P8106 Final Project

Phoebe Mo(km3624), Stella Li(cl4043), Yihan Feng()

5/7/2021

1. INTRODUCTION

1.1 Motivation and Objective

Pokemon are fictional creatures, which can be captured by players and trained to battle each other in the augmented reality game Pokémon Go!. In this project, we use a dataset from Kaggle that contains different attributes and catch rates for 721 unique Pokemons. We try to understand the relationship between Pokemons' different attributes and their respective catch rates. Here are some questions we want to answer: Which predictor(s) play important roles in predicting catch rates? Which type of model (linear or non-linear) serves as a better method to predict the catch rate?

1.2 Data Preparation and Cleaning

The original dataset has 20 predictors such as HP, and the outcome 'catch_rate'. After cleaning the names of these variables, I did the following steps to clean the data:

1.2.1 Notice that 'type_2' and 'egg_group_2' are indicators of if a pokemon has a second type or belongs to a second egg group. The 'Null' values in the data means the pokemon does not has the second type/group, so I changed them into "none" to make them as a category to be meaningful;

1.2.2 The 'generation' predictor is originally a numeric type, but it has only 6 integer values, so I decided to mutate it to be categorical;

1.2.3. Irrelevant variables 'number' and 'name' are dropped. 'total', which is the total base battle statistic for each pokemon, is calculated and reflected in other battle attributes such as 'hp' and 'attack'. So I dropped it to avoid intercollinearity. Later, after plotting the correlation map, I found 'weight_kg' and 'height_m' has relatively high correlation(correlation plot is shown in section 2). After consideration, I chose to drop 'height_m' since it may has less effect on catch rate compared to 'weight_kg'. 'has_gender' is also dropped because all values are "TRUE".

2. Exploratory Analysis / Visualization

After plotting scatter plots for continuous predictors and boxplots for categorical predictors, significant trends were not observed in different types, egg groups, body_styles or colors. However, the battle attributes: 'hp', 'attack', 'defense', 'sp_atk', 'sp_def', and 'speed' seem to have a negative association with catch rate. The 'weight_kg' also seems to have negative association with catch rate. And the pokemons that are legendary have significantly lower catch rates. Below are some selected visualizations:

