Pokemon Data Analysis

Data Mining Lab Part - C

TEAM MEMBERS

1.RITHIKA MEHTA (1MS17IS092)

2. SHIRSHVARDHAN KASHYAP (1MS17IS112)

3. TANISHA SABHERWAL (1MS17IS146)

Content

Abstract

Pokémon is media franchise that began as a pair of Role Playing Games (RPG) video games for the original Game Boy that was developed by Game Freak and published by Nintendo, in 1996. Pokémon are fictitious animal-like monsters that live in the (of course, also invented) 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 his strategy and in the strength of his or her Pokémon, whereas other video-games are more ability-based.

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 -seven 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 Pokemon, 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, and also to cluster the Pokémon according to some criteria. In the rest of the report we will explore the Pokémon and their corresponding variables that appear in the RPGs.

Introduction

Dataset Description

There are 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. From the resting 21 variables, 12 are numerical (10 continuous and 2 discrete), 6 categorical and 3 boolean.

•Type_1. Primary type of the Pokémon. It is related 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 than 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 to 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 to be even harder to catch.

- Color. Color of the Pokémon according to the 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 the 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 the 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, and Water_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 the 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.

Data Preprocessing

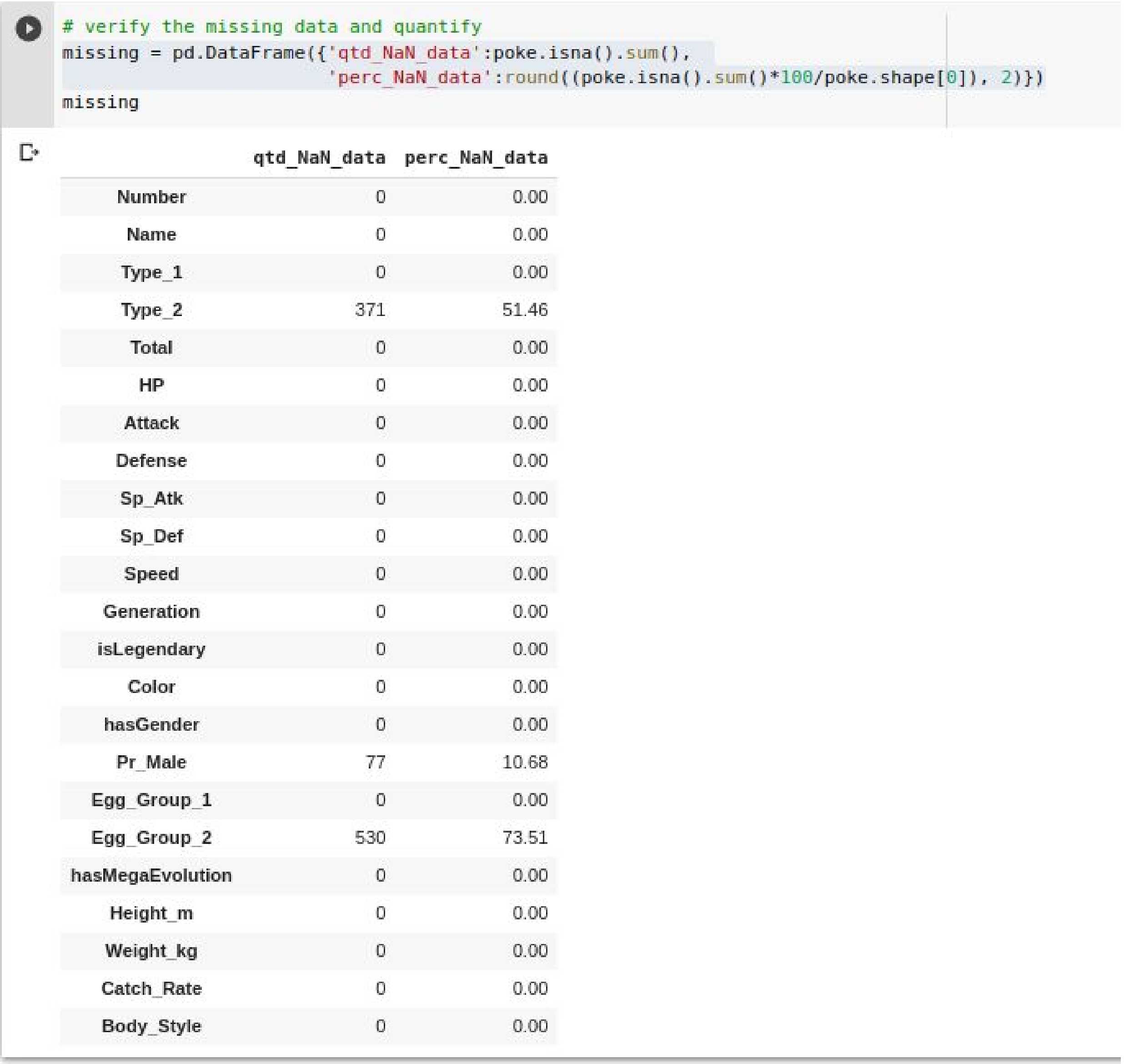
Initially the dataset contained unwanted columns and the rows contained NULL values. Also the values were skewed and contained duplicates.

