Лабораторная работа №2

Визуальный анализ данных

Подключение библиотек

```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
%matplotlib inline
```

Загрузка данных

```
data_path = "/content/Pokemon.csv"
data = pd.read_csv(data_path)
data.head(10)
# data.columns
```


```
{"summary":"{\n  \"name\": \"# data\",\n  \"rows\": 10,\n  \"fields\":
[\n    {\n      \"column\": \"#\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 2,\n        \"min\": 1,\n
\"max\": 7,\n        \"num_unique_values\": 7,\n        \"samples\":
[\n          1,\n          2,\n          6\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Name\",\n      \"properties\": {\n
\"dtype\": \"string\",\n        \"num_unique_values\": 10,\n
\"samples\": [\n          \"CharizardMega Charizard Y\",\n
\"Ivysaur\",\n          \"Charmeleon\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Type 1\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
3,\n        \"samples\": [\n          \"Grass\",\n          \"Fire\",\
n          \"Water\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Type 2\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 3,\n        \"samples\":
[\n          \"Poison\",\n          \"Flying\",\n          \"Dragon\"\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Total\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 136,\n        \"min\": 309,\n        \"max\": 634,\n
\"num_unique_values\": 8,\n        \"samples\": [\n          405,\n
534,\n          318\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
```

\"HP\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 16,\n   \"min\": 39,\n   \"max\": 80,\n
\"num_unique_values\": 7,\n   \"samples\": [\n   45,\n
60,\n   78\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Attack\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 27,\n   \"min\": 48,\n   \"max\": 130,\n
\"num_unique_values\": 10,\n   \"samples\": [\n   104,\n
62,\n   64\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Defense\",\n   \"properties\": {\n   \"dtype\": \"number\",\
n   \"std\": 25,\n   \"min\": 43,\n   \"max\": 123,\n
\"num_unique_values\": 9,\n   \"samples\": [\n   111,\n
63,\n   58\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Sp.
Atk\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 34,\n   \"min\": 50,\n   \"max\": 159,\n
\"num_unique_values\": 9,\n   \"samples\": [\n   159,\n
80,\n   109\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Sp.
Def\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 23,\n   \"min\": 50,\n   \"max\": 120,\n
\"num_unique_values\": 8,\n   \"samples\": [\n   80,\n
85,\n   65\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Speed\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 21,\n   \"min\": 43,\n   \"max\": 100,\n
\"num_unique_values\": 6,\n   \"samples\": [\n   45,\n
60,\n   43\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Generation\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 0,\n   \"min\": 1,\n
\"max\": 1,\n   \"num_unique_values\": 1,\n   \"samples\":
[\n   1\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Legendary\",\n   \"properties\": {\n   \"dtype\":
\"boolean\",\n   \"num_unique_values\": 1,\n   \"samples\":
[\n   false\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe"}

Одиночные признаки

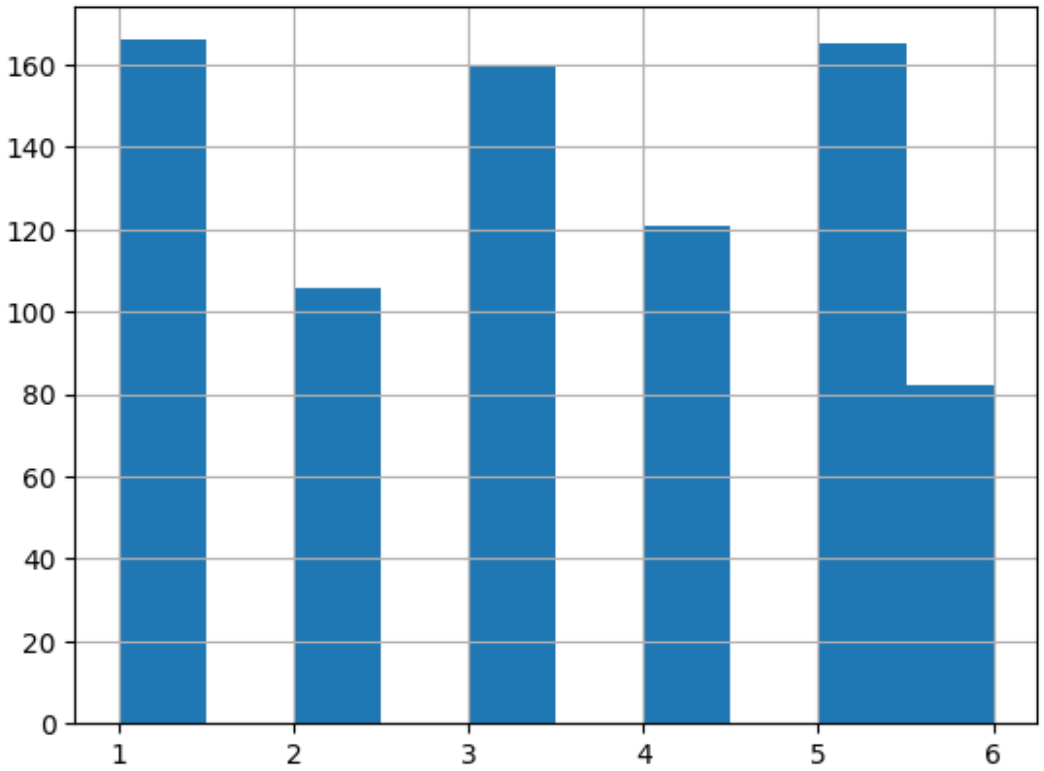
Количественные признаки

```
data.columns
```

