Zajęcia 24.03.2022

Biblioteki

```
import sys
import numpy as np
import pandas as pd
import os
import seaborn as sns
import matplotlib.pyplot as plt
import statsmodels.formula.api as sm
from sklearn import linear_model
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report
from sklearn.model_selection import train_test_split
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
from sklearn.feature_selection import f_classif
from sklearn.feature_selection import SelectKBest, f_classif, mutual_info_classif as MIC
from sklearn.pipeline import make_pipeline
from sklearn.svm import LinearSVC
from sklearn.metrics import classification_report
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeClassifier as DTC
from sklearn.ensemble import RandomForestClassifier as RF
from sklearn.datasets import make_classification
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import RepeatedStratifiedKFold
from numpy import mean
from numpy import std
from sklearn.naive_bayes import GaussianNB
```

##

CZĘŚĆ 1: Podstawowa eksploracja danych

##

Wydruk ścieżki do bieżącego katalogu:

print(os.getcwd())

Wczytujemy dane

data = pd.read_excel(r'C:\\Users\\User\\Desktop\\Projekt uczenie maszynowe\\Rice_MSC_Dataset.xlsx')

data.head()

	AREA	PERIMETER	MAJOR_AXIS	...	ALLdaub4YY	ALLdaub4ZZ	CLASS
0	7805	437.915	209.8215	...	0.3793	0.4733	Basmati
1	7503	340.757	138.3361	...	0.3144	0.3641	Arborio
2	5124	314.617	141.9803	...	0.3445	0.4448	Jasmine
3	7990	437.085	201.4386	...	0.4020	0.4904	Basmati
4	7433	342.893	140.3350	...	0.3303	0.3928	Arborio

[5 rows x 107 columns]

Sprawdzamy czy są jakieś missing values

data.isnull().values.any() # Są

ile_brakuje = data.isnull().sum()

print(ile_brakuje)

ile_brakuje[ile_brakuje > 0] # Zmienne dla których są jakies brakujące wartości

```
skewB         6
kurtosisB     6
skewCb        3
skewCr        2
kurtosisCb    3
kurtosisCr    2
dtype: int64
```

Usunę wiersze z brakującymi wartosciami.

data2 = data.dropna() # Nowe dane

len(data.index) - len(data2.index) # Czyli było 8 wierszy z brakującymi zmiennymi

data2 = data2.reset_index()

Tworzymy roboczy zbiór danych bez ostatniej kolumny

data3 = data2.drop(['CLASS'], axis=1)

data3 = data3.drop(['index'], axis=1)

