Klasyfikator oparty na twierdzeniu Bayesa przy naiwnym założeniu o wzajemnej niezależności atrybutów

Łukasz Odwrot 218283

13.03.2018

Spis treści

1 Wstęp

Naiwny klasyfikator bayesowski to prosty klasyfikator probabilistyczny oparty o twierdzenie Bayesa i założeniu o niezależności zmiennych losowych. Dla danej klasy obiektu y i wektora cech X na podstawie twierdzenia Bayesa prawdziwy jest wzór:

$$P(y \mid x_1, ..., x_n) = \frac{P(y)P(x_1, ...x_n \mid y)}{P(x_1, ..., x_n)}$$

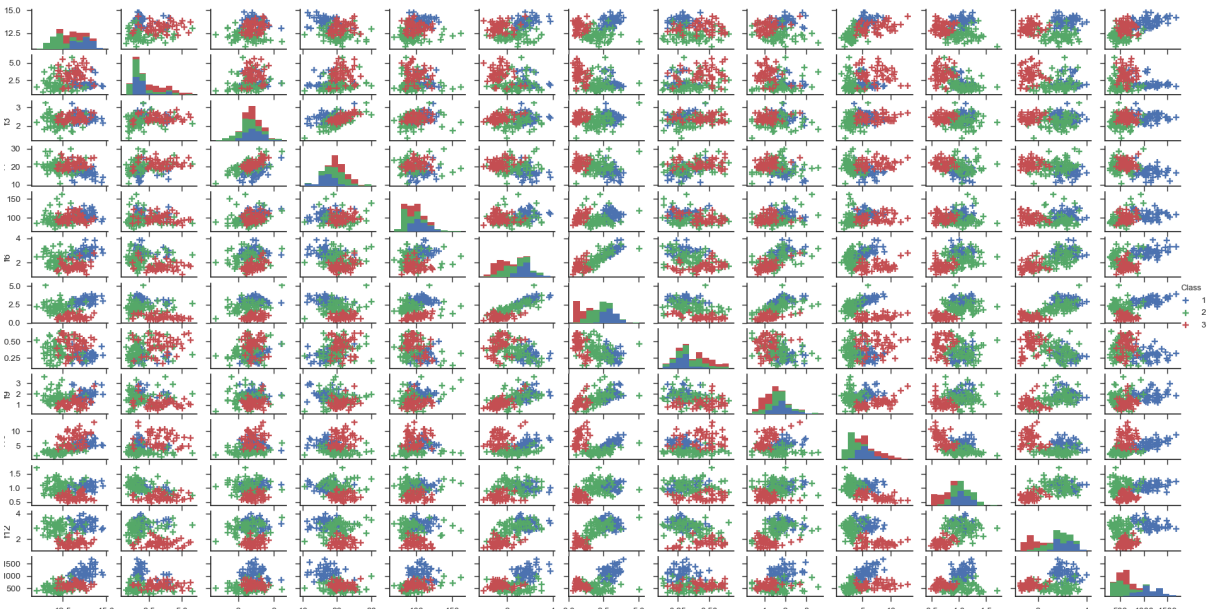
Korzystając z założenia o niezależności zdarzeń i przekształceń można dojść do wzoru:

$$\hat{y} = arg \max_y P(y) \prod_{i=1}^{n} P(x_i \mid y)$$

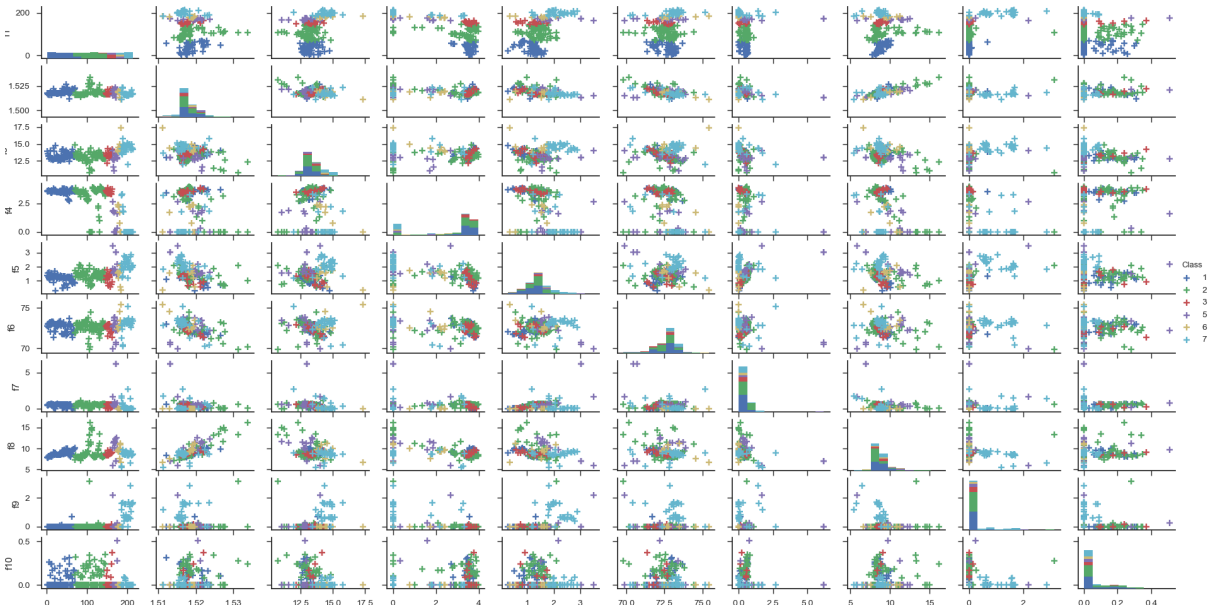
Dzięki takiemu mechanizmowi na podstawie ciągu uczącego można wytrenować klasyfikator, a następnie wykorzystać go do klasyfikacji nowych obiektów. Do badania jakości uzyskanych klasyfikatorów użyte zostaną następujące mechanizmy: Confusion Matrix, ccuracy, Precision, Recall, Fscore. Badania zostaną przeprowadzone na trzech zbiorach danych: Glass, Wine oraz Diabetes.

