Akademia Ekonomiczno-Humanistyczna w Warszawie

SPRAWOZDANIE

INTELIGENTNA ANALIZA DANYCH

**LAB2**

**WALIDACJA KRZYŻOWA, STRATYFIKACJA,**

**DRZEWA LOSOWE I NAIWNY KLASYFIKATOR bayesowski**

07.11.2021

JOANNA PRAJZENDANC

36358

MIŁOSZ SAKOWSKI

36381

**Spis treści**

[1. Cel i przebieg ćwiczenia 3](#_Toc11555)

[2. Definicje i założenia 3](#_Toc15627)

[2.1. Wyjaśnienie pojęć 3](#_Toc19820)

[2.2. Przygotowanie bazy danych 4](#_Toc31407)

[i. Importowanie bazy danych i oryginalna zawartość 4](#_Toc15261)

[ii. Docelowa baza danych 5](#_Toc13023)

[3. Walidacja krzyżowa i stratyfikacja 6](#_Toc8811)

[3.1. Zadanie #1 6](#_Toc2434)

[i. Treść polecenia 6](#_Toc20987)

[ii. Rozwiązanie 6](#_Toc25219)

[4. Klasyfikator drzewa losowego 8](#_Toc2433)

[4.1. Zadanie #2 8](#_Toc1905)

[i. Omówienie kodu 8](#_Toc32468)

[ii. Pytania 9](#_Toc10053)

[4.2. Zadanie #3 10](#_Toc7067)

[i. Treść polecenia 10](#_Toc26958)

[ii. Rozwiązanie 10](#_Toc25863)

[4.3. Wnioski 13](#_Toc21652)

[5. Naiwny klasyfikator bayesowski 14](#_Toc27188)

[5.1. Zadanie #4 14](#_Toc5933)

[i. Treść polecenia 14](#_Toc13986)

[ii. Rozwiązanie 14](#_Toc21849)

[5.2. Wnioski 15](#_Toc29028)

[6. Porównanie klasyfikatorów 15](#_Toc9824)

[7. Wnioski końcowe 15](#_Toc22274)

# Cel i przebieg ćwiczenia

Celem ćwiczenia było utrwalenie wiedzy w zakresie klasyfikatorów. W pierwszej kolejności należało zapoznać się z problemem niezbalansowanego zbioru danych i sposobów rozwiązania go za pomocą walidacji krzyżowej i stratyfikacji w przypadku klasyfikatora binarnego. Następnym krokiem było zapoznanie się z klasyfikatorem drzewa losowego oraz naiwnym klasyfikatorem bayesowski.

# Definicje i założenia

## Wyjaśnienie pojęć

W sprawozdaniu pojawiają się następujące pojęcia:

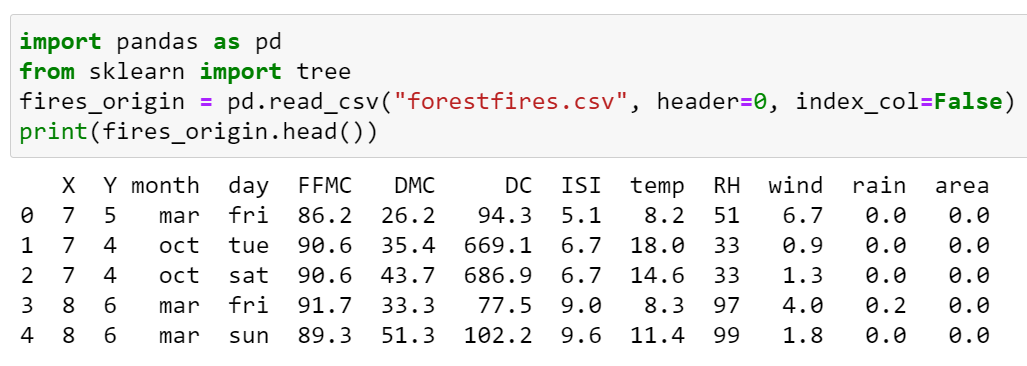
* niezbalansowana baza danych: zbiór danych, w którym występuje wyraźna dysproporcja jeśli chodzi o ilość elementów dla każdej z klas,
* walidacja krzyżowa: zbiór początkowy dzielony jest na kilka podzbiorów (w zależności od parametrów podanych przez użytkownika); następnie część zbiorów jest jako zbiór trenujący, a część jako zbiór testujący; potem następuje zmiana i inna część zbiorów jest zbiorem testującym itd.
* stratyfikacja: zbiory trenujące i testujące są dobierane tak, aby mieć pewność że w każdym z podzbiorów znajdą się elementy każdej możliwej klasy,
* klasyfikator binarny: prosty podział na zbiór trenujący i testujący w ustalonych proporcjach np. 10% danych zbiór testujący i 90% zbiór trenujący,
* klasyfikator lasu losowego: las losowy składa się z kilku drzew decyzyjnych, które powstają na 3 sposoby:
  + tylko część pierwotnego zbioru jest dzielona pomiędzy zbiór testujący i zbiór trenujący, tak aby każde drzewo decyzyjne powstało i testowało na innym zbiorze danych,
  + dla każdego drzewa decyzyjnego wymuszany jest warunek, np. w pierwszym kroku użyj innego parametru,
  + połączenie dwóch powyższych zasad.
* naiwny klasyfikator bayesowski: dopasowanie elementu jest przeprowadzane na podstawie prawdopodobieństwa *a prori.*

