pokeML

By Lucas Jamar

Overview

- The code is written in R.
- Requires 5 CRAN packages:
  - install.packages(c("data.table","ggplot2","stringr","MLmetrics","caret","rmarkdown"))
- And the R version of lightGBM:
  - https://lightgbm.readthedocs.io/en/latest/Installation-Guide.html
- Data folder has the same layout as a Kedro data folder.
- Code is divided into 5 scripts:
  - The functioning codes are located in “code/”
  - The rendered PDF versions of the codes are located in “docs/”

Overview

- The rendered PDF versions are fully documented with plots and links to sources.

Create Features

Lucas Jamar

2020-05-28

Feature engineering is centred around the use of the damage calculation formula. Since we are interested in the maximum damage a player can inflict on, we start by taking the weather and type influences to be equal to the maximum weather and type influence available for each player. Then, by approximating the power of all moves to 80, we can calculate the damage and special damage each player can inflict on the other. The damage values are then rounded down and used to approximate how many rounds a player would need to defeat the other. The approximate number of rounds of the battle is equal to the smallest number of rounds one player would need to defeat the other. By ignoring all priority moves, we can consider that the faster player always moves first. Therefore, if the winning player is also the faster one, we assume that the defeated player will have one less chance to strike. From here, we can approximate the battle result as well as the portion of ramining HP of player 1. Since our approximations correlate well with the target variable (R2 = 0.95), we redefine the target variable to be the difference between the true remaining HP of player 1 and our approximation. This, in turn, should allow our ML model to focus on the points that are difficult to explain with our approximation. Finally, we calculate some ratios to create features for our model.

Read in combined data

```
library(data.table)

dt <- data.table::fread("data/03_primary/pokemon.csv")
```

Context

- Presence of type Fairy
  - Game is of generation 6 or more
    - Critical Attack Probability is no longer dependent on speed.
    - Special Attack is no longer dependent on type.

Source: https://bulbapedia.bulbagarden.net/wiki/Type/Type_chart

Merging the data

01merge_data.R

- The data is slightly asymmetric:
  - Player 2 wins more often
  - But player 1 has a higher average absolute Battle Result

Player 1 Wins	Count	Mean(absolute(BattleResult))
True	1359053	397.8265
False	1360110	397.7180

01merge_data.R

- BattleResults data was made symmetric for both players:
  - Data was copied.
  - Player 1 and player 2 were inverted for the copied data.
  - Copied data was recombined with original data.
  - This process also makes more data available for training
- Only available pokemons that were cheaper than 3500 minus the sum of the 5 cheapest available pokemons were kept. This ensures that each pokemon will exist in at least one combination.
- Each available pokemon was battled against each submission opponent.

01merge_data.R

- Both types of both players were added to the data.
- The weakness matrix was melted and used to determine the strength of each type of the attacking player against each type of the defending pokemon.

Feature Engineering

02create_features.R

- Feature engineering is centred around the calculation of the damage formula:
  - Source: https://bulbapedia.bulbagarden.net/wiki/Damage#Damage_calculation
  - Level is level of attacking pokemon.
  - A is attack/ special attack of attacking pokemon.
  - D is defense/ special defense of defending pokemon.
  - Power is the power of a move.
    i. It is set to 80, the most frequent move power in generation VII.
    ii. Source: https://pokemondb.net/pokebase/300699/what-is-the-average-power-of-all-damaging-moves

$$Damage = \left(\frac{\left(\frac{2 \times Level}{5} + 2 \right) \times Power \times A/D}{50} + 2 \right) \times Modifier$$

02create_features.R

- Targets, Badge, Critical Strike, STAB, Burn and Other are unknown → Set to 1.
- Random is a number between 0.85 and 1 → Set to 0.925.
- Type effectiveness is taken as the maximum strength of each player.
  - Strength of attacking type is equal to weakness of defending type.
- Weather influence.
  - Normally = 1.
  - If at least one type has a weather advantage → Weather influence = 1.5.
  - If the only type has a weather disadvantage → Weather influence = 0.5.

$$Modifier = Targets \times Weather \times Badge \times Critical \times random \times STAB \times Type \times Burn \times other$$

02create_features.R

- Damage and Special damage values are then rounded down.
  - Source: https://www.youtube.com/watch?v=OMA7hY48jG8&t=297s
- From this, the maximum damage each player can inflict is determined.
  - Maximum of normal damage and special damage for each player.
- Then, we calculate the number of rounds each pokemon would need to defeat other.
  - Divide HP of defending player by maximum damage of attacking player (rounded up).
- The expected number of rounds of the battle is determined.
  - Equal to the minimum of the expected number of rounds of both players.

