QTM 220 Final

AUTHOR
Veronica Vargas

```r
## HW 1-4
library(tidyverse)
```

```
Warning: package 'tidyverse' was built under R version 4.3.3

── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
✔ dplyr     1.1.3     ✔ readr     2.1.4
✔ forcats   1.0.0     ✔ stringr   1.5.0
✔ ggplot2   3.4.3     ✔ tibble    3.2.1
✔ lubridate 1.9.2     ✔ tidyr     1.3.0
✔ purrr     1.0.2
── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become
errors
```

```r
library(ggplot2)
library(dplyr)

## HW 6 - Cross Validation
library(mosaic)
```

```
Warning: package 'mosaic' was built under R version 4.3.3

Registered S3 method overwritten by 'mosaic':
  method                           from
  fortify.SpatialPolygonsDataFrame ggplot2

The 'mosaic' package masks several functions from core packages in order to add
additional features.  The original behavior of these functions should not be affected by this.

Attaching package: 'mosaic'

The following object is masked from 'package:Matrix':

    mean

The following objects are masked from 'package:dplyr':

    count, do, tally

The following object is masked from 'package:purrr':
```

```
    cross

The following object is masked from 'package:ggplot2':

    stat

The following objects are masked from 'package:stats':

    binom.test, cor, cor.test, cov, fivenum, IQR, median, prop.test,
    quantile, sd, t.test, var

The following objects are masked from 'package:base':

    max, mean, min, prod, range, sample, sum
```

```r
library(mosaicData)
library(leaps)
```

```
Warning: package 'leaps' was built under R version 4.3.3
```

```r
library(caret)
```

```
Warning: package 'caret' was built under R version 4.3.3


Attaching package: 'caret'

The following object is masked from 'package:mosaic':

    dotPlot

The following object is masked from 'package:purrr':

    lift
```

```r
library(ISLR2)
```

```
Warning: package 'ISLR2' was built under R version 4.3.3
```

```r
## HW 7 - Confidence Intervals for Regression Modeling
library(datasets)
library(boot)
```

```
Warning: package 'boot' was built under R version 4.3.3


Attaching package: 'boot'
```

```
The following object is masked from 'package:mosaic':

    logit

The following object is masked from 'package:lattice':

    melanoma
```

```
library(lmtest) # sandwich package
```

```
Warning: package 'lmtest' was built under R version 4.3.3

Loading required package: zoo

Warning: package 'zoo' was built under R version 4.3.3


Attaching package: 'zoo'

The following objects are masked from 'package:base':

    as.Date, as.Date.numeric
```

```
## HW 8 - Diagnostic Plots
library(experimentr)
library(datasets)
```

```
pokemon.data <- read.csv("C:/Users/13015/OneDrive - Emory University/Documents/Fall 2024/QTM 220/

pokemon.data <- pokemon.data %>%
  filter(Generation %in% c(1, 2, 3, 4, 5, 6, 7))

head(pokemon.data)
```

```
           Name Generation Type1 HP Attack Defense SP_Attack SP_Defense Speed
1     Bulbasaur          1 Grass 45     49      49        65         65    45
2       Ivysaur          1 Grass 60     62      63        80         80    60
3      Venusaur          1 Grass 80     82      83       100        100    80
4 Mega Venusaur          1 Grass 80    100     123       122        120    80
5    Charmander          1  Fire 39     52      43        60         50    65
6    Charmeleon          1  Fire 58     64      58        80         65    80
  predicted_speed
1        56.23955
2        62.26298
3        70.29423
4        79.12860
5        67.27336
6        71.99359
```

```r
summary(pokemon.data)
```

```
     Name             Generation        Type1                 HP        
 Length:940         Min.   :1.000   Length:940         Min.   :  1.00  
 Class :character   1st Qu.:2.000   Class :character   1st Qu.: 50.00  
 Mode  :character   Median :4.000   Mode  :character   Median : 66.00  
                    Mean   :3.818                      Mean   : 69.55  
                    3rd Qu.:5.000                      3rd Qu.: 80.00  
                    Max.   :7.000                      Max.   :255.00  
     Attack          Defense         SP_Attack        SP_Defense    
 Min.   :  5.00   Min.   :  5.00   Min.   : 10.00   Min.   : 20.00  
 1st Qu.: 55.00   1st Qu.: 50.00   1st Qu.: 50.00   1st Qu.: 50.00  
 Median : 75.00   Median : 70.00   Median : 65.00   Median : 70.00  
 Mean   : 80.05   Mean   : 74.38   Mean   : 73.28   Mean   : 72.12  
 3rd Qu.:100.00   3rd Qu.: 90.00   3rd Qu.: 95.00   3rd Qu.: 90.00  
 Max.   :190.00   Max.   :230.00   Max.   :194.00   Max.   :230.00  
     Speed       predicted_speed 
 Min.   :  5.0   Min.   : 24.54  
 1st Qu.: 45.0   1st Qu.: 58.21  
 Median : 65.0   Median : 66.86  
 Mean   : 68.8   Mean   : 68.80  
 3rd Qu.: 90.0   3rd Qu.: 78.84  
 Max.   :180.0   Max.   :131.00  
```

Exercise #1

(a) Scatter Plot #1-2

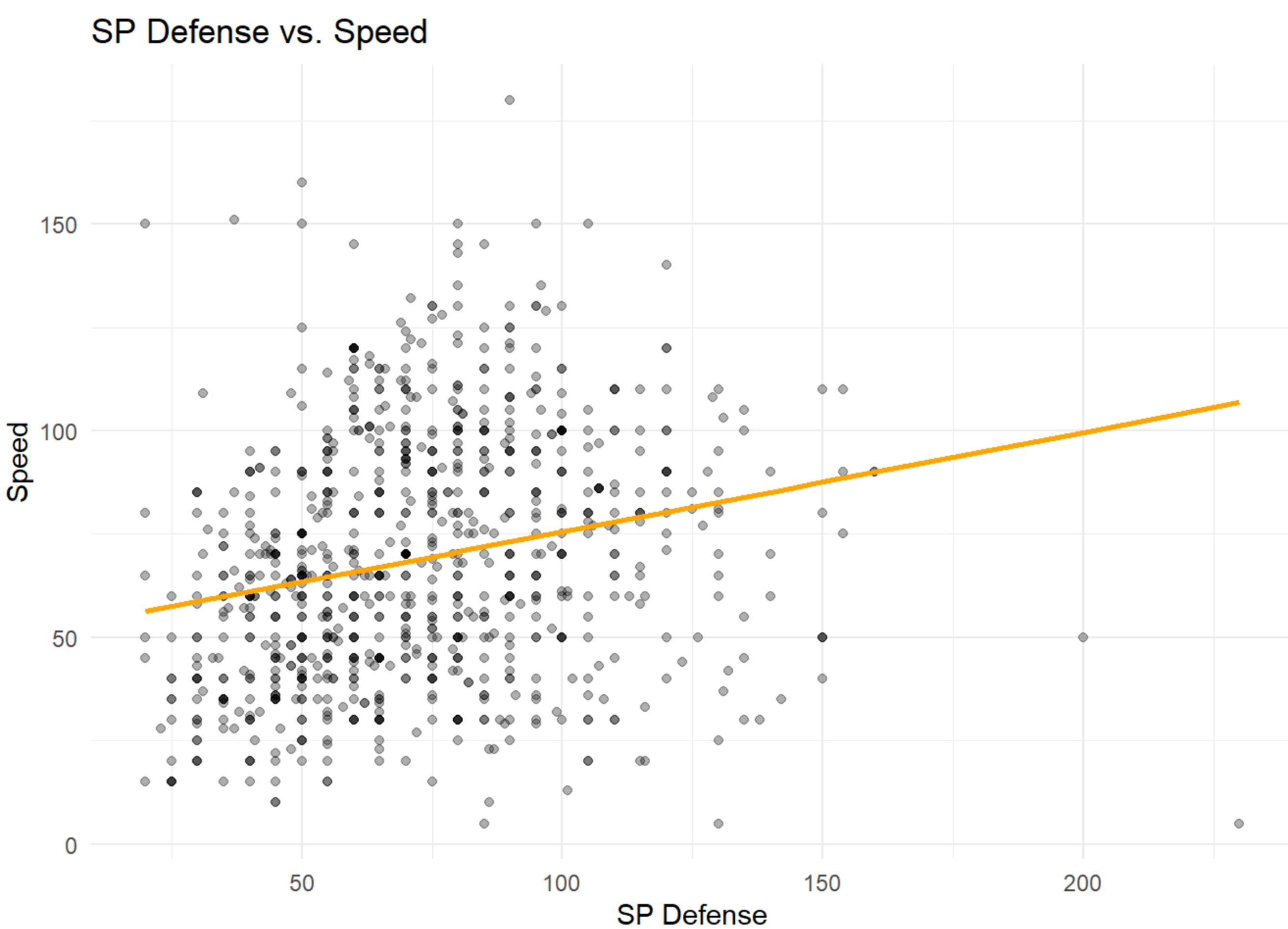
```r
mod.simple <- lm(Speed ~ SP_Defense, data = pokemon.data)
summary(mod.simple)
```

```

Call:
lm(formula = Speed ~ SP_Defense, data = pokemon.data)

Residuals:
    Min      1Q  Median      3Q     Max 
-101.80  -21.89   -1.07   20.89  106.90 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 51.44342    2.62866  19.570   <2e-16 ***
SP_Defense   0.24067    0.03404   7.071    3e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 28.84 on 938 degrees of freedom
```

```
Multiple R-squared:  0.05061,	Adjusted R-squared:  0.04959
F-statistic:    50 on 1 and 938 DF,  p-value: 3.004e-12
```

```
pokemon.data$predicted_score_simple <- predict(mod.simple)

ggplot(pokemon.data, aes(x = SP_Defense, y = Speed)) +
  geom_point(aes(x = SP_Defense, y = Speed),
             alpha = 0.3) +
  geom_line(aes(y = predicted_score_simple), color = "orange", size = 1) +
  labs(
    title = "SP Defense vs. Speed",
    x = "SP Defense",
    y = "Speed") +
  theme_minimal()
```

```
Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
ℹ Please use `linewidth` instead.
```

