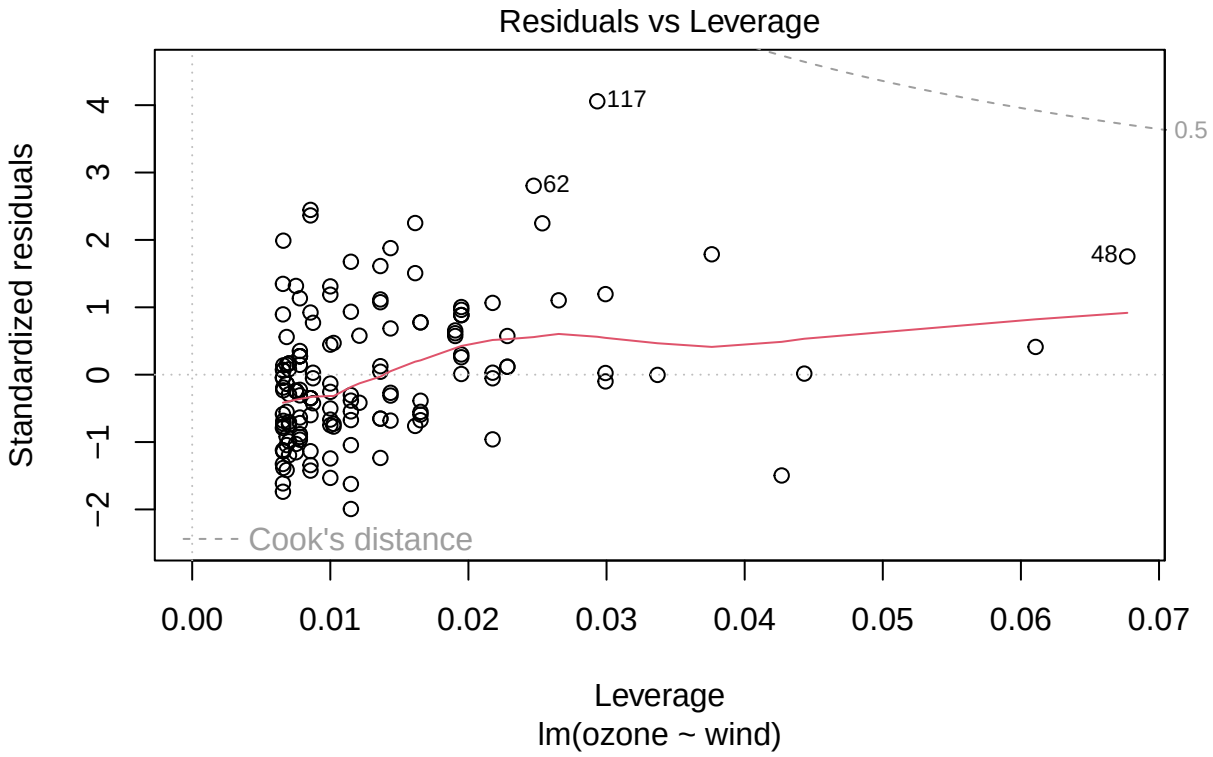
3. **Homoscedasticity**

```
airData <- read.csv('airquality_data.csv')
Airmodel <- lm(ozone ~ temp, data=airData)
plot(Airmodel,3)
```