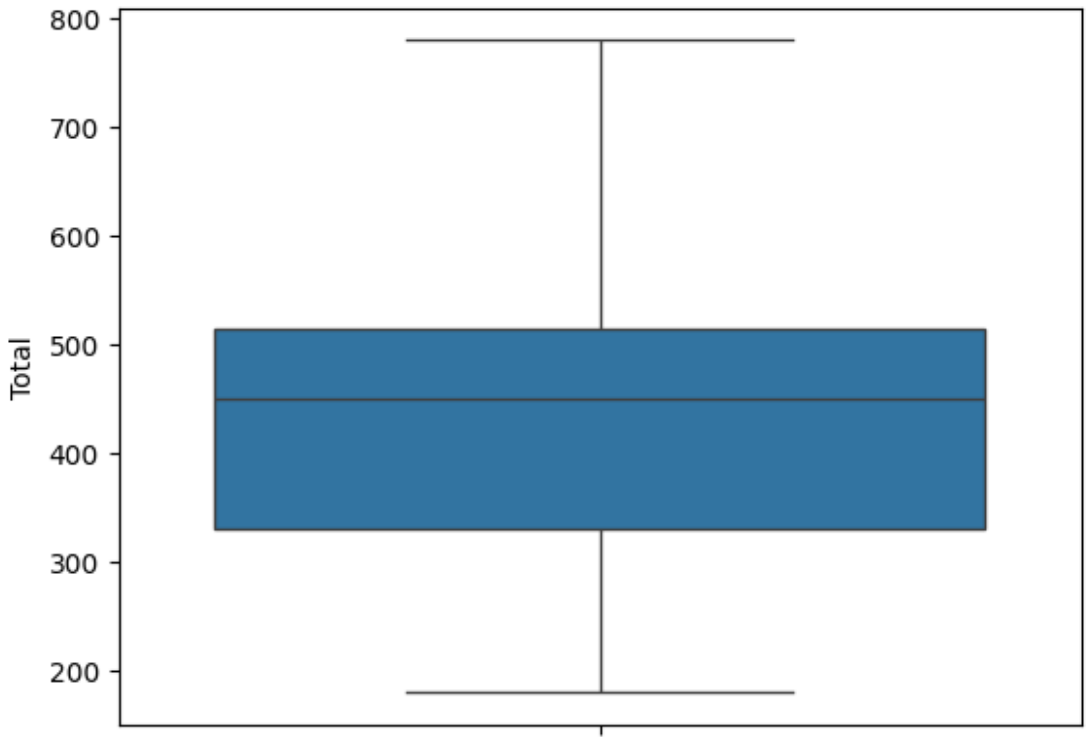
```
Index(['#', 'Name', 'Type 1', 'Type 2', 'Total', 'HP', 'Attack',
'Defense',
```

```
       'Sp. Atk', 'Sp. Def', 'Speed', 'Generation', 'Legendary'],
      dtype='object')
```

```
# Применение pandas для визуализации данных
# Pandas работает как настройка над matplotlib
data['Generation'].hist();
```

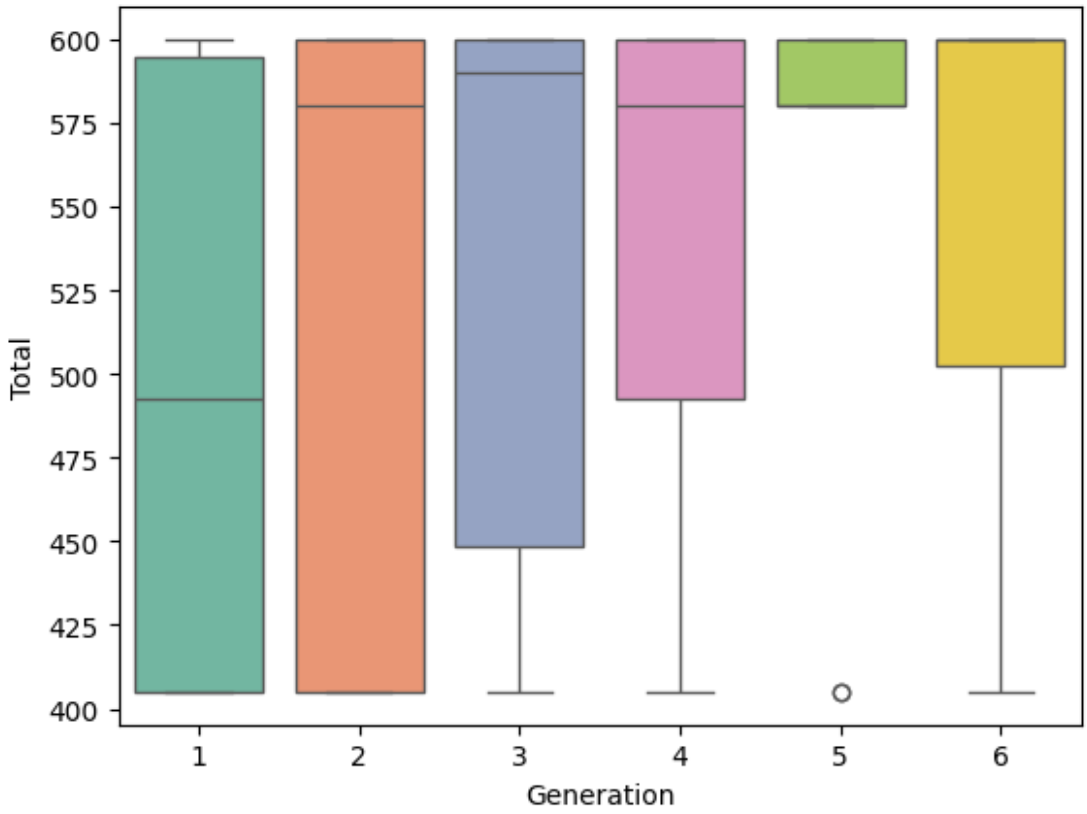



```
# использование Seaborn
# Построение диаграммы типа "ящик с усами"
# по диаграмме можно определить медиану, квартили,
# интерквартильный размах, выбросы
sns.boxplot(data['Total']);
```

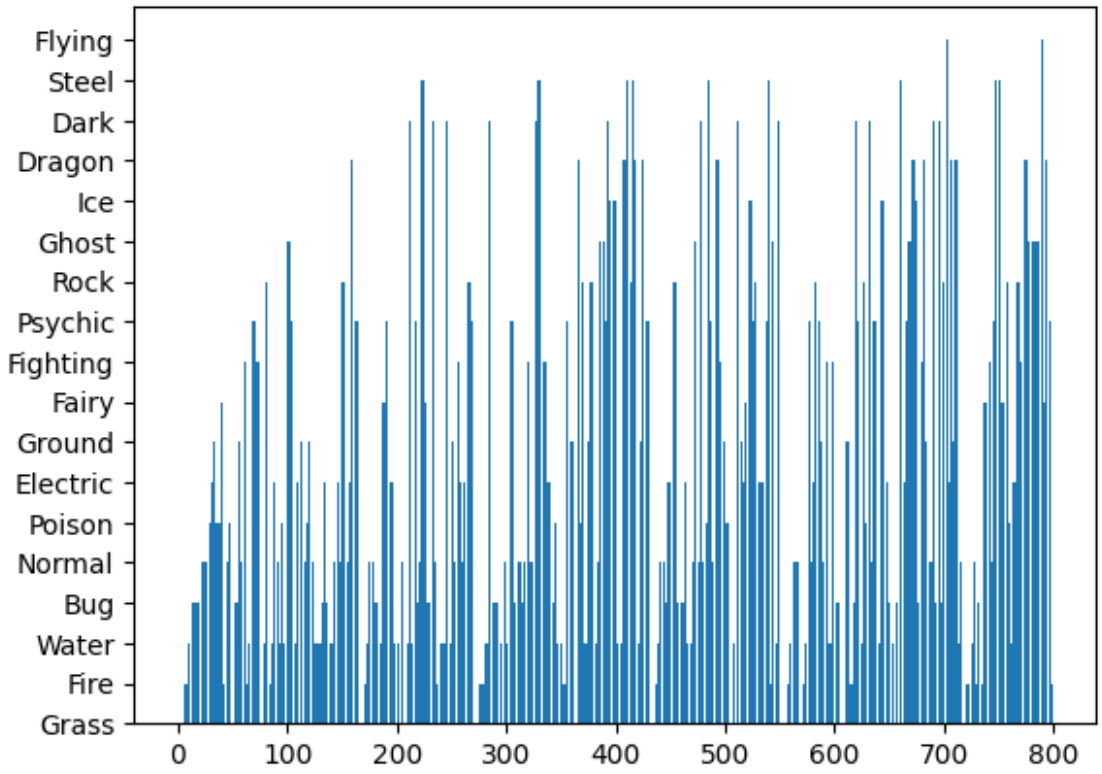



```
top_data = data[['Total','Generation']]
top_data = top_data.groupby('Total').sum()
top_data = top_data.sort_values('Generation',ascending=False)
top_data = top_data[:3].index.values
sns.boxplot(y='Total',
            x='Generation',
            data=data[data.Total.isin(top_data)], palette='Set2');
```

