

## Praca domowa 1

Eksploracja zbioru <https://www.apispreadsheets.com/datasets/129> Import bibliotek i wczytanie danych

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv("forest_fires_dataset.csv")
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 517 entries, 0 to 516
Data columns (total 13 columns):
 #   Column   Non-Null Count  Dtype  
---  --  
 0   X         517 non-null    int64  
 1   Y         517 non-null    int64  
 2   month     517 non-null    object  
 3   day       517 non-null    object  
 4   FFMC      517 non-null    float64 
 5   DMC       517 non-null    float64 
 6   DC        517 non-null    float64 
 7   ISI        517 non-null    float64 
 8   temp      517 non-null    float64 
 9   RH        517 non-null    float64 
 10  wind      517 non-null    float64 
 11  rain      517 non-null    float64 
 12  area      517 non-null    float64 
dtypes: float64(9), int64(2), object(2)
memory usage: 52.6+ KB
```

Od razu możemy zauważyc, że nie mamy braków danych, więc nie musimy się tym przejmować.

```
df.describe()
```

```
X
Y
FFMC
DMC
DC
ISI
temp
RH
wind
rain
```

area  
count  
517.000000  
517.000000  
517.000000  
517.000000  
517.000000  
517.000000  
517.000000  
517.000000  
517.000000  
517.000000  
517.000000  
517.000000  
mean  
4.669246  
4.299807  
90.644681  
110.872340  
547.940039  
9.021663  
18.889168  
44.288201  
4.017602  
0.021663  
12.847292  
std  
2.313778  
1.229900  
5.520111  
64.046482  
248.066192  
4.559477  
5.806625  
16.317469  
1.791653  
0.295959

63.655818

min

1.000000

2.000000

18.700000

1.100000

7.900000

0.000000

2.200000

15.000000

0.400000

0.000000

0.000000

25%

3.000000

4.000000

90.200000

68.600000

437.700000

6.500000

15.500000

33.000000

2.700000

0.000000

0.000000

50%

4.000000

4.000000

91.600000

108.300000

664.200000

8.400000

19.300000

42.000000

4.000000

0.000000

```
0.520000  
75%  
7.000000  
5.000000  
92.900000  
142.400000  
713.900000  
10.800000  
22.800000  
53.000000  
4.900000  
0.000000  
6.570000  
max  
9.000000  
9.000000  
96.200000  
291.300000  
860.600000  
56.100000  
33.300000  
100.000000  
9.400000  
6.400000  
1090.840000
```

```
df.hist(figsize=(21, 12), bins=50)  
plt.show()
```

Naszą zmienną objaśnianą jest powierzchnia lasu. Widzimy, że ma ona rozkład skośny do 0.

Wśród zmiennych objaśniających mamy:

- zmienne kategoryczne (X, Y, month, day)
- zmienne numeryczne (reszta) ##### Wyjaśnijmy skrótowe zmienne: -FFMC - wilgotność ściulk w lesie -DMC - wilgotność warstw organicznych średniej głębokości -DC - wilgotność błębokich warstw -ISI - określa jak szybko pożar może się rozprzestrzenić -RH - wilgotność powietrza

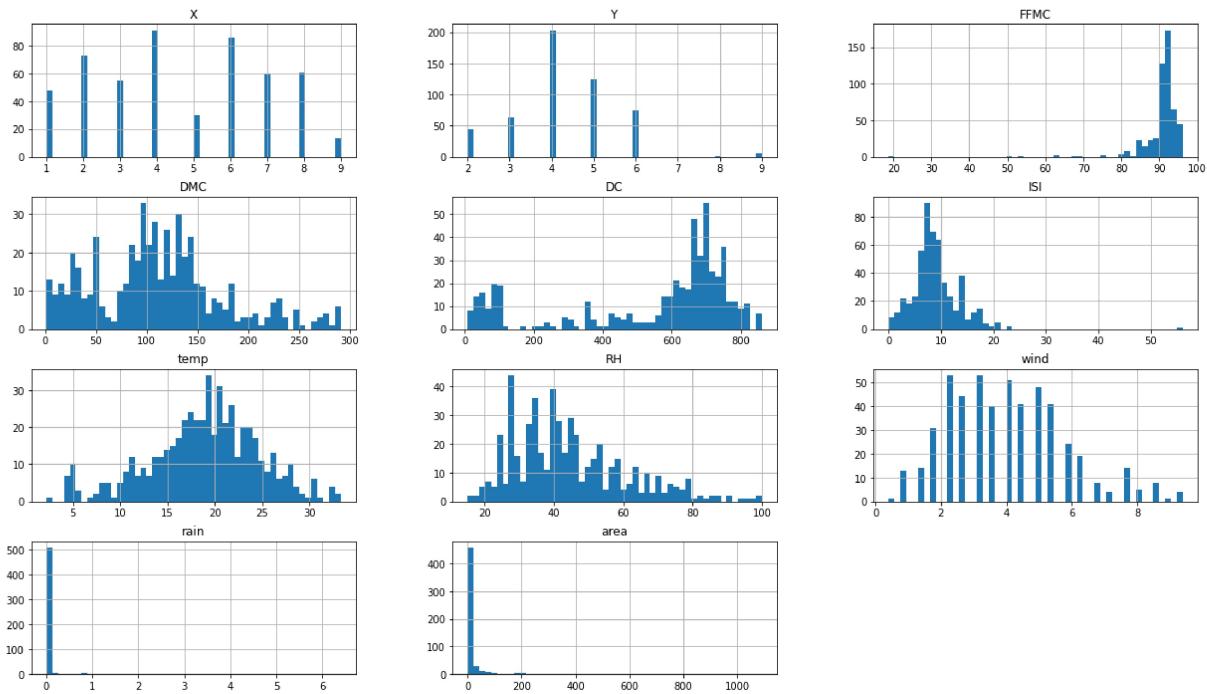


Figure 1: png