## Przygotowanie bazy danych

Podczas wykonywania zadań omówionych w sprawozdaniu, do rozwiązania użyto tej samej bazy danych z informacjami o pożarach w parku XX.

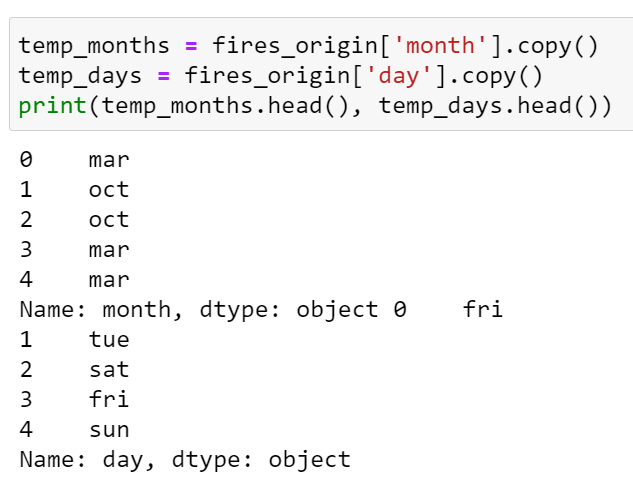
### Importowanie bazy danych i oryginalna zawartość

Dane wykorzystane w zadaniach pochodzą z pliku forestfires.csv, który został pobrany ze strony <https://archive.ics.uci.edu/ml/index.php.> Jest to zbiór informacji na temat pożarów w parku Montensinho.



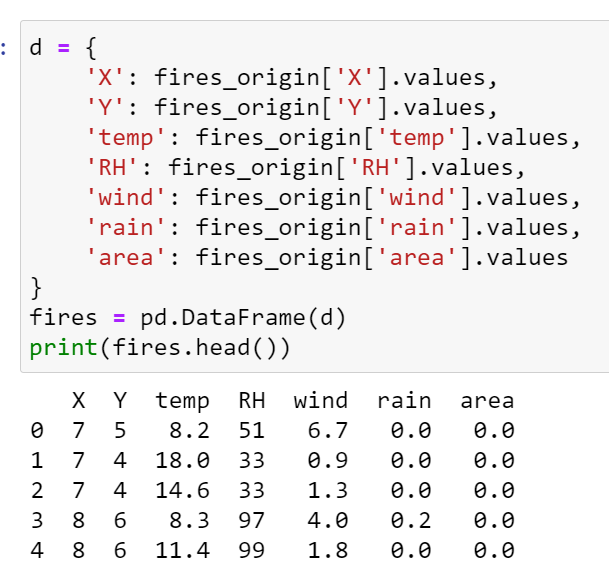
Obraz 1: Fragment kodu przedstawiający importowanie danych z pliku

Wśród danych znajdują się dane typu string, dlatego zostaną one na potrzeby ćwiczeń pominięte w docelowej bazie danych.



