POKEMON DATASET

Data Preparation and Modelling

Data Supplementation

Additional data was taken from the internet in order to gain more insight into the effect the individual pokemon types have on combat prowess. The type matchup chart below shows the damage multipliers for all pairs of types for Generation VI onward. In case of dual-typed pokemon, a product of multipliers is calculated.

	Normal	Fight	Flying	Poison	Ground	Rock	Bug	Ghost	Steel	Fire	Water	Grass	Electr	Psychc	Ice	Dragon	Dark	Fairy
Normal	1,0	1,0	1,0	1,0	1,0	0,5	1,0	0,0	0,5	1,0	1,0	1,0	1,0	1,0	1,0	1,0	1,0	1,0
Fight	2,0	1,0	0,5	0,5	1,0	2,0	0,5	0,0	2,0	1,0	1,0	1,0	1,0	0,5	2,0	1,0	2,0	0,5
Flying	1,0	2,0	1,0	1,0	1,0	0,5	2,0	1,0	0,5	1,0	1,0	2,0	0,5	1,0	1,0	1,0	1,0	1,0
Poison	1,0	1,0	1,0	0,5	0,5	0,5	1,0	0,5	0,0	1,0	1,0	2,0	1,0	1,0	1,0	1,0	1,0	2,0
Ground	1,0	1,0	0,0	2,0	1,0	2,0	0,5	1,0	2,0	2,0	1,0	0,5	2,0	1,0	1,0	1,0	1,0	1,0
Rock	1,0	0,5	2,0	1,0	0,5	1,0	2,0	1,0	0,5	2,0	1,0	1,0	1,0	1,0	2,0	1,0	1,0	1,0
Bug	1,0	0,5	0,5	0,5	1,0	1,0	1,0	0,5	0,5	0,5	1,0	2,0	1,0	2,0	1,0	1,0	2,0	0,5
Ghost	0,0	1,0	1,0	1,0	1,0	1,0	1,0	2,0	1,0	1,0	1,0	1,0	1,0	2,0	1,0	1,0	0,5	1,0
Steel	1,0	1,0	1,0	1,0	1,0	2,0	1,0	1,0	0,5	0,5	0,5	1,0	0,5	1,0	2,0	1,0	1,0	2,0
Fire	1,0	1,0	1,0	1,0	1,0	0,5	2,0	1,0	2,0	0,5	0,5	2,0	1,0	1,0	2,0	0,5	1,0	1,0
Water	1,0	1,0	1,0	1,0	2,0	2,0	1,0	1,0	1,0	2,0	0,5	0,5	1,0	1,0	1,0	0,5	1,0	1,0
Grass	1,0	1,0	0,5	0,5	2,0	2,0	0,5	1,0	0,5	0,5	2,0	0,5	1,0	1,0	1,0	0,5	1,0	1,0
Electr	1,0	1,0	2,0	1,0	0,0	1,0	1,0	1,0	1,0	1,0	2,0	0,5	0,5	1,0	1,0	0,5	1,0	1,0
Psychc	1,0	2,0	1,0	2,0	1,0	1,0	1,0	1,0	0,5	1,0	1,0	1,0	1,0	0,5	1,0	1,0	0,0	1,0
Ice	1,0	1,0	2,0	1,0	2,0	1,0	1,0	1,0	0,5	0,5	0,5	2,0	1,0	1,0	0,5	2,0	1,0	1,0
Dragon	1,0	1,0	1,0	1,0	1,0	1,0	1,0	1,0	0,5	1,0	1,0	1,0	1,0	1,0	1,0	2,0	1,0	0,0
Dark	1,0	0,5	1,0	1,0	1,0	1,0	1,0	2,0	1,0	1,0	1,0	1,0	1,0	2,0	1,0	1,0	0,5	0,5
Fairy	1,0	2,0	1,0	0,5	1,0	1,0	1,0	1,0	0,5	0,5	1,0	1,0	1,0	1,0	1,0	2,0	2,0	1,0

