PD1

March 30, 2021

Dominik Pawlak Wstęp do Uczenia Maszynowego Praca Domowa 1

1 Wczytanie bibliotek, zbioru danych i ich opisu

[1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib
```

[2]:
```
data = pd.read_csv('forest_fires_dataset.csv')
data.head()
```

[2]:

	X	Y	month	day	FFMC	DMC	DC	ISI	temp	RH	wind	rain	area
0	7	5	mar	fri	86.2	26.2	94.3	5.1	8.2	51.0	6.7	0.0	0.0
1	7	4	oct	tue	90.6	35.4	669.1	6.7	18.0	33.0	0.9	0.0	0.0
2	7	4	oct	sat	90.6	43.7	686.9	6.7	14.6	33.0	1.3	0.0	0.0
3	8	6	mar	fri	91.7	33.3	77.5	9.0	8.3	97.0	4.0	0.2	0.0
4	8	6	mar	sun	89.3	51.3	102.2	9.6	11.4	99.0	1.8	0.0	0.0

[3]:
```
desc = pd.read_csv('attributes_forest_fires.csv')
desc
```

[3]:

	name	type	description
0	X	integer	x-axis spatial coordinate within the Montesinh…
1	Y	integer	y-axis spatial coordinate within the Montesinh…
2	month	string	month of the year: 'jan' to 'dec'
3	day	string	day of the week: 'mon' to 'sun'
4	FFMC	float	FFMC index from the FWI system: 18.7 to 96.20
5	DMC	float	DMC index from the FWI system: 1.1 to 291.3
6	DC	float	DC index from the FWI system: 7.9 to 860.6
7	ISI	float	ISI index from the FWI system: 0.0 to 56.10
8	temp	float	temperature in Celsius degrees: 2.2 to 33.30
9	RH	float	relative humidity in %: 15.0 to 100
10	wind	float	wind speed in km/h: 0.40 to 9.40
11	rain	float	outside rain in mm/m2 : 0.0 to 6.4
12	area	float	the burned area of the forest (in ha): 0.00 to…

2 Wstępna eksploracja zbioru danych

[4]:
```
data.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
```

[5]:
```
data.describe()
```

[5]:

	X	Y	FFMC	DMC	DC	ISI	\
count	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	
std	2.313778	1.229900	5.520111	64.046482	248.066192	4.559477	
min	1.000000	2.000000	18.700000	1.100000	7.900000	0.000000	
25%	3.000000	4.000000	90.200000	68.600000	437.700000	6.500000	
50%	4.000000	4.000000	91.600000	108.300000	664.200000	8.400000	
75%	7.000000	5.000000	92.900000	142.400000	713.900000	10.800000	
max	9.000000	9.000000	96.200000	291.300000	860.600000	56.100000	

	temp	RH	wind	rain	area
count	517.000000	517.000000	517.000000	517.000000	517.000000
mean	18.889168	44.288201	4.017602	0.021663	12.847292
std	5.806625	16.317469	1.791653	0.295959	63.655818
min	2.200000	15.000000	0.400000	0.000000	0.000000
25%	15.500000	33.000000	2.700000	0.000000	0.000000
50%	19.300000	42.000000	4.000000	0.000000	0.520000
75%	22.800000	53.000000	4.900000	0.000000	6.570000
max	33.300000	100.000000	9.400000	6.400000	1090.840000

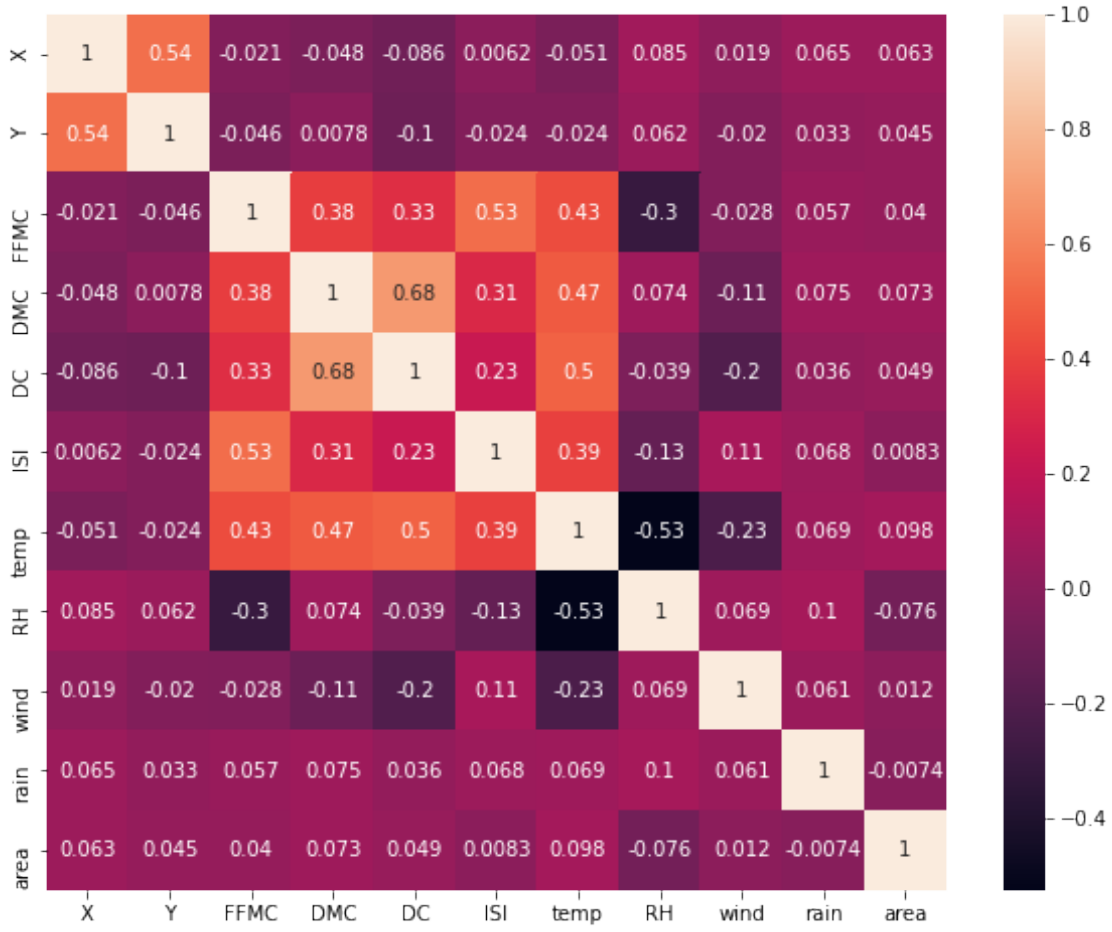
Wniosek: mamy dane kompletne, bez żadnych braków. W dwóch kolumnach mamy dane kat-

egoryczne, oznaczające miesiące i dni tygodnia. Zamieńmy je na numer miesiąca / tygodnia, a następnie stwórzmy macierz korelacji dla zadanej ramki danych.

[6]:
```
days = data['day']
mon = data['month']

