David Leshon

Professor Butler

DSC-140S

3 December 2024

Best Pokemon Type

Which Pokemon type is the best? It's an interesting question to answer since the answer would affect both the casual players and the competitive ones. Therefore, the procedure looked more at the competitive side of things, because while it could shake things up there, the casual side of the game is much bigger. With this in mind, this question was answered via two methods: Which types have the highest total stat average, and which Pokemon would be best to bring to a competition based on multiple factors? All important things to consider, since players only get 6 Pokemon per team, and competitive players are typically only able to use 4.

Method one will be looked at first to start the process. To begin, we will head into R to find the number of clusters a k-means algorithm thinks the data set has. This is important because we will try to see if a k-nearest neighbors machine-learning algorithm can find the best Pokemon types independently without assistance. For the k-means algorithm, we had the algorithm take into consideration all of the numerical columns of data, which were the categorical numbers relating to each type, all the stat averages individually, the total stat average, the total number of Pokemon, the number of physical, special, and status moves, the number of immunities, resistances, and weaknesses, the number of super effective, normally effective, and not effective moves, and the number of restricted Pokemon. Chart 1 shows an elbow plot that represents the number of clusters most likely to be in the data set, with the best choice for the number of clusters being 5. Although it does seem to not be that good, it is the best because the next best number would be too close to 18, which would just mean that every type is in its own separate cluster, and that gives no useful information. With this number, we can then add a column assigning all the types to their clusters and head over to Python to do the machine learning.

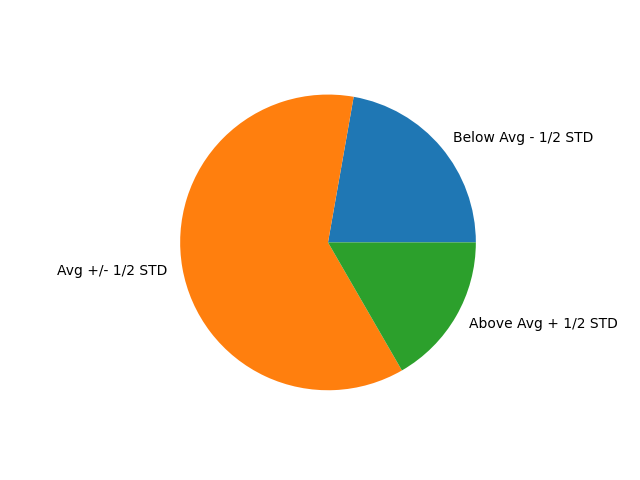
**Figure 1**

A graph with a dotted line

Description automatically generated

Once here, since the train-test split for the algorithm is random on what data points go for the test set and what points go with the train set, we will do the k-nearest neighbors test multiple times to get a range of accuracies to truly see if the algorithm can do what we need it to do. First, the train-test split divides the data set in half based on the same data that was used for the k-means test, so nine data points are used for the training, and the other nine are tested. This is because it is a small data set and using too much for training leaves too little for testing, and not having enough for training leads to bad performance when testing, so half and half is the best way to minimize these concerns. After doing this 10 times, we conclude that the algorithm cannot do our job for us, as while it does have the ability to be 100% accurate, its average accuracy score was just under 75%, while the range of its accuracy was around 45%, meaning the lowest score that the algorithm found was 55%; therefore, we will have to manually group the data to see which Pokemon type is the best when looking at total average stats alone.

**Figure 2**



Using a mapper function to separate the types into groups based on where they were positioned around the mean and standard deviation, we get the pie chart shown in Chart 2. The splits described in the labels are such that each group had the chance to grab multiple types, but still be manageable to work with. This is useful since it shows that most of the types’ total stat average is around the average, while there are a few types that fall above it. From here, we can pull out only the types in the green group, which are the types that are above the mean plus one-half of the standard deviation. This gives us the final answer, being that the best type to have in a battle, only considering the total stat average of the types, would be Dragon, Steel, and Fighting.

