ForestFires

June 11, 2018

Załadowanie bibliotek

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
%matplotlib inline
from matplotlib import pyplot as plt

#Ustalenie stylu wykresów jako ggplot
#plt.style.use('ggplot')
from sklearn import preprocessing
from sklearn import tree
import sklearn.metrics as metrics
from sklearn.model_selection import train_test_split
import pydotplus
from IPython.display import Image
from sklearn.model_selection import cross_val_score
from sklearn.metrics import r2_score, mean_absolute_error
```

Wczytanie danych

In [2]:
```
def dataframe_size_formated(dataframe, extra=""):
    print("Rozmiar danych{}: ".format(extra), dataframe.shape)

# Ustalenie ścieżki do datasetu
filename_forestfires = './forestfires.csv'

# Wczytanie datasetu jako dataframe
forestfires_dataframe = pd.read_csv(filename_forestfires, sep=";")
```

```
# Wyświetlenie dataframe
display(forestfires_dataframe)
```

	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	6.7	0.0	0.00
1	7	4	oct	tue	90.6	35.4	669.1	6.7	18.0	33	0.9	0.0	0.00
2	7	4	oct	sat	90.6	43.7	686.9	6.7	14.6	33	1.3	0.0	0.00
3	8	6	mar	fri	91.7	33.3	77.5	9.0	8.3	97	4.0	0.2	0.00
4	8	6	mar	sun	89.3	51.3	102.2	9.6	11.4	99	1.8	0.0	0.00
5	8	6	aug	sun	92.3	85.3	488.0	14.7	22.2	29	5.4	0.0	0.00
6	8	6	aug	mon	92.3	88.9	495.6	8.5	24.1	27	3.1	0.0	0.00
7	8	6	aug	mon	91.5	145.4	608.2	10.7	8.0	86	2.2	0.0	0.00
8	8	6	sep	tue	91.0	129.5	692.6	7.0	13.1	63	5.4	0.0	0.00
9	7	5	sep	sat	92.5	88.0	698.6	7.1	22.8	40	4.0	0.0	0.00
10	7	5	sep	sat	92.5	88.0	698.6	7.1	17.8	51	7.2	0.0	0.00
11	7	5	sep	sat	92.8	73.2	713.0	22.6	19.3	38	4.0	0.0	0.00
12	6	5	aug	fri	63.5	70.8	665.3	0.8	17.0	72	6.7	0.0	0.00
13	6	5	sep	mon	90.9	126.5	686.5	7.0	21.3	42	2.2	0.0	0.00
14	6	5	sep	wed	92.9	133.3	699.6	9.2	26.4	21	4.5	0.0	0.00
15	6	5	sep	fri	93.3	141.2	713.9	13.9	22.9	44	5.4	0.0	0.00
16	5	5	mar	sat	91.7	35.8	80.8	7.8	15.1	27	5.4	0.0	0.00
17	8	5	oct	mon	84.9	32.8	664.2	3.0	16.7	47	4.9	0.0	0.00
18	6	4	mar	wed	89.2	27.9	70.8	6.3	15.9	35	4.0	0.0	0.00
19	6	4	apr	sat	86.3	27.4	97.1	5.1	9.3	44	4.5	0.0	0.00
20	6	4	sep	tue	91.0	129.5	692.6	7.0	18.3	40	2.7	0.0	0.00
21	5	4	sep	mon	91.8	78.5	724.3	9.2	19.1	38	2.7	0.0	0.00
22	7	4	jun	sun	94.3	96.3	200.0	56.1	21.0	44	4.5	0.0	0.00
23	7	4	aug	sat	90.2	110.9	537.4	6.2	19.5	43	5.8	0.0	0.00
24	7	4	aug	sat	93.5	139.4	594.2	20.3	23.7	32	5.8	0.0	0.00
25	7	4	aug	sun	91.4	142.4	601.4	10.6	16.3	60	5.4	0.0	0.00
26	7	4	sep	fri	92.4	117.9	668.0	12.2	19.0	34	5.8	0.0	0.00
27	7	4	sep	mon	90.9	126.5	686.5	7.0	19.4	48	1.3	0.0	0.00
28	6	3	sep	sat	93.4	145.4	721.4	8.1	30.2	24	2.7	0.0	0.00
29	6	3	sep	sun	93.5	149.3	728.6	8.1	22.8	39	3.6	0.0	0.00
..	..	..	...	...	...	...	...	...	...	..	...	...	...
487	5	4	aug	tue	95.1	141.3	605.8	17.7	26.4	34	3.6	0.0	16.40
488	4	4	aug	tue	95.1	141.3	605.8	17.7	19.4	71	7.6	0.0	46.70
489	4	4	aug	wed	95.1	141.3	605.8	17.7	20.6	58	1.3	0.0	0.00