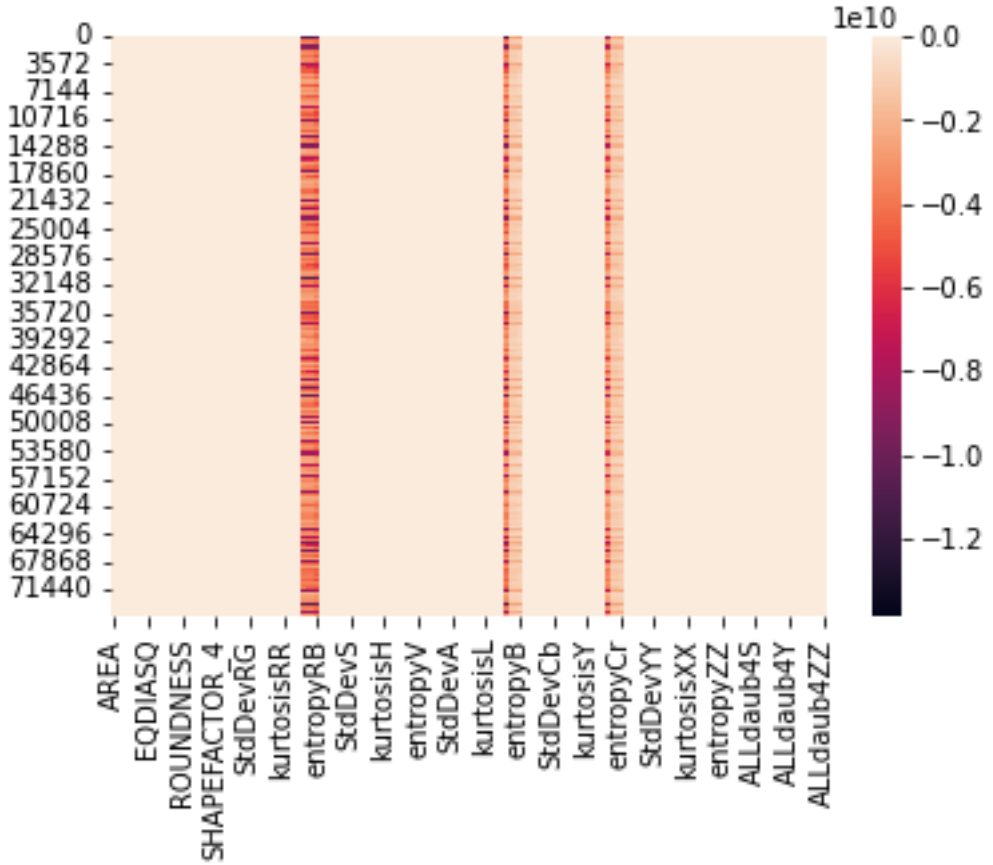
data4 = data2[['CLASS']]

Macierz korelacji

Zbadamy skorelowanie poszczególnych zmiennych (docelowo objasniających),

żeby zdecydować których nie ma sensu wspólnie używać przy przewidywaniu rodzaju ryżu.

sns.heatmap(data3) # nie widać tutaj zbyt wiele



M = data3.corr()

Wypisujemy "Mocno" dla elementów macierzy, w których korelacja > 0.9

M[abs(M) > 0.9] = 'Mocno'