for i in range(len(data)):
    if data.iloc[i, 3]=='mon': data.iloc[i, 3]=1;
    elif data.iloc[i, 3]=='tue': data.iloc[i, 3]=2;
    elif data.iloc[i, 3]=='wed': data.iloc[i, 3]=3;
    elif data.iloc[i, 3]=='thu': data.iloc[i, 3]=4;
    elif data.iloc[i, 3]=='fri': data.iloc[i, 3]=5;
    elif data.iloc[i, 3]=='sat': data.iloc[i, 3]=6;
    elif data.iloc[i, 3]=='sun': data.iloc[i, 3]=7;
    if data.iloc[i, 2]=='jan': data.iloc[i, 2]=1;
    elif data.iloc[i, 2]=='feb': data.iloc[i, 2]=2;
    elif data.iloc[i, 2]=='mar': data.iloc[i, 2]=3;
    elif data.iloc[i, 2]=='apr': data.iloc[i, 2]=4;
    elif data.iloc[i, 2]=='may': data.iloc[i, 2]=5;
    elif data.iloc[i, 2]=='jun': data.iloc[i, 2]=6;
    elif data.iloc[i, 2]=='jul': data.iloc[i, 2]=7;
    elif data.iloc[i, 2]=='aug': data.iloc[i, 2]=8;
    elif data.iloc[i, 2]=='sep': data.iloc[i, 2]=9;
    elif data.iloc[i, 2]=='oct': data.iloc[i, 2]=10;
    elif data.iloc[i, 2]=='nov': data.iloc[i, 2]=11;
    elif data.iloc[i, 2]=='dec': data.iloc[i, 2]=12;
```

[7]:
```
corr = data.corr()
f, ax = plt.subplots(figsize=(10, 8))
ax = sns.heatmap(corr,
            xticklabels=corr.columns.values,
            yticklabels=corr.columns.values,
           annot = True)
```

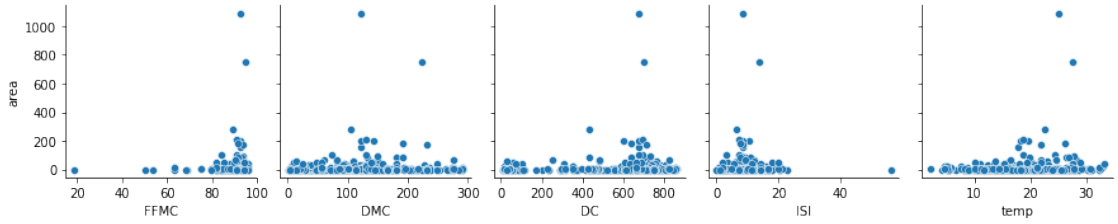


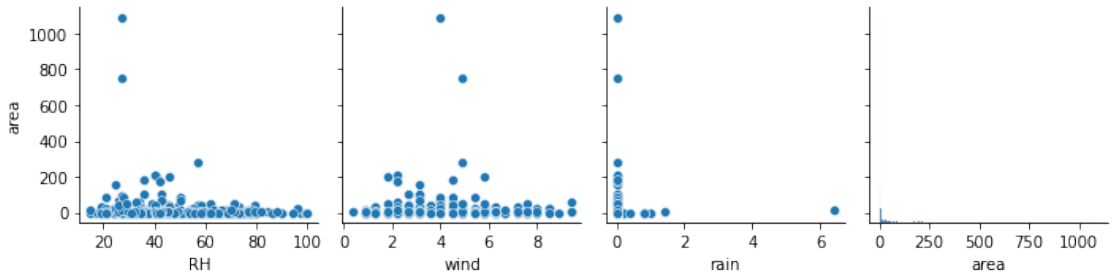
Widzimy, że poszczególne kolumny danych są ze sobą słabo skorelowane, bowiem największy współczynnik korelacji między dwiema kolumnami wynosi tylko 0.68.

Sprawdźmy jak poszczególne zmienne są skorelowane ze zmienną wynikową. W analizie zdecydowałem się pominąć kolumny położenia i daty.

[8]:
```
sns.pairplot(data, y_vars="area", x_vars=data.columns.values[4:9])
sns.pairplot(data, y_vars="area", x_vars=data.columns.values[9:])