```
490  4  4   aug  wed  95.1  141.3  605.8  17.7  28.7  33   4.0   0.0   0.00
491  4  4   aug  thu  95.8  152.0  624.1  13.8  32.4  21   4.5   0.0   0.00
492  1  3   aug  fri  95.9  158.0  633.6  11.3  32.4  27   2.2   0.0   0.00
493  1  3   aug  fri  95.9  158.0  633.6  11.3  27.5  29   4.5   0.0  43.32
494  6  6   aug  sat  96.0  164.0  643.0  14.0  30.8  30   4.9   0.0   8.59
495  6  6   aug  mon  96.2  175.5  661.8  16.8  23.9  42   2.2   0.0   0.00
496  4  5   aug  mon  96.2  175.5  661.8  16.8  32.6  26   3.1   0.0   2.77
497  3  4   aug  tue  96.1  181.1  671.2  14.3  32.3  27   2.2   0.0  14.68
498  6  5   aug  tue  96.1  181.1  671.2  14.3  33.3  26   2.7   0.0  40.54
499  7  5   aug  tue  96.1  181.1  671.2  14.3  27.3  63   4.9   6.4  10.82
500  8  6   aug  tue  96.1  181.1  671.2  14.3  21.6  65   4.9   0.8   0.00
501  7  5   aug  tue  96.1  181.1  671.2  14.3  21.6  65   4.9   0.8   0.00
502  4  4   aug  tue  96.1  181.1  671.2  14.3  20.7  69   4.9   0.4   0.00
503  2  4   aug  wed  94.5  139.4  689.1  20.0  29.2  30   4.9   0.0   1.95
504  4  3   aug  wed  94.5  139.4  689.1  20.0  28.9  29   4.9   0.0  49.59
505  1  2   aug  thu  91.0  163.2  744.4  10.1  26.7  35   1.8   0.0   5.80
506  1  2   aug  fri  91.0  166.9  752.6   7.1  18.5  73   8.5   0.0   0.00
507  2  4   aug  fri  91.0  166.9  752.6   7.1  25.9  41   3.6   0.0   0.00
508  1  2   aug  fri  91.0  166.9  752.6   7.1  25.9  41   3.6   0.0   0.00
509  5  4   aug  fri  91.0  166.9  752.6   7.1  21.1  71   7.6   1.4   2.17
510  6  5   aug  fri  91.0  166.9  752.6   7.1  18.2  62   5.4   0.0   0.43
511  8  6   aug  sun  81.6   56.7  665.6   1.9  27.8  35   2.7   0.0   0.00
512  4  3   aug  sun  81.6   56.7  665.6   1.9  27.8  32   2.7   0.0   6.44
513  2  4   aug  sun  81.6   56.7  665.6   1.9  21.9  71   5.8   0.0  54.29
514  7  4   aug  sun  81.6   56.7  665.6   1.9  21.2  70   6.7   0.0  11.16
515  1  4   aug  sat  94.4  146.0  614.7  11.3  25.6  42   4.0   0.0   0.00
516  6  3   nov  tue  79.5    3.0  106.7   1.1  11.8  31   4.5   0.0   0.00

[517 rows x 13 columns]
```

```
In [3]: dataframe_size_formated(forestfires_dataframe)

Rozmiar danych:  (517, 13)
```

Zbiór danych ma 517 wierszy i 13 kolumn (ostatnia kolumna to atrybut decyzyjny, a pozostałe 12 kolumn to atrybuty warunkowe). W celu dalszego zbadania datasetu i weryfikacji typów danych kategorycznych w każdej kolumnie, wypisano unikalne wartości każdej kolumny. Sprawdzono również, czy zbiór danych zawiera brakujące wartości lub niepotrzebne kolumny.