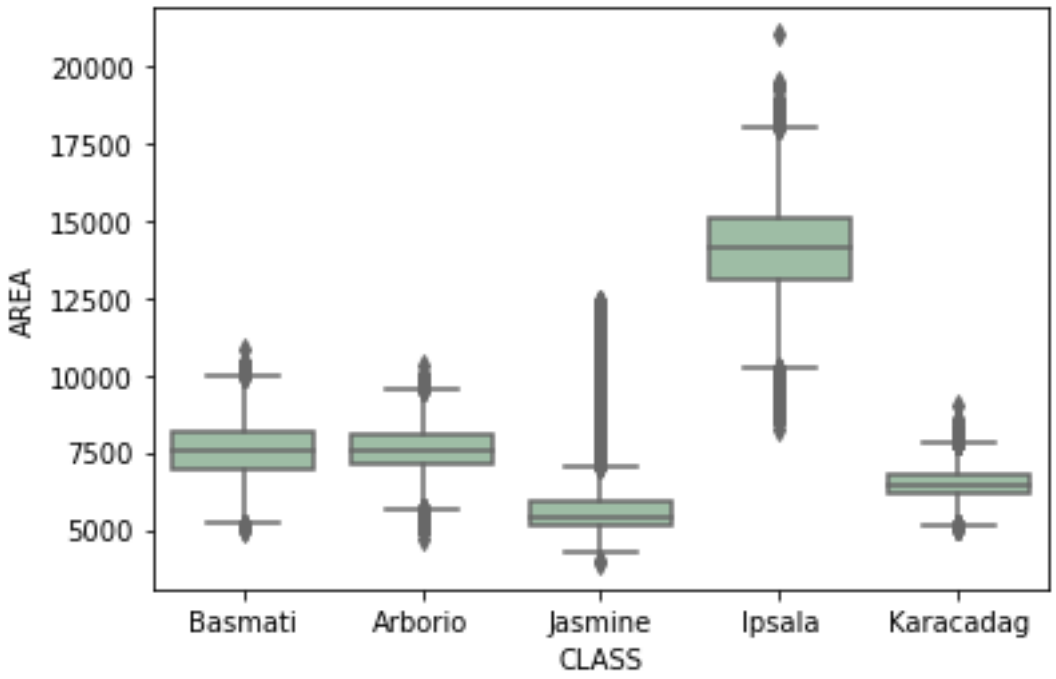
#fragment

Index	AREA	PERIMETER	MAJOR_AXIS	MINOR_AXIS	ECCENTRICITY	EQDIASQ	SOLIDITY	CONVEX_ARE	EXTENT
AREA	Mocno	0.841377	0.626212	0.781638	0.023491	Mocno	0.0704175	Mocno	0.0994519
PERIMETER	0.841377	Mocno	Mocno	0.340601	0.486245	0.85062	-0.222881	0.846707	-0.253211
MAJOR_AXIS	0.626212	Mocno	Mocno	0.0174657	0.717269	0.637032	-0.335401	0.633218	-0.422099
MINOR_AXIS	0.781638	0.340601	0.0174657	Mocno	-0.581973	0.778685	0.359023	0.775987	0.468006
ECCENTRICITY	0.023491	0.486245	0.717269	-0.581973	Mocno	0.0234624	-0.508364	0.0325081	-0.591706
EQDIASQ	Mocno	0.85062	0.637032	0.778685	0.0234624	Mocno	0.0692171	Mocno	0.0976643
SOLIDITY	0.0704175	-0.222881	-0.335401	0.359023	-0.508364	0.0692171	Mocno	0.0496643	0.383106
CONVEX_AREA	Mocno	0.846707	0.633218	0.775987	0.0325081	Mocno	0.0496643	Mocno	0.0923332
EXTENT	0.0994519	-0.253211	-0.422099	0.468006	-0.591706	0.0976643	0.383106	0.0923332	Mocno
ASPECT_RATIO	-0.161773	0.38521	0.660746	-0.726882	0.886485	-0.152921	-0.487489	-0.15292	-0.638612

Rysujemy boxploty dotyczące AREA dla różnych rodzajów ryżu

data_powierzchnia = data2[['AREA','CLASS']]

box_area = sns.boxplot(x='CLASS', y='AREA', data=data_powierzchnia, color='#99c2a2')



Ipsala = data_powierzchnia.loc[data_powierzchnia['CLASS'] == 'Ipsala']

Ipsala = Ipsala.reset_index()

Ipsala[(Ipsala['AREA'] > 12500) & (Ipsala['AREA'] < 15000)].shape[0] # ponad połowa wartosci w srodku boxplota: 8827