Data taken from Bulbapedia, the community-driven Pokémon encyclopedia

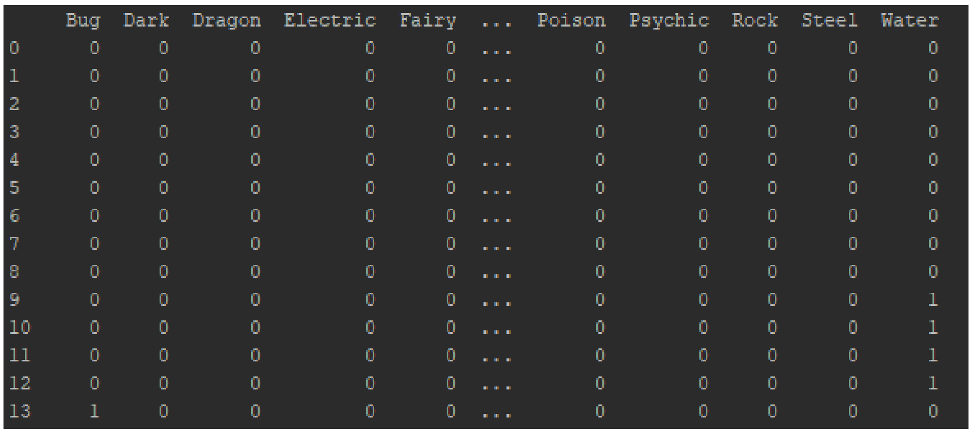
The chart was saved as a CSV file and used to model the decision trees which were used to predict the results of future matchups.

Created by: Grzegorz Meller, Mateusz Rock, Tomasz Wierciński, Robert Zwierzycki

POKEMON DATASET

Data Conversion

In order to fit kNN classifier with data we had to convert all categorical attributes like Type 1 or Type 2 into numbers. But we didn't wanted to simply assign random number to each category, because model could misunderstand the data to be in some kind of order, 0 < 1 < 2. The solution was One Hot Encoder which encodes categorical variables into one-hot numeric array. And the results of one code encoding for Type 1 column looks like this:



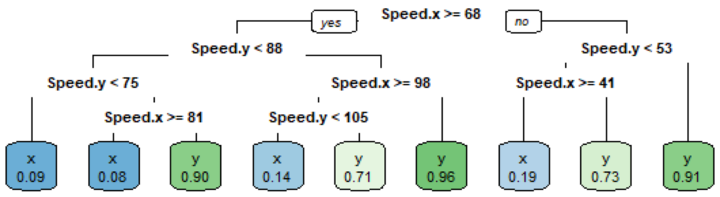
```
    Bug  Dark  Dragon  Electric  Fairy  ...  Poison  Psychic  Rock  Steel  Water
0     0     0       0         0      0  ...       0        0     0      0      0
1     0     0       0         0      0  ...       0        0     0      0      0
2     0     0       0         0      0  ...       0        0     0      0      0
3     0     0       0         0      0  ...       0        0     0      0      0
4     0     0       0         0      0  ...       0        0     0      0      0
5     0     0       0         0      0  ...       0        0     0      0      0
6     0     0       0         0      0  ...       0        0     0      0      0
7     0     0       0         0      0  ...       0        0     0      0      0
8     0     0       0         0      0  ...       0        0     0      0      0
9     0     0       0         0      0  ...       0        0     0      0      1
10    0     0       0         0      0  ...       0        0     0      0      1
11    0     0       0         0      0  ...       0        0     0      0      1
12    0     0       0         0      0  ...       0        0     0      0      1
13    1     0       0         0      0  ...       0        0     0      0      0
```

Created Models

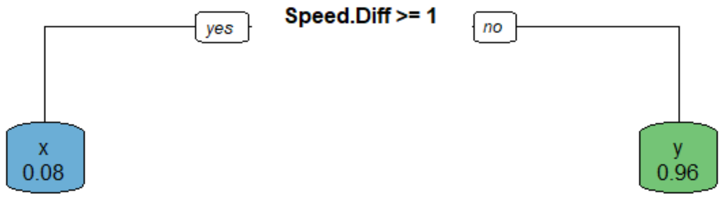
1. Will you be able to predict the outcome of future combats?

The prediction of results of combats was done using a decision tree. The tree was trained on 75% of the available samples and tested on the remaining 25%. Two models were created. The first one - model A - trained on all the numerical and nominal attributes related to the two combatants. The second one - model B - trained on the multipliers calculated using the types of the two pokemon and the differences between their respective numerical attributes.

