Optimization for Data Science

POKÉMON GO!

Professor: Ioualitene Fatah
Master of Data Science & Analytics 2019-2020

by
Tran Thi-Thuong-Hoai
Zorana Vukosa
Emmanuel Adeniyi

Kremlin Bicêtre, November 2019

Table of Contents

1. Introduction

Pokémon also known as ***Pocket Monsters*** in Japan, is a media franchise managed by The Pokémon Company, and began life in the mid 90's as a game for the Game Boy. Pokémon started as a Role Playing Game (RPG), however, as time went on, Pokémon has become more and more popular, and so its owners produced various animated TV shows, movies, card games, manga comics, as well as several different video games, like the recently released augmented reality game **Pokémon Go!**. For many players, the game is nostalgic, Pokémon became a phenomenon in the late 90's early 00's. This is also the first time an augmented reality game has come into the mainstream. Young people in particular have been captured by the gaming experience which overlays animated Pokémon characters into the real world. ***Pokémon Go*** is a 2016 augmented reality (AR) mobile game developed and published for iOS and Android devices.

Pokémon are fictitious animal-like monsters that live in the Pokémon world. Pokémon like fighting with each other, and they usually fight according to their (human) trainers' orders. Almost all the Pokémon games include these fights, but in different manners. In some of them the user needs to rely on her or his strategy and in the strength of his or her Pokémon, whereas other video-games are more ability-based. Hence, an interesting fact of this games may well be the way the strength or the ability to fight of a Pokémon is described.

As fans of ***Pokémon Go!***, we made a team of 3 people to work together on Dataset of Pokémon. For statistical analysis purposes, the most attractive way of describing the Pokémon is that of the RPGs. First of all, because a big number of Pokémon have been introduced throughout these years, sixth generations of Pokémon with the order of 100 of Pokémon in each of them. Second, in the RPGs each Pokémon is described with a big number of variables. Not only do we have the combat stats (the variables that describe the ability to fight), but also many variables that describe more details of each Pokémon, e.g. the color or the probability of being female or male. Thus, we can statically analyze the wide variety of variables used to describe the Pokémon, and there is a chance to find relationships between them. In addition, we would write a model to predict if a particular Pokémon is legendary or not. In the rest of the report we will explore the Pokémon and their corresponding variables that appear in the RPGs.

This project is structured according to the following, firstly we will introduce the variables of the dataset as in section 2. Then we will introduce exploratory data analysis, data mining the variables and their potential dependencies, dealing with missing data and frequency encoding in section 3. We will make some predictions to check if a Pokémon is legendary or not using specific data points in our dataset others in Section 4, and with the conclusions in Section 5.

2. Dataset

We selected a Dataset of Pokémon from Kaggle.com. With the recent release of the seventh generation of the Pokémon RGB, the database is already a little out-of-date, since it only includes 21 variables per each of the 721 Pokémon, the ones corresponding to the first six generations, plus the Pokémon ID and its name. Let us now examine the 23 columns of the dataset. The first two are unique identifiers of the Pokémon, the number in the Pokédex and the name. The Pokédex is encyclopedia-like tool that can be used in the Pokémon RGBs to get information of the Pokémon. In fact, most of the variables we will use in this work are taken from the Pokédex. We will explain in more detail what these variables represent next.

Variables

- **Type_1.** Primary type of the Pokémon. It is related to the nature, with its lifestyle and with the movements it is able to learn for the fighting time. This categorical value can take 18 different values: Bug, Dark, Dragon, Electric, Fairy, Fighting, Fire, Flying, Ghost, Grass, Ground, Ice, Normal, Poison, Psychic ,Rock, Steel, and Water.
- **Type_2.** Pokémon can have two types, but not all of them do. The possible values this secondary type can take are the same as the variable Type_1.
- **Total.** The sum of all the base battle stats of a Pokémon. It should be a good indicator of the overall strength of a Pokémon. It is the sum of the next six variables. Each of them represents a base battle stat. All the battle stats are continuous yet integer variables, i.e. the number of values they can take is infinite in theory, or just very big in the practice.
- **HP.** Base health points of the Pokémon. The bigger it is, the longer the Pokémon will be able to stay in a fight before they faint and leave the combat.
- **Attack.** Base attack of the Pokémon. The bigger it is, the more damage its physical attacks will deal to the enemy Pokémon.
- **Defense**. Base defense of the Pokémon. The bigger it is, the less damage it will receive when being hit by a physical attack.
- **Sp_Atk.** Base special attack of the Pokémon. The bigger it is, the more damage its special attacks will deal to the enemy Pokémon.
- **Sp_Def.** Base special defense of the Pokémon. The bigger it is, the less damage it will receive when being hit by a special attack.
- **Speed.** Base speed of the Pokémon. The bigger it is, the more times the Pokémon will be able to attack the enemy.
- **Generation.** The generation where the Pokémon was released. It is an integer between 1 and 6, so it is a numerical discrete variable. It could let us analyze the development or the growth of the game through the years.
- **isLegendary.** Boolean indicating whether the Pokémon is legendary or not. Legendary Pokémon tend to be stronger, to have unique abilities, to be really hard to find, and even harder to catch.
- **Color.** Color of the Pokémon according to Pokédex. The Pokédex distinguishes between ten colors: Black, Blue, Brown, Green, Grey, Pink, Purple, Red, White, and Yellow.
- **hasGender.** Boolean indicating the Pokémon can be classified as male or female.
- **Pr_Male.** In case the Pokémon has Gender, the probability of its being male. The probability of being female is, of course, 1 minus this value. Like Generation, this variable is numerical and discrete, because although it is the probability of the Pokémon to appear as a female or male in nature, it can only take 7 values: 0, 0.125, 0.25, 0.5, 0.75, 0.875, and 1.