2 Badane zbiory

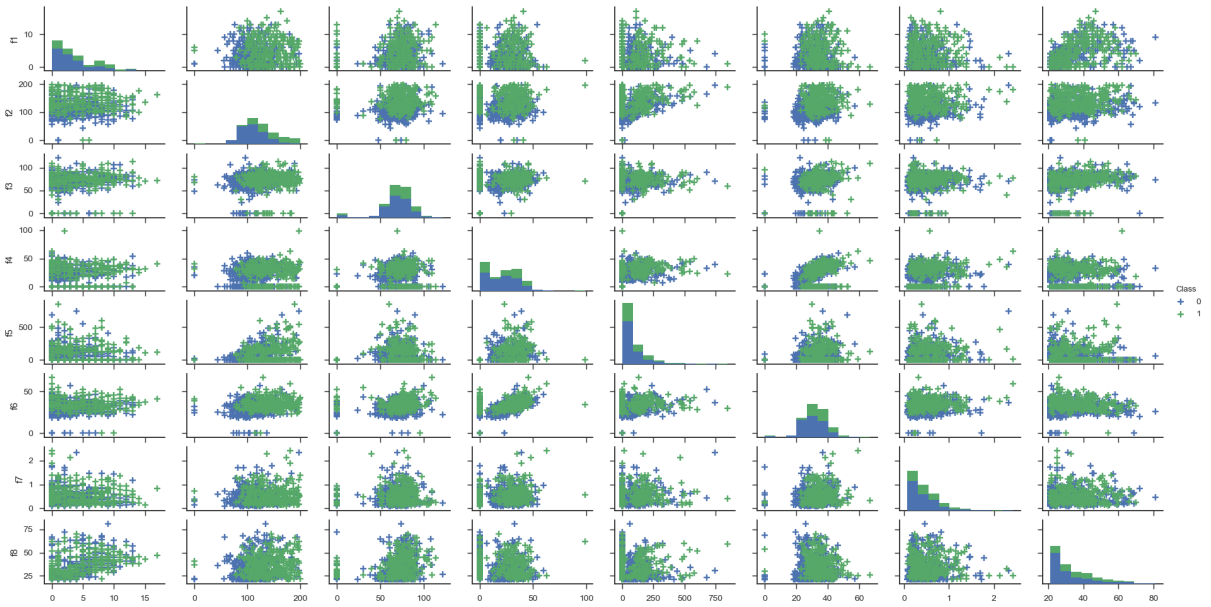
Rozkłady cech dla poszczególnych klas przedstawiono na poniższych rysunkach.



Rysunek 1: Rozkład cech dla zbioru Wine



Rysunek 2: Rozkład cech dla zbioru Glass



Rysunek 3: Rozkład cech dla zbioru Diabetes

3 Implementacja klasyfikatora i problem wygładzania

Na podstawie zaprezentowanych wcześniej wzorów można stwierdzić, że w przypadku, gdy dana kombinacja cechy i wartości nie wystąpi w zbiorze uczącym, wyzeruje ona prawdopodobieństwo klasyfikacji do danej klasy przy wystąpieniu cechy w czasie klasyfikacji właściwej. Z tym zjawiskiem można poradzić sobie poprzez wygładzenie danych, czyli eliminację zerowych prawdopodobieństw lub założenie, że dane mają rozkład normalny. W tym wypadku można skorzystać ze wzoru, który likwiduje zerowe prawdopodobieństwa.

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp(-\frac{(x_i - \mu_y)^2)}{2\sigma_y^2})$$

Do badań wykorzystane zostaną dwie implementacje pochodzące z biblioteki *sklearn*: *GaussianNB* oraz *MultinomialNB* ze współczynnikiem wygładzania $\alpha = 1$, który wykorzystuje *wygładzanie laplace*.

```
def getClassifireGausian(_df):
    featureVals = [x for x in _df if x != 'Class']
    gnb = GaussianNB()
    gnb.fit(_df[featureVals], _df['Class'])
    return gnb

def getClassifireMultinomial(_df):
    featureVals = [x for x in _df if x != 'Class']
    gnb = MultinomialNB(alpha=1.0)
    gnb.fit(_df[featureVals], _df['Class'])
    return gnb


def predict(_classifier, _df):
    featureVals = [x for x in _df if x != 'Class']
    y_pred = _classifier.predict(_df[featureVals])
    return y_pred
```

4 Metody dyskretyzacji

Jednym z celów zadania jest zbadanie wpływu dyskretyzacji danych na jakość klasyfikatora. W programie zaimplementowany zostały trzy rodzaje dyskretyzacji.

