Systemy Sztucznej Inteligencji

dokumentacja projektu *Przewidywanie cukrzycy*

Kamil Giziński, Bartosz Rolnik, Dominik Sigulski
Informatyka, semestr IV
Wydział Matematyki Stosowanej
Politechnika Śląska

10 czerwca 2022

Spis treści

1 Część I

1.1 Opis programu

Klasyfikator miękki dla przewidywania cukrzycy u pacjentów.
W projekcie należy zaproponować przynajmniej trzy różne sposoby na tworzenie tabeli zbioru miękkiego na podstawie zbioru treningowego oraz porównać otrzymane wyniki.

Poprawka 1:
Po konsultacji z prowadzącym laboratoria zdecydowaliśmy się na stworzenie dwóch klasyfikatorów zbiorów miękkich i jednego klasyfikatora zbiorów rozmytych.

1.2 Instrukcja obsługi

Podstawą projektu jest baza danych zawierająca dane dotyczące stanu zdrowia pacjentów. W szczególności wszyscy pacjenci w tym zbiorze danych to kobiety w wieku co najmniej 21 lat pochodzące z indiańskiego plemienia Pima. Dane zostały zebrane przez Narodowy Instytut Cukrzycy oraz Chorób Układu Pokarmowego i Nerek (ang. National Institute of Diabetes and Digestive and Kidney Diseases - NIDDK) w Stanach Zjednoczonych. Bazę można pobrać ze strony kaggle.com.

Kolumny zbioru danych to kolejno:

1. **Pregnancies** (ciąże) — liczba ciąż danej pacjentki;
2. **Glucose** (glukoza) — stężenie glukozy w osoczu w doustnym teście tolerancji glukozy;
3. **BloodPressure** (ciśnienie krwi) — rozkurczowe ciśnienie krwi (mm Hg);
4. **SkinThickness** (grubość skóry) — grubość fałdu skórnego na tricepsie (mm);
5. **Insulin** (insulina) — insulina w surowicy 2-godzinnej (mj./ml);
6. **BMI** — wskaźnik masy ciała (waga w kg / (wzrost w m)^2);
7. **DiabetesPedigreeFunction** (funkcja metryki cukrzycy) — funkcja oceniająca prawdopodobieństwo wystąpienia cukrzycy na podstawie wywiadu rodzinnego;
8. **Age** (wiek) — wiek pacjentki w latach;
9. **Outcome** (wynik) — zmienna klasy (0 lub 1). 268 z 768 to 1, pozostałe to 0;

Program uruchamiamy za pomocą środowiska Jupyter. Po jego uruchomieniu program zacznie wyliczać dokładność wszystkich klasyfikatorów, które poznaliśmy w ciągu tego semestru, oraz tych, które sami zaprogramowaliśmy. Wszystkie wyżej wymienione algorytmy służą do przewidywania na podstawie zebranych danych, czy pacjent choruje na cukrzycę, czy jest zdrowy. Na samym początku wyliczana jest dokładność klasyfikatorów: Bayesa oraz KNN. Następnie program wyliczy dokładność dwóch zaprojektowanych przez nas klasyfikatorów miękkich. W osobnym pliku znajduje się Klasyfikator wnioskujący ze zbioru rozmytego,

którego dokładność jest wyświetlana przy pomocy macierzy pomyłek. To oznacza, że otrzymujemy dokładne dane, jaki procent pacjentów został zdiagnozowanych poprawnie jako chorzy, fałszywie jako chorzy, poprawnie jako zdrowi oraz fałszywie jako zdrowi.

1.3 Analiza bazy danych

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               768 non-null    int64
 1   Glucose                   768 non-null    int64
 2   BloodPressure             768 non-null    int64
 3   SkinThickness             768 non-null    int64
 4   Insulin                   768 non-null    int64
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64
 8   Outcome                   768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

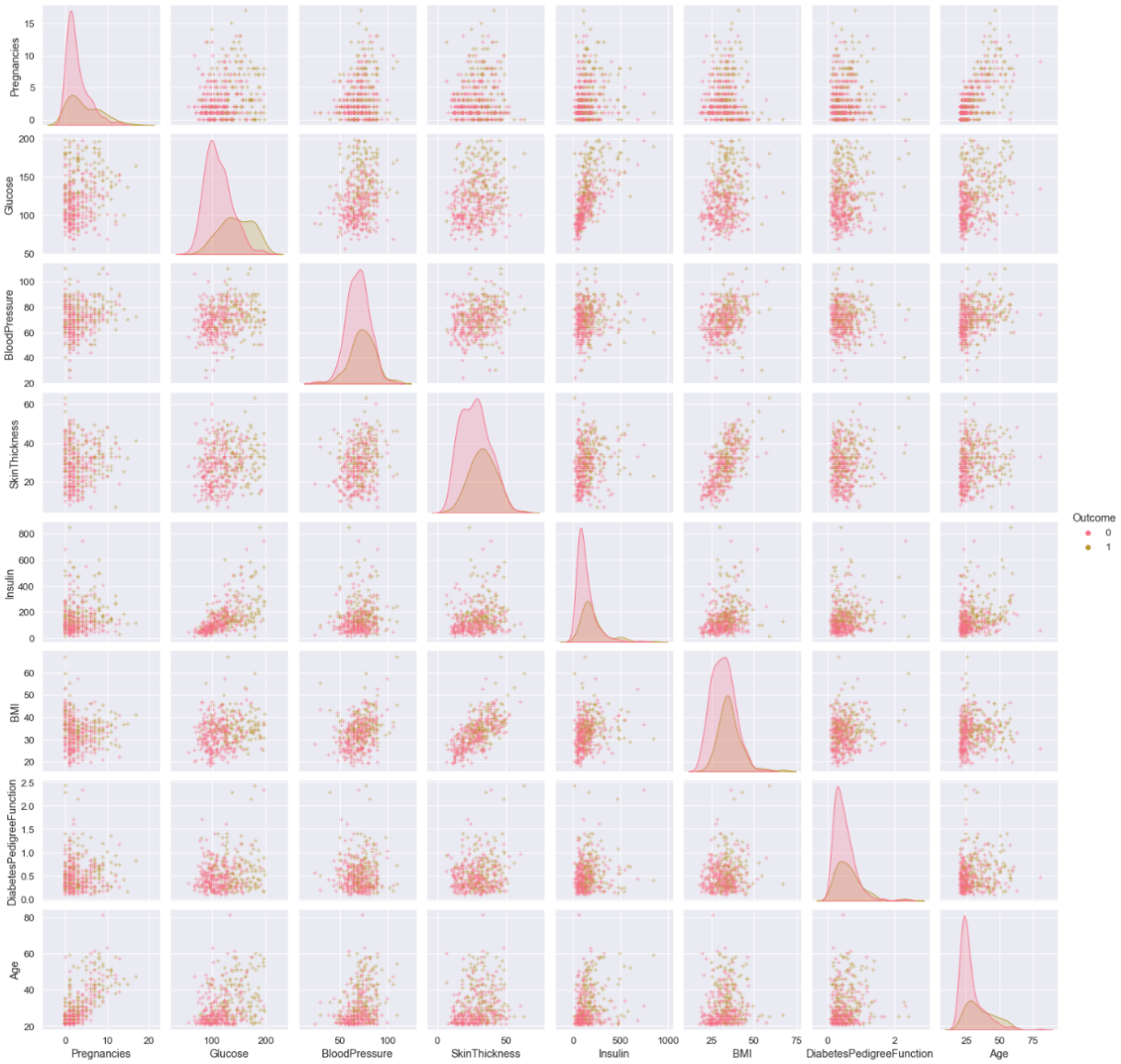
Rysunek 1: Wyświetlenie podstawowych informacji o bazie

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

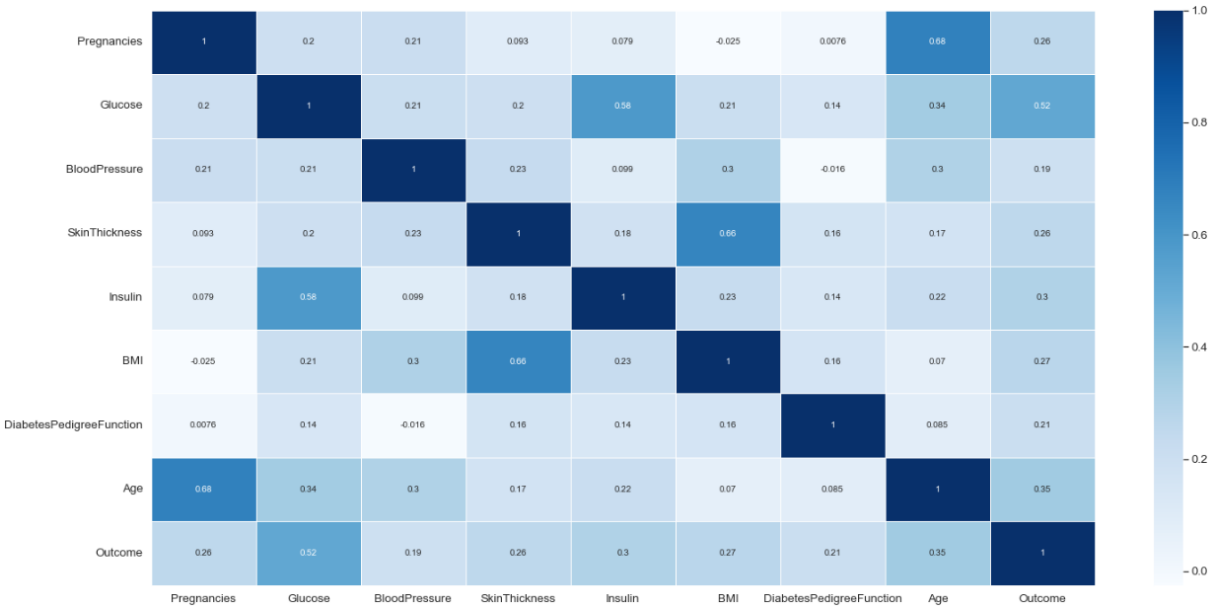
Rysunek 2: Wyświetlenie podstawowej statystyki bazy

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	392.000000	392.000000	392.000000	392.000000	392.000000	392.000000	392.000000	392.000000	392.000000
mean	3.301020	122.627551	70.663265	29.145408	156.056122	33.086224	0.523046	30.864796	0.331633
std	3.211424	30.860781	12.496092	10.516424	118.841690	7.027659	0.345488	10.200777	0.471401
min	0.000000	56.000000	24.000000	7.000000	14.000000	18.200000	0.085000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	21.000000	76.750000	28.400000	0.269750	23.000000	0.000000
50%	2.000000	119.000000	70.000000	29.000000	125.500000	33.200000	0.449500	27.000000	0.000000
75%	5.000000	143.000000	78.000000	37.000000	190.000000	37.100000	0.687000	36.000000	1.000000
max	17.000000	198.000000	110.000000	63.000000	846.000000	67.100000	2.420000	81.000000	1.000000

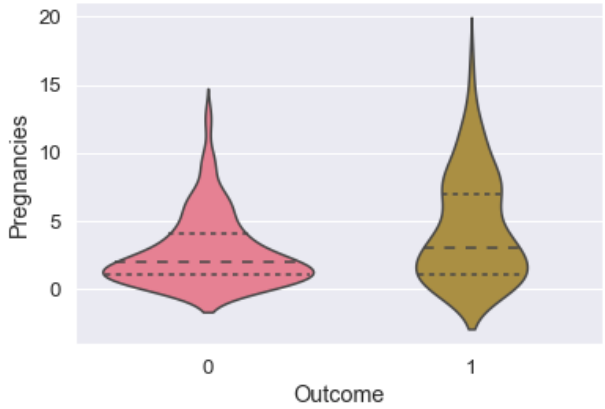
Rysunek 3: Wyświetlenie podstawowej statystyki bazy z wyeliminowanymi niektórymi wartościami równymi 0



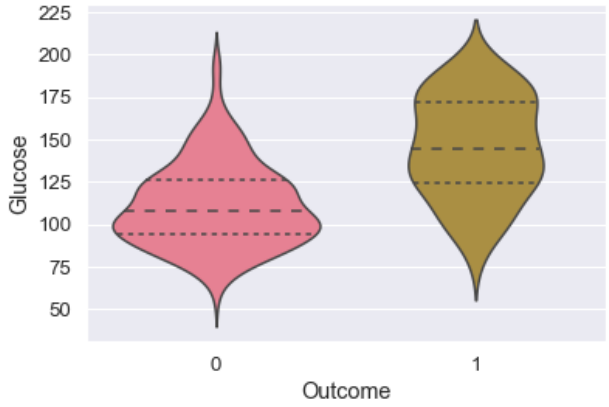
Rysunek 4: Wykres rozkładu Wyników dla par cech



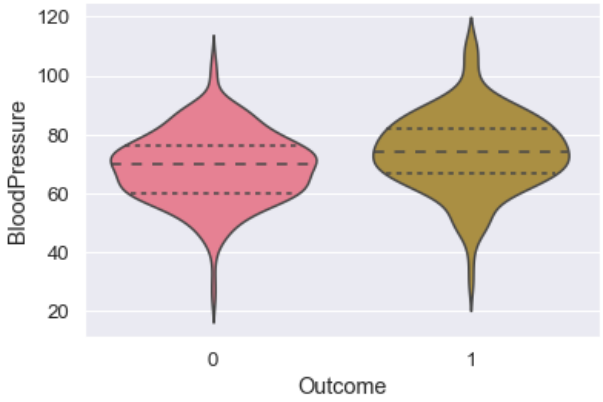
Rysunek 5: Wykres korelacji wartości



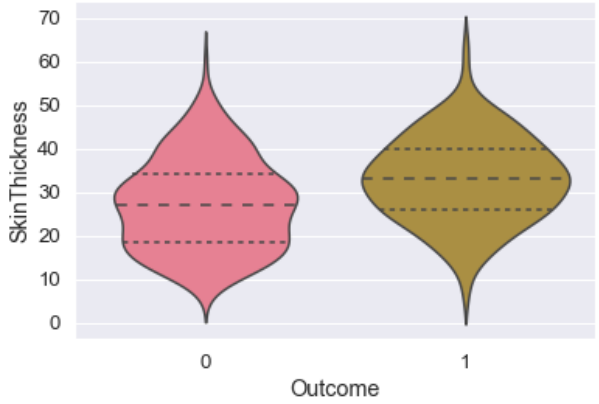
Rysunek 6: Rozkład Ciąże - Wynik



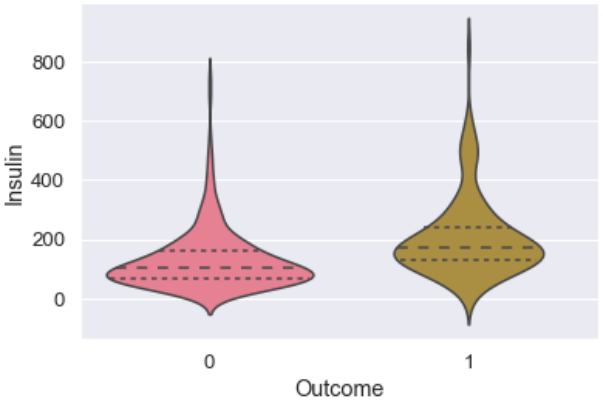
Rysunek 7: Rozkład Glukoza - Wynik



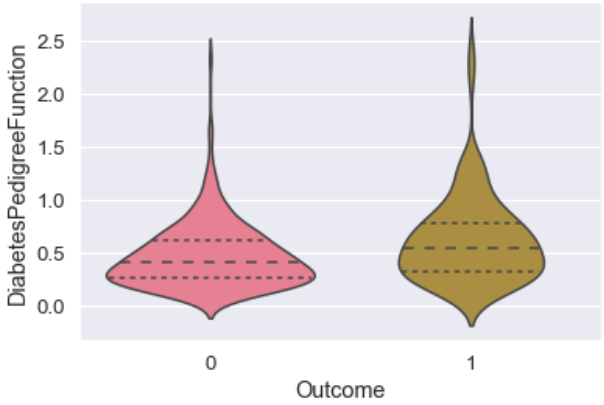
Rysunek 8: Rozkład CiśnienieKrwi - Wynik



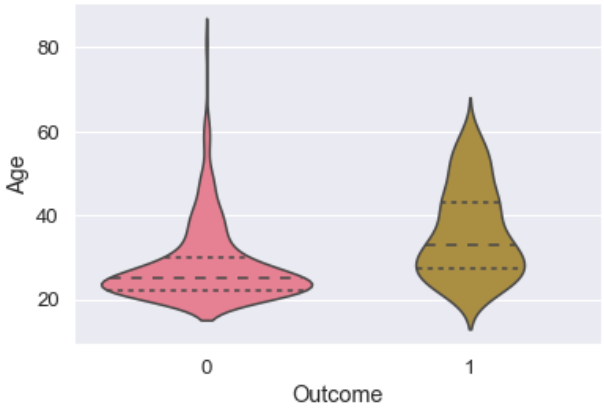
Rysunek 9: Rozkład GrubośćSkóry - Wynik



Rysunek 10: Rozkład Insulina - Wynik



Rysunek 11: Rozkład Funkcja metryki cukrzycy - Wynik



Rysunek 12: Rozkład Wiek - Wynik

1.4 Wprowadzenie od strony medycznej

W tej części zajmiemy się przybliżeniem informacji o kolumnach w zbiorze danych, z którego korzystamy.

Glukoza to cukier prosty, który jest podstawowym źródłem energii w organizmie człowieka. Oznaczenie jej stężenia we krwi (czyli glikemii) na czczo pomaga określić, czy pacjent ma cukrzycę.
Orientacyjna norma stężenia glukozy we krwi wynosi:

- **70-99 mg/dl** — wartości prawidłowe
- **100-125 mg/dl** — wartości nieprawidłowe, wymagają wykonania doustnego testu tolerancji glukozy
- **ponad 126 mg/dl** — wartości nieprawidłowe, wymagają powtórzenia badania, po dwukrotnym otrzymaniu takiego wyniku rozpoznaje się cukrzycę

Kolejna kolumna to **ciśnienie krwi**. Ciśnienie tętnicze to siła nacisku, jaką płynąca krew wywiera na ściany tętnic. Pomiaru dokonuje się za pomocą ciśnieniomierza, a otrzymane wartości przedstawia się w mm Hg, czyli milimetrach słupa rtęci.
Prawidłowe ciśnienie krwi jest niższe niż 90 mm Hg.

Ciśnienie tętnicze w zakresie normy dzieli się na:

- **normalne**: <80 mm Hg;
- **podwyższone faza 1**: 80–89 mm Hg;
- **podwyższone faza 2**: >90 mm Hg.

Grubość fałdu skórnego na tricepsie jest jednym z wyznaczników wartości tkanki tłuszczowej organizmu. Według danych Amerykańskiej agencji CDC (Basic data on anthropometric measurements and angular measurements of the hip and knee joints for selected age groups 1-74 years of age) na podstawie badań w latach 1971-1975, mediana grubości fałdu skórnego na tricepsie dla kobiet nie przekracza **25 mm**. Jednakowoż nie udało nam się znaleźć informacji na temat konkretnych norm (i czy takie normy dla całej populacji w ogóle istnieją).

Insulina to hormon, który obniża poziom cukru we krwi. Jej norma 2 godziny po posiłku to **do 30 mIU/ml** (za: www.doz.pl). W zbiorze danych, z którego korzystamy, wartości zdecydowanie wykraczają poza normę - mediana wynosiła 125.5 muU/ml, co może sugerować użycie innych jednostek, a nie jedynie literówkę w ich nazwie. Nie udało się nam znaleźć normy dla jednostek 'muU/ml'. Podwyższone stężenie insuliny zazwyczaj jest objawem cukrzycy typu 2, natomiast wynik poniżej normy wskazuje na cukrzycę typu 1. Wyniki, które przekraczają normę lub znajdują się poniżej przyjętych wartości, wymagają konsultacji u diabetologa.

Funkcja metryki cukrzycy
Niestety nie udało nam się dotrzeć do szczegółów dotyczących wzoru Funkcji metryki cukrzycy. Jak podaje strona kaggle.com, jest to "funkcja oceniająca prawdopodobieństwo wystąpienia cukrzycy na podstawie wywiadu rodzinnego". Możemy się jedynie domyślać, że im wyższa wartość, tym większe ryzyko.

BMI to wskaźnik masy ciała. Jest wskaźnikiem, który jest obliczany przez porównanie wzrostu z masy ciała. Jego wartość jest pomocna w ocenie ryzyka wystąpienia chorób związanych z nadwagą takich jak miażdżyca, cukrzyca lub choroba niedokrwienna serca. Im mniejsza wartość BMI, tym ryzyko wystąpienia chorób jest mniejsze.

Normy BMI za CDC:

- **niedowaga**: <18.5;

- **waga prawidłowa**: 18.5 - 24.9;
- **nadwaga**: 25.0 - 29.9;
- **otyłość**: >30;

Wiek
Jak podaje amerykańska agencja CDC, pacjent jest w grupie ryzyka, jeśli jego wiek wynosi co najmniej **45 lat**.

1.5 Dodatkowe informacje

Wymagania:

- Zaproponowanie dwóch klasyfikatorów zbiorów miękkich i jednego klasyfikatora zbioru rozmytego;
- Porównanie wyników wszystkich klasyfikatorów.

2 Część II

2.1 Opis działania

- **ProcessingData**:
  Jest to klasa, która służy do prawidłowego sformatowanie bazy danych. Posiada ona metodę do tasowania zbioru, do normalizacji próbek oraz finalnie do podzielenia bazy na zbiór treningowy i walidacyjny (w naszym programie wszystkie zbiory będą dzielone w proporcjach 70:30).