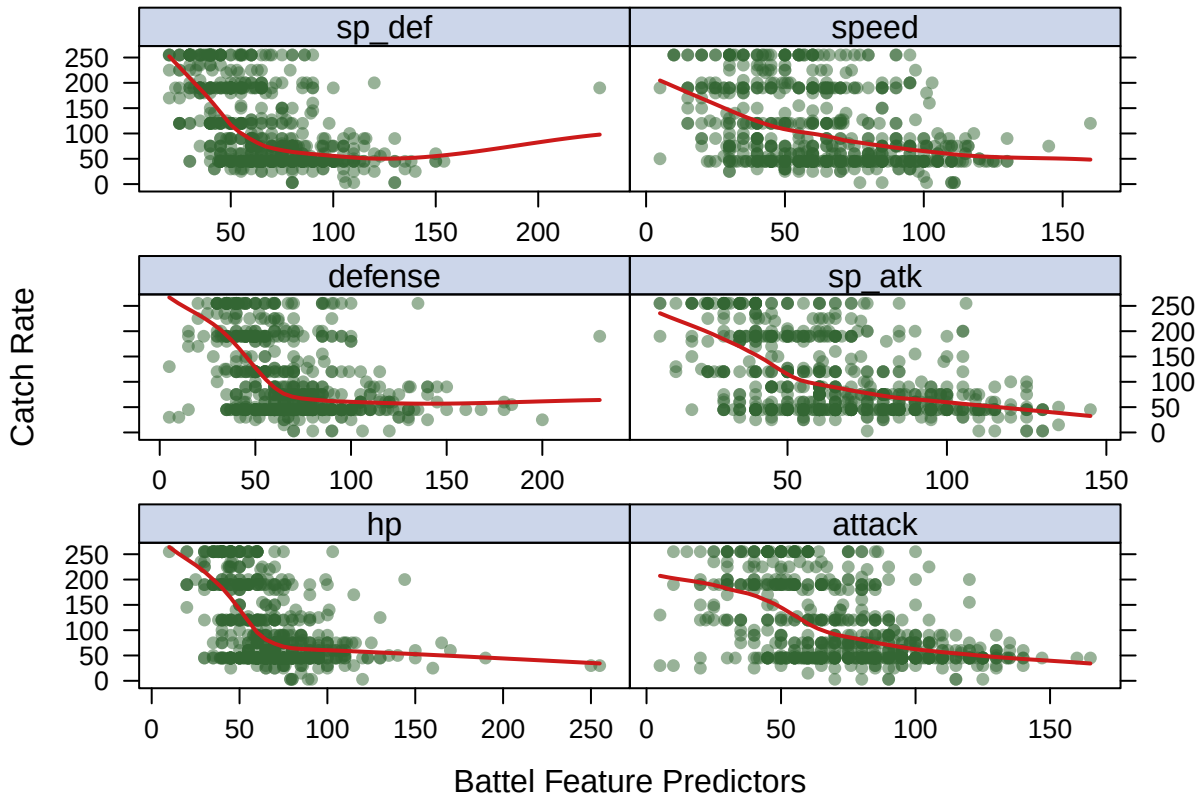
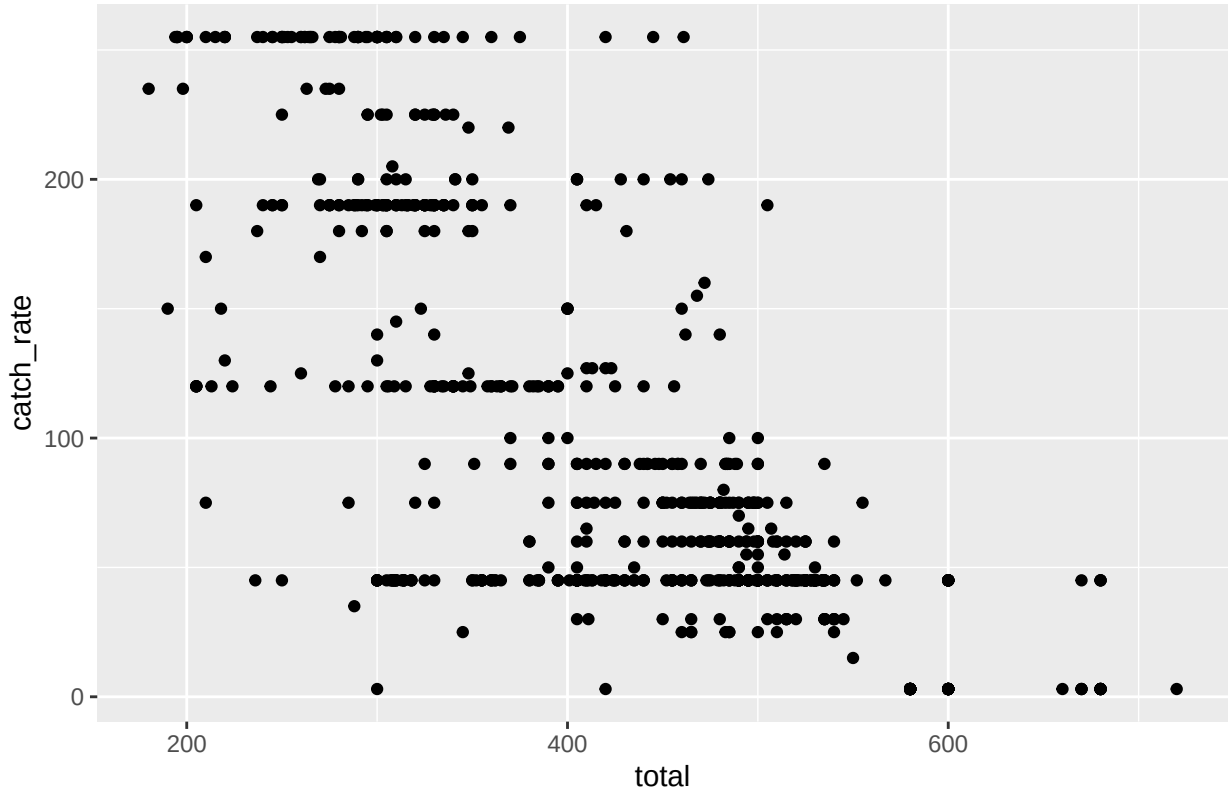