```
plt.figure(figsize=(16,16))
sns.heatmap(df.corr(), annot=True)
plt.show()
```

### Heatmapa korelacji

#### Przenalizujmy teraz zależności pomiędzy zmiennymi

```
sns.pairplot(df.drop('area', axis=1))
plt.show()
```

Możemy zauważać korelację pomiędzy współczynnikami wilgotności gleby na różnych poziomach między sobą oraz z temperaturą. Możemy zauważać, że w niektórych obserwacjach istnieją pojedyncze obserwacje odstające zaburzające odbiór wykresów. Szczególnie rzuca się w oczy wyjątkowo duże obserwacje odstające w kolumnach rain i ISI. Zobaczmy zatem jak wykresy będzie wyglądał bez nich.

```
tdf = df.loc[(df['rain']<=5) & (df['ISI']<=40)]
sns.pairplot(tdf.drop('area', axis=1))
plt.show()
```

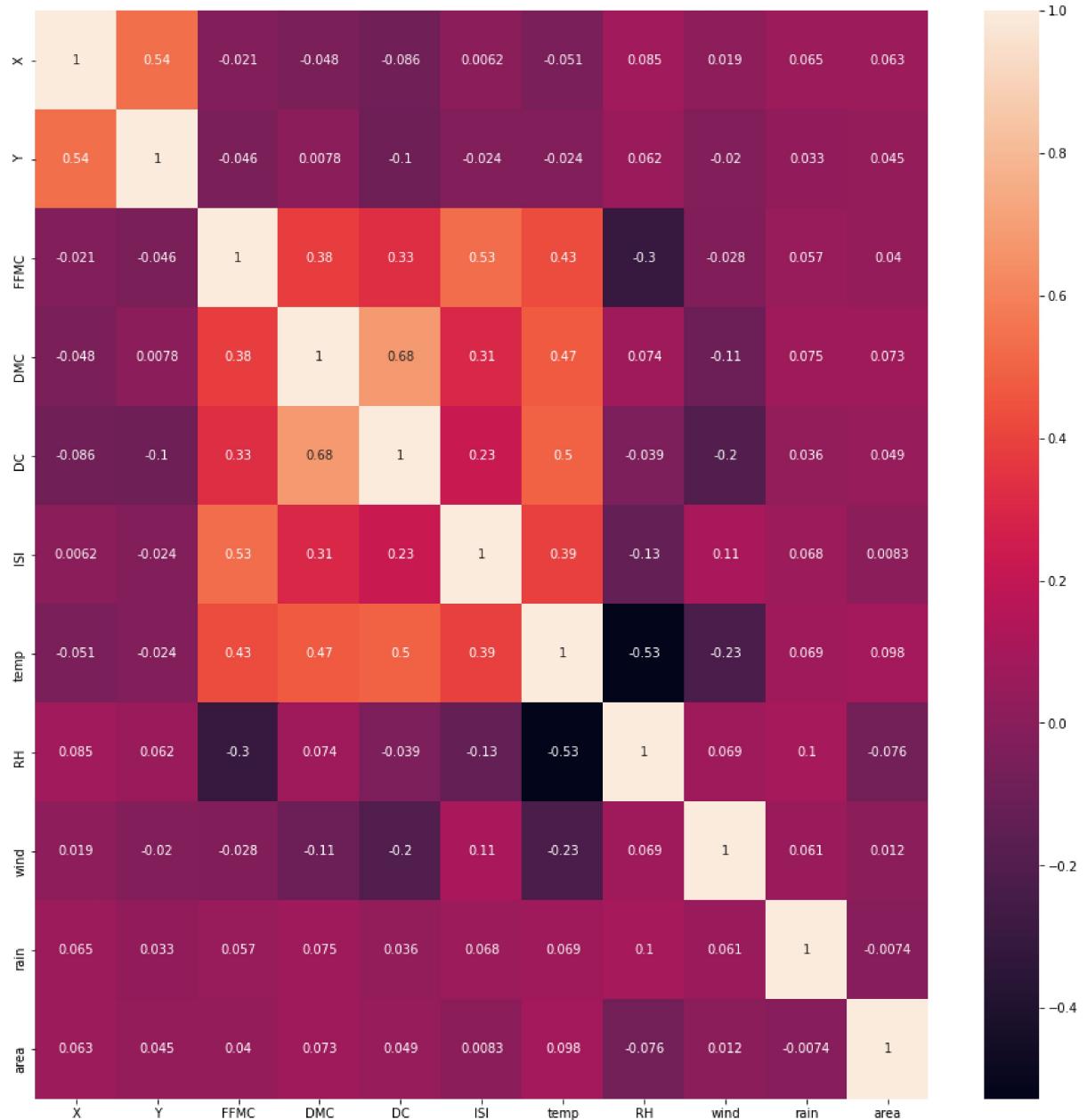