1. *Equal width intervals*: podział zakresu wartości atrybutu ciągłego na k przedziałów o jednakowej szerokości.

   ```
   def equalWidth(_df):
       featureVals = [x for x in _df if x != 'Class']
       discretizedMap = {'Class': _df['Class']}

       for x in featureVals:
           discretizedMap[x] = pd.cut(_df[x], 5, labels=False)

       nFrame = pd.DataFrame.from_dict(discretizedMap)
       return nFrame
   ```

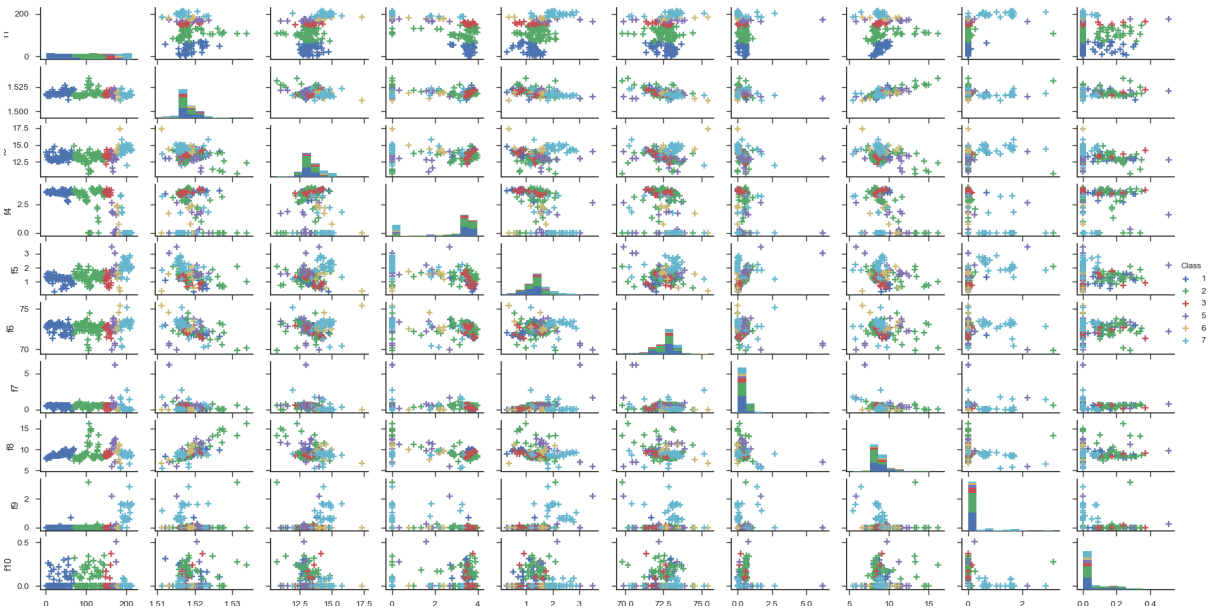
2. *Equal frequency intervals* : podział zakresu wartości atrybutu ciągłego na k przedziałów, z których każdemu odpowiada możliwie tyle samo przykładów ze zbioru trenującego.

   ```
   def equalFreq(_df):
       featureVals = [x for x in _df if x != 'Class']
       discretizedMap = {'Class': _df['Class']}

       for x in featureVals:
           discretizedMap[x] = pd.qcut(_df[x], 5, labels=False,
                                       duplicates='drop')

       nFrame = pd.DataFrame.from_dict(discretizedMap)
       return nFrame
   ```

3. *Minimum Description Length Binning*: bazująca na entropii metoda opracowana przez *Usama Fayyad's*. Dokładny opis dostępny pod linkiem: mdlp.