Ipsala[Ipsala['AREA'] > 17000].shape[0] # 260

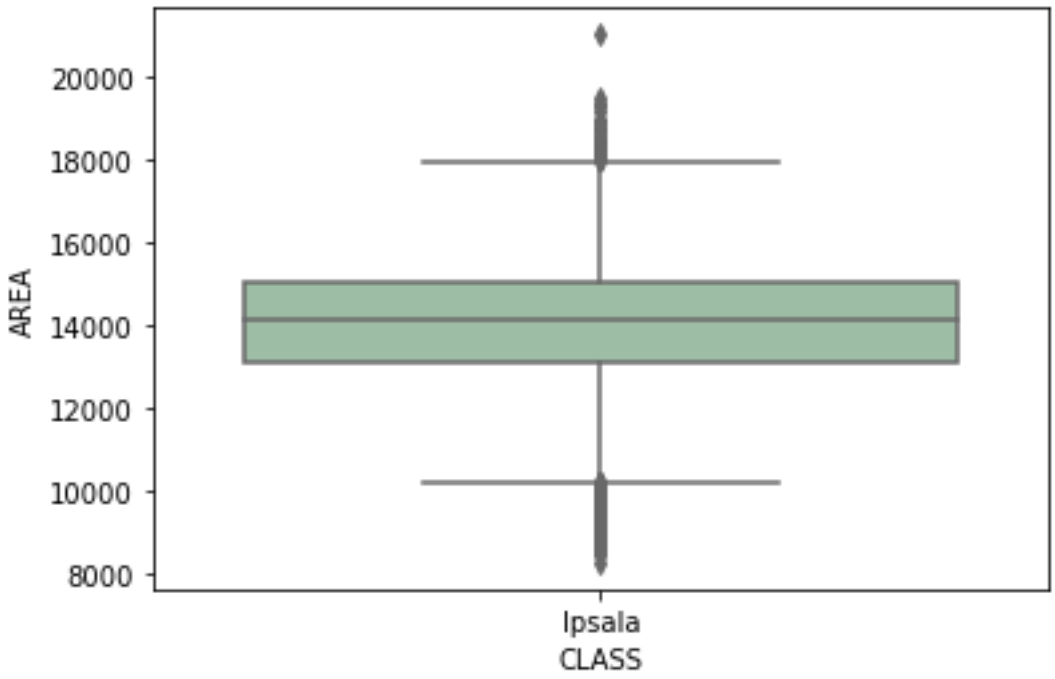
Ipsala[Ipsala['AREA'] > 20000].shape[0] # 1 mocny outlier

Ipsala[Ipsala['AREA'] < 10000].shape[0] # 111

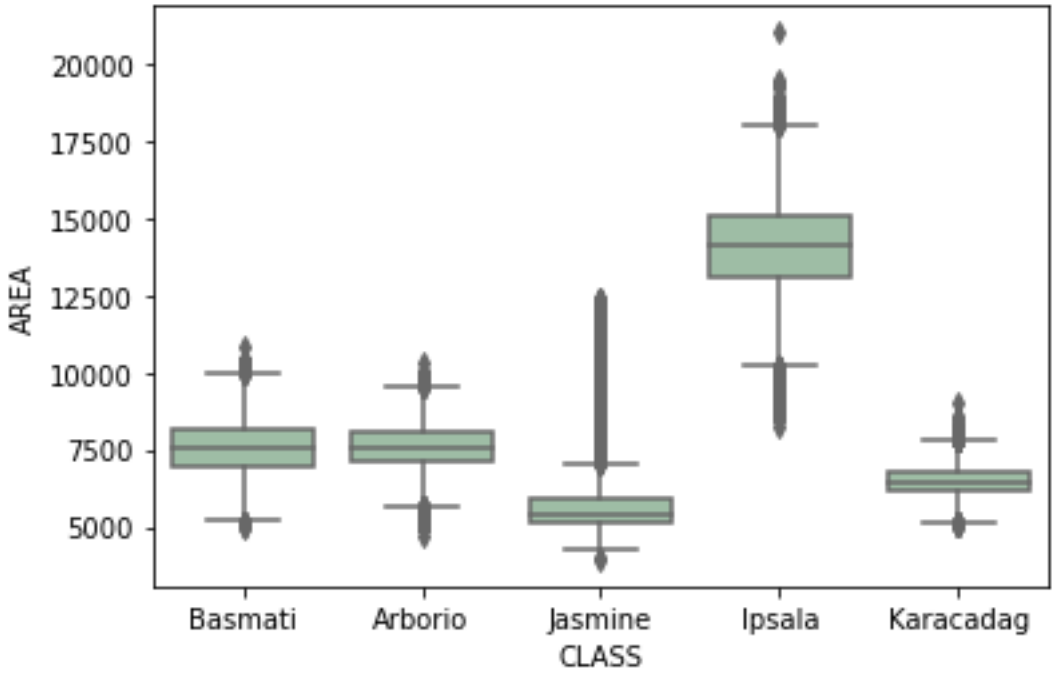
Ipsala['AREA'].mean() #14048.670868347339