So the first step would be to remove unwanted columns

Step 1: Drop unnecessary columns

```
df = df.drop('Number', axis=1)
```

Step 2 : Check Null Values



```
# verify the missing data and quantify
missing = pd.DataFrame({'qtd_NaN_data':poke.isna().sum(),
                        'perc_NaN_data':round((poke.isna().sum()*100/poke.shape[0]), 2)})
missing
```

	qtd_NaN_data	perc_NaN_data
Number	0	0.00
Name	0	0.00
Type_1	0	0.00
Type_2	371	51.46
Total	0	0.00
HP	0	0.00
Attack	0	0.00
Defense	0	0.00
Sp_Atk	0	0.00
Sp_Def	0	0.00
Speed	0	0.00
Generation	0	0.00
isLegendary	0	0.00
Color	0	0.00
hasGender	0	0.00
Pr_Male	77	10.68
Egg_Group_1	0	0.00
Egg_Group_2	530	73.51
hasMegaEvolution	0	0.00
Height_m	0	0.00
Weight_kg	0	0.00
Catch_Rate	0	0.00
Body_Style	0	0.00

Missing data

- 371 has no second type
- 77 has no Pr_Male
- 530 has no Egg_Group_2

```
df = df.fillna(df.median())
```

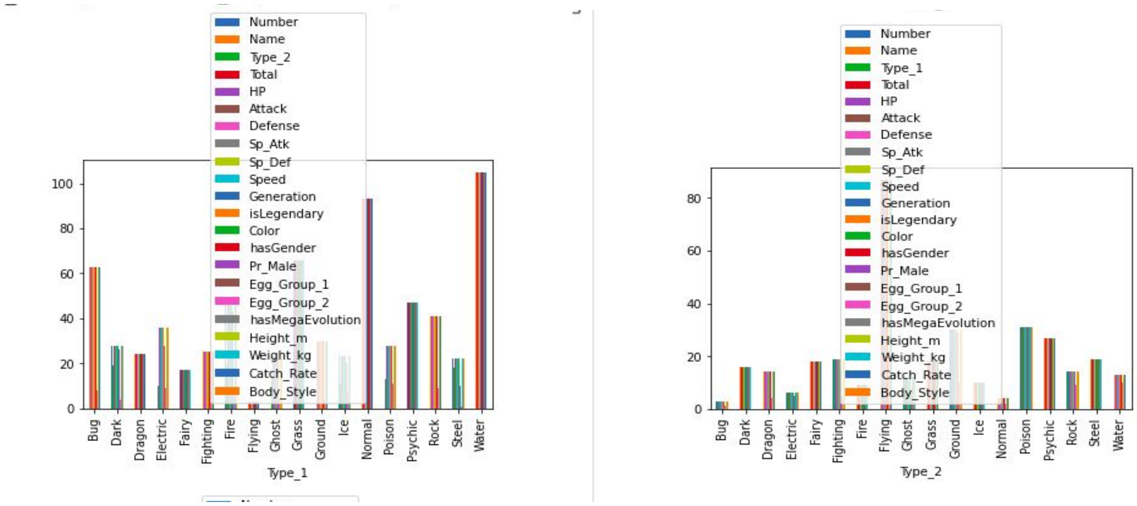
Step 3: Handling categorical variables

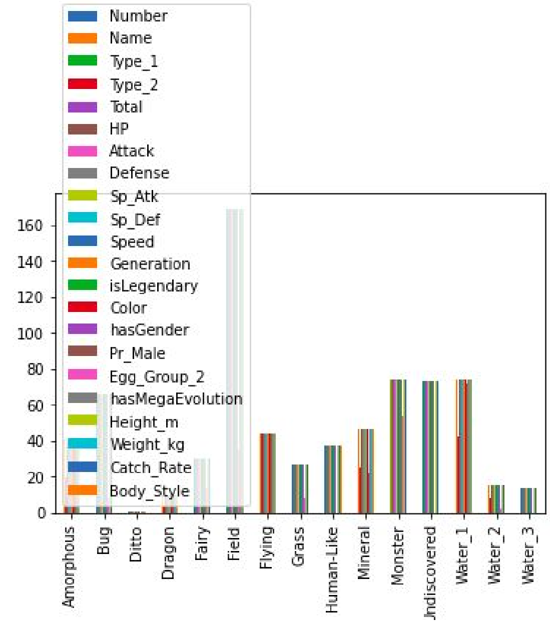
The speed belongs to multiple categories which are highly interleaved hence the quality was split into 2 categories called “low” and “high” based on the threshold as mean values. If quality was higher than mean, then it was categorized as “high” and others as “low”.

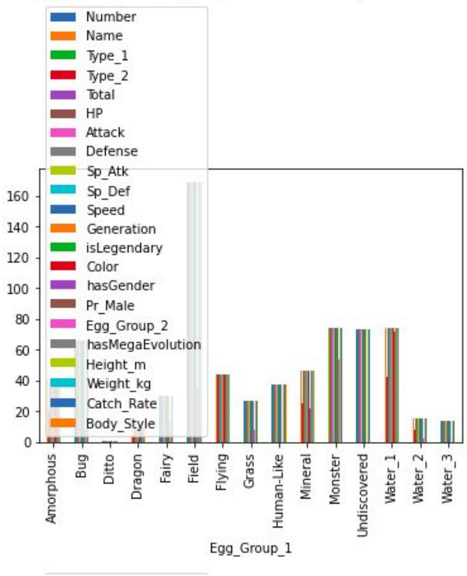
Data Visualization

Here we have visualized different plots between the features and correlation between different variables.