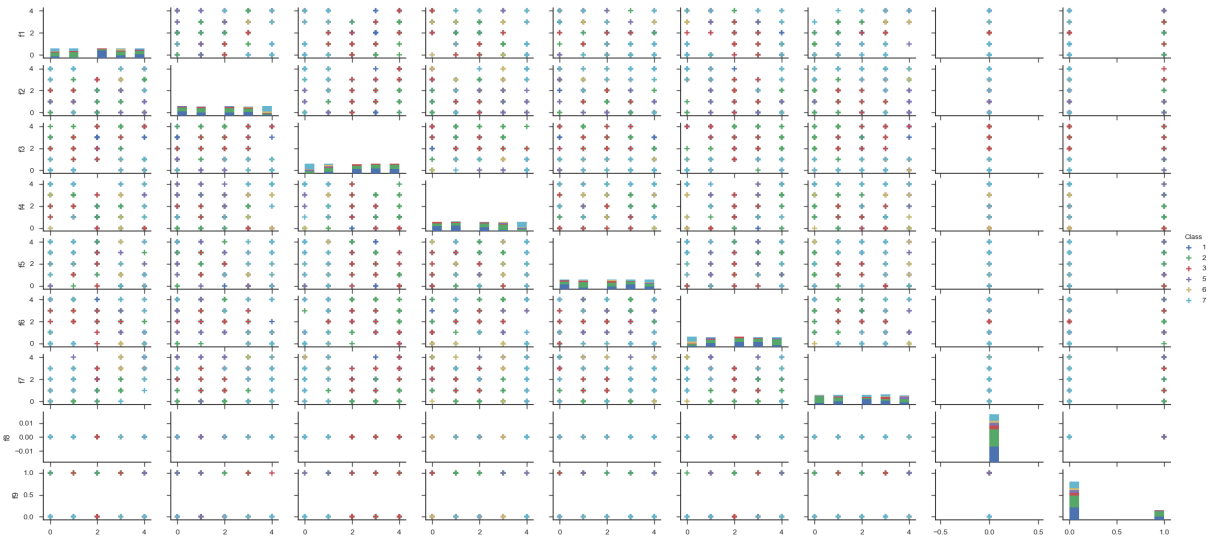
```
def discMdlp(_df):
    featureVals = [x for x in _df if x != 'Class']
    transformer = MDLP()
    discretizedMap = {'Class': _df['Class']}

    discret = transformer.fit_transform(_df[featureVals],_df['Class
                                                    '])
    nFrame = pd.DataFrame(data=discret, columns=featureVals)
    nFrame.loc[:,'Class'] = pd.Series(_df['Class'])

    return nFrame
```

Działanie poszczególnych metod dyskretyzacji dla instancji Glass przedstawiono poniżej.

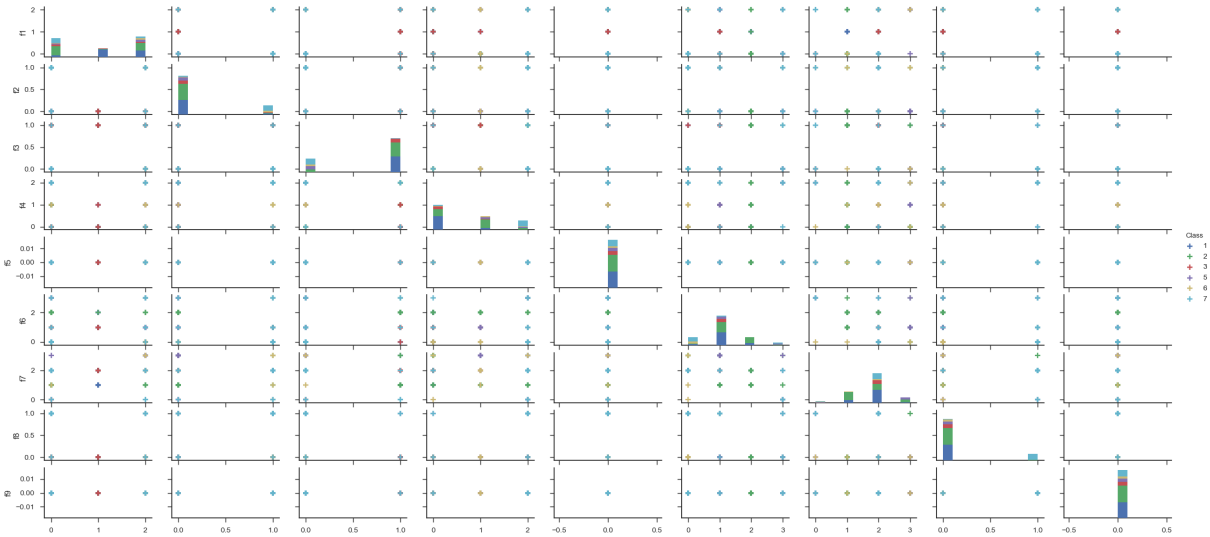


Rysunek 4: Brak dyskretyzacji

Rysunek 5: Equal width



Rysunek 6: Equal frequency



Rysunek 7: MDLP

5 Badanie metod kroswalidacji

Do oceny klasyfikatora zostanie użyta metoda kroswalidacji z metodą podziału *X-fold*. Polega ona na tym, że zbiór dzielimy na X w miarę możliwości równych części. W przypadku odmiany stratyfikowanej każda część zawiera możliwie tyle samo danych z każdej klasy. Jedna część zostanie użyta do oceny klasyfikatora, a pozostałe wejdą w skład zbioru trenującego. Następnie X razy zmianie ulegnie część do klasyfikacji, a cały proces zostanie powtórzony. Wpływ rodzaju kroswalidacji dla różnych zbiorów przedstawiono w poniższych tabelach.