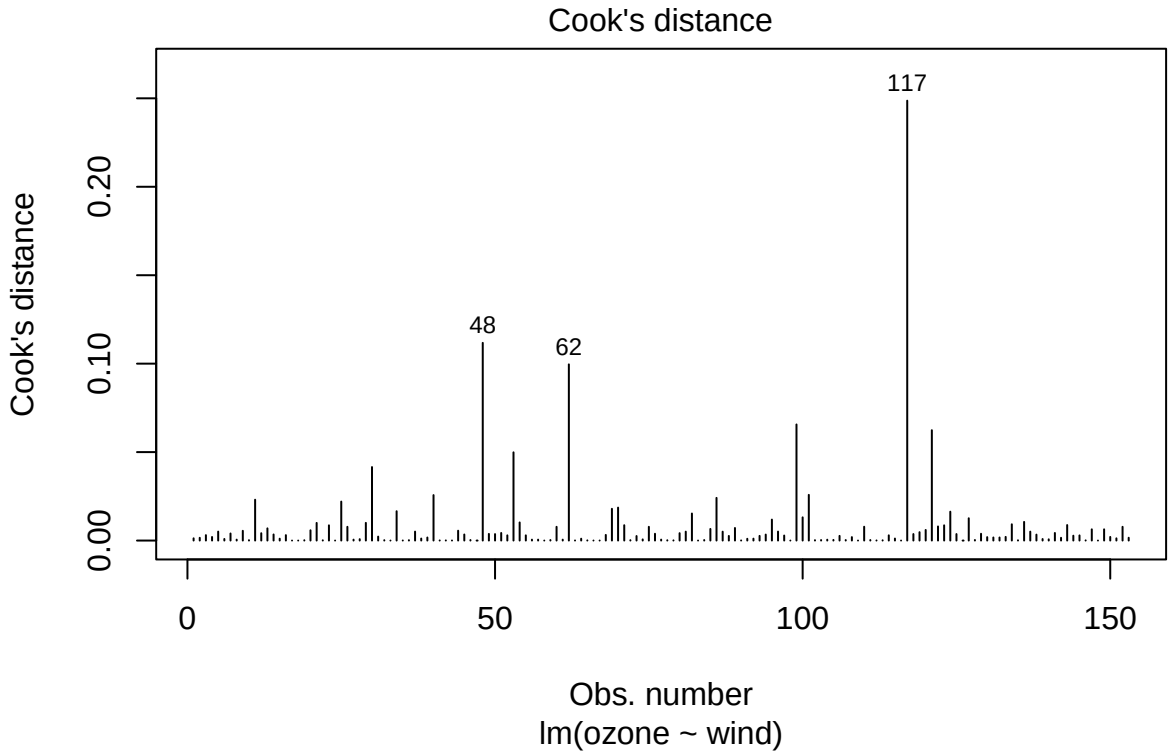
4. **Influential**

```
airData <- read.csv('airquality_data.csv')
Airmodel <- lm(ozone ~ wind, data=airData)
plot(Airmodel,5)
```



- **Cook's Distance**

```
airData <- read.csv('airquality_data.csv')
Airmodel <- lm(ozone ~ wind, data=airData)
plot(Airmodel,4)
```



6.1 Exercise-4

Fit a multiple linear regression on **airquality_data** where the response variable is **ozone** and all explanatory variables are added?

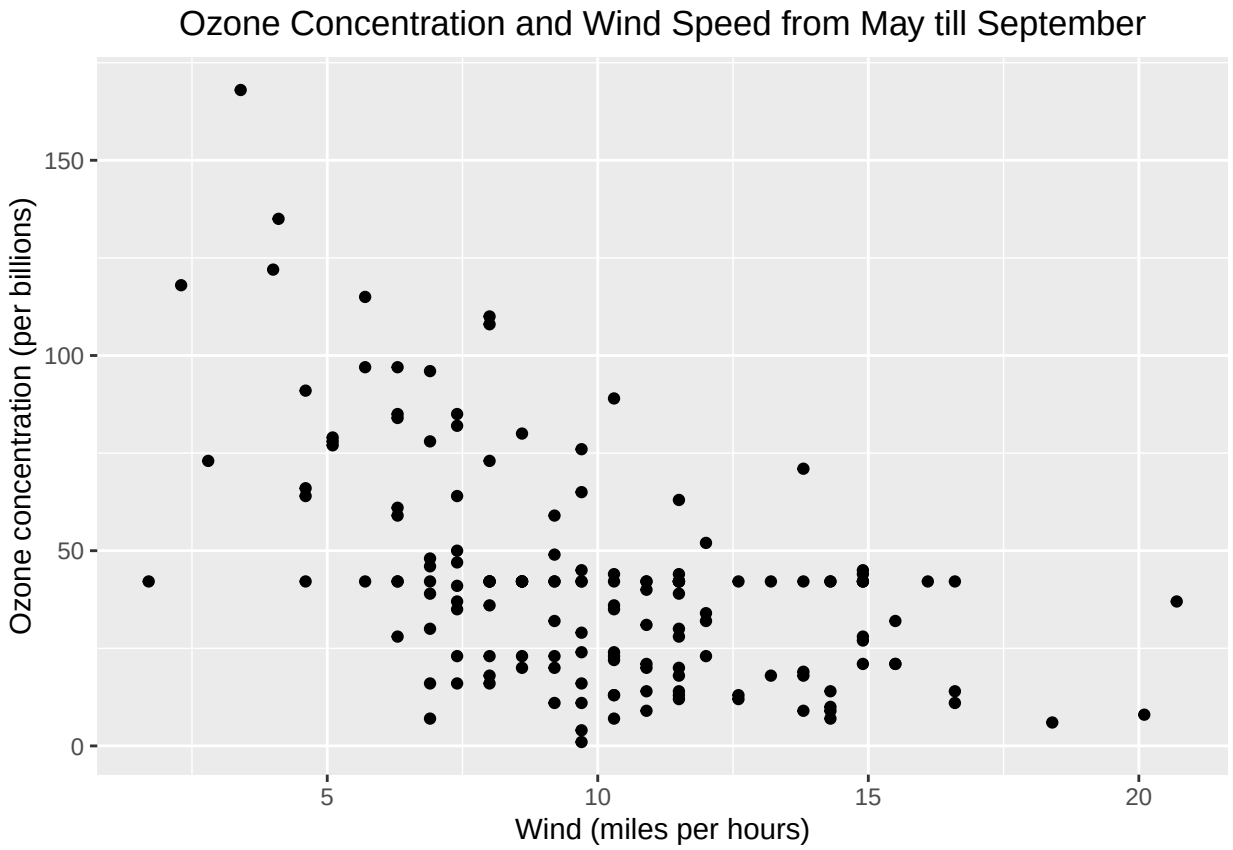
The solution can be located in Section 11, which is situated at the end of this document.