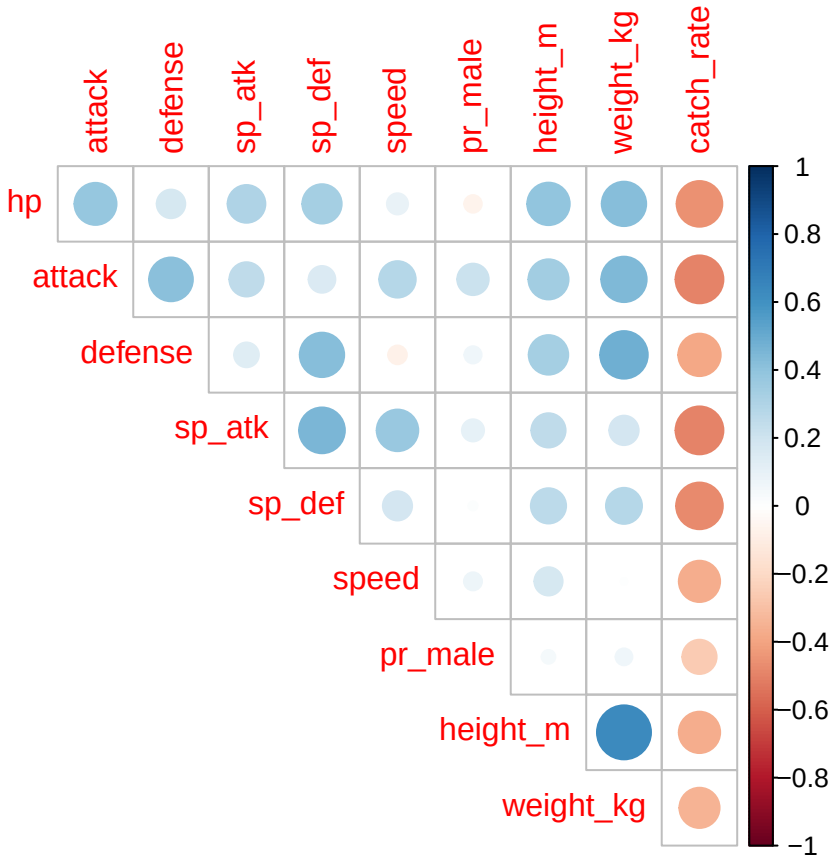