```
In [4]: def attributes_count(dataframe):
            print("Liczba różnych wartości atrybutów dla każdej kolumny:")
            for x in dataframe.columns:
                uniq = dataframe[x].unique()
                print("{:>8}: {:>2}".format(x, uniq.shape[0]))

        attributes_count(forestfires_dataframe)
```

```
Liczba różnych wartości atrybutów dla każdej kolumny:
       X:  9
       Y:  7
   month: 12
     day:  7
    FFMC: 106
     DMC: 215
      DC: 219
     ISI: 119
    temp: 192
      RH: 75
    wind: 21
    rain:  7
    area: 251
```

Zauważono, że spośród 12 atrybutów warunkowych, 4 z nich mają liczbę klas mniejeszą niż 10. Również atrybut month nie charakteryzuje się dużą liczbą klas. Z uwagi na chęć wyeliminowania zalezności modelu od położenia, dnia tygodnia i opadów deszczu, podjęto decyzję o usunięciu kolumn o liczbie klas mniejszej równej 10, w tym celu utworzono poniższą funkcję.

Wartości atrybutu month przekształcono w następujący sposób: 'jan'=1, 'feb'=2, . . . , 'dec'=12.

```
In [5]: def del_with_classes_no_less_than(dataframe, less):
            for col in dataframe.columns.values:
                col_unique = dataframe[col].unique()
                if len(col_unique) <= less:
                    print("Usunięto kolumnę '{}',która zawiera liczbę klas mnieszą równą {}: {}".format(col, less, col_unique))
                    dataframe = dataframe.drop(col, 1)
            return dataframe

In [6]: dataframe_size_formated(forestfires_dataframe, " przed usunięciem atrybutów")
        forestfires_dataframe = del_with_classes_no_less_than(forestfires_dataframe, 10)
        dataframe_size_formated(forestfires_dataframe, " po usunięciu atrybutów")
```

```
        forestfires_dataframe.month = forestfires_dataframe.month.map({
            'jan': 1,
            'feb': 2,
            'mar': 3,
            'apr': 4,
            'may': 5,
            'jun': 6,
            'jul': 7,
            'aug': 8,
            'sep': 9,
            'oct': 10,
            'nov': 11,
            'dec': 12,
        })
```

```
Rozmiar danych przed usunięciem atrybutów:  (517, 13)
Usunięto kolumnę 'X',która zawiera liczbę klas mnieszą równą 10: [7 8 6 5 4 2 9 1 3]
Usunięto kolumnę 'Y',która zawiera liczbę klas mnieszą równą 10: [5 4 6 3 2 9 8]
Usunięto kolumnę 'day',która zawiera liczbę klas mnieszą równą 10: ['fri' 'tue' 'sat' 'sun' 'mon' 'wed' 'thu']
Usunięto kolumnę 'rain',która zawiera liczbę klas mnieszą równą 10: [ 0.   0.2  1.   6.4  0.8  0.4  1.4]
Rozmiar danych po usunięciu atrybutów:  (517, 9)
```

```
In [7]: attributes_count(forestfires_dataframe)
```

```
Liczba różnych wartości atrybutów dla każdej kolumny:
   month: 12
    FFMC: 106
     DMC: 215
      DC: 219
     ISI: 119
    temp: 192
      RH: 75
    wind: 21
    area: 251
```

Z uwagi na brak danych kategorycznych w oczyszczonym zbiorze, kodowanie wartości atrybutów (kolumn) nie jest konieczne. Dokonać podziału danych na atrybuty warunkowe (zmienna X) i decyzyjne (zmienna Y).

```
In [8]: X = forestfires_dataframe.drop(['area'], axis=1)
        Y = forestfires_dataframe['area']
```

Kolejnym podziałem, który należy wykonać, jest podział danych na część treningową i testową. Założono, że rozmiar części testowej będzie wynosił 33% wszystkich danych. W celu zachowania powtarzalności wyników parametr random_state ustawiono na wartość 34 (ustawienie innej wartości bedzie powodowało wygenerowanie innego podziału danych i innego drzewa decyzyjnego).