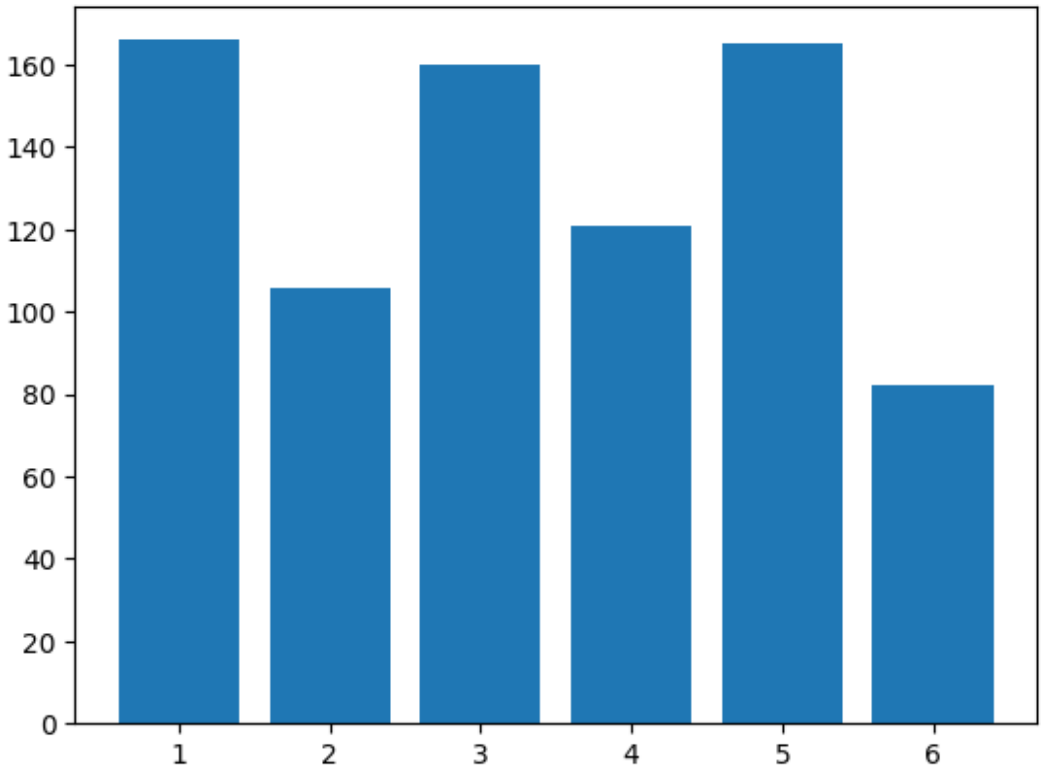
```
<ipython-input-6-4e2ed6dd854c>:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.boxplot(y='Total',
```




```
plt.bar(data.index, data['Type 1'])
plt.show()
```



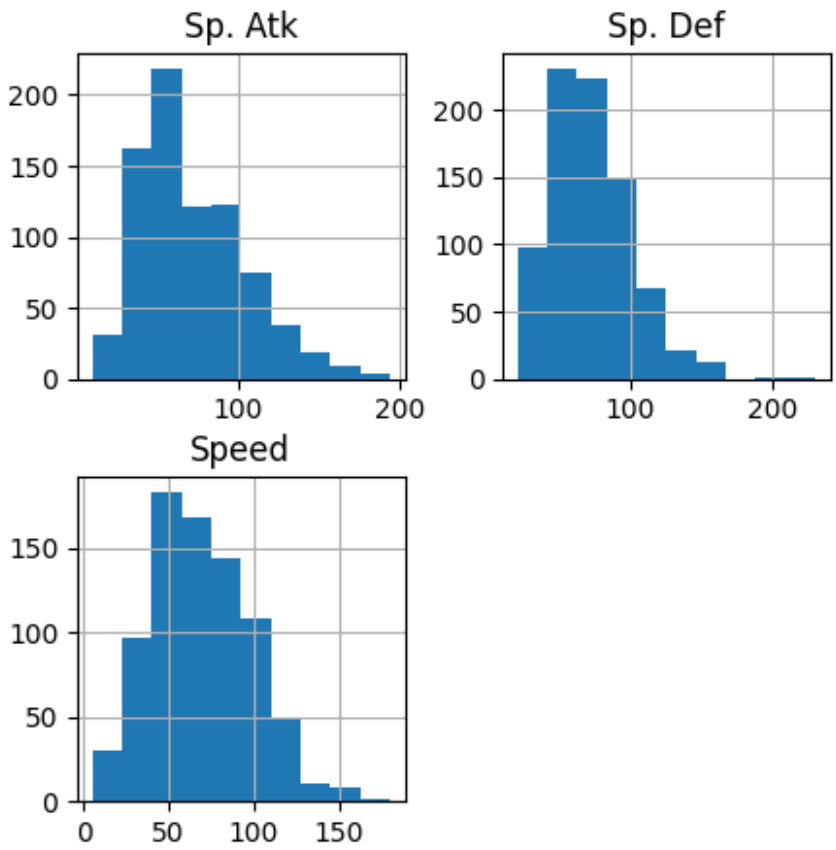

```
hist = data['Generation'].value_counts()
plt.bar(hist.index, hist);
```