- **Egg_Group_1.** Categorical value indicating the egg group of Pokémon. It is related with the race of the Pokémon, and it is a determinant factor in the breeding of the Pokémon. Its 15 possible values are: Amorphous, Bug, Ditto, Dragon, Fairy, Field, Flying, Grass, Human-Like, Mineral, Monster, Undiscovered, Water_1,Water_2, andWater_3.
- **Egg_Group_2.** Similarly to the case of the Pokémon types, Pokémon can belong to two egg groups.
- **hasMegaEvolution.** Boolean indicating whether a Pokémon can mega-evolve or not. Mega-evolving is property that some Pokémon have and allows them to change their appearance, types, and stats during a combat into a much stronger form.
- **Height_m.** Height of the Pokémon according to Pokédex, measured in meters. It is a numerical continuous variable.
- **Weight_kg.** Weight of the Pokémon according to the Pokédex, measured kilograms. It is also a numerical continuous variable.
- **Catch_Rate.** Numerical variable indicating how easy is to catch a Pokémon when trying to capture it to make it part of your team. It is bounded between 3 and 255. The number of different values it takes is not too high notwithstanding, we can consider it is a continuous variable.
- **Body_Style.** Body style of the Pokémon according to the Pokédex. 14 categories of body style are specified: bipedal_tailed, bipedal_tailless, four_wings, head_arms, head_base, head_legs, head_only, insectoid,multiple_bodies, quadruped, serpentine_body, several_limbs, two_wings, and with_fins.

3. Exploratory Data Analysis(EDA)

3.1 Integer variables

In this section, we will try to gain an insight of the distributions of the different integer values. In case the distributions are non-normally distributed we will attempt to transform them in order to ensure the better fitting of the algorithms. In case the scales of the variables are multiple orders of magnitude apart we will subject them to scaling and/or transformation.

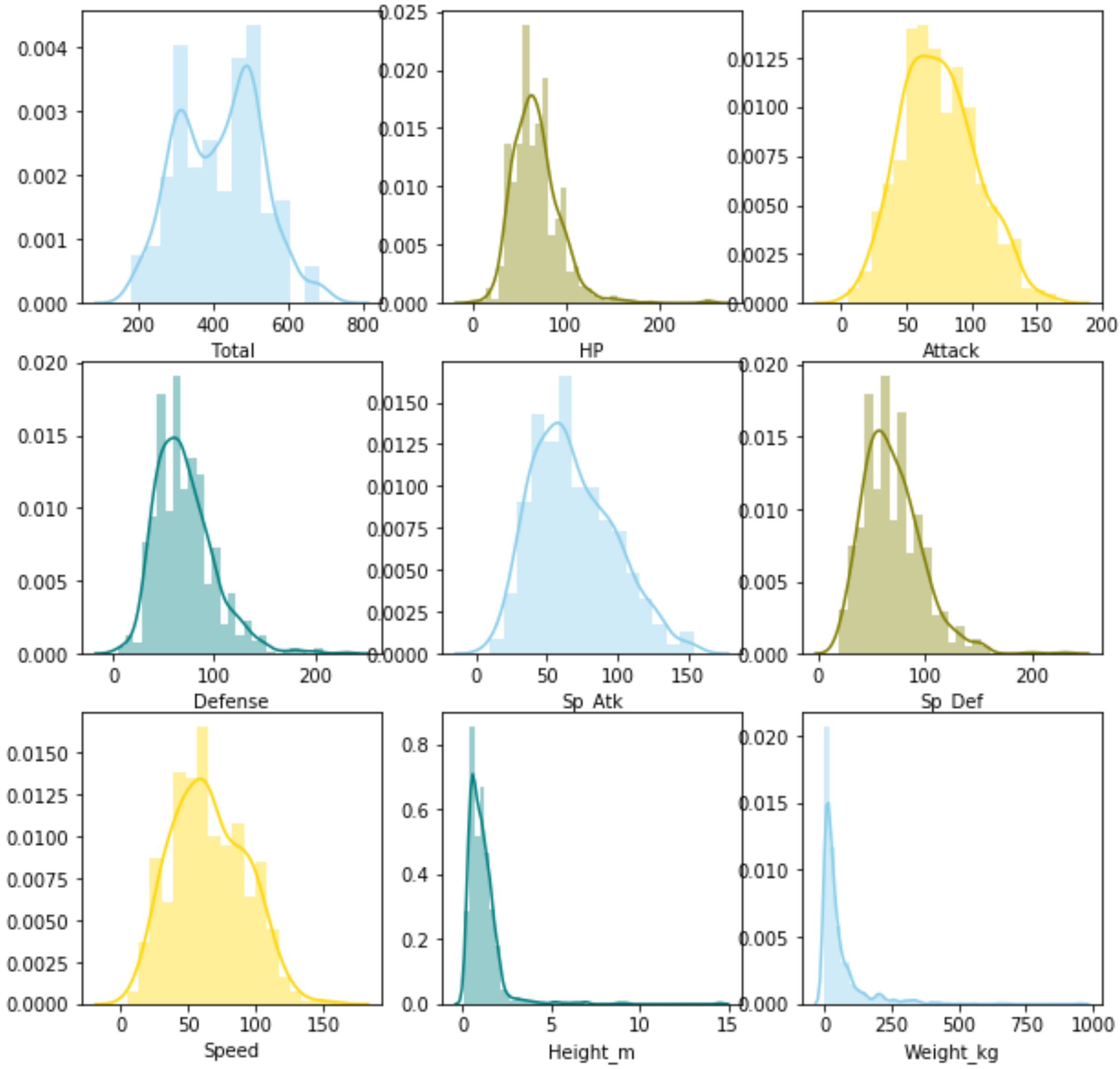
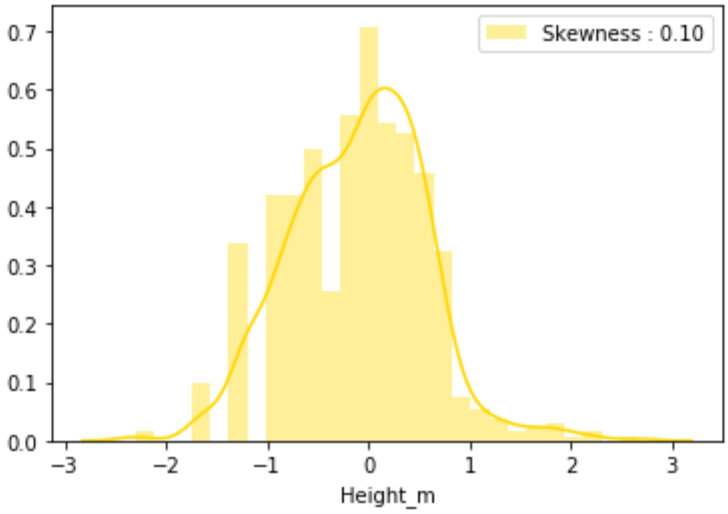