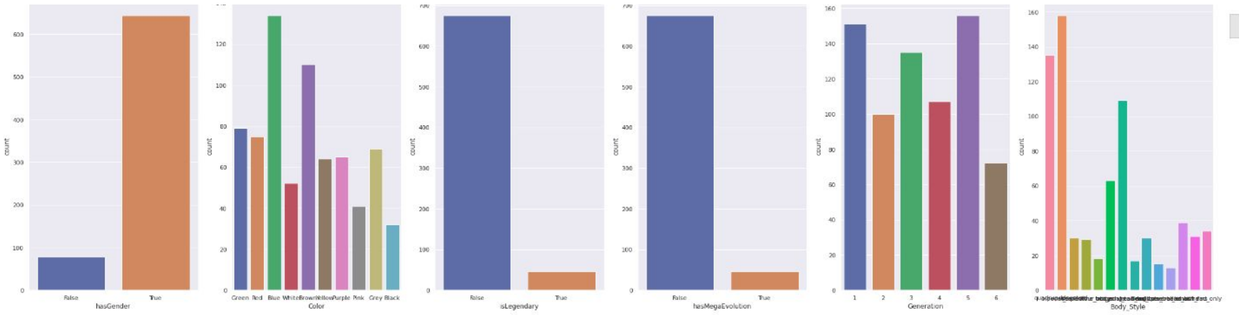
Unvariate Analysis



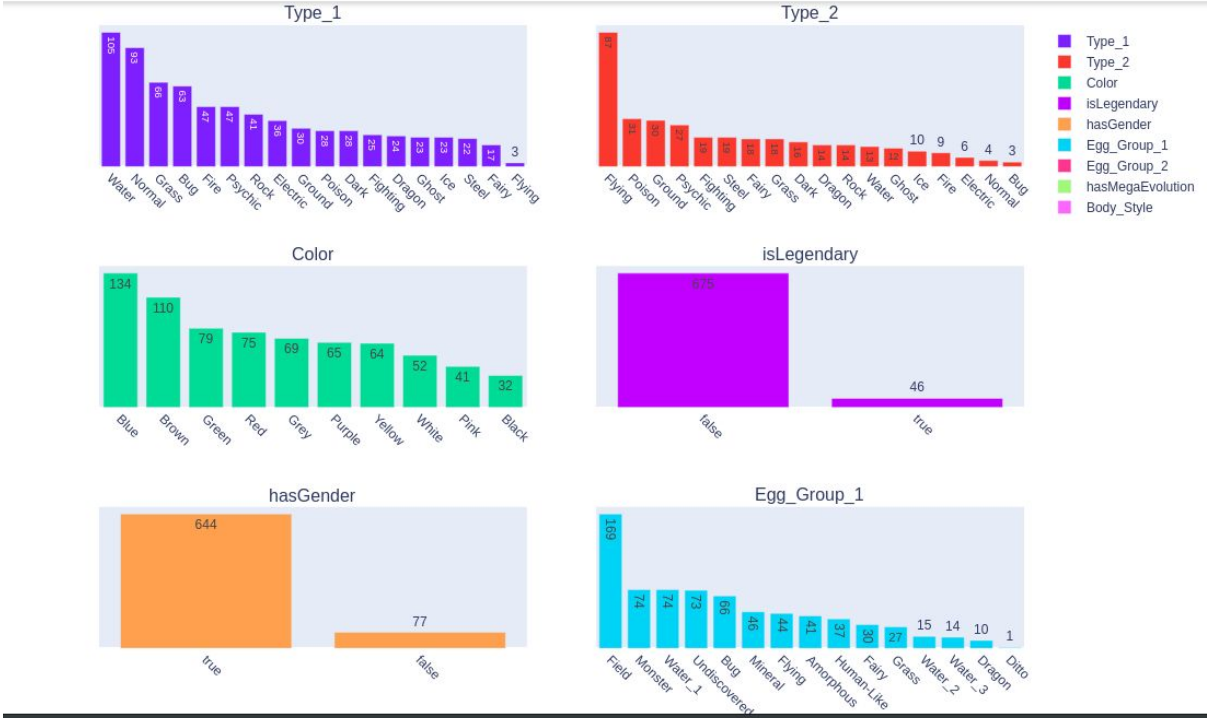




