Who is the Strongest Pokemon?

Israel Nolazco
Introduction to Data Mining and Analytics
Lewis University
Romeoville, Illinois
israelnolazco@lewisu.edu

***Abstract*—There are several factors that people find Pokemon an appealing game. The game offers different sets of features that attracts many different personalities, from those who are interested in collecting and those who are interested in raw power. The game usually starts by having the players pick one out of the three "Starting Pokemon." These three Pokemon are always with in the element type of Fire, Water and Grass. Three elements that do not overpower one to another; like the classic game of rock paper scissors each one of them can beat the other. Although the game developers attempt to ensure a well-balanced game. This document will explore and answer who truly is the strongest Pokemon**

Keywords

Pokemon, Cluster, Data Analytics

I. Introduction

Pokemon Go released on July 5th, 2016 and with it brought back several great memories from those who passionately play the Pokemon series [1]. The Pokemon series has greatly expanded from its initial launch. The game originally had one hundred Pokemon; today the series has expanded to about eight hundred and nine Pokemon [2]. However, it is not the number of Pokemon that accredits the success of the franchise but, rather it's the attempt to create a well-balanced/fair game for its users. In the introduction to the game, the player is always asked to choose one out of three Pokemon. The three Pokemon are representatives of the Fire, Water and Grass Elements. This is like the classical rock, paper and scissors game; like that game each elemental Pokemon is meant to check and balance each other. However, this document will make use the programming language Python and its libraries NumPy, Pandas, Matplotlib and Kmeans clustering algorithm to officially find out which elemental Pokemon is the strongest in the game.

II. Methods

In order to find out who truly is the strongest elemental Pokemon from its three types; a data set was gathered from DataWorld website. The data set consists of seven hundred and twenty-one Pokemon. The data itself is missing the most recent generation of Pokemon. The data was originally source from author Alberto Barradas who has not updated his file for approximately three years; thus explaining it shortages [3]. The data provided the Pokemon's name, types, health points, attack points, defense points, special attack points, special defense points [4]. Additionally, JupyterLab was utilized to run the Python environment and its libraries. The following are its appropriate imports necessary to run this analysis:

```
%matplotlib inline

import matplotlib.pyplot as plt

import numpy as np

import pandas as pd

from sklearn.cluster import KMeans

from pandas import Series, DataFrame

import pandas as pd
```

The file obtain from DataWorld is in a CSV format; with it we are to read the document and store only the key component from this analysis as a Data Frame. Additionally, "Type 1" was set to be the index of the data frame as it will be crucial to properly sort by elemental type of Pokemon.

```
#reads csv file and give it a reference name

pokedt = pd.read_csv('Pokemon.csv')


#create a dataframe from csv file

pokedata = DataFrame(pokedt,columns=['Name','Type 1','HP','Attack','Defense'])


#set an index of type. Thus able to select categories

pokedata.set_index("Type 1", inplace=True)
```

**Type1 was only necessary to simplify the process as this document will only explore the raw power output of Pokemon rather than looking at its secondary factors.

The following is a code utilize to sort out on the Grass, Fire and Water type Pokemon. Furthermore, the Name and HP columns were removed from its respected category since this document is not exploring one unpacific Pokemon, but rather an element. That leaves only the Attack and Defense points of each Pokemon with in given category and storing those values in an array.

```
#sets reference name for all Pokemon with Grass Type
typegrass = pokedata.loc['Grass']

#massaging data to remove names and HP to only get Attack and Defense points
grassraw = typegrass.drop(columns=['Name','HP'])

#creates array from remaining columns
grassrarray = np.array(grassraw)

#sets reference name for all Pokemon with fire Type
typefire = pokedata.loc['Fire']

#massaging data to remove names and HP to only get Attack and Defense points
fireraw = typefire.drop(columns=['Name','HP'])

#creates array from remaining columns
firearray = np.array(fireraw)

#sets reference name for all Pokemon with water Type
typewater = pokedata.loc['Water']

#massaging data to remove names and HP to only get Attack and Defense points
waterraw = typewater.drop(columns=['Name','HP'])

#creates array from remaining columns
waterarray = np.array(waterraw)
```

Finally, we need a neutral party that will be the one test out which elemental Pokemon is the strongest. In order, to keep this as accurate as possible the only possible solution to this problem is to utilize a Normal Type Pokemon. The reason a Normal Type Pokemon is use as the individual party our three competitors will fight against with; is because Normal Type Pokemon do not have any multiplier or weakness against our three contesters [5]. Thus, the same process of data analysis will be tested.

```
#sets reference name for all Pokemon with Normal Type
typenormal = pokedata.loc['Normal']

#massaging data to remove names and HP to only get Attack and Defense points
normalraw = typenormal.drop(columns=['Name','HP'])

#creates array from remaining columns
normalarray = np.array(normalraw)
```

III. Analysis

The use of Kmean algorithm was utilize in order to find three clusters. Typically, the starting Pokemon go through two evolution from its initial stage, thus three clusters are necessary to represent this natural progress [2]. Each cluster was set to find the overall distance between Attack and Defense points. Additionally, each cluster represents the overall low average, medium average and high average of given element type.