Figure 1: From left to right, and from top to bottom, histograms of Total, HP, Attack, Defense, Sp Atk, Sp Def, Speed, Height_m, Weight_kg

From the Figure 1. we can see that the variables “Height_m” and “Weight_kg” are skewed and the measure for their skewness is 5.51 and 4.01 respectively. For the “Total” feature we can notice a bimodal distribution, but since it is the sum of HP, Attack, Defense, Sp Atk, Sp Def, Speed which are all normally distributed and the “bimodality” is not too strong we decided to continue without transforming this variable. For the “Height_m” and “Weight_kg” we decided to try out the log-transformation with the hope that it will scale the data (so these two variables are in approximately the same range) and that the distribution will start looking more like a normal distribution. In order to see it better, we show Height_m and Weight_kg again with new skewness is 0.10 and -0.51 respectively in Figure 2.



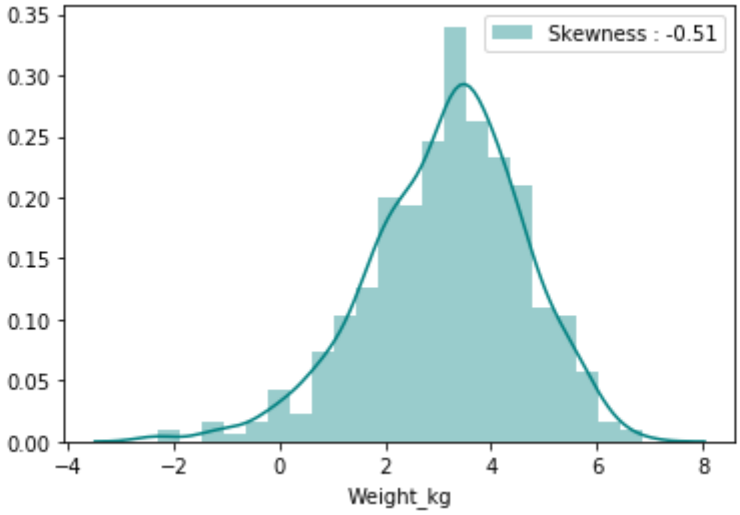


Figure 2: Histograms of Height (yellow) and Weight (Blue) of Pokémon

3.2 Categorical variables

In this section, we will plot all the categorical variables to get some insights into their distributions. This will also help us in determining the best strategy for the categorical variables encoding. We will follow the rule:

A) ONE HOT ENCODING - if there is a small number of different unique values a class can take
B) FREQUENCY ENCODING - there is a large number of different unique values a class can take

After loading in the dataset, preliminary analyses and plots were run to visualize the distribution of Pokémon in the dataset based on their ***Type_1*** (Figure 3). Type is a fairly important aspect of Pokémon as Pokémon types determine special attributes that determine the strengths and weaknesses of different Pokémon species. From the visualization, it appears that ***water*** is a very popular type of Pokémon.