7. Model Transformation

We have seen that the `predict()` function gives us negative values for ozone levels and we have seen above that our regression assumptions are not met. In this case, we can perform transformations inside the model formula where we fit the linear regression model with the response variable `ozone` and the explanatory variable `wind`. First, we will create a scatterplot of `ozone` and `wind` variables to see the relationship between them.

```
#Create a scatter plot
# Specify the x and y axes
# Specify the default geom_point()
# Specify the titles, x and y labels
# Add the regression line
# Centrally align the title text

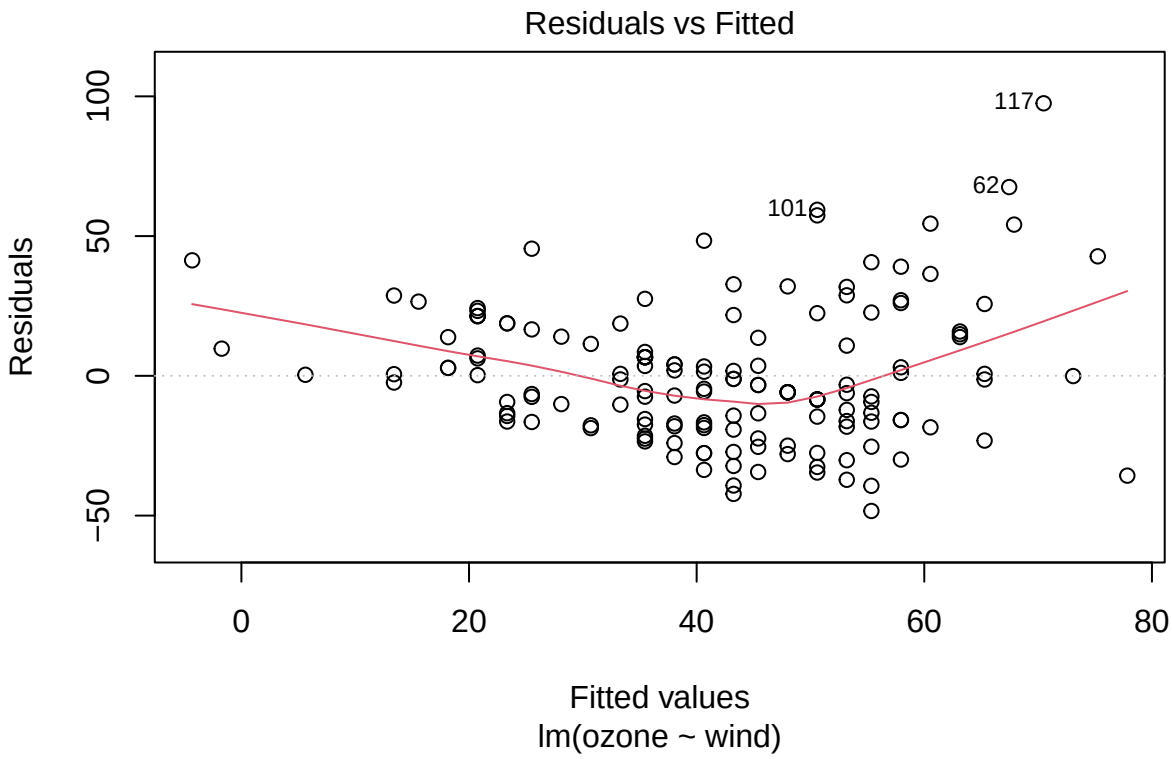
ggplot2::ggplot(airquality_data) +
                aes(x=wind, y=ozone) +
                geom_point() +
                labs(x="Wind (miles per hours)",
                     y = "Ozone concentration (per billions)") +
                ggtitle("Ozone Concentration and Wind Speed from May till September") +
                theme(plot.title = element_text(hjust = 0.5))
```