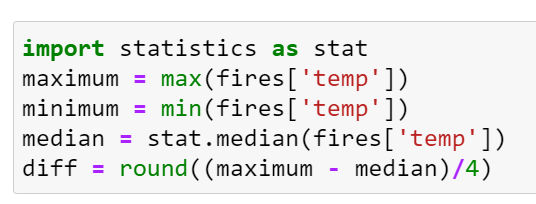
Obraz 2: Fragment kodu przedstawiający kolumny z danymi typu string, które zostały pomięte na potrzeby ćwiczenia

### Docelowa baza danych

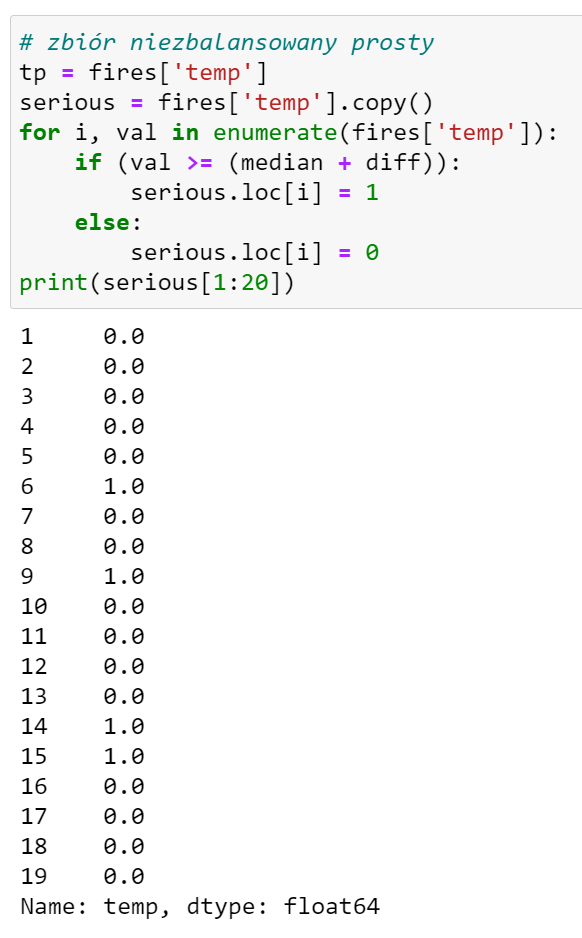


Obraz 3: Podgląd pierwszych 5 wierszy docelowego zbioru danych

W celu przypisania potrzebnych klas, do których będą się odnosić testowane klasyfikatory, przeprowadzono prostą statystykę danych i przydzielono klasy „1” lub „0” tak, aby zbiór był niezbalansowany.

