**Pokémon Combat**

**Introduction:**

This data set includes about 721 Pokémon, including their number, name, first and second type, and basic stats: HP, Attack, Defense, Special Attack, Special Defense, and Speed. Based on HP, Attack, Speed, Defense, Special Attack, Special Defense and other features to predict which Pokémon will win the battle. At the same time, we have classification for Pokémon’s type. Each Pokémon has a different type. Here we use Naive Bayes, SVM, Random Forest, and KNN to do the classification. The best way for us to compare the accuracy of each method is to perform cross-validation tests on each algorithm one by one, compare the result, and then adjust the parameters to ensure that each algorithm achieves the best solution. With those certain types, we are going to develop a machine learning model for final project. Pokémon GO is an interactive, mobile game that uses your phone's GPS data and clock to show Pokémon hidden near your current, physical location. The Pokémon which appear on-screen in the app can be captured.

**Definition of the method used:**

Naive Bayes: A naive Bayes classifier is an algorithm that uses Bayes' theorem to classify objects. Naive Bayes classifiers assume strong, or naive, independence between attributes of data points. Popular uses of naive Bayes classifiers include spam filters, text analysis and medical diagnosis. These classifiers are widely used for machine learning because they are simple to implement.

Support Vector Machine (SVM): A support vector machine (SVM) is machine learning algorithm that analyzes data for classification and regression analysis. SVM is a supervised learning method that looks at data and sorts it into one of two categories. An SVM outputs a map of the sorted data with the margins between the two as far apart as possible. SVMs are used in text categorization, image classification, handwriting recognition and in the sciences.

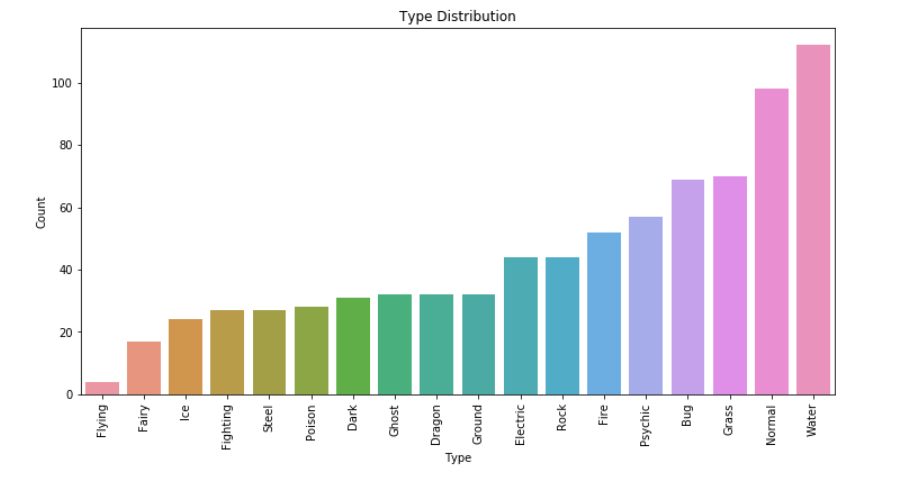
Random Forest: A random forest is a data construct applied to machine learning that develops large numbers of random decision trees analyzing sets of variables. This type of algorithm helps to enhance the ways that technologies analyze complex data.

KNN: A k-nearest-neighbor algorithm, often abbreviated K-NN, is an approach to data classification that estimates how likely a data point is to be a member of one group or the other depending on what group the data points nearest to it are in. The k-nearest-neighbor is an example of a "lazy learner" algorithm, meaning that it does not build a model using the training set until a query of the data set is performed.

**Why we used:**

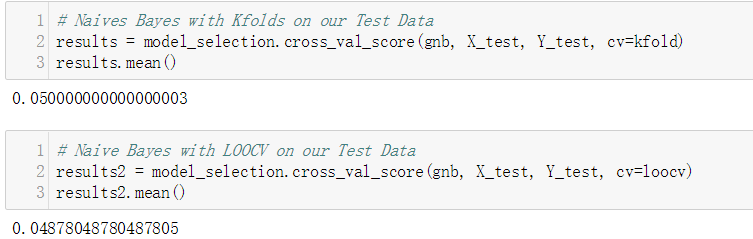
In our dataset, there are 18 characteristics. How to select the features that have the greatest effect on the results in order to reduce the number of features when building a model is a problem that we are more concerned about. The idea of assessing the importance of features using these four methods is actually very simple. It is to look at how much each feature contributes, then take an average, and finally make up the contribution between features.

**Experimental Results and Analysis:**

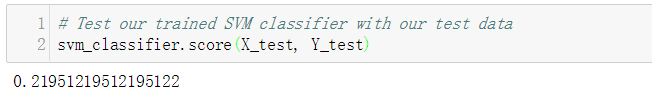
****

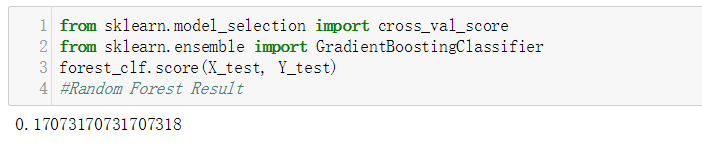
In our project, we do the data cleaning first and then some data visualization. This plot shows the distribution of the type. It’s clearly that type water has the most Pokémon in the dataset and normal ranks 2. Flying Pokémon is the rarest type of Pokémon,

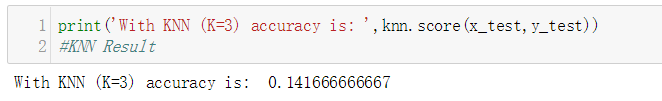
Then we do the classifier based on different methods.