- **Naiwny Klasyfikator Bayesa**:
  Naiwny klasyfikator Bayesowski, bazujący na twierdzeniu Bayesa, nadaje się szczególnie do problemów o bardzo wielu wymiarach na wejściu. W wyniku działania algorytmu otrzymujemy listę prawdopodobieństw przynależności obiektu do danych klasy. Ostatecznym rezultatem klasyfikacji jest wartość zwrócona przez metodę maksimum prawdopodobieństwa spośród wyników dla poszczególnych klas.

- **Klasyfikator KNN**:
  Przy użyciu tego algorytmu przydzielamy dany obiekt do klasy, gdzie należy największa liczba jego sąsiadów, gdzie liczba sąsiadów jest z góry ustaloną liczbą 'k'. W przypadku naszego programu do liczenia odległości między obiektami wykorzystaliśmy metrykę euklidesową.

- **Zbiory miękkie**:
  Zbiory miękkie są w rzeczywistości uogólnieniem zbiorów rozmytych (pewnym ich uproszczeniem), zostały stworzone, żeby poradzić sobie z niepewnością w sposób parametryczny. Zbiór miękki to sparametryzowana rodzina zbiorów – intuicyjnie jest to „miękkie", ponieważ granica zbioru zależy od parametrów.

- **Zbiory rozmyte**:
  W matematyce zbiory rozmyte (tzw. zbiory niepewne) to zbiory, których elementy mają stopnie przynależności i jest to rozszerzenie klasycznego pojęcia zbioru. W klasycznej teorii mnogości przynależność elementów do zbioru ocenia się zero-jedynkowo — element albo należy, albo nie należy do zbioru. Z kolei teoria zbiorów rozmytych pozwala na stopniową ocenę przynależności elementów do zbioru. Oceny przynależności elementu dokonuje się przy pomocy funkcji przynależności o wartości w rzeczywistym przedziale jednostkowym [0, 1].

- **Wzory:**

  (1) ProcessingData - normalizacja std

  $$\frac{x - \bar{x}}{\sigma}$$

  * σ - odchylenie standardowe
  * $\bar{x}$ - średnia

(2) ProcessingData - normalizacja minmax

$$\frac{x - min}{max - min}$$

* min - wartość minimalna
* max - wartość maksymalna

(3) NaiveBayes - twierdzenie Bayesa

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$

* A, B - zdarzenia
* $P(A|B)$ - oznacza prawdopodobieństwo warunkowe, tj. prawdopodobieństwo zajścia zdarzenia A, o ile zajdzie zdarzenie B
* $P(B|A)$ - oznacza prawdopodobieństwo zajścia zdarzenia B, o ile zajdzie zdarzenie A

(4) NaiveBayes - gęstość (Gauss)

$$\frac{1}{\sigma\sqrt{2\pi}} * e^{-\frac{(x-m)^2}{2\sigma^2}}$$

* σ - odchylenie standardowe
* x - wynik obserwacji
* m - wartość oczekiwana zmiennej

(5) KNN - Euklidesowa odległość między wektorami

$$\sqrt{\sum_{i=1}^{n} |x_i - y_i|^2}$$

* x_i - współrzędna wektora x
* y_i - współrzędna wektora y

(6) Rozmyte - funkcja typu trójkąt

$$\mu_{triangular}(x; a, b, c) = \begin{cases} 0, & \text{if } x \leqslant a. \\ \frac{x-a}{b-a}, & \text{if } a < x \leqslant b. \\ \frac{c-x}{c-b}, & \text{if } b < x \leqslant c. \\ 0, & \text{if } c < x. \end{cases}$$

* a - początek podstawy trójkąta, członkostwo przyjmuje wartość 0

* b - wysokość trójkąta, członkostwo przyjmuje wartość 1
* c - długość podstawy trójkąta, członkostwo przyjmuje wartość 0

(7) Rozmyte - funkcja typu trapez

$$\mu_{trapezoidal}(x; a, b, c, d) = \begin{cases} 0, & \text{if } x \leqslant a. \\ \frac{x-a}{b-a}, & \text{if } a < x \leqslant b. \\ 1, & \text{if } b < x \leqslant c. \\ \frac{d-x}{d-c}, & \text{if } c < x \leqslant d. \\ 0, & \text{if } d < x. \end{cases}$$

* a - początek podstawy trapezu, członkostwo przyjmuje wartość 0
* zakres od b do c reprezentuje najwyższą wartość członkostwa równą 1
* d - długość podstawy dolnej trapezu, członkostwo przyjmuje wartość 0

(8) Stopień spełnienia przesłanki

$$\mu_i = max\{ssp\}$$

* ssp - stopnie spełnienia etykiet przesłanki (np.: ssp=(0,6, 0,75))

(9) Stopień spełnienia reguły dla wszystkich przesłanek:

$$\mu_{cale}(x) = min\{\mu_1, \mu_2, ..., \mu_i\}$$

* μ_i - stopień spełnienia danej przesłanki

(10) Środek ciężkości trójkąta

$$S = \frac{a + b + c}{3}$$

* a, b, c - współrzędne wierzchołków w trójkącie

(11) Środek ciężkości figury

$$h = \frac{\sum_i \mu_i A_i c_i}{\sum_i \mu_i A_i}$$

* A_i - powierzchnia zbioru i
* μ_i - stopień przynależności zbioru i
* c_i - środek ciężkości zbioru i

Przykłady zasad dla zbiorów rozmytych:

```
        # Same niskie wartosc lub niskie | srednie sprawiaja ze nie
            przewidujemy cukrzycy
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})

         # Wartosci oscylujace miedzy niskimi i srednimi z wyjatkiem
             grubosci skory i ilosci ciaz czy wieku wciaz sa przewidywane
              jako brak cukrzycy
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'low'})

        # sporo srednich, srednich niskich lub wysokich czynnikow
            sklaniaja nas do przewidzenia cukrzycy
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
```

2.2 Implementacja

2.2.1 Klasyfikator miękki nr 1 - Klasa SoftSetClassifierMean

Klasa SoftSetClassifierMean zawiera klasyfikator miękki korzystający ze średniej arytmetycznej oraz metody pomocnicze.

```
# lista etykiet
labels: list[str] = []
# lista par 0, 1 dla kazdej kolumny i etykiety
pairs: list[list[list[int, int]]] = []
# lista srednich dla kazdej kolumny i etykiety
means: list[list[float]] = []
```

Opis list pomocniczych *labels*, *pairs*, *means* przechowujących kolejno etykiety, pary 0, 1 w odpowiedniej kolejności dla każdej kolumny i wiersza, średnie wartości każdej kolumny i etykiety.

```python
@staticmethod
def mean(column: list[float]) -> float:
    """
    Receives a column of the dataframe with numerical values and returns its mean value
    :param column: list[float]
    :return mean value of the values in a dataframe column: float
    """
    # czyli zwykla suma przez ilosc skladnikow
    return sum(column) / len(column)
```

Metoda pomocnicza obliczająca średnie wartości kolumn. Czyli suma wartości dzielona przez ilość wartości.

```python
def calculate(self, df: pd.DataFrame, label: str) -> None:
    """
    Receives a dataframe and the label of the column we want to predict and decides whether pairs are [0,1] or [1,0]
    :param df: pd.DataFrame
    :param label: str
    :return: None
    """
    self.pairs = []
    self.means = []
    # lista unikalnych wartosci z podanej kolumny 'label'
    # ma byc zawsze ta sama kolejnosc wiec sortujemy
    self.labels = sorted(df[label].unique())
    # 'category' to 0 i 1
    for category in self.labels:
        # wartosci ktorych unikalna wartosc kolumny 'label' to 'category'
        # czyli podzial na czesci ktore maja 0 i 1
        category_values = df[df[label] == category].drop(labels=label, axis=1)
        temp_mean = []
        temp_pair = []
        # item to wszystkie wartosci z danej kolumny
        for _, item in category_values.items():
            # srednia z kolumny, dodajemy ja do listy srednich
            mean = self.mean(item)
            temp_mean.append(mean)
            len_lower = 0
            len_upper = 0
            # zliczamy elementy wieksze od sredniej i mniejsze od sredniej
            for value in item:
                if value > mean:
                    len_upper += 1
                else:
                    len_lower += 1

            # jesli jest tyle samo powyzej srendiej co ponizej to
```

```
35              # wybieramy losowo pare [0, 1] lub [1, 0]
36              if len_upper == len_lower:
37                  temp_pair.append(random.choice(([0, 1],[1, 0])))
38              # jesli wiecej powyzej sredniej to zapisujemy pare [1, 0]
39              elif len_upper > len_lower:
40                  temp_pair.append([1, 0])
41              # jesli wiecej ponizej sredniej to zapisujemy pare [0, 1]
42              else:
43                  temp_pair.append([0, 1])
44
45          # uzupelniamy listy wszystkich 'srednich' i wszystkich 'par'
46          self.means.append(temp_mean)
47          self.pairs.append(temp_pair)
```

Metoda tworząca tabelę ważoną zbioru miękkiego. Dzielimy zbiór na część, w której Wynik badania oznacza diabetyka oraz część, w której Wynik badania oznacza osobę zdrową. Dla każdej kolumny w każdej części obliczamy wartość średnią. Zliczamy ilość elementów większych od średniej oraz elementów mniejszych od średniej. Tworzymy tabelę ważoną zbioru z par [0, 1] i [1, 0]. Pary [0, 1] dodajemy do tabeli ważonej w przypadku, gdy jest więcej elementów poniżej średniej. Pary [1, 0] dodajemy do tabeli ważonej w przypadku, gdy jest więcej elementów powyżej średniej. Jeżeli elementów jest tyle samo, to losujemy czy dodać parę [1, 0] czy [0, 1].

```
1  def predict(self, sample: pd.Series, label: str) -> str:
2      """
3      Receives a sample in the form of pd.Series and returns its predicted
           label
4      :param sample: pd.Series
5      :param label: str
6      :return predicted label of the sample: str
7      """
8      # liczenie elementow dla sampla dla kazdej z klas (0, 1)
9
10     sample = sample.drop(labels=label)
11     probabilities = []
12     # bierzemy po kolei odpowiednie pary i srednie
13     for category, mean in zip(self.pairs, self.means):
14         sample_pairs = []
15         # bierzemy odpowiednia srednia i wartosc z sample'a
16         for m, x in zip(mean, sample):
17             # jesli element rowny sredniej
18             # wybieramy losowo pare [0, 1] lub [1, 0]
19             if x == m:
20                 sample_pairs.append(random.choice(([0, 1],[1, 0])))
21             # jesli element wiekszy od sredniej to zapisujemy pare [1,
                   0]
22             elif x > m:
23                 sample_pairs.append([1, 0])
24             # jesli element mniejszy od sredniej to zapisujemy pare [0,
                   1]
25             else:
26                 sample_pairs.append([0, 1])
27         temp = []
```

```
28
29          # liczenie prawdopodobienstwa dla kazdej klasy
30          for x, y in zip(category, sample_pairs):
31              temp.append(x[0] * y[0] + x[1] * y[1])
32          probabilities.append(sum(temp))
33
34      # naszym indeksem jest indeks najwiekszego prawdopodobienstwa
35      idx = max(range(len(probabilities)), key=probabilities.__getitem__)
36      # zwracamy najprawdopodobniejszy wynik z 'labels' (u nas 0 lub 1)
37      return self.labels[idx]
```

Metoda przewidująca etykietę klasy próbki. Znowu tworzymy listę par [0, 1] i [1, 0] jednak teraz bierzemy pod uwagę odpowiednią (wcześniej obliczoną) średnią oraz kolejne elementy z kolejnych kolumn w naszej próbce. Schemat dobierania par jest identyczny do wyżej opisanego podczas opisu metody *calculate()*. Następnie liczymy prawdopodobieństwo wystąpienia każdej etykiety. Opiera się to na zwykłym iloczynie skalarnym wektorów. W tym przypadku naszymi 'wektorami' są indeksy etykiet oraz pary [0, 1] i [1, 0]. Na koniec zwracamy przewidywaną etykietę (czyli tę, która uzyskała największą wartość na liście prawdopodobieństw).

```
1  def test(self, validation_set: pd.DataFrame, label: str) -> None:
2      """
3      Test how accurate the prediction method is
4      :param validation_set: pd.DataFrame - a record from the validation
           dataset
5      :param label: str
6      :return: None
7      """
8      correct = 0
9      # iterrows() jest uzywane do iteracji wieszy dataframe'a
10     for _, row in validation_set.iterrows():
11         # jesli poprawnie przewidzimy etykiete
12         if self.predict(sample=row, label=label) == row[label]:
13             correct += 1
14     # informacja podsumowujaca
15     # print(f'Accuracy: {correct / len(validation_set) * 100:.2f}%; {
           correct}/{len(validation_set)}')
16     return correct / len(validation_set) * 100
```

Metoda testująca dokładność działania klasyfikatora miękkiego. Prosty mechanizm, który w pętli wywołuje metodę *predict()* oraz porównuje wynik z wynikiem poprawnym. Zwrócona zostaje skuteczność przewidywania w procentach.

2.2.2 Klasyfikator miękki nr 2 - Klasa SoftSetClassifierPercentage

Klasa SoftSetClassifierPercentage zawiera klasyfikator miękki korzystający z procentów oraz metody pomocnicze.

```
1 # lista etykiet
2 labels: list[str] = []
3 # lista par 0, 1 dla kazdej kolumny i kazdej etykiety
4 pairs: list[list[list[int, int]]] = []
```

```
5 # lista min'ow i max'ow dla kazdej kolumny i kazdej etykiety
6 minmaxs: list[list[list[float, float]]] = []
```

Opis list pomocniczych *labels*, *pairs*, *minmax* przechowujących kolejno etykiety, pary 0, 1 w odpowiedniej kolejności dla każdej kolumny i wiersza, minima i maksima każdej kolumny i etykiety.

```
1 @staticmethod
2 def mean(column: list[float]) -> float:
3     """
4     Receives a column of the dataframe with numerical values and returns
          its mean value
5     :param column: list[float]
6     :return mean value of the values in a dataframe column: float
7     """
8     # czyli zwykla suma przez ilosc skladnikow
9     return sum(column) / len(column)
```

Metoda pomocnicza obliczająca średnie wartości kolumn. Czyli suma wartości dzielona przez ilość wartości.

```
1 def calculate(self, df: pd.DataFrame, label: str) -> None:
2     """
3     Receives a dataframe and the label of the column we want to predict
         and decides whether pairs are [0,1] or [1,0]
4     :param df: pd.DataFrame
5     :param label: str
6     :return: None
7     """
8     self.pairs = []
9     self.minmaxs = []
10    # lista unikalnych wartosci z podanej kolumny 'label'
11    # ma byc zawsze ta sama kolejnosc wiec sortujemy
12    self.labels = sorted(df[label].unique())
13
14    # 'category' to 0 i 1
15    for category in self.labels:
16        # wartosci ktorych unikalna wartosc kolumny 'label' to 'category
              '
17        # czyli podzial na czesci ktore maja 0 i 1
18        category_values = df[df[label] == category].drop(labels=label,
              axis=1)
19        temp_minmax = []
20        temp_pair = []
21        # item to wszystkie wartosci z danej kolumny
22        for _, item in category_values.items():
23            # minimum z kolumny
24            min_t = min(item)
25            # maximum z kolumny
26            max_t = max(item)
27            temp_minmax.append([min_t, max_t])
28            # srednia
```

```
29              mean = self.mean(item)
30              len_lower = 0
31              len_upper = 0
32              # zliczamy elementy wieksze od sredniej i mniejsze od
                    sredniej
33              for value in item:
34                  if value > mean:
35                      len_upper += 1
36                  else:
37                      len_lower += 1
38
39              # jesli jest tyle samo powyzej srendiej co ponizej to
40              # wybieramy losowo pare [0, 1] lub [1, 0]
41              if len_upper == len_lower:
42                  temp_pair.append(random.choice(([0, 1],[1, 0])))
43              # jesli wiecej powyzej sredniej to zapisujemy pare [1, 0]
44              elif len_upper > len_lower:
45                  temp_pair.append([1, 0])
46              # jesli wiecej ponizej sredniej to zapisujemy pare [0, 1]
47              else:
48                  temp_pair.append([0, 1])
49          # uzupelniamy listy wszystkich 'minmaxow' i wszystkich 'par'
50          self.minmaxs.append(temp_minmax)
51          self.pairs.append(temp_pair)
```

Metoda tworząca tabelę ważoną zbioru miękkiego. Dzielimy zbiór na część, w której Wynik badania oznacza diabetyka oraz część, w której Wynik badania oznacza osobę zdrową. Dla każdej kolumny w każdej części obliczamy wartość średnią, wartość minimalną oraz wartość maksymalną. Zliczamy ilość elementów większych od średniej oraz elementów mniejszych od średniej. Tworzymy tabelę ważoną zbioru z par [0, 1] i [1, 0]. Pary [0, 1] dodajemy do tabeli ważonej w przypadku, gdy jest więcej elementów poniżej średniej. Pary [1, 0] dodajemy do tabeli ważonej w przypadku, gdy jest więcej elementów powyżej średniej. Jeżeli elementów jest tyle samo, to losujemy czy dodać parę [1, 0] czy [0, 1].

```
1 def predict(self, sample: pd.Series, label: str) -> str:
2     """
3     Receives a sample in the form of pd.Series and returns its predicted
          label
4     :param sample: pd.Series
5     :param label: str
6     :return predicted label of the sample: str
7     """
8     # liczenie elementow dla sampla dla kazdej z klas (0, 1)
9
10    sample = sample.drop(labels=label)
11    probabilities = []
12
13    # bierzemy po kolei odpowiednie pary i srednie
14    for category, minmax in zip(self.pairs, self.minmaxs):
15        sample_pairs = []
16        # bierzemy odpowiednia pare minmax i wartosc z sample'a
17        for pair, x in zip(minmax, sample):
18            # proportion = (wartosc_sampla - min) / (max - min)
```

```
19              proportion = (x - pair[0]) / (pair[1] - pair[0])
20              # dodajemy pare [1 - prop, prop] do listy wszytskich par
21              sample_pairs.append([1 - proportion, proportion])
22
23          temp = []
24          # liczenie prawdopodobienstwa dla kazdej klasy
25          for x, y in zip(category, sample_pairs):
26              temp.append(x[0] * y[0] + x[1] * y[1])
27          probabilities.append(sum(temp))
28      # naszym indeksem jest indeks najwiekszego prawdopodobienstwa
29      idx = max(range(len(probabilities)), key=probabilities.__getitem__)
30      # zwracamy najprawdopodobniejszy wynik z 'labels' (u nas 0 lub 1)
31      return self.labels[idx]
```

Metoda przewidująca etykietę klasy próbki. Znowu tworzymy listę par, jednak teraz bierzemy pod uwagę odpowiednie pary [min, max] oraz kolejne elementy z kolejnych kolumn w naszej próbce. Liczymy proporcję ze wzoru *(wartosc_sampla - min) / (max - min)* i na jej podstawie dodajemy parę *[1 - proporcja, proporcja]* do listy wszystkich par. Następnie na podstawie tych par liczymy prawdopodobieństwo wystąpienia każdej etykiety. Opiera się to na zwykłym iloczynie skalarnym wektorów. W tym przypadku naszymi 'wektorami' są indeksy etykiet oraz pary *[1 - proporcja, proporcja]*. Na koniec zwracamy przewidywaną etykietę (czyli tę, która uzyskała największą wartość na liście prawdopodobieństw).

```
1  def test(self, validation_set: pd.DataFrame, label: str) -> None:
2      """
3      Test how accurate the prediction method is
4      :param validation_set: pd.DataFrame - a record from the validation
           dataset
5      :param label: str
6      :return: None
7      """
8      correct = 0
9      # iterrows() jest uzywane do iteracji wieszy dataframe'a
10     for _, row in validation_set.iterrows():
11         # jesli poprawnie przewidzimy etykiete
12         if self.predict(sample=row, label=label) == row[label]:
13             correct += 1
14     # informacja podsumowujaca
15     # print(f'Accuracy: {correct / len(validation_set) * 100:.2f}%; {
           correct}/{len(validation_set)}')
16     return correct / len(validation_set) * 100
```

Metoda testująca dokładność działania klasyfikatora miękkiego. Prosty mechanizm, który w pętli wywołuje metodę *predict()* oraz porównuje wynik z wynikiem poprawnym. Zwrócona zostaje skuteczność przewidywania w procentach.