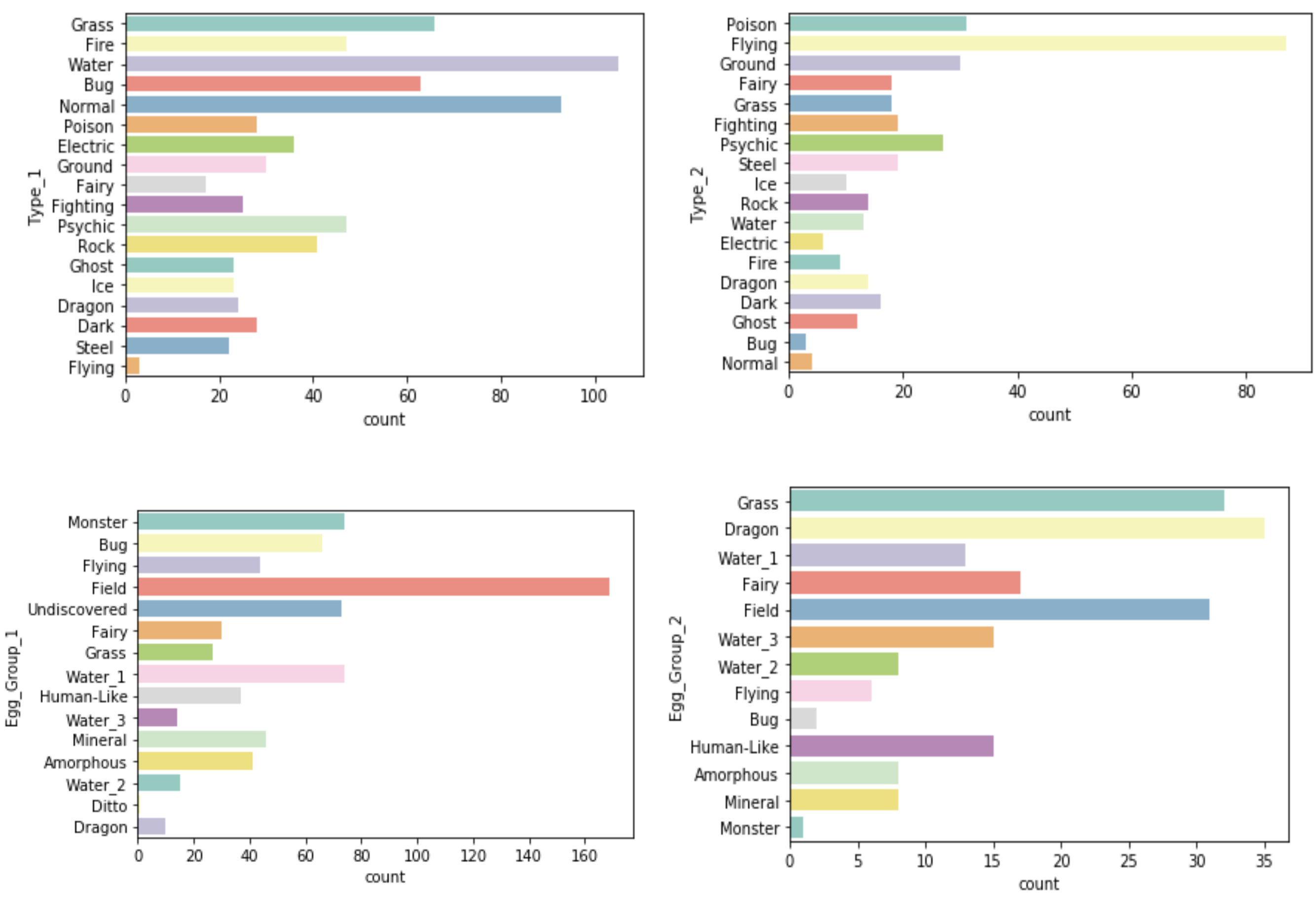
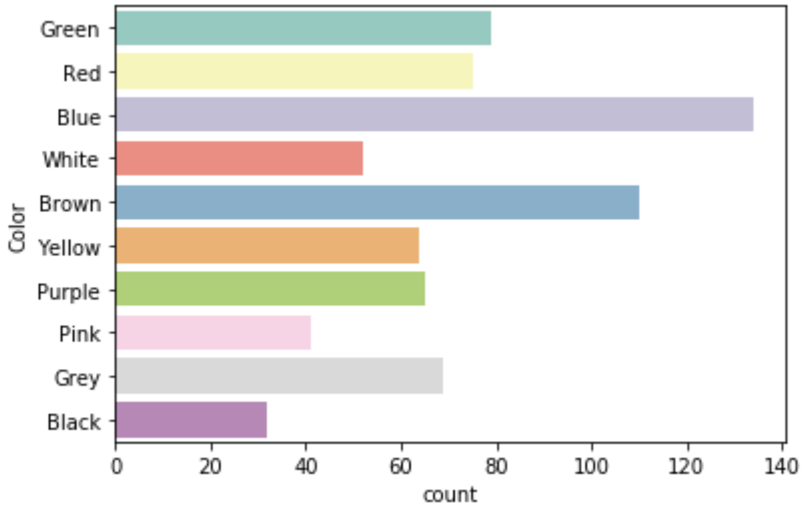


Figure 3: From left to right, and from top to bottom, histograms of the primary and secondary types, the first and second egg groups of the Pokémon

As we can see, *Figure 3* shows the histograms of the primary and secondary types, as well as the first and second egg groups of the Pokémon. The most common primary types are Water, Normal, and Grass, while the most common secondary type is Flying, because most of the times a Pokémon is able to fly the other type is considered first. We can also see that more or less half of the Pokémon do not have any secondary type. We can also look for the more and less common egg groups in the histograms shown. We can observe that the most common egg groups (that can somehow be understood like races) are *Field*, followed by *Water_{1+2+3}, Monster, Undiscovered,* and *Bug*. Pokémon from the *Field* egg group tend to be terrestrial creatures. Pokémon from the three water egg groups live in or around the *water,* and *Monste*r group Pokémon are usually among the most powerful. *Undiscovered* egg group is characterized by its members' inability to breed. Most of the Pokémon in the group are baby Pokémon, or legendary Pokémon.