plt.show()
```

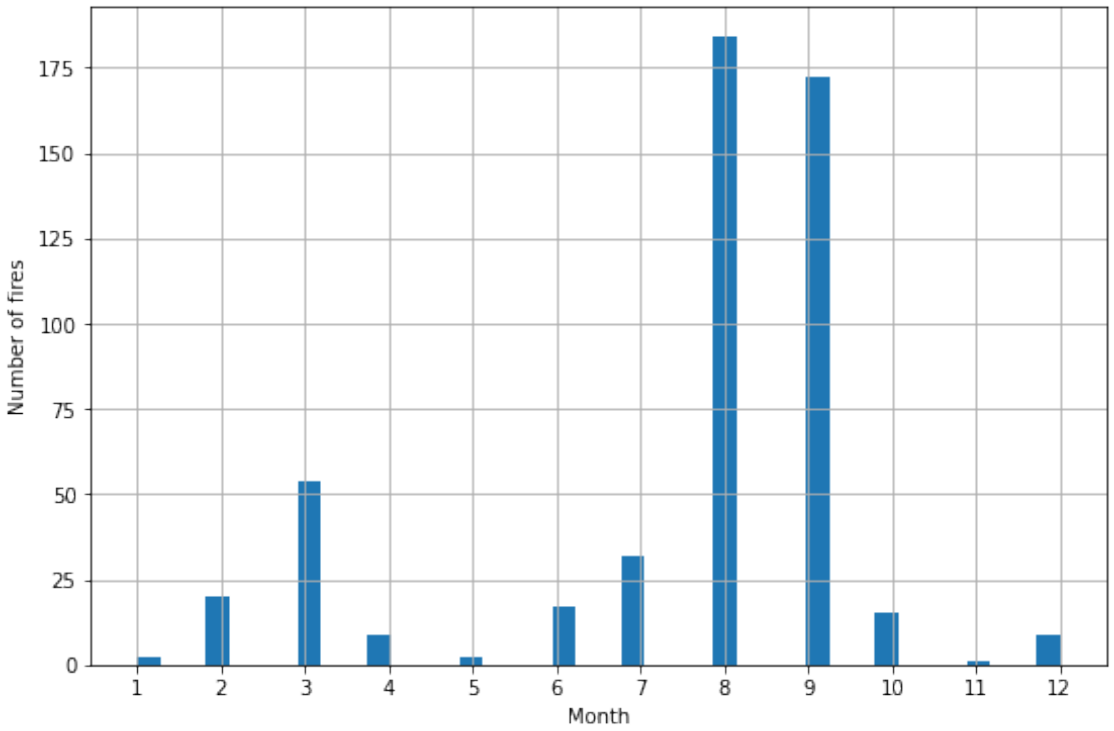


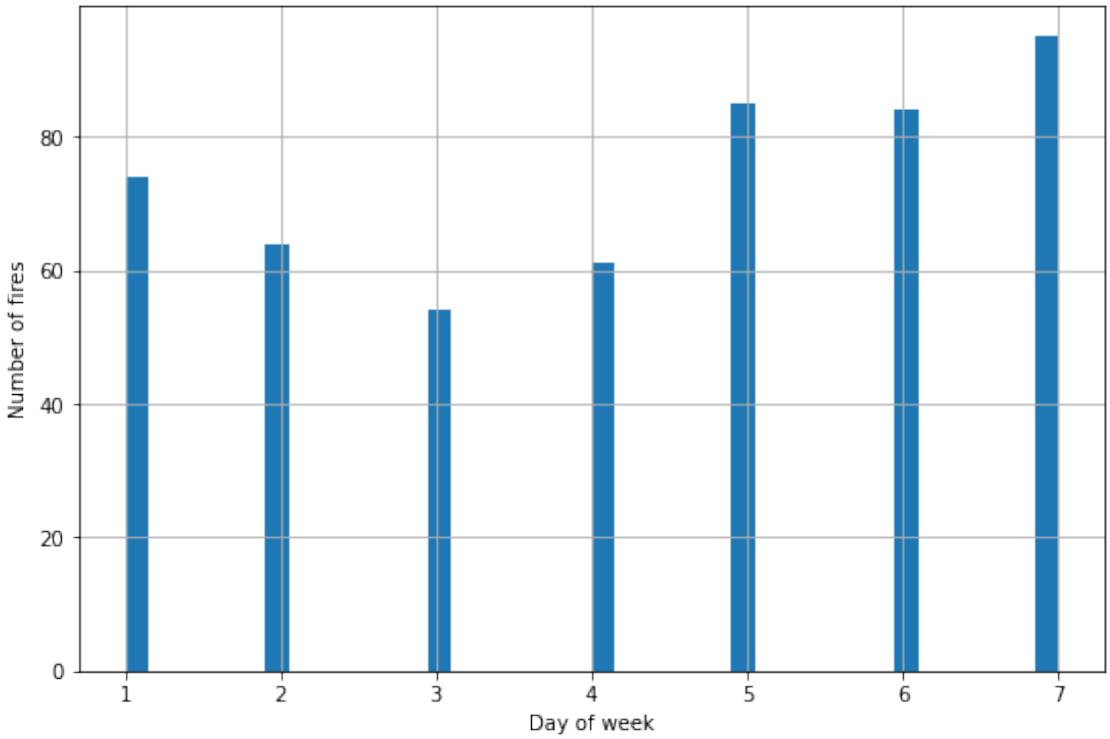


Dla kolumn z datą stwórzmy histogramy.

[9]:
```
data['month'].hist(bins = 40, figsize=(9, 6))
plt.ylabel('Number of fires')
plt.xlabel('Month');
plt.xticks(range(1, 13))
plt.show()

data['day'].hist(bins = 40, figsize=(9, 6))
plt.ylabel('Number of fires')
plt.xlabel('Day of week');
plt.xticks(range(1, 8))
plt.show()
```

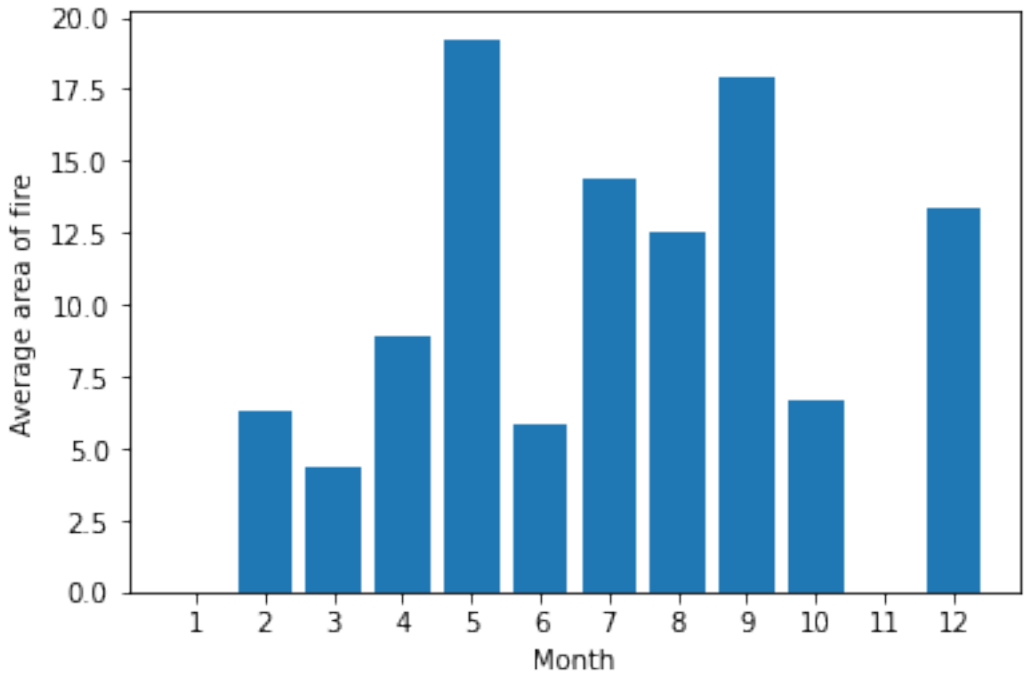




Widzimy, że najwięcej pożarów występuje w sierpniu i wrześniu. Natomiast wśród dni tygodnia, żaden dzień się nie wyróżnia.

[10]:
```
df_months = data.loc[:, ['month', 'area']].groupby('month').mean()

