Project Task 4 - An Investigation into Factors Influencing Pokémon Weight

Jeffrey Yu – 20932312

Julia Shen – 21028444

Xiaoda Zhang - 21028837

Xinyi Nie 20941485

July 24, 2025

Executive Summary

As a group, we set out to explore two hypotheses: the relationship between a Pokémon's defense and its weight, and the relationship between its height and weight. After examining these individual relationships, we extended our analysis by constructing a multivariable linear regression model to evaluate whether a combination of features — including base stats (attack, defense, hp, speed, etc.) and other characteristics (capture rate, base total, etc.) — could improve the model's ability to explain a Pokémon's weight. We determined that a multivariable linear model would be the most effective approach for examining how different Pokémon stats interact with weight. To develop this model, we first performed exploratory analysis by examining the distributions of our selected variables and their correlations. We then cleaned the dataset to prepare it for automatic model selection. Using both backward and stepwise selection based on the Bayesian Information Criterion (BIC), we generated an initial model. However, this model failed several key diagnostic checks. As a result, we pivoted to a manual selection process informed by our prior knowledge of Pokémon. This approach produced a multivariable linear model that satisfied all diagnostic criteria. Through this process, we found that height and defense are indeed significant predictors of weight.

Motivation

In the Pokemon world, a creature's weight is more than trivia. Weight affects catch rates, move mechanics, shipping costs for merchandise, and even the realism of augmented-reality models. Unfortunately, official weight values are revealed only after a new generation launches.

If we can predict weight from base stats that are published on reveal day, our studio will gain three advantages:

- Size physical products and digital assets earlier, which shortens production timelines
- Identify stat combinations that look physically implausible during game-balance reviews
- Deliver data-driven marketing copy, for example pointing out that high defense hints at a tank-like body

Preliminary work showed that legendary Pokémon cluster at high stats and that weight seems to rise with defense and height. We therefore focus on two questions:

Do higher defense scores signal heavier Pokémon Do taller Pokémon also weigh more

The report that follows combines results from Task 2 and Task 3, corrects remaining diagnostic issues, and presents a log-scale regression that explains about sixty-two percent of weight variation using only four readily available stats. Our aim is to show that a modest investment in statistical modeling will let stakeholders act with confidence long before official weight data appear.

About The Data Set

The data set pokemon_subset.csv is a subset of data obtained from Kaggle. It records information on 802 Pokemon from all seven generations.The data set contains 801 observations with 20 variables. Our hypothesis is that the defense variable is positively correlated to the weight_kg variable. Our second hypothesis is that height_m and weight_kg are also positively correlated. Defense is a discrete explanatory variable whereas weight_kg is the continuous response variable.

Exploratory Data Analysis

```
df_base <- read.csv("pokemon_subset.csv")
df_base = na.omit(df_base)
summary(df_base$weight_kg)
```

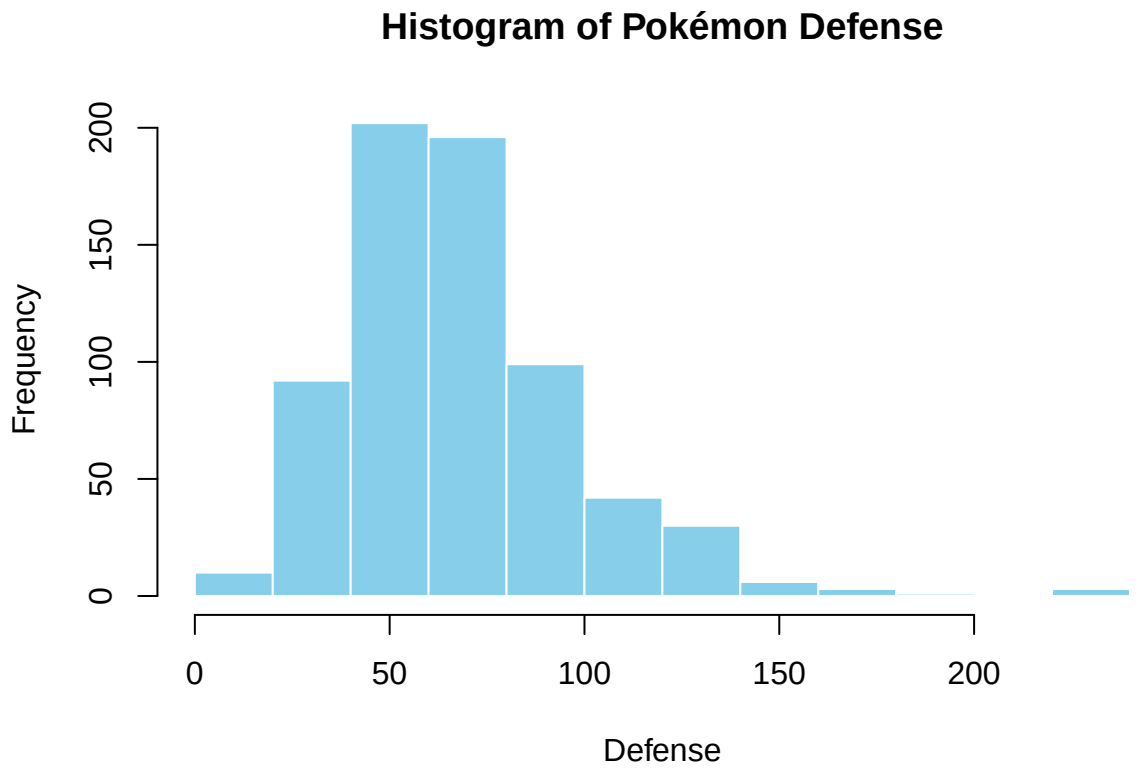
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.100   8.575  24.450  46.328  55.000 920.000
```