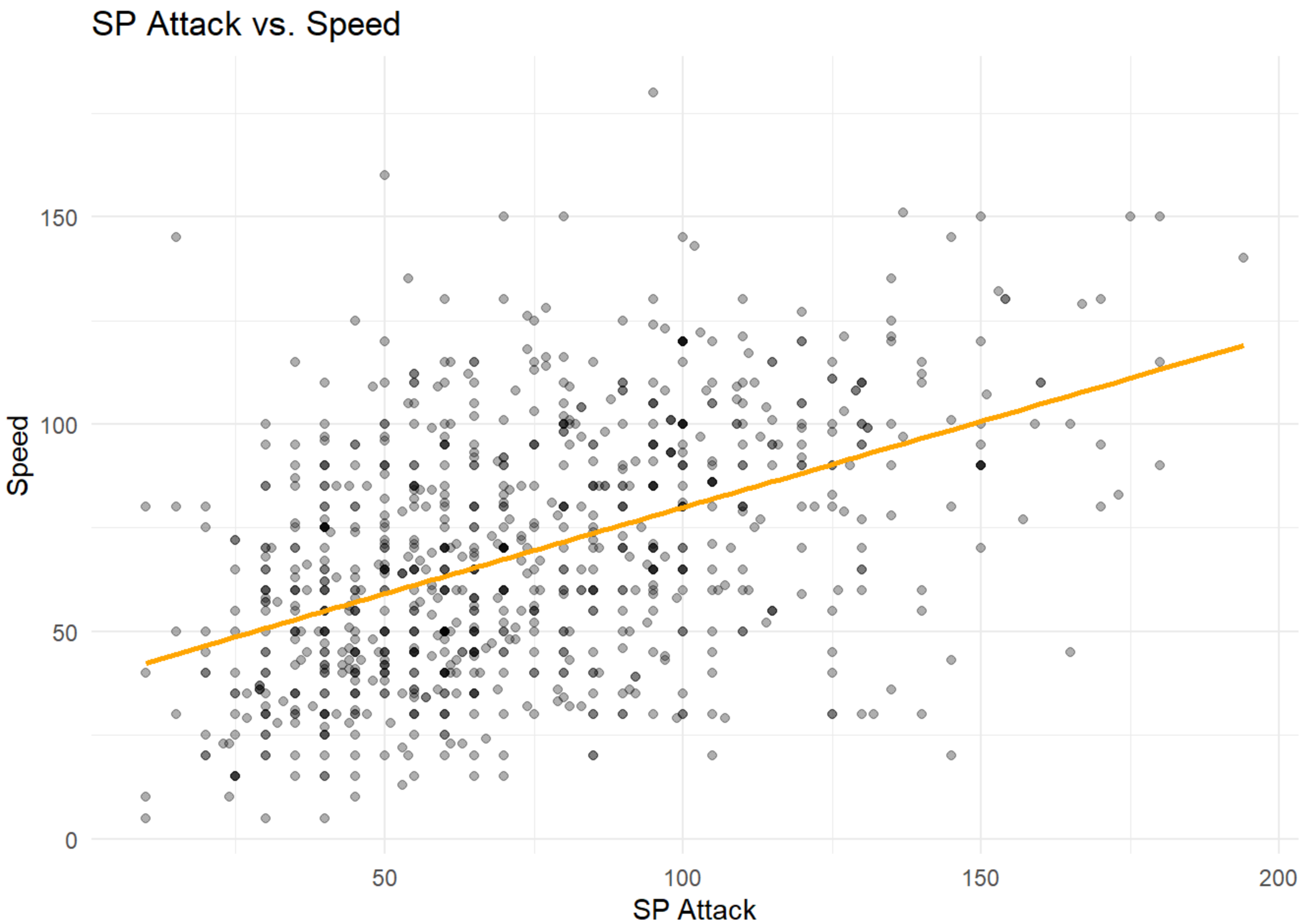



```
mod.simple <- lm(Speed ~ SP_Attack, data = pokemon.data)
summary(mod.simple)
```

```
Call:
```

```
lm(formula = Speed ~ SP_Attack, data = pokemon.data)

Residuals:
    Min      1Q  Median      3Q     Max 
-78.645 -18.685  -0.354  17.970 102.161 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 38.30555    2.07103   18.50   <2e-16 ***
SP_Attack    0.41614    0.02575   16.16   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 26.18 on 938 degrees of freedom
Multiple R-squared:  0.2178,	Adjusted R-squared:  0.217 
F-statistic: 261.2 on 1 and 938 DF,  p-value: < 2.2e-16
```

```r
pokemon.data$predicted_score_simple <- predict(mod.simple)

ggplot(pokemon.data, aes(x = SP_Attack, y = Speed)) +
  geom_point(aes(x = SP_Attack, y = Speed),
             alpha = 0.3) +
  geom_line(aes(y = predicted_score_simple), color = "orange", size = 1) +
  labs(
    title = "SP Attack vs. Speed",
    x = "SP Attack",
    y = "Speed") +
  theme_minimal()
```



SP Attack is more highly correlated with Speed. When we're looking at the coefficients for each simple regression model, the coefficient between SP Attack and Speed is 0.41614, which is larger than the coefficient between SP Defense and Speed which is 0.24067.

(b) Quantile w/ Bootstrapped Estimated Sampling Distribution

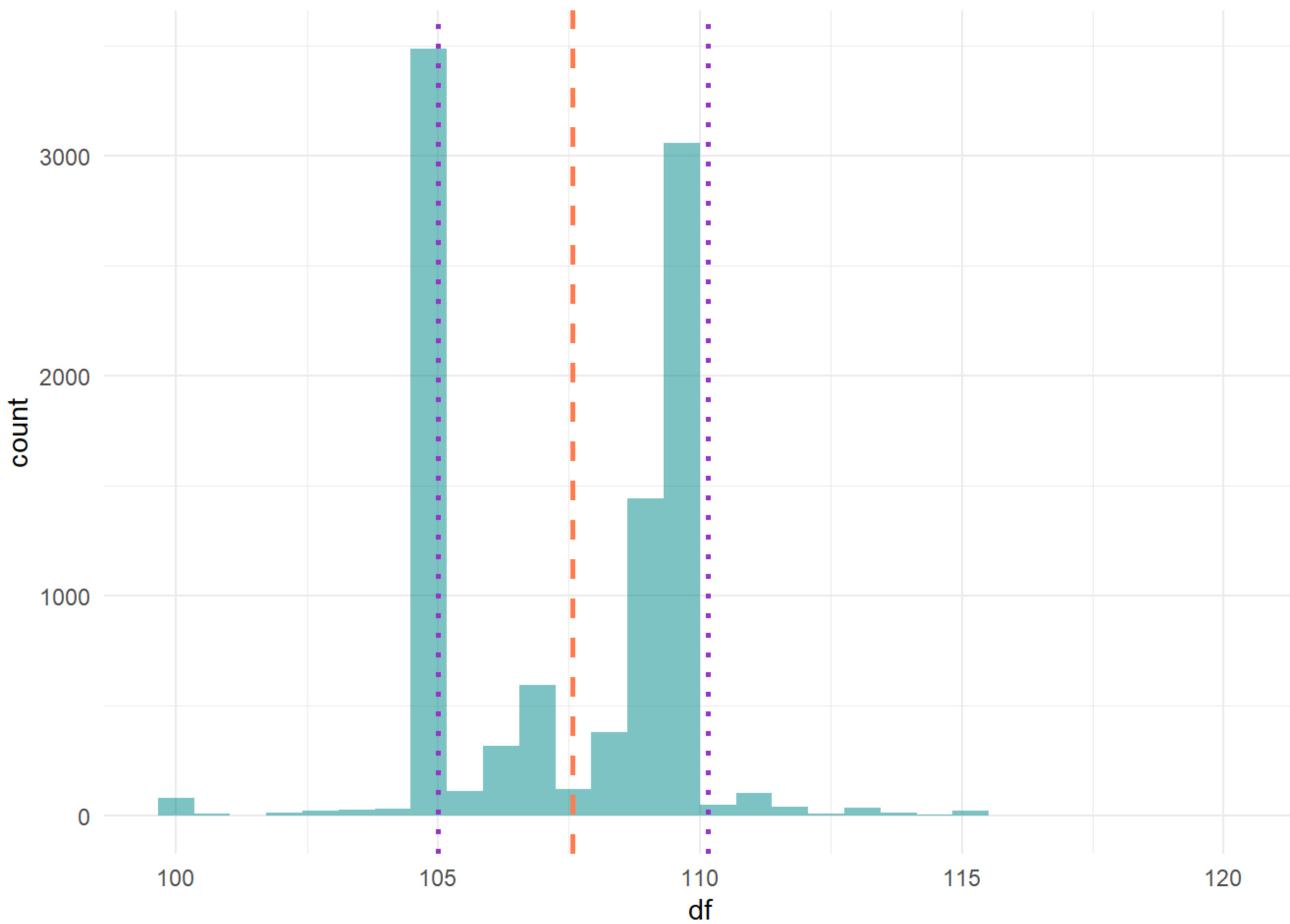
```
quantile(pokemon.data$SP_Attack, 0.85)
```

```
   85%
108.15
```

```
n <- 10000
df <-  rep(NA, n)

for(i in 1:n){
  sample <- sample(pokemon.data$SP_Attack, 940, replace = T)

  df[i] <- quantile(sample, 0.85)
}
```

```r
ggplot(data = data.frame(df = df), aes(x = df)) +
  geom_histogram(fill = "cyan4", alpha = 0.5, Type1 = "identity") +
  geom_vline(xintercept = mean(df), linetype="dashed",
              color = "coral", linewidth=1) +
  geom_vline(xintercept = quantile(df, 0.025), linetype = 'dotted',
              color = "darkorchid", linewidth = 1) +
    geom_vline(xintercept = quantile(df, 0.975), linetype = "dotted",
              color = "darkorchid", linewidth=1) +
  theme_minimal()
```

```
Warning in geom_histogram(fill = "cyan4", alpha = 0.5, Type1 = "identity"):
Ignoring unknown parameters: `Type1`

`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```r
lower.bound <- quantile(df, 0.05)
upper.bound <- quantile(df, 0.95)

print(paste0("The Bootstrapped 90% CI is {", lower.bound,", ",upper.bound,"}"))
```

```
[1] "The Bootstrapped 90% CI is {105, 110}"
```