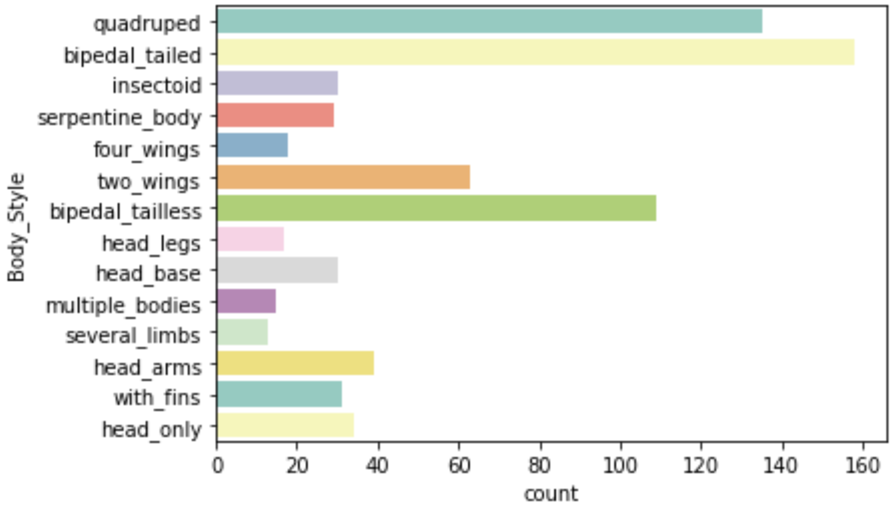


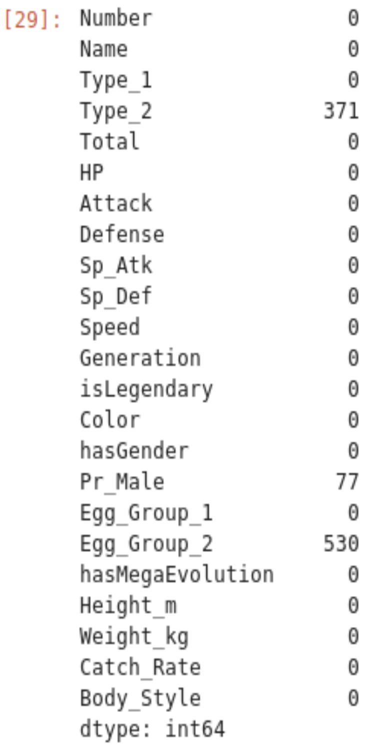
Figure 4: Histograms of the color (left) and Body style (right) of the Pokémon.

Plotting all of the categorical variables shows us that for all of the classes there is in the class frequencies are quite different and maybe that is a feature we want to preserve while transforming the categorical variables. Using frequency encoding will also reduce the number of dimensions as opposed to one-hot encoding where each value a class can take becomes a new variable. The missing values in “Type_2” and “Egg_Group_2” will be replaced by zero. For the variable “Pr_male” we will just fill the missing values with 0, because from the description it indicates that a Pokémon does not have a male form.

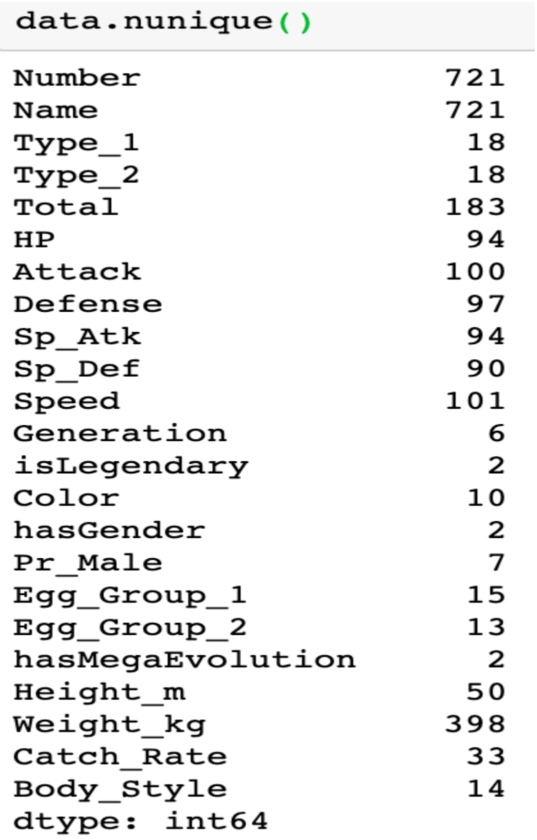
3.3 Dealing with Missing Data

Our machine learning model cannot accept null/NaN values, we needed to either remove them or fill them with a logical value. To investigate how many nulls, we have in each column. We had 3 options. Drop the missing values, fill the missing values with test static or predict the missing values with a machine learning algorithm.

First step is to detect the missing values with EDA then showing the sum of the null data in the dataset.

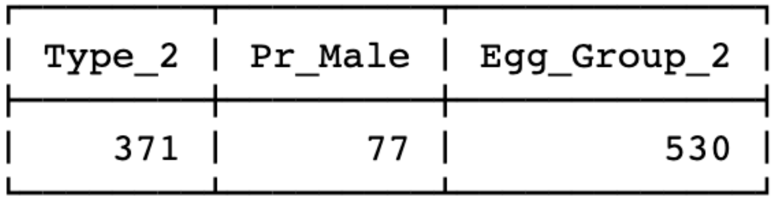


```
[29]: Number                0
      Name                  0
      Type_1                0
      Type_2              371
      Total                 0
      HP                    0
      Attack                0
      Defense               0
      Sp_Atk                0
      Sp_Def                0
      Speed                 0
      Generation            0
      isLegendary           0
      Color                 0
      hasGender             0
      Pr_Male              77
      Egg_Group_1           0
      Egg_Group_2         530
      hasMegaEvolution      0
      Height_m              0
      Weight_kg             0
      Catch_Rate            0
      Body_Style            0
      dtype: int64
```



```
data.nunique()

Number              721
Name                721
Type_1               18
Type_2               18
Total               183
HP                   94
Attack              100
Defense              97
Sp_Atk               94
Sp_Def               90
Speed               101
Generation            6
isLegendary           2
Color                10
hasGender             2
Pr_Male               7
Egg_Group_1          15
Egg_Group_2          13
hasMegaEvolution      2
Height_m             50
Weight_kg           398
Catch_Rate           33
Body_Style           14
dtype: int64
```