As you can see the graph does not suggest a linear relationship between ozone and wind. We saw in the previous diagnostic plots that the regression assumptions were not fulfilled. We saw that the data was non-linear and residuals did not have equal variance. We can use variance stabilising transformations such as $\log(Y)$ when we see increasing variance in a fitted versus residuals plot. Hence, we will use the logarithm transformation which causes our mathematical equation of regression model to look like this:

$$\log(ozone_i) = \beta_0 + \beta_1 * wind_i * \epsilon_i$$

```
log_model <- lm(formula=ozone ~ wind, data=airquality_data)
plot(log_model,1)
```




```
summary(log_model)
```

```
##
## Call:
## lm(formula = ozone ~ wind, data = airquality_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -48.351 -16.648  -3.276  13.866  97.514
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  85.1881     5.9303  14.365  < 2e-16 ***
## wind         -4.3243     0.5617  -7.699 1.67e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 24.4 on 151 degrees of freedom
## Multiple R-squared:  0.2819, Adjusted R-squared:  0.2771
## F-statistic: 59.28 on 1 and 151 DF,  p-value: 1.667e-12
```

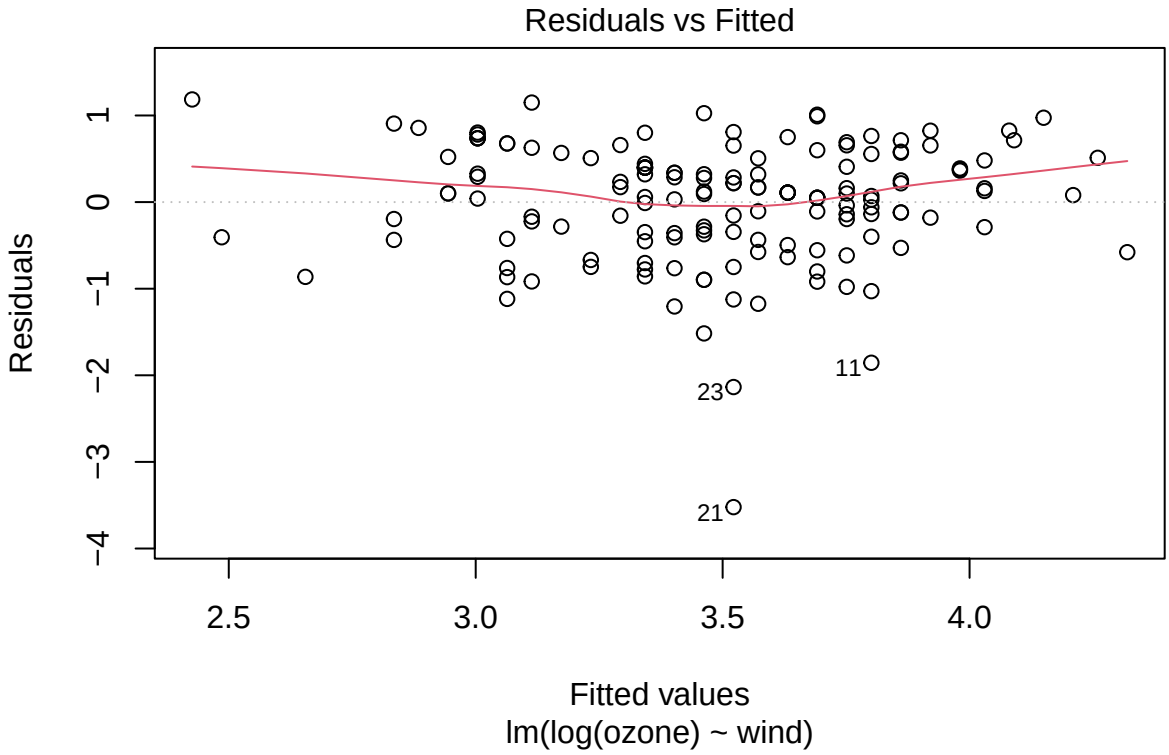
```
new_wind <- data.frame(wind = c(23))
# Use the predict function on new values
# And also get confidence intervals for these
predict(log_model, newdata = new_wind, interval = "confidence")
```

```
##         fit      lwr       upr
## 1 -14.26965 -29.2587 0.7193894
```

```
new_wind <- data.frame(wind = c(24))
predict(log_model, newdata = new_wind, interval = "confidence")
```

```
##        fit       lwr       upr
## 1 -18.5939 -34.65711 -2.530697
```

```
log_model <- lm(formula=log(ozone) ~ wind, data=airquality_data)
plot(log_model,1)
```




```
summary(log_model)
```

```
##
## Call:
## lm(formula = log(ozone) ~ wind, data = airquality_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.5221 -0.3999  0.1004  0.4805  1.1850
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.48875    0.16589  27.059  < 2e-16 ***
## wind        -0.09965    0.01571  -6.343 2.48e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6824 on 151 degrees of freedom
## Multiple R-squared:  0.2104, Adjusted R-squared:  0.2052
## F-statistic: 40.23 on 1 and 151 DF,  p-value: 2.483e-09
```

```
new_wind <- data.frame(wind = c(23))