2.2.3 Klasyfikator rozmyty - Klasa Fuzzy

Klasa Fuzzy zawiera klasyfikator rozmyty oraz metody pomocnicze.

```
1  def __init__(self):
```

```
2       # slownik slownikow rozmycia dla kazdej kolumny
3       self.antecedents: dict[str: dict[str: tuple[float]]] = {}
4       # slownik zawierajacy nazwe etykiety klasy oraz slownikow z jej
            rozmyciem
5       self.consequent: dict[str: dict[str: tuple[float]]] = {}
6       # lista slownikow; kazda regula to jeden slownik
7       self.rules: list[dict[str: str]] = []
```

Opis atrybutów pomocniczych antecedents, consequent, rules przechowujących słownik słowników rozmycia dla każdej kolumny, słownik zawierający nazwę etykiety klasy oraz słowników z jej rozmyciem, lista słowników, w których każda reguła to jeden słownik.

```
1 def add_antecedent(self, parameter: str, linguistic_value: str, *args)
      -> None:
2       if parameter not in self.antecedents:
3           self.antecedents[parameter] = {}
4       self.antecedents[parameter][linguistic_value] = args
```

Metoda pozwalająca na dodanie poprzednika do słownika **antecedens**, gdzie każdy poprzednik to słownik.

```
1 def add_consequent(self, parameter: str, linguistic_value: str, *args)
      -> None:
2       if parameter not in self.consequent:
3           self.consequent[parameter] = {}
4       self.consequent[parameter][linguistic_value] = args
```

Metoda pozwalająca na dodanie konsekwencji do słownika **consequent**, gdzie każda konsekwencja to słownik.

```
1 def add_rule(self, rule: dict[str: str]) -> None:
2       assert set(rule.keys()) == set.union(set(self.antecedents.keys()),
            set(self.consequent.keys())), print("Nieprawidlowa liczba
            kategorii w regule.")
3       self.rules.append(rule)
```

Metoda służąca do dodawania zasad do słownika **rules**, gdzie każda zasada to słownik.

```
1 @staticmethod
2 def triangular_function(n: float, a: float, b: float, c: float) -> float
      :
3       assert a <= b <= c, print("Nieprawidlowe wartosci")
4       if n == a == b or n == b == c:
5           return 1
6       if n <= a:
7           return 0
8       if a < n < b:
9           return (n - a) / (b - a)
10      if n == b:
11          return 1
```

```
12      if b < n < c:
13          return (c - n) / (c - b)
14      if n >= c:
15          return 0
```

Metoda obliczająca przynależność dla funkcji trójkątnej.

```
1  @staticmethod
2  def trapezoidal_function(n: float, a: float, b: float, c: float, d:
       float) -> float:
3      assert a <= b <= c <= d, print("Nieprawidlowe wartosci")
4      if c == d and n >= d:
5          return 1
6      if a == b and n <= a:
7          return 1
8      if n <= a:
9          return 0
10     if a < n < b:
11         return (n - a) / (b - a)
12     if b <= n <= c:
13         return 1
14     if c < n < d:
15         return (d - n) / (d - c)
16     if n >= d:
17         return 0
```

Metoda obliczająca przynależność dla funkcji trapezoidalnej.

```
1  def membership_function(self, n: float, *args) -> float:
2      assert len(args) in (3, 4), print("Nieprawidlowa liczba argumentow")
3      if len(args) == 3:
4          return self.triangular_function(n, *args)
5      else:
6          return self.trapezoidal_function(n, *args)
```

Metoda wywołująca funkcję przynależności. W zależności od liczby podanych argumentów odwoła się ona do funkcji trójkątnej lub trapezoidalnej.

```
1  def fuzzify(self, column: str, n: float) -> dict[str, float]:
2      return {antecedent: self.membership_function(n, *self.antecedents[
           column][antecedent]) for antecedent in self.antecedents[column].
           keys()}
```

Metoda służąca do rozmycia danych wejściowych. Dla każdej kolumny zwraca słownik, gdzie jest informacja, w jakim stopniu i jakie wartości są dla danej próbki. Z konkretnej wartości liczbowej przechodzimy w rozmyte.

```
1  @staticmethod
2  def rule_fulfillment(rule: dict[str: str], fuzzy_values: dict[str: dict[
       str: float]]) -> tuple[str, float]:
```

```
    label = list(rule.keys())[-1]
    minimum = float('inf')
    # dla kazdej wartosci w danej zasadzie
    for name, linguistic_value in list(rule.items())[:-1]:
        maximum = float('-inf')
        # jesli mamy wartosci rozdzielone 'or', to bierzemy ich maksimum
        for lvs in linguistic_value.split(' | '):
            maximum = max(fuzzy_values[name][lvs], maximum)
        # bierzemy minimum stopni spelnienia wartosci danej zasady
        minimum = min(minimum, maximum)
    return rule[label], minimum
```

Metoda obliczająca stopień spełnienia danej zasady.

```
@staticmethod
def area(a: float, c: float) -> float:
    return (c - a) / 2
```

Metoda obliczająca powierzchnię danego wyniku.

```
@staticmethod
def cog(a: float, b: float, c: float) -> float:
    return (a + b + c) / 3
```

Metoda obliczająca środek ciężkości.

```
def aggregate(self, label: str, outputs: defaultdict) -> float:
    # licznik i mianownik dla funkcji agregujacej
    counter = []
    denominator = []
    # obliczanie powierzchni oraz srodka ciezkosci kazdej etykiety
       funkcji przynaleznosci
    for name, membership in outputs.items():
        if not membership:
            continue
        a, b, c = self.consequent[label][name]
        area = self.area(a, c)
        cog = self.cog(a, b, c)
        counter.append(membership * area * cog)
        denominator.append(membership * area)
    return round(sum(counter) / sum(denominator), 3)
```

Metoda agregująca, wyostrzająca wartość końcową za pomocą metody środka ciężkości.

```
def compute(self, sample: pd.Series) -> tuple[str, float]:
    # rozmycie probki
    fuzzy_values = {column: self.fuzzify(column, sample[column]) for
       column in self.antecedents.keys()}

    # obliczenie stopnia spelnienia zasad
    fulfillments = [self.rule_fulfillment(rule, fuzzy_values) for rule
       in self.rules]
```

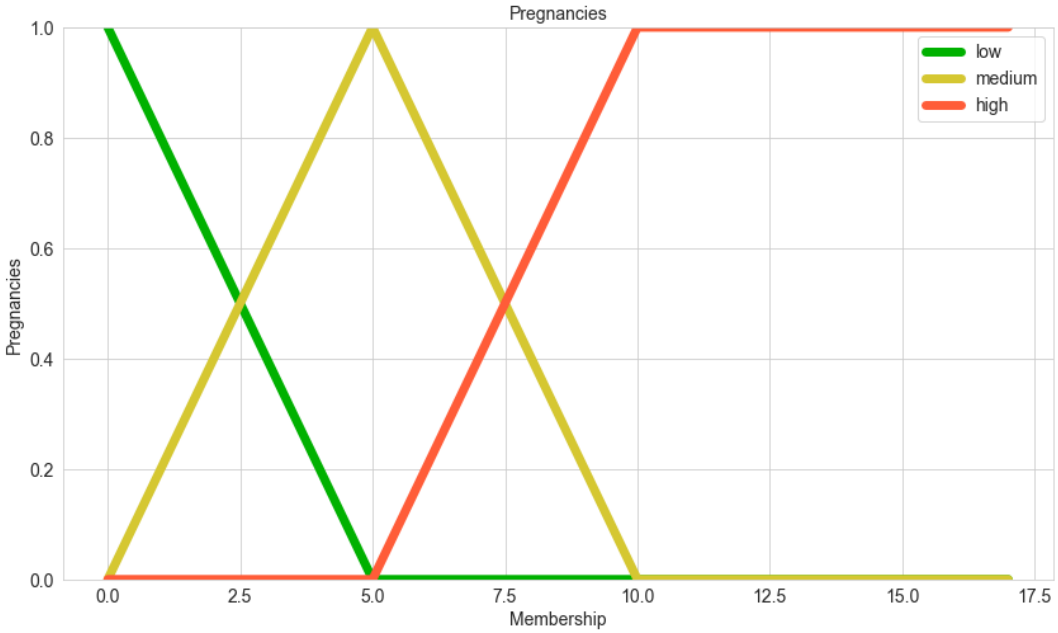
```
7
8       # wybranie maksymalnego spelnienia reguly dla danych wartosci
9       outputs: defaultdict = defaultdict(lambda: 0)
10      for linguistic_value, fulfillment in fulfillments:
11          outputs[linguistic_value] = max(outputs[linguistic_value],
                fulfillment)
12      outputs = dict(outputs)
13      label = list(self.consequent.keys())[0]
14
15      # wybor przewidzianej etykiety klasy: w przypadku remisu dla dwoch
            etykiet wybieramy chorobe
16      output_names = sorted(outputs.keys(), key=lambda x: (-1 * outputs[x
            ], x))
17
18      # print(sample['Outcome'], self.aggregate(label, outputs), outputs)
19      return output_names[0], self.aggregate(label, outputs)
```

Metoda przewidująca etykietę klasy.

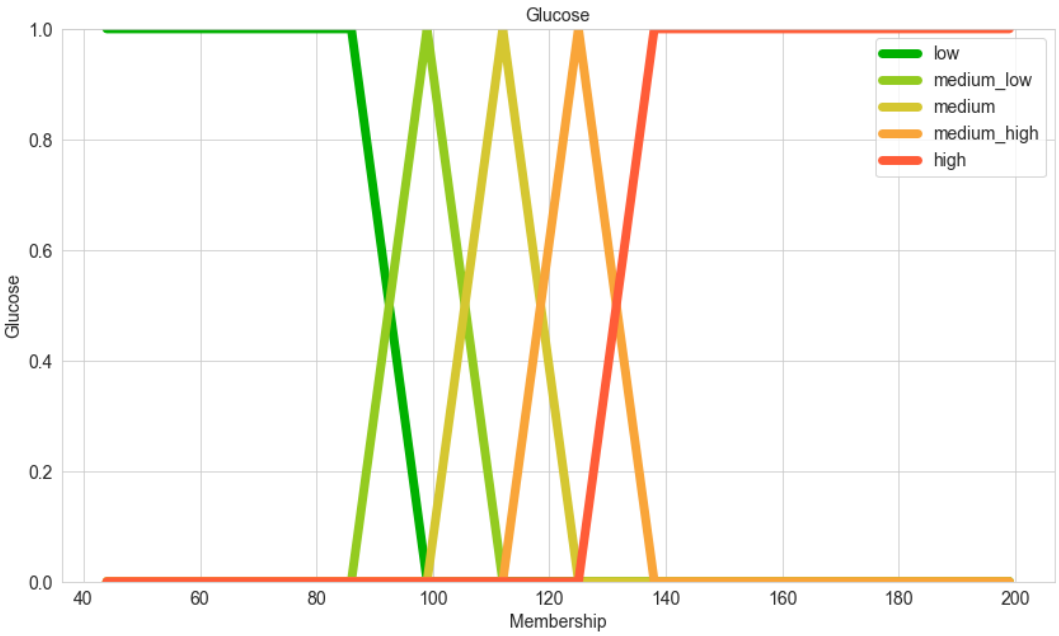
```
1 def view(self, parameter: str) -> None:
2       # lista kolorow do wykresow
3       colors: dict[int: tuple] = {2: ('#00b100', '#ff5c38'),
4                                   3: ('#00b100', '#d5c731', '#ff5c38'),
5                                   4: ('#00b100', '#adcd27', '#f1b438', '#
                                        ff5c38'),
6                                   5: ('#00b100', '#93cb20', '#d5c731', '#
                                        f9a539', '#ff5c38')}
7       # globalne parametry dla wykresow matplotlib.pyplot
8       plt.rcParams["figure.figsize"] = (14, 8)
9       if parameter in self.consequent.keys():
10          source = self.consequent[parameter]
11      else:
12          source = self.antecedents[parameter]
13
14      # stworzenie dziedziny
15      minimum: int = floor(list(source.values())[0][0])
16      maximum: int = ceil(list(source.values())[-1][-1])
17      x = np.linspace(minimum, maximum, num=(maximum - minimum) * 100 + 1)
18      # stworzenie wykresow funkcji przynaleznosci
19      for k, v, color in zip(source.keys(), source.values(), colors[len(
            source.keys())]):
20          y = np.array([self.membership_function(i, *v) for i in x])
21          plt.plot(x, y, color, label=k)
22
23      # nadanie parametrow wykresu
24      plt.rc('font', size=16)
25      plt.rc('axes', titlesize=14)
26      plt.rc('axes', labelsize=14)
27      plt.rc('xtick', labelsize=14)
28      plt.rc('ytick', labelsize=14)
29      plt.rc('legend', fontsize=14)
30      plt.rc('figure', titlesize=16)
31      plt.title(parameter)
32      plt.xlabel('Membership')
33      plt.ylabel(parameter)
```

```
34      plt.legend(loc="upper right")
35      plt.ylim(0, 1)
36      plt.show()
```

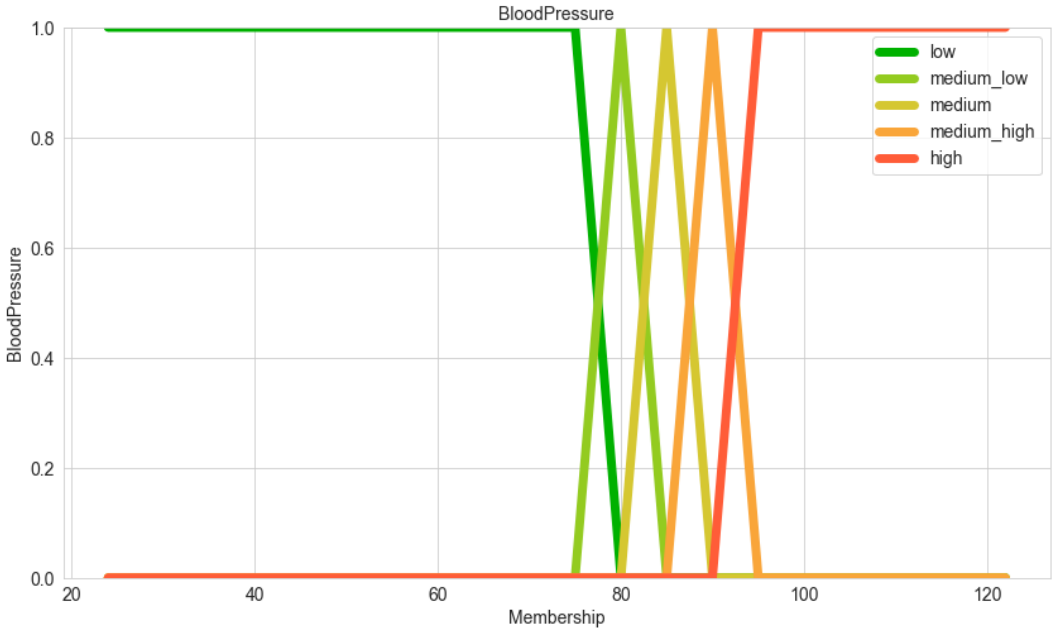
Metoda pomocnicza wyświetlająca wykres rozmycia cechy.
Poniżej wykresy wygenerowane przy pomocy metody *view()*.



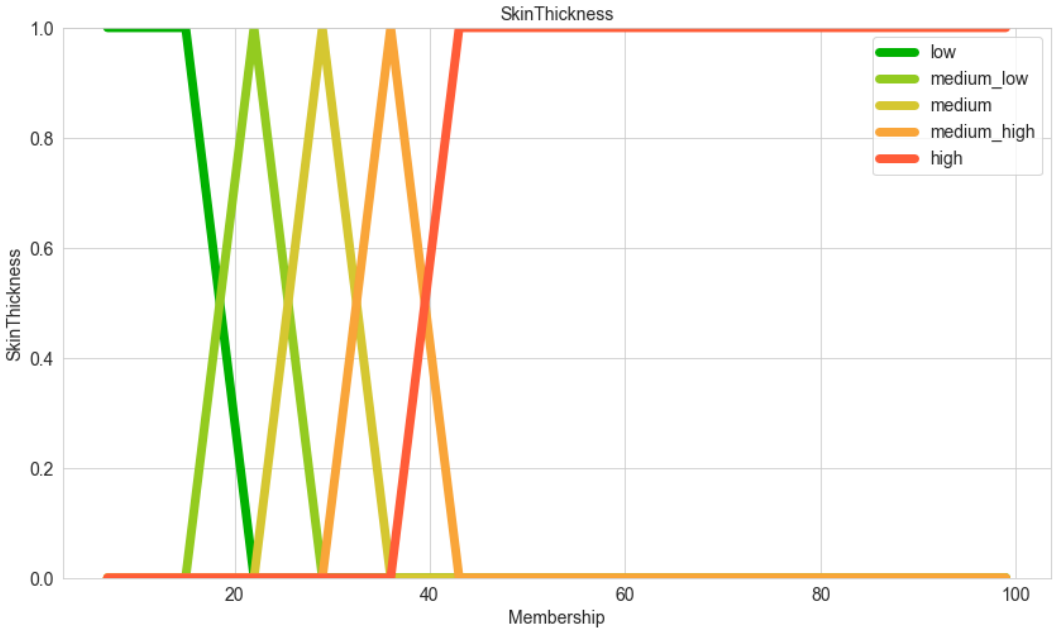
Rysunek 13: Wykres funkcji przynależności dla kolumny Pregnancies



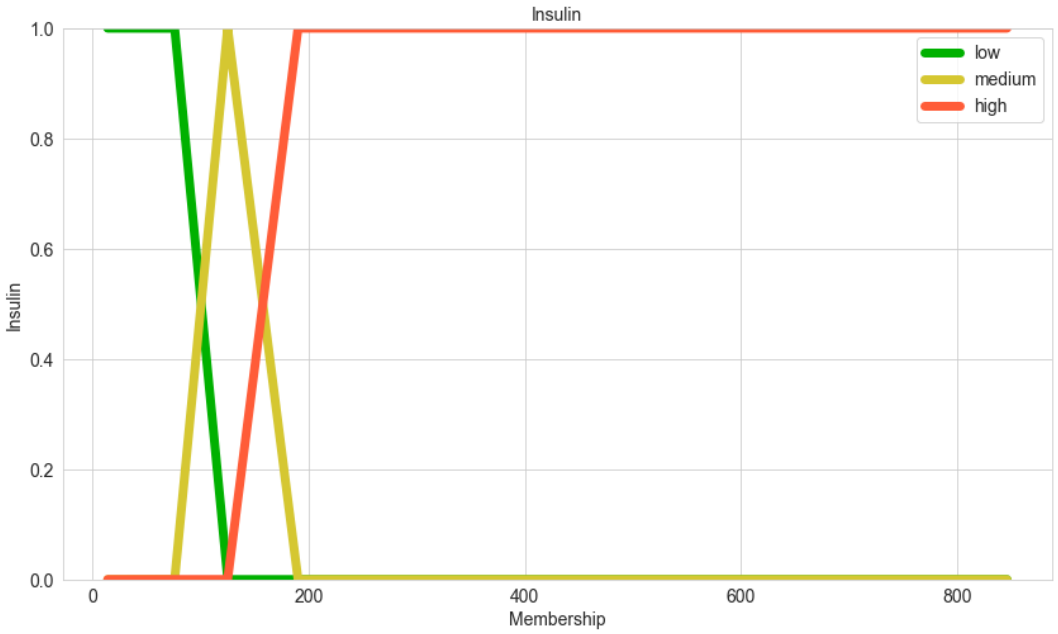
Rysunek 14: Wykres funkcji przynależności dla kolumny Glucose



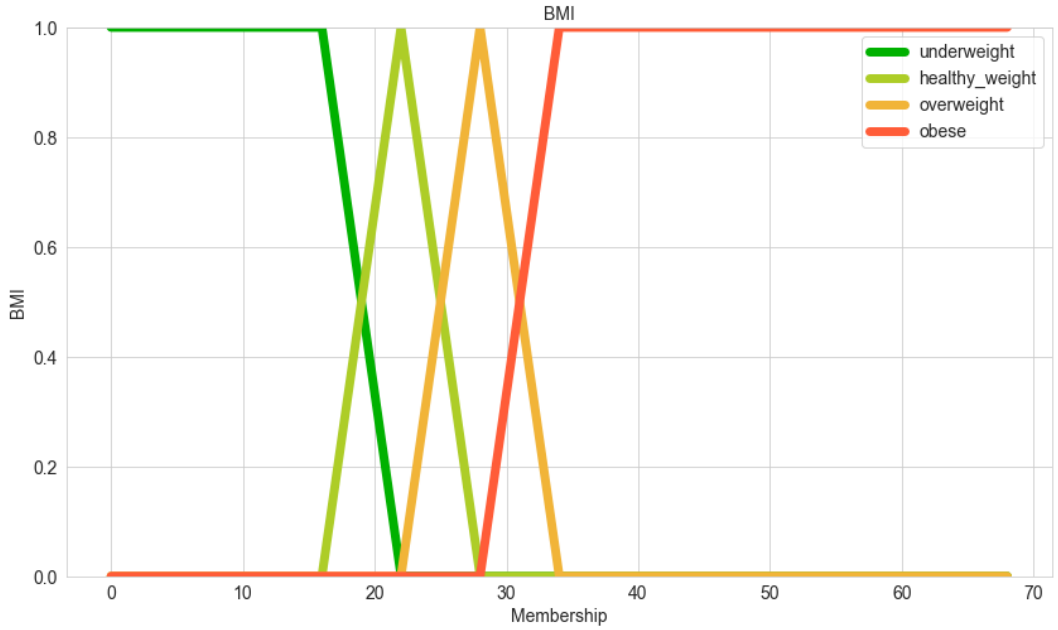
Rysunek 15: Wykres funkcji przynależności dla kolumny BloodPressure