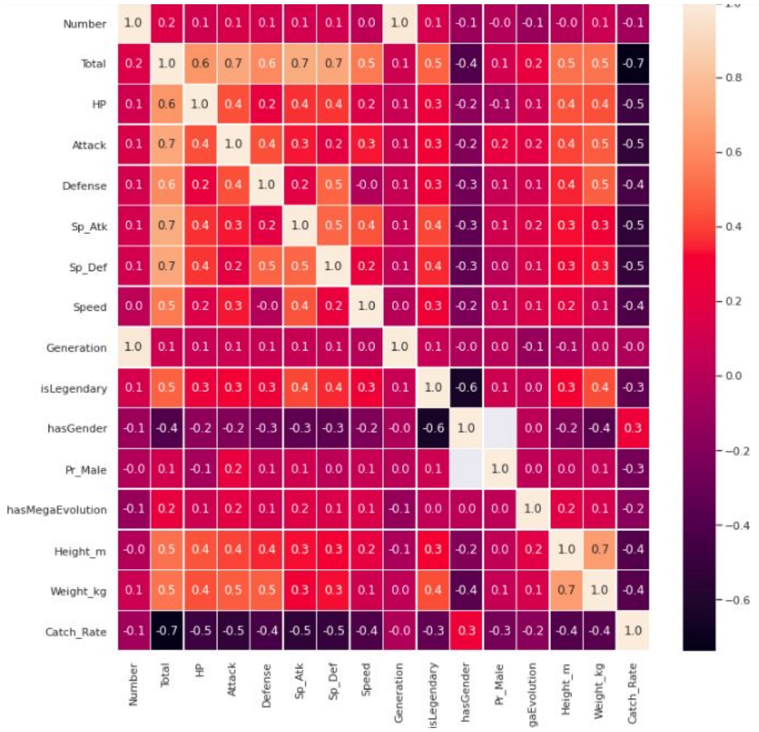
Relations and Dependencies between variables



