Laboratorium 7 – wskazówki

https://python-course.eu/machine-learning/k-nearest-neighbor-classifier-with-sklearn.php

https://www.kdnuggets.com/2022/07/knearest-neighbors-scikitlearn.html

https://towardsdatascience.com/knn-using-scikit-learn-c6bed765be75

https://scikit-learn.org/stable/modules/neighbors.html

https://www.geeksforgeeks.org/ml-implementation-of-knn-classifier-using-sklearn/

https://www.datacamp.com/tutorial/k-nearest-neighbor-classification-scikit-learn

https://realpython.com/knn-python/

```
from sklearn import datasets
import matplotlib.pyplot as plt
import numpy as np
from statistics import mode
from sklearn import datasets
from sklearn.decomposition import PCA
from sklearn.neighbors import KDTree
import time
import pandas as pd
```

Zad.3.1

```
X, y = datasets.make_classification(
        n_samples=100,
        n_features=2,
        n_informative=2,
        n_redundant=0,
        n_repeated=0,
        random_state=3
)
```

Opis:

https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_classification.html

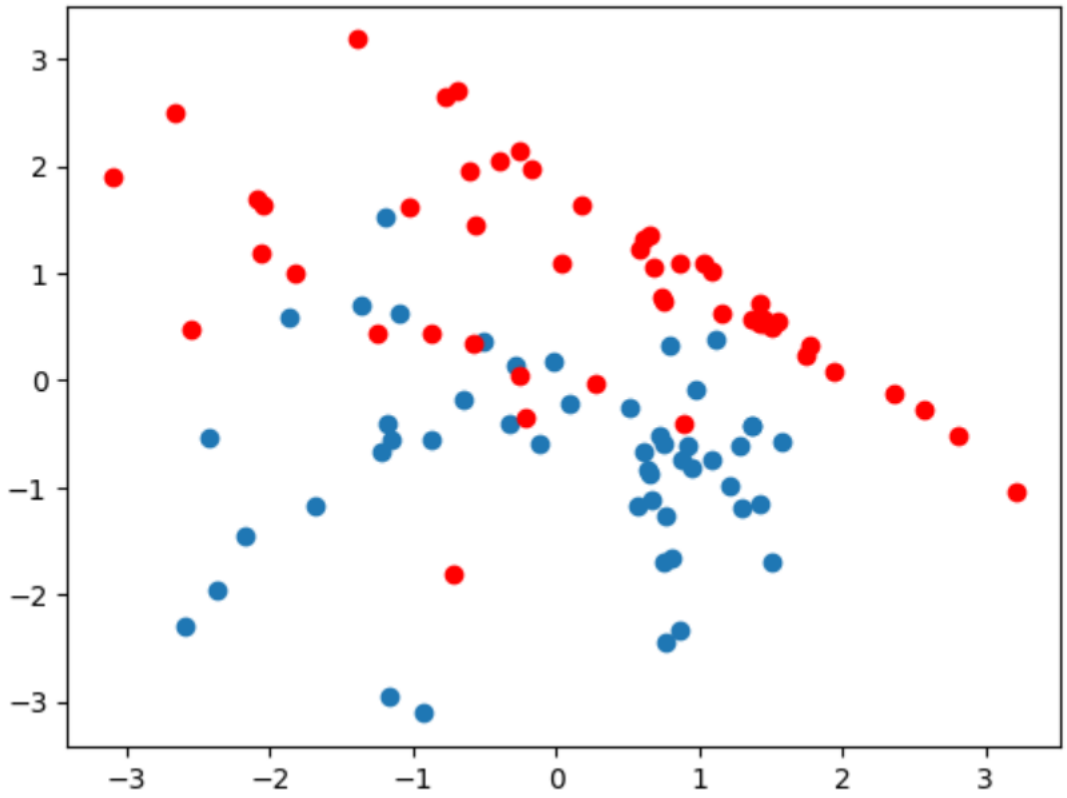
Wizualizacja danych wejściowych

https://scikit-learn.org/stable/auto_examples/neighbors/plot_classification.html#sphx-glr-auto-examples-neighbors-plot-classification-py

```
#wizualizacja zbioru uczacego
plt.plot(X[:,0][y==0],X[:,1][y==0],'o')