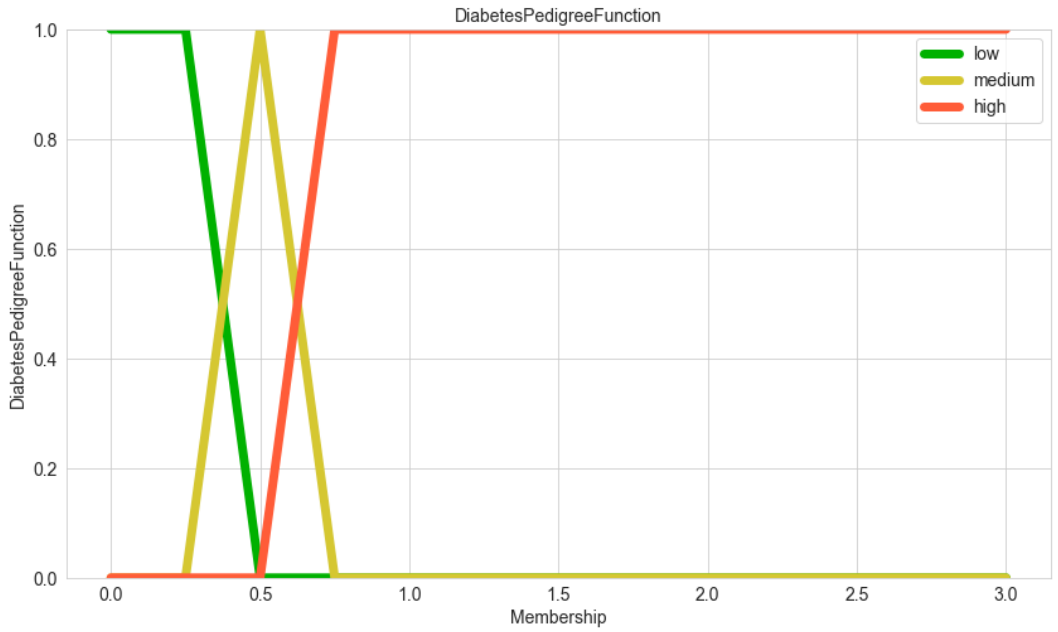
Rysunek 16: Wykres funkcji przynależności dla kolumny SkinThickness



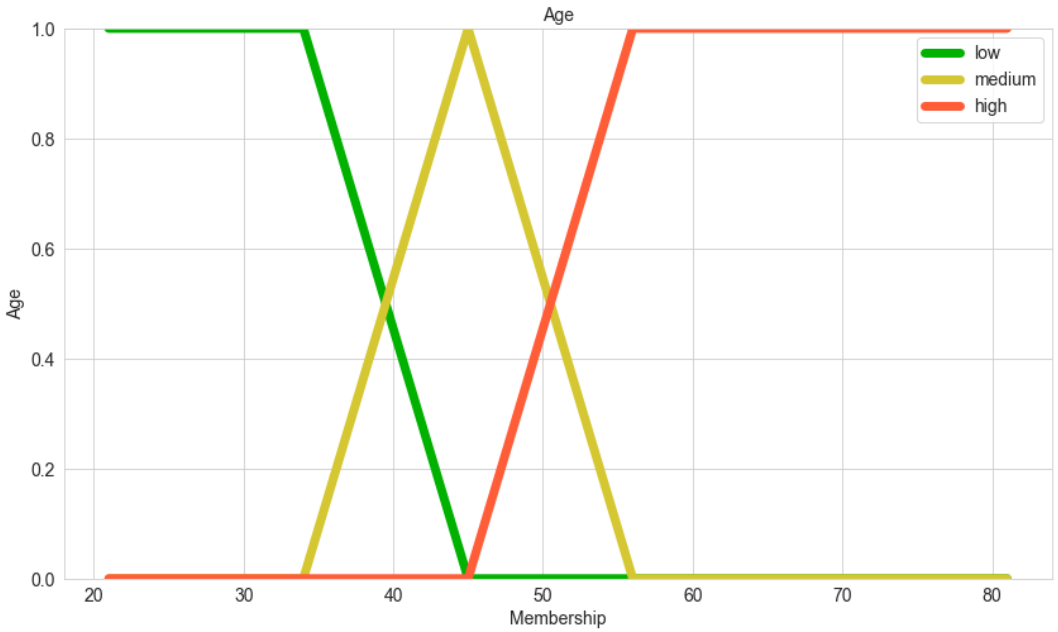
Rysunek 17: Wykres funkcji przynależności dla kolumny Insulin



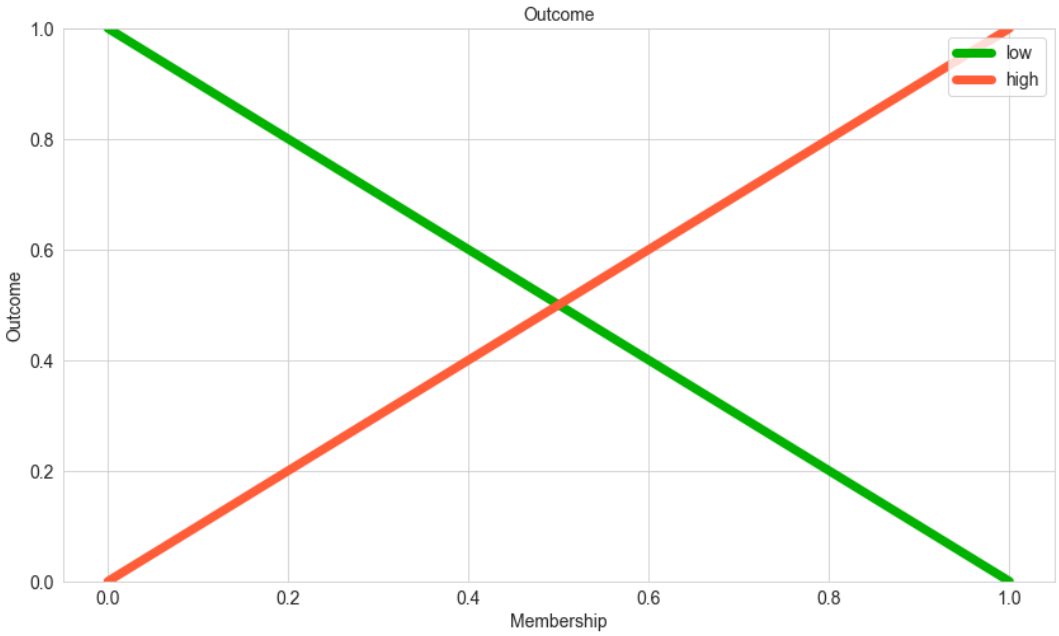
Rysunek 18: Wykres funkcji przynależności dla kolumny BMI



Rysunek 19: Wykres funkcji przynależności dla kolumny DiabetesPedigreeFunction



Rysunek 20: Wykres funkcji przynależności dla kolumny Age



Rysunek 21: Wykres funkcji przynależności dla kolumny Outcone

2.3 Testy

2.3.1 Test metod klasy ProcessingData

```
shuffled_data = ProcessingData.shuffle(data)
shuffled_data
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	4	84	90	23	56	39.5	0.159	25	0
1	8	126	88	36	108	38.5	0.349	49	0
2	3	115	66	39	140	38.1	0.150	28	0
3	0	177	60	29	478	34.6	1.072	21	1
4	1	79	80	25	37	25.4	0.583	22	0
...	...	...	...	...	...	...	...	...	...
387	0	124	56	13	105	21.8	0.452	21	0
388	2	144	58	33	135	31.6	0.422	25	1
389	9	154	78	30	100	30.9	0.164	45	0
390	1	84	64	23	115	36.9	0.471	28	0
391	6	165	68	26	168	33.6	0.631	49	0

Rysunek 22: Wyświetlenie przetasowanych danych przy pomocy metody *shuffle()*

```
normalized_data_std = ProcessingData.normalize_std(shuffled_data, 'Outcome')
normalized_data_std
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	0.217654	-1.251671	1.547423	-0.584363	-0.841928	0.912647	-1.053715	-0.574936	0
1	1.463207	0.109279	1.387373	0.651799	-0.404371	0.770353	-0.503768	1.777826	0
2	-0.093734	-0.247160	-0.373178	0.937067	-0.135105	0.713435	-1.079765	-0.280841	0
3	-1.027899	1.761862	-0.853328	-0.013827	2.709015	0.215403	1.588924	-0.967063	1
4	-0.716511	-1.413689	0.747172	-0.394184	-1.001804	-1.093710	0.173534	-0.869031	0
...	...	...	...	...	...	...	...	...	...
387	-1.027899	0.044472	-1.173428	-1.535257	-0.429615	-1.605972	-0.205639	-0.967063	0
388	-0.405123	0.692544	-1.013378	0.366531	-0.177178	-0.211482	-0.292473	-0.574936	1
389	1.774596	1.016580	0.587122	0.081263	-0.471687	-0.311089	-1.039243	1.385699	0
390	-0.716511	-1.251671	-0.533228	-0.584363	-0.345469	0.542681	-0.150645	-0.280841	0
391	0.840431	1.373019	-0.213128	-0.299095	0.100502	0.073108	0.312468	1.777826	0

Rysunek 23: Wyświetlenie znormalizowanych danych przy pomocy metody *normalize_std()*

```
normalized_data_minmax = ProcessingData.normalize_minmax(shuffled_data, 'Outcome')
normalized_data_minmax
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	0.235294	0.197183	0.767442	0.285714	0.050481	0.435583	0.031692	0.066667	0
1	0.470588	0.492958	0.744186	0.517857	0.112981	0.415133	0.113062	0.466667	0
2	0.176471	0.415493	0.488372	0.571429	0.151442	0.406953	0.027837	0.116667	0
3	0.000000	0.852113	0.418605	0.392857	0.557692	0.335378	0.422698	0.000000	1
4	0.058824	0.161972	0.651163	0.321429	0.027644	0.147239	0.213276	0.016667	0
...	...	...	...	...	...	...	...	...	...
387	0.000000	0.478873	0.372093	0.107143	0.109375	0.073620	0.157173	0.000000	0
388	0.117647	0.619718	0.395349	0.464286	0.145433	0.274029	0.144325	0.066667	1
389	0.529412	0.690141	0.627907	0.410714	0.103365	0.259714	0.033833	0.400000	0
390	0.058824	0.197183	0.465116	0.285714	0.121394	0.382413	0.165310	0.116667	0
391	0.352941	0.767606	0.511628	0.339286	0.185096	0.314928	0.233833	0.466667	0

Rysunek 24: Wyświetlenie znormalizowanych danych przy pomocy metody *normalize_minmax()*

```
# Test dla normalizacji std
normalized_training_data_std, normalized_validation_data_std = ProcessingData.split(normalized_data_std, 0.7)
normalized_training_data_std
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	0.217654	-1.251671	1.547423	-0.584363	-0.841928	0.912647	-1.053715	-0.574936	0
1	1.463207	0.109279	1.387373	0.651799	-0.404371	0.770353	-0.503768	1.777826	0
2	-0.093734	-0.247160	-0.373178	0.937067	-0.135105	0.713435	-1.079765	-0.280841	0
3	-1.027899	1.761862	-0.853328	-0.013827	2.709015	0.215403	1.588924	-0.967063	1
4	-0.716511	-1.413689	0.747172	-0.394184	-1.001804	-1.093710	0.173534	-0.869031	0
...	...	...	...	...	...	...	...	...	...
269	-0.716511	-0.700810	-1.653578	-1.345078	-1.010219	-1.264464	0.008550	-0.476904	0
270	-0.405123	1.113791	0.267022	0.556709	2.389262	0.898418	-1.126076	-0.084777	0
271	1.463207	1.729459	1.547423	0.461620	1.211224	0.087337	-0.162222	2.660112	1
272	0.217654	-0.376774	0.106972	1.697782	0.428670	0.571140	2.509361	2.464048	1
273	-0.405123	-0.020335	-0.853328	-1.059810	-0.421200	-0.467613	0.561392	-0.869031	0

274 rows × 9 columns Open in new tab

```
normalized_validation_data_std
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	-1.027899	-0.668407	1.227323	-1.154899	-0.429615	-0.538760	0.497714	-0.378873	0
1	-0.716511	0.174087	-1.813628	1.507603	0.319281	1.054942	0.260368	-0.672968	1
2	-0.405123	0.141683	-1.013378	-0.489274	1.000860	-0.766432	3.117196	-0.574936	0
3	-0.093734	-0.506389	-0.693278	-1.535257	-0.909244	-1.449448	0.448508	-0.771000	1
4	-0.716511	0.238894	-0.053078	-1.535257	-0.429615	-1.022563	-0.147750	-0.869031	0
...	...	...	...	...	...	...	...	...	...
113	-1.027899	0.044472	-1.173428	-1.535257	-0.429615	-1.605972	-0.205639	-0.967063	0
114	-0.405123	0.692544	-1.013378	0.366531	-0.177178	-0.211482	-0.292473	-0.574936	1
115	1.774596	1.016580	0.587122	0.081263	-0.471687	-0.311089	-1.039243	1.385699	0
116	-0.716511	-1.251671	-0.533228	-0.584363	-0.345469	0.542681	-0.150645	-0.280841	0
117	0.840431	1.373019	-0.213128	-0.299095	0.100502	0.073108	0.312468	1.777826	0

Rysunek 25: Wyświetlenie zbioru treningowego i walidacyjnego podzielonych przy pomocy metody *split()* dla danych znormalizowanych metodą *normalize_std()*

```
# Test dla normalizacji minmax
normalized_training_data_minmax, normalized_validation_data_minmax = ProcessingData.split(normalized_data_minmax, 0.7)
normalized_training_data_minmax
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	0.235294	0.197183	0.767442	0.285714	0.050481	0.435583	0.031692	0.066667	0
1	0.470588	0.492958	0.744186	0.517857	0.112981	0.415133	0.113062	0.466667	0
2	0.176471	0.415493	0.488372	0.571429	0.151442	0.406953	0.027837	0.116667	0
3	0.000000	0.852113	0.418605	0.392857	0.557692	0.335378	0.422698	0.000000	1
4	0.058824	0.161972	0.651163	0.321429	0.027644	0.147239	0.213276	0.016667	0
...	...	...	...	...	...	...	...	...	...
269	0.058824	0.316901	0.302326	0.142857	0.026442	0.122699	0.188865	0.083333	0
270	0.117647	0.711268	0.581395	0.500000	0.512019	0.433538	0.020985	0.150000	0
271	0.470588	0.845070	0.767442	0.482143	0.343750	0.316973	0.163597	0.616667	1
272	0.235294	0.387324	0.558140	0.714286	0.231971	0.386503	0.558887	0.583333	1
273	0.117647	0.464789	0.418605	0.196429	0.110577	0.237219	0.270664	0.016667	0

274 rows × 9 columns Open in new tab

```
normalized_validation_data_minmax
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	0.000000	0.323944	0.720930	0.178571	0.109375	0.226994	0.261242	0.100000	0
1	0.058824	0.507042	0.279070	0.678571	0.216346	0.456033	0.226124	0.050000	1
2	0.117647	0.500000	0.395349	0.303571	0.313702	0.194274	0.648822	0.066667	0
3	0.176471	0.359155	0.441860	0.107143	0.040865	0.096115	0.253961	0.033333	1
4	0.058824	0.521127	0.534884	0.107143	0.109375	0.157464	0.165739	0.016667	0
...	...	...	...	...	...	...	...	...	...
113	0.000000	0.478873	0.372093	0.107143	0.109375	0.073620	0.157173	0.000000	0
114	0.117647	0.619718	0.395349	0.464286	0.145433	0.274029	0.144325	0.066667	1
115	0.529412	0.690141	0.627907	0.410714	0.103365	0.259714	0.033833	0.400000	0
116	0.058824	0.197183	0.465116	0.285714	0.121394	0.382413	0.165310	0.116667	0
117	0.352941	0.767606	0.511628	0.339286	0.185096	0.314928	0.233833	0.466667	0

Rysunek 26: Wyświetlenie zbioru treningowego i walidacyjnego podzielonych przy pomocy metody *split()* dla danych znormalizowanych metodą *normalize_minmax()*

2.3.2 Test metod klasy NaiveBayes

```
Accuracy: 76.27%; 90/118
Accuracy: 73.73%; 87/118
Accuracy: 76.27%; 90/118
Accuracy: 77.12%; 91/118
Accuracy: 75.42%; 89/118
```

Rysunek 27: Test klasyfikatora *NaiveBayes* dla danych znormalizowanych przy pomocy metody *normalize_std()*

Testy dokładności przewidywań zostały przeprowadzone dla 50 próbek. Średnia dokładność wyniosła 74.98%

```
Accuracy: 74.58%; 88/118
Accuracy: 73.73%; 87/118
Accuracy: 72.88%; 86/118
Accuracy: 77.97%; 92/118
Accuracy: 78.81%; 93/118
```

Rysunek 28: Test klasyfikatora *NaiveBayes* dla danych znormalizowanych przy pomocy metody *normalize_minmax()*

Testy dokładności przewidywań zostały przeprowadzone dla 50 próbek. Średnia dokładność wyniosła 75.97%

```
Accuracy: 74.58%; 88/118
Accuracy: 76.27%; 90/118
Accuracy: 82.20%; 97/118
Accuracy: 77.97%; 92/118
Accuracy: 73.73%; 87/118
```

Rysunek 29: Test klasyfikatora *NaiveBayes* dla danych nieznormalizowanych

Testy dokładności przewidywań zostały przeprowadzone dla 50 próbek. Średnia dokładność wyniosła 75.56%

2.3.3 Test metod klasy KNN

```
Accuracy for k=2: 67.80%, correct predictions: 80/118
Accuracy for k=3: 66.95%, correct predictions: 79/118
Accuracy for k=4: 59.32%, correct predictions: 70/118
Accuracy for k=2: 80.51%, correct predictions: 95/118
Accuracy for k=3: 77.12%, correct predictions: 91/118
Accuracy for k=4: 72.88%, correct predictions: 86/118
```

Rysunek 30: Test klasyfikatora *KNN* dla danych znormalizowanych przy pomocy metody *normalize_std()*

Testy dokładności przewidywań zostały przeprowadzone dla 50 próbek. Średnia dokładność wyniosła 71.31%

```
Accuracy for k=2: 70.34%, correct predictions: 83/118
Accuracy for k=3: 69.49%, correct predictions: 82/118
Accuracy for k=4: 67.80%, correct predictions: 80/118
Accuracy for k=2: 68.64%, correct predictions: 81/118
Accuracy for k=3: 65.25%, correct predictions: 77/118
Accuracy for k=4: 63.56%, correct predictions: 75/118
```

Rysunek 31: Test klasyfikatora *KNN* dla danych znormalizowanych przy pomocy metody *normalize_minmax()*

Testy dokładności przewidywań zostały przeprowadzone dla 50 próbek. Średnia dokładność wyniosła 71.51%

```
Accuracy for k=2: 63.56%, correct predictions: 75/118
Accuracy for k=3: 59.32%, correct predictions: 70/118
Accuracy for k=4: 54.24%, correct predictions: 64/118
Accuracy for k=2: 67.80%, correct predictions: 80/118
Accuracy for k=3: 66.10%, correct predictions: 78/118
Accuracy for k=4: 58.47%, correct predictions: 69/118
```

Rysunek 32: Test klasyfikatora *KNN* dla danych nieznormalizowanych

Testy dokładności przewidywań zostały przeprowadzone dla 50 próbek. Średnia dokładność wyniosła 65.59%

2.3.4 Test metod klasy SoftSetClassifierMean

```
Accuracy: 43.22%; 51/118
Accuracy: 45.76%; 54/118
Accuracy: 54.24%; 64/118
Accuracy: 67.80%; 80/118
Accuracy: 61.86%; 73/118
```

Rysunek 33: Test *klasyfikatora miękkiego* bazującego na średniej dla danych znormalizowanych przy pomocy metody *normalize_std()*

Testy dokładności przewidywań zostały przeprowadzone dla 50 próbek. Średnia dokładność wyniosła 48.44%

```
Accuracy: 34.75%; 41/118
Accuracy: 34.75%; 41/118
Accuracy: 41.53%; 49/118
Accuracy: 39.83%; 47/118
Accuracy: 53.39%; 63/118
```

Rysunek 34: Test *klasyfikatora miękkiego* bazującego na średniej dla danych znormalizowanych przy pomocy metody *normalize_minmax()*

Testy dokładności przewidywań zostały przeprowadzone dla 50 próbek. Średnia dokładność wyniosła 50.08%

```
Accuracy: 36.44%; 43/118
Accuracy: 50.85%; 60/118
Accuracy: 52.54%; 62/118
Accuracy: 53.39%; 63/118
Accuracy: 63.56%; 75/118
```

Rysunek 35: Test *klasyfikatora miękkiego* bazującego na średniej dla danych nieznormalizowanych

Testy dokładności przewidywań zostały przeprowadzone dla 50 próbek. Średnia dokładność wyniosła 50.12%

2.3.5 Test metod klasy SoftSetClassifierPercentage

```
Accuracy: 27.97%; 33/118
Accuracy: 41.53%; 49/118
Accuracy: 70.34%; 83/118
Accuracy: 33.05%; 39/118
Accuracy: 30.51%; 36/118
```

Rysunek 36: Test *klasyfikatora miękkiego* bazującego na procentach dla danych znormalizowanych przy pomocy metody *normalize_ std()*

Testy dokładności przewidywań zostały przeprowadzone dla 50 próbek. Średnia dokładność wyniosła 55.12%