A. Grass type:

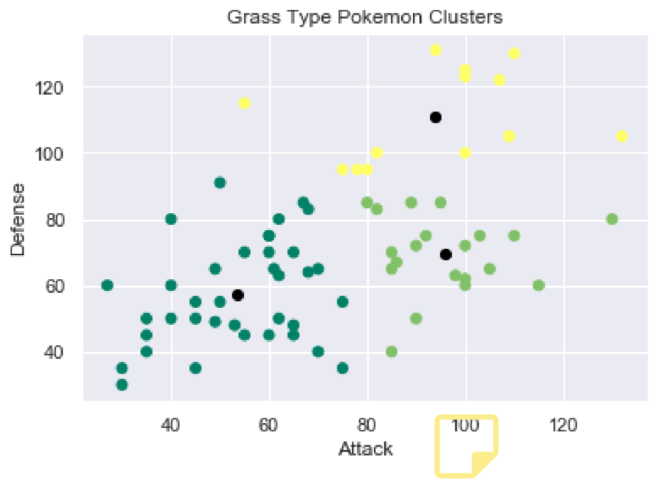
```
#creates 3 clusters for grass Type
grasskmeans = KMeans(n_clusters=3)
grasskmeans.fit(grassrarray)

#saves clusters points for further comparison
grasscluster = grasskmeans.cluster_centers_

print(grasscluster)
```

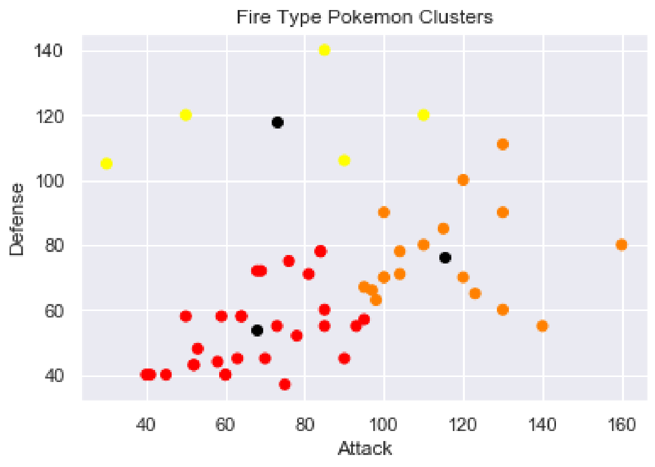
The following results were gathered from the three averages of the grass type:

```
[[ 53.59459459  57.45945946]
 [ 96.          69.45      ]
 [ 94.         110.84615385]]
```

The results show an overall increase in defense from the grass Pokemon. The following code create the image below. The graph shows where do the average cluster point are in comparison to all the grass type Pokemon. The black dots represent the average points of each cluster and the colors themselves represent how big the clusters are and the number of Pokemon in them.

```
plt.title('Grass Type Pokemon Clusters')
plt.xlabel('Attack')
plt.ylabel('Defense')
plt.scatter(grassrarray[:,0],grassrarray[:,1],c=grasskmeans.labels_, cmap='summer')
plt.scatter(grasskmeans.cluster_centers_[:,0],grasskmeans.cluster_centers_[:,1], color='black')
```





B. Fire type:

```
#creates 3 clusters for Fire Type
firekmeans = KMeans(n_clusters=3)
firekmeans.fit(firearray)

#saves clusters points for further comparison
firecluster = firekmeans.cluster_centers_

print(firecluster)
```

The following results were gathered from the three averages of the fire type:

```
[[ 67.82758621  53.86206897]
 [115.33333333  76.16666667]
 [ 73.         118.2       ]]
```

The results show an overall increase in attack from the fire Pokemon. The following code create the image below. The graph shows where do the average cluster point are in comparison to all the grass type Pokemon. The black dots represent the average points of each cluster and the colors themselves represent how big the clusters are and the number of Pokemon in them.

```
plt.title('Fire Type Pokemon Clusters')
plt.xlabel('Attack')
plt.ylabel('Defense')
plt.scatter(firearray[:,0],firearray[:,1],c=firekmeans.labels_,
cmap='autumn')
plt.scatter(firekmeans.cluster_centers_[:,0]
,firekmeans.cluster_centers_[:,1], color='black')
```

C. Water type

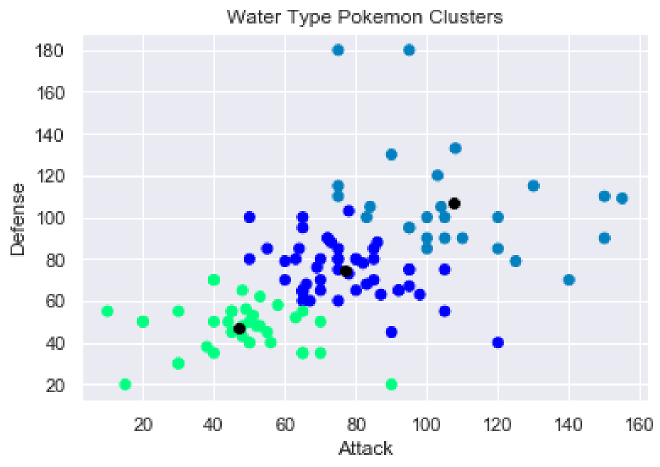
```
#creates 3 clusters for Water Type
waterkmeans = KMeans(n_clusters=3)
waterkmeans.fit(waterarray)

#saves clusters points for further comparison
watercluster = waterkmeans.cluster_centers_

print(watercluster)
```