Type_2	Pr_Male	Egg_Group_2
371	77	530

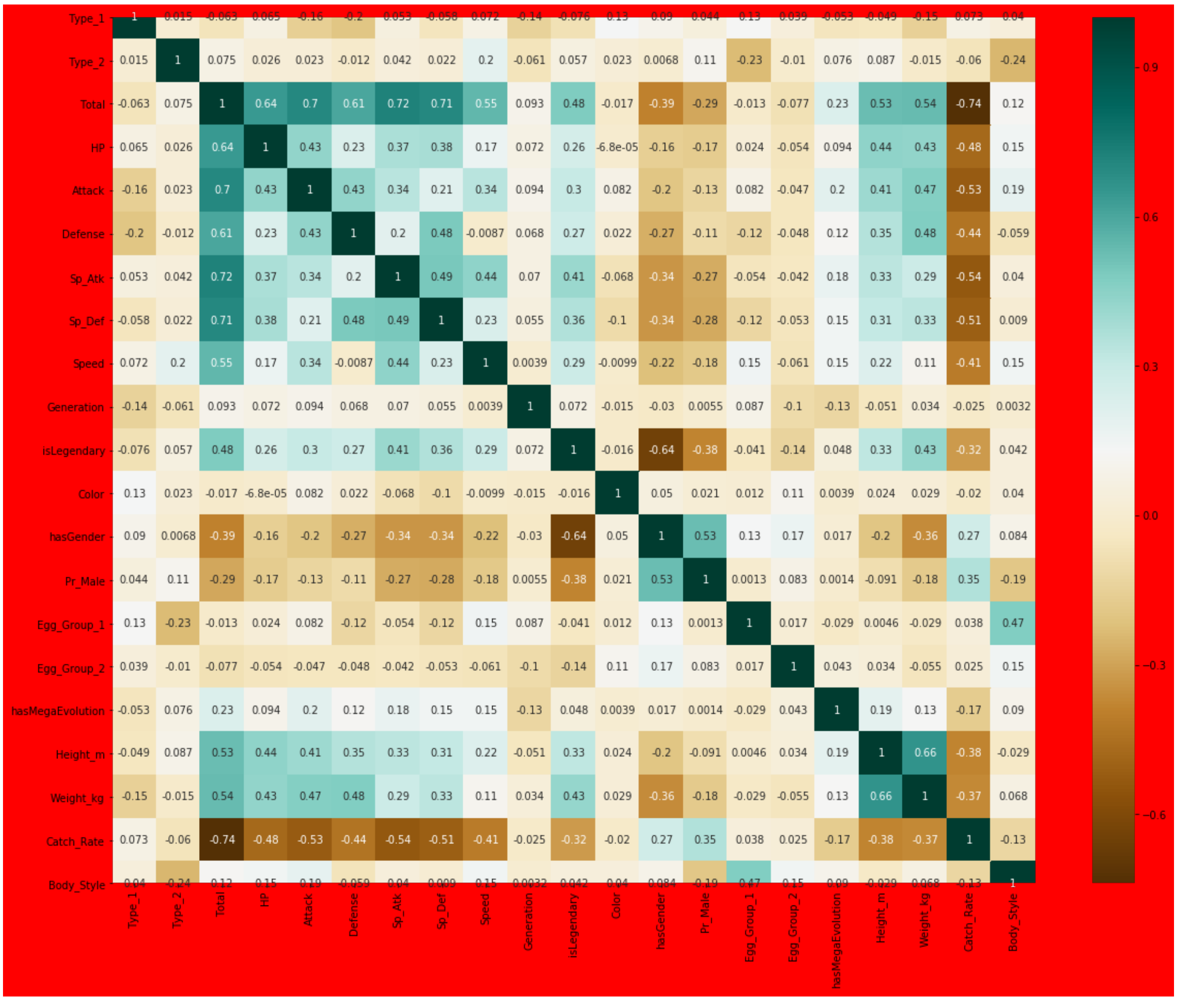
From the picture, above, it can be seen that we have 3 rows with missing data, Type_2, Pr_Male and Egg_group 2. Populating the missing data with values was done using assertion to check and fill in missing rows with zero.

3.4 Frequency encoding

The idea of encoding the categorical variables with the use of the target variable (continuous or categorical depending on the task). This type of encoding is called frequency, likelihood encoding or target encoding. In this case for independent variables Type_1, Type_2, Color, Egg_Group1, Egg_Group_2 and body_style. Frequency encoding is not plain mean. I.e we take the total amount of times a particular value appears in our dataset and divide it by the total amount of values present in our dataset. Take for instance we have 4 blue values out of 20 total values, we divide 4 by 20 then replace every appearance of blue with 0.2

For the boolean values we alternated the values between the 1's and 0’s. 1’s for true and 0’s for false. This includes hasMegaEvolution, hasGender and isLegendary.