plt.bar(range(1, 13), df_months['area'])
plt.xticks(range(1,13))
plt.ylabel('Average area of fire')
plt.xlabel('Month');
plt.show()
```



Wykres średniego rozmiaru pożaru w poszczególnych miesiącach pokazuje, że największe pożary są maju, na tym wykresie nie jest zauważalny żaden znany rozkład. Ciekawe również są "0" w styczniu i listopadzie.

3 Zmienna area

Przyjrzyjmy się bliżej najważniejszej zmiennej w naszej ramce

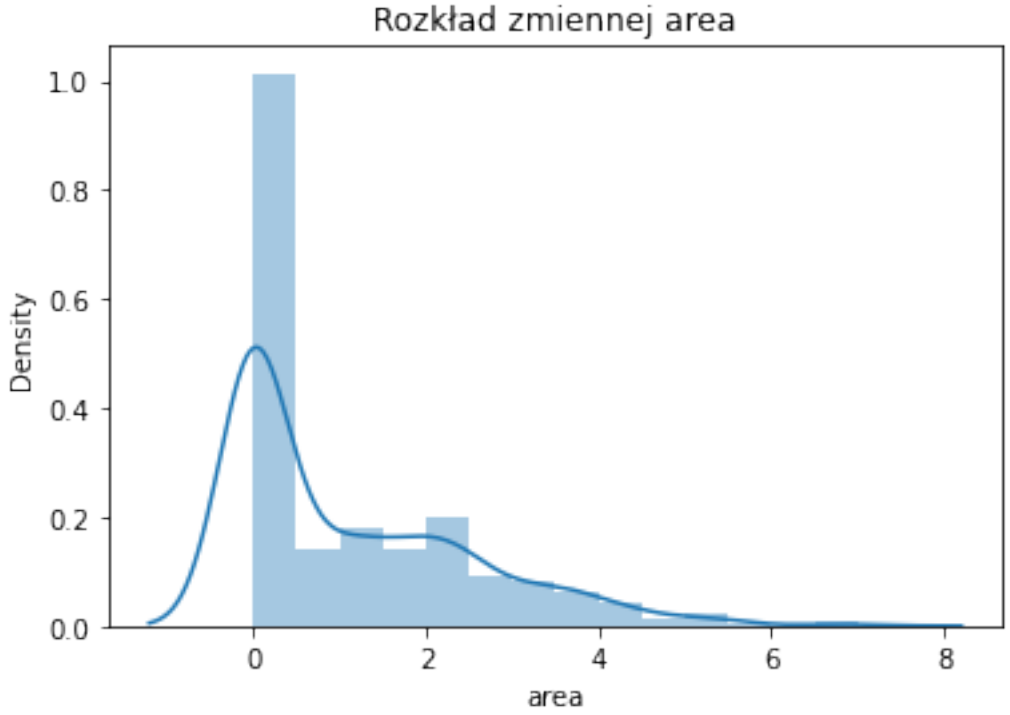
[11]:
```
print(data['area'].describe())

plot_dens=sns.distplot(np.log1p(data['area']))
plot_dens.set_title('Rozkład zmiennej area')
plt.show()
```

```
count     517.000000
mean       12.847292
std        63.655818
min         0.000000
25%         0.000000
50%         0.520000
75%         6.570000
max      1090.840000
Name: area, dtype: float64
```

```
c:\users\domin\appdata\local\programs\python\python37\lib\site-
packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a
```

```
deprecated function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar flexibility)
or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