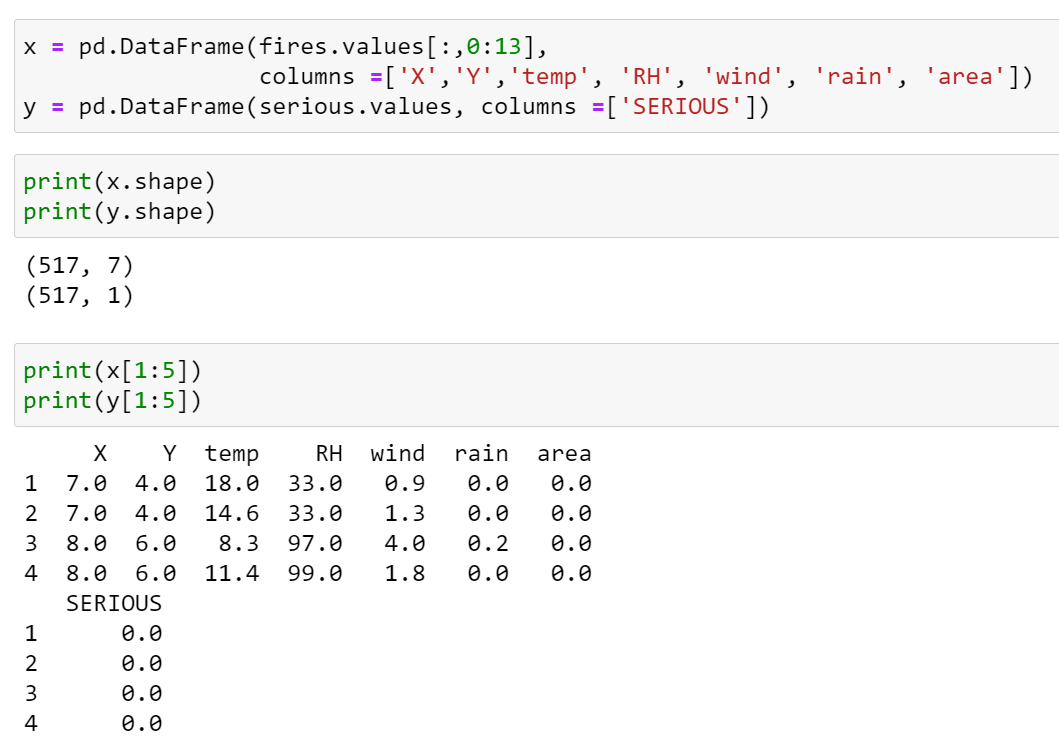


Obraz 4: Fragment kodu przedstawiający wartości zmiennych wykorzystanych w algorytmie klasyfikującym



Obraz 5: Fragment kodu przedstawiający algorytm klasyfikujący

Na koniec wszystkie kolumny ustawiono jako DataFrame:



Obraz 6: Fragment kodu przedstawiający ostateczny układ zbioru danych wykorzystanego do ćwiczeń