3.5 Correlation between variables



	Type_1	Type_2	Total	HP	Attack	Defense	Sp_Atk	Sp_Def	Speed	Generation	isLegendary	Color	hasGender	Pr_Male	Egg_Group_1	Egg_Group_2	hasMegaEvolution	Height_m	Weight_kg	Catch_Rate	Body_Style
Type_1	1	0.015	-0.063	0.065	-0.16	-0.2	0.053	-0.058	0.072	-0.14	-0.076	0.13	0.09	0.044	0.13	0.039	-0.053	-0.049	-0.15	0.073	0.04
Type_2	0.015	1	0.075	0.026	0.023	-0.012	0.042	0.022	0.2	-0.061	0.057	0.023	0.0068	0.11	-0.23	-0.01	0.076	0.087	-0.015	-0.06	-0.24
Total	-0.063	0.075	1	0.64	0.7	0.61	0.72	0.71	0.55	0.093	0.48	-0.017	-0.39	-0.29	-0.013	-0.077	0.23	0.53	0.54	-0.74	0.12
HP	0.065	0.026	0.64	1	0.43	0.23	0.37	0.38	0.17	0.072	0.26	-6.8e-05	-0.16	-0.17	0.024	-0.054	0.094	0.44	0.43	-0.48	0.15
Attack	-0.16	0.023	0.7	0.43	1	0.43	0.34	0.21	0.34	0.094	0.3	0.082	-0.2	-0.13	0.082	-0.047	0.2	0.41	0.47	-0.53	0.19
Defense	-0.2	-0.012	0.61	0.23	0.43	1	0.2	0.48	-0.0087	0.068	0.27	0.022	-0.27	-0.11	-0.12	-0.048	0.12	0.35	0.48	-0.44	-0.059
Sp_Atk	0.053	0.042	0.72	0.37	0.34	0.2	1	0.49	0.44	0.07	0.41	-0.068	-0.34	-0.27	-0.054	-0.042	0.18	0.33	0.29	-0.54	0.04
Sp_Def	-0.058	0.022	0.71	0.38	0.21	0.48	0.49	1	0.23	0.055	0.36	-0.1	-0.34	-0.28	-0.12	-0.053	0.15	0.31	0.33	-0.51	0.009
Speed	0.072	0.2	0.55	0.17	0.34	-0.0087	0.44	0.23	1	0.0039	0.29	-0.0099	-0.22	-0.18	0.15	-0.061	0.15	0.22	0.11	-0.41	0.15
Generation	-0.14	-0.061	0.093	0.072	0.094	0.068	0.07	0.055	0.0039	1	0.072	-0.015	-0.03	0.0055	0.087	-0.1	-0.13	-0.051	0.034	-0.025	0.0032
isLegendary	-0.076	0.057	0.48	0.26	0.3	0.27	0.41	0.36	0.29	0.072	1	-0.016	-0.64	-0.38	-0.041	-0.14	0.048	0.33	0.43	-0.32	0.042
Color	0.13	0.023	-0.017	-6.8e-05	0.082	0.022	-0.068	-0.1	-0.0099	-0.015	-0.016	1	0.05	0.021	0.012	0.11	0.0039	0.024	0.029	-0.02	0.04
hasGender	0.09	0.0068	-0.39	-0.16	-0.2	-0.27	-0.34	-0.34	-0.22	-0.03	-0.64	0.05	1	0.53	0.13	0.17	0.017	-0.2	-0.36	0.27	0.084
Pr_Male	0.044	0.11	-0.29	-0.17	-0.13	-0.11	-0.27	-0.28	-0.18	0.0055	-0.38	0.021	0.53	1	0.0013	0.083	0.0014	-0.091	-0.18	0.35	-0.19
Egg_Group_1	0.13	-0.23	-0.013	0.024	0.082	-0.12	-0.054	-0.12	0.15	0.087	-0.041	0.012	0.13	0.0013	1	0.017	-0.029	0.0046	-0.029	0.038	0.47
Egg_Group_2	0.039	-0.01	-0.077	-0.054	-0.047	-0.048	-0.042	-0.053	-0.061	-0.1	-0.14	0.11	0.17	0.083	0.017	1	0.043	0.034	-0.055	0.025	0.15
hasMegaEvolution	-0.053	0.076	0.23	0.094	0.2	0.12	0.18	0.15	0.15	-0.13	0.048	0.0039	0.017	0.0014	-0.029	0.043	1	0.19	0.13	-0.17	0.09
Height_m	-0.049	0.087	0.53	0.44	0.41	0.35	0.33	0.31	0.22	-0.051	0.33	0.024	-0.2	-0.091	0.0046	0.034	0.19	1	0.66	-0.38	-0.029
Weight_kg	-0.15	-0.015	0.54	0.43	0.47	0.48	0.29	0.33	0.11	0.034	0.43	0.029	-0.36	-0.18	-0.029	-0.055	0.13	0.66	1	-0.37	0.068
Catch_Rate	0.073	-0.06	-0.74	-0.48	-0.53	-0.44	-0.54	-0.51	-0.41	-0.025	-0.32	-0.02	0.27	0.35	0.038	0.025	-0.17	-0.38	-0.37	1	-0.13
Body_Style	0.04	-0.24	0.12	0.15	0.19	-0.059	0.04	0.009	0.15	0.0032	0.042	0.04	0.084	-0.19	0.47	0.15	0.09	-0.029	0.068	-0.13	1

Figure 5: Correlation matrix of dataset

The table and the correlation figures provide information in the form of a heat map, the diagonal of the plot of all correlations are 0.9, which is perfectly positively correlated. This is because the diagonal compares each feature to itself. The top half above the diagonal provides the same information as the lower half. While it might be interesting to look at how the independent features are correlated. Looking at the correlations to answer the following questions

1. **Can we predict the 'isLegendary' variable from other variables?**
2. **Can we predict 'isLegendary' using the rest of the data?**

4. Can we predict ‘isLegendary’ using the rest of the data?

4.1 Analysis

From the correlation matrix, we can see that the features that have the greatest correlation with the ‘isLegendary’ are these:

'Catch_Rate', 'Total', "Sp_Atk", 'hasGender', 'Weight_kg', 'Catch_Rate'