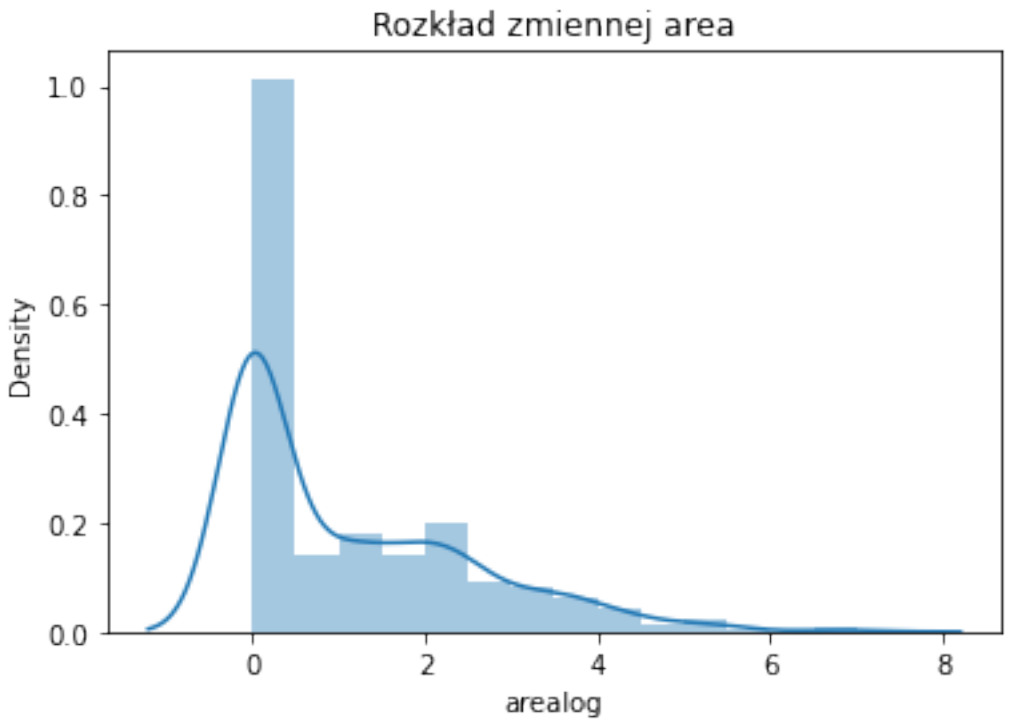


Przedstawiony histogram jest"mało przyjemny". Zgodnie z radą autorów ramki, zlogarytmujmy jego osie.

[12]:
```
area = data['area']
arealog = np.log(area+1)
data['arealog'] = arealog

plot_dens=sns.distplot(data['arealog'])
plot_dens.set_title('Rozkład zmiennej area')
plt.show()
```