Figure 2: png

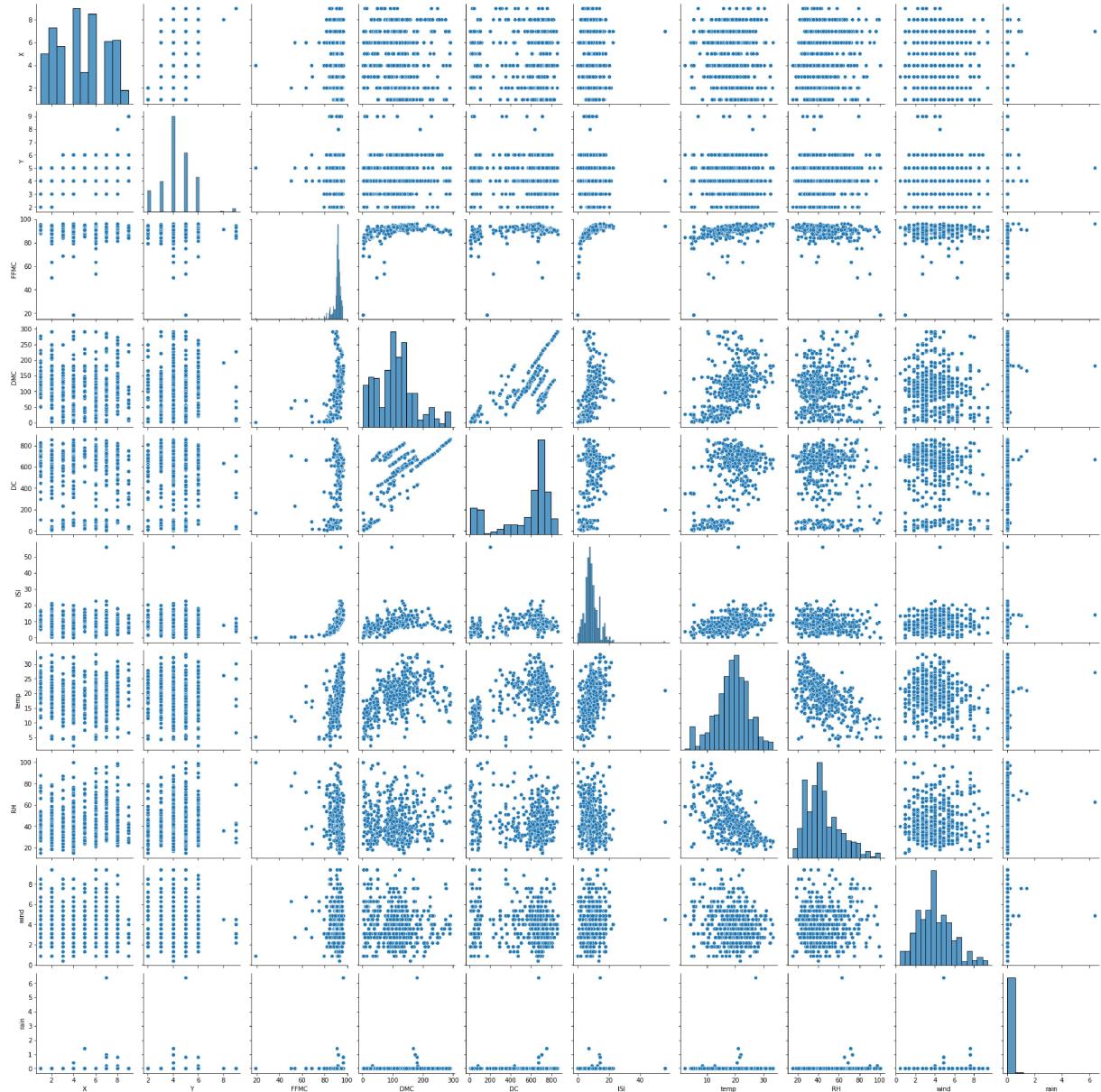


Figure 3: png

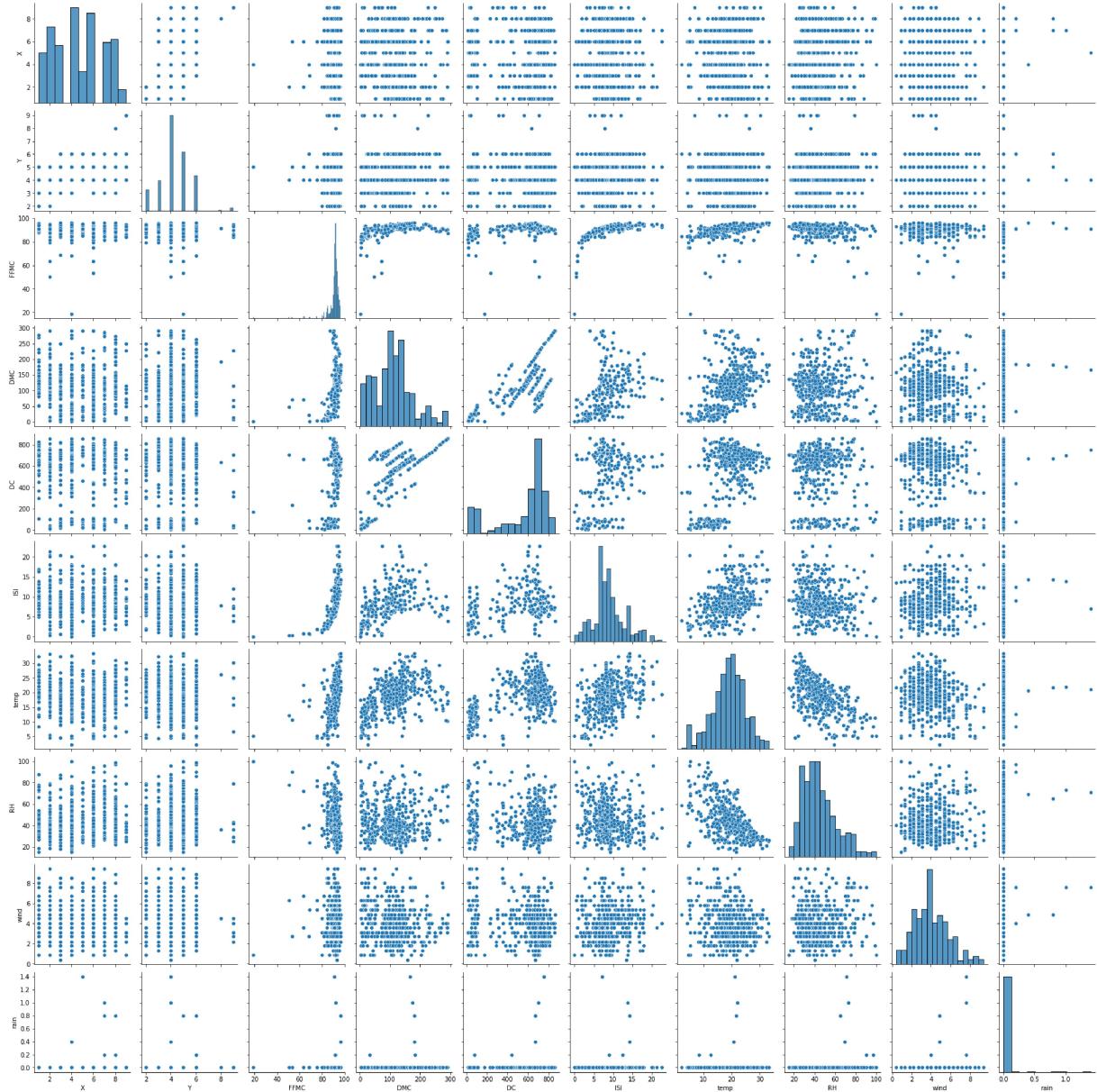


Figure 4: png

Raz jeszcze widzimy skorelowanie wcześniej wspomnianych współczynników. Zauważmy, że współżędne na tych wykresach nie mówią nam za wiele. Sprawdźmy lecz czy występują jakieś miejsca wyjątkowo narażone na pożary.