Ilość części	Aaccracy	Precision	Recall	fscore
Instancja Wine				
Brak randomizacji, brak stratyfikacji				
2	0.37	0.15	0.37	0.21
3	0.30	0.13	0.30	0.18
4	0.66	0.66	0.66	0.60
5	0.93	0.93	0.93	0.93
6	0.93	0.93	0.93	0.93
7	0.93	0.93	0.93	0.93
8	0.96	0.96	0.96	0.96
9	0.95	0.95	0.95	0.95
10	0.96	0.96	0.96	0.96
Randomizacja, brak stratyfikacji				
2	0.98	0.98	0.98	0.98
3	0.98	0.98	0.98	0.98
4	0.97	0.97	0.97	0.97
5	0.97	0.97	0.97	0.97
6	0.98	0.98	0.98	0.98
7	0.97	0.97	0.97	0.97
8	0.97	0.97	0.97	0.97
9	0.96	0.96	0.96	0.96
10	0.98	0.98	0.98	0.98
Brak randomizacji, stratyfikacja				
2	0.97	0.97	0.97	0.97
3	0.96	0.96	0.96	0.96
4	0.96	0.96	0.96	0.96
5	0.95	0.95	0.95	0.95
6	0.96	0.96	0.96	0.96
7	0.96	0.96	0.96	0.96
8	0.96	0.96	0.96	0.96
9	0.95	0.95	0.95	0.95
10	0.96	0.96	0.96	0.96
Randomizacja, stratyfikacja				
2	0.97	0.97	0.97	0.97
3	0.96	0.96	0.96	0.96
4	0.96	0.96	0.96	0.96
5	0.95	0.95	0.95	0.95
6	0.96	0.96	0.96	0.96
7	0.96	0.96	0.96	0.96
8	0.96	0.96	0.96	0.96
9	0.95	0.95	0.95	0.95
10	0.96	0.96	0.96	0.96

Instancja Glass				
Brak randomizacji, brak stratyfikacji				
2	0.09	0.08	0.09	0.08
3	0.23	0.16	0.23	0.19
4	0.13	0.12	0.13	0.13
5	0.20	0.25	0.20	0.21
6	0.12	0.18	0.12	0.13
7	0.28	0.33	0.28	0.28
8	0.17	0.20	0.17	0.17
9	0.25	0.37	0.25	0.28
10	0.33	0.41	0.33	0.34
Randomizacja, brak stratyfikacji				
2	0.40	0.48	0.40	0.40
3	0.43	0.53	0.43	0.46
4	0.40	0.48	0.40	0.40
5	0.40	0.45	0.40	0.40
6	0.46	0.48	0.46	0.44
7	0.43	0.45	0.43	0.41
8	0.43	0.49	0.43	0.42
9	0.44	0.48	0.44	0.43
10	0.45	0.46	0.45	0.42
Brak randomizacji, stratyfikacja				
2	0.36	0.47	0.36	0.39
3	0.37	0.44	0.37	0.38
4	0.39	0.47	0.39	0.40
5	0.33	0.38	0.33	0.34
6	0.41	0.44	0.41	0.40
7	0.39	0.39	0.39	0.37
8	0.42	0.42	0.42	0.39
9	0.44	0.44	0.44	0.42
10	0.43	0.43	0.43	0.41
Randomizacja, stratyfikacja				
2	0.36	0.47	0.36	0.39
3	0.37	0.44	0.37	0.38
4	0.39	0.47	0.39	0.40
5	0.33	0.38	0.33	0.34
6	0.41	0.44	0.41	0.40
7	0.39	0.39	0.39	0.37
8	0.42	0.42	0.42	0.39
9	0.44	0.44	0.44	0.42
10	0.43	0.43	0.43	0.41

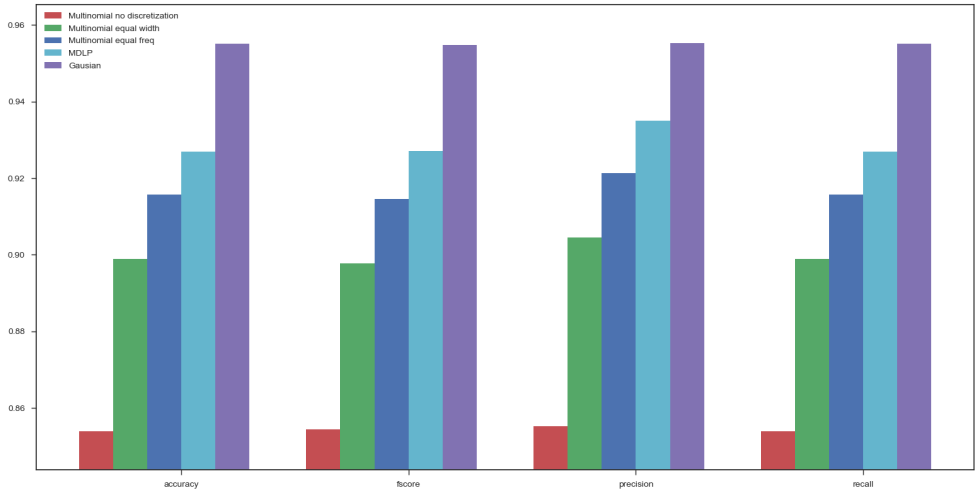
Instancja Diabetes				
Brak randomizacji, brak stratyfikacji				
2	0.75	0.66	0.60	0.63
3	0.74	0.64	0.58	0.61
4	0.75	0.66	0.60	0.63
5	0.75	0.66	0.59	0.62
6	0.75	0.66	0.60	0.63
7	0.75	0.66	0.59	0.62
8	0.75	0.66	0.59	0.62
9	0.75	0.66	0.60	0.63
10	0.75	0.66	0.59	0.62
Randomizacja, brak stratyfikacji				
2	0.75	0.65	0.59	0.62
3	0.75	0.65	0.60	0.62
4	0.75	0.65	0.60	0.63
5	0.76	0.67	0.60	0.63
6	0.75	0.66	0.60	0.63
7	0.75	0.66	0.59	0.62
8	0.74	0.65	0.59	0.62
9	0.75	0.66	0.60	0.63
10	0.75	0.65	0.59	0.62
Brak randomizacji, stratyfikacja				
2	0.75	0.66	0.60	0.63
3	0.74	0.64	0.58	0.61
4	0.75	0.65	0.59	0.62
5	0.75	0.66	0.58	0.62
6	0.75	0.65	0.60	0.63
7	0.75	0.66	0.59	0.62
8	0.75	0.66	0.59	0.62
9	0.75	0.65	0.59	0.62
10	0.75	0.67	0.59	0.62
Randomizacja, stratyfikacja				
2	0.75	0.66	0.60	0.63
3	0.74	0.64	0.58	0.61
4	0.75	0.65	0.59	0.62
5	0.75	0.66	0.58	0.62
6	0.75	0.65	0.60	0.63
7	0.75	0.66	0.59	0.62
8	0.75	0.66	0.59	0.62
9	0.75	0.65	0.59	0.62
10	0.75	0.67	0.59	0.62