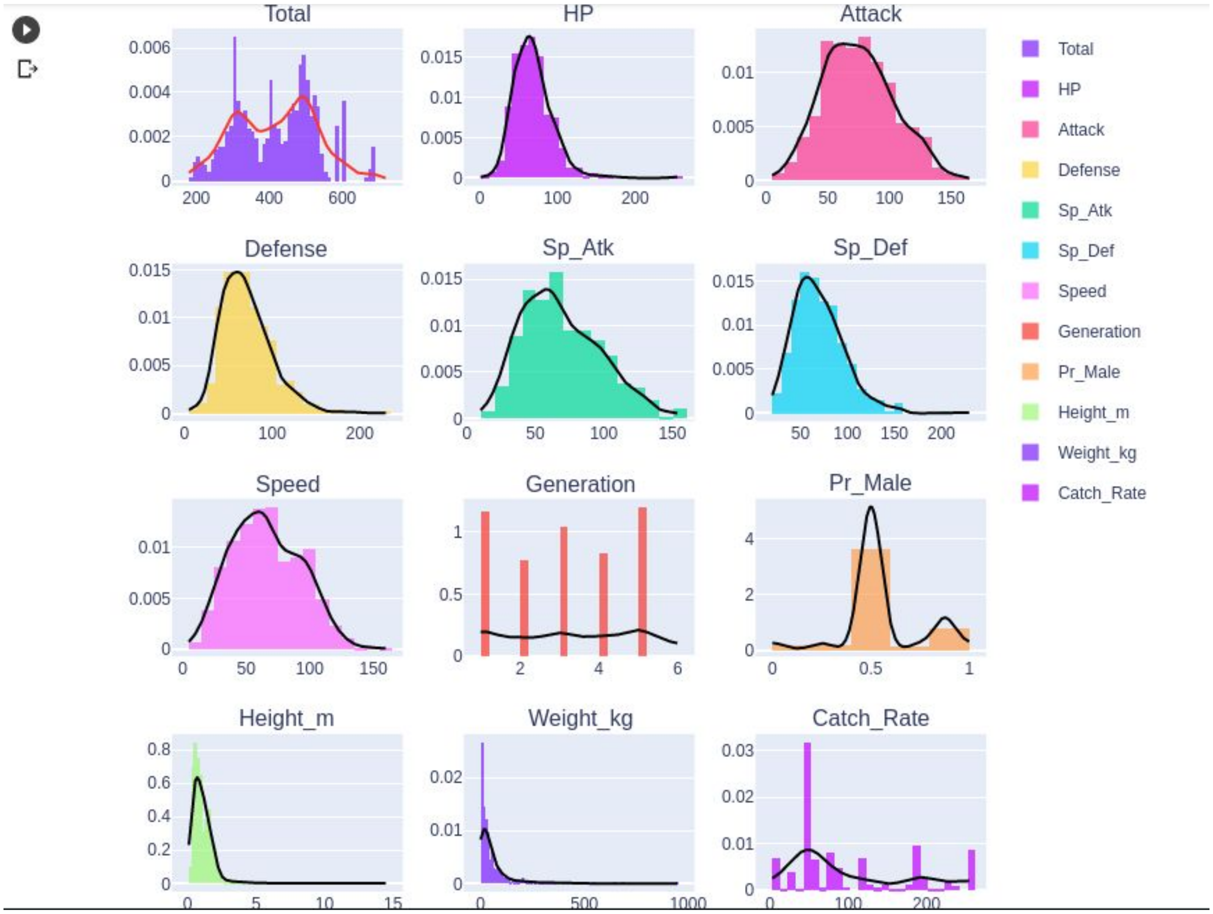
Categorical features analysis



Correlation



Numerical features distribution



Algorithms Used

Following Machine learning algorithms were used to classify the wine as of high or low quality.