```
fdf = (df.groupby(["X", "Y"]).size().reset_index(name='fires_n')).pivot(index="Y", columns="X", values="fires_n")
sns.heatmap(fdf, cmap="Reds")
plt.show()
```

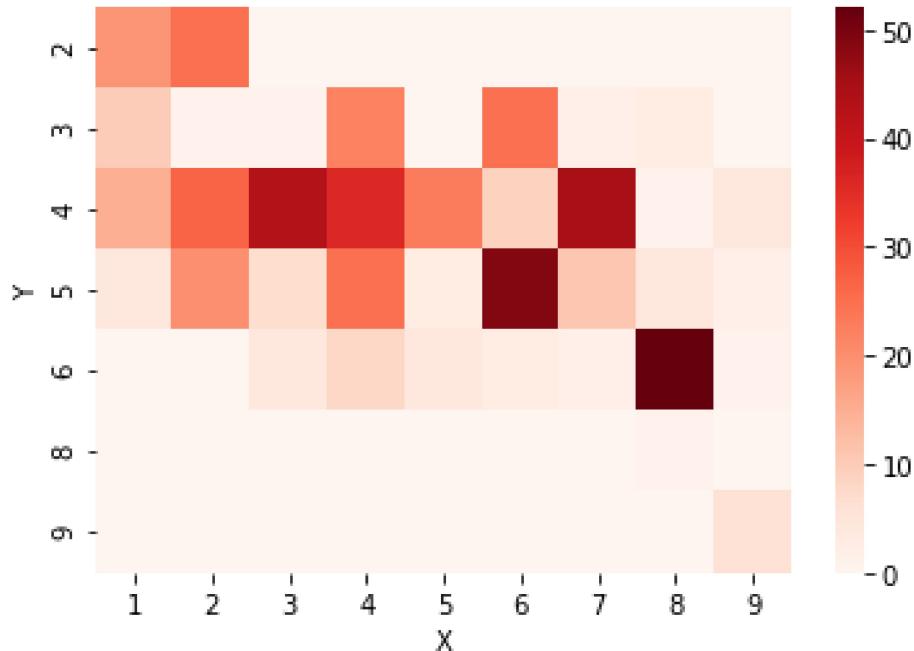


Figure 5: png

Możemy zauważyc, że znacznie wyróżniają się miejsca, w których pożary występują wyjątkowo często.