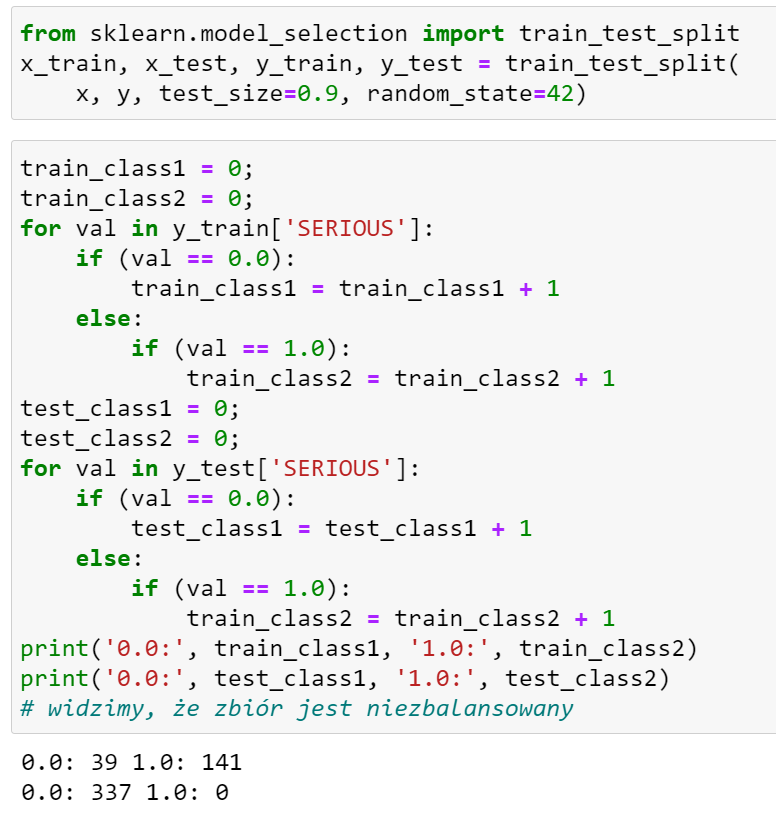
# Walidacja krzyżowa i stratyfikacja

## Zadanie #1

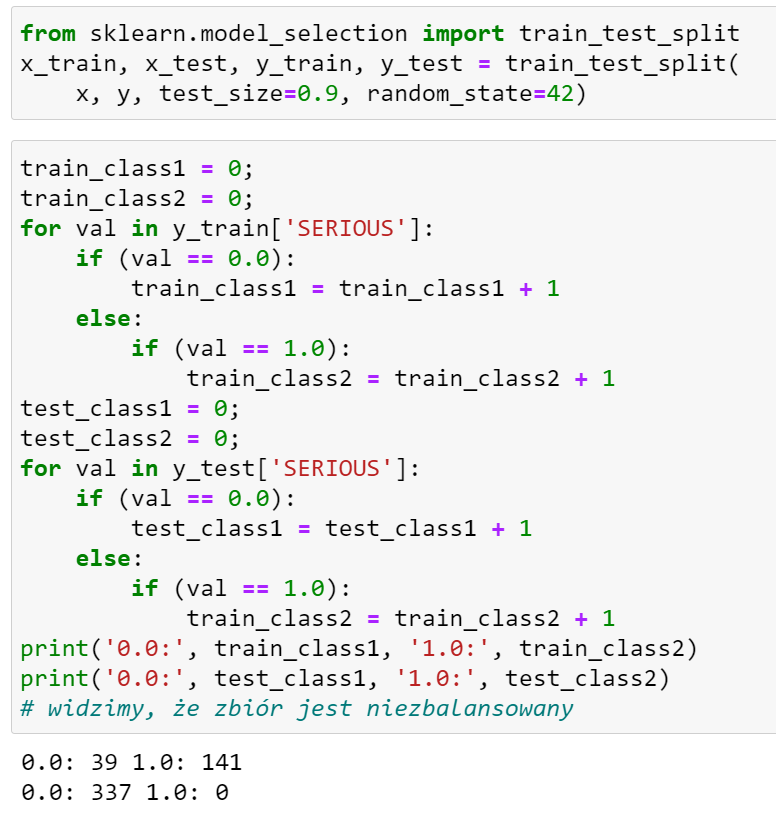
### Treść polecenia

Proszę pobrać inny zbiór danych, zmienić klasy w taki sposób żeby zbiór był niezbalansowany oraz przetestować drzewo decyzyjne stosując 5-krotną walidację krzyżową ze stratyfikacją

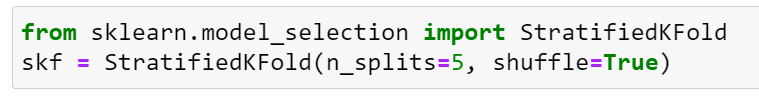
### Rozwiązanie



Obraz 7: Fragment kodu z importem biblioteki train\_test\_split oraz dokonanie podziału   
klasyfikatorem binarnym na zbiór trenujący i testujący

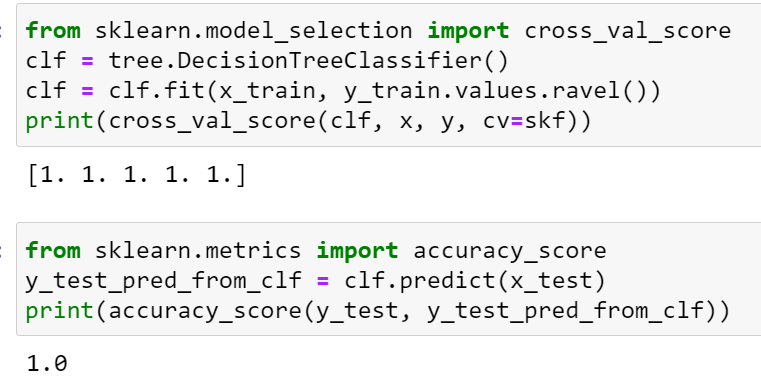


Obraz 8: Fragment kodu pokazujący, że wybrany zbiór danych jest niezbalansowany



Obraz 9: Fragment kodu przedstawiający zastosowanie stratyfikacji

Wyniki dokładności modelu były takie same niezależnie od test\_size.



Obraz 10: Fragment kodu przedstawiający zastosowanie walidacji krzyżowej ze stratyfikacją;   
wyniki przedstawiają stopień dopasowania każdej iteracji krzyżowania i dla każdego test\_size

Tabela 1: Wyniki dokładności modelu z klasyfikatorem binarnym   
ze stratyfikacją i walidacją krzyżową