```
In [9]: random_state = 34

        X_train, X_test ,Y_train, Y_test = train_test_split(X, Y, test_size = 0.33, random_state=random_state)
```

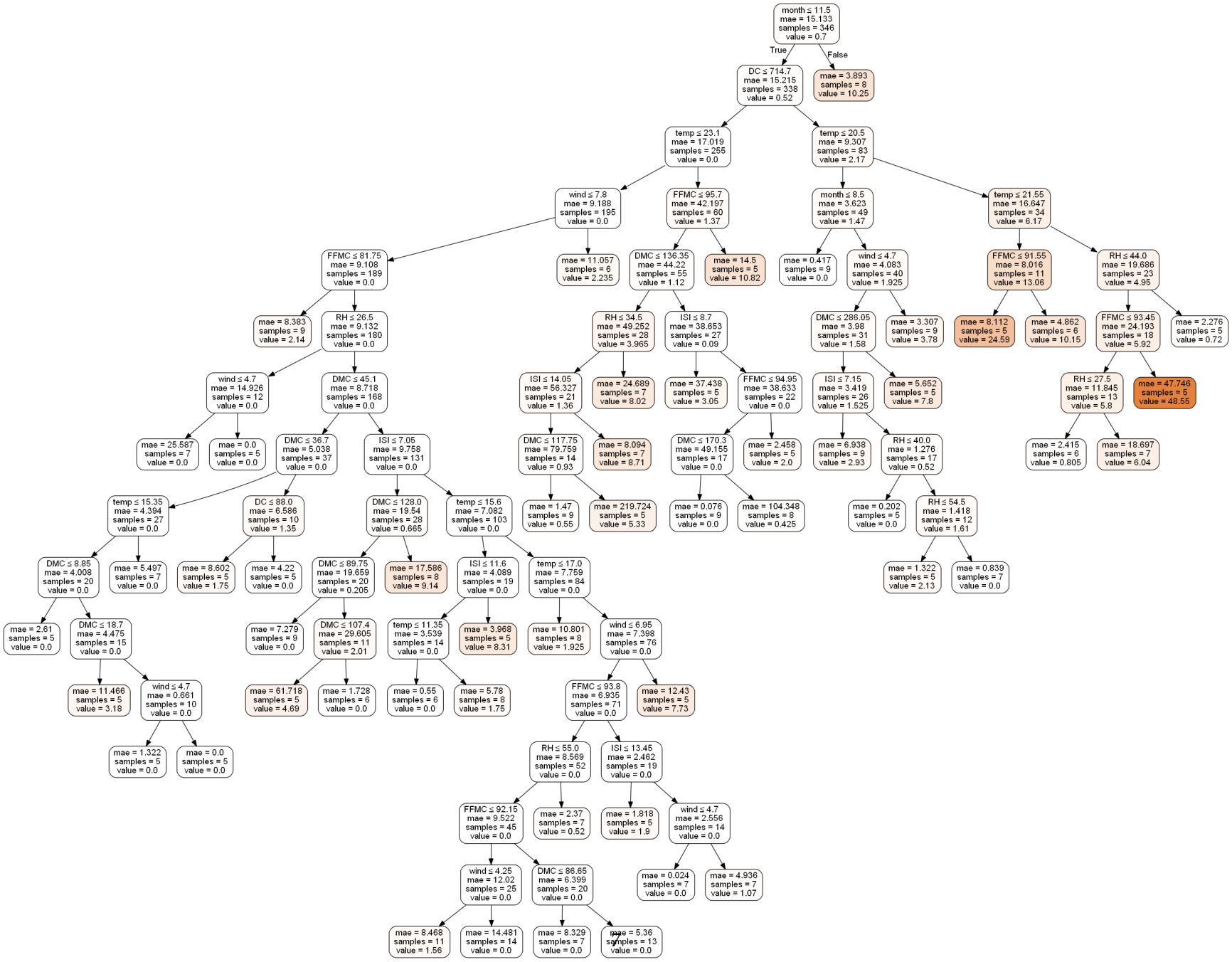
Utworzono funkcję sprawdzającą jakość klasyfikacji zbudowanego drzewa, funkcję budującą drzewo oraz funkcję sprawdzająca istotność atrybutu. Funkcji sprawdzająca jakość klasyfikacji zbudowanego drzewa, wypisuje wartości Accuracy, czyli współczynnik dokładności modelu do danych testowych, Precision - procent elementów będących istotnymi oraz Recall - procent istotnych elementów, które zostały wybrane.