1. Decision Tree

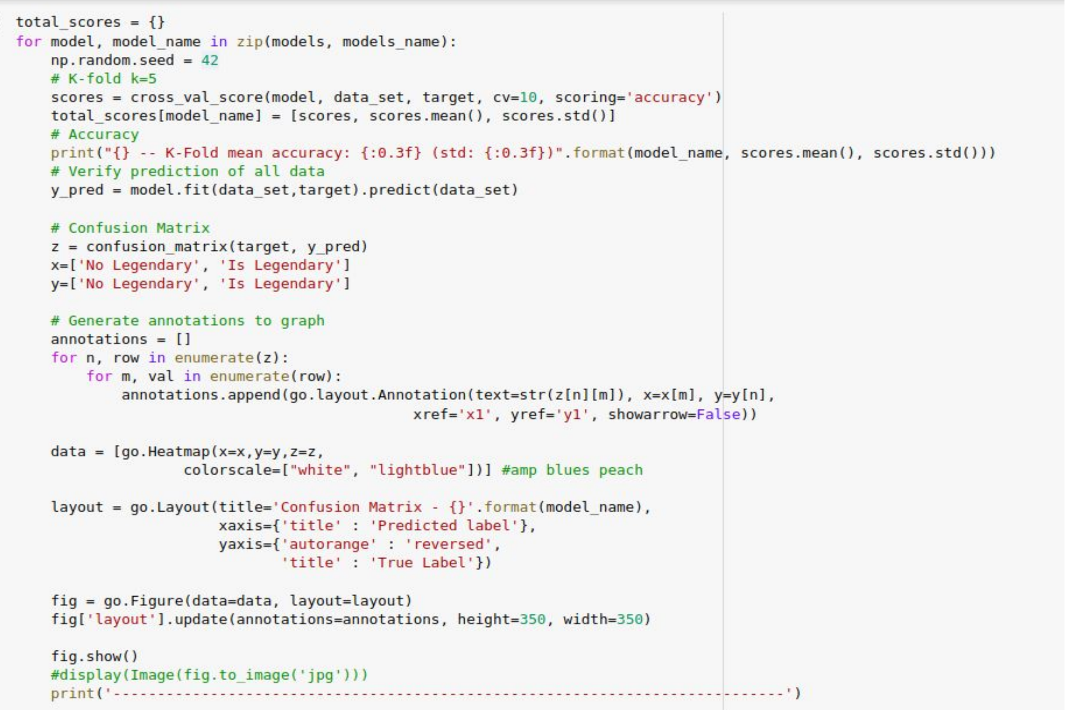
a. **Decision Trees (DTs)** are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. A tree can be seen as a piecewise constant approximation.