# Use the predict function on new values
# And also get confidence intervals for these
```

```
predict(log_model, newdata = new_wind, interval = "confidence")
```

```
##        fit      lwr      upr
## 1 2.196698 1.777414 2.615981
```

```
new_wind <- data.frame(wind = c(24))

# Use the predict function on new values
# And also get confidence intervals for these
predict(log_model, newdata = new_wind, interval = "confidence")
```

```
##        fit      lwr      upr
## 1 2.097043 1.647712 2.546374
```

7.1 Exercise-5

Apply the logarithmic transformation on the multiple linear regression containing all explanatory variables from airquality_data.

The solution can be located in Section 11, which is situated at the end of this document.

8. Model Comparison - R-squared, RMSE, RSE, MAE

Now we will compare two multiple linear regression models.

```
# Multiple linear regression
# the response variable is ozone
# explanatory variables are temp and solar_r

model_one <- lm(formula=ozone ~ temp + solar_r, data=airquality_data)

# Multiple linear regression
# the response variable is ozone
# all explanatory variable

model_two <- lm(formula=ozone ~ ., data=airquality_data)
#We will use the glance function in the broom package (broom package is in tidyverse)
#to compare both models.

broom::glance(model_one)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC
##       <dbl>         <dbl> <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1     0.393         0.384  22.5      48.5 5.78e-17     2  -692. 1392. 1404.
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```
broom::glance(model_two)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC
##       <dbl>         <dbl> <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1     0.493         0.475  20.8      28.6 3.71e-20     5  -678. 1370. 1392.
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

8.1 Exercise-6

1. Create two multiple linear regressions as follows:

   Model-1= ozone =solar_r +month

   Model-2=ozone=all variables

2. Compare both models using the `glance()` function, which model is better?

The solution can be located in Section 11, which is situated at the end of this document.

9. Multicollinearity

```
# Multiple linear regression
# the response variable is ozone
# all explanatory variable

model <- lm(formula=ozone ~ ., data=airquality_data)
#We will use the glance function in the broom package (broom package is in tidyverse)
#to compare both models.