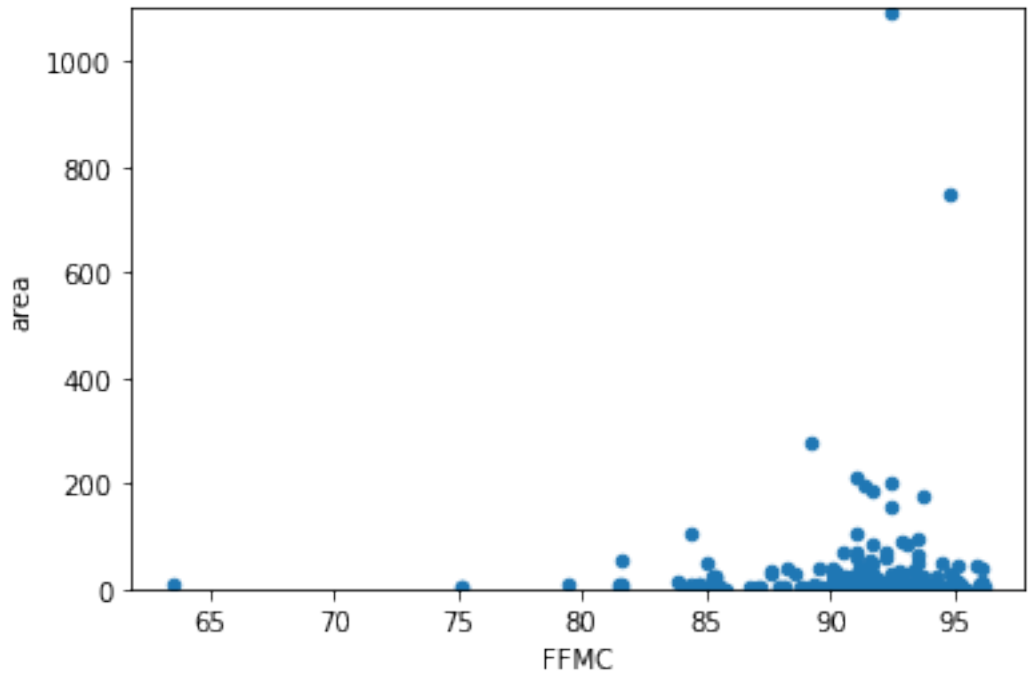
```
c:\users\domin\appdata\local\programs\python\python37\lib\site-
packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a
deprecated function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar flexibility)
or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



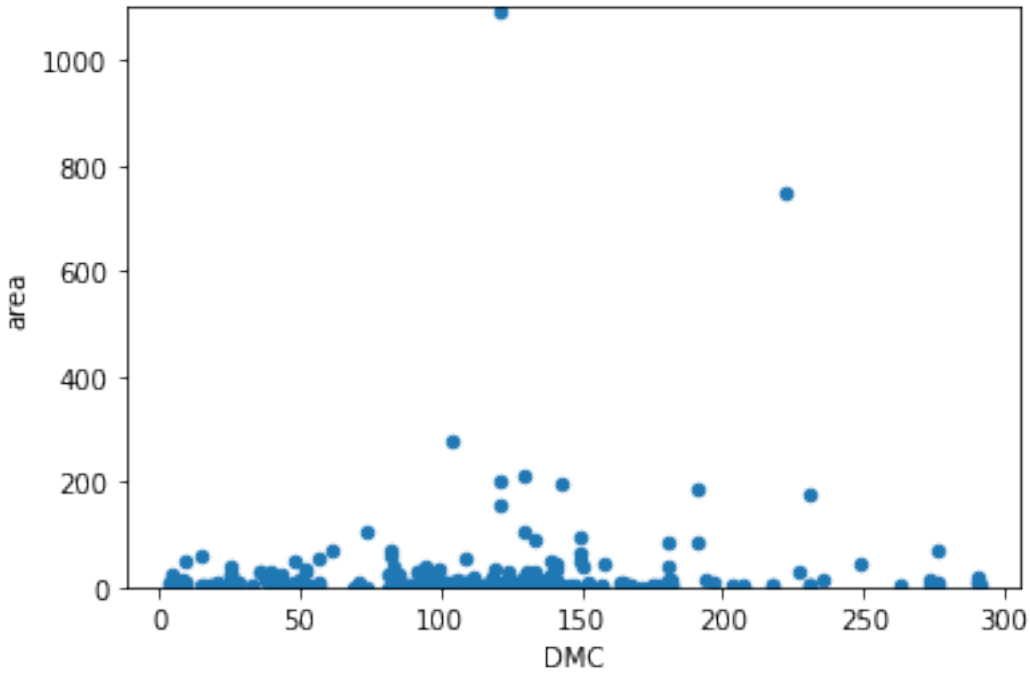
4 Korelacje poszczególnych zmiennych ze zmienną area

W poniższych testach wykorzystamy jedynie rekordy, w których zmienna *area* jest różna od 0

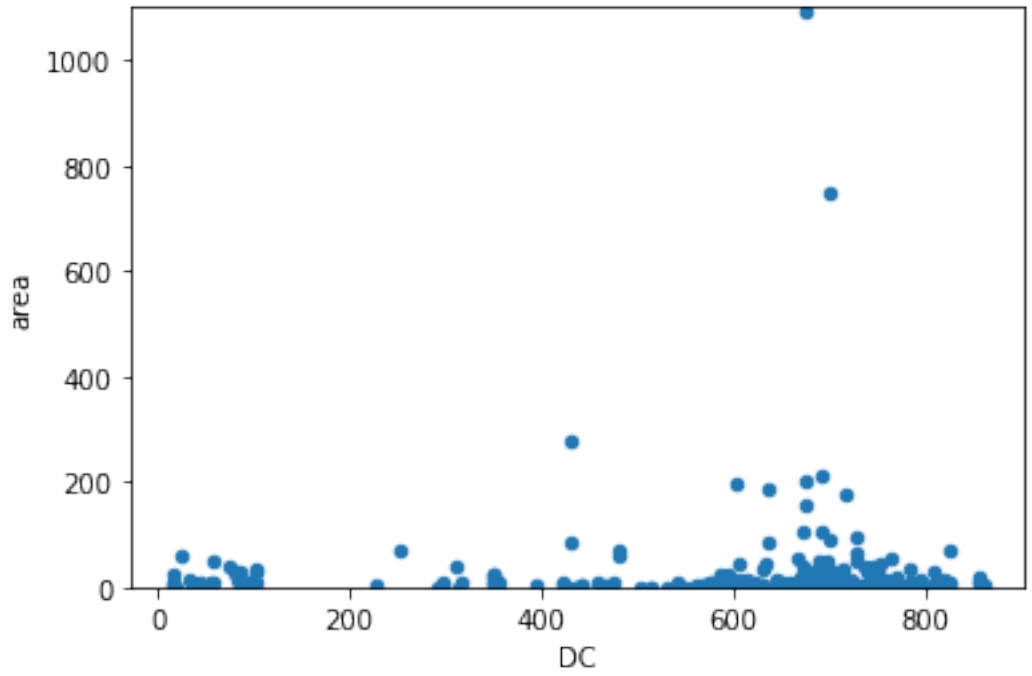
[13]:
```
data0 = data.query("area != 0")
var = 'FFMC'
plot_1 = pd.concat([data0['area'], data0[var]], axis=1)
data0.plot.scatter(x=var, y='area', ylim=(0,1100));
```



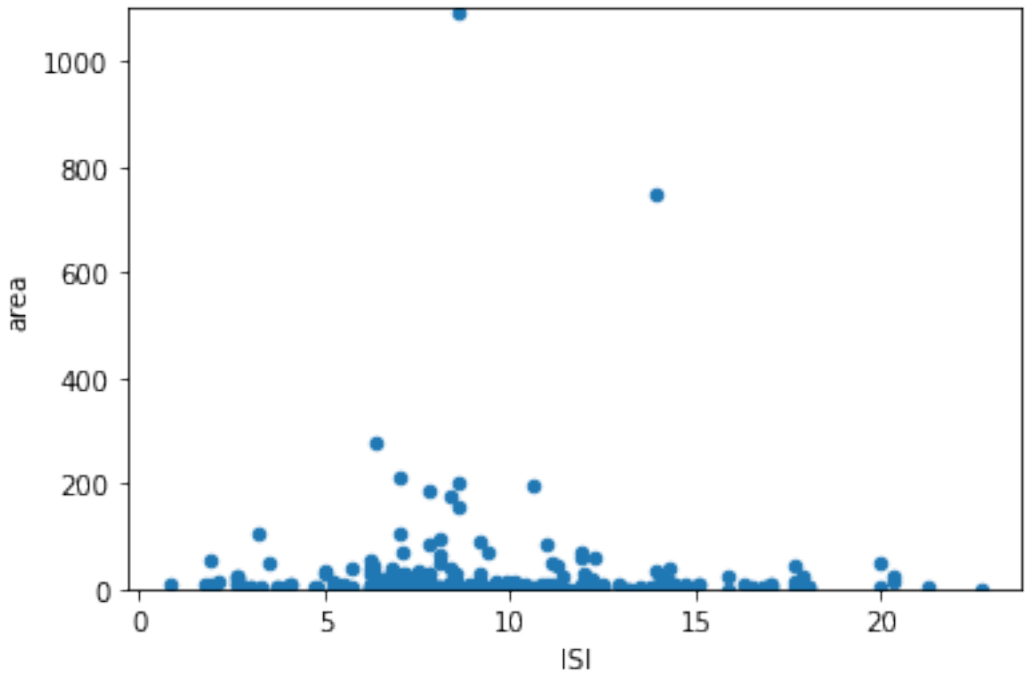
[14]:
```
var = 'DMC'
plot_1 = pd.concat([data0['area'], data0[var]], axis=1)
data0.plot.scatter(x=var, y='area', ylim=(0,1100));