Using the Navies Bayes with both Kfolds and LOOCV, we get a result like this. 

This result is obviously not satisfying, so we switched to using SVM for classification in order to achieve better results. The fact is that we have succeed and the results of the model prediction have risen to above 0.21.



In addition to SVM, we also used random forests to do data classification. The result is as follow. The result is slightly lower than SVM, but it is still much higher than Navies Bayes. 

Finally, we also used KNN for classification.

Comparing the four classification methods we have done, we can see that the accuracy of the SVM is the highest. Naive Bayes needs to calculate the prior probability. There is an error rate in the classification decision and the accuracy is minimal. SVM can handle the interaction of non-linear features, which has a high accuracy. In order to avoiding overfitting, SVM provides a good theoretical guarantee. Random forests are often the best method to solve for many classification problems. It is fast and adjustable to train. It doesn't need to worry about adjusting a lot of parameters like a support vector machine. It can handle irrelevant features, but it will ignore the correlation of data. KNN can be used for non-linear classification, and sample imbalance problems result in poor performance. The reason is that the number of samples in some categories is large, while the number of other samples is small, which causing a low accuracy. For our data, SVM has the highest accuracy of 0.26. However, since all four method accuracies are very low, we decided to join the prediction of Pokémon combat.

**Predicting the Winner:**

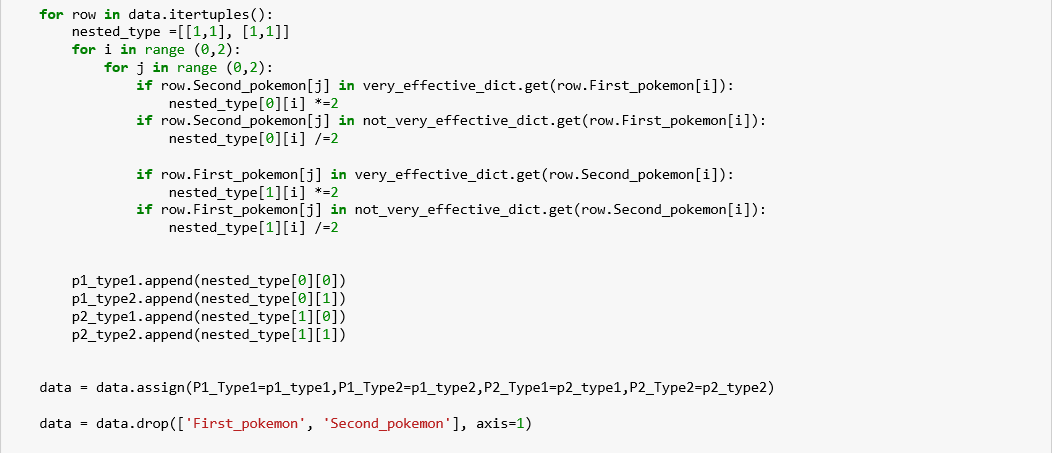
Prediction of a winner in a combat between two Pokémon’s is a simple classification problem. Being a big fan of Pokémon and watching the combats of the different Pokémon’s on television, we were curious to know if we could determine the winner of a combat just by looking at the type and different features of the Pokémon as mentioned above. Our main idea behind taking this dataset was if looking at only the features we could know what type of Pokémon was and based on the other features taking into account what type effectiveness could do we predict the winner.

But since we were unable to classify type appropriately we moved forward with predicting the winner taking into account all the features including type.

**Working Code:**

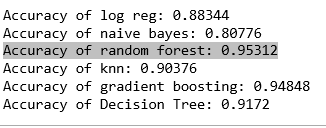
After completion of the classification of types, we used two datasets which has the combats results of the previous matches between Pokémon’s.

We have already done the cleaning part with the previous classification, so we jumped into data engineering part which was the most challenging for us, because we were unable to think what could be done with the type and thought of dropping the feature, but while searching online we read an article where they used a similar thing and gave weights to this kind of feature.

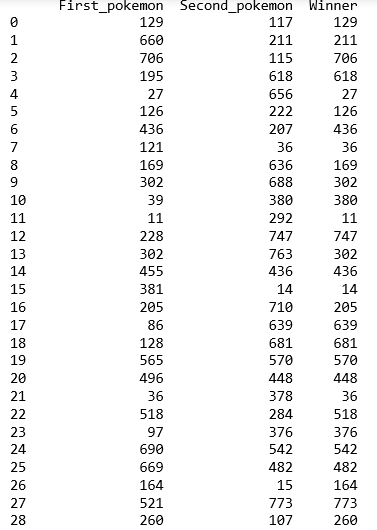


So that we did the similar thing and we started giving weights by starting everything at 1 and if we found that a particular type of Pokémon is more effective on the other we would multiply the factor by 2 otherwise divide it by 2. By doing this we will get weights for every combat based.

We parsed everything through different Classifiers and found that the random forest is the best classifier that gives us the prediction accuracy of 95%.



Based on our model we predicted ‘tests’ dataset.



**Conclusion:**

In conclusion, we can say that we were able to predict who the winner could be but failed to predict the type of Pokémon. The reason is that we used four methods to predict the type of Pokémon, but because the accuracy of the four methods are very low, so that we decided to predict whose the winner in the Pokémon’s battle. It was our first time to do a bit of data engineering and use Machine Learning Algorithms and we did encounter with a lot of difficulties but were able to cope with them and get to our result.