plt.plot(X[:,0][y==1],X[:,1][y==1],'ro')
plt.show()

#stworzenie zbioru testowego
X_value_samples = 21 #liczba punktow
X_value1 = np.array([[np.random.uniform(np.min(X[:,0])*0.8 ,np.max(X[:,0])*0.8 )] for x in range(X_value_samples)])#maksymalne x-wspolrzedne
X_value2 = np.array([[np.random.uniform(np.min(X[:,1])*0.8 ,np.max(X[:,1])*0.8 )] for x in range(X_value_samples)])#maksymalne y-wspolrzedne
X_value = np.hstack((X_value1,X_value2))

#odpowiadajace wartosci dla zbioru testoweg
y_value = np.random.randint(0,2,X_value_samples)
```

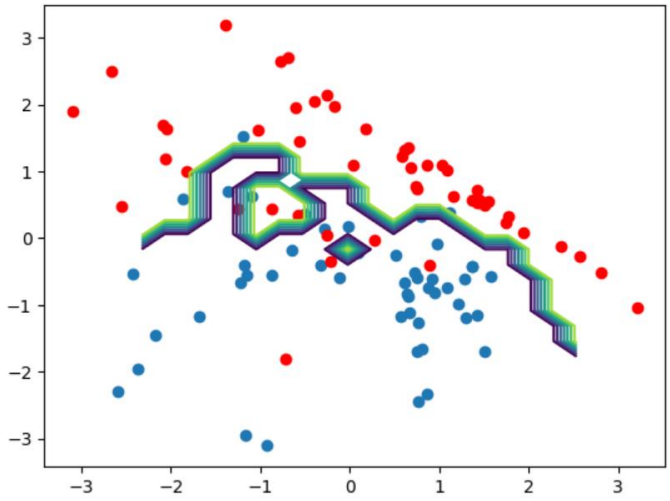


Zad.3.2

Implementacja + użycie funkcji fit, predict, score, porównaj z:

https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

Zad.3.3

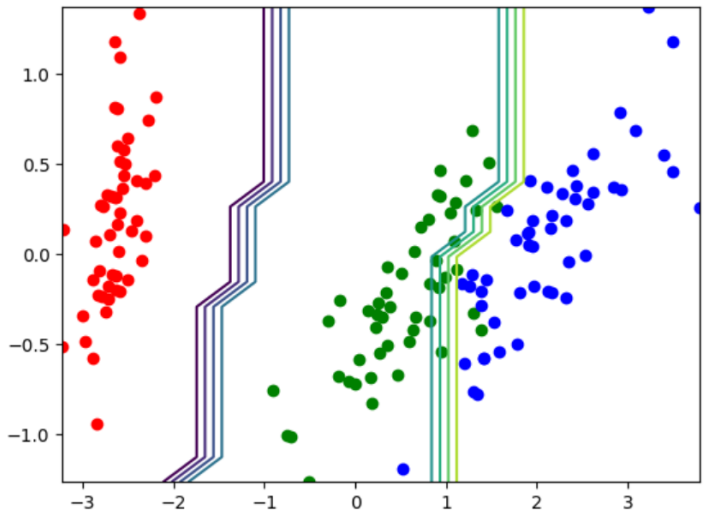


Zad.3.4 - datasets.load_iris(), klasyfikacja j.w. (fit, predict, score)

https://scikit-learn.org/stable/auto_examples/datasets/plot_iris_dataset.html#sphx-glr-auto-examples-datasets-plot-iris-dataset-py

https://scikit-learn.org/stable/auto_examples/neighbors/plot_classification.html#sphx-glr-auto-examples-neighbors-plot-classification-py

Zad. 3.5 – użyć gotowego PCA, np. PCA(n_components = 2), pca.fit(), plt.contour()



Zad.4.1

https://scikit-learn.org/stable/auto_examples/neighbors/plot_regression.html#sphx-glr-auto-examples-neighbors-plot-regression-py