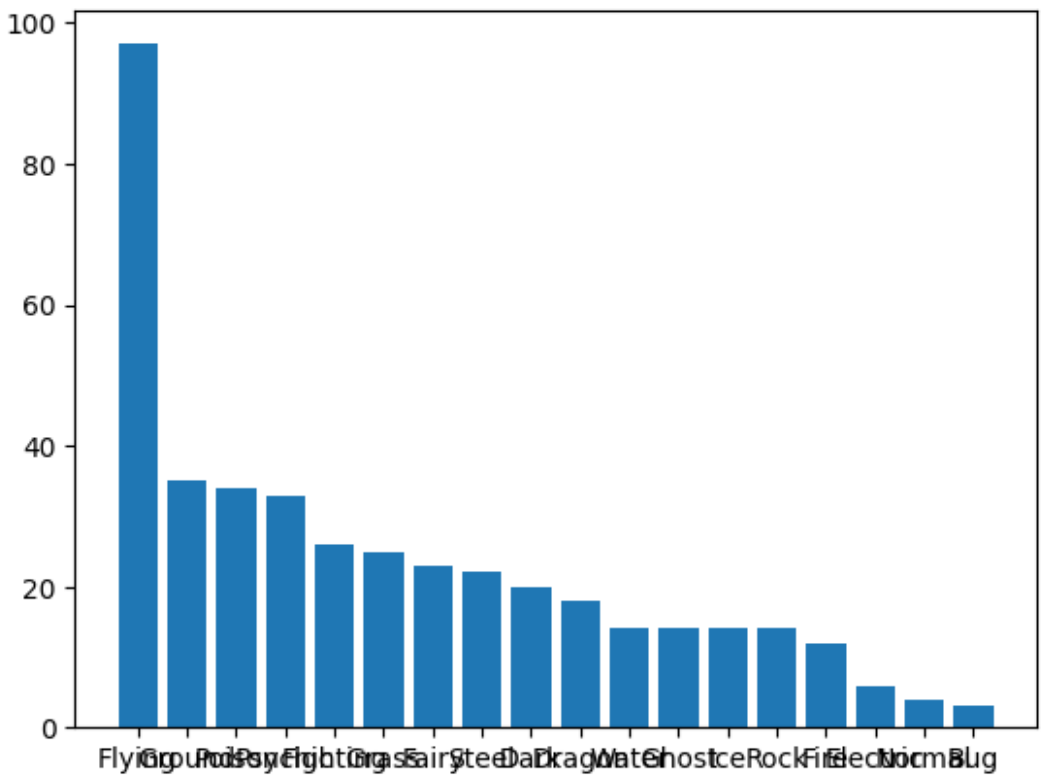

```
# jn,jh ghbpyfrjd
feats=[f for f in data.columns if 'Sp' in f]
feats
```

```
['Sp. Atk', 'Sp. Def', 'Speed']
```

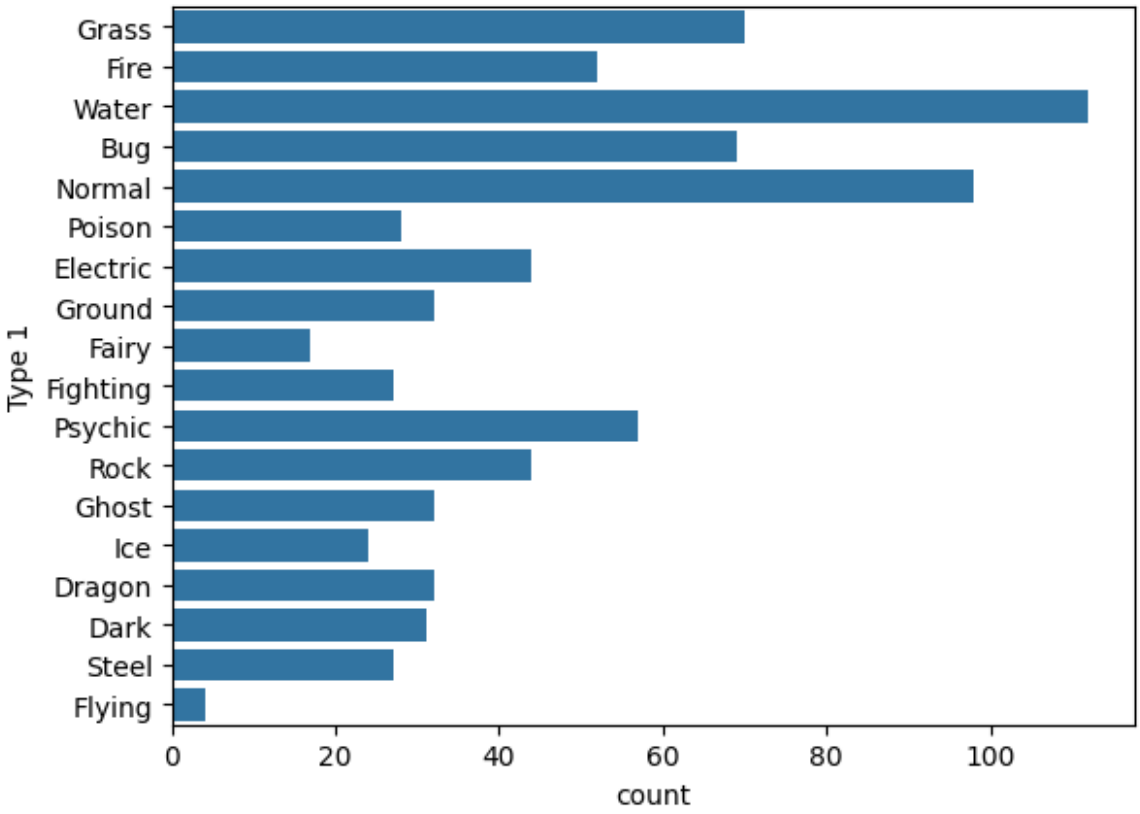
```
# построение гистограммы для нескольких признаков
data[feats].hist(figsize=(5,5));
```