```
Accuracy: 71.19%; 84/118
Accuracy: 44.92%; 53/118
Accuracy: 29.66%; 35/118
Accuracy: 64.41%; 76/118
Accuracy: 46.61%; 55/118
```

Rysunek 37: Test *klasyfikatora miękkiego* bazującego na procentach dla danych znormalizowanych przy pomocy metody *normalize_ minmax()*

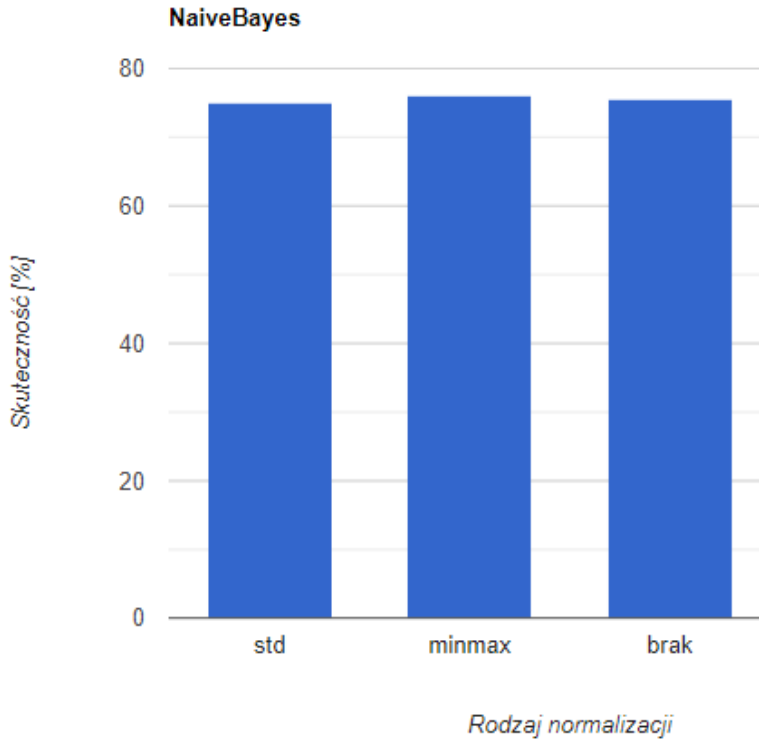
Testy dokładności przewidywań zostały przeprowadzone dla 50 próbek. Średnia dokładność wyniosła 53.32%

```
Accuracy: 68.64%; 81/118
Accuracy: 58.47%; 69/118
Accuracy: 61.02%; 72/118
Accuracy: 33.90%; 40/118
Accuracy: 68.64%; 81/118
```

Rysunek 38: Test *klasyfikatora miękkiego* bazującego na procentach dla danych nieznormalizowanych

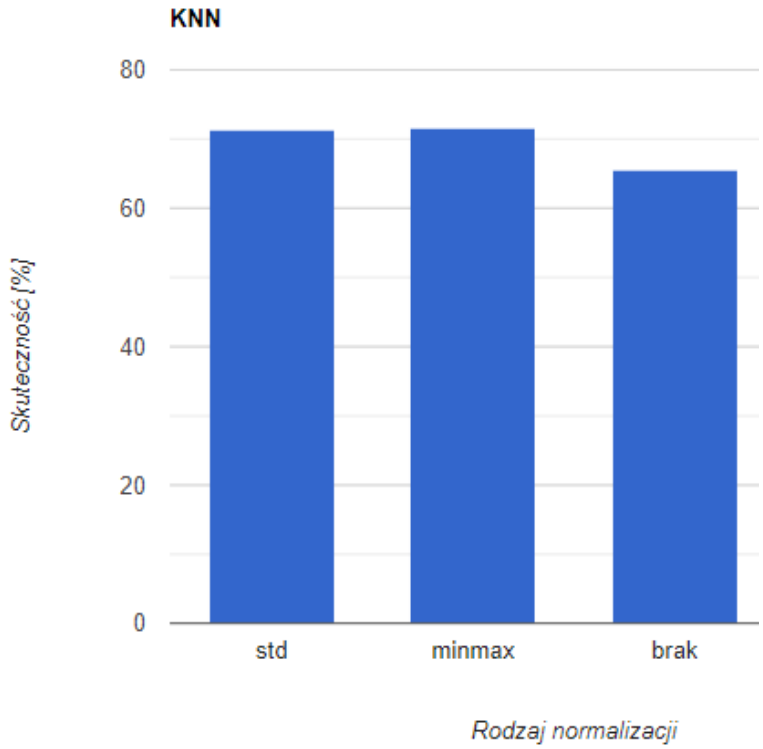
Testy dokładności przewidywań zostały przeprowadzone dla 50 próbek. Średnia dokładność wyniosła 52.93%

2.4 Eksperymenty



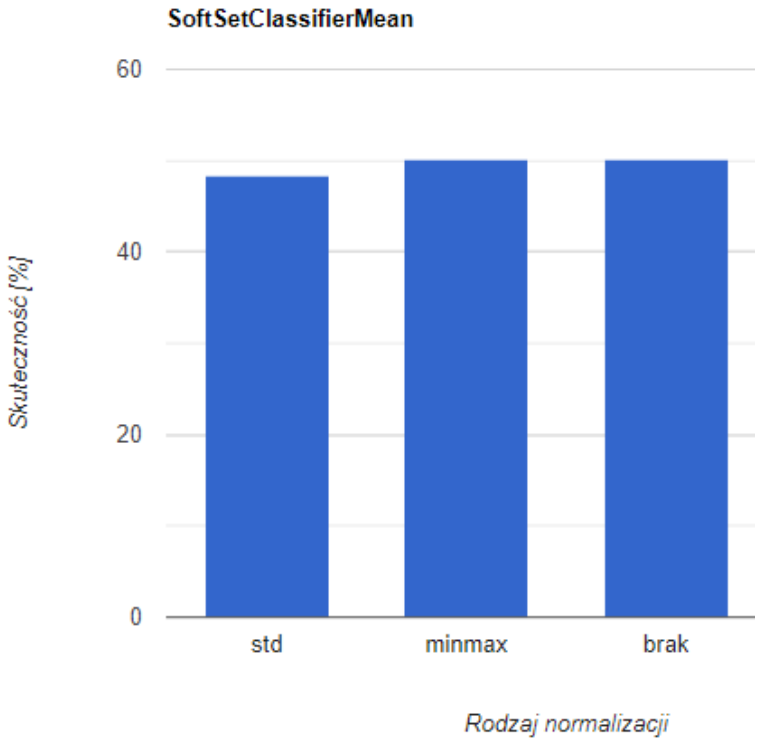
Rysunek 39: Podsumowanie wyników dokładności klasy NaiveBayes

Najlepszą dokładność przewidywanych wyników za pomocą klasy NaiveBayes uzyskaliśmy dla danych znormalizowanych przy użyciu wzoru opierającego się na minimum i maksimum - 75.97%. Niewiele mniej, bo 75.56% dla danych nieznormalizowanych. Najgorszy wynik uzyskaliśmy dla danych znormalizowanych przy użyciu wzoru opierającego się na odchyleniu standardowym - 74.98%. Jednocześnie należy pamiętać, że są to najlepsze wyniki spośród wszystkich zaprezentowanych klasyfikatorów.



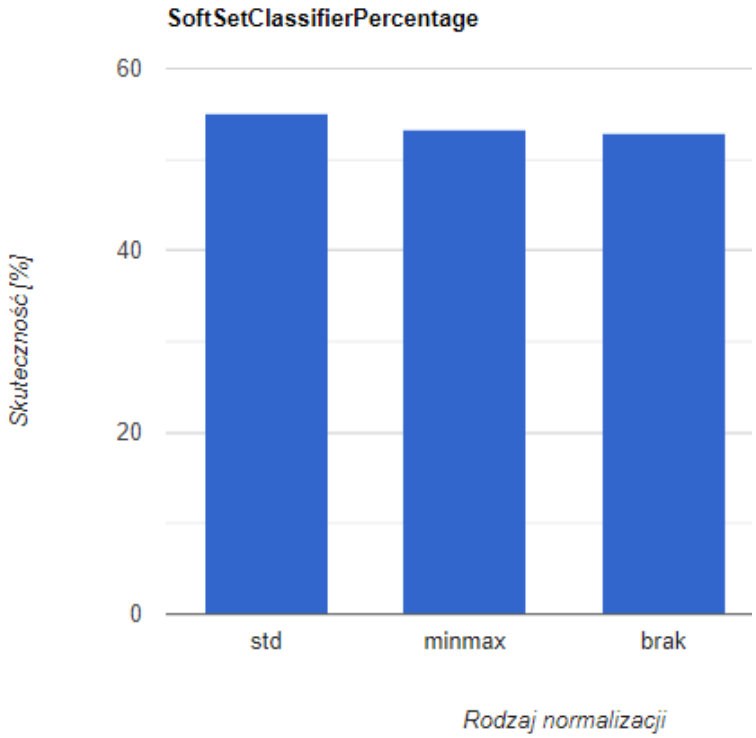
Rysunek 40: Podsumowanie wyników dokładności klasy KNN

Najlepszą dokładność przewidywanych wyników za pomocą klasy KNN uzyskaliśmy dla danych znormalizowanych przy użyciu wzoru opierającego się na minimum i maksimum - 71.51%. Wynik niemal identyczny - 71.31% - dla danych znormalizowanych przy użyciu wzoru opierającego się na odchyleniu standardowym. Najgorszy wynik uzyskaliśmy dla danych nieznormalizowanych - 65.59%. Uzyskane wyniki plasują klasyfikator KNN na drugim miejscu zaraz za klasyfikatorem NaiveBayes w naszym osobistym rankingu testowanych klasyfikatorów.



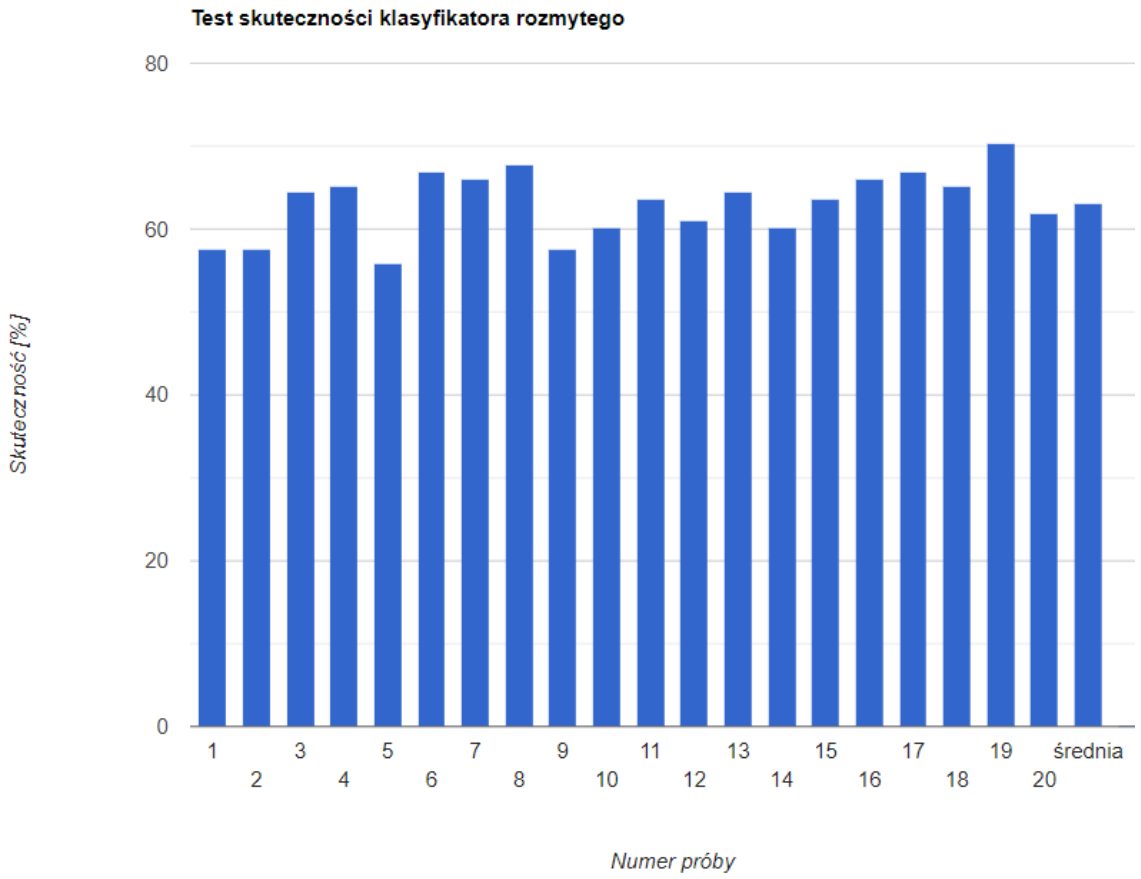
Rysunek 41: Podsumowanie wyników dokładności klasy SoftSetClassifierMean

Najlepszą dokładność przewidywanych wyników za pomocą klasy SoftSetClassifierMean uzyskaliśmy dla danych nieznormalizowanych - 50.12%. Wynik niemal identyczny - 50.08% - dla danych znormalizowanych przy użyciu wzoru opierającego się na minimum i maksimum. Najgorszy wynik uzyskaliśmy dla danych znormalizowanych przy użyciu wzoru opierającego się na odchyleniu standardowym - 48.44%. Wyniki są niezadowalające ze względu na użycie średniej w klasyfikatorze. Można powiedzieć, że przypominają losowe przewidywania ze względu na skuteczność bliską 50%. Jest to najmniej efektywny klasyfikator spośród zaprezentowanych.



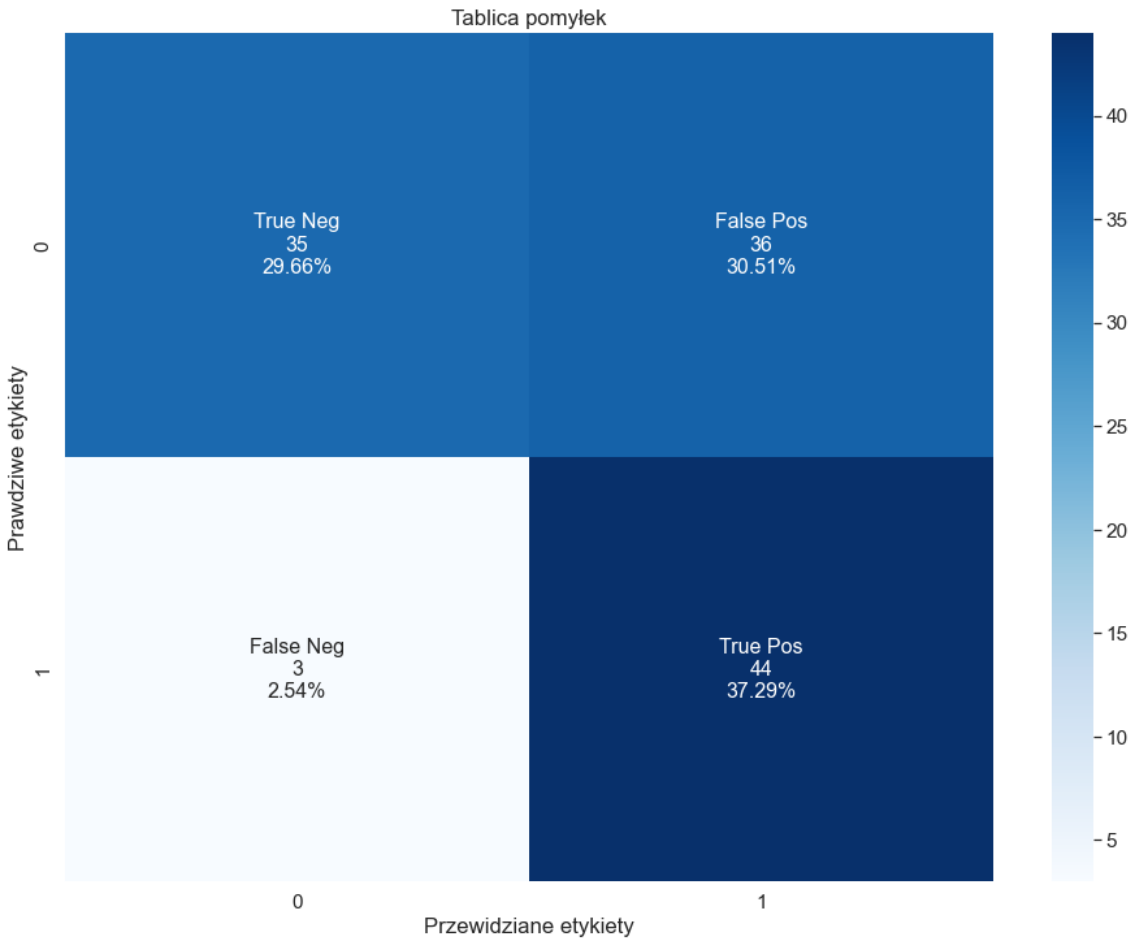
Rysunek 42: Podsumowanie wyników dokładności klasy SoftSetClassifierPercentage

Najlepszą dokładność przewidywanych wyników za pomocą klasy SoftSetClassifierPercentage uzyskaliśmy dla danych znormalizowanych przy użyciu wzoru opierającego się na odchyleniu standardowym - 55.12%. Niemal o 2 punkty procentowe gorszy wynik został uzyskany dla danych znormalizowanych przy użyciu wzoru opierającego się na minimum i maksimum - 53.32%. Najgorszy wynik uzyskaliśmy dla danych nieznormalizowanych - 52.93%. Uzyskane wyniki plasują w naszym rankingu klasyfikator SoftSetClassifierPercentage na przedostatnim miejscu zaraz za klasyfikatorem rozmytym.



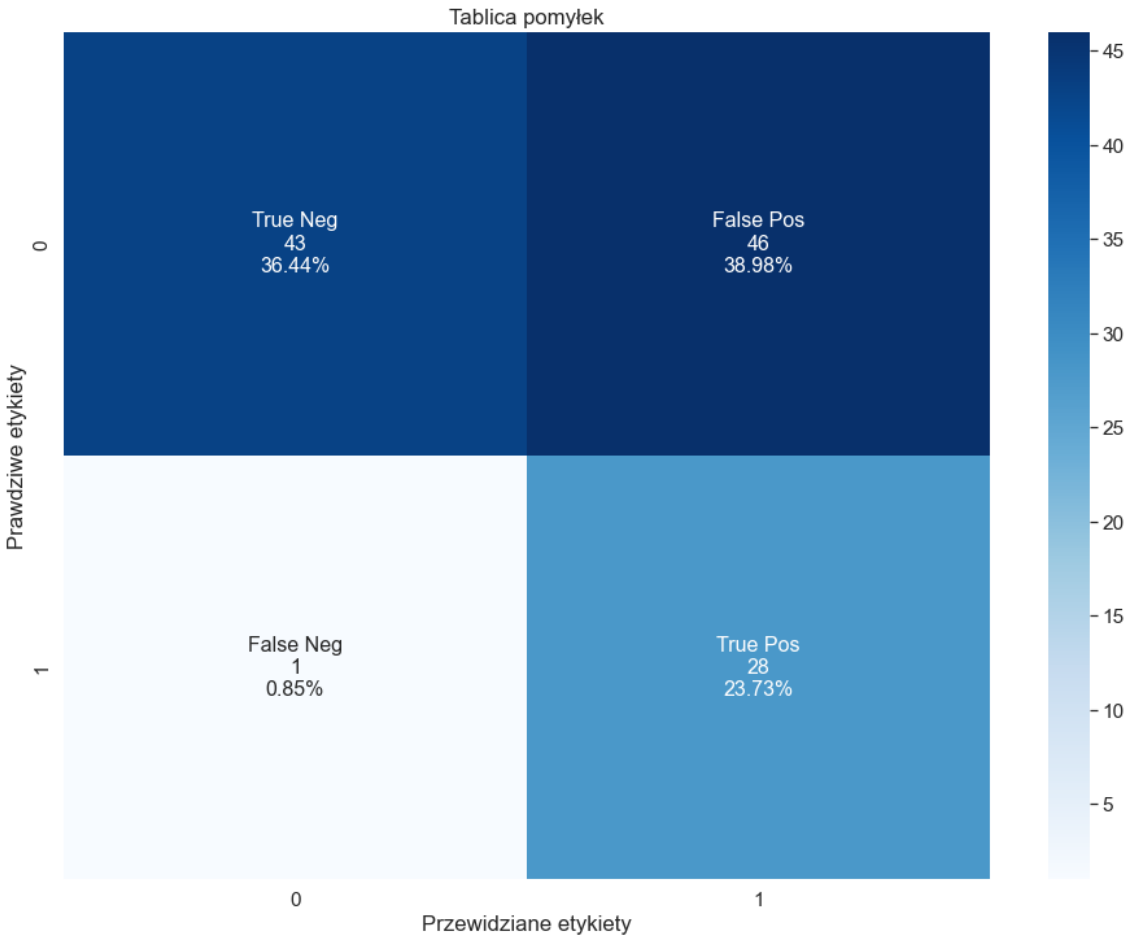
Rysunek 43: Podsumowanie wyników dokładności Klasyfikatora rozmytego

Najlepszy spośród zaproponowanych przez nas klasyfikatorów i jednocześnie trzecim spośród wszystkich poznanych okazał się klasyfikator rozmyty ze średnią dokładnością przewidywań równą 63.14%. Wyniki oscylują między 55% a 75%, co w niektórych przypadkach oznacza zrównanie się dokładnością z klasyfikatorem Naiwnym Bayesa.

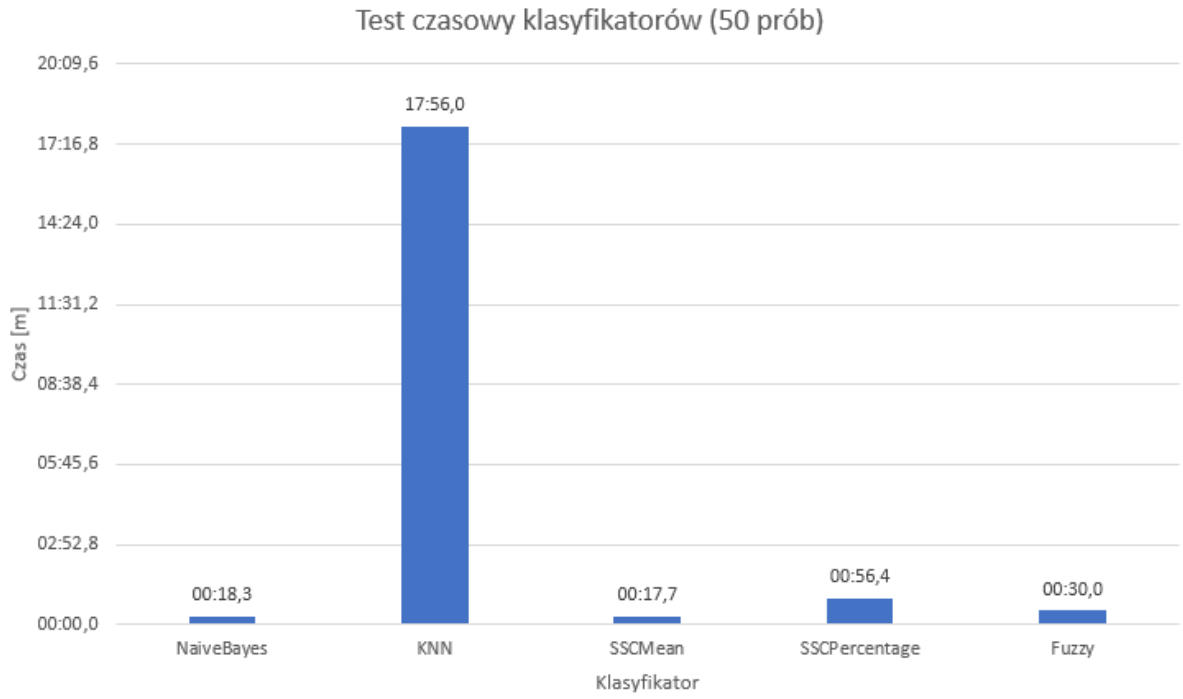


Rysunek 44: Przykładowa tablica pomyłek nr 1

- True Positive (Actual: True, Predicted: True)
- False Positive (Actual: False, Predicted: True)
- True Negative (Actual: False, Predicted: False)
- False Negative (Actual: True, Predicted: False)

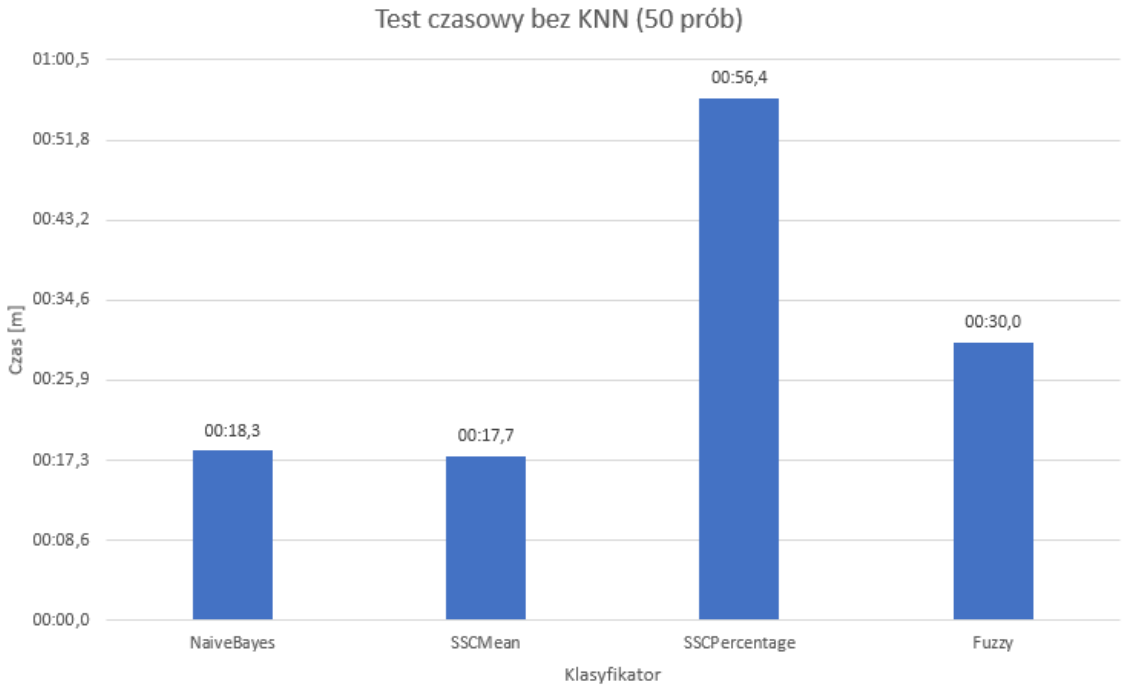


Rysunek 45: Przykładowa tablica pomyłek nr 2



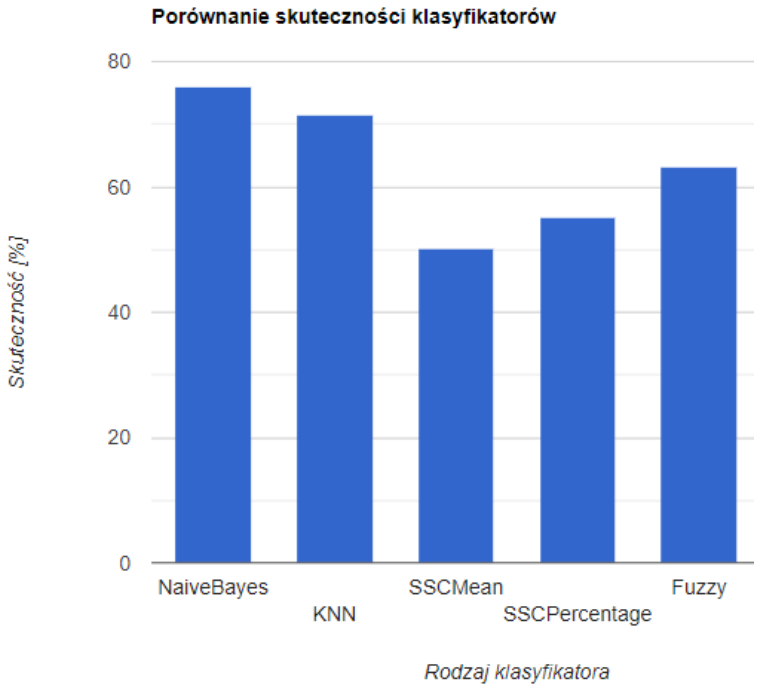
Rysunek 46: Podsumowanie czasowe klasyfikatorów

Czas był mierzony za pomocą biblioteki *datetime* i metody *datetime.now()*. Klasyfikator KNN pod względem szybkości obliczeniowej nie może konkurować z pozostałymi testowanymi przez nas klasyfikatorami. 50 prób klasyfikacji zajęło klasyfikatorowi KNN dokładnie 17 min 56 sek.



Rysunek 47: Podsumowanie czasowe klasyfikatorów bez KNN

Dodatkowy wykres podsumowania szybkości klasyfikatorów bez KNN. Najlepszy wynik osiągnął klasyfikator SSCMean - 17.68 sek. Zaraz za nim klasyfikator Naiwny Bayesa z wynikiem 18.25 sek. Klasyfikator zbiorów rozmytych osiągnął wynik 30.00 sek. Przedostatni, zaraz za KNN procentowy klasyfikator zbiorów miękkich - 56.35 sek.



Rysunek 48: Podsumowanie ogólne

Wyjaśnienie:

- **NaiveBayes**: wybrany został najkorzystniejszy przypadek, czyli dla danych znormalizowanych minmax.
- **KNN**: wybrany został najkorzystniejszy przypadek, czyli dla danych znormalizowanych minmax.
- **SoftSetClassifierMean**: wybrany został najkorzystniejszy przypadek, czyli dla danych nieznormalizowanych.
- **SoftSetClassifierPercentage**: wybrany został najkorzystniejszy przypadek, czyli dla danych znormalizowanych std.

3 Pełen kod aplikacji

3.1 Klasa ProcessingData