```r
var(df_base$weight_kg, na.rm = TRUE)
```

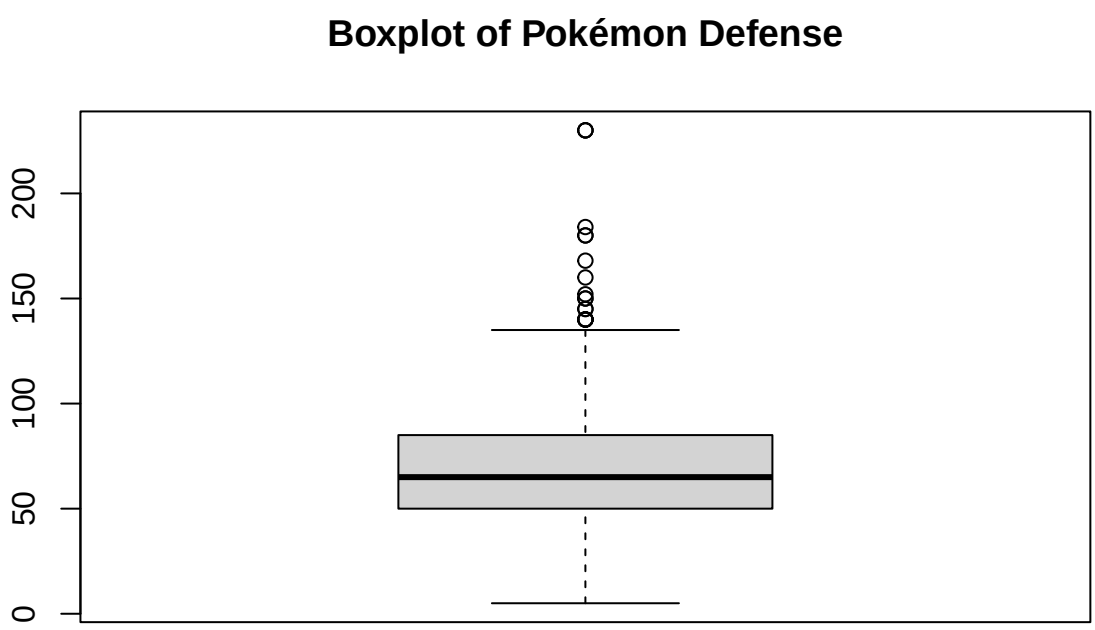
```
## [1] 5087.887
```

The variable weight_kg ranges from 0.10 to 999.90 with a variance of 11958.46. The following histogram shows the distribution of weight_kg.

```r
hist(df_base$defense,
     main = "Histogram of Pokémon Defense",
     xlab = "Defense",
     col = "skyblue",
     border = "white")
```

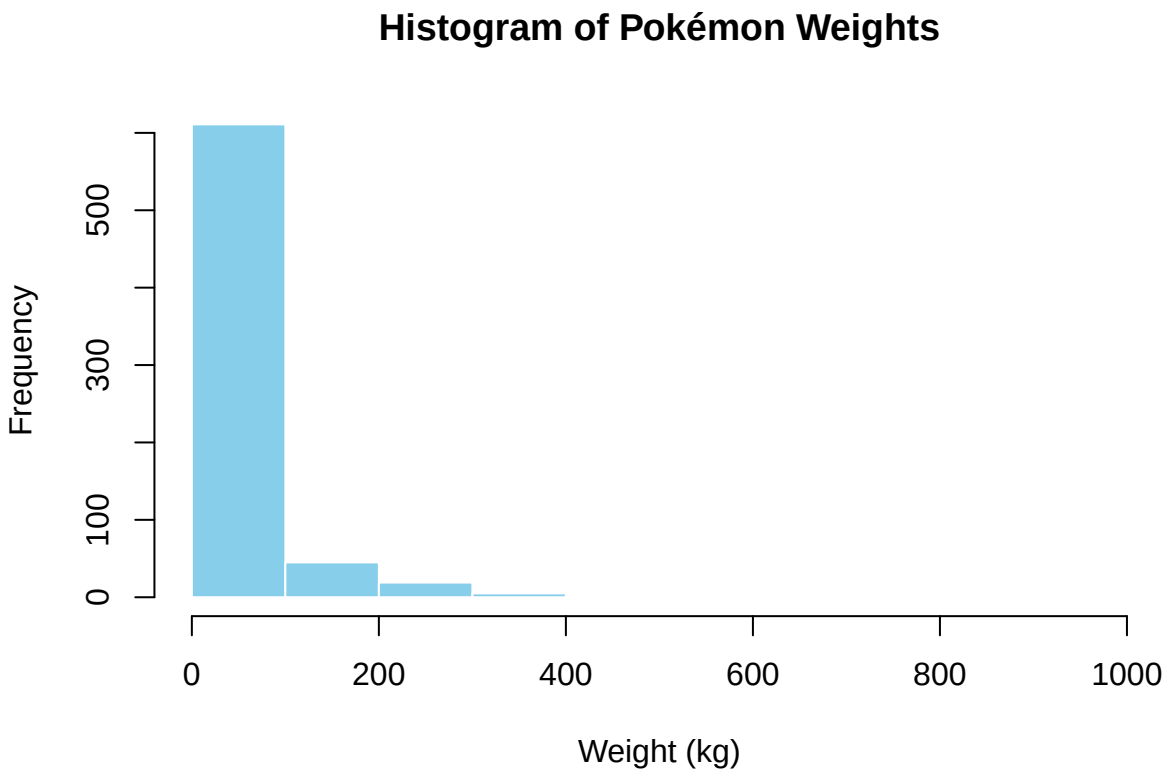