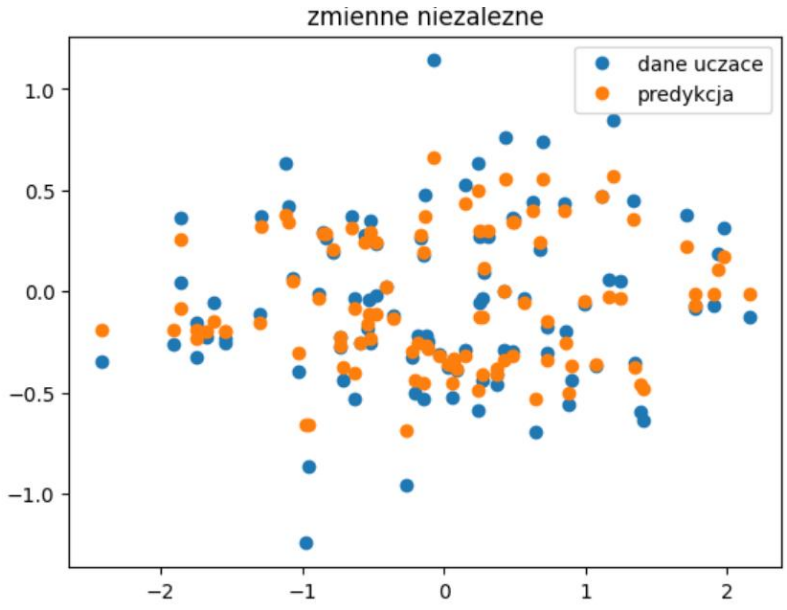
```
X, y = datasets.make_regression(
        n_samples=100,
        n_features=2,
        n_informative=1,
        #n_redundant=0,
        #n_repeated=0,
        noise=0,
        random_state=3
)
```

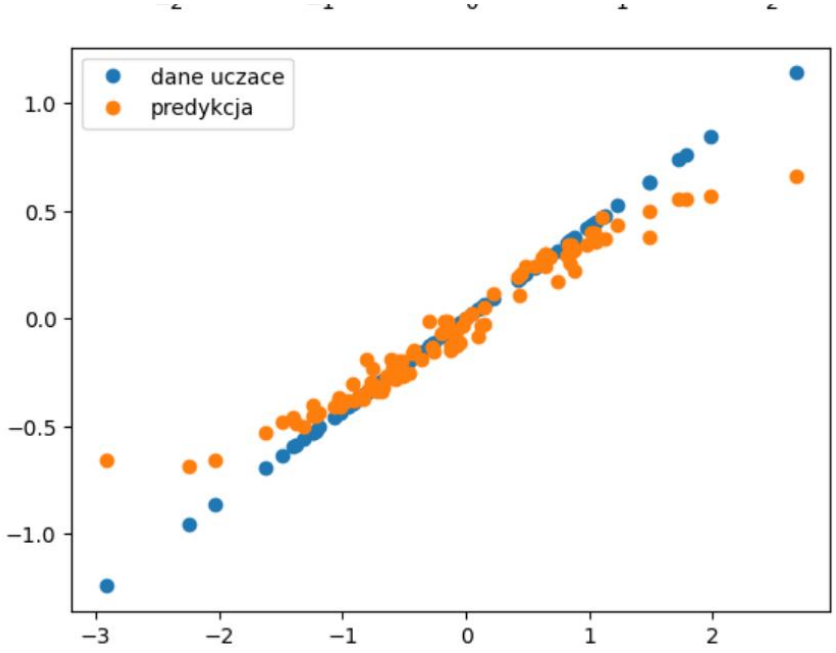
Do zdefiniowanej wcześniej klasy dodać metodę predict_regression