If we were to repeat this experiment under the same conditions with a sufficiently large sample size, the quantile 85 would fall somewhere within the estimated interval for about 95 out of 100 trials. In this experiment, expected value of the quantile 85 for our estimator is predicted to be somewhere in between 105 and 110.

(c) Not-Necessarily Parallel Lines Model #1

```
mod.interaction1 <- lm(Speed ~ SP_Attack + Type1 + SP_Attack*Type1, data = pokemon.data)
summary(mod.interaction1)
```

```
Call:
lm(formula = Speed ~ SP_Attack + Type1 + SP_Attack * Type1, data = pokemon.data)

Residuals:
    Min      1Q  Median      3Q     Max 
-68.148 -16.774  -1.511  16.212 104.246 

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)    
(Intercept)              3.263e+01  6.069e+00   5.377 9.64e-08 ***
SP_Attack                5.415e-01  9.615e-02   5.632 2.38e-08 ***
Type1Dark                1.505e+01  1.195e+01   1.259  0.20851    
Type1Dragon              1.163e+01  1.212e+01   0.960  0.33754    
Type1Electric            4.099e+01  1.269e+01   3.229  0.00129 ** 
Type1Fairy              -3.571e+01  1.936e+01  -1.845  0.06539 .  
Type1Fighting            3.938e+00  1.162e+01   0.339  0.73485    
Type1Fire                2.048e+01  1.199e+01   1.708  0.08799 .  
Type1Flying             -6.756e+00  4.191e+01  -0.161  0.87199    
Type1Ghost              -1.253e+01  1.306e+01  -0.960  0.33733    
Type1Grass              -2.494e+00  1.027e+01  -0.243  0.80824    
Type1Ground              1.082e+01  1.150e+01   0.941  0.34706    
Type1Ice                -3.505e+00  1.417e+01  -0.247  0.80473    
Type1Normal              2.035e+01  8.490e+00   2.397  0.01674 *  
Type1Poison             -8.686e+00  1.404e+01  -0.619  0.53617    
Type1Psychic            -3.401e+00  1.037e+01  -0.328  0.74298    
Type1Rock               -1.519e+01  1.017e+01  -1.494  0.13558    
Type1Steel              -1.281e+01  1.231e+01  -1.041  0.29828    
Type1Water               7.241e+00  8.516e+00   0.850  0.39537    
SP_Attack:Type1Dark     -1.410e-01  1.618e-01  -0.872  0.38365    
SP_Attack:Type1Dragon   -1.329e-01  1.410e-01  -0.943  0.34606    
SP_Attack:Type1Electric -3.877e-01  1.504e-01  -2.577  0.01012 *  
SP_Attack:Type1Fairy     1.512e-01  2.410e-01   0.627  0.53061    
SP_Attack:Type1Fighting  6.202e-02  1.926e-01   0.322  0.74749    
SP_Attack:Type1Fire     -3.055e-01  1.463e-01  -2.088  0.03708 *  
SP_Attack:Type1Flying    2.715e-01  4.300e-01   0.631  0.52799    
SP_Attack:Type1Ghost     2.525e-04  1.663e-01   0.002  0.99879    
SP_Attack:Type1Grass    -1.399e-01  1.405e-01  -0.996  0.31970    
```

```
SP_Attack:Type1Ground   -1.595e-01  1.860e-01  -0.858  0.39137
SP_Attack:Type1Ice      -5.483e-02  1.903e-01  -0.288  0.77331
SP_Attack:Type1Normal   -2.340e-01  1.352e-01  -1.731  0.08386 .
SP_Attack:Type1Poison    1.007e-01  2.119e-01   0.475  0.63485
SP_Attack:Type1Psychic  -1.913e-02  1.239e-01  -0.154  0.87732
SP_Attack:Type1Rock      1.682e-01  1.482e-01   1.135  0.25650
SP_Attack:Type1Steel    -5.255e-02  1.644e-01  -0.320  0.74938
SP_Attack:Type1Water    -2.081e-01  1.206e-01  -1.725  0.08481 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 25.24 on 904 degrees of freedom
Multiple R-squared:  0.2989,	Adjusted R-squared:  0.2718 
F-statistic: 11.01 on 35 and 904 DF,  p-value: < 2.2e-16
```

According to the summary, there are least four predictors that is in a relationship with the response variable. These predictor variables are SP_Attack, Type1Electric, Type1Normal, and the SP_Attack*Type1Fire interaction variable. These are distinguishable due to their reported p-value being less than 0.05.

(d) Not-Necessarily Parallel Lines Model #2

```
mod.interaction2 <- lm(Speed ~ SP_Defense + Type1 + SP_Defense*Type1, data = pokemon.data)
summary(mod.interaction2)
```

```
Call:
lm(formula = Speed ~ SP_Defense + Type1 + SP_Defense * Type1, 
    data = pokemon.data)

Residuals:
    Min      1Q  Median      3Q     Max 
-69.821 -19.245  -1.964  18.228  98.030 

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)    
(Intercept)              58.651220   7.167887   8.182 9.39e-16 ***
SP_Defense                0.066373   0.101193   0.656   0.5120    
Type1Dark                 6.962557  15.564572   0.447   0.6547    
Type1Dragon             -18.942497  16.394883  -1.155   0.2482    
Type1Electric            -2.255614  16.051501  -0.141   0.8883    
Type1Fairy              -54.511405  21.984922  -2.479   0.0133 *  
Type1Fighting           -30.584150  17.369807  -1.761   0.0786 .  
Type1Fire                -4.233655  14.373727  -0.295   0.7684    
Type1Flying             -58.634271  54.771031  -1.071   0.2847    
Type1Ghost               -5.308765  16.394816  -0.324   0.7462    
Type1Grass              -24.346870  12.507340  -1.947   0.0519 .  
Type1Ground             -23.844350  16.652060  -1.432   0.1525    
Type1Ice                -16.207604  14.326395  -1.131   0.2582    
Type1Normal              -0.987178  10.156964  -0.097   0.9226    
```

```
Type1Poison                  0.573155  15.898562   0.036   0.9712
Type1Psychic                -3.158566  12.599191  -0.251   0.8021
Type1Rock                    1.713205  11.794652   0.145   0.8845
Type1Steel                 -32.574608  16.487137  -1.976   0.0485 *
Type1Water                  -2.528245   9.697600  -0.261   0.7944
SP_Defense:Type1Dark         0.096684   0.217902   0.444   0.6574
SP_Defense:Type1Dragon       0.421686   0.189263   2.228   0.0261 *
SP_Defense:Type1Electric     0.356461   0.212465   1.678   0.0937 .
SP_Defense:Type1Fairy        0.482971   0.250402   1.929   0.0541 .
SP_Defense:Type1Fighting     0.556814   0.252428   2.206   0.0276 *
SP_Defense:Type1Fire         0.206953   0.194017   1.067   0.2864
SP_Defense:Type1Flying       1.347186   0.731037   1.843   0.0657 .
SP_Defense:Type1Ghost        0.062274   0.209246   0.298   0.7661
SP_Defense:Type1Grass        0.308848   0.171809   1.798   0.0726 .
SP_Defense:Type1Ground       0.407313   0.248950   1.636   0.1022
SP_Defense:Type1Ice          0.226365   0.181028   1.250   0.2115
SP_Defense:Type1Normal       0.135476   0.145570   0.931   0.3523
SP_Defense:Type1Poison       0.013611   0.226189   0.060   0.9520
SP_Defense:Type1Psychic      0.229739   0.150420   1.527   0.1270
SP_Defense:Type1Rock        -0.008551   0.155864  -0.055   0.9563
SP_Defense:Type1Steel        0.295343   0.199762   1.478   0.1396
SP_Defense:Type1Water        0.058492   0.130929   0.447   0.6552
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 27.81 on 904 degrees of freedom
Multiple R-squared:  0.1493,    Adjusted R-squared:  0.1164 
F-statistic: 4.534 on 35 and 904 DF,  p-value: 5.464e-16
```

The multiple R-squared is 0.1493, meaning that 0.1493 of the variance can be explained by the relationships in the model. Since this number is very low, it's likely that this combination of predictor variables in this multivariate regression do not have much to do with the response variable.