```r
boxplot(df_base$defense,
        main = "Boxplot of Pokémon Defense")
```

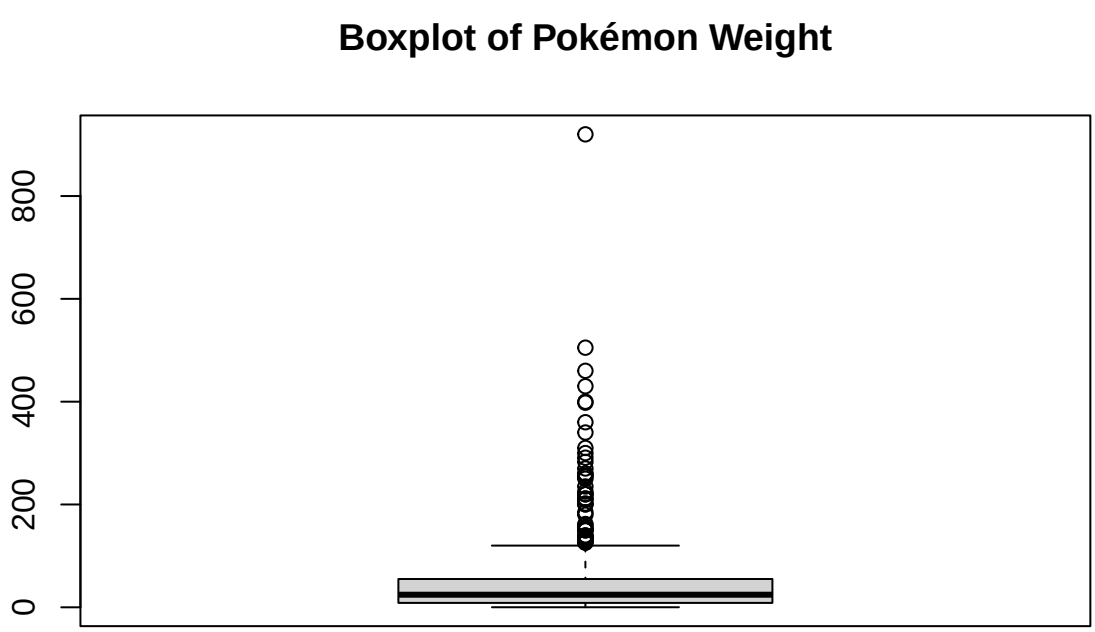


The histogram of defense is noticeably right-skewed. Most defense values fall between 50 and 90, with a mode around 60–70. A tail tapers off past 100, and a few extreme outliers reach up to 230. The box plot of defense shows a median of 70 and an IQR from 50 to 90. The lower whisker drops to about 5, and the upper whisker sits near 150. Several outliers above 150 confirm the right skew shown in the histogram.

```
hist(df_base$weight_kg,
     main = "Histogram of Pokémon Weights",
     xlab = "Weight (kg)",
     col = "skyblue",
     border = "white")
```

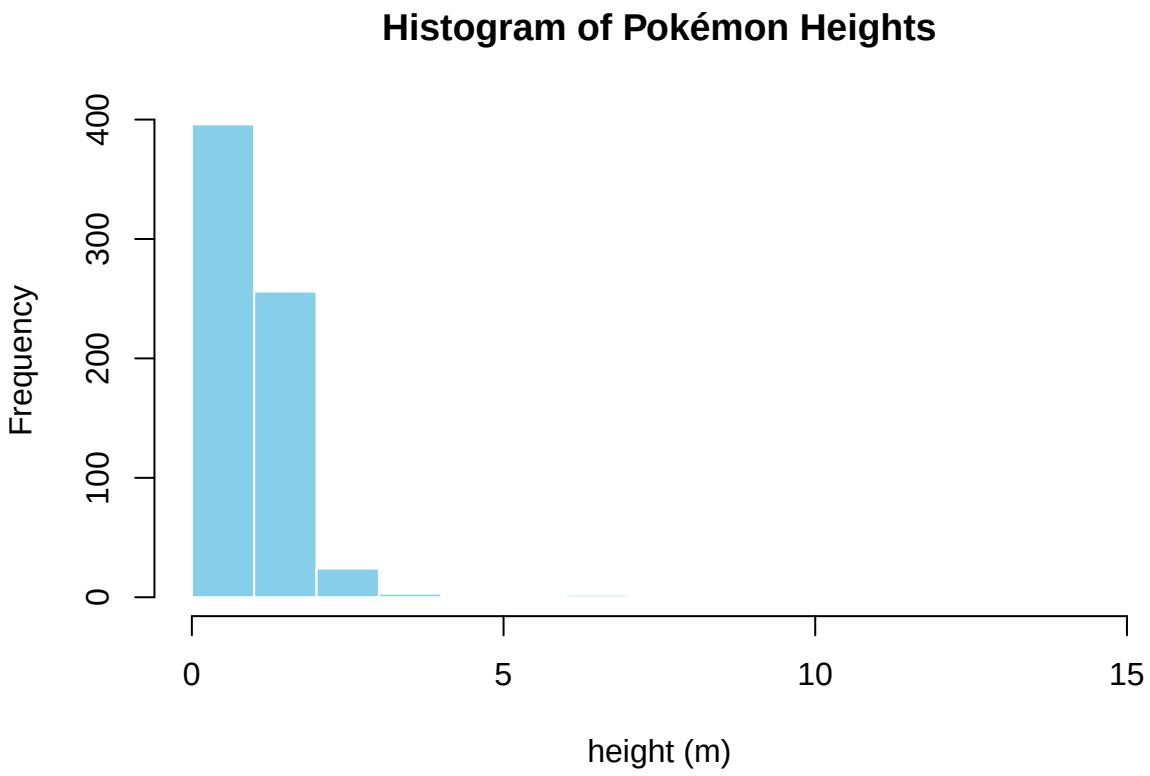



```
boxplot(df_base$weight_kg,
        main = "Boxplot of Pokémon Weight")
```

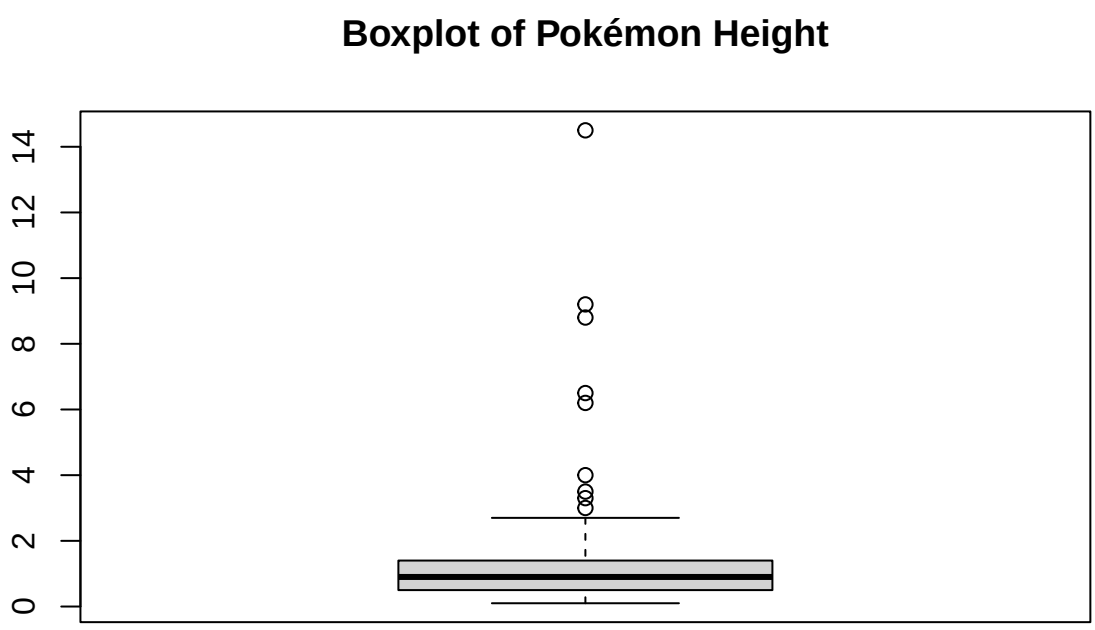


The histogram of height is heavily right-skewed. Most heights fall between 1 m and 2 m, with a mode at 1 m. A long tail stretches from 3 m up to 14 m, though those very tall Pokémon are uncommon. The boxplot of weight shows a median of 27 kg and an IQR from 9 kg to 65 kg. The lower whisker drops to about 0 kg, and the upper whisker reaches roughly 150 kg. A handful of very heavy outliers extend from 150 kg up to 999 kg, confirming the pronounced right skew.