2. K-Nearest Neighbours

k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until function evaluation. Since this algorithm relies on distance for classification, if the features represent different physical units or come in vastly different scales then normalizing the training data can improve its accuracy dramatically.

Both for classification and regression, a useful technique can be to assign weights to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of 1/d, where d is the distance to the neighbor.[

Code Screenshot



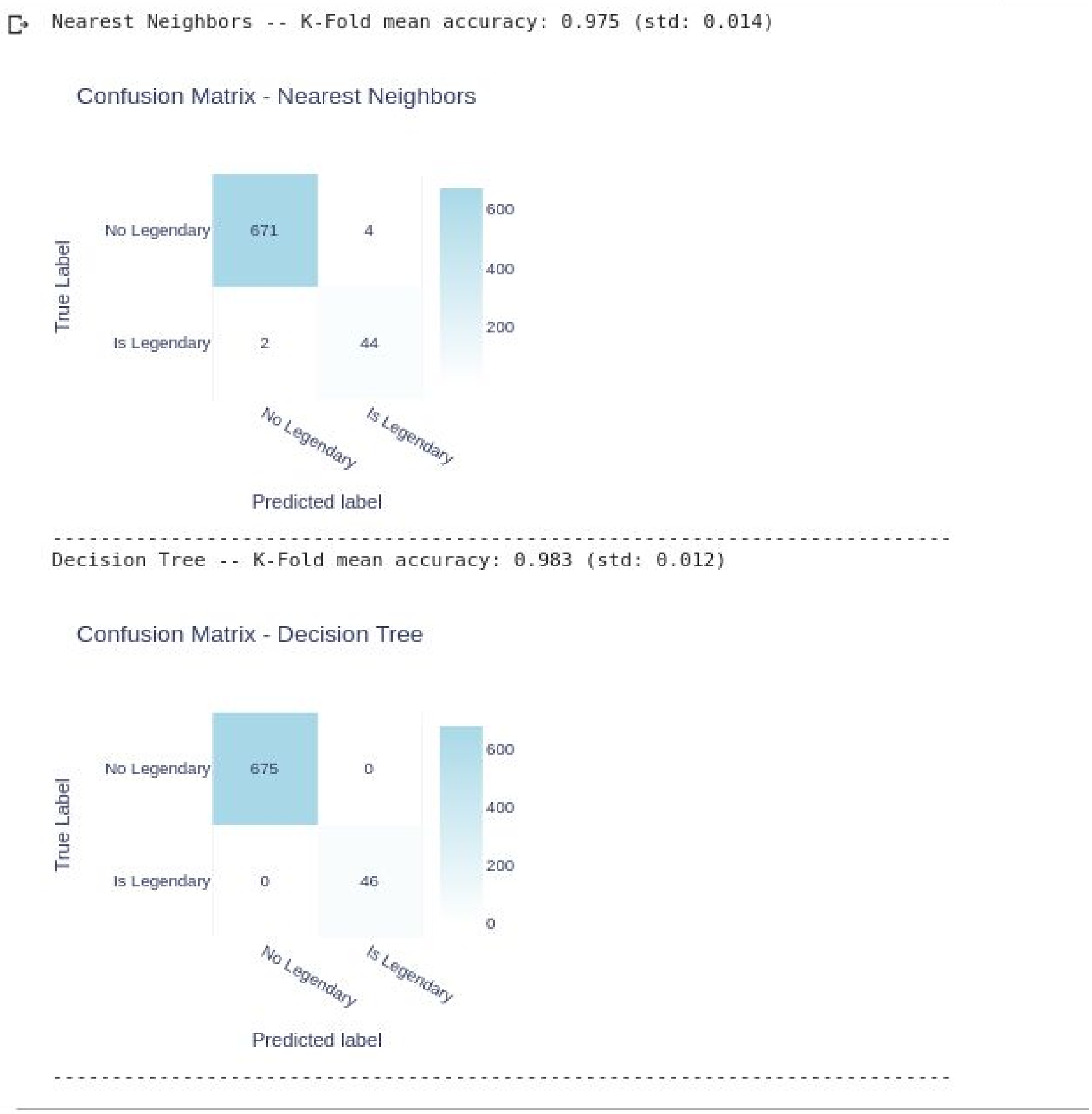
```
total_scores = {}
for model, model_name in zip(models, models_name):
    np.random.seed = 42
    # K-fold k=5
    scores = cross_val_score(model, data_set, target, cv=10, scoring='accuracy')
    total_scores[model_name] = [scores, scores.mean(), scores.std()]
    # Accuracy
    print("{} -- K-Fold mean accuracy: {:0.3f} (std: {:0.3f})".format(model_name, scores.mean(), scores.std()))
    # Verify prediction of all data
    y_pred = model.fit(data_set,target).predict(data_set)

    # Confusion Matrix
    z = confusion_matrix(target, y_pred)
    x=['No Legendary', 'Is Legendary']
    y=['No Legendary', 'Is Legendary']

    # Generate annotations to graph
    annotations = []
    for n, row in enumerate(z):
        for m, val in enumerate(row):
            annotations.append(go.layout.Annotation(text=str(z[n][m]), x=x[m], y=y[n],
                                             xref='x1', yref='y1', showarrow=False))

    data = [go.Heatmap(x=x,y=y,z=z,
                   colorscale=["white", "lightblue"])] #amp blues peach

    layout = go.Layout(title='Confusion Matrix - {}'.format(model_name),
                       xaxis={'title' : 'Predicted label'},
                       yaxis={'autorange' : 'reversed',
                              'title' : 'True Label'})

    fig = go.Figure(data=data, layout=layout)
    fig['layout'].update(annotations=annotations, height=350, width=350)

    fig.show()
    #display(Image(fig.to_image('jpg')))
    print('-----------------------------------------------------------------------------')
```