(e) LOOCV

```r
rss_summary <- function(data, lev = NULL, model = NULL) {
  residuals <- data$obs - data$pred
  rss <- sum(residuals^2)
  rmse <- sqrt(mean(residuals^2))
  return(c(RMSE = rmse, RSS = rss))
}
```

```r
train_control_loocv <- trainControl(
  method = "LOOCV",
  summaryFunction = rss_summary,
  savePredictions = "all",
  classProbs = FALSE,
  allowParallel = FALSE
)
```

```
# Train Model A: Model #1
set.seed(123)
model_A_caret_loocv <- train(
  Speed ~ SP_Attack + Type1 + SP_Attack*Type1,
  data = pokemon.data,
  method = "lm",
  trControl = train_control_loocv,
  metric = "RMSE"
)

# Train Model B: Model #2
set.seed(123)
model_B_caret_loocv <- train(
  Speed ~ SP_Defense + Type1 + SP_Defense*Type1,
  data = pokemon.data,
  method = "lm",
  trControl = train_control_loocv,
  metric = "RMSE"
)

model_A_caret_loocv$results
```

```
  intercept     RMSE      RSS
1      TRUE 25.81087 626228.8
```

```
model_B_caret_loocv$results
```

```
  intercept     RMSE      RSS
1      TRUE 28.55635 766537.3
```

Using the LOOCV, I prefer Model #1 since the RMSE value is lower than the RMSE of Model #2. In other words, Model #1 features less error than Model #2.

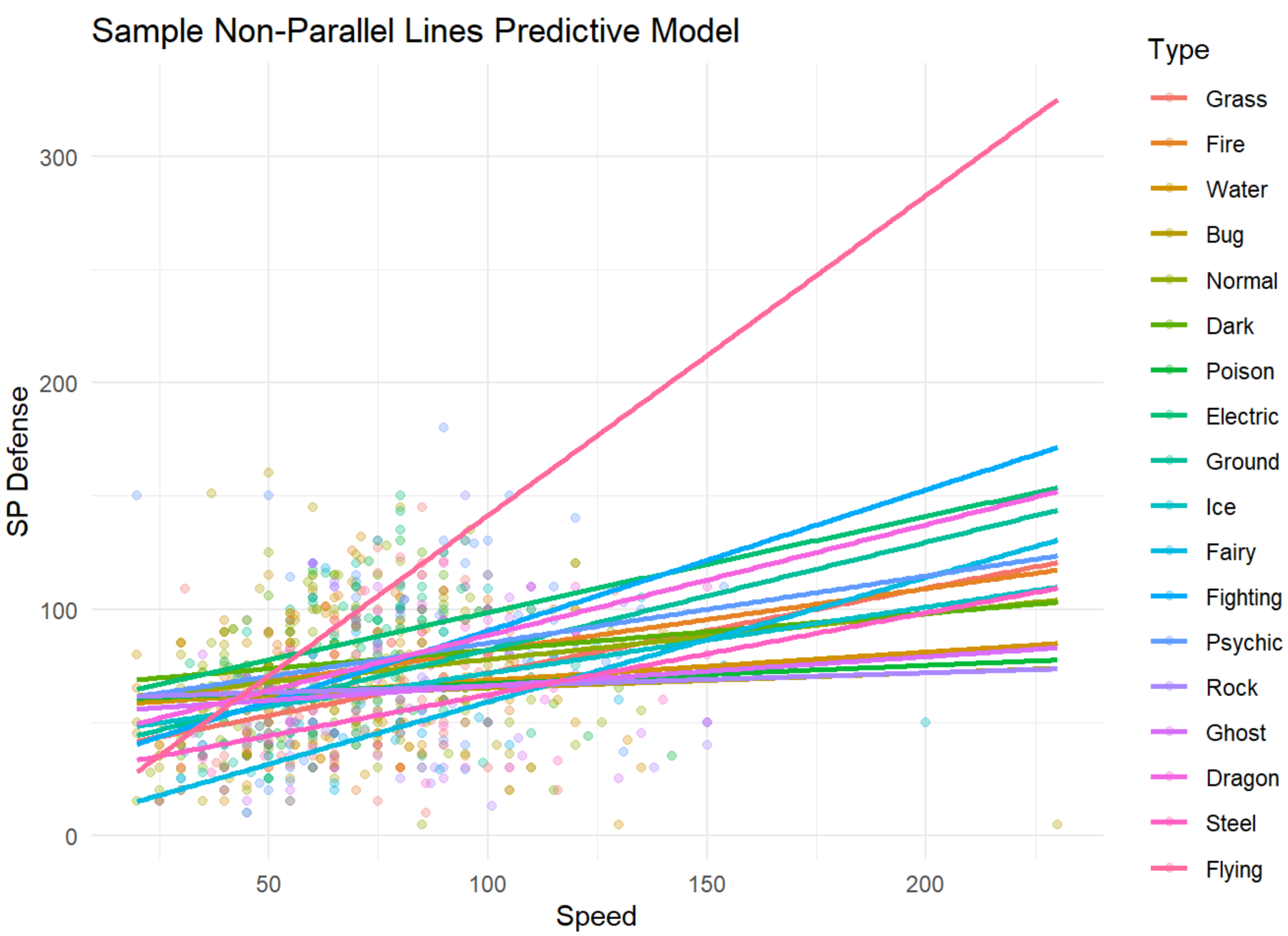
Model #1 Equation

Speed = SP_Attack + Type1 + SP_Attack*Type1

(f) Scatter Plot of Model #2

```
defense_seq <- seq(min(pokemon.data$SP_Defense), max(pokemon.data$SP_Defense), by = 1)

pred_data <- expand.grid(
  SP_Defense = defense_seq,
  Type1 = c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18))

pred_data$Type1 <- factor(pred_data$Type1,
                          levels = c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17
                          labels = c( "Grass",    "Fire", "Water", "Bug", "Normal", "Dark", "P
```

```r
pred_data$predicted_speed <- predict(mod.interaction2, newdata = pred_data)

ggplot() +
  geom_point(data = pokemon.data, aes(x = SP_Defense, y = Speed, color = Type1),
             alpha = 0.3) +
  geom_line(data = pred_data, aes(x = SP_Defense, y = predicted_speed,
                                   color = Type1), size = 1) +
  labs(
    title = "Sample Non-Parallel Lines Predictive Model",
    x = "Speed",
    y = "SP Defense",
    color = "Type"
  ) +
  theme_minimal()
```




```r
coef(mod.interaction2)
```

```
   (Intercept)     SP_Defense      Type1Dark
   58.65121971     0.06637337     6.96255726
   Type1Dragon  Type1Electric     Type1Fairy
  -18.94249740    -2.25561409   -54.51140518
```