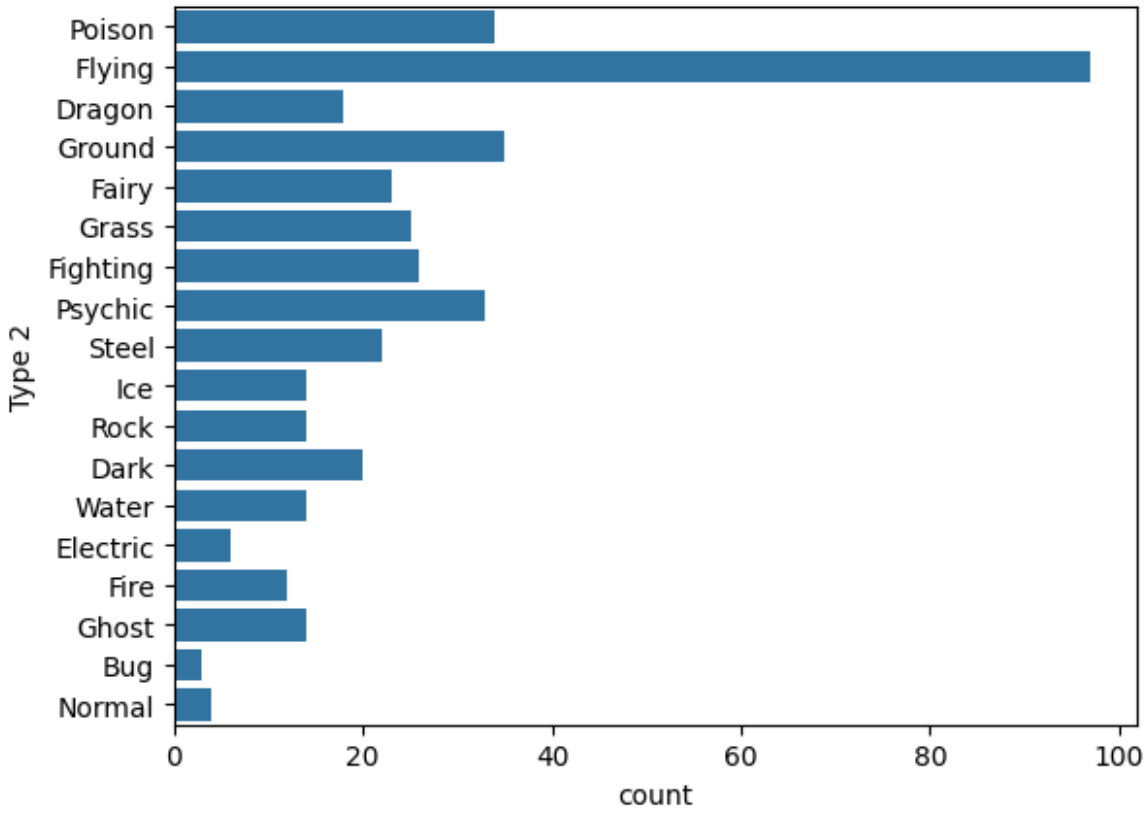

```
# определение первых n "популярных" типов покемонов из Type 2
# data['Type 2'].value_counts().head(10)
hist = data['Type 2'].value_counts()
plt.bar(hist.index, hist);
```



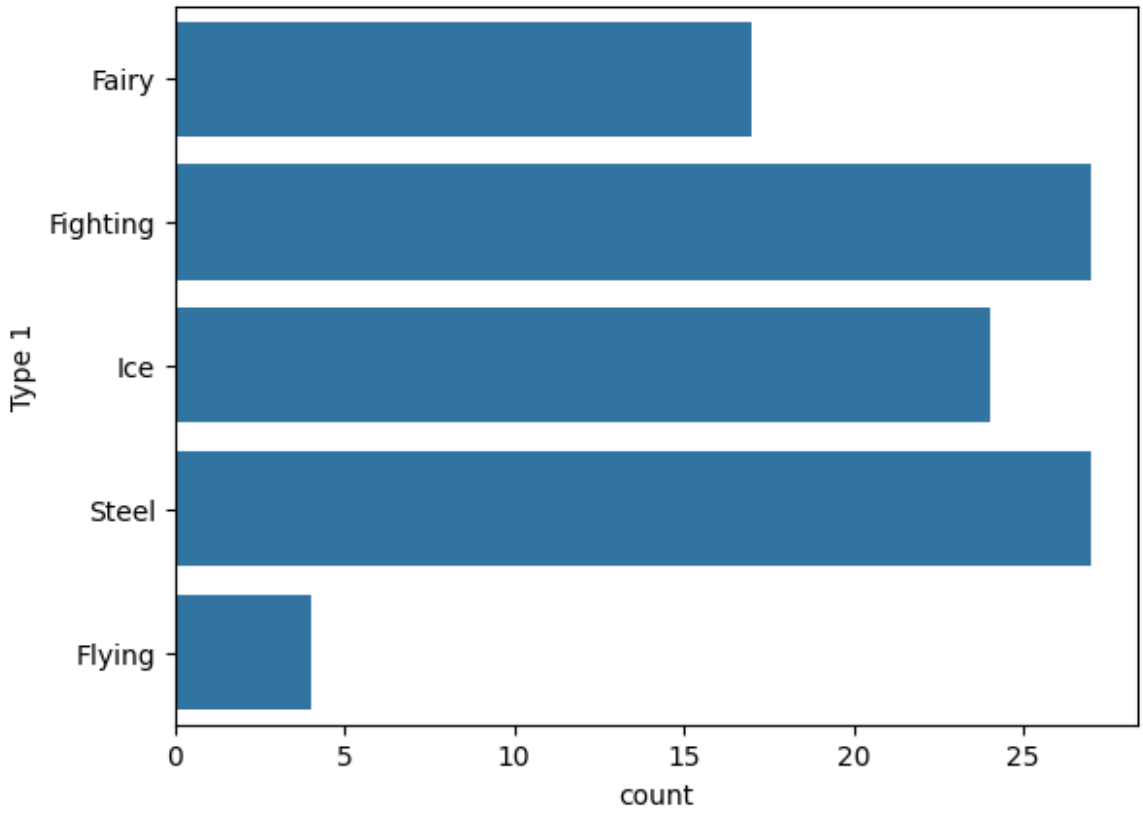

```
# фактически бинарный признак
data['Legendary'].value_counts()

Legendary
False    735
True      65
Name: count, dtype: int64

sns.countplot(data['Type 1']);
```




```
# гистограмма для всех покемонов Type 2
sns.countplot(data['Type 2']);
```




```
# гистограмма "популярных" покемонов из Type 1
sns.countplot(data[data['Type 1'].isin(data['Type
1'].value_counts().tail(5).index)]['Type 1']);
```