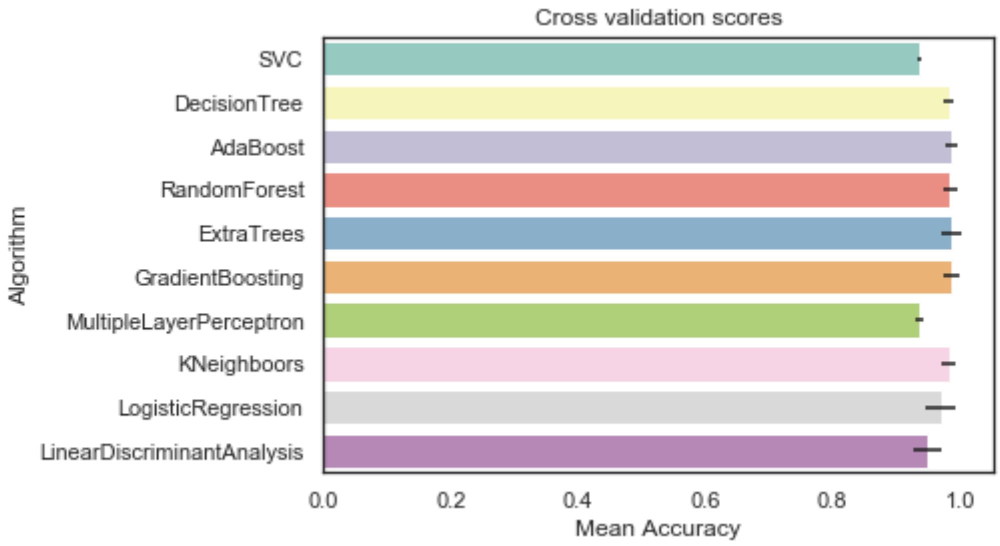
They are also to some extent correlated to each other, but we will nevertheless use all of them and test what the outcome is.

We will try the accuracy of these 10 simple models

I compared 10 popular classifiers and evaluate the mean accuracy of each of them by a stratified K-Fold cross validation procedure.

* SVC
* Decision Tree
* AdaBoost
* Random Forest
* Extra Trees
* Gradient Boosting
* Multiple layer perceptron (neural network)
* KNN
* Logistic regression
* Linear Discriminant Analysis

The result:



In numbers:

[0.9358431516936673, 0.9833762886597939, 0.9875, 0.985438144329897, 0.987563994670033, 0.9875214776632303, 0.9379264850270005, 0.9834398450101691, 0.9709188056665965, 0.9501924223297566]

4.2. Hyperparameter tuning for Adaboost, Extra-trees and gradient boosting

We decided to try out the hyperparameter tuning in order to test if we could further improve the accuracy given by the default classifier parameters.

4.2.1. AdaBoost

For the Adaboost we checked these parameters:

PARAMETER	VALUES
base_estimator__criterion	"gini", "entropy"
base_estimator__splitter	"best", "random"
algorithm	"SAMME","SAMME.R"
n_estimators	1,2
learning_rate	0.0001, 0.001, 0.01, 0.1, 0.2, 0.3,1.5

and we saw insignificant increase on the training set from 0.9875 - 0.9875776397515528

4.2.2. Extratrees

For the Extra Trees, we checked these parameters:

PARAMETER	VALUES
max_depth	None
max_features	1, 3, 10

min_samples_split	2, 3, 10
min_samples_leaf	1, 3, 10
bootstrap	False
n_estimators	100,300
criterion	"gini"

And we noticed an increase from 0.987563994670033 to 0.989648033126294.

4.2.3. Gradient Boosting

For the Adaboost we checked these parameters:

PARAMETER	VALUE
loss	“deviance”
n_estimators	100,200,300
learning_rate	0.1, 0.05, 0.01
max_depth	4, 8
min_samples_leaf	100, 150
max_features	0.3, 0.1

We saw an increase from 0.9875214776632303 to 0.989648033126294.

4.3. Testing the accuracy on the test data

The final accuracy rate on test set is as follows:

Accuracy of Extratrees: 0.9831932773109243
Accuracy of AdaBoost: 0.9747899159663865
Accuracy of GradientBoosting: 0.9747899159663865

We see that the Extratrees classifier performs the best. We are also happy that the models were just slightly overfitting with just 0.01 - 0.005 difference. The prediction of all three classifiers is quite close and we cannot guarantee that with a slightly different test set Extratrees would be the best. What we can almost surely say is that they would be performing similarly because this is the result that has been repeated from the cross-validation score. Furthermore, our test and train sets where prior stratified with respect to ‘isLegendary’ feature.
These are the mean values of the cross_validation scores over the 5 cross-validation splits. All the classifiers seem to be performing very well, above 92%.

7. Conclusion

In this work a statistical analysis of the ***Pokémon Go!*** dataset, we performed analysis for all the variables in the dataset that we have previously built. We have dealt with missing data, analyzed how the numerical variables are distributed and all of them showed a behavior. We have also used histograms to visualize the data. We also tried to find the correlations between the numerical values. We have carried out integer variables and categorical variables analysis for all the variables in the dataset that we have previously built. We have analyzed how the numerical variables are distributed, and all of them have shown to have.

Making use of some reasonable clusters available with the help of principal component analysis we have gained an insight into the properties of this cluster. In general, the first cluster is formed from the weakest and smallest Pokémon, the second one includes strong and hard to catch Pokémon. The remaining one is the smallest, yet it includes the heaviest highest and strongest Pokémon. Finally, we have built predictive models using SVC, Decision Tree, AdaBoost, Random Forest, Extra Trees, Gradient Boosting, Multiple layer perceptron (neural network), KNN, Logistic regression, Linear Discriminant Analysis We tried to predict if a Pokémon is legendary or not based on its numerical values. Predicting with the independent variables. Having incredible prediction accuracy, the fitting was precise and the analysis to predict whether a Pokémon is legendary or not, which has worked quite well.