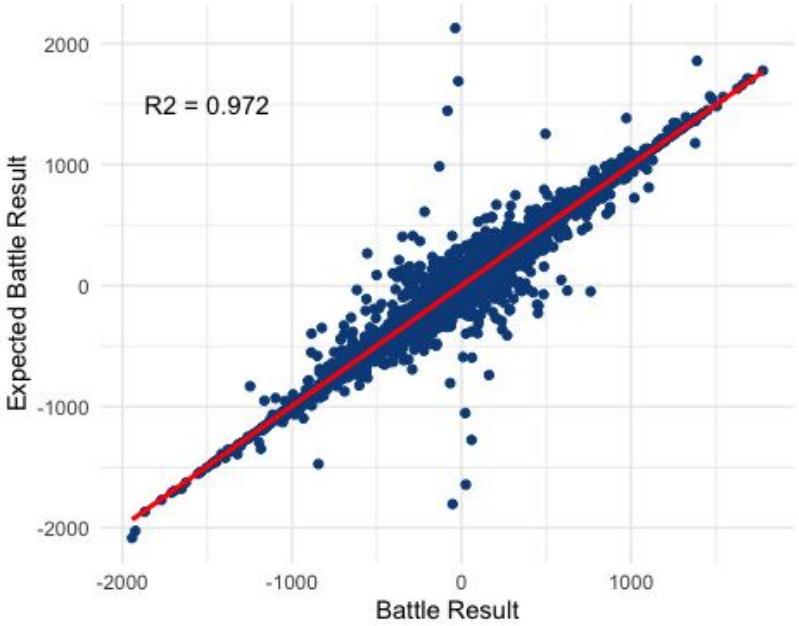
02create_features.R

- The expected remaining HP of each player is calculated:
  - HP of player minus expected number of rounds times maximum damage of opponent.
  - All priorities aside, the first player to move will be the player with the higher speed.
    - If the winning player is also the fastest, the losing player will have one less chance to strike.
    - Otherwise, both pokemons will strike the same number of times
    - Source:https://gaming.stackexchange.com/questions/52075/how-do-i-know-who-attacks-first

02create_features.R

- The expected battle result is calculated:
  - If the expected remaining HP of player 1 is larger than that of player 2, the expected battle result is the expected remaining HP of player 1.
  - Otherwise it is -absolute(Expected remaining HP of player 2).
- The expected portion of remaining HP of player 1 is found by dividing the expected battle result by HP of player 1.
  - If it is negative, it is rounded up to 0.

02create_features.R



- Additionally, ratios of several features were calculated to feed into ML model.
- Feature engineering provides solid approximation of target variable:
  - R^2 of BattleResult is 0.972.
  - R^2 of PortionRemainingHP_1 is 0.931.