Взаимосвязанные признаки

Количественный - количественный

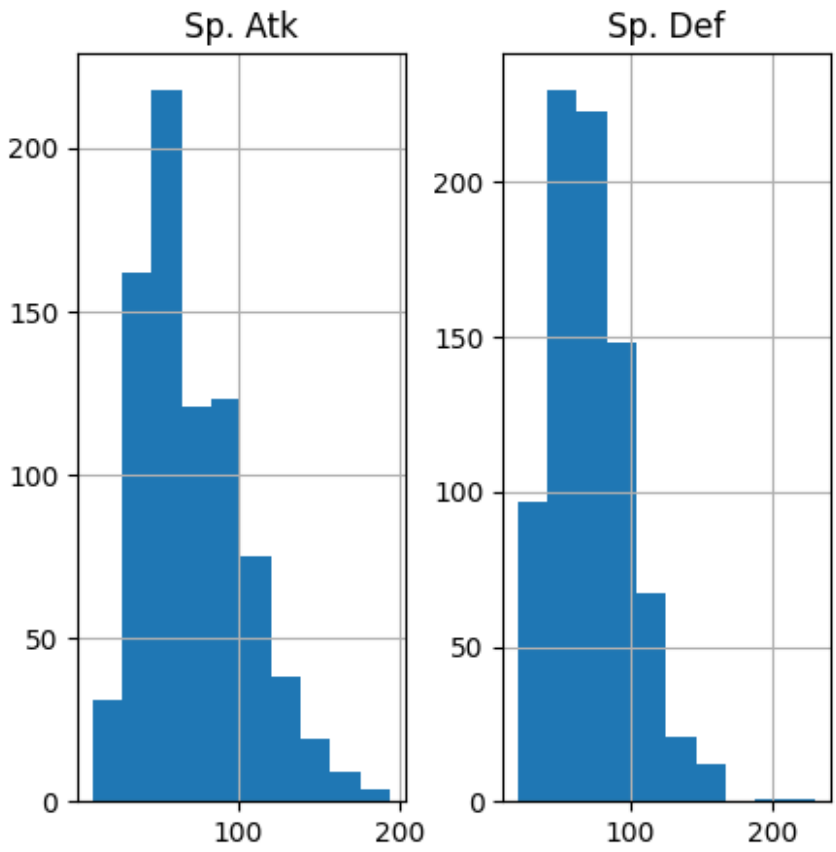
```
# список колонок
data.columns
```

```
Index(['#', 'Name', 'Type 1', 'Type 2', 'Total', 'HP', 'Attack',
'Defense',
       'Sp. Atk', 'Sp. Def', 'Speed', 'Generation', 'Legendary'],
      dtype='object')
```

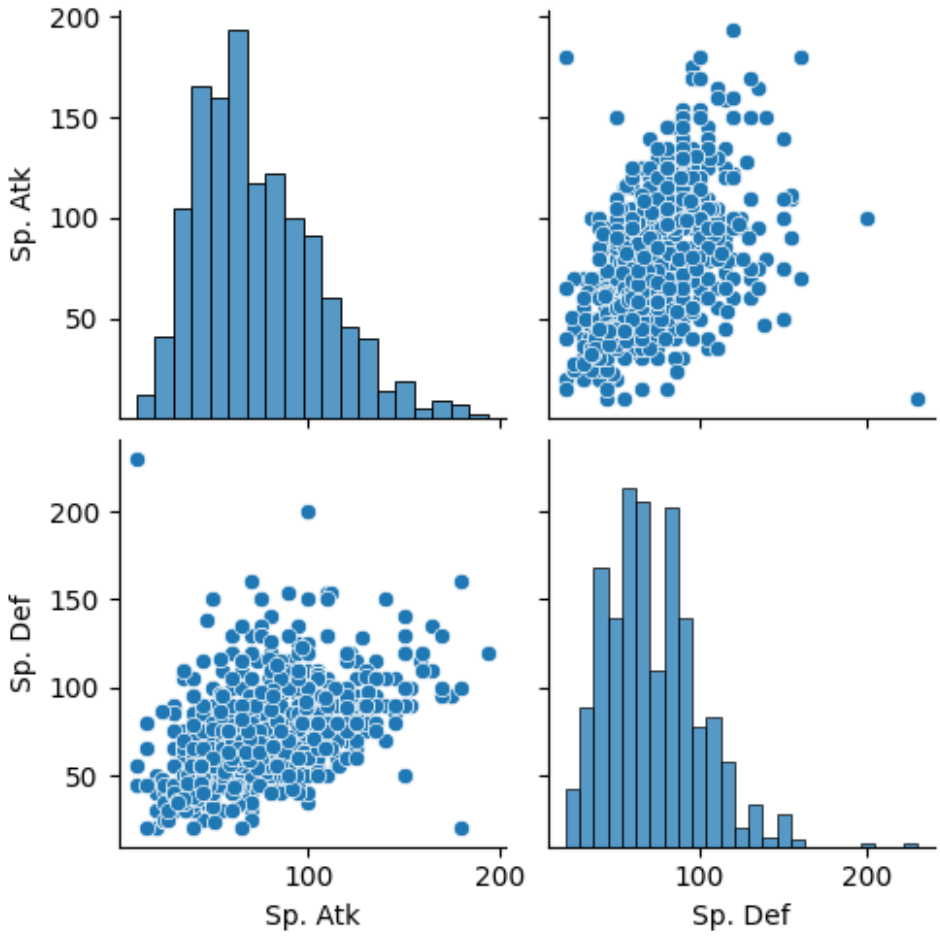
```
# Отбор числовых признаков, содержащих слово 'Sp.'
feats = [f for f in data.columns if 'Sp.' in f]
len(feats)
# feats=['Total day calls', 'Total day Sp.']
```

```
2
```