```
In [10]: def build_tree(X, X_train, X_test, Y_train, Y_test, random_state, **kwargs):
             regr = tree.DecisionTreeRegressor(random_state=random_state, **kwargs)
             regr = regr.fit(X_train, Y_train)

             dot_data = tree.export_graphviz(regr, out_file=None,
                                             feature_names=X.columns,
                                             filled=True, rounded=True,
                                             special_characters=True)
             graph = pydotplus.graph_from_dot_data(dot_data)
             display(Image(graph.create_png()))

             return regr
```

W oparciu o przygotowane dane zbudowano drzewa decyzyjne.

1. Dokonano klasyfikacji zbioru danych z usuniętymi wierszami, które zawierały braki danych:

```
In [11]: regr = build_tree(X, X_train, X_test, Y_train, Y_test, random_state, max_depth=15, min_samples_leaf = 5, criterion='mae')
```

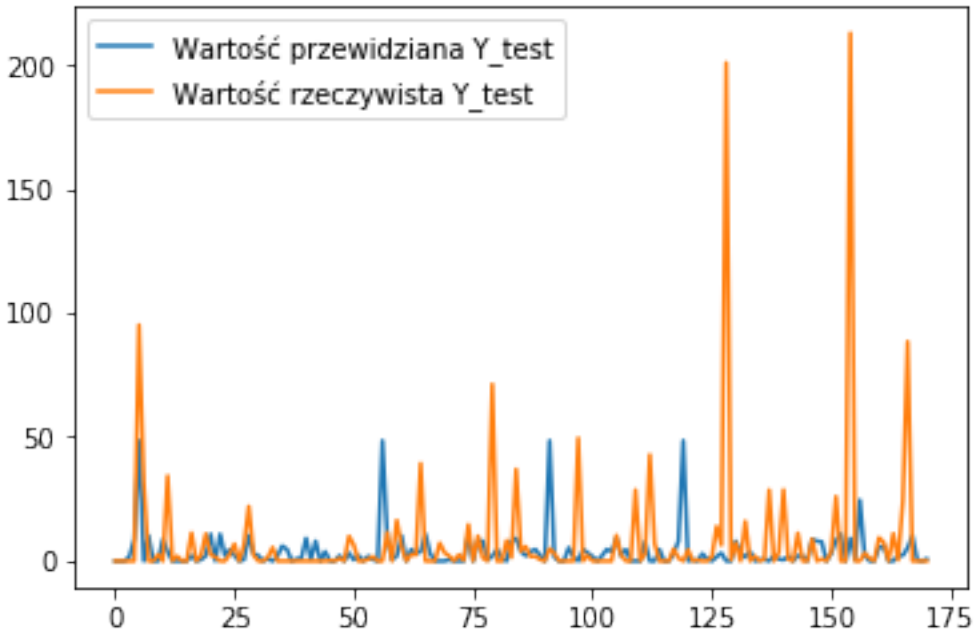

month ≤ 11.5
mae = 15.133
samples = 346
value = 0.7
True
False
DC ≤ 714.7
mae = 15.215
samples = 338
value = 0.52
mae = 3.893
samples = 8
value = 10.25
temp ≤ 23.1
mae = 17.019
samples = 255
value = 0.0
temp ≤ 20.5
mae = 9.307
samples = 83
value = 2.17
wind ≤ 7.8
mae = 9.188
samples = 195
value = 0.0
FFMC ≤ 95.7
mae = 42.197
samples = 60
value = 1.37
month ≤ 8.5
mae = 3.623
samples = 49
value = 1.47
temp ≤ 21.55
mae = 16.647
samples = 34
value = 6.17
FFMC ≤ 81.75
mae = 9.108
samples = 189
value = 0.0
mae = 11.057
samples = 6
value = 2.235
DMC ≤ 136.35
mae = 44.22
samples = 55
value = 1.12
mae = 14.5
samples = 5
value = 10.82
mae = 0.417
samples = 9
value = 0.0
wind ≤ 4.7
mae = 4.083
samples = 40
value = 1.925
FFMC ≤ 91.55
mae = 8.016
samples = 11
value = 13.06
RH ≤ 44.0
mae = 19.686
samples = 23
value = 4.95
mae = 8.383
samples = 9
value = 2.14
RH ≤ 26.5
mae = 9.132
samples = 180
value = 0.0
RH ≤ 34.5
mae = 49.252
samples = 28
value = 3.965
ISI ≤ 8.7
mae = 38.653
samples = 27
value = 0.09
DMC ≤ 286.05
mae = 3.98
samples = 31
value = 1.58
mae = 3.307
samples = 9
value = 3.78
mae = 8.112
samples = 5
value = 24.59
mae = 4.862
samples = 6
value = 10.15
FFMC ≤ 93.45
mae = 24.193
samples = 18
value = 5.92
mae = 2.276
samples = 5
value = 0.72
wind ≤ 4.7
mae = 14.926
samples = 12
value = 0.0
DMC ≤ 45.1
mae = 8.718
samples = 168
value = 0.0
ISI ≤ 14.05
mae = 56.327
samples = 21
value = 1.36
mae = 24.689
samples = 7
value = 8.02
mae = 37.438
samples = 5
value = 3.05
FFMC ≤ 94.95
mae = 38.633
samples = 22
value = 0.0
ISI ≤ 7.15
mae = 3.419
samples = 26
value = 1.525
mae = 5.652
samples = 5
value = 7.8
RH ≤ 27.5
mae = 11.845
samples = 13
value = 5.8
mae = 47.746
samples = 5
value = 48.55