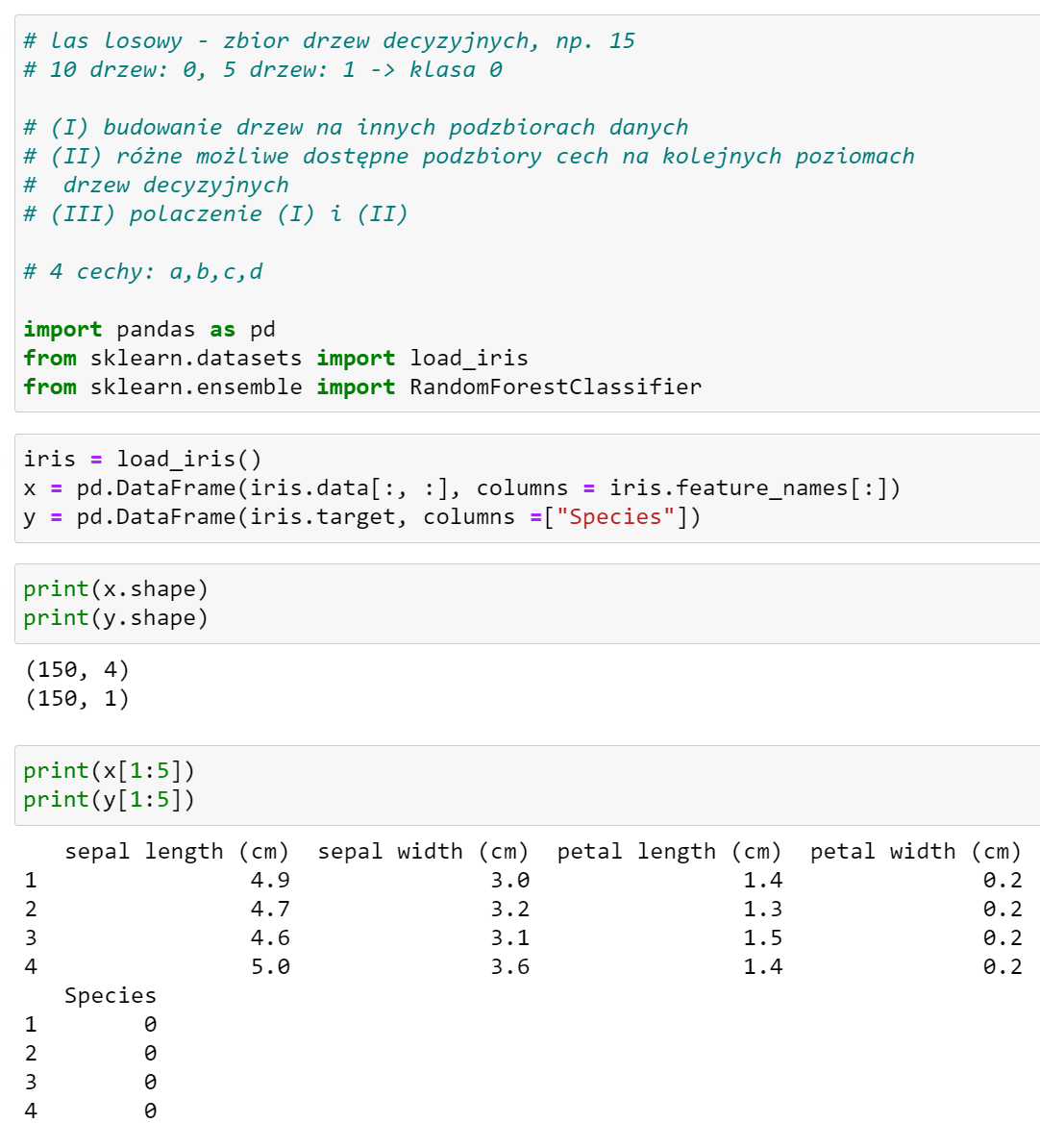
|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test\_size = 0.1** | **Test\_size = 0.5** | **Test\_size = 0.9** |
| **Klasyfikator binarny ze stratyfikacją i walidacją krzyżową** | 1.0 | 1.0 | 1.0 |