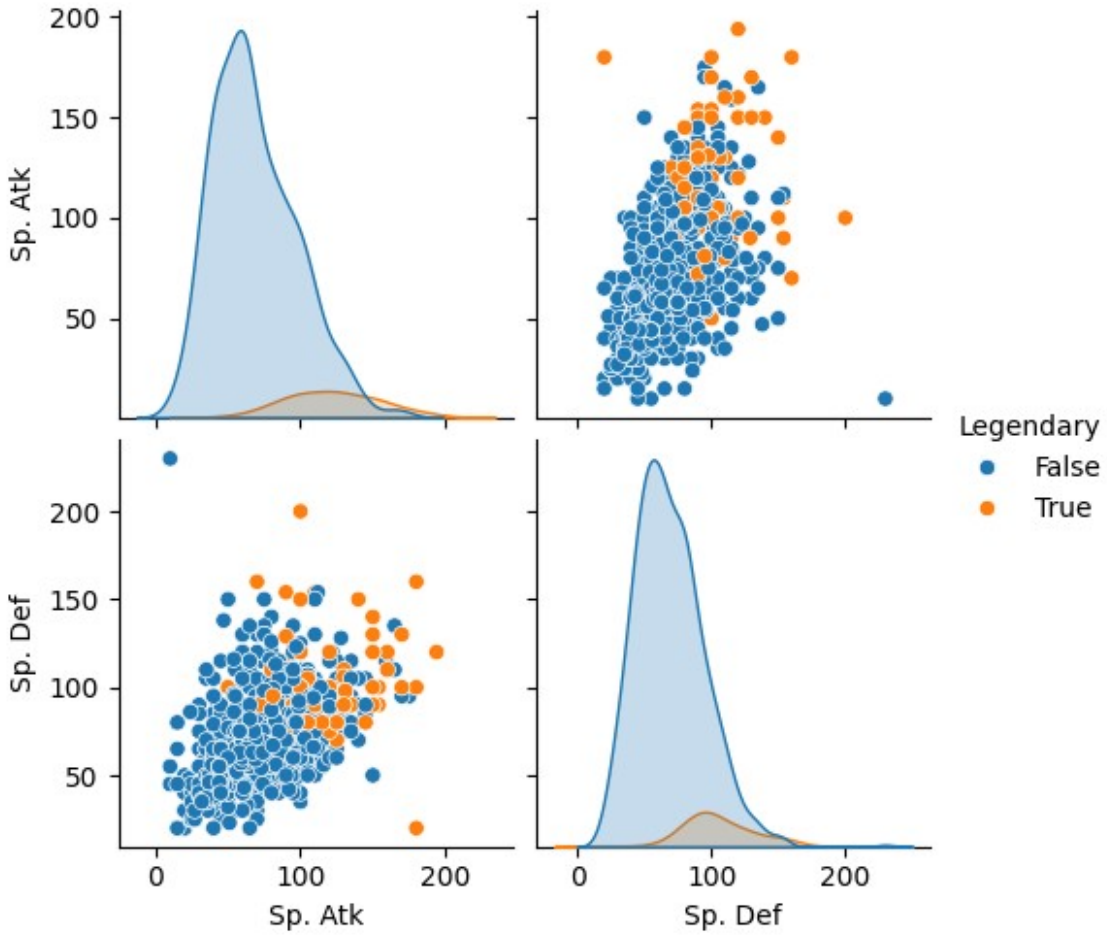
```
# строим отдельные гистограммы
# для нескольких признаков
data[feats].hist(figsize=(5,5));
```




```
# Попарное распределение признаков
# Применение Seaborn
sns.pairplot(data[feats]);
```

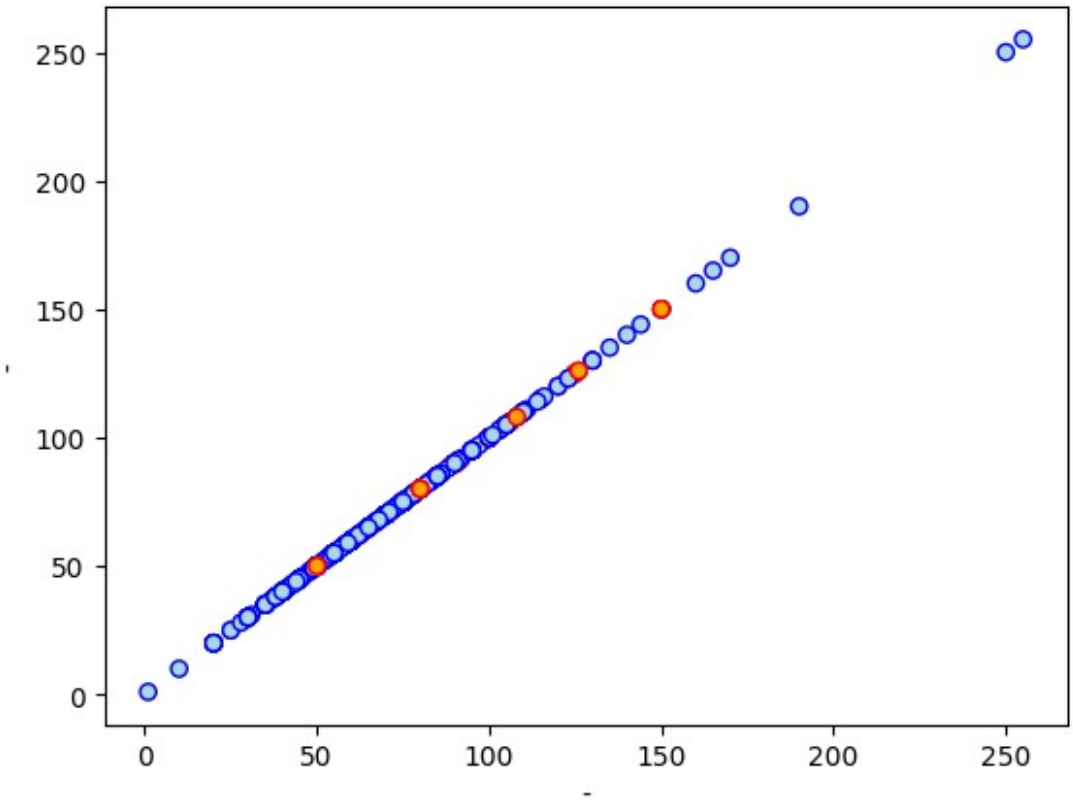



```
# Можно строить более сложные попарные распределения признаков
(добавил колонку "Legendary")
sns.pairplot(data[feats + ['Legendary']], hue='Legendary');
```

