Predicting a Pokémon's Strength Using Variables Other Than Base Stats

Team Name: Nate Krall, Daniel Cohen, Brian Kim

2023-11-15

Introduction and Data:

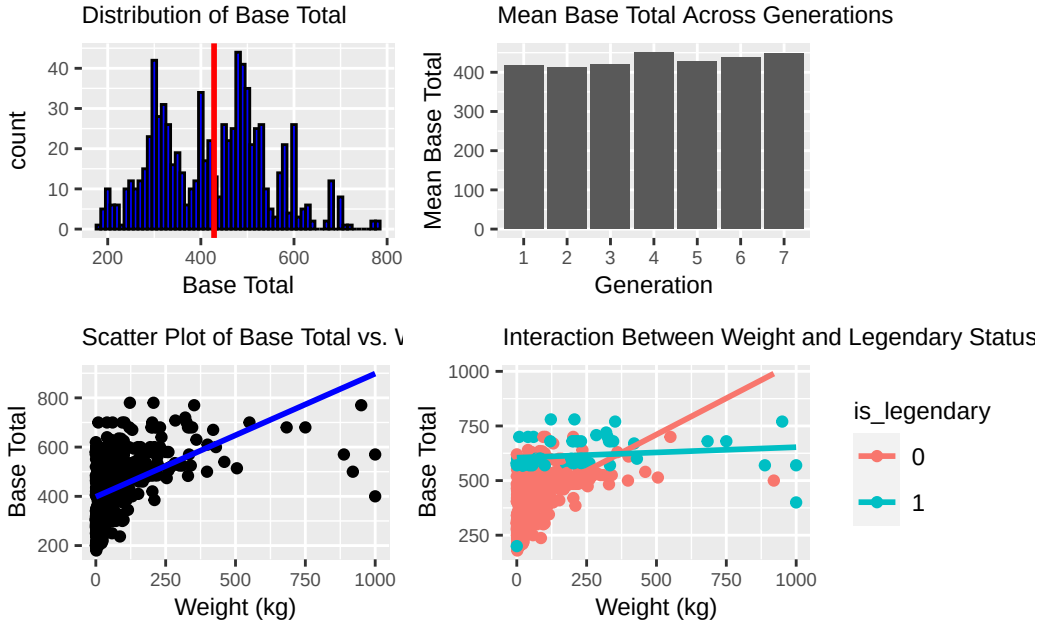
The expansive world of Pokémon, at its core, is a children's game. However, a deeper look at the numbers and statistics the game is built from reveals several intricate relationships among the different pokémons' characteristics. Each creature has its own specific set of base statistics, colloquially referred to as "base stats," including attack, special attack, defense, special defense, and speed, which indicate that pokémon's battle prowess. Summing these stats yields a pokémon's total base stats, which is the best measure of a pokémon's overall strength when all pokémon are put at an even playing field – common knowledge for any pokémon fan. We are interested in measuring a pokémon's strength without using base stats as predictors, giving us insight on how strong the relationships among pokémon's different characteristics actually are. Thus, we are looking to answer the following research question: **Can we predict a Pokémon's Base Stat Total from other variables?** In other words, we are analyzing how well variables such as the pokémon's type, capture rate, growth rate, generation, height, weight, base happiness, weaknesses, and if the pokemon is legendary or not can predict a pokémon's total base stats. We hypothesize that a multiple linear regression model including some formation of these predictor variables will be a somewhat strong predictor for base_total – thinking about the game, stronger pokémon would seem to have certain values for these predictor variables when compared to weaker ones: for example, legendary pokémon tend to be stronger in battle than non legendary pokémon, so we might expect is_legendary to be a useful predictor for base_total, for example.

We retrieved the dataset from kaggle.com, a large data science online community, and the dataset is called "The Complete Pokemon Dataset" created by Rounak Banik in 2017. The dataset was retrieved via web scraper from the website serebii.net, an all-in-one, reliable data hub for all things pokemon in 2017. Since it was formed in 2017, the dataset does not include pokémon from more recent games, but still includes a whopping 801 pokémon, meaning the dataset has 801 observations, one for each pokémon.