```r
hist(df_base$height_m,
     main = "Histogram of Pokémon Heights",
     xlab = "height (m)",
     col = "skyblue",
     border = "white")
```

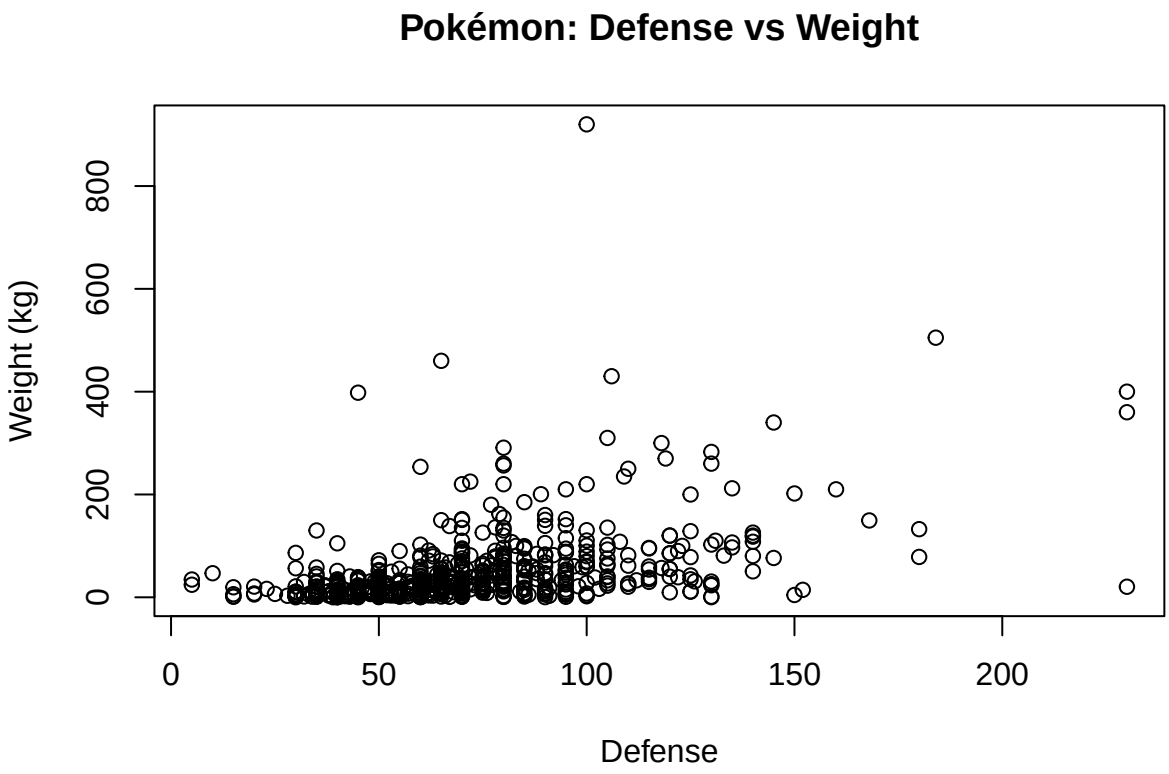



```
boxplot(df_base$height_m,
        main = "Boxplot of Pokémon Height")
```



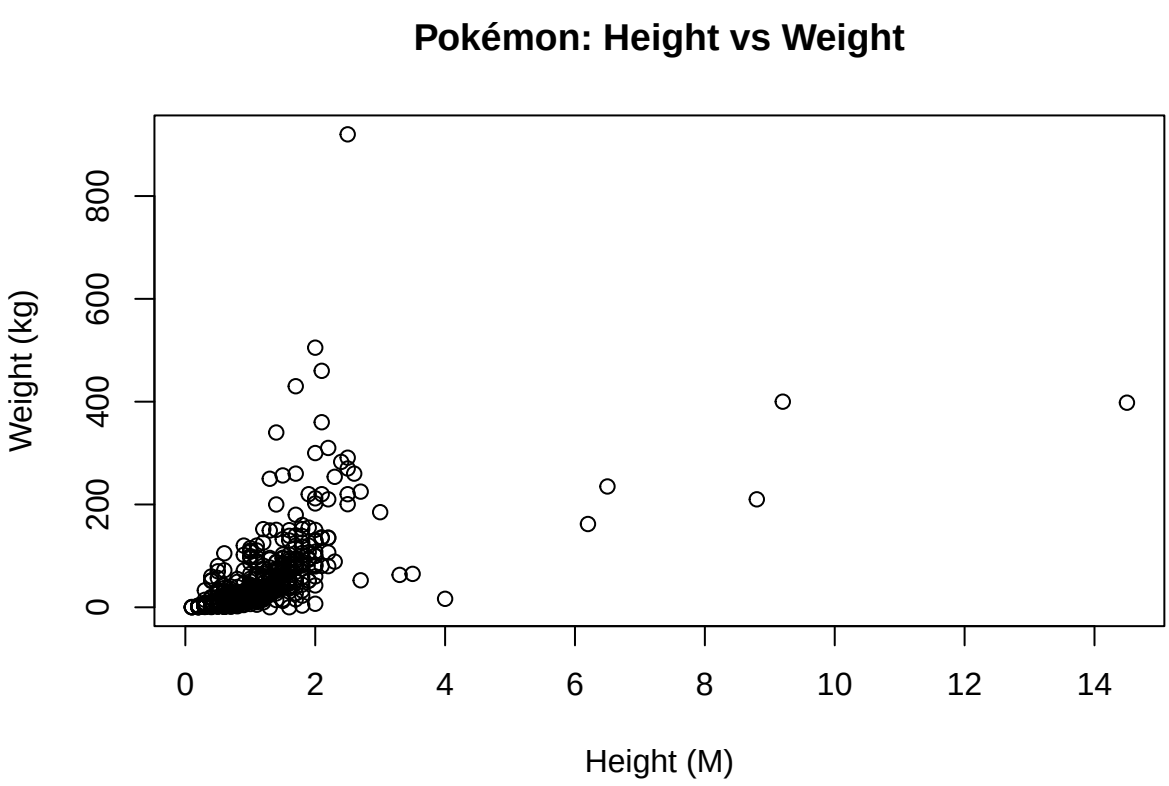
The histogram of height is heavily right-skewed. Most heights fall between 1 m and 2 m, with a mode at 1 m. A long tail stretches from 3 m up to 14 m, though those very tall Pokémon are uncommon. The box plot of height shows a median of 1 m and an IQR from 1 m to 2 m. The lower whisker drops to 0 m, and the upper whisker reaches about 3 m. A few outliers extend beyond 3 m, up to 14 m, confirming the heavy right skew.