# Klasyfikator drzewa losowego

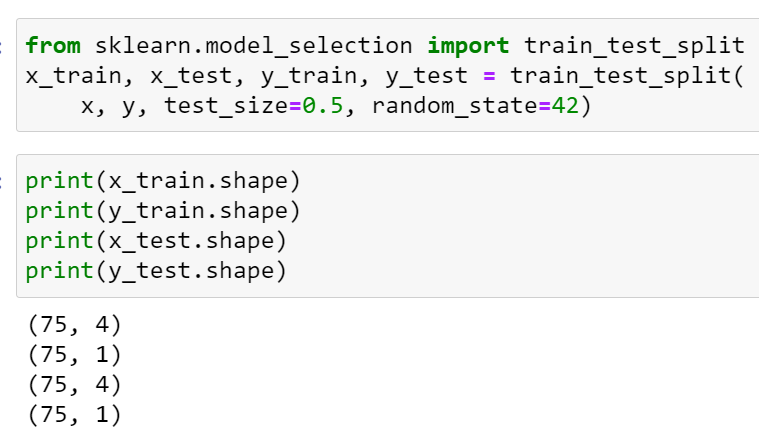
## Zadanie #2

### Omówienie kodu

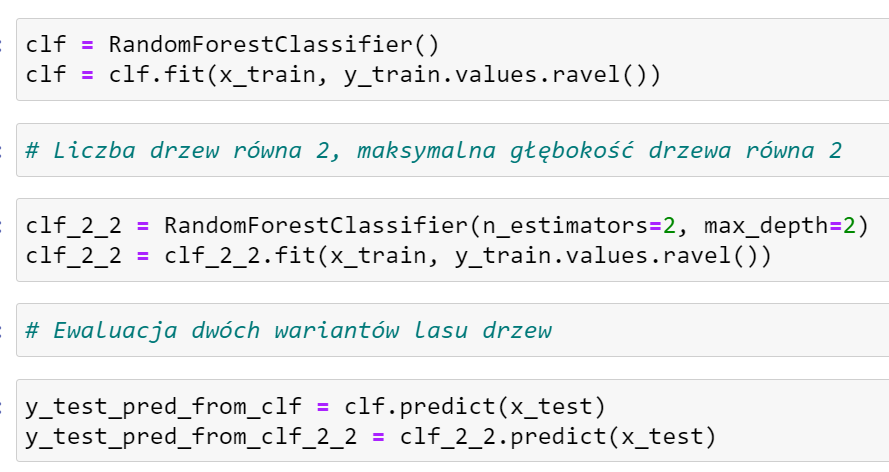
Zbiór danych z biblioteki iris zawierający przykładowe informacje na temat kwiatów, został podzielony równo na pół na zbiór trenujący i testujący. Następnie na zbiorach użyto klasyfikatora lasu losowego: najpierw z ustawieniami domyślnymi, potem z maksymalną ilością drzew równą dwa o maksymalnej głębokości równej 2.



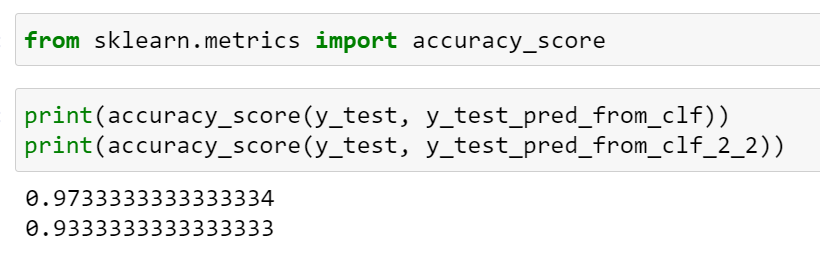
Obraz 11: Fragment kodu przedstawiający zaimportowanie danych z iris



Obraz 12: Fragment kodu pokazujący sposób dzielenia na zbiór trenujący i testujący - tutaj równo na pół



Obraz 13: Fragment kodu z zastosowaniem klasyfikatora lasu losowego



Obraz 14: Fragment kodu z obliczeniami dokładności klasyfikatora lasu losowego

### Pytania

#### Czy uzyskane wyniki są zadowalające?

Dokładność dla obu przypadków użycia klasyfikatora jest ponad 90%, co jest bardzo dobrym wynikiem.

#### Czy model został przetestowany w prawidłowy sposób?

Nie, ponieważ istnieje ryzyko, że w zbiorze trenującym są same „0” i żadnych „1” albo w zbiorze testującym nie ma żadnych „1” więc nie zbadamy skuteczności modelu w dopasowywaniu „1”.