Model A - 88.464% accuracy

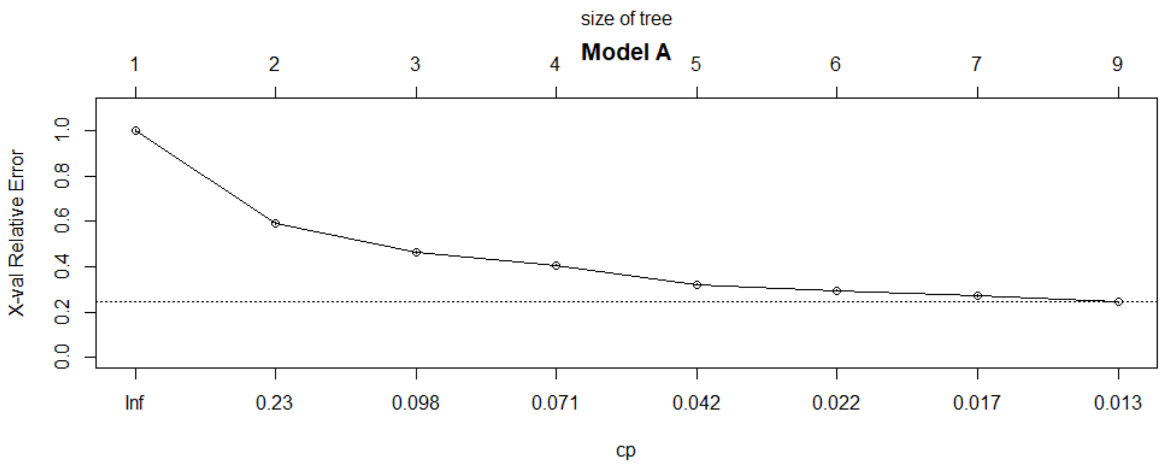


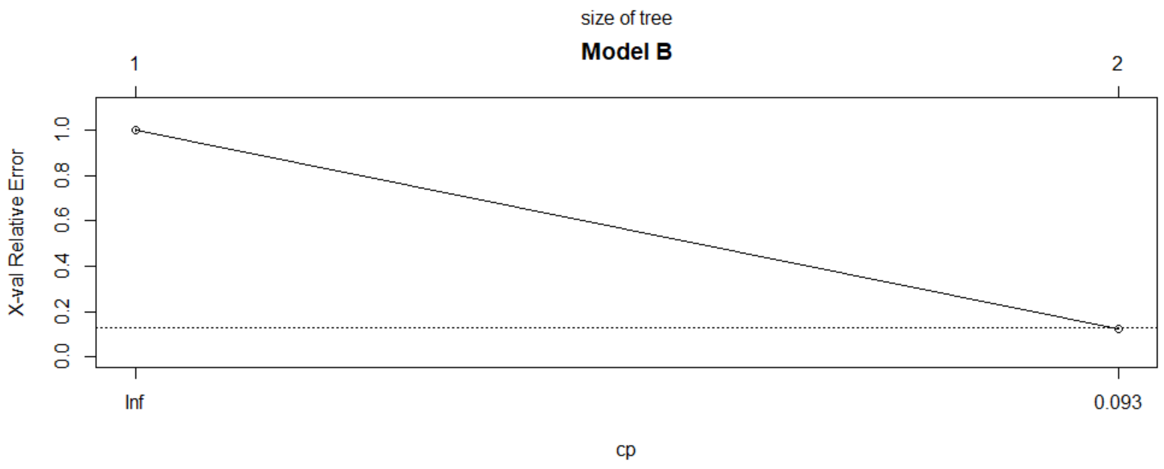
Model B - 94.096% accuracy



POKEMON DATASET

Both of the trees were modelled using only the speed attribute, as it was the only attribute which could result in a split decreasing the overall lack of fit by a factor equal to the complexity parameter = 0.01. The results of the cross-validation of the two models is shown below.



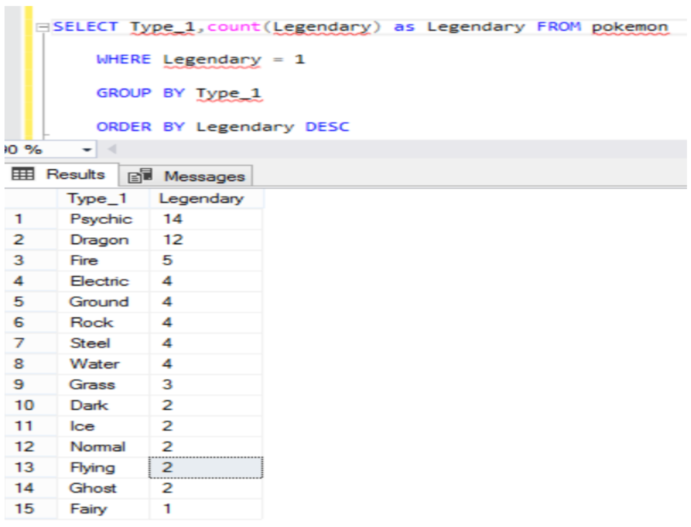


2. Which type is the most likely to be legendary?

To obtain the answer for this question, we have made a simple sql query to find this type.We were taking only the Type_1 because Type_2 contains null values.