```
plot(df_base$defense, df_base$weight_kg,
     xlab = "Defense",
     ylab = "Weight (kg)",
     main = "Pokémon: Defense vs Weight")
```



Whilst the correlation may appear to be weak there does appear to be a positive correlation between defense and weight_kg as such we will pursue fitting a linear model to it.

```
plot(df_base$height_m, df_base$weight_kg,
     xlab = "Height (M)",
     ylab = "Weight (kg)",
     main = "Pokémon: Height vs Weight")
```



The scatter plot of height vs. weight shows a clear upward pattern, i.e., taller Pokémon generally weigh more. Most of the points form a loose upward cloud, which suggests a positive relationship between the two variables. While there is quite a bit of spread (i.e., some Pokémon with the same height have very different weights), the overall trend still seems to go up. A few very tall or heavy outliers stand out, but they do not take away from the main pattern. All of these points seem to support our hypothesis that taller Pokémon tend to be heavier.

Method

To investigate which factors influence a Pokémon's weight, we began by building a linear regression model, as our response variable weight_kg is continuous. Given the number of potential explanatory variables in the dataset (e.g., base stats, biological traits), we initially chose to use automated model selection methods.

Automatic selection offers a systematic and objective way to identify influential variables, especially useful when working with multiple candidates. It helps guard against human bias and reduces the chance of over fitting by applying formal model comparison criteria.

We applied two model selection techniques — backward selection and stepwise selection, both based on the Bayesian Information Criterion (BIC). The purpose of using BIC (instead of AIC) is that BIC penalizes model complexity more heavily, aligning better with our goal of finding meaningful relationships, not just predictive power.

To prepare for automatic selection, we cleaned the data set by:

- Removing irrelevant variables like name and pokedex_number.
- Excluding categorical variables (type1, type2, generation, is_legendary) for initial simplicity.
- Fixing malformed data in the capture_rate variable.

- Converting character variables to numeric where appropriate.
- Dropping rows with missing values.

```
### PART A: CLEANING THE DATA SET

# Read in the data, treat blank strings as NA
poke_og <- read.csv("pokemon_subset.csv",
                    na.strings = c("", "NA"));

# We will not consider name, pokedex_number, and categorical variables (type1,
# type2, generation, is_legendary) for the potential explanatory variables.
poke <- subset(poke_og, select = -c(name,
                                    pokedex_number,
                                    type1,
                                    type2,
                                    generation,
                                    is_legendary))

# Removing capture_rate "bad" data
unique(poke$capture_rate)
```

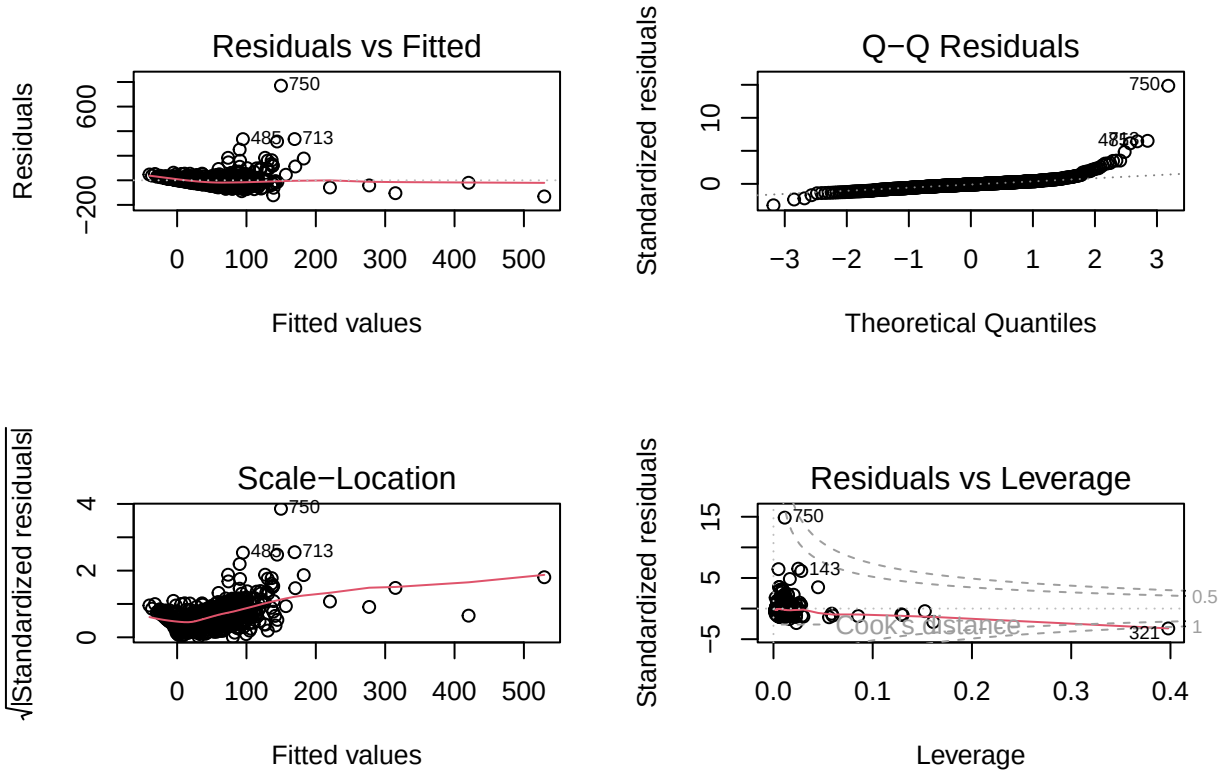
```
##  [1] "45"                        "255"
##  [3] "120"                       "127"
##  [5] "90"                        "190"
##  [7] "75"                        "235"
##  [9] "150"                       "25"
## [11] "170"                       "50"
## [13] "200"                       "100"
## [15] "180"                       "60"
## [17] "225"                       "30"
## [19] "35"                        "3"
## [21] "65"                        "70"
## [23] "125"                       "205"
## [25] "155"                       "145"
## [27] "130"                       "140"
## [29] "15"                        "220"
## [31] "160"                       "80"
## [33] "55"                        "30 (Meteorite)255 (Core)"
```