Figure 2. Scatterplot of total battle features vs. catch rate





3. Models

The dataset for analysis contains 644 samples and 18 predictors:

- `type_1`: primary type
- `typw_2`: second type, in case the Pokémon has it
- `hp`: health point
- `attack`: base attack
- `defense`: base defense
- `sp_atk`: attack speed
- `sp_def`: defense speed
- `speed`: base speed
- `generation`: number of the generation when the Pokémon was introduced
- `is_legendary`: Boolean that indicates whether the Pokémon is Legendary or not
- `color`: color of the Pokémon according to the Pokédex
- `pr_male`: probability of being a male Pokemon
- `egg_group_1`: egg group of the Pokémon
- `egg_group_2`: second egg group of the Pokémon, in case it has two
- `has_mega_evolution`: Boolean that indicates whether the Pokémon is able to Mega-evolve or not
- `height_m`: height of the Pokémon, in meters
- `weight_kg`: weight of the Pokémon, in kilograms
- `body_style`: body style of the Pokémon according to the Pokédex
- `catch_rate`: the response (on 0 - 255 scale)

The dataset is split into 75:25 training to test dataset. Based on our midterm project, We include GAM model, since it has a dominant performance among other methods(e.g. lda, qda, nb, lasso, ridge…), as a

competitive model for predicting the catch rate. We also choose Boosting, Random Forest, Bagging, and Support Vector Machine models to train the data with cross validation. RMSE is used to compare and choose the final model. Finally, we use clustering to have a better visualization of pokemons that have similar attributes and to see if significant relationship between combat attributes and catch rate can be found.