Most likely : Psychic

Least likely: Fairy



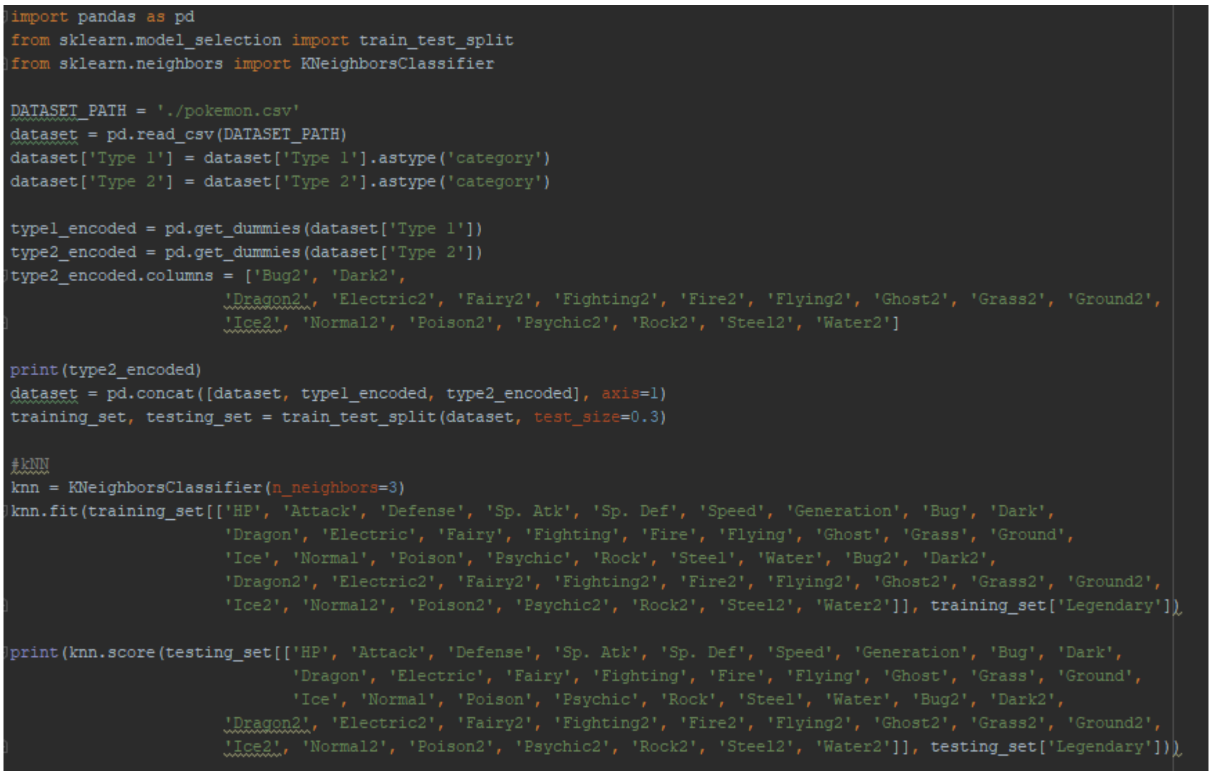
```sql
SELECT Type_1,count(Legendary) as Legendary FROM pokemon
    WHERE Legendary = 1
    GROUP BY Type_1
    ORDER BY Legendary DESC
```

	Type_1	Legendary
1	Psychic	14
2	Dragon	12
3	Fire	5
4	Electric	4
5	Ground	4
6	Rock	4
7	Steel	4
8	Water	4
9	Grass	3
10	Dark	2
11	Ice	2
12	Normal	2
13	Flying	2
14	Ghost	2
15	Fairy	1

POKEMON DATASET

To this challenge we also built classifier using kNN algorithm to predict whether pokemon is going to be legendary or not. In order to do this, we had to convert all categorical variables like Type 1 or Type 2 into nominal values using one hot encoding method. After data preprocessing, we divided our dataset into training set (70% of the dataset) and testing set (30% of the dataset). We built our model by fitting appropriate columns from training data into the corresponding legendary status. The most optimal number of neighbours happened to be 3 and gave 95.4% of accuracy.