```
poke <- poke[poke$capture_rate != "30 (Meteorite)255 (Core)", ]
poke$capture_rate <- as.numeric(poke$capture_rate)

# Drop any remaining NA rows
poke_df <- na.omit(poke)

# OPTIONAL: Check structure
str(poke_df)
```

```
## 'data.frame':    684 obs. of  14 variables:
##  $ attack           : int  49 62 100 52 64 104 48 63 103 30 ...
##  $ base_egg_steps   : int  5120 5120 5120 5120 5120 5120 5120 5120 5120 3840 ...
##  $ base_happiness   : int  70 70 70 70 70 70 70 70 70 70 ...
```

```
##  $ base_total       : int  318 405 625 309 405 634 314 405 630 195 ...
##  $ capture_rate     : num  45 45 45 45 45 45 45 45 45 255 ...
##  $ defense          : int  49 63 123 43 58 78 65 80 120 35 ...
##  $ experience_growth: int  1059860 1059860 1059860 1059860 1059860 1059860 1059860 1059860 1059860 1
##  $ height_m         : num  0.7 1 2 0.6 1.1 1.7 0.5 1 1.6 0.3 ...
##  $ hp               : int  45 60 80 39 58 78 44 59 79 45 ...
##  $ percentage_male  : num  88.1 88.1 88.1 88.1 88.1 88.1 88.1 88.1 88.1 50 ...
##  $ sp_attack        : int  65 80 122 60 80 159 50 65 135 20 ...
##  $ sp_defense       : int  65 80 120 50 65 115 64 80 115 20 ...
##  $ speed            : int  45 60 80 65 80 100 43 58 78 45 ...
##  $ weight_kg        : num  6.9 13 100 8.5 19 90.5 9 22.5 85.5 2.9 ...
##  - attr(*, "na.action")= 'omit' Named int [1:116] 19 20 26 27 28 37 38 50 51 52 ...
##   ..- attr(*, "names")= chr [1:116] "19" "20" "26" "27" ...
```

```r
### PART B: DEFINING THE INITIAL (FULL) MODEL

# Define your "intercept-only" null model
m0 <- lm(weight_kg ~ 1, data = poke_df)

# Define your full model with all remaining numeric predictors
# (weight_kg is the response; adjust this vector if you add/drop any columns)
full_predictors <- c("height_m", "capture_rate", "percentage_male",
                     "attack", "defense", "hp", "speed",
                     "sp_attack", "sp_defense",
                     "base_total", "base_happiness",
                     "base_egg_steps", "experience_growth")

model_full <- lm(
  as.formula(paste("weight_kg ~", paste(full_predictors, collapse = " + "))),
  data = poke_df
)


### PART C: BACKWARD SELECTION USING BIC
back_bic <- step(object    = model_full,
                 scope     = list(lower = m0, upper = model_full),
                 k         = log(nrow(poke_df)), # BIC
                 direction = "backward",         # Backward selection
                 trace     = FALSE)


### PART D: STEPWISE SELECTION USING BIC
step_bic <- step(object    = model_full,
                 scope     = list(lower = m0, upper = model_full),
                 k         = log(nrow(poke_df)), # BIC
                 direction = "both",             # Stepwise selection
                 trace     = FALSE)

par(mfrow = c(2, 2), ask = FALSE)
plot(back_bic)
```




```
plot(step_bic)
```

We conducted several diagnostic tests on this model to verify the assumptions of linear regression: Residual vs Fitted Plot: Displayed a curved pattern, suggesting a violation of the linearity assumption. Scale-Location Plot: Showed increasing residual spread, indicating heteroscedasticity (non-constant variance). Q-Q Plot: Deviations from the diagonal in the tails suggested non-normal residuals. Cook's Distance: Identified a small number of influential outliers. These issues indicated that some assumptions of the linear regression model were not fully satisfied.

As automatic selection did not yield a suitable linear model, we decided to try manual selection as We already have prior knowledge of the data set through different Pokemon media. We determined that the following would be a good starting point.