W przypadku próby kroswalidacji stratyfikowanej dla instancji, w której któryś z atrybutów występuje mniej razy niż ilość części, na które chcemy dokonać podziału, otrzymamy następujące ostrzeżenie: **The least populated class in y has only 9 members, which is too few.The minimum number of members in any class cannot be less than n_splits**, a w jednym ze zbiorów egzaminacyjnych obiekt z jednej klasy nie wystąpi. Widać wyraźnie, że w przypadku kroswalidacji stratyfikowanej zmiana ilości czę-

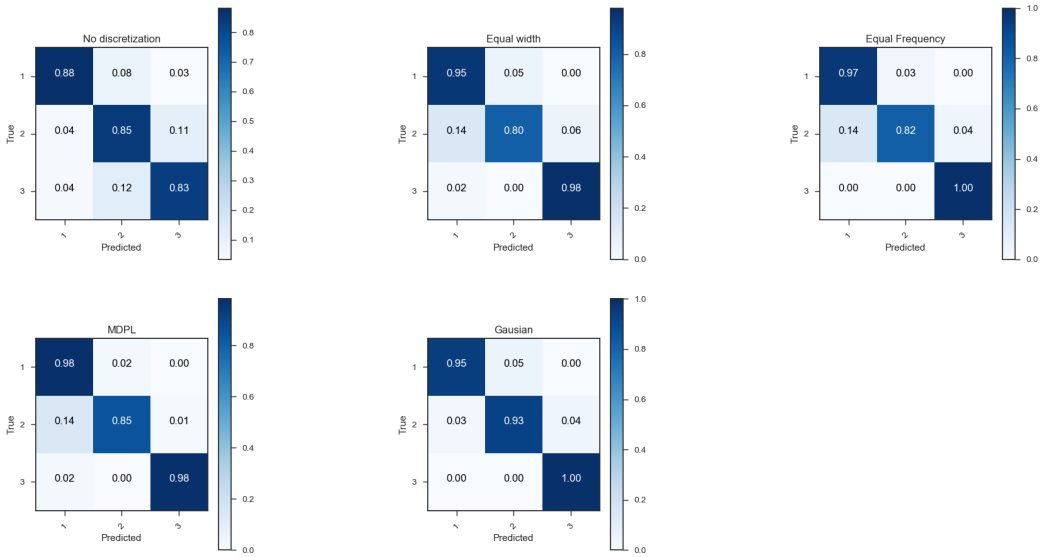
ści nie wpływa na wyniki klasyfikacji znacząco. W przypadku braku stratyfikacji nie ma sensu oceniać klasyfikatora bez randomizacji, ponieważ wszystko wtedy zależy od kolejności danych. Najlepszą metodą oceny jest kroswalidacja startyfikowana z randomizacją i to właśnie ta metoda będzie używana do oceny klasyfikatorów.