Results and Discussion

Classification was performed using a number of models on the given using the following four algorithms

1. Decision Tree Algorithm
2. KNN

Out of all the algorithms, Decision Tree Algorithm gave the maximum accuracy of 98.3 %. The confusion matrix is as follows:



```
Nearest Neighbors -- K-Fold mean accuracy: 0.975 (std: 0.014)
```

Confusion Matrix - Nearest Neighbors

	No Legendary	Is Legendary
No Legendary	671	4
Is Legendary	2	44

```
--------------------------------------------------------------------------
Decision Tree -- K-Fold mean accuracy: 0.983 (std: 0.012)
```

Confusion Matrix - Decision Tree

	No Legendary	Is Legendary
No Legendary	675	0
Is Legendary	0	46

```
--------------------------------------------------------------------------
```

Conclusion

This report uses the Pokemon Dataset and classifies the pokemon to be of legendary/ not legendary. Before the modelling, the report makes a brief summary of data preprocessing, analysis and data visualization.

In data preparation, the datasets are downloaded and imported in python. In data exploration and visualization we look for features that may provide good classification results.

Modeling including Decision Tree Algorithm and K-Nearest Neighbours Algorithm. The Decision Tree gave the best accuracy on classifying the pokemon.

References

1. Asier LÃşpez Zorrilla. PokÃľmon for Data Mining https://www.kaggle.com/alopez247/pokemon.
2. David W Scott. Multivariate density estimation: theory, practice, and visualization, 2015.
3. S. S. Shapiro and M. B. Wilk. An analysis of variance test for normality (complete samples). Biometrika.
4. "Introduction to Data Science" - https://rafalab.github.io/dsbook/
5. Alboukadel Kassambara, Fabian Mundt, (2019) - "Extract and Visualize the Results of Multivariate Data Analyses" - https://cran.r-project.org/web/packages/factoextra/factoextra.pdf