```
model1 = lm(weight_kg ~ defense + speed + base_egg_steps, data = df_base)
```

However, the R^2 value was much too low. To fix this, we decided to add more explanatory variables. This lead insignificant variables being added to the model. As such we utilized p-values and LRT tests to come to a simpler model with a high R^2 value.

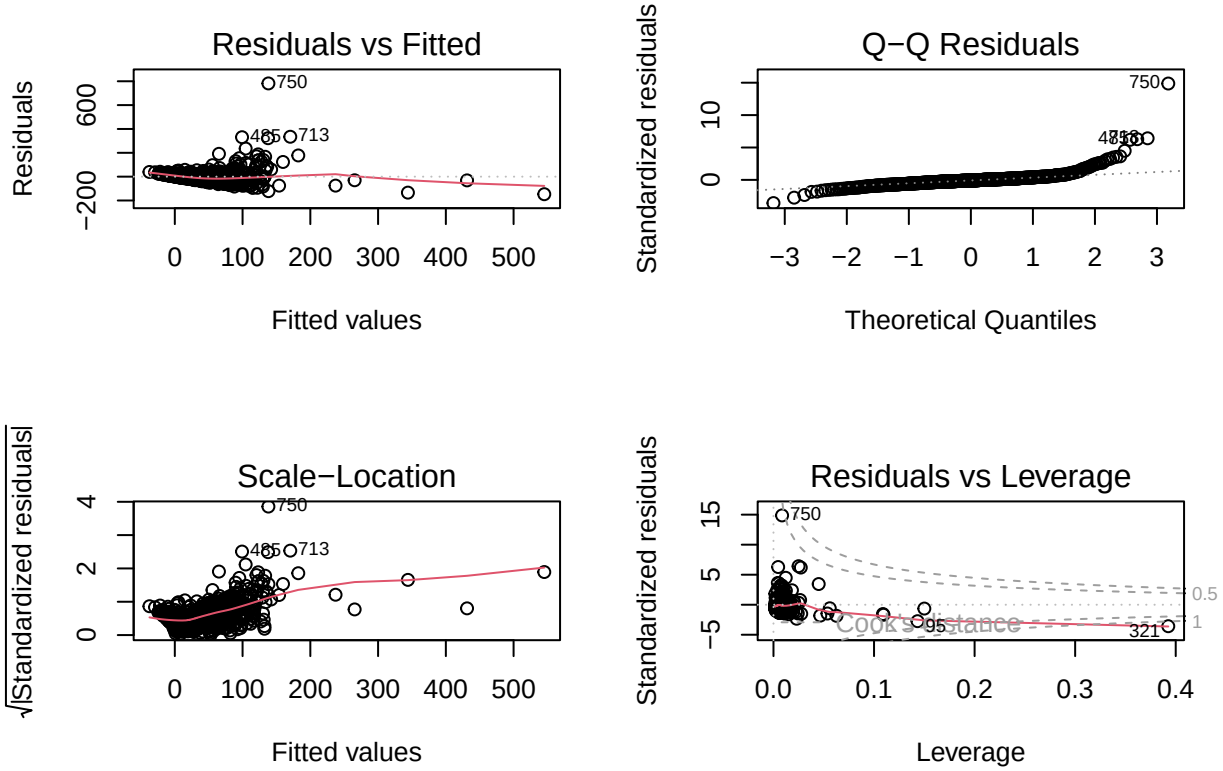
```
library(car)
```

```
## Loading required package: carData
```

```
model2 = lm(weight_kg ~ defense + speed + height_m + hp, data = df_base)
vif(model2)
```

```
##  defense    speed height_m       hp
## 1.173252 1.045589 1.363337 1.201298
```

```
par(mfrow = c(2,2))
plot(model2)
```

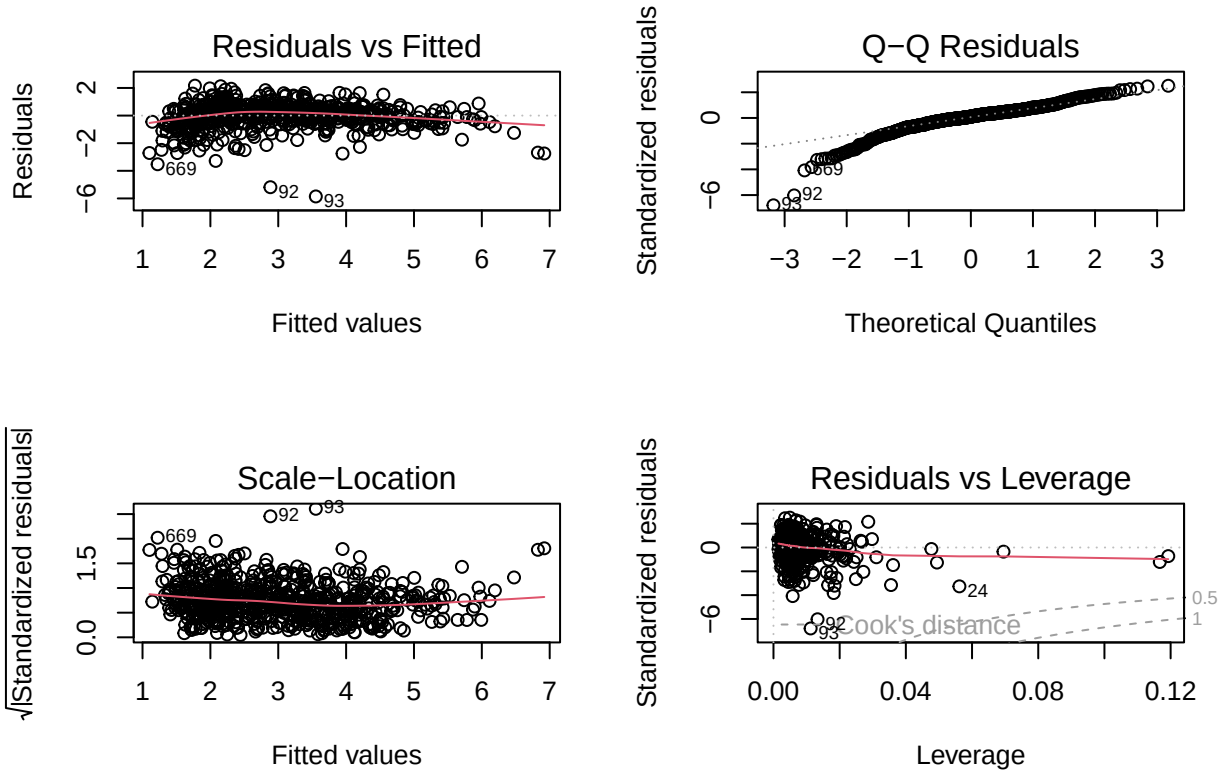


The vif of the model also suggested no multicollinearity issues. However the diagnostic plots showed that there was an issue with skewness. To solve for this we log transformed weight_kg and removed identified outliers. This improved the R^2 value and the diagnostic plots.