```
# Klasa statyczna zawierajaca metody przydatne do przetwarzania danych
class ProcessingData:

    # metoda tasujaca wiersze pd.DataFrame
    @staticmethod
    def shuffle(df: pd.DataFrame) -> pd.DataFrame:
        """
        Receives a DataFrame, randomly shuffles it and returns a
            shuffled version.
        :param df: pd.DataFrame
        :return df: pd.DataFrame
        """
        # kazdy element od konca do poczatku
        for idx in range(len(df) - 1, 0, -1):
            # wybieramy losowo 'rand_idx', czyli nowe miejsce elementu
            rand_idx = random.randint(0, idx)
            # zamieniamy
            df.iloc[idx], df.iloc[rand_idx] = df.iloc[rand_idx], df.iloc
                [idx]
        # zwracamy potasowany zbior ze zresetowanymi indeksami
        return df.reset_index(drop=True)

    # metoda normalizujaca kolumny pd.DataFrame za pomoca normalizacji z
        uzyciem odchylenia standardowego
    @staticmethod
    def normalize_std(df: pd.DataFrame, label: str) -> pd.DataFrame:
        """
        Receives a DataFrame and a class label to skip during
            normalization, normalizes data using std normalization and
            returns the normalized version.
        :param label: str
        :param df: pd.DataFrame
        :return df: pd.DataFrame
        """
        # usuwamy kolumne z klasami
        normalized = df.drop(labels=label, axis=1)
        # normalizujemy wg wzoru z odchyleniem standardowym i zwracamy
            zbior razem z wczesniej usunieta kolumna
        normalized = (normalized-normalized.mean())/(normalized.std())
        return normalized.join(df[label])

    # metoda normalizujaca kolumny pd.DataFrame za pomoca normalizacji
       minmax
    @staticmethod
    def normalize_minmax(df: pd.DataFrame, label: str) -> pd.DataFrame:
        """
        Receives a DataFrame and a class label to skip during
            normalization, normalizes data using min-max normalization
            and returns the normalized version.
        :param label: str
```

```
42          :param df: pd.DataFrame
43          :return df: pd.DataFrame
44          """
45          # usuwamy kolumne z klasami
46          normalized = df.drop(labels=label, axis=1)
47          # normalizujemy wg wzoru z 'min' i 'max' i zwracamy zbior razem
                z wczesniej usunieta kolumna
48          normalized = (normalized-normalized.min())/(normalized.max()-
                normalized.min())
49          return normalized.join(df[label])
50
51      # metoda dzielaca pd.DataFrame na zbior treningowy oraz walidacyjny
52      @staticmethod
53      def split(df: pd.DataFrame, ratio: float) -> tuple[pd.DataFrame, pd.
            DataFrame]:
54          """
55          Receives a DataFrame and returns two dataframes, split into
                training and validation DataFrames.
56          :param df: pd.DataFrame
57          :param ratio: float
58          :return training_df, validation_df: tuple[pd.DataFrame, pd.
                DataFrame]
59          """
60          # miejsce elementu dzielacego zbior w odpowiedniej proporcji wg
                'ratio'
61          cutoff = int(len(df) * ratio)
62          # zwracamy zbior treningowy i walidacyjny
63          return df[:cutoff].reset_index(drop=True), df[cutoff:].
                reset_index(drop=True)
```

3.2 Klasa NaiveBayes

```
1 # Klasa NaiveBayes zawierajaca klasyfikator Bayesa oraz metody
      pomocnicze
2 from functools import reduce
3
4
5 class NaiveBayes:
6     """
7     labels - stores all the labels existing within the column that we
          want to predict
8     values - stores the mean and standard deviation values of all the
          columns and labels into the class attribute values
9     """
10    # labels - unikalne wartosci z podanej kolumny, ktora chcemy
          przewidziec (czyli u nas 0 i 1)
11    labels: list[float] = []
12    values: list[list[list[float, float]]] = []
13
14    # metoda pomocnicza liczaca srednia wartosc w kolumnie
15    @staticmethod
16    def mean(column: list[float]) -> float:
17        """
18        Receives a column of the dataframe with numerical values and
              returns its mean value
19        :param column: list[float]
20        :return mean value of the values in a dataframe column: float
21        """
22        # czyli suma wszystkich wartosci w kolumnie przez dlugosc
              kolumny
23        return sum(column) / len(column)
24
25    # metoda pomocnicza liczaca odchylenie standardowe wartosci w
          kolumnie
26    @staticmethod
27    def std_dev(column: list[float], m: float) -> float:
28        """
29        Receives a column of the dataframe and its mean value and
              returns its standard deviation.
30        :param column: list[float]
31        :param m: float
32        :return standard deviation of the values in a dataframe column:
              float
33        """
34        return math.sqrt(sum(map(lambda x: (x - m) ** 2, column)) / len(
              column))
35
36    # metoda pomocnicza obliczajaca gestosc prawdopodobienstwa
37    @staticmethod
38    def density(x: float, m: float, sd: float) -> float:
39        """
40        Receives a value, plus a mean value and a standard deviation of
              the values in a certain column, and returns its Gaussian
              probability.
41        :param x: float
```

```
42          :param m: float
43          :param sd: float
44          :return Gaussian probability: float
45          """
46          # rozklad normalny / Gaussa
47          return math.exp(-(x - m) ** 2 / (2 * sd ** 2)) / (math.sqrt(2 *
                math.pi) * sd)
48
49      # metoda obliczajaca wartosci przechowywane w tablicach pomocniczych
            klasy
50      def calculate(self, df: pd.DataFrame, label: str) -> None:
51          """
52          Receives a dataframe and the label of the column we want to
                predict and loads mean and standard deviation values into the
                 class attribute values.
53          :param df: pd.DataFrame
54          :param label: str
55          :return: None
56          """
57          # lista wartosci oraz lista unikalnych wartosci z podanej
                kolumny 'label'
58          self.values = []
59          self.labels = df[label].unique()
60
61          # dla kazdej unikalnej wartosci z przewidywanej kolumny (czyli u
                nas dla 0 i 1)
62          for category in self.labels:
63              # wartosci ktorych unikalna wartosc kolumny 'label' to '
                    category', czyli podzial na czesci ktore maja 0 i 1
64              category_values = df[df[label] == category].drop(labels=
                    label, axis=1)
65              # pary: [srednia, odchylenie std] dla kazdej kolumny dla
                    kazdej kategorii 'category_values'
66              column = [[self.mean(item), self.std_dev(item, self.mean(
                    item))] for _, item in category_values.items()
67              # dodanie do 'values' wszystkich par [srednia, odchylenie
                    std]
68              self.values.append(column)
69
70      # metoda przewidujaca etykiety klas probek
71      def predict(self, sample: pd.Series, label: str) -> float:
72          """
73          Receives a sample in the form of pd.Series and returns its
                predicted label.
74          :param sample: pd.Series
75          :param label: str - column with class labels
76          :return predicted label of the sample: str
77          """
78          sample = sample.drop(labels=label)
79          probabilities = []
80          # dla kazdej kategorii w 'values'
81          for category in self.values:
82              category_pairs = []
83              # pair to para [srednia, odchylenie]
84              # x to odpowiednia wartosc w 'sample'
```

```
85              for pair, x in zip(category, sample):
86                  # dodajemy kolejne obliczone Gaussy
87                  category_pairs.append(self.density(x, pair[0], pair[1]))
88              # liczymy prawdopodobienstwa czyli wymnazamy ze soba kolejne
                    odpowiednie Gaussy
89              probabilities.append(reduce(lambda z, y: z * y,
                    category_pairs))
90          # naszym indeksem jest indeks najwiekszego prawdopodobienstwa
91          idx = max(range(len(probabilities)), key=probabilities.
                __getitem__)
92          # zwracamy przewidywany wynik z 'labels' (u nas 0 lub 1)
93          return self.labels[idx]
94
95      # metoda testujaca dokladnosc przewidywan
96      def test(self, validation_set: pd.DataFrame, label: str) -> None:
97          """
98          Test how accurate the prediction method is
99          :param validation_set: pd.DataFrame - a record from the
                validation dataset
100         :param label: str - column with class labels
101         :return: None
102         """
103         correct = 0
104         # iterrows() jest uzywane do iteracji wierszy dataframe'a
105         for _, row in validation_set.iterrows():
106             # jesli poprawnie przewidzimy etykiete (czyli 0 lub 1)
107             if self.predict(sample=row, label=label) == row[label]:
108                 correct += 1
109         # informacja podsumowujaca
110         print(f'Accuracy: {correct / len(validation_set) * 100:.2f}%; {
                correct}/{len(validation_set)}')
```

3.3 Klasa KNN

```
# Klasa statyczna KNN zawierajaca klasyfikator KNN oraz metody pomocnicze
from collections import Counter

class KNN:

    # metoda obliczajaca odleglosc miedzy probkami za pomoca metryki
    @staticmethod
    def distance(x: pd.Series, y: pd.Series, m: int, label: str) -> float:
        """
        :param x: pd.Series - a record from a dataset to calculate the distance
        :param y: pd.Series - a record from a dataset to calculate the distance
        :param m: int - to calculate the mth root in Euclidean metric
        :param label: str - column with class labels
        :return: float - the distance between two records
        """
        s = 0
        x = x.drop(labels=label)
        y = y.drop(labels=label)
        # odleglosc miedzy dwoma wektorami, czyli pierwiastek sumy wartosci bezwzglednych kwadratow roznic kolejnych wspolrzednych
        for xi, yi in zip(x, y):
            s += pow(abs(xi - yi), m)
        return pow(s, 1 / m)

    # metoda przewidujaca etykiete klasy probki na podstawie etykiet jej k-najblizszych sasiadow
    @staticmethod
    def cluster(sample: pd.Series, training_set: pd.DataFrame, k: int, m: int, label: str) -> str:
        """
        :param sample: pd.Series - a record from the validation dataset
        :param training_set: pd.DataFrame - training dataset
        :param k: int - k-nearest elements
        :param m: int - to pass into KNN.distance
        :param label: str - name of the label that we want to predict
        :return: str - predicted label of the object
        """
        # pary (dystans, outcome) dla zbioru treningowego
        distances = [(KNN.distance(sample, training_set.iloc[idx], m, label), training_set.iloc[idx][label]) for idx in training_set.index]
        # wybieramy 'k' najblizszych wynikow
        distances = sorted(distances, key=lambda d: (d, random.random()))[:k]
        # zwracamy etykiete, ktora wytypowalismy najwiecej razy
        return max(dict(Counter(elem[1] for elem in distances)))

    # metoda testujaca dokladnosc dzialania
```

```
43      @staticmethod
44      def test(training_set: pd.DataFrame, validation_set: pd.DataFrame,
            k0: int, kn: int, m: int, label: str) -> None:
45          """
46          :param training_set: pd.Series - a record from the validation
                dataset
47          :param validation_set: pd.DataFrame - training dataset
48          :param k0: int - starting value of k to pass into KNN.cluster
49          :param kn: int - end value of k to pass into KNN.cluster
50          :param m: int - to pass into KNN.cluster
51          :param label: str - name of the label that we want to predict
52          :return: None
53          """
54          # testujemy dla 'k' od 'k0' do 'kn'
55          for k in range(k0, kn + 1):
56              correct = 0
57              # bierzemy kazda probkie (sample) ze zbioru walidacyjnego
58              for sample in validation_set.iloc:
59                  x = KNN.cluster(sample, training_set, k, m, label)
60                  # sprawdzamy czy przewidziany jest taki jak rzeczywisty
61                  if x == sample[label]:
62                      correct += 1
63              # informacja podsumowujaca
64              print(f'Accuracy for {k=}: {float(correct) / len(
                    validation_set) * 100:.2f}%, correct predictions: {
                    correct}/{len(validation_set)}')
```

3.4 Klasa SoftSetClassifierMean

```
# Klasa SoftSetClassifierMean zawierajaca klasyfikator miekki korzystajacy ze sredniej arytmetycznej oraz metody pomocnicze
class SoftSetClassifierMean:
    """
    labels - stores all the labels existing within the column that we want to predict
    pairs - stores 0 and 1 pairs for every label for every column
    means - stores mean values for every label for every column
    """
    # lista etykiet
    labels: list[str] = []
    # lista par 0, 1 dla kazdej kolumny i kazdej cechy
    pairs: list[list[list[int, int]]] = []
    # lista srednich dla kazdej kolumny i kazdej cechy
    means: list[list[float]] = []

    # metoda pomocnicza obliczajaca srednie kolumn
    @staticmethod
    def mean(column: list[float]) -> float:
        """
        Receives a column of the dataframe with numerical values and returns its mean value
        :param column: list[float]
        :return mean value of the values in a dataframe column: float
        """
        # czyli zwykla suma przez ilosc skladnikow
        return sum(column) / len(column)

    # metoda tworzaca tabele wazona zbioru miekkiego
    def calculate(self, df: pd.DataFrame, label: str) -> None:
        """
        Receives a dataframe and the label of the column we want to predict and decides whether pairs are [0,1] or [1,0]
        :param df: pd.DataFrame
        :param label: str
        :return: None
        """
        self.pairs = []
        self.means = []
        # lista unikalnych wartosci z podanej kolumny 'label'
        # ma byc zawsze ta sama kolejnosc wiec sortujemy
        self.labels = sorted(df[label].unique())

        # 'category' to 0 i 1
        for category in self.labels:
            # wartosci ktorych unikalna wartosc kolumny 'label' to 'category'
            # czyli podzial na czesci ktore maja 0 i 1
            category_values = df[df[label] == category].drop(labels=label, axis=1)
            temp_mean = []
            temp_pair = []
            # item to wszystkie wartosci z danej kolumny
```