mae = 25.587
samples = 7
value = 0.0
mae = 0.0
samples = 5
value = 0.0
DMC ≤ 36.7
mae = 5.038
samples = 37
value = 0.0
ISI ≤ 7.05
mae = 9.758
samples = 131
value = 0.0
DMC ≤ 117.75
mae = 79.759
samples = 14
value = 0.93
mae = 8.094
samples = 7
value = 8.71
DMC ≤ 170.3
mae = 49.155
samples = 17
value = 0.0
mae = 2.458
samples = 5
value = 2.0
mae = 6.938
samples = 9
value = 2.93
RH ≤ 40.0
mae = 1.276
samples = 17
value = 0.52
mae = 2.415
samples = 6
value = 0.805
mae = 18.697
samples = 7
value = 6.04
temp ≤ 15.35
mae = 4.394
samples = 27
value = 0.0
DC ≤ 88.0
mae = 6.586
samples = 10
value = 1.35
DMC ≤ 128.0
mae = 19.54
samples = 28
value = 0.665
temp ≤ 15.6
mae = 7.082
samples = 103
value = 0.0
mae = 1.47
samples = 9
value = 0.55
mae = 219.724
samples = 5
value = 5.33
mae = 0.076
samples = 9
value = 0.0
mae = 104.348
samples = 8
value = 0.425
mae = 0.202
samples = 5
value = 0.0
RH ≤ 54.5
mae = 1.418
samples = 12
value = 1.61
DMC ≤ 8.85
mae = 4.008
samples = 20
value = 0.0
mae = 5.497
samples = 7
value = 0.0
mae = 8.602
samples = 5
value = 1.75
mae = 4.22
samples = 5
value = 0.0
DMC ≤ 89.75
mae = 19.659
samples = 20
value = 0.205
mae = 17.586
samples = 8
value = 9.14
ISI ≤ 11.6
mae = 4.089
samples = 19
value = 0.0
temp ≤ 17.0
mae = 7.759
samples = 84
value = 0.0
mae = 1.322
samples = 5
value = 2.13
mae = 0.839
samples = 7
value = 0.0
mae = 2.61
samples = 5
value = 0.0
DMC ≤ 18.7
mae = 4.475
samples = 15
value = 0.0
mae = 7.279
samples = 9
value = 0.0
DMC ≤ 107.4
mae = 29.605
samples = 11
value = 2.01
temp ≤ 11.35
mae = 3.539
samples = 14
value = 0.0
mae = 3.968
samples = 5
value = 8.31
mae = 10.801
samples = 8
value = 1.925
wind ≤ 6.95
mae = 7.398
samples = 76
value = 0.0
mae = 11.466
samples = 5
value = 3.18
wind ≤ 4.7
mae = 0.661
samples = 10
value = 0.0
mae = 61.718
samples = 5
value = 4.69
mae = 1.728
samples = 6
value = 0.0
mae = 0.55
samples = 6
value = 0.0
mae = 5.78
samples = 8
value = 1.75
FFMC ≤ 93.8
mae = 6.935
samples = 71
value = 0.0
mae = 12.43
samples = 5
value = 7.73
mae = 1.322
samples = 5
value = 0.0
mae = 0.0
samples = 5
value = 0.0
RH ≤ 55.0
mae = 8.569
samples = 52
value = 0.0
ISI ≤ 13.45
mae = 2.462
samples = 19
value = 0.0
FFMC ≤ 92.15
mae = 9.522
samples = 45
value = 0.0
mae = 2.37
samples = 7
value = 0.52
mae = 1.818
samples = 5
value = 1.9
wind ≤ 4.7
mae = 2.556
samples = 14
value = 0.0
mae = 0.024
samples = 7
value = 0.0
mae = 4.936
samples = 7
value = 1.07
wind ≤ 4.25
mae = 12.02
samples = 25
value = 0.0
DMC ≤ 86.65
mae = 6.399
samples = 20
value = 0.0
mae = 8.468
samples = 11
value = 1.56
mae = 14.481
samples = 14
value = 0.0
mae = 8.329
samples = 7
value = 0.0
mae = 5.36
samples = 13
value = 0.0