```
drop_ids = c(321, 208, 932, 935, 95, 130, 350, 148)
df_base = subset(df_base, !(pokedex_number %in% drop_ids))
model_final = lm(log(weight_kg) ~ defense + speed + height_m + hp, data = df_base)
vif(model_final)
```

```
##  defense    speed height_m       hp
## 1.305223 1.136917 1.839819 1.382061
```

```
par(mfrow = c(2,2))
plot(model_final)
```



The vif shows that there are no multicollinearity issues. The diagnostic plots also show that the we have acheived a reasonable linear model. The Residuals vs Fitted graph shows that the residuals are mostly randomly scattered around 0. This means there is no strong signs of non-linearity. The QQ plot shows that the residuals mostly follow the diagonal line save minor deviations in the lower and upper tails. That means the residuals are reasonably normally distributed. The scale location graph has a relatively flat trend line. The spread of the residuals remain mostly constant. As such there are no serious issues with heteroscedasticity. From the Residuals vs Leverage graph, we can see that most observations are low leverage. All observations are within the 0.5 threshold. This suggests that the model is stable and no very influential outliers remain. All these points suggest that model_final is a useful model.

Results and Interpretation

The regression analysis revealed several significant predictors of a Pokémon's log-transformed weight. Holding other variables constant, higher defense scores were associated with significantly greater weight, Beta = 0.00725, t(673) = 5.57, p < .001. Additionally, height in meters was also a strong positive predictor of weight, Beta = 1.6016, t(673) = 19.76, p < .001. For every one-meter increase in height, the expected log weight increases by approximately 1.60 units.

Hit points (hp) also had a significant positive relationship with log(weight_kg), Beta = 0.00763, t(673) = 4.81, p < .001. In contrast, speed showed a marginally significant negative association with weight, Beta = -0.00232, t(673) = -1.81, p = .071.

The overall model was statistically significant, F(4, 673) = 272.7, p < .001, with an R^2 of 0.619, indicating that approximately 61.9% of the variance in log-transformed weight was explained by the model. The residual standard error was 0.867 on 673 degrees of freedom.

We are 95% confident that, on average, a one-unit increase in defense corresponds to an increase in log(weight_kg) between 0.00470 and 0.00981, while a one-meter increase in height corresponds to an increase between approximately 1.44 and 1.76 in log(weight_kg).

Limitations

Snapshot data. We work with one record per Pokémon, so we can spot patterns but cannot prove cause-and-effect. Missing labels. Type, generation and other categories are not in the equation, which helps explain why 38% of the weight variation is still unaccounted for.Extreme cases dropped. Eight massive legendaries were excluded as outliers, so the model likely under-predicts the very heaviest Pokémon. Numbers can change. Pokédex heights and weights get rounded, corrected or buffed in new games; regular retraining will be needed.

Appendix

```
summary(back_bic)
```

```
##
## Call:
## lm(formula = weight_kg ~ height_m + attack + defense + hp + speed,
##     data = poke_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -131.39  -21.78   -4.71   11.40  770.25
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -54.15808    8.26486  -6.553 1.12e-10 ***
## height_m     32.87398    2.53382  12.974  < 2e-16 ***
## attack        0.34653    0.08207   4.222 2.75e-05 ***
## defense       0.45939    0.07925   5.796 1.04e-08 ***
## hp            0.45087    0.09051   4.982 8.01e-07 ***
## speed        -0.36054    0.07838  -4.600 5.04e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 52.16 on 678 degrees of freedom
## Multiple R-squared:  0.4692, Adjusted R-squared:  0.4653
## F-statistic: 119.9 on 5 and 678 DF,  p-value: < 2.2e-16
```

```
summary(step_bic)
```

```
##
## Call:
## lm(formula = weight_kg ~ height_m + attack + defense + hp + speed,
##     data = poke_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -131.39  -21.78   -4.71   11.40  770.25
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -54.15808    8.26486  -6.553 1.12e-10 ***
## height_m     32.87398    2.53382  12.974  < 2e-16 ***
## attack        0.34653    0.08207   4.222 2.75e-05 ***
## defense       0.45939    0.07925   5.796 1.04e-08 ***
```

```
## hp            0.45087    0.09051   4.982 8.01e-07 ***
## speed        -0.36054    0.07838  -4.600 5.04e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 52.16 on 678 degrees of freedom
## Multiple R-squared:  0.4692, Adjusted R-squared:  0.4653
## F-statistic: 119.9 on 5 and 678 DF,  p-value: < 2.2e-16
```

```
summary(model1)
```

```
##
## Call:
## lm(formula = weight_kg ~ defense + speed + base_egg_steps, data = df_base)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -194.89  -25.21   -9.59    9.36  842.44
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -35.715557   9.073863  -3.936 9.13e-05 ***
## defense          1.059371   0.082259  12.879  < 2e-16 ***
## speed           -0.003853   0.088820  -0.043   0.9654
## base_egg_steps   0.001459   0.000869   1.679   0.0937 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 63.56 on 680 degrees of freedom
## Multiple R-squared:  0.2096, Adjusted R-squared:  0.2061
## F-statistic: 60.09 on 3 and 680 DF,  p-value: < 2.2e-16
```

```
summary(model2)
```

```
##
## Call:
## lm(formula = weight_kg ~ defense + speed + height_m + hp, data = df_base)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -147.27  -20.17   -5.14   10.75  781.84
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -52.62093    8.35852  -6.295 5.50e-10 ***
## defense       0.59827    0.07299   8.196 1.24e-15 ***
## speed        -0.24787    0.07460  -3.323  0.00094 ***
## height_m     33.99710    2.55085  13.328  < 2e-16 ***
## hp            0.54639    0.08871   6.159 1.25e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 52.8 on 679 degrees of freedom
```

```
## Multiple R-squared:  0.4553, Adjusted R-squared:  0.4521
## F-statistic: 141.9 on 4 and 679 DF,  p-value: < 2.2e-16
```

```
summary(model_final)
```

```
##
## Call:
## lm(formula = log(weight_kg) ~ defense + speed + height_m + hp,
##     data = df_base)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.8562 -0.3506  0.0944  0.4669  2.1652
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.541179   0.139508   3.879 0.000115 ***
## defense      0.007254   0.001302   5.571 3.67e-08 ***
## speed       -0.002316   0.001279  -1.811 0.070536 .
## height_m     1.601644   0.081055  19.760  < 2e-16 ***
## hp           0.007632   0.001587   4.811 1.86e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8665 on 673 degrees of freedom
## Multiple R-squared:  0.6185, Adjusted R-squared:  0.6162
## F-statistic: 272.7 on 4 and 673 DF,  p-value: < 2.2e-16
```