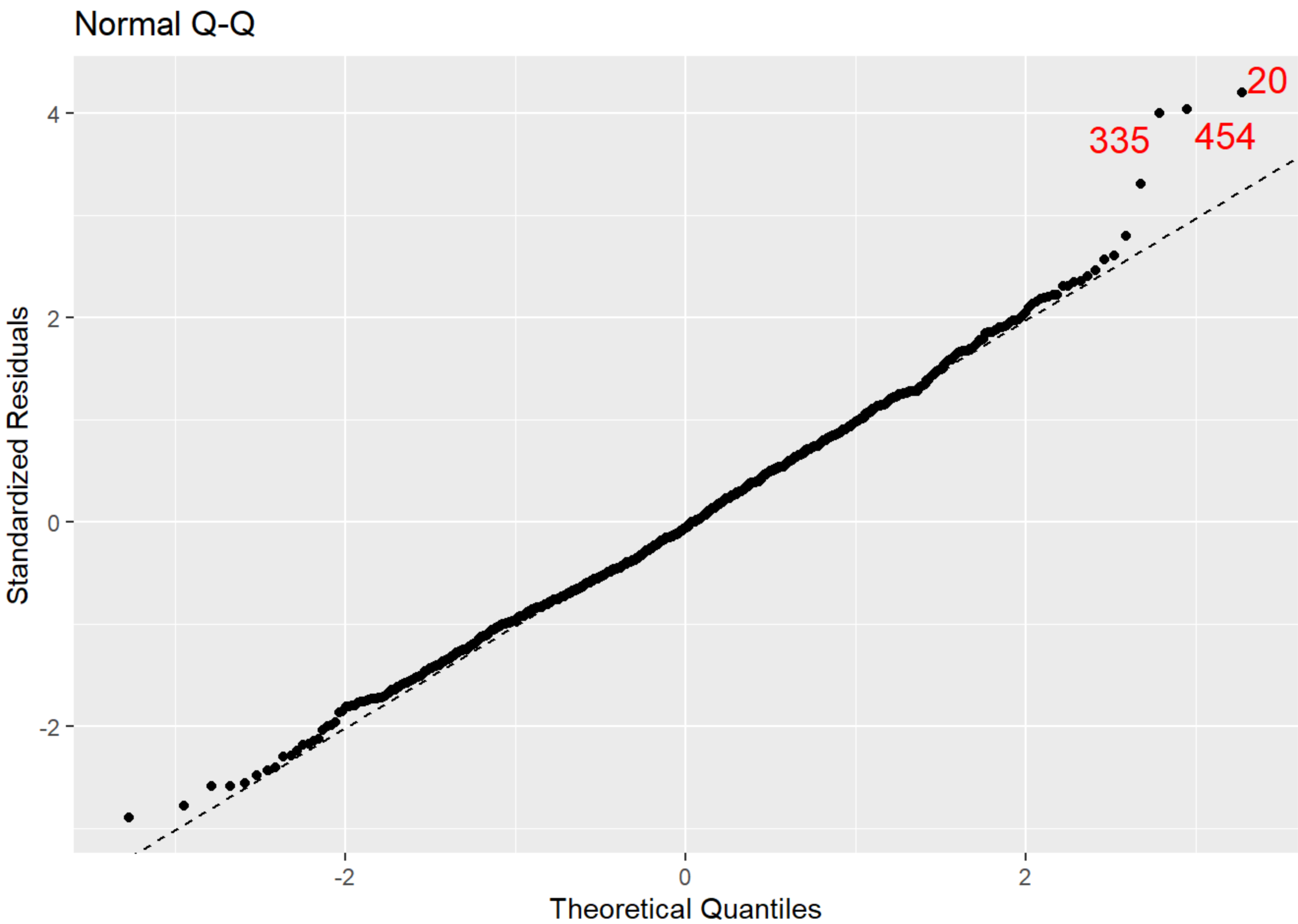
```
         Type1Fighting              Type1Fire            Type1Flying 
          -30.58414998            -4.23365458           -58.63427056 
           Type1Ghost             Type1Grass            Type1Ground 
           -5.30876527           -24.34687003           -23.84434976 
             Type1Ice            Type1Normal            Type1Poison 
          -16.20760418            -0.98717761             0.57315475 
          Type1Psychic              Type1Rock             Type1Steel 
           -3.15856633             1.71320463           -32.57460791 
           Type1Water     SP_Defense:Type1Dark SP_Defense:Type1Dragon 
           -2.52824537             0.09668430             0.42168623 
SP_Defense:Type1Electric    SP_Defense:Type1Fairy SP_Defense:Type1Fighting 
            0.35646129             0.48297072             0.55681441 
   SP_Defense:Type1Fire   SP_Defense:Type1Flying    SP_Defense:Type1Ghost 
            0.20695298             1.34718595             0.06227394 
  SP_Defense:Type1Grass   SP_Defense:Type1Ground      SP_Defense:Type1Ice 
            0.30884774             0.40731319             0.22636459 
 SP_Defense:Type1Normal   SP_Defense:Type1Poison  SP_Defense:Type1Psychic 
            0.13547610             0.01361093             0.22973934 
   SP_Defense:Type1Rock    SP_Defense:Type1Steel    SP_Defense:Type1Water 
           -0.00855145             0.29534327             0.05849221 
```

Fairy Type Equation

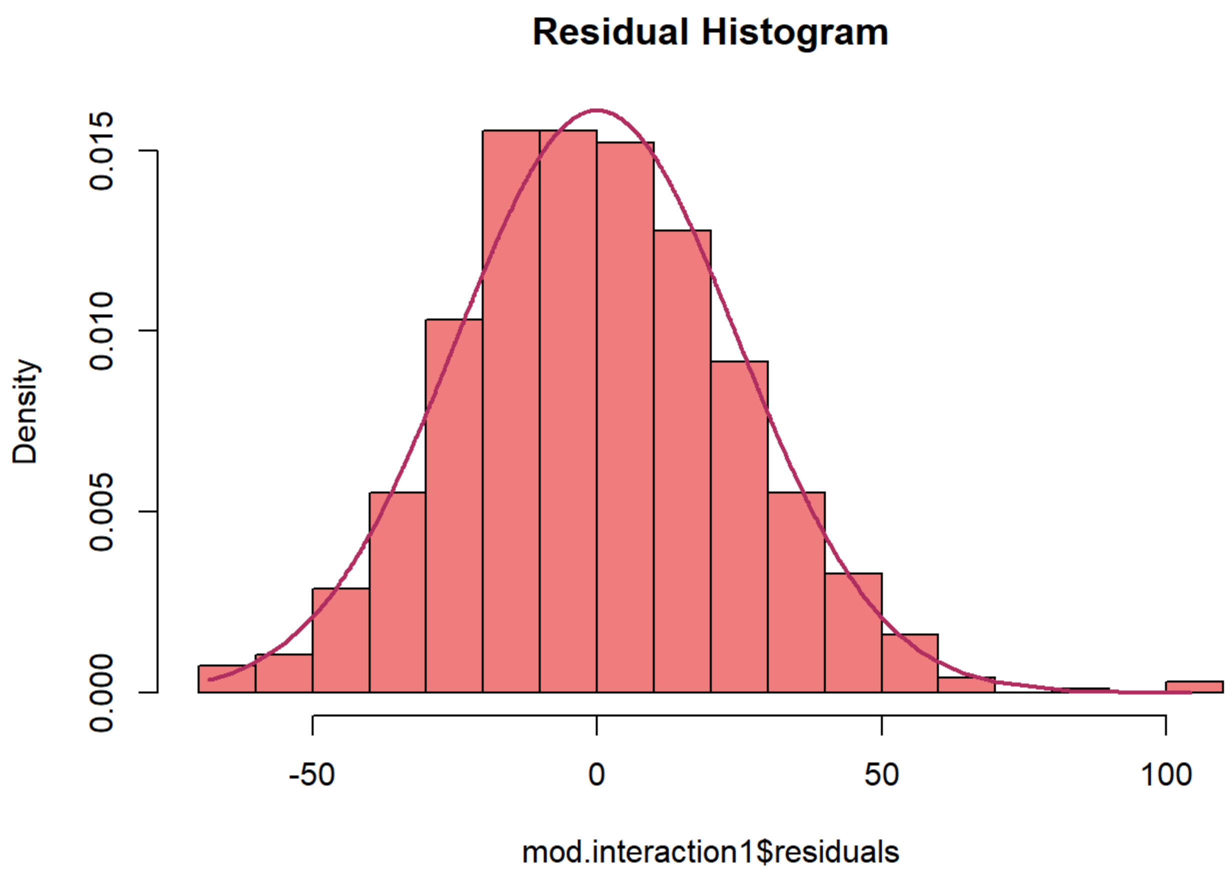
Speed = 58.65 + 0.066xSP_Defense + 0.483xSP_Defense:Type1Fairy

(g) Diagnostic Plots

```r
mplot(mod.interaction1, which = 2)
```




```
hist(mod.interaction1$residuals, prob = TRUE, breaks = 20, col = "lightcoral", main = "Residual Hi

grid = sort(mod.interaction1$residuals)
lines(grid,
      dnorm(grid,
            mean = mean(mod.interaction1$residuals),
            sd = sd(mod.interaction1$residuals)),
      col = 'maroon', lwd = 2)
```



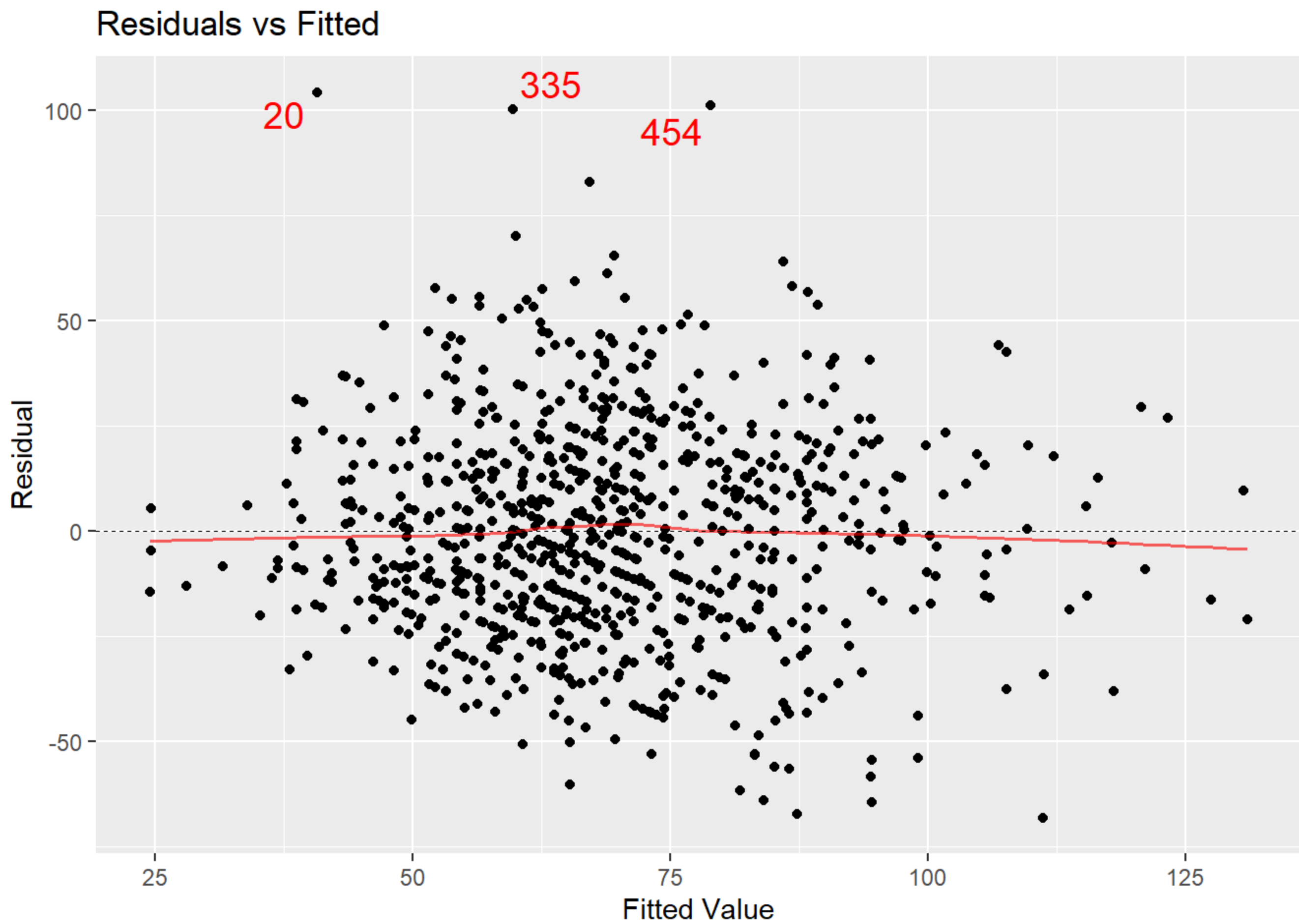
The points follow the line in the Q-Q plot and the histogram follows the density line, meaning that this data meets the assumption of normally distributed residuals.