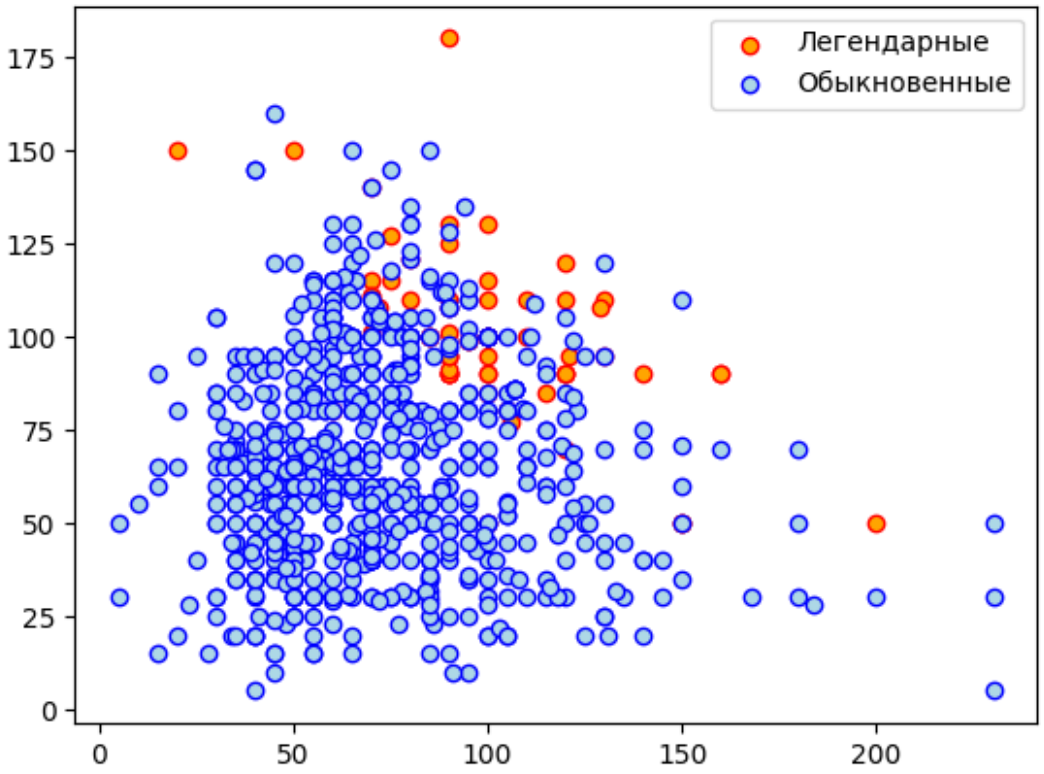



```
# Использование matplotlib, подписей данных, заголовков Использование
простейших пользовательских цветов
plt.scatter(data['HP'],
            data['Attack'],
            color='lightblue', edgecolors='blue')
plt.xlabel('Кол-во очков жизни')
plt.ylabel('Кол-во очков атаки')
plt.title('Распределение по 2 признакам');
```


```
# Раскрашивание данных
# Цвет в зависимости от типа кол-ва очков жизни
c = data['Legendary'].map({False: 'lightblue', True: 'orange'})
edge_c = data['Legendary'].map({False: 'blue', True: 'red'})
# Настройка графика
plt.scatter(data['HP'], data['HP'],
            color=c, edgecolors=edge_c
           )
plt.xlabel('-')
plt.ylabel('-');
```



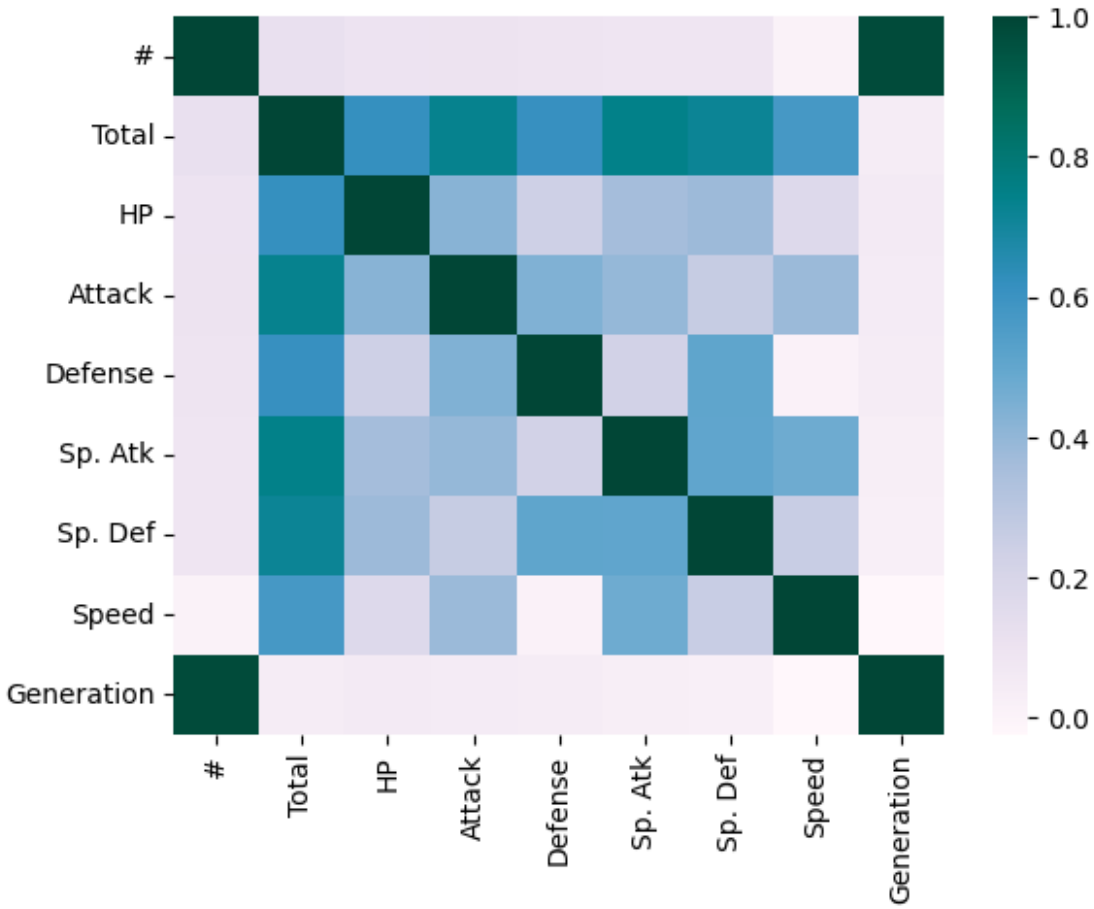

```
# Раскраска легендарных и обыкновенных покемонов,

# Легендарные покемоны
data_churn = data[data['Legendary']]
# Обыкновенные покемоны
data_loyal = data[~data['Legendary']]

plt.scatter(data_churn['Defense'],
            data_churn['Speed'],
            color='orange',
            edgecolors='red',
            label='Легендарные'
           )
plt.scatter(data_loyal['Defense'],
            data_loyal['Speed'],
            color='lightblue',
            edgecolors='blue',
            label='Обыкновенные'
           )

plt.legend();
```



Корреляция признаков

```
# Выберим только числовые столбцы
numeric_data = data.select_dtypes(include=[np.number])
sns.heatmap(numeric_data.corr(), cmap=plt.cm.PuBuGn)
```

```
<Axes: >
```




```
# Удаление коррелирующих признаков
data_uncorr = data.drop(feats, axis=1)
data_uncorr.columns
```

```
Index(['#', 'Name', 'Type 1', 'Type 2', 'Total', 'HP', 'Attack',
'Defense',
       'Speed', 'Generation', 'Legendary'],
      dtype='object')
```