car:: vif(model_two)
```

```
##  solar_r     wind     temp    month      day
## 1.143555 1.275384 1.685000 1.273851 1.033068
```

10. Model selection

Best subset linear model using `leaps::regsubsets()`

```
# Use the regsubsets() function to get the best model
# Specify the method to be seqrep (sequential replacement)
# Set nvmax to be 5 since we have 5 explanatory variables

airquality_data=read.csv('airquality_data.csv')
best_model <- leaps::regsubsets(ozone~., data = airquality_data, nvmax = 5, method = "seqrep")
# summary() reports the best set of variables for each model size

summarise_model <- summary(best_model)

summarise_model
```

```
## Subset selection object
## Call: regsubsets.formula(ozone ~ ., data = airquality_data, nvmax = 5,
##     method = "seqrep")
## 5 Variables  (and intercept)
##         Forced in Forced out
## solar_r     FALSE      FALSE
## wind        FALSE      FALSE
## temp        FALSE      FALSE
## month       FALSE      FALSE
## day         FALSE      FALSE
```

```
## 1 subsets of each size up to 5
## Selection Algorithm: 'sequential replacement'
##          solar_r wind temp month day
## 1  ( 1 ) " "     " "  "*"  " "   " "
## 2  ( 1 ) " "     "*"  "*"  " "   " "
## 3  ( 1 ) "*"     "*"  "*"  " "   " "
## 4  ( 1 ) "*"     "*"  "*"  "*"   " "
## 5  ( 1 ) "*"     "*"  "*"  "*"   "*"
```

```
# Find which model has the maximum Adjusted R2
# and minimum residual sum of square RSS
# and minimum BIC

data.frame(Adj_R2 = which.max(summarise_model$adjr2),
           RSS = which.min(summarise_model$rss),BIC = which.min(summarise_model$bic))
```

```
##   Adj_R2 RSS BIC
## 1      5   5   3
```

Stepwise Selection

The stepwise selection consists of iteratively adding and removing explanatory variables into the regression model, to find the subset of variables in the data set resulting in the best performing model; that is a model that lowers prediction error.

- **Forward selection:** It starts with no explanatory variables in the regression model, iteratively adds the most contributive explanatory variables, and stops when the improvement is no longer statistically significant.
- **Backward selection:** It starts with all explanatory variables in the model, iteratively removes the least contributive explanatory variable, and stops when you have a model where all explanatory variables are statistically significant.
- **Stepwise selection:** It is a combination of forward and backward selections. You start with no explanatory variables, then sequentially add the most contributive explanatory variables (like forward selection). After adding each new variable, remove any variables that no longer provide an improvement in the model fit (like backward selection).

We can run a backward selection in R as shown below:

```
airquality_data=read.csv('airquality_data.csv')
full_model <- lm(formula=ozone~., data = airquality_data)

backward_selection <- step(full_model, direction = "backward")
```

```
## Start:  AIC=934.29
## ozone ~ solar_r + wind + temp + month + day
##
##           Df Sum of Sq   RSS    AIC
## - day      1     812.4 64294 934.24
## - month    1     824.5 64306 934.27
## <none>                 63482 934.29
## - solar_r  1    3102.1 66584 939.59
## - wind     1   10546.8 74028 955.81
## - temp     1   15965.1 79447 966.62
##
## Step:  AIC=934.24
## ozone ~ solar_r + wind + temp + month
```

```
##
##           Df Sum of Sq   RSS    AIC
## - month    1     777.3 65071 934.08
## <none>                 64294 934.24
## - solar_r  1    2807.9 67102 938.78
## - wind     1   10729.6 75024 955.85
## - temp     1   15398.6 79693 965.09
##
## Step:  AIC=934.08
## ozone ~ solar_r + wind + temp
##
##           Df Sum of Sq   RSS    AIC
## <none>                 65071 934.08
## - solar_r  1    3630.4 68702 940.38
## - wind     1   10944.3 76016 955.86
## - temp     1   15395.8 80467 964.57
```

11. Exercise solutions:

Exercise 1

- ```
  # Fit linear model using lm()
  simple_model_exercise <- lm(formula=ozone ~ temp, data=airquality_data)