```
months_order=['jan', 'feb', 'mar', 'apr', 'may', 'jun', 'jul', 'aug', 'sep', 'oct', 'nov', 'dec']
grouped_by_month = df.groupby("month").size().reindex(months_order)
sns.lineplot(data=grouped_by_month)
plt.show()
```

```
days_order=['mon', 'tue', 'wed', 'thu', 'fri', 'sat', 'sun']
grouped_by_day=df.groupby("day").size().reindex(days_order)
sns.lineplot(data=grouped_by_day).axes.set_ylim(0,110)
plt.show()
```

**Spójrzmy teraz jak na liczbę pożarów wpływa miesiąc i dzień** Wyraźnie widać, że pożary znacznie częściej występowali w wakacyjne miesiące. Wyróżnia się również spora liczba pożarów w marcu. Możemy także zauważyc, że więcej pożarów miało miejsce w weekendy.

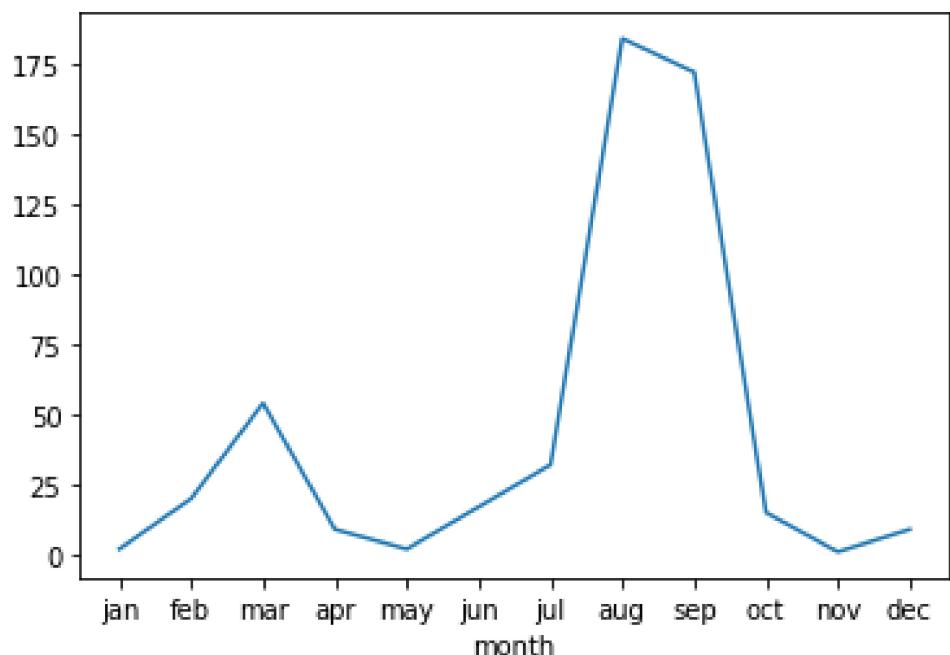


Figure 6: png

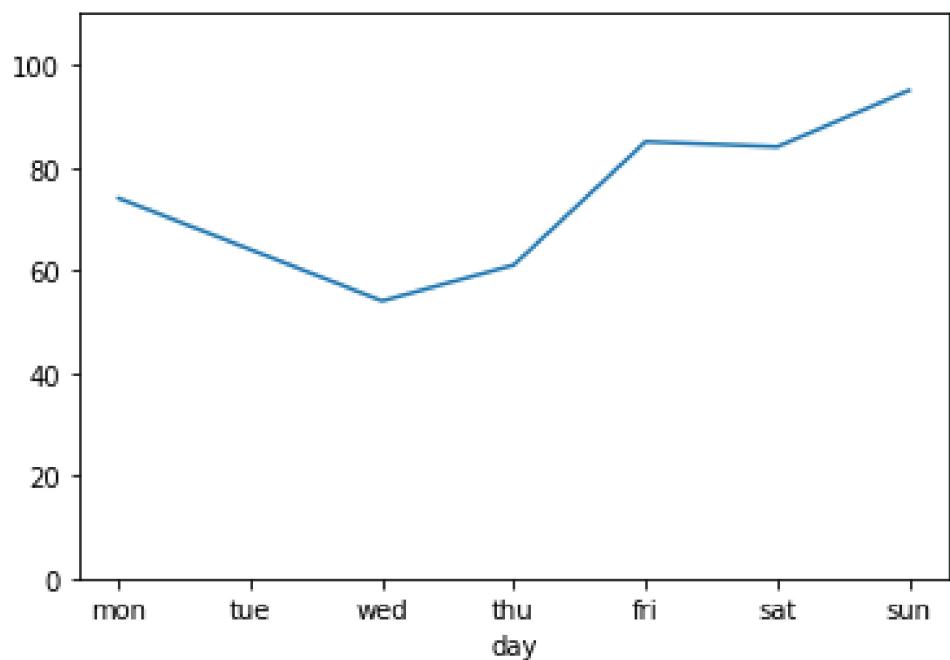


Figure 7: png

## Pakiet do automatycznej eksploracji

```
from pandas_profiling import ProfileReport

profile = ProfileReport(df, title="Pandas Profiling Report")
profile.to_notebook_iframe()

profile.to_file("raport.html")
```

Narzędzie to pozwala na dosyć szybkie (zależne od wielkości danych) przeanalizowanie danego zbioru. Oczywiście jest ograniczone i nie zapewnia takich możliwości jak własna analiza. Na pewno narzędzie to może pomóc przy wstępnej analizie danych i po przeanalizowaniu może pozwolić nam do zabrania się za bardziej szczegółowej, wnikliwej analizy.