The dataset contains 23 variables, taken directly from the kaggle website for the dataset, explanations of which can be viewed in our data dictionary. For our analysis, we will be focusing on these specific variables:

- ADD VARIABLES HERE

As you'll notice, the dataset splits the base stats of each pokemon into the individual stats, but we only need to know about the base_total variable, which is included in the csv file. Each variable describes the pokémon at hand in a different way. Some, like is_legendary, may prove to be extremely important in our regression model, while with others, like name and Japanese name, we can remove them from consideration as they are simply unique identifiers.




```
# A tibble: 1 x 6
  minimum    q1 median  mean    q3 maximum
    <dbl> <dbl>  <dbl> <dbl> <dbl>   <dbl>
1     180   320    435  428.   505     780
```

p1: Distribution of Base total: We can see from the distribution of the base total, that is seems to be roughly trimodal with three peaks (one around 300, 400, and 475). The base total points vary from about 180 to 780 with a mean at around 428.4. The median is 435 which is higher than the mean by a little bit which could mean that there are more extreme low base total Pokemon bringing the mean down a little bit. The IQR is 185 (505 - 320) and looking at the data there might be a potential outlier around 780 base total.

p2: Mean Base Total Across Generations: This is the graph showing the relationship between different generation Pokemon and their mean base total. We can see from this graph that the mean base total of generation 4 Pokemon were the highest but overall we can't see any distinct relationships between the generation of the Pokemon and their mean base total. One can maybe say there could be a very slight positive linear relationship between the generation of the Pokemon and their mean base total as we obsere that as generation increases the mean base total tends to increase slightly.

p3: Scatter Plot of Base Total vs Weight: From the scatter plot I can see that there is no apparent correlation between the weight and the base total. There may be a very low positive correlation between these two variables but most of the points are focused around when the weight of the Pokemon is less than 100kg so it's hard to tell the relationship. There seems to be a few outliers near 900-1000kg.

p4: Interaction Between Weight and Legendary Status: This graph is showing the relationship between the weight of the Pokemon and the base total for legendary and non legendary Pokemon. The red line is for non legendary Pokemons (is_legendary = 0) and the blue line is for legendary Pokemon (is_legendary = 1). Something interesting we can note is that legendary Pokemon all tend to have a much higher and consistent base total for all weights. Across all weights most base total for legendary Pokemons stay between 550- 700. Among the legendary Pokemon there doesn't seem to be any correlation between the weight and the base total. Among the non-legendary Pokemon however, there seems to be a very slight positive correlation between the weight of the Pokemon and the base total. As the weight of the non-legendary Pokemon goes up, the Base total seems to go up slightly although with the cluster of points under 100 kg Pokemon, it is hard to conclude, and regardless, the correlation does not appear to be strong between these variables.

Note that from this graph we can tell that Legendary Pokemon appear to tend to have much higher base stat totals than non-legendary Pokemon on average, which is something that will definitely impact our multiple linear regression model.

Methodology:

> **!** Important
>
> Before you submit, make sure your code chunks are turned off with `echo: false` and there are no warnings or messages with `warning: false` and `message: false` in the YAML.

We first split our data into 75% training and 25% testing data.

In our recipe, we do the following:

1. We update the role of "name" to be an ID.
2. We remove irrelevant, non-predicting information like abilities, classification, and Japanese name.
3. We remove all against_* variables, since the predictive power of these variables are already represented by their types, so we reduce redundancy and collinearity by doing so.
4. We finally create 1-hot encoding dummy variables for type1 and type2 so they can be used in our model.

```
== Workflow ====================================================================
Preprocessor: Recipe
Model: linear_reg()

-- Preprocessor ----------------------------------------------------------------
6 Recipe Steps

* step_rm()
* step_rm()
* step_rm()
* step_rm()
* step_rm()
* step_dummy()

-- Model ---------------------------------------------------------------------
Linear Regression Model Specification (regression)

Computational engine: lm


# A tibble: 72 x 5
   term              estimate std.error statistic  p.value
   <chr>                <dbl>     <dbl>     <dbl>    <dbl>
 1 (Intercept)      309.       48.3         6.41  3.81e-10
 2 base_egg_steps     0.00263   0.00212     1.24  2.15e- 1
 3 base_happiness     0.234     0.291       0.803 4.23e- 1
 4 capture_rate120  -56.8      30.4        -1.87  6.22e- 2
 5 capture_rate127   14.7      44.1         0.333 7.40e- 1
 6 capture_rate130 -216.       74.1        -2.91  3.77e- 3
 7 capture_rate140  -28.3      48.3        -0.585 5.59e- 1
 8 capture_rate145 -105.       72.3        -1.45  1.47e- 1
 9 capture_rate150  -46.6      44.3        -1.05  2.93e- 1
```

```
10 capture_rate160   54.7      70.8         0.773 4.40e- 1
# i 62 more rows
```

BRIAN:

1. fix recipe and model so that capture rate is taken as a continuous predictor
2. make sure is_legendary is working properly as a categorical predictor
3. make sure type1 and type2 are functioning properly
4. fix anything else with the model that's weird that i'm not noticing
5. keep checking by running the linear model and looking for a reasonable result, once you get there, move on

then,

compare different models with Adj R^2, AIC, BIC, etc. to choose the best one

POTENTIAL KIND-OF UNIMPORTANT VARIABLES:

weight or height?

generation?

check collinearity between generation and pokedex number?

do other general stuff to select the best model (for now, this is just a draft)

I'll pick up where you leave off when I'm back

1.