The following results were gathered from the three averages of the water type:

```
[[ 76.97959184  74.55102041]
 [107.38461538 106.96153846]
 [ 47.05405405  46.91891892]]
```

The results show an overall increase in attack and defense from the water Pokemon. The following code create the image below. The graph shows where do the average cluster point are in comparison to all the grass type Pokemon. The black dots represent the average points of each cluster and the colors themselves represent how big the clusters are and the number of Pokemon in them.

```
plt.title('Water Type Pokemon Clusters')
plt.xlabel('Attack')
plt.ylabel('Defense')
plt.scatter(waterarray[:,0],waterarray[:,1],c=waterkmeans.label
s_, cmap='winter')
plt.scatter(waterkmeans.cluster_centers_[:,0]
,waterkmeans.cluster_centers_[:,1], color='black')
```



D. Normal Type

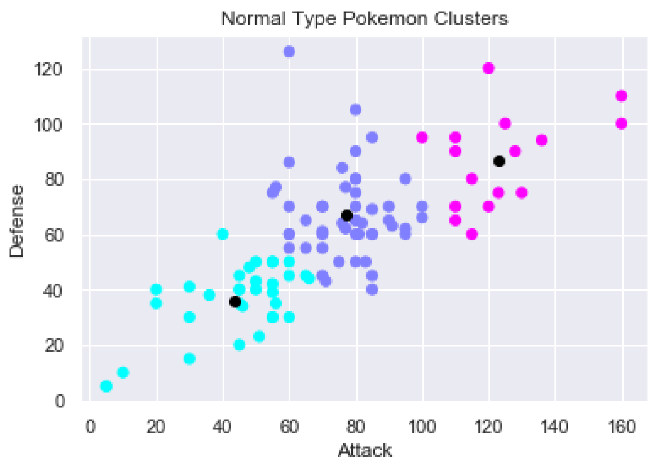
```
#creates 3 clusters for Water Type
normalkmeans = KMeans(n_clusters=3)
normalkmeans.fit(normalarray)

#saves clusters points for further comparison
normalcluster = normalkmeans.cluster_centers_

print(normalcluster)
```

The following results were gathered from the three averages of the normal type:

```
[[ 43.72727273  35.96969697]
 [ 77.24489796  67.12244898]
 [123.25        86.8125    ]]
```

The results show an overall tremendous increase in attack and defense from the normal Pokemon. The following code create the image below. The graph shows where do the average cluster point are in comparison to all the grass type Pokemon. The black dots represent the average points of each cluster and the colors themselves represent how big the clusters are and the number of Pokemon in them.

```
plt.title('Normal Type Pokemon Clusters')
plt.xlabel('Attack')
plt.ylabel('Defense')
plt.scatter(normalarray[:,0],normalarray[:,1],c=normalkmeans.
labels_, cmap='cool')
plt.scatter(normalkmeans.cluster_centers_[:,0]
,normalkmeans.cluster_centers_[:,1], color='black')
```



IV. CONCLUSION

The following research showed that the fire Pokemon has the greatest attack points with a high average of 115. In raw strength the fire Pokemon dominates; however, in defense the grass type Pokemon has shown a tremendous increase. As for the water Pokemon it shown an overall grown in both attack and defense. Incidentally, the analysis also prove that the normal Pokemon far surpasses the strengths of all three types. Thus, the only clear victor of this analysis is the water Pokemon. Its overall growth in both attack and defense shows a Pokemon that can be reliable in any type of battles, that being fighting with raw strength or relying on its defense points to eventually beat its opponent.

REFERENCES

[1] Webster, A. (2019). Pokémon Go’s wild first year: a timeline. [online] TheVerge Available at: https://www.theverge.com/2017/7/6/15888210/pokemon-go-one-year-anniversary-timeline [Accessed 17 Sep. 2019].

[2] Pokemon.com. (2019). Pokédex | Pokemon.com. [online] Available at: https://www.pokemon.com/us/pokedex/ [Accessed 17 Sep. 2019

[3] Barradas, A. (2019). Pokemon with stats. [online] Kaggle.com. Available at: https://www.kaggle.com/abcsds/pokemon [Accessed 17 Sep. 2019].

[4] Data.world. (2019). Pokemon With Stats - dataset by data-society. [online] Available at: https://data.world/data-society/pokemon-with-stats [Accessed 17 Sep. 2019].

[5] Pokemondb.net. (2019). Pokémon Pokédex: a database of stats, information and pictures. [online] Available at: https://pokemondb.net/pokedex [Accessed 17 Sep. 2019].