(h) Mean Zero & Homoscedasticity

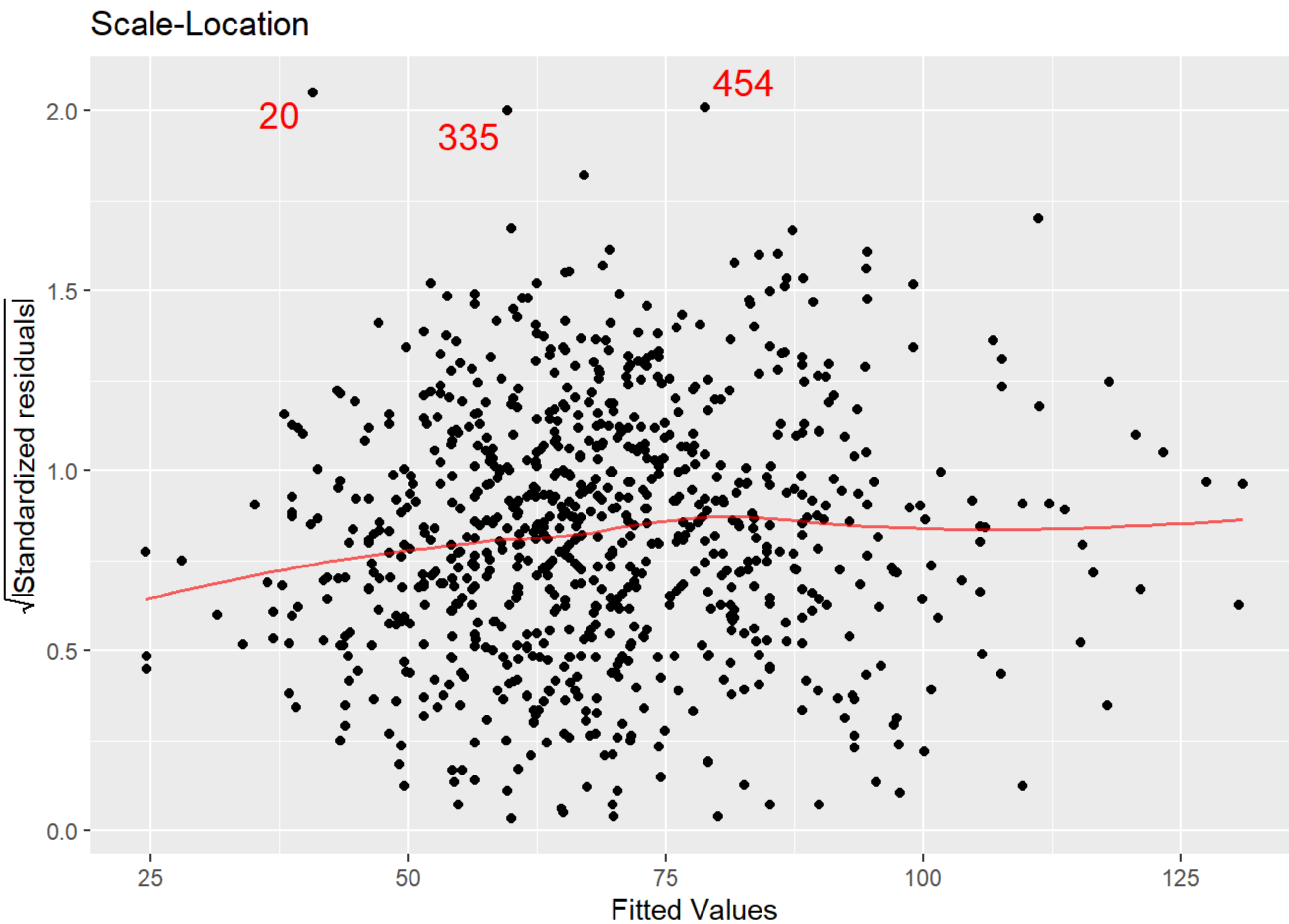
```
mplot(mod.interaction1, which = c(1, 3))
```

```
[[1]]
```

```
`geom_smooth()` using formula = 'y ~ x'
```




```
[[2]]

`geom_smooth()` using formula = 'y ~ x'
```



While the mean line is around zero, the residuals are scattered all over the place (the residuals are all different), meaning that while this data meets the assumption of mean zero, it does not meet the assumption of homoscedasticity.

(i) 95% Confidence Interval (CI)

```
mod.simple <- lm(Speed ~ SP_Defense, data = pokemon.data)
```

```
coef_hp <- coef(mod.simple)["SP_Defense"]
se_hp <- summary(mod.simple)$coefficients["SP_Defense", "Std. Error"]

df <- mod.simple$df.residual
t_critical <- qt(1 - 0.05 / 2, df)

lower_bound <- coef_hp - t_critical * se_hp
upper_bound <- coef_hp + t_critical * se_hp

lower_bound
```

```
SP_Defense
 0.1738758
```

```
upper_bound
```

```
SP_Defense
  0.307469
```

```
boot_fn <- function(data, indices) {
  d <- data[indices, ]
  fit <- lm(Speed ~ SP_Defense, data = d)
  return(coef(fit))
}
```

```
set.seed(123)
boot_results <- boot(data = pokemon.data, statistic = boot_fn, R = 10000)
```

```
boot_results
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = pokemon.data, statistic = boot_fn, R = 10000)


Bootstrap Statistics :
      original        bias    std. error
t1* 51.4434225 -0.0204071449  2.93565814
t2*  0.2406724  0.0004347145  0.04029732
```

Considering that this model does not meet the assumption of homoscedasticity but does meet the assumptions of mean zero and normality, it would be okay to use the R summary to calculate the 95% CI. The bootstrapped method would be okay as well, but it might be better to use R summary since we are meeting multiple assumptions rather than none.

(j) New Estimator

```
pokemon.subsample <- pokemon.data %>%
  filter ( Type1 %in% c("Dragon", "Bug")) %>%
  select (Name, Generation, Type1, HP )

head(pokemon.subsample)
```

```
        Name Generation Type1 HP
1   Caterpie          1   Bug 45
2    Metapod          1   Bug 50
3 Butterfree          1   Bug 60
```

```
4     Weedle          1    Bug 40
5     Kakuna          1    Bug 45
6   Beedrill          1    Bug 65
```

```r
summary(pokemon.subsample)
```

```
     Name             Generation       Type1                  HP
 Length:117         Min.   :1.000   Length:117         Min.   :  1.0
 Class :character   1st Qu.:3.000   Class :character   1st Qu.: 50.0
 Mode  :character   Median :4.000   Mode  :character   Median : 65.0
                    Mean   :3.846                      Mean   : 65.6
                    3rd Qu.:5.000                      3rd Qu.: 76.0
                    Max.   :7.000                      Max.   :216.0
```

```r
## raw difference in mean HP
dragon <- pokemon.subsample[pokemon.subsample$Type1 == "Dragon",]
bug <- pokemon.subsample[pokemon.subsample$Type1 == "Bug",]

mean(dragon$HP)
```

```
[1] 84.08108
```

```r
mean(bug$HP)
```

```
[1] 57.05
```

```r
mean(dragon$HP) - mean(bug$HP)
```

```
[1] 27.03108
```