Python code:



```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier

DATASET_PATH = './pokemon.csv'
dataset = pd.read_csv(DATASET_PATH)
dataset['Type 1'] = dataset['Type 1'].astype('category')
dataset['Type 2'] = dataset['Type 2'].astype('category')

type1_encoded = pd.get_dummies(dataset['Type 1'])
type2_encoded = pd.get_dummies(dataset['Type 2'])
type2_encoded.columns = ['Bug2', 'Dark2',
                      'Dragon2', 'Electric2', 'Fairy2', 'Fighting2', 'Fire2', 'Flying2', 'Ghost2', 'Grass2', 'Ground2',
                      'Ice2', 'Normal2', 'Poison2', 'Psychic2', 'Rock2', 'Steel2', 'Water2']

print(type2_encoded)
dataset = pd.concat([dataset, type1_encoded, type2_encoded], axis=1)
training_set, testing_set = train_test_split(dataset, test_size=0.3)

#kNN
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(training_set[['HP', 'Attack', 'Defense', 'Sp. Atk', 'Sp. Def', 'Speed', 'Generation', 'Bug', 'Dark',
                      'Dragon', 'Electric', 'Fairy', 'Fighting', 'Fire', 'Flying', 'Ghost', 'Grass', 'Ground',
                      'Ice', 'Normal', 'Poison', 'Psychic', 'Rock', 'Steel', 'Water', 'Bug2', 'Dark2',
                      'Dragon2', 'Electric2', 'Fairy2', 'Fighting2', 'Fire2', 'Flying2', 'Ghost2', 'Grass2', 'Ground2',
                      'Ice2', 'Normal2', 'Poison2', 'Psychic2', 'Rock2', 'Steel2', 'Water2']], training_set['Legendary'])

print(knn.score(testing_set[['HP', 'Attack', 'Defense', 'Sp. Atk', 'Sp. Def', 'Speed', 'Generation', 'Bug', 'Dark',
                             'Dragon', 'Electric', 'Fairy', 'Fighting', 'Fire', 'Flying', 'Ghost', 'Grass', 'Ground',
                             'Ice', 'Normal', 'Poison', 'Psychic', 'Rock', 'Steel', 'Water', 'Bug2', 'Dark2',
                      'Dragon2', 'Electric2', 'Fairy2', 'Fighting2', 'Fire2', 'Flying2', 'Ghost2', 'Grass2', 'Ground2',
                      'Ice2', 'Normal2', 'Poison2', 'Psychic2', 'Rock2', 'Steel2', 'Water2']], testing_set['Legendary']))
```

3. Can you build a Pokemon dream team (6 pokemons)?

We have chosen the best six pokemon to the dream team based on number of won combats. They are:

Mewtwo
Aerodactyl
Infernape
Jirachi
Slaking
Deoxys Speed Forme

4. Find strongest and weakest pokemon overall

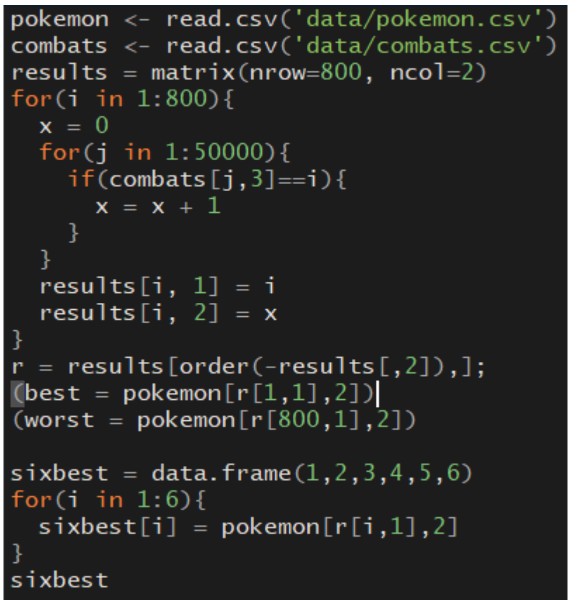
We have chosen the strongest and the weakest pokemon based on number of won combats. To find the strongest and weakest pokemon overall we have chosen a subset of 50 000 combats that have occured.

Our results:

The strongest pokemon is *Mewtwo*.

The weakest pokemon is *Pumpkaboo Small Size*.

R code:



```
pokemon <- read.csv('data/pokemon.csv')
combats <- read.csv('data/combats.csv')
results = matrix(nrow=800, ncol=2)
for(i in 1:800){
  x = 0
  for(j in 1:50000){
    if(combats[j,3]==i){
      x = x + 1
    }
  }
  results[i, 1] = i
  results[i, 2] = x
}
r = results[order(-results[,2]),];
(best = pokemon[r[1,1],2])
(worst = pokemon[r[800,1],2])

sixbest = data.frame(1,2,3,4,5,6)
for(i in 1:6){
  sixbest[i] = pokemon[r[i,1],2]
}
sixbest
```