6 Porównanie działania algorytmów

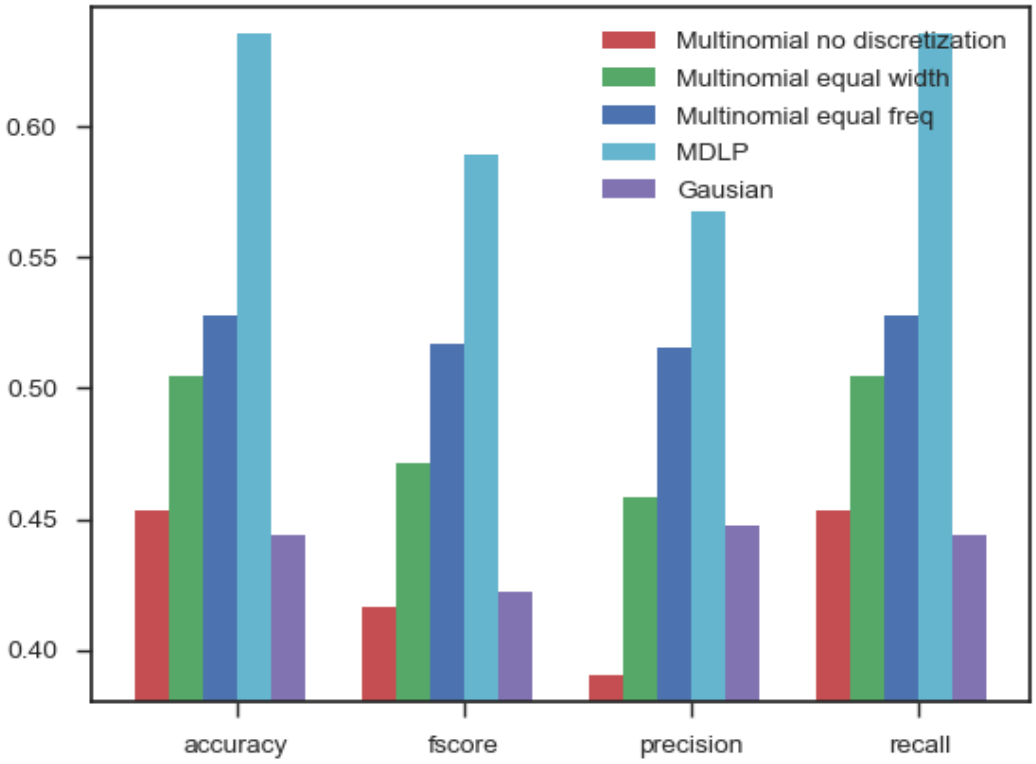
Dla wszystkich ze zbiorów zbadano jakość klasyfikatorów w zależności od sposobu liczenia prawdopodobieństwa, a w przypadku użycia *multinomialNB*, wpływ różnych metod dyskretyzacji na zmianę jakości klasyfikatora. Badania wykonano dla wszystkich zbiorów. W przypadku dyskretyzacji dla każdego ze zbiorów próbowano dobrać optymalne parametry algorytmów, a poniżej przedstawiono najlepsze rezultaty.



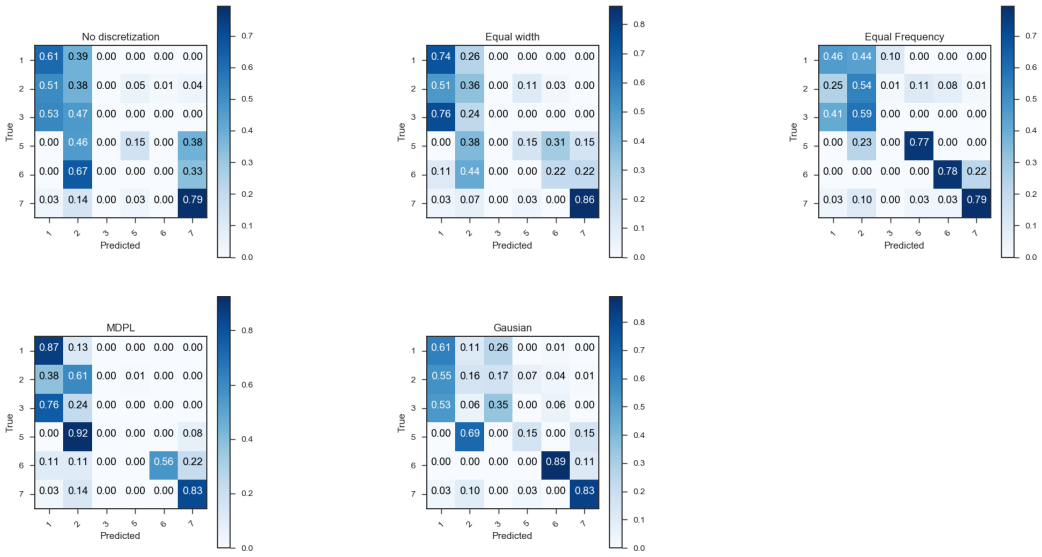
Rysunek 8: Statystyki dla instancji Wine