  # Get a summary of the model
  summary(simple_model_exercise)
  ```

  ```
  ##
  ## Call:
  ## lm(formula = ozone ~ temp, data = airquality_data)
  ##
  ## Residuals:
  ##     Min      1Q  Median      3Q     Max
  ## -39.037 -15.883  -2.062   8.437 120.117
  ##
  ## Coefficients:
  ##              Estimate Std. Error t value Pr(>|t|)
  ## (Intercept) -101.5919    15.3546  -6.616 6.02e-10 ***
  ## temp           1.8454     0.1957   9.429  < 2e-16 ***
  ## ---
  ## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
  ##
  ## Residual standard error: 22.84 on 151 degrees of freedom
  ## Multiple R-squared:  0.3706, Adjusted R-squared:  0.3664
  ## F-statistic:  88.9 on 1 and 151 DF,  p-value: < 2.2e-16
  ```

  ```
  #Intercept interpretation: The model predicted mean ozone concentration is -101.6 parts
  #in billion when the temperature is 0 degrees Fahrenheit.

  #Slope interpretation: If the temperature increase by 1 degree Fahrenheit,
  #we predict the daily average ozone concentration increases by 1.85 parts per billion
  ```

  ```

Exercise 2

```r
# Add some new data
simple_model_exercise <- lm(formula=ozone ~ temp, data=airquality_data)

new_temp <- data.frame(temp = c(48, 52, 55))

# Use the predict function on new values
# And also get confidence intervals for these
predict(simple_model_exercise, newdata = new_temp, interval = "confidence")
```

```
##            fit        lwr        upr
## 1 -13.01448564 -25.132313 -0.8966585
## 2  -5.63303263 -16.285998  5.0199328
## 3  -0.09694288  -9.668199  9.4743137
```

Exercise 3

```r
# First specify your formula then the dataset
multiple_model_exercise <- lm(formula=ozone ~ wind + temp + solar_r, data=airquality_data)
# Get a summary of the model
summary(multiple_model_exercise)
```

```
##
## Call:
## lm(formula = ozone ~ wind + temp + solar_r, data = airquality_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -38.618 -14.491  -5.054  12.270 101.176
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -38.22315   18.88338  -2.024  0.04474 *
## wind         -2.71725    0.54280  -5.006 1.55e-06 ***
## temp          1.24126    0.20906   5.937 1.96e-08 ***
## solar_r       0.05775    0.02003   2.883  0.00452 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20.9 on 149 degrees of freedom
## Multiple R-squared:   0.48,  Adjusted R-squared:  0.4696
## F-statistic: 45.85 on 3 and 149 DF,  p-value: < 2.2e-16
```