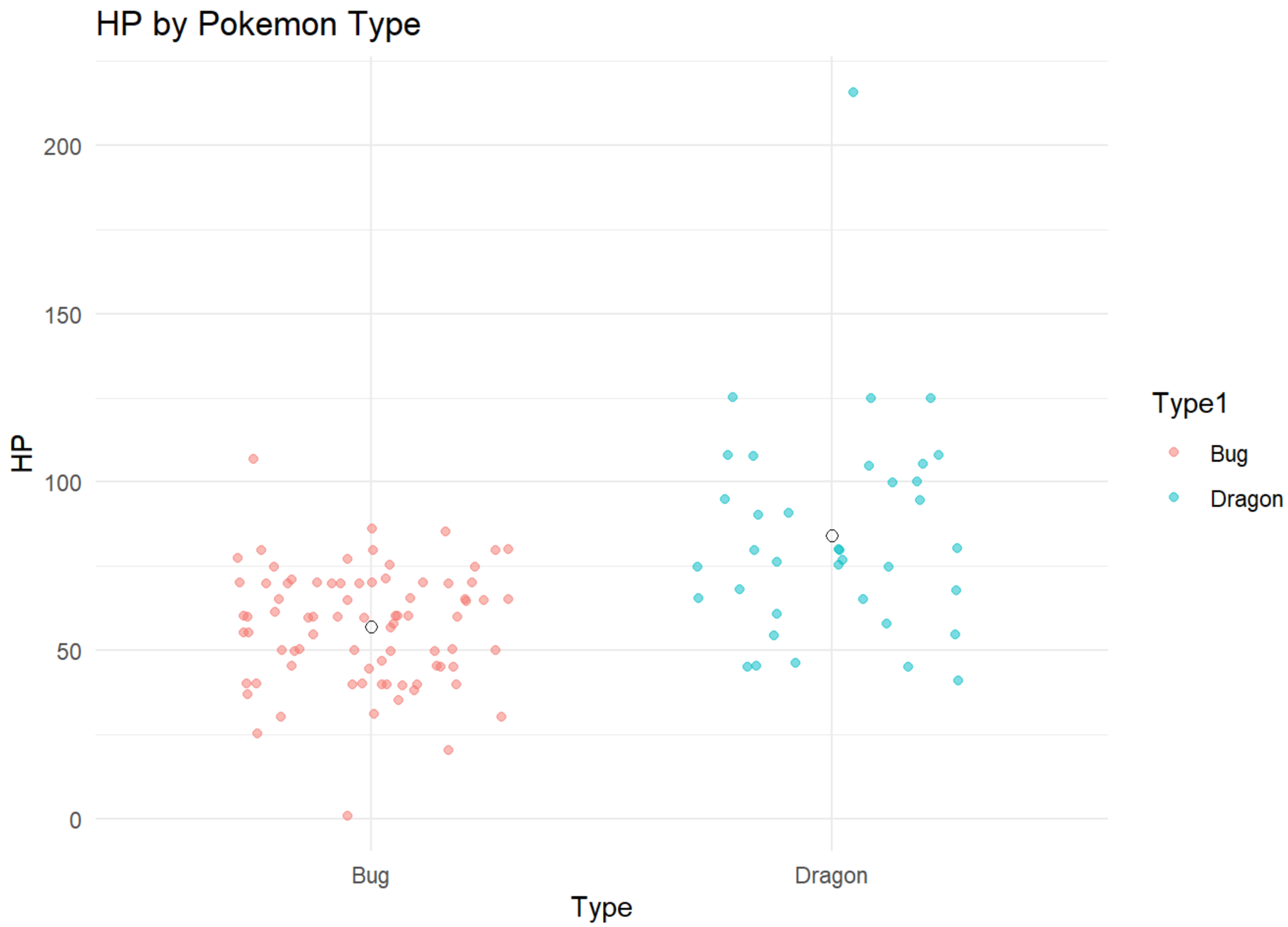
```r
## difference in mean HP with a focus on bug pokemon

bug <- bug %>%
  group_by(Generation) %>%
  summarise(avg_HP_bug = mean(HP, na.rm = TRUE), n_bug = n()) %>%
  ungroup()

dragon <- dragon %>%
  group_by(Generation) %>%
  summarise(avg_HP_dragon = mean(HP, na.rm = TRUE), n_dragon = n()) %>%
  ungroup()

df <- full_join(bug, dragon, by = "Generation")

df <- df %>%
  mutate(mean_diff = coalesce(avg_HP_bug, 0) - coalesce(avg_HP_dragon, 0))

(1/sum(df$n_bug)) * sum(df$n_bug * df$mean_diff)
```

```
[1] -10.47708
```

```r
mod.simple <- lm(HP ~ Type1, data = pokemon.subsample)
summary(mod.simple)
```

```

Call:
lm(formula = HP ~ Type1, data = pokemon.subsample)

Residuals:
   Min     1Q Median     3Q    Max 
-56.05 -16.08   0.95  12.95 131.92 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   57.050      2.594  21.996  < 2e-16 ***
Type1Dragon   27.031      4.612   5.861 4.48e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 23.2 on 115 degrees of freedom
Multiple R-squared:   0.23,	Adjusted R-squared:  0.2233 
F-statistic: 34.35 on 1 and 115 DF,  p-value: 4.481e-08
```

```r
ggplot(pokemon.subsample, aes(x = Type1, y = HP, color = Type1)) +
  geom_jitter(width = 0.3, alpha = 0.5) +
  geom_point(aes(y = fitted(mod.simple)), color = "black", shape = 1, size = 2) +
  theme_minimal() +
  labs(title = "HP by Pokemon Type", x = "Type", y = "HP")
```



I would like to use the second estimator to maximize the difference in HP. As we can see from the two estimators, the second one is the one that features the highest absolute difference. Furthermore, we see that the groups are not balanced, we seem to have more bug pokemon than dragon pokemon. Therefore, we would want to pick the estimator that focuses on the largest subgroup to maximize the difference in HP, which in this case, is the second one.

Exercise #2

```r
library(gmm)
```

```
Warning: package 'gmm' was built under R version 4.3.3
```

```
Loading required package: sandwich
```

```
Warning: package 'sandwich' was built under R version 4.3.3
```

```r
data("nsw")

head(nsw)
```

```
  treat age ed black hisp married nodeg     re75     re78
1     0  23 10     1    0       0     1    0.000     0.00
```

```
2      0  26 12     0    0        0     0    0.000 12383.68
3      0  22  9     1    0        0     1    0.000     0.00
4      0  34  9     1    0        0     1 4368.413 14051.16
5      0  18  9     1    0        0     1    0.000 10740.08
6      0  45 11     1    0        0     1    0.000 11796.47
```

```
summary(nsw)
```

```
     treat              age              ed             black       
 Min.   :0.0000   Min.   :17.00   Min.   : 3.00   Min.   :0.0000  
 1st Qu.:0.0000   1st Qu.:19.00   1st Qu.: 9.00   1st Qu.:1.0000  
 Median :0.0000   Median :23.00   Median :10.00   Median :1.0000  
 Mean   :0.4114   Mean   :24.52   Mean   :10.27   Mean   :0.8006  
 3rd Qu.:1.0000   3rd Qu.:27.00   3rd Qu.:11.00   3rd Qu.:1.0000  
 Max.   :1.0000   Max.   :55.00   Max.   :16.00   Max.   :1.0000  
      hisp           married          nodeg             re75        
 Min.   :0.0000   Min.   :0.000   Min.   :0.0000   Min.   :    0.0  
 1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:1.0000   1st Qu.:    0.0  
 Median :0.0000   Median :0.000   Median :1.0000   Median :  936.3  
 Mean   :0.1053   Mean   :0.162   Mean   :0.7798   Mean   : 3042.9  
 3rd Qu.:0.0000   3rd Qu.:0.000   3rd Qu.:1.0000   3rd Qu.: 3993.2  
 Max.   :1.0000   Max.   :1.000   Max.   :1.0000   Max.   :37431.7  
      re78      
 Min.   :    0  
 1st Qu.:    0  
 Median : 3952  
 Mean   : 5455  
 3rd Qu.: 8772  
 Max.   :60308  
```

(a) Average Treatment Effect (ATE)

```
treat <- nsw %>%
  filter(treat == 1)

untreat <- nsw %>%
  filter(treat == 0)

mean(treat$re78) - mean(untreat$re78)
```

```
[1] 886.3037
```

This is the raw difference in mean earnings between those that were treated and those that were untreated across all groups.

(b) Bootstrapped 95% CI (ATE)

```
set.seed(123)

n <- 10000
ate_boot <- rep(NA, n)

for(i in 1:n){
  sample <-  nsw[sample(1:nrow(nsw), nrow(nsw), replace = T),]

  treat <- sample %>%
  filter(treat == 1)

untreat <- sample %>%
  filter(treat == 0)

  ate_boot[i] <- mean(treat$re78) - mean(untreat$re78)
}
```

```
lower.bound <- quantile(ate_boot, 0.025)
upper.bound <- quantile(ate_boot, 0.975)

print(paste0("The Bootstrapped 95% CI is {", lower.bound,", ",upper.bound,"}"))
```

```
[1] "The Bootstrapped 95% CI is {-56.7367156251803, 1857.96121080787}"
```

If we were to repeat this experiment under the same conditions with a sufficiently large sample size, the average treatment effect would fall somewhere within the estimated interval for about 95 out of 100 trials. In this experiment, expected average treatment effect for our estimator is predicted to be somewhere in between -56.74 and 1857.96.