```
In [12]: def check_tree_mae(regr, X_train, X_test, Y_train, Y_test, print_score=True):
             preds=regr.predict(X_test)

             line_pred, = plt.plot(preds, label='Wartość przewidziana Y_test')
             line_real, = plt.plot(Y_test.values, label='Wartość rzeczywista Y_test')
             plt.legend(handles=[line_pred, line_real])

             mae = mean_absolute_error(Y_test, preds)
             if print_score:
                 print('Mean absolute error: %.3f' % mae)
             return mae

         def check_tree_cross_val_score(regr, X, Y, print_score=True):
             scores = cross_val_score(regr, X, Y, cv=3, scoring='neg_mean_squared_error')
             current_score = np.mean(np.sqrt(-scores))

             if print_score:
                 print('Accuracy between Y_pred and Y_test: %.3f' % current_score)
             return current_score
```

Metody mean absolute error oraz cross_val_score (mean_squared_error) zostały wykorzystane do określenia dokładności między Y_pred a Y_test. Im mniejsza wartość obydwu parametrów tym drzewo decyzyjne jest lepiej dopasowane do danych testowych.

```
In [13]: check_tree_mae(regr, X_train, X_test, Y_train, Y_test)
         check_tree_cross_val_score(regr, X, Y)

         print("")
```