Method two is slightly more involved, as it covers multiple topics for the competitive Pokemon scene. The first topic is which type would be the best for using status moves. This was found by recalculating each type’s total stat average, but this time, it excluded the speed stat. Speed was excluded so that when we did the next check, it wasn’t counted on each side of the check. Then, it checked each type to see if its average speed was higher than the new total stat average. For those that were, we then pulled out the number of status moves that each type had and ranked them by that number. With that, the best fast-moving status types are Normal, with an average speed stat of 73.61 and 102 status moves; Dark, with an average speed stat of 79.62 and 15 status moves; and finally, Bug, with an average speed stat of 58.96 and 10 status moves, all of which are a little higher than the average for the speed stat. The second topic is separating the good physical and special attackers that are fast and the good physically and specially bulky Pokemon. This was done slightly differently. This time, for both physical and special attackers, we started by pulling out only the types that had a combined attack (corresponding to whether we were checking for physical or special) and speed stat that was higher than one-third of their total stat average. One-third was chosen because we are adding A black screen with white text

Description automatically generatedtwo of the stats together, and two stats are one-third of the amount of stats a Pokemon has; it was also to minimize the number of types that were chosen, as we are adding two stats together for the check. Then, we did another filter to make sure that the types chosen to be the best physical attackers had more physical moves than special moves, and vice versa for the special attackers. Finally, they were then sorted in order of highest physical or special stats, respectively. Next were the defenders, and while the process was similar, the filters were quite different. First, instead of doing attack plus speed, the check was just if the respective type was bigger than one-sixth of the total stat average (one-sixth is because it’s just one stat to be tested here, and that is one-sixth of the number of stats a Pokemon can have). Then, they are narrowed down more by a filter that checks if they have more weaknesses than resistances or immunities; if so, then they are cut from the list. Finally, they are sorted into such that the higher on the list they are, the higher their respective defense stat is. Table 1 shows all the results of these tests. The last topic is meant to narrow down the collective list, as competitions don’t allow more than two restricted Pokemon per team if they even allow any at all. So, the first thing was to hop back over to R and make a box-and-whisker plot to see the spread of the data and determine if there were any outliers, which Chart 3 shows. Once those A screenshot of a black screen

Description automatically generatedoutliers were identified as the Psychic and Dragon types, I then went back to Python to finish out the calculations. There, the first thing that was done was to make a sub-dataframe containing the Pokemon type and their number of restricted Pokemon; then, using that subframe, we found the average amount of restricted Pokemon (rounded to two places got us 9.81), and the standard deviation (rounded to zero places got us 4). With that, we got values for the upper and lower bound of what we would contain the search to by doing the average plus the standard deviation and the average A graph with a line and dots

Description automatically generatedminus the standard deviation, respectively. With the bounds in place, we then ran through the sub-dataframe to get a list containing only the types that call within our limits, shown in Table 2. So, with all three topics covered, if we were to count each time a type had made a list and record its position within the list, we could find the best competitive Pokemon type. So therefore, the types that appeared the most and had the lowest average position should equate to the best types to bring to a competition, which would be Fairy, Dark, Ground, Bug, and finally, the Normal Type.

**Figure 3**

**Figure 4**

**Figure 5**

In conclusion, the best types in battles are Dragon, Steel, and Fighting, while the best types competitively are Fairy, Dark, Ground, Bug, and Normal. This is an interesting finding because it lines up decently well with the average opinion of competitive players, but it isn’t exact. For example, most players would not say that the Bug type is a very good type to have, but it is the fifth best type found from this study. It’s also interesting because the types that were found to be the best in competition aren’t always the ones that were chosen, and vice versa for the types that are best in battle. One example of where it lines up is that the most dangerous unrestricted Pokemon is Garchomp, who is a Dragon/Ground type, which lines up for both best in battle and best for competitions. However, in the current format, most times a Steel type is brought into battle its function is to terrastalize into a completely different type (terrastilizing just means that it changes the Pokemon’s type to a singular one, either one it didn’t have previously or one it did have. Either way, it ends up with just that one type). Other than that, it was a joy to work with this genre in a way that I rarely get the chance to do, although it was stressful at times to get things to work, like sorting a list and figuring out tuples.

Works Cited

*Pokemon*. Game Boy, Game Boy Color, Game Boy Advance, Nintendo DS, Nintendo 3DS,

Nintendo Switch, Game Freak, 1996-2023.

Bulbapedia. “Bulbapedia, the community-driven Pokémon encyclopedia.” *Bulbapedia*, 30 Oct.

2024, bulbapedia.bulbagarden.net/wiki/Main\_Page.

Leshon, David. Python. Final Project.ipynb, 6 December 2024.

Leshon, David. R, Final Project.R, 6 December 2024.