```
48              for _, item in category_values.items():
49                  # srednia z kolumny, dodajemy ja do listy srednich
50                  mean = self.mean(item)
51                  temp_mean.append(mean)
52                  len_lower = 0
53                  len_upper = 0
54                  # zliczamy elementy wieksze od sredniej i mniejsze od
                        sredniej
55                  for value in item:
56                      if value > mean:
57                          len_upper += 1
58                      else:
59                          len_lower += 1
60
61                  # jesli jest tyle samo powyzej sredniej co ponizej to
62                  # wybieramy losowo pare [0, 1] lub [1, 0]
63                  if len_upper == len_lower:
64                      temp_pair.append(random.choice(([0, 1],[1, 0])))
65                  # jesli wiecej powyzej sredniej to zapisujemy pare [1,
                        0]
66                  elif len_upper > len_lower:
67                      temp_pair.append([1, 0])
68                  # jesli wiecej ponizej sredniej to zapisujemy pare [0,
                        1]
69                  else:
70                      temp_pair.append([0, 1])
71
72              # uzupelniamy listy wszystkich 'srednich' i wszystkich 'par'
73              self.means.append(temp_mean)
74              self.pairs.append(temp_pair)
75
76      # metoda przewidujaca etykiete klasy probki
77      def predict(self, sample: pd.Series, label: str) -> str:
78          """
79          Receives a sample in the form of pd.Series and returns its
                predicted label
80          :param sample: pd.Series
81          :param label: str
82          :return predicted label of the sample: str
83          """
84          # liczenie elementow dla probki (sample) dla kazdej z klas (0,
                1)
85
86          sample = sample.drop(labels=label)
87          probabilities = []
88
89          # bierzemy po kolei odpowiednie pary i srednie
90          for category, mean in zip(self.pairs, self.means):
91              sample_pairs = []
92              # bierzemy odpowiednia srednia i wartosc z probki (sample)
93              for m, x in zip(mean, sample):
94                  # jesli element rowny sredniej
95                  # wybieramy losowo pare [0, 1] lub [1, 0]
96                  if x == m:
97                      sample_pairs.append(random.choice(([0, 1],[1, 0])))
```

```
98                  # jesli element wiekszy od sredniej to zapisujemy pare
                        [1, 0]
99                  elif x > m:
100                     sample_pairs.append([1, 0])
101                 # jesli element mniejszy od sredniej to zapisujemy pare
                        [0, 1]
102                 else:
103                     sample_pairs.append([0, 1])
104             temp = []
105
106             # liczenie prawdopodobienstwa dla kazdej klasy
107             for x, y in zip(category, sample_pairs):
108                 temp.append(x[0] * y[0] + x[1] * y[1])
109             probabilities.append(sum(temp))
110
111         # naszym indeksem jest indeks najwiekszego prawdopodobienstwa
112         idx = max(range(len(probabilities)), key=probabilities.
                __getitem__)
113         # zwracamy najprawdopodobniejszy wynik z 'labels' (u nas 0 lub
                1)
114         return self.labels[idx]
115
116     # metoda testujaca dokladnosc dzialania Klasyfikatora miekkiego
117     def test(self, validation_set: pd.DataFrame, label: str) -> None:
118         """
119         Test how accurate the prediction method is
120         :param validation_set: pd.DataFrame - a record from the
                validation dataset
121         :param label: str
122         :return: None
123         """
124         correct = 0
125         # iterrows() jest uzywane do iteracji wieszy dataframe'a
126         for _, row in validation_set.iterrows():
127             # jesli poprawnie przewidzimy etykiete
128             if self.predict(sample=row, label=label) == row[label]:
129                 correct += 1
130         # informacja podsumowujaca
131         print(f'Accuracy: {correct / len(validation_set) * 100:.2f}%; {
                correct}/{len(validation_set)}')
```

3.5 Klasa SoftSetClassifierPercentage

```
# Klasa SoftSetClassifierPercentage zawierajaca klasyfikator miekki korzystajacy z procentow oraz metody pomocnicze
class SoftSetClassifierPercentage:
    """
    labels - stores all the labels existing within the column that we want to predict
    pairs - stores pairs of values between 0 and 1 for every label for every column
    minmaxs - stores min and max pairs of values for every label for every column
    """
    # lista etykiet
    labels: list[str] = []
    # lista par 0, 1 dla kazdej kolumny i kazdej cechy
    pairs: list[list[list[int, int]]] = []
    # lista minow i maxow dla kazdej kolumny i kazdej cechy
    minmaxs: list[list[list[float, float]]] = []

    # metoda pomocnicza obliczajaca srednie kolumn
    @staticmethod
    def mean(column: list[float]) -> float:
        """
        Receives a column of the dataframe with numerical values and returns its mean value
        :param column: list[float]
        :return mean value of the values in a dataframe column: float
        """
        # czyli suma przez ilosc skladnikow
        return sum(column) / len(column)

    # metoda tworzaca tabele wazona zbioru miekkiego
    def calculate(self, df: pd.DataFrame, label: str) -> None:
        """
        Receives a dataframe and the label of the column we want to predict and decides whether pairs are [0,1] or [1,0]
        :param df: pd.DataFrame
        :param label: str
        :return: None
        """
        self.pairs = []
        self.minmaxs = []
        # lista unikalnych wartosci z podanej kolumny 'label'
        # ma byc zawsze ta sama kolejnosc wiec sortujemy
        self.labels = sorted(df[label].unique())

        # 'category' to 0 i 1
        for category in self.labels:
            # wartosci ktorych unikalna wartosc kolumny 'label' to 'category'
            # czyli podzial na czesci ktore maja 0 i 1
            category_values = df[df[label] == category].drop(labels=label, axis=1)
            temp_minmax = []
```

```
46              temp_pair = []
47              # item to wszystkie wartosci z danej kolumny
48              for _, item in category_values.items():
49                  # minimum z kolumny
50                  min_t = min(item)
51                  # maximum z kolumny
52                  max_t = max(item)
53                  temp_minmax.append([min_t, max_t])
54                  # srednia
55                  mean = self.mean(item)
56                  len_lower = 0
57                  len_upper = 0
58                  # zliczamy elementy wieksze od sredniej i mniejsze od
                        sredniej
59                  for value in item:
60                      if value > mean:
61                          len_upper += 1
62                      else:
63                          len_lower += 1
64
65                  # jesli jest tyle samo powyzej srendiej co ponizej to
66                  # wybieramy losowo pare [0, 1] lub [1, 0]
67                  if len_upper == len_lower:
68                      temp_pair.append(random.choice(([0, 1],[1, 0])))
69                  # jesli wiecej powyzej sredniej to zapisujemy pare [1,
                        0]
70                  elif len_upper > len_lower:
71                      temp_pair.append([1, 0])
72                  # jesli wiecej ponizej sredniej to zapisujemy pare [0,
                        1]
73                  else:
74                      temp_pair.append([0, 1])
75              # uzupelniamy listy wszystkich 'minmaxow' i wszystkich 'par'
76              self.minmaxs.append(temp_minmax)
77              self.pairs.append(temp_pair)
78
79      # metoda przewidujaca etykiete klasy probki
80      def predict(self, sample: pd.Series, label: str) -> str:
81          """
82          Receives a sample in the form of pd.Series and returns its
                predicted label
83          :param sample: pd.Series
84          :param label: str
85          :return predicted label of the sample: str
86          """
87          # liczenie elementow dla probki (sample) dla kazdej z klas (0,
                1)
88
89          sample = sample.drop(labels=label)
90          probabilities = []
91
92          # bierzemy po kolei odpowiednie pary i srednie
93          for category, minmax in zip(self.pairs, self.minmaxs):
94              sample_pairs = []
95              # bierzemy odpowiednia pare minmax i wartosc z probki (
```

```
                sample)
96              for pair, x in zip(minmax, sample):
97                  # proportion = (wartosc_sampla - min) / (max - min)
98                  proportion = (x - pair[0]) / (pair[1] - pair[0])
99                  # dodajemy liste [1 - prop, prop] do listy wszytskich
                        par
100                 sample_pairs.append([1 - proportion, proportion])
101
102             temp = []
103             # liczenie prawdopodobienstwa dla kazdej klasy
104             for x, y in zip(category, sample_pairs):
105                 temp.append(x[0] * y[0] + x[1] * y[1])
106             probabilities.append(sum(temp))
107         # naszym indeksem jest indeks najwiekszego prawdopodobienstwa
108         idx = max(range(len(probabilities)), key=probabilities.
                __getitem__)
109         # zwracamy najprawdopodobniejszy wynik z 'labels' (u nas 0 lub
                1)
110         return self.labels[idx]
111
112     # metoda testujaca dokladnosc dzialania Klasyfikatora miekkiego
113     def test(self, validation_set: pd.DataFrame, label: str) -> None:
114         """
115         Test how accurate the prediction method is
116         :param validation_set: pd.DataFrame - a record from the
                validation dataset
117         :param label: str
118         :return: None
119         """
120         correct = 0
121         # iterrows() jest uzywane do iteracji wieszy dataframe'a
122         for _, row in validation_set.iterrows():
123             # jesli poprawnie przewidzimy etykiete
124             if self.predict(sample=row, label=label) == row[label]:
125                 correct += 1
126
127         # informacja podsumowujaca
128         print(f'Accuracy: {correct / len(validation_set) * 100:.2f}%; {
                correct}/{len(validation_set)}')
```

3.6 Klasa FuzzyClassifier

```
class Fuzzy:
    def __init__(self):
        # slownik slownikow rozmycia dla kazdej kolumny
        self.antecedents: dict[str: dict[str: tuple[float]]] = {}
        # slownik zawierajaca nazwe etykiety klasy oraz slownikow z jej rozmyciem
        self.consequent: dict[str: dict[str: tuple[float]]] = {}
        # lista slownikow; kazda regula to jeden slownik
        self.rules: list[dict[str: str]] = []

    # metoda pozwalajaca na dodanie poprzednika
    def add_antecedent(self, parameter: str, linguistic_value: str, *args) -> None:
        if parameter not in self.antecedents:
            self.antecedents[parameter] = {}
        self.antecedents[parameter][linguistic_value] = args

    # metoda pozwalajaca na dodanie konsekwencji
    def add_consequent(self, parameter: str, linguistic_value: str, *args) -> None:
        if parameter not in self.consequent:
            self.consequent[parameter] = {}
        self.consequent[parameter][linguistic_value] = args

    # metoda pozwalajaca dodac regule
    def add_rule(self, rule: dict[str: str]) -> None:
        assert set(rule.keys()) == set.union(set(self.antecedents.keys()), set(self.consequent.keys())), print("Nieprawidlowa liczba kategorii w regule.")
        self.rules.append(rule)

    # metoda obliczajaca przynaleznosc dla funkcji typu trojkat
    @staticmethod
    def triangular_function(n: float, a: float, b: float, c: float) -> float:
        assert a <= b <= c, print("Nieprawidlowe wartosci")
        if n == a == b or n == b == c:
            return 1
        if n <= a:
            return 0
        if a < n < b:
            return (n - a) / (b - a)
        if n == b:
            return 1
        if b < n < c:
            return (c - n) / (c - b)
        if n >= c:
            return 0

    # metoda obliczajaca przynaleznosc dla funkcji typu trapez
    @staticmethod
    def trapezoidal_function(n: float, a: float, b: float, c: float, d: float) -> float:
```

```
47          assert a <= b <= c <= d, print("Nieprawidlowe wartosci")
48          if c == d and n >= d:
49              return 1
50          if a == b and n <= a:
51              return 1
52          if n <= a:
53              return 0
54          if a < n < b:
55              return (n - a) / (b - a)
56          if b <= n <= c:
57              return 1
58          if c < n < d:
59              return (d - n) / (d - c)
60          if n >= d:
61              return 0
62
63      # metoda wywolujaca funkcje przynaleznosci na podstawie liczby
            argumentow(dla 3: funkcje typu trojkat,  a dla 4: funkcja typu
            trapez)
64      def membership_function(self, n: float, *args) -> float:
65          assert len(args) in (3, 4), print("Nieprawidlowa liczba
                argumentow")
66          if len(args) == 3:
67              return self.triangular_function(n, *args)
68          else:
69              return self.trapezoidal_function(n, *args)
70
71      # metoda rozmywajaca dane wejsciowe
72      def fuzzify(self, column: str, n: float) -> dict[str, float]:
73          return {antecedent: self.membership_function(n, *self.
                antecedents[column][antecedent]) for antecedent in self.
                antecedents[column].keys()}
74
75      # metoda obliczajaca stopien spelnienia danej zasady
76      @staticmethod
77      def rule_fulfillment(rule: dict[str: str], fuzzy_values: dict[str:
            dict[str: float]]) -> tuple[str, float]:
78          label = list(rule.keys())[-1]
79          minimum = float('inf')
80          # dla kazdej wartosci w danej zasadzie
81          for name, linguistic_value in list(rule.items())[:-1]:
82              maximum = float('-inf')
83              # jesli mamy wartosci rozdzielone 'or', to bierzemy ich
                    maksimum
84              for lvs in linguistic_value.split(' | '):
85                  maximum = max(fuzzy_values[name][lvs], maximum)
86              # bierzemy minimum stopni spelnienia wartosci danej zasady
87              minimum = min(minimum, maximum)
88          return rule[label], minimum
89
90      # metoda obliczajaca powierzchnie danego wyniku
91      @staticmethod
92      def area(a: float, c: float) -> float:
93          return (c - a) / 2
94
```

```
95      # metoda srodka ciezkosci
96      @staticmethod
97      def cog(a: float, b: float, c: float) -> float:
98          return (a + b + c) / 3
99
100     # metoda agregujaca, wyostrzajaca wartosc koncowa za pomoca metody
            srodka ciezkosci
101     def aggregate(self, label: str, outputs: defaultdict) -> float:
102         # licznik i mianownik dla funkcji agregujacej
103         counter = []
104         denominator = []
105         # obliczanie powierzchni oraz srodka ciezkosci kazdej etykiety
                funkcji przynaleznosci
106         for name, membership in outputs.items():
107             if not membership:
108                 continue
109             a, b, c = self.consequent[label][name]
110             area = self.area(a, c)
111             cog = self.cog(a, b, c)
112             counter.append(membership * area * cog)
113             denominator.append(membership * area)
114         return round(sum(counter) / sum(denominator), 3)
115
116     # metoda przewidujaca etykiete klasy
117     def compute(self, sample: pd.Series) -> tuple[str, float]:
118         # rozmycie probki
119         fuzzy_values = {column: self.fuzzify(column, sample[column]) for
                column in self.antecedents.keys()}
120
121         # obliczenie stopnia spelnienia zasad
122         fulfillments = [self.rule_fulfillment(rule, fuzzy_values) for
                rule in self.rules]
123
124         # wybranie maksymalnego spelnienia reguly dla danych wartosci
125         outputs: defaultdict = defaultdict(lambda: 0)
126         for linguistic_value, fulfillment in fulfillments:
127             outputs[linguistic_value] = max(outputs[linguistic_value],
                    fulfillment)
128         outputs = dict(outputs)
129         label = list(self.consequent.keys())[0]
130         # wybor przewidzianej etykiety klasy: w przypadku remisu dla
                dwoch etykiet wybieramy chorobe
131         output_names = sorted(outputs.keys(), key=lambda x: (-1 *
                outputs[x], x))
132         # print(sample['Outcome'], self.aggregate(label, outputs),
                outputs)
133         return output_names[0], self.aggregate(label, outputs)
134
135     # metoda pomocnicza wyswietlajaca wykres rozmycia cechy
136     def view(self, parameter: str) -> None:
137         # lista kolorow do wykresow
138         colors: dict[int: tuple] = {2: ('#00b100', '#ff5c38'),
139                                     3: ('#00b100', '#d5c731', '#ff5c38')
                                            ,
140                                     4: ('#00b100', '#adcd27', '#f1b438',
```

```
                                        '#ff5c38'),
141                                 5: ('#00b100', '#93cb20', '#d5c731',
                                        '#f9a539', '#ff5c38')}
142         # globalne parametry dla wykresow matplotlib.pyplot
143         plt.rcParams["figure.figsize"] = (14, 8)
144         if parameter in self.consequent.keys():
145             source = self.consequent[parameter]
146         else:
147             source = self.antecedents[parameter]
148 
149         # stworzenie dziedziny
150         minimum: int = floor(list(source.values())[0][0])
151         maximum: int = ceil(list(source.values())[-1][-1])
152         x = np.linspace(minimum, maximum, num=(maximum - minimum) * 100
                + 1)
153         # stworzenie wykresow funkcji przynaleznosci
154         for k, v, color in zip(source.keys(), source.values(), colors[
                len(source.keys())]):
155             y = np.array([self.membership_function(i, *v) for i in x])
156             plt.plot(x, y, color, label=k)
157 
158         # nadanie parametrow wykresu
159         plt.rc('font', size=16)
160         plt.rc('axes', titlesize=14)
161         plt.rc('axes', labelsize=14)
162         plt.rc('xtick', labelsize=14)
163         plt.rc('ytick', labelsize=14)
164         plt.rc('legend', fontsize=14)
165         plt.rc('figure', titlesize=16)
166         plt.title(parameter)
167         plt.xlabel('Membership')
168         plt.ylabel(parameter)
169         plt.legend(loc="upper right")
170         plt.ylim(0, 1)
171         plt.show()
```

3.7 Dodawanie poprzedników i konsekwencji

```
1   # Dodawanie poprzednikow i konsekwencji
2         fuzzy = Fuzzy()
3 
4         fuzzy.add_antecedent('Pregnancies', 'low', 0, 0, 5)
5         fuzzy.add_antecedent('Pregnancies', 'medium', 0, 5, 10)
6         fuzzy.add_antecedent('Pregnancies', 'high', 5, 10, 17, 17)
7 
8         fuzzy.add_antecedent('Glucose', 'low', 44, 44, 86, 99)
9         fuzzy.add_antecedent('Glucose', 'medium_low', 86, 99, 112)
10        fuzzy.add_antecedent('Glucose', 'medium', 99, 112, 125)
11        fuzzy.add_antecedent('Glucose', 'medium_high', 112, 125, 138)
12        fuzzy.add_antecedent('Glucose', 'high', 125, 138, 199, 199)
13 
14        fuzzy.add_antecedent('BloodPressure', 'low', 24, 24, 75, 80)
15        fuzzy.add_antecedent('BloodPressure', 'medium_low', 75, 80, 85)
```

```
16          fuzzy.add_antecedent('BloodPressure', 'medium', 80, 85, 90)
17          fuzzy.add_antecedent('BloodPressure', 'medium_high', 85, 90, 95)
18          fuzzy.add_antecedent('BloodPressure', 'high', 90, 95, 122, 122)
19
20          fuzzy.add_antecedent('SkinThickness', 'low', 7, 7, 15, 22)
21          fuzzy.add_antecedent('SkinThickness', 'medium_low', 15, 22, 29)
22          fuzzy.add_antecedent('SkinThickness', 'medium', 22, 29, 36)
23          fuzzy.add_antecedent('SkinThickness', 'medium_high', 29, 36, 43)
24          fuzzy.add_antecedent('SkinThickness', 'high', 36, 43, 99, 99)
25
26          fuzzy.add_antecedent('Insulin', 'low', 14, 14, 76, 125)
27          fuzzy.add_antecedent('Insulin', 'medium', 76, 125, 190)
28          fuzzy.add_antecedent('Insulin', 'high', 125, 190, 846, 846)
29
30          fuzzy.add_antecedent('BMI', 'underweight', 0, 0, 16, 22)
31          fuzzy.add_antecedent('BMI', 'healthy_weight', 16, 22, 28)
32          fuzzy.add_antecedent('BMI', 'overweight', 22, 28, 34)
33          fuzzy.add_antecedent('BMI', 'obese', 28, 34, 67.1, 67.1)
34
35          fuzzy.add_antecedent('DiabetesPedigreeFunction', 'low', 0, 0,
               0.25, 0.5)
36          fuzzy.add_antecedent('DiabetesPedigreeFunction', 'medium', 0.25,
                0.5, 0.75)
37          fuzzy.add_antecedent('DiabetesPedigreeFunction', 'high', 0.5,
               0.75, 2.42, 2.42)
38
39          fuzzy.add_antecedent('Age', 'low', 21, 21, 34, 45)
40          fuzzy.add_antecedent('Age', 'medium', 34, 45, 56)
41          fuzzy.add_antecedent('Age', 'high', 45, 56, 81, 81)
42
43          for elem in fuzzy.antecedents:
44              fuzzy.view(elem)
45
46          fuzzy.add_consequent('Outcome', 'low', 0, 0, 1)
47          fuzzy.add_consequent('Outcome', 'high', 0, 1, 1)
48          fuzzy.view('Outcome')
```

3.8 Dodawanie reguł

```
1           # Dodawanie regul
2           # Wiekszosc niskich
3           # Same niskie (1) 1
4           fuzzy.add_rule({'Pregnancies': 'low | medium',
5                           'Glucose': 'low | medium_low | medium',
6                           'BloodPressure': 'low | medium_low | medium',
7                           'SkinThickness': 'low | medium_low | medium',
8                           'Insulin': 'low | medium',
9                           'BMI': 'underweight | healthy_weight',
10                          'DiabetesPedigreeFunction': 'low | medium',
11                          'Age': 'low | medium',
12                          'Outcome': 'low'})
13          # Pojedyncze wysokie (8) 9
14          fuzzy.add_rule({'Pregnancies': 'medium | high',
```

```
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'low | medium',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
                    'Outcome': 'low'})
fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'low | medium',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
                    'Outcome': 'low'})
fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'low | medium',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
                    'Outcome': 'low'})
fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'low | medium',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
                    'Outcome': 'low'})
fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
                    'Outcome': 'low'})
fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'low | medium',
                    'BMI': 'obese | overweight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
                    'Outcome': 'low'})
fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
```

```
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'low'})
    # Pary Pregnancies (7) 16
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
```

```
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'low'})
        # Pary Glucose (6) 22
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
```

```
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'low'})
        # Pary BloodPressure (5) 27
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
```

```
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'low'})
        # Pary SkinThickness (4) 31
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'low'})
        # Pary Insulin (3) 34
        fuzzy.add_rule({'Pregnancies': 'low | medium',
```