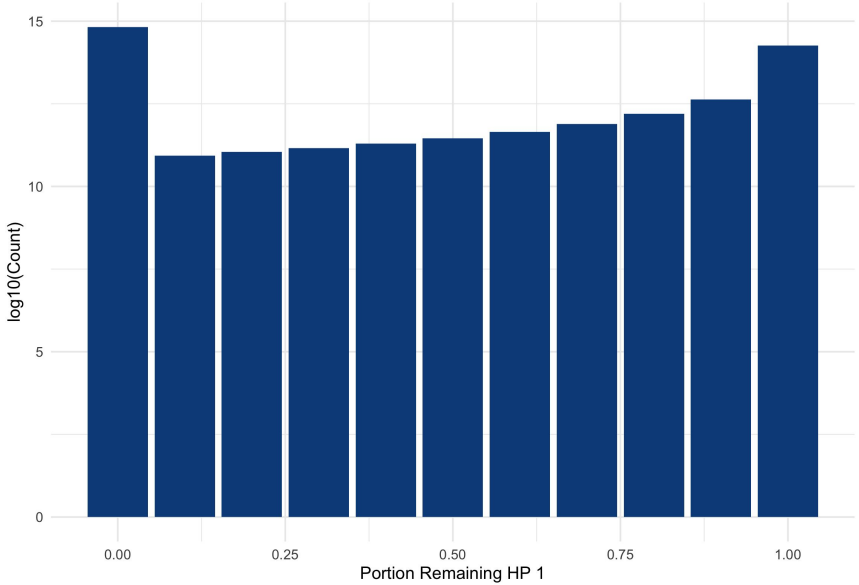
Model Training

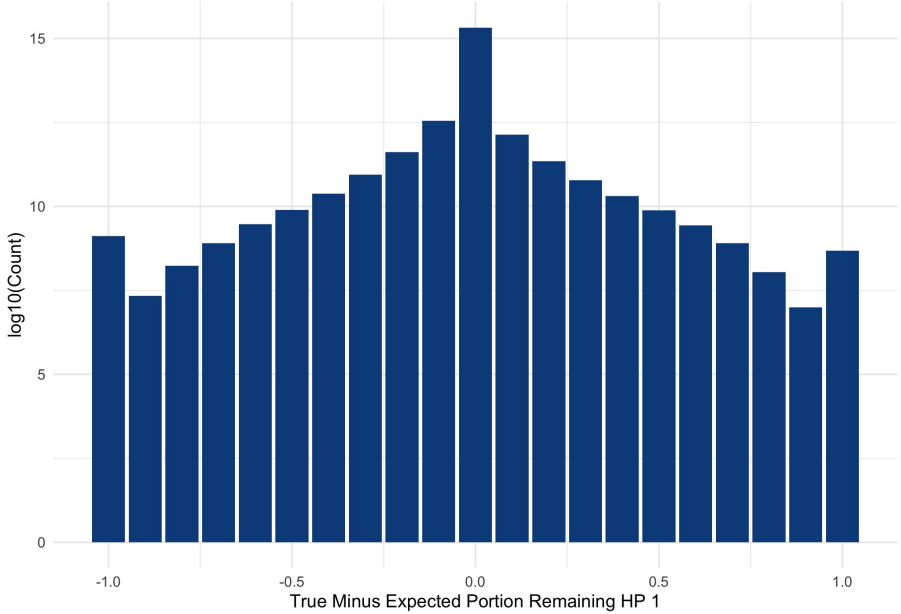
03parameter_search.R/ 04train_model.R

- Because feature engineering provides a solid approximation of the remaining portion of HP of 1, we redefine the target variable of our ML model as the difference between the true and the expected portion of remaining HP of 1. This has the advantage of:
  - Reducing the number of features fed into model.
  - Forces the model to focus on points that are difficult to explain with damage calculation.
  - Provides a normal distribution of values.

03parameter_search.R/ 04train_model.R

The log10 histogram of the true portion of remaining HP of 1 is dominated by values around 0 and 1 whereas the log10 histogram of the difference between the true and approximated value is more gaussian like.





03parameter_search.R/ 04train_model.R

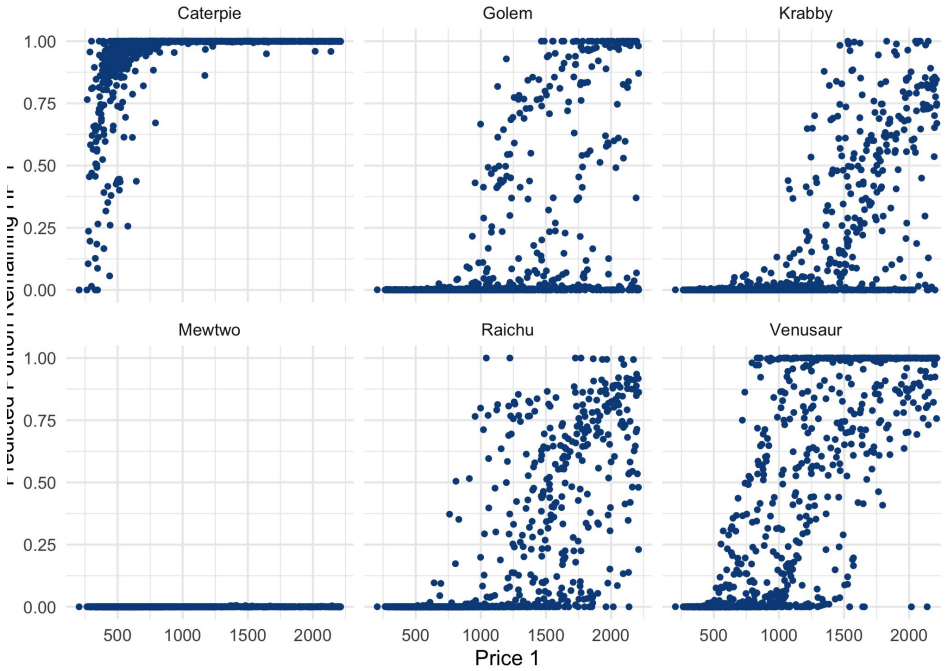
- None of the submission opponents exist in the training data. For our model to work well on opponents it has not seen previously, the following steps are taken:
  - 25% of Name_2 values of training data are kept for validation.
  - Other Name_2 values of training data are grouped into 6 folds for CV.
  - Name_2 than removed from list of features.
  - Name_1 is kept since all available pokemon are in the training data.
- Model is an L2 regressor.
- Choice of ML model → LightGBM.
  - Fast training.
  - Capable of handling categorical features with high cardinality.
  - Good performance.