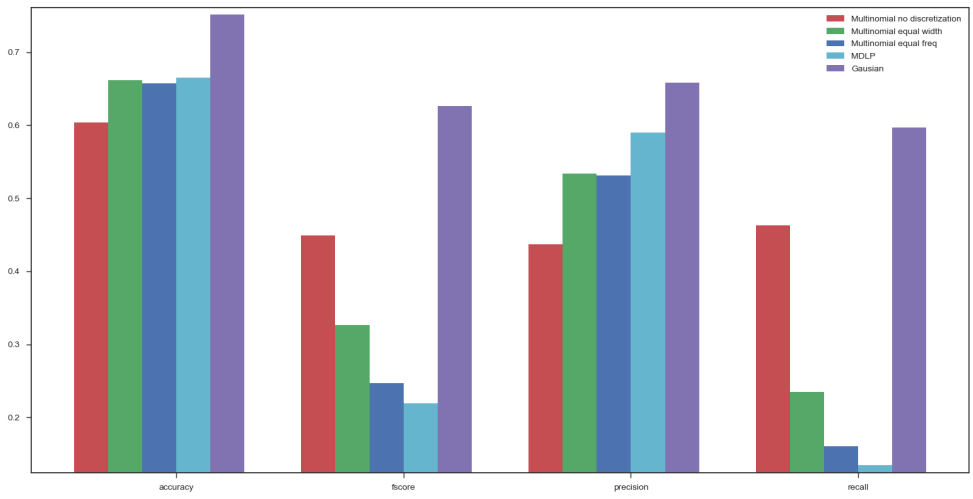
Rysunek 9: Confusion Matrix dla instancji Wine



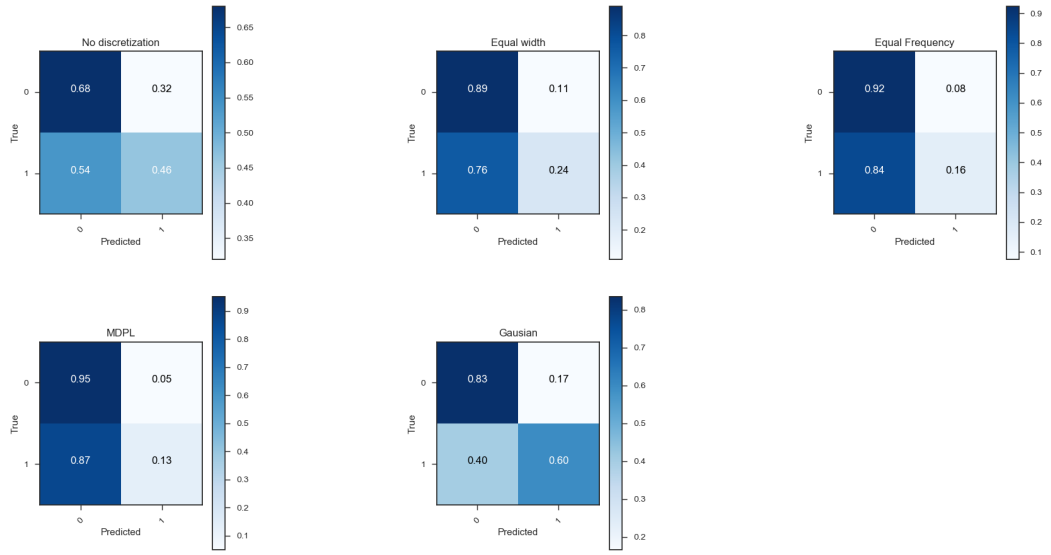
Rysunek 10: Statystyki dla instancji Glass



Rysunek 11: Confusion Matrix dla instancji Glass



Rysunek 12: Statystyki dla instancji Diabetes



Rysunek 13: Confusion Matrix dla instancji Diabetes

7 Wnioski

W ramach zadania zaimplementowany został naiwny klasyfikator bayesa. Dobór metody liczenia prawdopodobieństwa oraz dyskretyzacji, a także parametrów dyskretyzacji zależy od badanej instancji. W przypadku zbioru Wine oraz Diabetes użycie *GausianNB* dawało o wiele lepsze wyniki niż użycie *MultinomialNB* niezależnie od wybranych współczynników metod dyskretyzacji.
Natomiast w przypadku instancji *Glass* znacząco lepsze rezultaty daje zastosowanie *MultinomialNB*, a ponadto odpowiednia metoda dyskretyzacji jeszcze bardziej wpływa na polepszenie jakości klasyfikatora. W wypadku instancji *Glass* metoda dyskretyzacji *MDLP* wpłynęła na poprawienie wyników o prawie 20% względem braku dyskretyzacji oraz o około 8% względem drugiej najlepszej w tym przypadku metody *equal frequency*.
W przypadku klasyfikacji danych w realnym świecie ważne jest, zdarzają się przypadki, w których większy nacisk kładziony jest na poprawną klasyfikację do danej klasy, kosztem błędnej klasyfikacji pozostałych próbek. W przypadku diagnozowania chorób lepiej, aby osobę chorą skierować na dodatkowe badania mimo tego, że nie cierpi na żadną chorobę, niż przeoczyć chorobę u osoby, u której w rzeczywistości występuje.
W przypadku sposobu oceny klasyfikatora *k-fold crossvalidation* jest dobrą metodą, szczególnie w wersji stratyfikowanej, gdzie ilość części nie odgrywa szczególnego znaczenia. W przypadku wersji niestratyfikowanej ważne jest, aby przemieszać dane, ponieważ sztywno określona kolejność zwykle skutkuje tym, że dane o podobnych cechach znajdują się w jednej z części co w rezultacie daje nam wyniki, które nie pokrywają się z realną jakością klasyfikatora. Ilość części, na które dzielimy zbiór powinna być nie mniejsza niż ilość klas w zbiorze. Najlepszą decyzją jest jednak posłużenie się kroswalidacją stratyfikowaną.