Exercise 4

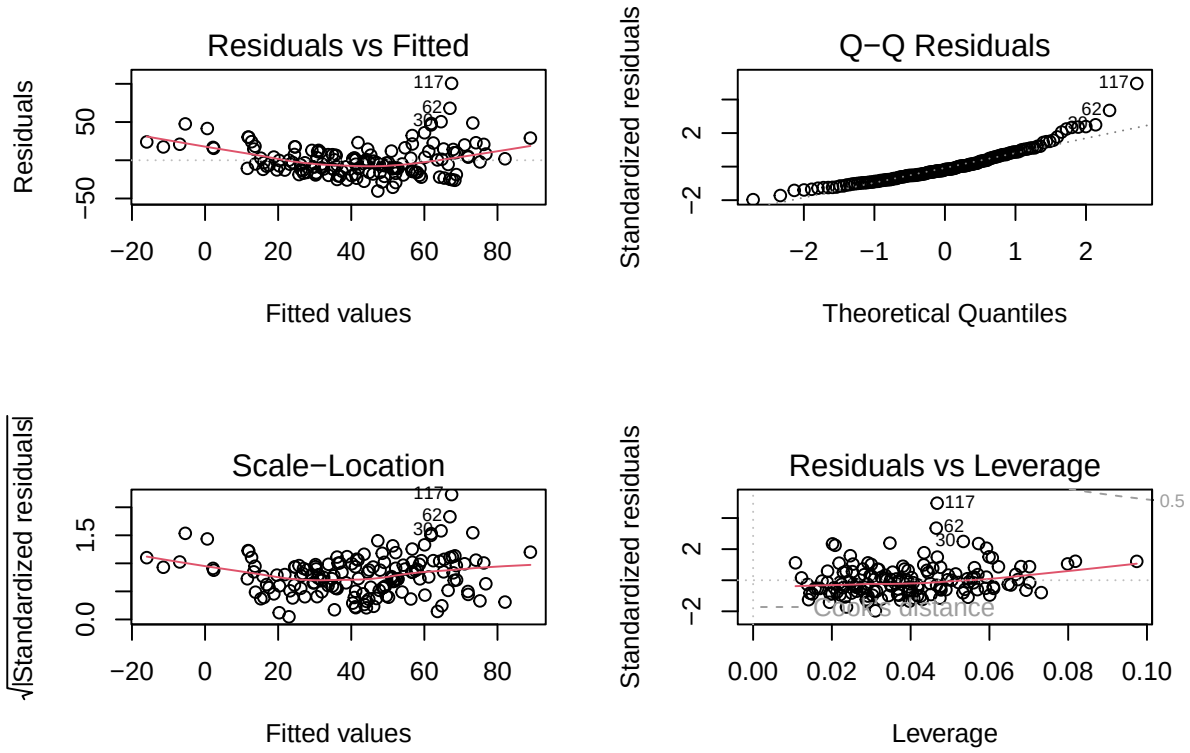
```r
# Fit a multiple linear regression
# the response variable is ozone
# all explanatory variable
airData <- read.csv('airquality_data.csv')
```

```r
multiple_exercise <- lm(formula=ozone ~ ., data=airData)
summary(multiple_exercise)
```

```
##
## Call:
## lm(formula = ozone ~ ., data = airData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -40.263 -13.699  -3.592  10.596 100.549
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -42.16840   19.61843  -2.149   0.0332 *
## solar_r       0.05492    0.02049   2.680   0.0082 **
## wind         -2.67022    0.54032  -4.942 2.08e-06 ***
## temp          1.40550    0.23116   6.080 9.87e-09 ***
## month        -1.85576    1.34301  -1.382   0.1691
## day           0.26508    0.19326   1.372   0.1723
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20.78 on 147 degrees of freedom
## Multiple R-squared:  0.4927, Adjusted R-squared:  0.4755
## F-statistic: 28.56 on 5 and 147 DF,  p-value: < 2.2e-16
```

Let's change the panel layout to 2 x 2 to look at all 4 plots at once

```r
# Change the panel layout to 2 x 2
# Look at all 4 plots at once
par(mfrow = c(2, 2))

# Use plot() function to create diagnostic plots
plot(multiple_exercise)
```



Exercise 5

```
# Apply logarithmic transformation on multiple linear regression
# containing all explanatory variables
log_model_exercise <- lm(formula=log(ozone) ~ ., data =airquality_data)
summary(log_model_exercise)
```

```
##
## Call:
## lm(formula = log(ozone) ~ ., data = airquality_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.46956 -0.29214  0.04401  0.30028  1.54483
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.594684   0.514136   1.157 0.249284
## solar_r      0.002167   0.000537   4.035 8.74e-05 ***
## wind        -0.049640   0.014160  -3.506 0.000604 ***
## temp         0.041244   0.006058   6.808 2.34e-10 ***
## month       -0.045826   0.035196  -1.302 0.194941
## day          0.006418   0.005065   1.267 0.207099
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5446 on 147 degrees of freedom
## Multiple R-squared:  0.5104, Adjusted R-squared:  0.4938
```

```
## F-statistic: 30.65 on 5 and 147 DF,  p-value: < 2.2e-16
```

Exercise 6

1. Create two multiple linear regressions as follows:

   Model-1= ozone =solar_r +month

   Model-2=ozone=all variables

2. Compare both models using the `glance()` function, which model is better?

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic   p.value    df logLik   AIC   BIC
##       <dbl>         <dbl> <dbl>     <dbl>     <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1     0.121         0.110  27.1      10.3 0.0000617     2  -720. 1449. 1461.
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC
##       <dbl>         <dbl> <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1     0.493         0.475  20.8      28.6 3.71e-20     5  -678. 1370. 1392.
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```