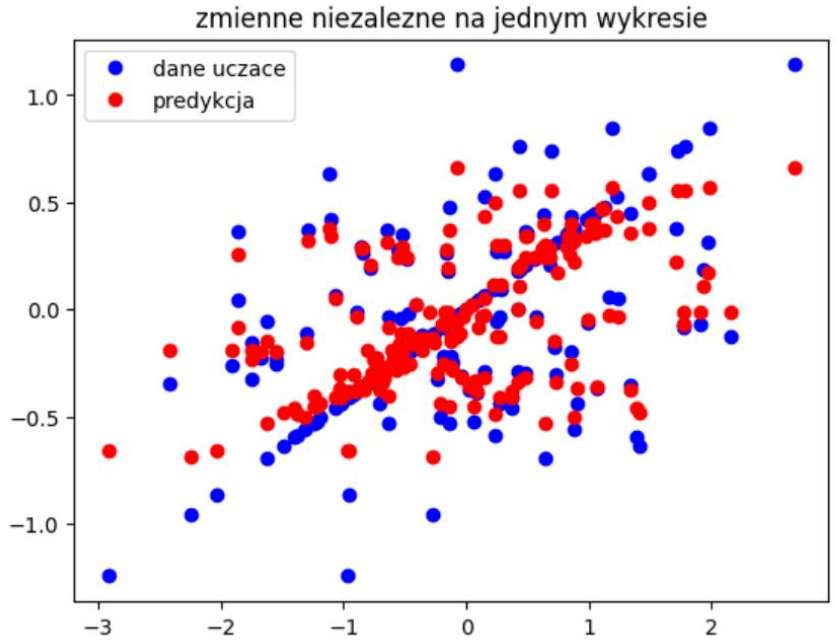
```
def predict_regression(self, X):
```

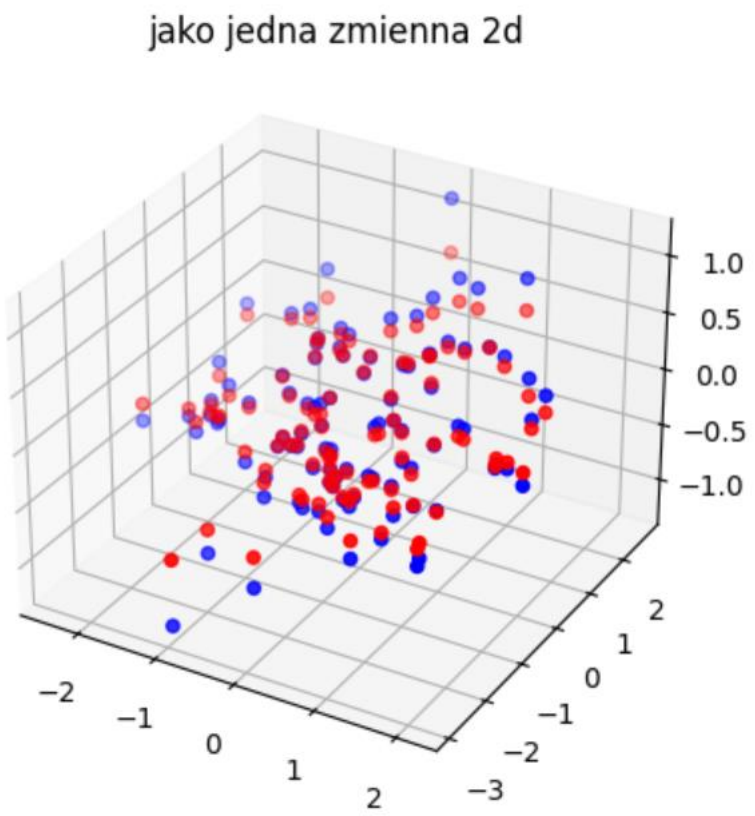
zwracającą decyzje.

Zad.4.3




dane uczace
predykcja
1.0
0.5
0.0
−0.5
−1.0
−3
−2
−1
0
1
2


zmienne niezalezne na jednym wykresie
dane uczace
predykcja
1.0
0.5
0.0
−0.5
−1.0
−3
−2
−1
0
1
2




```
blad sredniokwadratowy= 0.013197843065786525 %
```

Zad.4.4

```
from sklearn.datasets import fetch_california_housing
import pandas
```

Zad. 4.5

```
from sklearn.model_selection import train_test_split
```

5 sąsiadów;

```
iteracja= 0
  średni błąd dopasowania: 1.959
iteracja= 1
  średni błąd dopasowania: 2.032
iteracja= 2
  średni błąd dopasowania: 2.055
iteracja= 3
  średni błąd dopasowania: 2.028
iteracja= 4
  średni błąd dopasowania: 2.046
```