```
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'low'})
    # Pary BMI (2) 36
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'low'})
    # Pary DiabetesPedigreeFunction (1) 37
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'low'})
```

```
345         # Trojki Pregnancies i Glucose (6) 43
346         fuzzy.add_rule({'Pregnancies': 'medium | high',
347                         'Glucose': 'medium | medium_high | high',
348                         'BloodPressure': 'medium | medium_high | high',
349                         'SkinThickness': 'low | medium_low | medium',
350                         'Insulin': 'low | medium',
351                         'BMI': 'underweight | healthy_weight',
352                         'DiabetesPedigreeFunction': 'low | medium',
353                         'Age': 'low | medium',
354                         'Outcome': 'low'})
355         fuzzy.add_rule({'Pregnancies': 'medium | high',
356                         'Glucose': 'medium | medium_high | high',
357                         'BloodPressure': 'low | medium_low | medium',
358                         'SkinThickness': 'medium | medium_high | high',
359                         'Insulin': 'low | medium',
360                         'BMI': 'underweight | healthy_weight',
361                         'DiabetesPedigreeFunction': 'low | medium',
362                         'Age': 'low | medium',
363                         'Outcome': 'low'})
364         fuzzy.add_rule({'Pregnancies': 'medium | high',
365                         'Glucose': 'medium | medium_high | high',
366                         'BloodPressure': 'low | medium_low | medium',
367                         'SkinThickness': 'low | medium_low | medium',
368                         'Insulin': 'medium | high',
369                         'BMI': 'underweight | healthy_weight',
370                         'DiabetesPedigreeFunction': 'low | medium',
371                         'Age': 'low | medium',
372                         'Outcome': 'low'})
373         fuzzy.add_rule({'Pregnancies': 'medium | high',
374                         'Glucose': 'medium | medium_high | high',
375                         'BloodPressure': 'low | medium_low | medium',
376                         'SkinThickness': 'low | medium_low | medium',
377                         'Insulin': 'low | medium',
378                         'BMI': 'overweight | obese',
379                         'DiabetesPedigreeFunction': 'low | medium',
380                         'Age': 'low | medium',
381                         'Outcome': 'low'})
382         fuzzy.add_rule({'Pregnancies': 'medium | high',
383                         'Glucose': 'medium | medium_high | high',
384                         'BloodPressure': 'low | medium_low | medium',
385                         'SkinThickness': 'low | medium_low | medium',
386                         'Insulin': 'low | medium',
387                         'BMI': 'underweight | healthy_weight',
388                         'DiabetesPedigreeFunction': 'medium | high',
389                         'Age': 'low | medium',
390                         'Outcome': 'low'})
391         fuzzy.add_rule({'Pregnancies': 'medium | high',
392                         'Glucose': 'medium | medium_high | high',
393                         'BloodPressure': 'low | medium_low | medium',
394                         'SkinThickness': 'low | medium_low | medium',
395                         'Insulin': 'low | medium',
396                         'BMI': 'underweight | healthy_weight',
397                         'DiabetesPedigreeFunction': 'low | medium',
398                         'Age': 'medium | high',
399                         'Outcome': 'low'})
```

```
        # Trojki Pregnancies i BloodPressure (5) 48
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'low'})
        # Trojki Pregnancies i SkinThickness (4) 52
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
```

```
                    'Outcome': 'low'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'low | medium',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
                    'Outcome': 'low'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'low | medium',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'low | medium',
                    'Outcome': 'low'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'low | medium',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'medium | high',
                    'Outcome': 'low'})
    # Trojki Pregnancies i Insulin (3) 55
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
                    'Outcome': 'low'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'low | medium',
                    'Outcome': 'low'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'medium | high',
```

```
                    'Outcome': 'low'})
        # Trojki Pregnancies i BMI (2) 57
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'low | medium',
                    'BMI': 'obese | overweight',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'low | medium',
                    'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'low | medium',
                    'BMI': 'obese | overweight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'medium | high',
                    'Outcome': 'low'})
        # Trojki Pregnancies i DiabetesPedigreeFunction (1) 58
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'low | medium',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'medium | high',
                    'Outcome': 'low'})
        # Trojki Glucose i BloodPressure (5) 63
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'low | medium',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
                    'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
                    'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'low | medium',
                    'BMI': 'overweight | obese',
```

```
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'low'})
        # Trojki Glucose i SkinThickness (4) 67
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
```

```
620                         'DiabetesPedigreeFunction': 'low | medium',
621                         'Age': 'medium | high',
622                         'Outcome': 'low'})
623         # Trojki Glucose i Insulin (3) 70
624         fuzzy.add_rule({'Pregnancies': 'low | medium',
625                         'Glucose': 'medium | medium_high | high',
626                         'BloodPressure': 'low | medium_low | medium',
627                         'SkinThickness': 'low | medium_low | medium',
628                         'Insulin': 'medium | high',
629                         'BMI': 'overweight | obese',
630                         'DiabetesPedigreeFunction': 'low | medium',
631                         'Age': 'low | medium',
632                         'Outcome': 'low'})
633         fuzzy.add_rule({'Pregnancies': 'low | medium',
634                         'Glucose': 'medium | medium_high | high',
635                         'BloodPressure': 'low | medium_low | medium',
636                         'SkinThickness': 'low | medium_low | medium',
637                         'Insulin': 'medium | high',
638                         'BMI': 'underweight | healthy_weight',
639                         'DiabetesPedigreeFunction': 'medium | high',
640                         'Age': 'low | medium',
641                         'Outcome': 'low'})
642         fuzzy.add_rule({'Pregnancies': 'low | medium',
643                         'Glucose': 'medium | medium_high | high',
644                         'BloodPressure': 'low | medium_low | medium',
645                         'SkinThickness': 'low | medium_low | medium',
646                         'Insulin': 'medium | high',
647                         'BMI': 'underweight | healthy_weight',
648                         'DiabetesPedigreeFunction': 'low | medium',
649                         'Age': 'medium | high',
650                         'Outcome': 'low'})
651         # Trojki Glucose i BMI (2) 72
652         fuzzy.add_rule({'Pregnancies': 'low | medium',
653                         'Glucose': 'medium | medium_high | high',
654                         'BloodPressure': 'low | medium_low | medium',
655                         'SkinThickness': 'low | medium_low | medium',
656                         'Insulin': 'low | medium',
657                         'BMI': 'obese | overweight',
658                         'DiabetesPedigreeFunction': 'medium | high',
659                         'Age': 'low | medium',
660                         'Outcome': 'low'})
661         fuzzy.add_rule({'Pregnancies': 'low | medium',
662                         'Glucose': 'medium | medium_high | high',
663                         'BloodPressure': 'low | medium_low | medium',
664                         'SkinThickness': 'low | medium_low | medium',
665                         'Insulin': 'low | medium',
666                         'BMI': 'obese | overweight',
667                         'DiabetesPedigreeFunction': 'low | medium',
668                         'Age': 'medium | high',
669                         'Outcome': 'low'})
670         # Trojki Glucose i DiabetesPedigreeFunction (1) 73
671         fuzzy.add_rule({'Pregnancies': 'low | medium',
672                         'Glucose': 'medium | medium_high | high',
673                         'BloodPressure': 'low | medium_low | medium',
674                         'SkinThickness': 'low | medium_low | medium',
```

```
                    'Insulin': 'low | medium',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'medium | high',
                    'Outcome': 'low'})
    # Trojki BloodPressure i SkinThickness (4) 77
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'medium | high',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
                    'Outcome': 'low'})
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'low | medium',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
                    'Outcome': 'low'})
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'low | medium',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'low | medium',
                    'Outcome': 'low'})
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'low | medium',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'medium | high',
                    'Outcome': 'low'})
    # Trojki BloodPressure i Insulin (3) 80
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
                    'Outcome': 'low'})
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'medium | medium_high | high',
```

```
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'low'})
    # Trojki BloodPressure i BMI (2) 82
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'low'})
    # Trojki BloodPressure i DiabetesPedigreeFunction (1) 83
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'low'})
    # Trojki SkinThickness i Insulin (3) 86
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
    fuzzy.add_rule({'Pregnancies': 'low | medium',
```

```
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'low'})
        # Trojki SkinThickness i BMI (2) 88
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'low'})
        # Trojki SkinThickness i DiabetesPedigreeFunction (1) 89
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'low'})
        # Trojki Insulin i BMI (2) 91
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
```

```
                    'Outcome': 'low'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'obese | overweight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'medium | high',
                    'Outcome': 'low'})
        # Trojki Insulin i DiabetesPedigreeFunction (1) 92
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'low | medium',
                    'Outcome': 'low'})
        # Trojki BMI i DiabetesPedigreeFunction (1) 93
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'low | medium',
                    'BMI': 'obese | overweight',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'medium | high',
                    'Outcome': 'low'})

        # 4 wysokie, 4 niskie
        # Czworki Pregnancies i Glucose i BloodPressure (5) 98
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'low | medium',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
```

```
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        # Czworki Pregnancies i Glucose i SkinThickness (4) 102
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
```

```
                    'Insulin': 'low | medium',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        # Czworki Pregnancies i Glucose i Insulin (3) 105
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'low | medium',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        # Czworki Pregnancies i Glucose i BMI (2) 107
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'low | medium',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'low | medium',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'low | medium',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        # Czworki Pregnancies i Glucose i DiabetesPedigreeFunction (1)
            10
        fuzzy.add_rule({'Pregnancies': 'medium | high',
```

```
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    # Czworki Pregnancies i BloodPressure i SkinThickness (4) 112
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    # Czworki Pregnancies i BloodPressure i Insulin (3) 115
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
```

```
1059        fuzzy.add_rule({'Pregnancies': 'medium | high',
1060                        'Glucose': 'low | medium_low | medium',
1061                        'BloodPressure': 'medium | medium_high | high',
1062                        'SkinThickness': 'low | medium_low | medium',
1063                        'Insulin': 'medium | high',
1064                        'BMI': 'underweight | healthy_weight',
1065                        'DiabetesPedigreeFunction': 'medium | high',
1066                        'Age': 'low | medium',
1067                        'Outcome': 'high'})
1068        fuzzy.add_rule({'Pregnancies': 'medium | high',
1069                        'Glucose': 'low | medium_low | medium',
1070                        'BloodPressure': 'medium | medium_high | high',
1071                        'SkinThickness': 'low | medium_low | medium',
1072                        'Insulin': 'medium | high',
1073                        'BMI': 'underweight | healthy_weight',
1074                        'DiabetesPedigreeFunction': 'low | medium',
1075                        'Age': 'medium | high',
1076                        'Outcome': 'high'})
1077        # Czworki Pregnancies i BloodPressure i BMI (2) 117
1078        fuzzy.add_rule({'Pregnancies': 'medium | high',
1079                        'Glucose': 'low | medium_low | medium',
1080                        'BloodPressure': 'medium | medium_high | high',
1081                        'SkinThickness': 'low | medium_low | medium',
1082                        'Insulin': 'low | medium',
1083                        'BMI': 'overweight | obese',
1084                        'DiabetesPedigreeFunction': 'medium | high',
1085                        'Age': 'low | medium',
1086                        'Outcome': 'high'})
1087        fuzzy.add_rule({'Pregnancies': 'medium | high',
1088                        'Glucose': 'low | medium_low | medium',
1089                        'BloodPressure': 'medium | medium_high | high',
1090                        'SkinThickness': 'low | medium_low | medium',
1091                        'Insulin': 'low | medium',
1092                        'BMI': 'overweight | obese',
1093                        'DiabetesPedigreeFunction': 'low | medium',
1094                        'Age': 'medium | high',
1095                        'Outcome': 'high'})
1096        # Czworki Pregnancies i BloodPressure i DiabetesPedigreeFunction
                (1) 118
1097        fuzzy.add_rule({'Pregnancies': 'medium | high',
1098                        'Glucose': 'low | medium_low | medium',
1099                        'BloodPressure': 'medium | medium_high | high',
1100                        'SkinThickness': 'low | medium_low | medium',
1101                        'Insulin': 'low | medium',
1102                        'BMI': 'underweight | healthy_weight',
1103                        'DiabetesPedigreeFunction': 'medium | high',
1104                        'Age': 'medium | high',
1105                        'Outcome': 'high'})
1106        # Czworki Pregnancies i SkinThickness i Insulin (3) 121
1107        fuzzy.add_rule({'Pregnancies': 'medium | high',
1108                        'Glucose': 'low | medium_low | medium',
1109                        'BloodPressure': 'low | medium_low | medium',
1110                        'SkinThickness': 'medium | medium_high | high',
1111                        'Insulin': 'medium | high',
1112                        'BMI': 'overweight | obese',
```

```
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        # Czworki Pregnancies i SkinThickness i BMI (2) 123
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        # Czworki Pregnancies i SkinThickness i DiabetesPedigreeFunction
            (1) 124
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        # Czworki Pregnancies i Insulin i BMI (2) 126
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
```

```
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'low | medium',
                    'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
    # Czworki Pregnancies i Insulin i DiabetesPedigreeFunction (1)
       127
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
    # Czworki Pregnancies i BMI i DiabetesPedigreeFunction (1) 128
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'low | medium',
                    'BMI': 'obese | overweight',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'medium | high',
                    'Outcome': 'high'})

    # Czworki Glucose i BloodPressure i SkinThickness (4) 132
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'medium | high',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
                    'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'low | medium',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
```

```
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'low | medium',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'low | medium',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'low | medium',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        # Czworki Glucose i BloodPressure i Insulin (3) 135
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'low | medium',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        # Czworki Glucose i BloodPressure i BMI (2) 137
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'low | medium',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'medium | high',
```

```
                    'Age': 'low | medium',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'low | medium',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        # Czworki Glucose i BloodPressure i DiabetesPedigreeFunction (1)
            138
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'low | medium',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        # Czworki Glucose i SkinThickness i Insulin (3) 141
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'medium | high',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'medium | high',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'low | medium',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'medium | high',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        # Czworki Glucose i SkinThickness i BMI (2) 143
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'medium | medium_high | high',
```

```
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        # Czworki Glucose i SkinThickness i DiabetesPedigreeFunction (1)
            144
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        # Czworki Glucose i Insulin i BMI (2) 146
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        # Czworki Glucose i Insulin i DiabetesPedigreeFunction (1) 147
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        # Czworki Glucose i BMI i DiabetesPedigreeFunction (1) 148
```

```
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})

        # Czworki BloodPressure i SkinThickness i Insulin (3) 151
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        # Czworki BloodPressure i SkinThickness i BMI (2) 153
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
```

```
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        # Czworki BloodPressure i SkinThickness i
            DiabetesPedigreeFunction (1) 154
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'low | medium',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        # Czworki BloodPressure i Insulin i BMI (2) 156
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'low | medium',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        # Czworki BloodPressure i Insulin i DiabetesPedigreeFunction (1)
            157
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'underweight | healthy_weight',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        # Czworki BloodPressure i BMI i DiabetesPedigreeFunction (1) 158
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'low | medium',
                    'BMI': 'obese | overweight',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'medium | high',
                    'Outcome': 'high'})

        # Czworki SkinThickness i Insulin i BMI (2) 160
```

```
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        # Czworki SkinThickness i Insulin i DiabetesPedigreeFunction (1)
            161
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'underweight | healthy_weight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        # Czworki SkinThickness i BMI i DiabetesPedigreeFunction (1) 162
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})

        # Czworki Insulin i BMI i DiabetesPedigreeFunction (1) 163
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})


        # Wiekszosc niskich
        # Same niskie (1) 164
        fuzzy.add_rule({'Pregnancies': 'medium | high',
```

```
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    # Pojedyncze niskie (8) 172
    fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
```

```
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        # Pary Pregnancies (7) 179
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
```

```
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        # Pary Glucose (6) 185
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
```

```
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
    # Pary BloodPressure (5) 190
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
```

```
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
    # Pary SkinThickness (4) 194
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
```

```
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
    # Pary Insulin (3) 197
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
    # Pary BMI (2) 199
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
```

```
        # Pary DiabetesPedigreeFunction (1) 200
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        # Trojki Pregnancies i Glucose (6) 206
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
```

```
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'medium | high',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'low | medium',
                    'Outcome': 'high'})
        # Trojki Pregnancies i BloodPressure (5) 211
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'low | medium',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'medium | high',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'medium | high',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'medium | high',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'low | medium',
```

```
                    'Outcome': 'high'})
        # Trojki Pregnancies i SkinThickness (4) 215
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'low | medium',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'low | medium',
                    'Outcome': 'high'})
        # Trojki Pregnancies i Insulin (3) 218
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'low | medium',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'low | medium',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'low | medium',
```

```
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'low | medium',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'low | medium',
                    'Outcome': 'high'})
        # Trojki Pregnancies i BMI (2) 220
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'medium | high',
                    'BMI': 'obese | overweight',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'medium | high',
                    'BMI': 'obese | overweight',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'low | medium',
                    'Outcome': 'high'})
        # Trojki Pregnancies i DiabetesPedigreeFunction (1) 221
        fuzzy.add_rule({'Pregnancies': 'low | medium',
                    'Glucose': 'medium | medium_high | high',
                    'BloodPressure': 'medium | medium_high | high',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'medium | high',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'low | medium',
                    'Age': 'low | medium',
                    'Outcome': 'high'})
        # Trojki Glucose i BloodPressure (5) 226
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'low | medium_low | medium',
                    'Insulin': 'medium | high',
                    'BMI': 'overweight | obese',
                    'DiabetesPedigreeFunction': 'medium | high',
                    'Age': 'medium | high',
                    'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                    'Glucose': 'low | medium_low | medium',
                    'BloodPressure': 'low | medium_low | medium',
                    'SkinThickness': 'medium | medium_high | high',
                    'Insulin': 'low | medium',
```

```
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
    # Trojki Glucose i SkinThickness (4) 230
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
```

```
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        # Trojki Glucose i Insulin (3) 233
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        # Trojki Glucose i BMI (2) 235
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
```

```
                        'Insulin': 'medium | high',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        # Trojki Glucose i DiabetesPedigreeFunction (1) 236
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'low | medium_low | medium',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        # Trojki BloodPressure i SkinThickness (4) 240
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        # Trojki BloodPressure i Insulin (3) 243
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
```

```
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
    # Trojki BloodPressure i BMI (2) 245
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
    # Trojki BloodPressure i DiabetesPedigreeFunction (1) 246
    fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'low | medium_low | medium',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
    # Trojki SkinThickness i Insulin (3) 249
```

```
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        # Trojki SkinThickness i BMI (2) 251
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        # Trojki SkinThickness i DiabetesPedigreeFunction (1) 252
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'low | medium_low | medium',
                        'Insulin': 'medium | high',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
```

```
                        'Outcome': 'high'})
        # Trojki Insulin i BMI (2) 254
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'medium | high',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
        # Trojki Insulin i DiabetesPedigreeFunction (1) 255
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'low | medium',
                        'BMI': 'overweight | obese',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'medium | high',
                        'Outcome': 'high'})
        # Trojki BMI i DiabetesPedigreeFunction (1) 256
        fuzzy.add_rule({'Pregnancies': 'medium | high',
                        'Glucose': 'medium | medium_high | high',
                        'BloodPressure': 'medium | medium_high | high',
                        'SkinThickness': 'medium | medium_high | high',
                        'Insulin': 'medium | high',
                        'BMI': 'obese | overweight',
                        'DiabetesPedigreeFunction': 'low | medium',
                        'Age': 'low | medium',
                        'Outcome': 'high'})
```

3.9 Testowanie klasyfiaktora zbioru rozmytego

```
    # Test dzialania Klasyfikatora rozmytego:
      for _ in range(3):
      shuffled_data = ProcessingData.shuffle(cleared_data)
      _, validation_data = ProcessingData.split(shuffled_data, 0.7)

      # true negative
      tn = 0
      # false positive
      fp = 0
      # false negative
```

```
        fn = 0
        # true positive
        tp = 0

        # sprawdzenie dla kazdej probki w zbiorze walidacyjnym
        for _, value in validation_data.iterrows():
            # zapisanie wyniku i jego wartosci liczbowej do zmiennych
            result = fuzzy.compute(value)
            outcome = value['Outcome']

            # zliczenie przypadkow
            if outcome == 0:
                if result[0] == 'low':
                    tn += 1
                elif result[0] == 'high':
                    fp += 1
            else:
                if result[0] == 'high':
                    tp += 1
                elif result[0] == 'low':
                    fn += 1

        # wyswietlenie dokladnosci
        print(f'Accuracy for all: {tn + tp}/{tn + fp + fn + tp}; {(tn +
            tp) / (tn + fp + fn + tp) * 100:0.2f}%')

        # Tworzenie tablicy pomylek
        cf_matrix = np.array([[tn, fp],
                              [fn, tp]])
        group_names = ['True Neg', 'False Pos', 'False Neg', 'True Pos']
        group_counts = [f'{value:0.0f}' for value in cf_matrix.flatten()
            ]
        group_percentages = [f'{value:.2%}' for value in cf_matrix.
            flatten() / np.sum(cf_matrix)]

        labels = [f'{v1}\n{v2}\n{v3}' for v1, v2, v3 in zip(group_names,
             group_counts, group_percentages)]

        labels = np.asarray(labels).reshape(2,2)

        plt.figure(figsize=(16, 12))
        sns.set(font_scale=1.4)
        plot = sns.heatmap(cf_matrix, annot=labels, annot_kws={'size':
            16}, fmt='', cmap='Blues')
        plot.set_title('Tablica pomylek')
        plot.set_xlabel('Przewidziane etykiety')
        plot.set_ylabel('Prawdziwe etykiety')
        plt.show()
```