```



[15]:
```
var = 'DC'
plot_1 = pd.concat([data0['DC'], data0[var]], axis=1)
data0.plot.scatter(x=var, y='area', ylim=(0,1100));
```

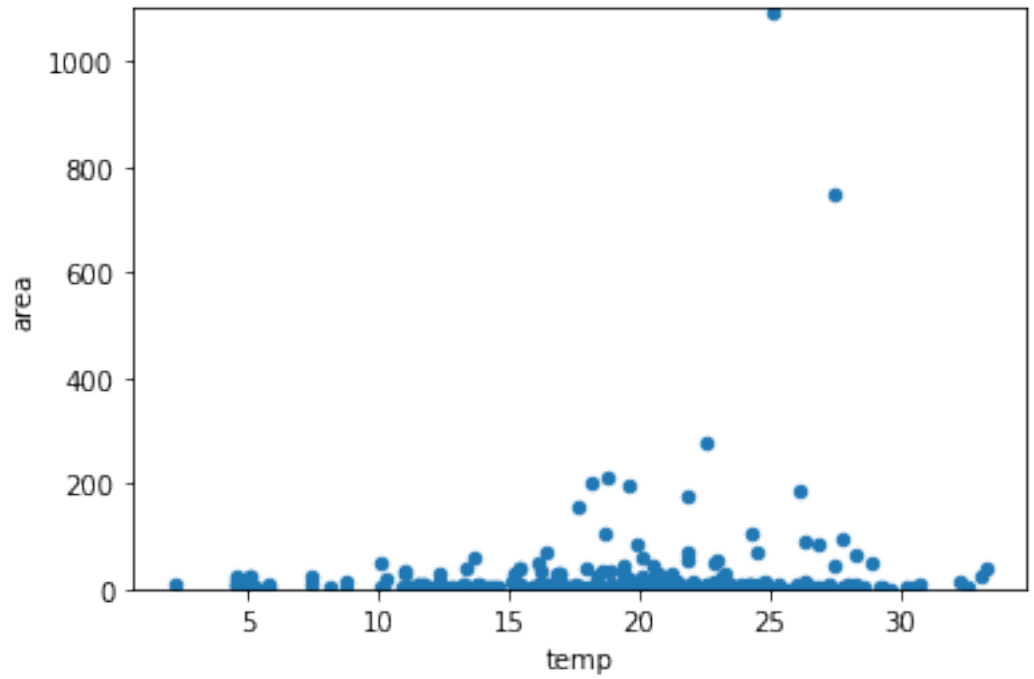


[16]:
```
var = 'ISI'
plot_1 = pd.concat([data0['ISI'], data0[var]], axis=1)
data0.plot.scatter(x=var, y='area', ylim=(0,1100));
```

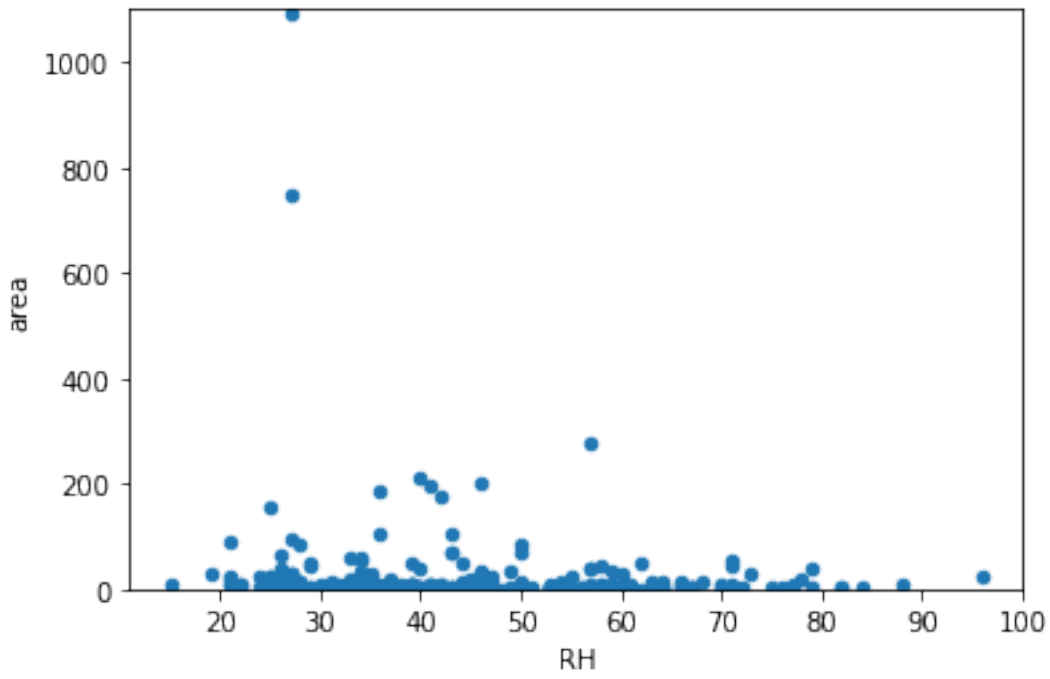


Niestety na powyższych wykresach nie widać na pierwszy rzut oka żadnej korelacji ani znanego rozkładu statystycznego. Sprawdźmy jeszcze czynniki atmosferyczne.

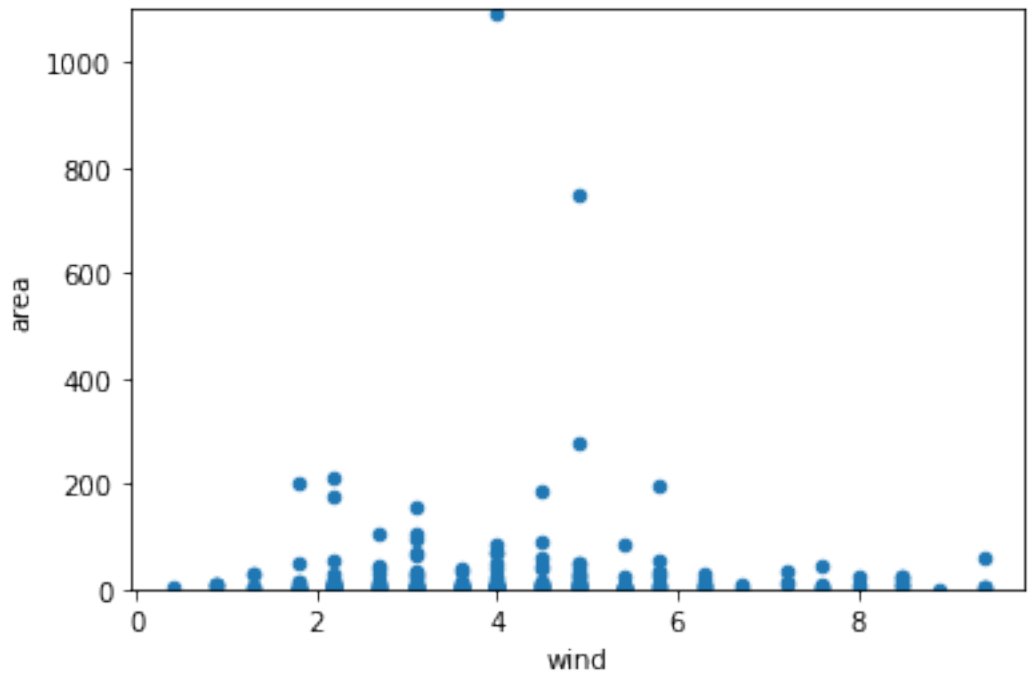
[17]:
```
var = 'temp'
plot_1 = pd.concat([data0['temp'], data0[var]], axis=1)
data0.plot.scatter(x=var, y='area', ylim=(0,1100));
```



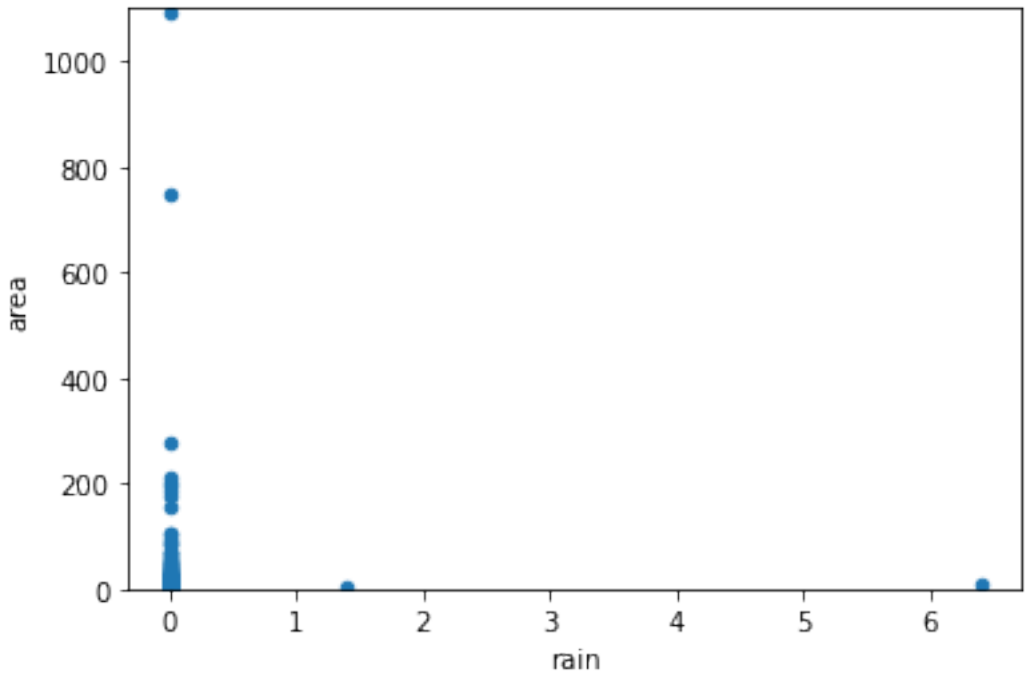
[18]:
```
var = 'RH'
plot_1 = pd.concat([data0['RH'], data0[var]], axis=1)