3.1 GAM

Based on the visualization plots, we can see that some continuous variables have curve-like trend in the right tail. In this case, I consider using GAM model in order to include this trend into the analysis. For such predictors, I allow GAM to make it non-parametric smoothing term. GCV is used to choose the degree of freedom for the model.

After fitting, it is observed that the smooth terms of 'hp', 'attack', 'defense', 'sp_atk', 'speed', and the coefficients of 'type_1Poison', 'type_2Water', 'generation3', 'pr_male', 'egg_group_1Undiscovered', 'body_stylemultiple_bodies', are significant. These result greatly coincides with the ones given by previous model.

3.2 Bagging, Boosting, Random Forest

We decide to use tree-based ensemble methods for regression. The function `randomForest()` implements Breiman's random forest algorithm for classification and regression, since our data is high dimensional, we also used function `ranger()` for training the random forest model.
Since Random Forest does not have any underlying models, so it does not have assumptions, other than there is no multicollinearity among the variables.
For bagging, we used all 18 predictors for training.

Boosting primarily aims at reducing bias and variance. For boosting, we first did a grid search using `caret`. We then used the fast implementation of random forest when tuning the model and tuned the gbm model based on its result.

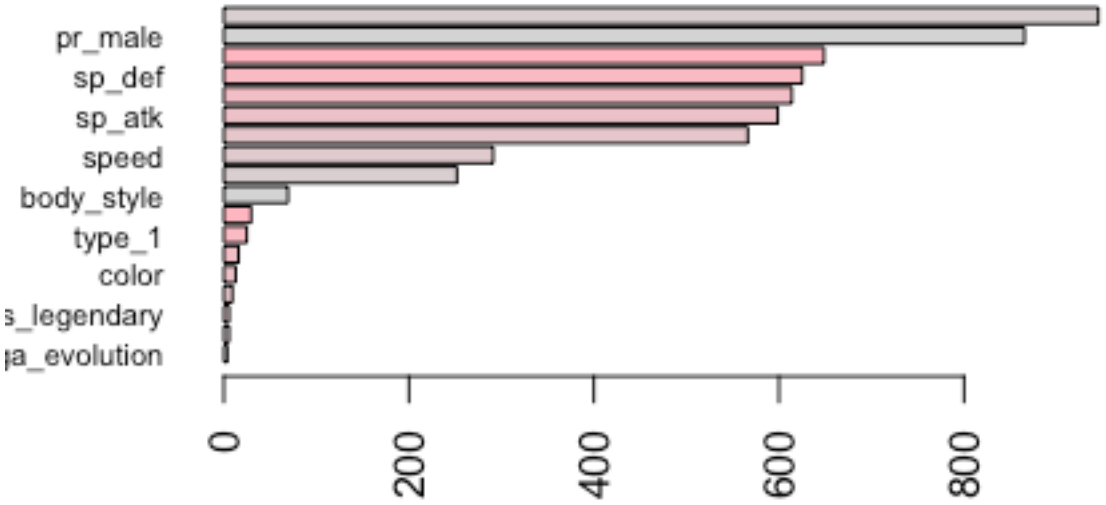
3.3 SVM

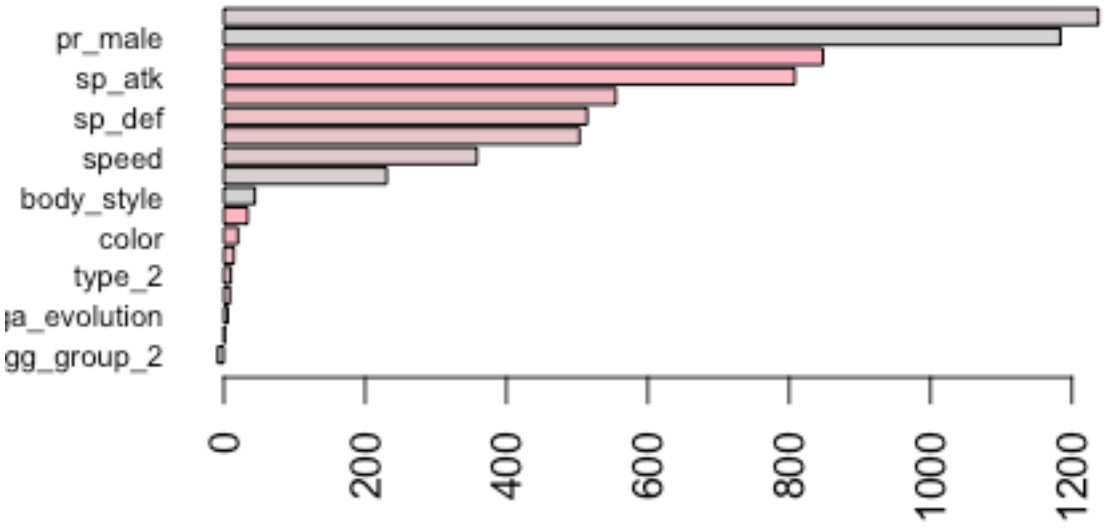
In SVM, we want to find a separating hyperplane that maximizes the gap between classes to optimally predict the outcome. Support Vector Machine with both linear and radial kernel is used to train the data. The tuning parameter epsilon(trade-off between correct classification and maximization of the gap) and gamma(how non-linear the fit could be) are tuned given a suitable range of numbers in the function.

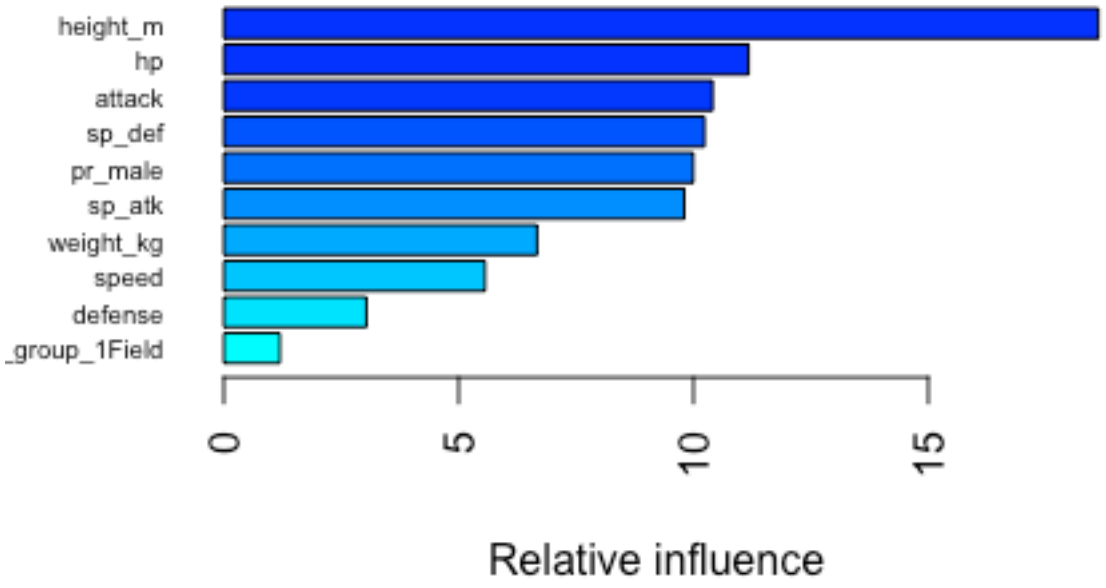
3.4 Clustering

In order to have a better visualization of the relationship between the pokemon and their different attributes, we use both k-means clustering and hierarchical clustering to group similar pokemons together. Here, we use only numerical variables. As can see from k-means clustering, we use the optimal number of clusters = 3 and use Euclidean distance to decide the similarity. From the plot briefly, cluster 1 tends to has lower hp, cluster 2 has lower catch_rate, and cluster 3 has relatively higher weight and defense.
For the hierarchical clustering, we use complete linkage(maximal inter-cluster dissimilarity) and euclidean distance. Since hierarchical clustering provides us all the possible combinations of clusters, to visualize, we made a heat map to see how all the variables vary for each observation. From the heat map, we found pokemons that have higher hp, height, and weight tend to have lower catch_rate, whereas those which have relatively low combat attributes tend to have higher catch_rate.