Ipsala['AREA'].median() #14134.0

ax2 = sns.boxplot(x='CLASS', y='AREA', data=Ipsala, color='#99c2a2')



ax = sns.boxplot(x='CLASS', y='AREA', data=data_powierzchnia, color='#99c2a2')



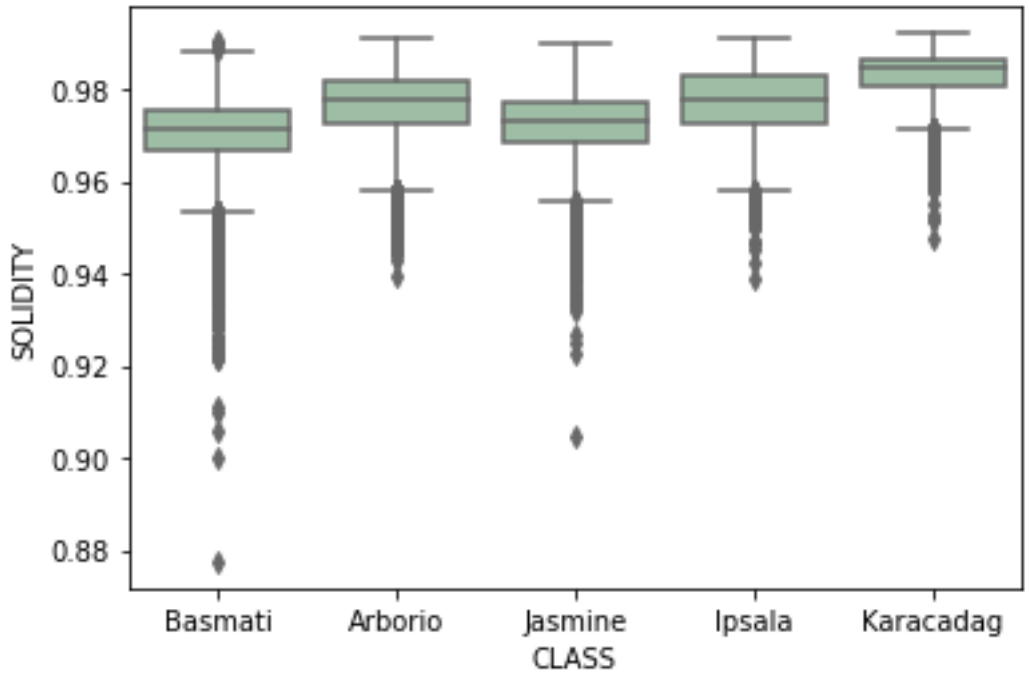
Boxplot stworzony dla wszystkich typów ryżu wskazuje na to, że są istotne różnice w powierzchni jeżeli chodzi o Ipsala, Karacadag, Jasmine,

natomiast zbadanie powierzchni nie odróżni nam od siebie Basmati i Arborio.

Weźmiemy jeszcze trwałość, żeby może odróżnić dwa pozostałe rodzaje ryżu

data_trwalosc = data2[['SOLIDITY','CLASS']]

box_trwalosc = sns.boxplot(x='CLASS', y='SOLIDITY', data=data_trwalosc, color='#99c2a2')



```
# Powinno nam odróżnić basmati od arborio

# KNN

X,Y = data[['AREA','SOLIDITY']], data['CLASS']

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.1)


knn = KNeighborsClassifier()

knn.fit(X_train, Y_train)

przewidziane = knn.predict(X_test)


print(classification_report(Y_test, przewidziane))
```

```
              precision    recall  f1-score   support

     Arborio       0.48      0.51      0.50      1516
     Basmati       0.52      0.50      0.51      1484
      Ipsala       0.99      0.98      0.98      1504
     Jasmine       0.82      0.85      0.83      1479
   Karacadag       0.73      0.70      0.72      1517

    accuracy                           0.71      7500
   macro avg       0.71      0.71      0.71      7500
weighted avg       0.71      0.71      0.71      7500
```