(c) Least Squares Regression

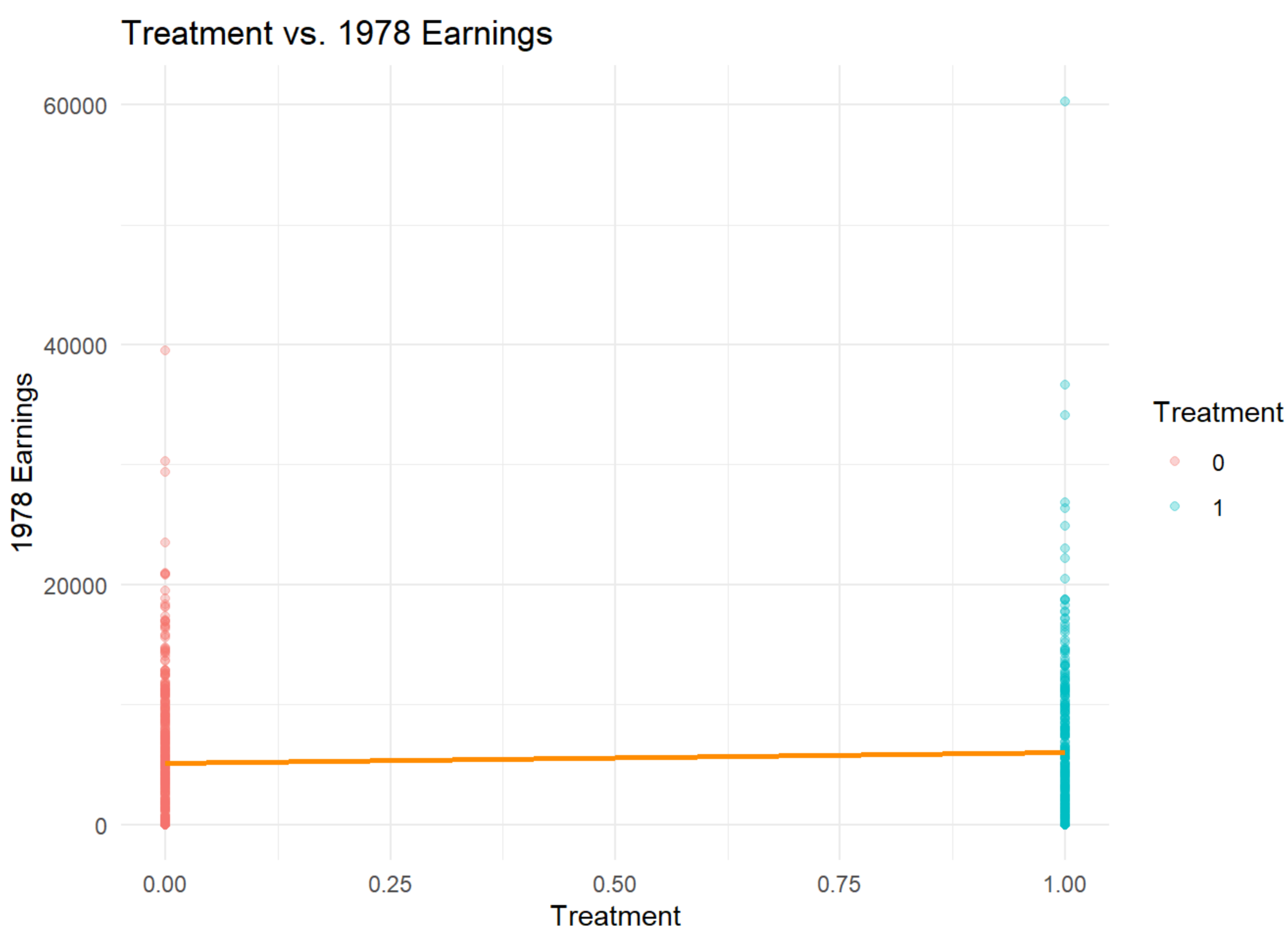
```
model <- lm(re78 ~ treat, data = nsw)
summary(model)
```

```
Call:
lm(formula = re78 ~ treat, data = nsw)

Residuals:
   Min     1Q Median     3Q    Max 
 -5976  -5090  -1519   3361  54332 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   5090.0      302.8  16.811   <2e-16 ***
treat          886.3      472.1   1.877   0.0609 .  
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6242 on 720 degrees of freedom
Multiple R-squared:  0.004872,  Adjusted R-squared:  0.003489
F-statistic: 3.525 on 1 and 720 DF,  p-value: 0.06086
```

```r
predicted_re78 <- data.frame(re78 =  predict(model), treat = nsw$treat)
```

```r
ggplot(nsw, aes(x = treat, y = re78)) +
  geom_point(aes(x = treat, y = re78, color = factor(treat)),
             alpha = 0.3) +
geom_line(data = predicted_re78,
            aes(x = treat, y = re78),color='darkorange', lwd= 1) +
  labs(
    title = "Treatment vs. 1978 Earnings",
    x = "Treatment",
    y = "1978 Earnings",
    color = "Treatment") +
  theme_minimal()
```



The intercept describes the average baseline salary for both treatment and control groups at the start of the study. Finally, the treat coefficient represents the average difference in salary between treated and untreated individuals. In other words, the average treatment effect. Here, the average treatment effect is estimated to be ~ $886.

(d) Discussion

I would say that adding certain factors would deffinitely change the average treatment effect. For instance, the variable of marriage will add an additional source of income. Respondents would likely submit a higher value for the household salary if they were living with their spouse. Therefore, it's important to consider covariates when estimating the average treatment effect.

```
model <- lm(re78 ~ treat + married, data = nsw)
summary(model)
```

```
Call:
lm(formula = re78 ~ treat + married, data = nsw)

Residuals:
   Min     1Q Median     3Q    Max 
 -6510  -4989  -1511   3362  54440 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   4989.0      318.7  15.656   <2e-16 ***
treat          879.4      472.1   1.863   0.0629 .  
married        641.2      630.5   1.017   0.3095    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6242 on 719 degrees of freedom
Multiple R-squared:  0.006301,	Adjusted R-squared:  0.003537 
F-statistic:  2.28 on 2 and 719 DF,  p-value: 0.1031
```

(e) Difference in Means (CATE)

```
df <- nsw %>%
  group_by(nodeg) %>%
  summarise(
    N_Total = n(),
    N_Treated = sum(treat == 1),
    N_Control = sum(treat == 0),
    Mean_Treated = mean(re78[treat == 1]),
    Mean_Control = mean(re78[treat == 0]),
    CATE = Mean_Treated - Mean_Control
  ) %>%
  ungroup()
```

```r
df
```

```
# A tibble: 2 × 7
  nodeg N_Total N_Treated N_Control Mean_Treated Mean_Control  CATE
  <dbl>   <int>     <int>     <int>        <dbl>        <dbl> <dbl>
1     0     159        80        79        6977.        5919. 1059.
2     1     563       217       346        5607.        4901.  706.
```

These estimates are not causally identified because we are not conditioning treatment on whether an individual has a high school diploma or not.

(f) Standardized CATE

```r
df <- df %>%
  mutate(CATE_standard = (N_Total/sum(N_Total))*CATE)

sum(df$CATE_standard)
```

```
[1] 784.0365
```

This value is less than my estimated ATE. This is likely because by conditioning on high school diploma you remove potential discrepancies by education. Therefore, the difference in the treatment is not so stark.

(g) Least Squares Regression

```r
model <- lm(re78 ~ treat + nodeg + treat*nodeg, data = nsw)
summary(model)
```

```

Call:
lm(formula = re78 ~ treat + nodeg + treat * nodeg, data = nsw)

Residuals:
   Min     1Q Median     3Q    Max 
 -6977  -4901  -1405   3287  54701 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   5918.6      701.0   8.443   <2e-16 ***
treat         1058.7      988.3   1.071    0.284    
nodeg        -1017.7      777.0  -1.310    0.191    
treat:nodeg   -352.2     1126.0  -0.313    0.755    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6231 on 718 degrees of freedom
```

```
Multiple R-squared:  0.01113,	Adjusted R-squared:  0.006994 
F-statistic: 2.693 on 3 and 718 DF,  p-value: 0.04521
```

Using this least squares regression, the intercept gives the mean earnings for the control group, otherwise known as the baseline. Additionally, the treat coefficient specifies how much change there is in earnings for each unit of treatment. In this case, this represented the CATE for those without a high school diploma. Additionally, the nodeg coefficient represents the change in earnings for each unit of nodeg. In other words, the earnings decrease by ~ $1017 on average for those with a high school diploma. Finally, the interaction coefficient, represents the relationship between treatment and nodeg, where those that have a high school diploma are less likely to receive treatment. The difference between the treat coefficient and the interaction variable represents the CATE for those without a high school diploma.

(h) Bootstrapped Estimated Sampling Distribution

```r
boot_fn <- function(data, indices) {
  d <- data[indices, ]
  fit <- lm(re78 ~ treat + nodeg + treat*nodeg, data = d)
  return(coef(fit))
}
```

```r
set.seed(123)
boot_results <- boot(data = nsw, statistic = boot_fn, R = 10000)
```

```r
boot_results
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = nsw, statistic = boot_fn, R = 10000)


Bootstrap Statistics :
      original     bias    std. error
t1*  5918.6075  -8.167754    691.9999
t2*  1058.7049  17.062629   1106.7927
t3* -1017.7389   8.547283    755.0760
t4*  -352.2391 -15.372697   1230.9072
```

```r
boot.ci(boot_results, type = "perc", index = 2)
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 10000 bootstrap replicates

CALL : 
boot.ci(boot.out = boot_results, type = "perc", index = 2)
```

```
Intervals :
Level     Percentile
95%   (-1003,  3312 )
Calculations and Intervals on Original Scale
```

```r
boot.ci(boot_results, type = "perc", index = 4)
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 10000 bootstrap replicates

CALL :
boot.ci(boot.out = boot_results, type = "perc", index = 4)

Intervals :
Level     Percentile
95%   (-2844.3,  1977.1 )
Calculations and Intervals on Original Scale
```

If we were to repeat this experiment under the same conditions with a sufficiently large sample size, the conditional average treatment effect would fall somewhere within the estimated interval for about 95 out of 100 trials. In this experiment, expected conditional average treatment effect for those without a high school diploma is predicted to be somewhere in between -1003 and 3312. Also, the expected conditional average treatment effect for this experiment for those with a high school diploma is expected to fall somewhere between -1841 and 1335 (if you were to subtract the values from both CI above).