03parameter_search.R/ 04train_model.R

- Using our 6-fold grouped cross-validation and an exact grid search, we explore several combinations of booster type, learning rate and number of leaves.
- After having selected the optimal combination of parameters, we train the model on the entire available data.
- Predictions are made for training and submission data.

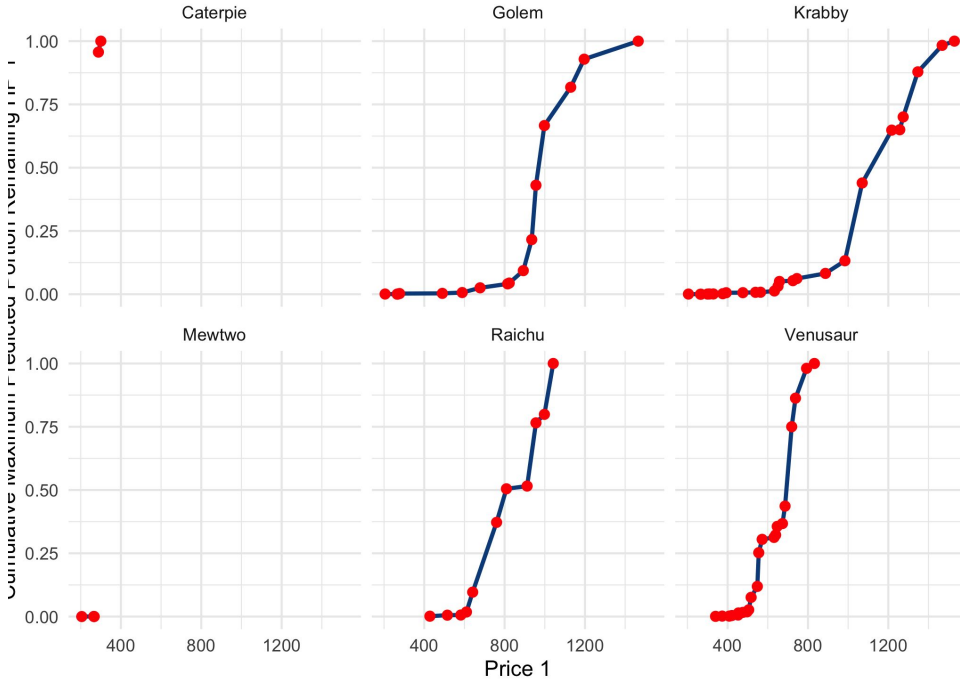
Finding Combinations

05find_combination.R



- Plot the predicted portion of remaining HP of 1 against the price of pokemon 1 for each opponent. Observations:
  - 99% chance of losing to Mewtwo.
  - Significant chance of losing to Golem and Krabby.
  - Total or partial win against Caterpie, Venusaur and Raichu guaranteed.

05find_combination.R



- Strategy:
  - Reserve 3 cheapest pokemons to fight Mewtwo, Golem and Krabby.
  - Mewtwo only allowed to fight 3 cheapest pokemons.
  - For pokemons who are not the 3 cheapest, we only keep the pokemons whose predicted value is equal to the cumulative maximum of the predicted value for each opponent. This ensures that an increase in spending will translate into an increase of the predicted target.

05find_combination.R

- Our filtered data now only contains about 100 rows.
- We can then therefore more easily compute each combination of pokemons fighting the 6 pokemons of the submission. For each combination, we calculate:
    - The mean of the predicted portion of remaining HP of player 1.
    - The total price of each combination.
    - The number of unique Pokemon identifiers.
- We then only keep the combinations that:
    - Have a total price less or equal than 3500.
    - Have 6 unique Pokemon identifiers.
- We select the combination with the highest mean predicted portion of remaining HP 1 as our combination of choice.