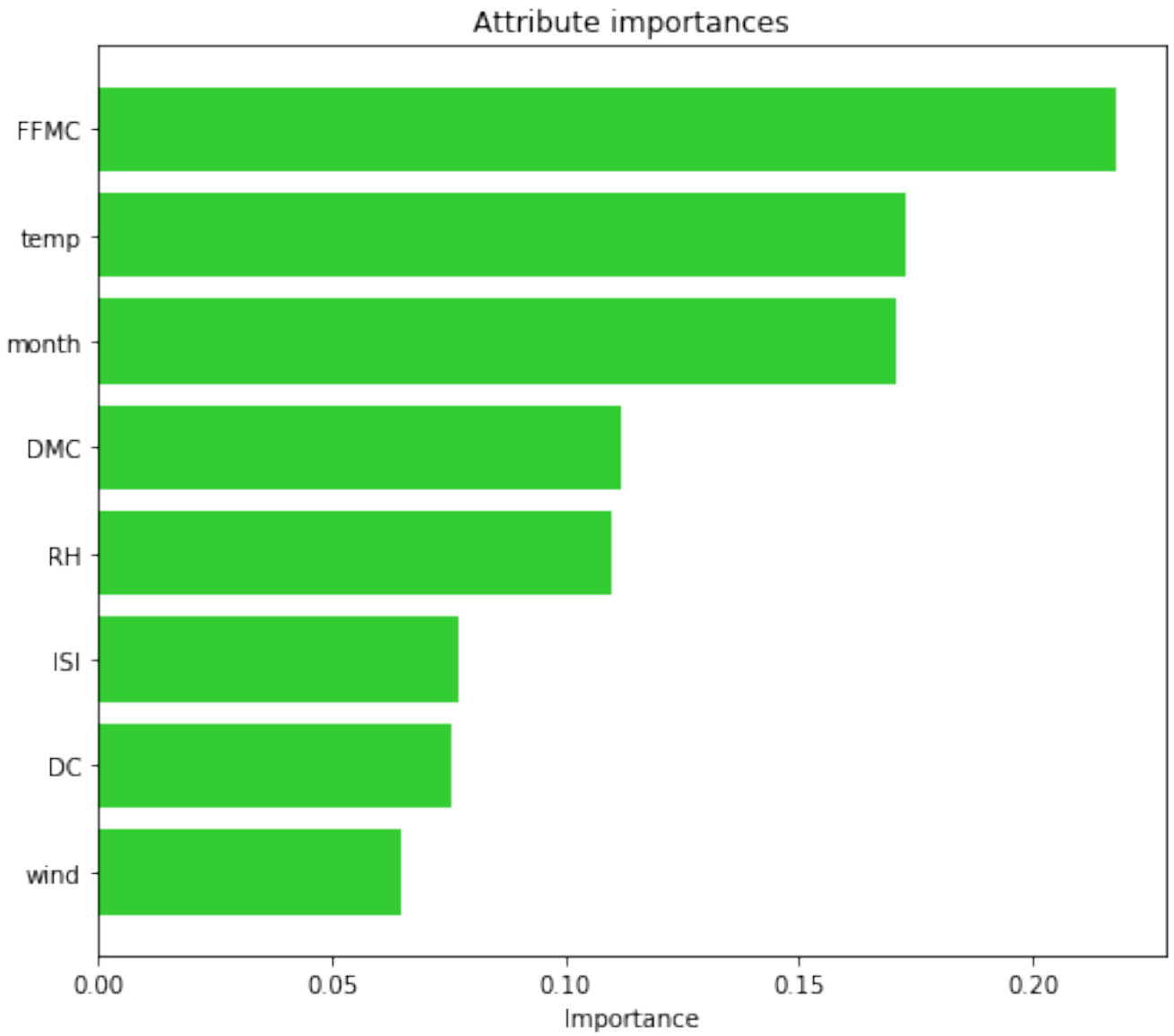
```
Mean absolute error: 9.115
Accuracy between Y_pred and Y_test: 58.656
```



Z wartości mse widać, że błąd drzewa z 2 poziomami wynosi średnio około 19 hektarów lasu na przewidywanie.

```
In [14]: def attribute_importance(regr, X):
             attrs = X.columns.values
             attr_importance = regr.feature_importances_
             sorted_attr_importance = np.argsort(attr_importance)
             range_sorted_attr_importance = range(len(sorted_attr_importance))

             plt.figure(figsize=(8, 7))
             plt.barh(range_sorted_attr_importance, attr_importance[sorted_attr_importance], color='#33cc33')
             plt.yticks(range_sorted_attr_importance, attrs[sorted_attr_importance])
             plt.xlabel('Importance')
             plt.title('Attribute importances')
             plt.draw()
             plt.show()
```

In [15]: attribute_importance(regr, X)



Wygenerowane drzewo decyzyjne posiada głębokość wynoszącą 5. Ponadto zauważono, że najważnijszymi atrybutami są: —-. Mniejsze znaczenie mają atrybuty: ——. Pozostałe atrybuty uznano za nieznaczące.

```
In [16]: print(regr.predict([[1, 20, 26, 10, 2, 30, 75, 1]]))
         print(regr.predict([[7, 100, 140, 550, 25, 30, 3, 6]]))
```

```
[ 8.02]
[ 10.82]
```