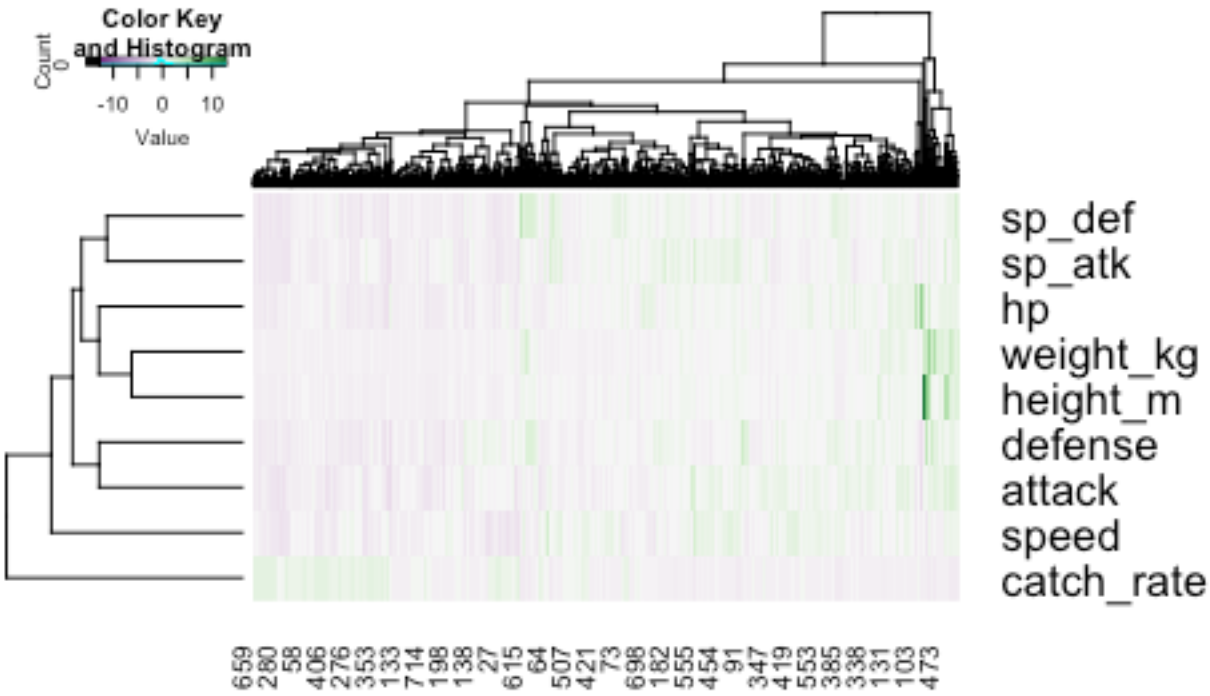
3.5 Variable Importance and Clustering Results






height_m
hp
attack
sp_def
pr_male
sp_atk
weight_kg
speed
defense
_group_1Field
0
5
10
15
Relative influence


K-means Clustering
Dim2 (14.1%)
Dim1 (44.4%)
cluster
1
2
3



3.6 Final Model and Visualization

```
##           Model     RMSE
## 1           GAM 42.10908
## 2      Boosting 19.87961
## 3 Random Forest 19.75706
## 4       Bagging 18.90574
## 5  SVM (linear) 44.78799
## 6  SVM (radial) 35.27461
```

4. Conclusions

According to the Cross-Validation RMSE, Bagging has the lowest RMSE 18.91, meaning among these models, Bagging is the best candidate to predict the catch rate. Although we picked GAM model before having these new models, it seems like all the new models, except SVM(using linear kernel), have better performance than the GAM model. Random Forest and boosting also have very good performance.

5. Limitations

5.1 The GAM model does not truly select the tuning parameter and do the model training, therefore this may be the limitation of the GAM model. However, since it has the smallest training RMSE, we still decide to choose it as our best method for the prediction in this project.

5.2 For the k-means clustering, since we need to decide k ourselves, there is no reference for us to decide which k is better. So the manually selected k and the given visualization may not be the best presentation.