data0.plot.scatter(x=var, y='area', ylim=(0,1100));
```



[19]:
```
var = 'wind'
plot_1 = pd.concat([data0['wind'], data0[var]], axis=1)
data0.plot.scatter(x=var, y='area', ylim=(0,1100));
```



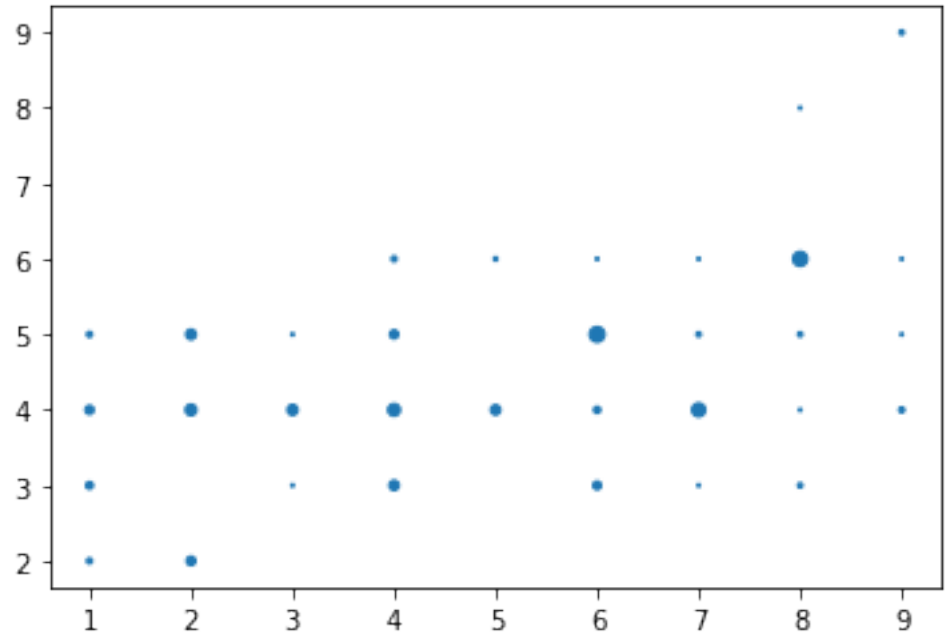
[20]:
```
var = 'rain'
plot_1 = pd.concat([data0['rain'], data0[var]], axis=1)
data0.plot.scatter(x=var, y='area', ylim=(0,1100));
```



Wnioski z powyższych wykresów. Czynniki sprzyjające dużym pożarom: -wysoka temperatura -niska wilgotność -średni wiatr, o wartości około 4-6 km/h -brak opadów

Poniżej uproszczona mapka parku pokazująca rozmieszczenie i rozmiar pożaru

[21]:
```
df_map = data0.loc[:, ['X', 'Y', 'area']].groupby(by = ['X', 'Y']).count().
 ↪reset_index()
plot_1 = plt.scatter(x = df_map['X'], y = df_map['Y'], s = df_map['area'])
plt.show()
```



Na “mapce” widzimy, że najwięcej pożarów wybuchło w środkowej części parku.