## Zadanie #3

### Treść polecenia

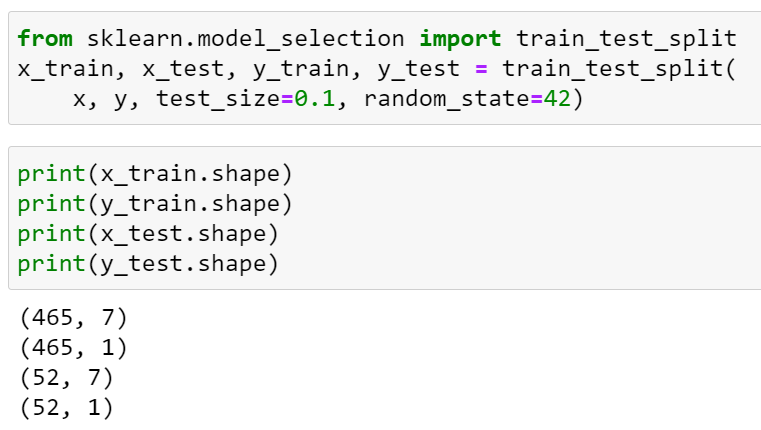
Proszę pobrać dowolny zbiór danych ze strony <https://archive.ics.uci.edu/ml/index.php.> Następnie proszę podzielić zbiór na dane trenujące i testujące, wytrenować 5 modeli lasów losowych z różną maksymalną głębokością i liczbą drzew w lesie, porównać wyniki. Proszę o sporządzenie sprawozdania z wnioskami.

### Rozwiązanie

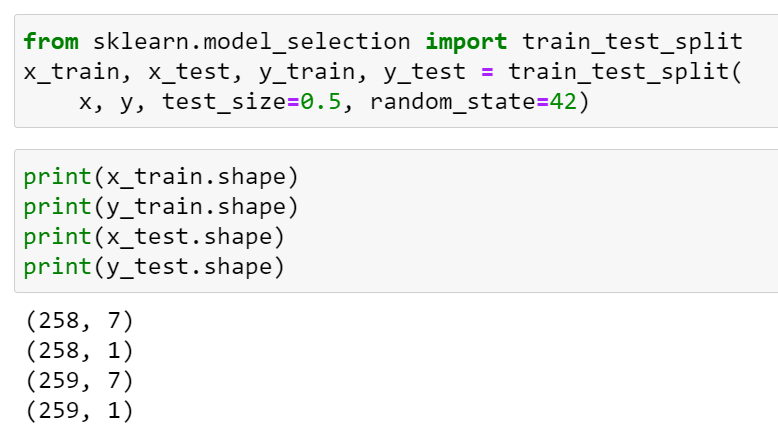
#### Adnotacja odnośnie podziału na zbiór testujący i trenujący

Podczas wykonywania zadań, zauważono że proporcja zbioru testującego do trenującego podczas dokonywania podziału ma duże znaczenie, dlatego podczas zbierania wyników uwzględniono trzy różne sposoby podzielenia:

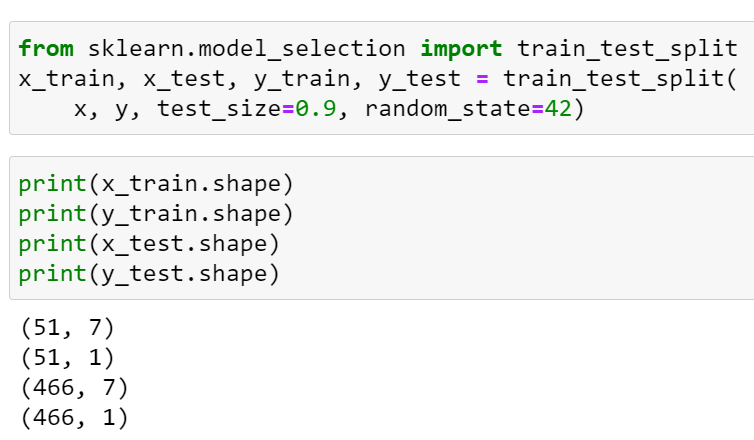
* testujący 10% i trenujący 90%: test\_size = 0.1
* testujący 50% i trenujący 50%: test\_size = 0.5
* testujący 90% i trenujący 10%: test\_size = 0.9



Obraz 15: Fragment kodu przedstawiający liczność zbiorów po podziale dla test\_size = 0.1



Obraz 16: Fragment kodu przedstawiający liczność zbiorów po podziale dla test\_size = 0.5

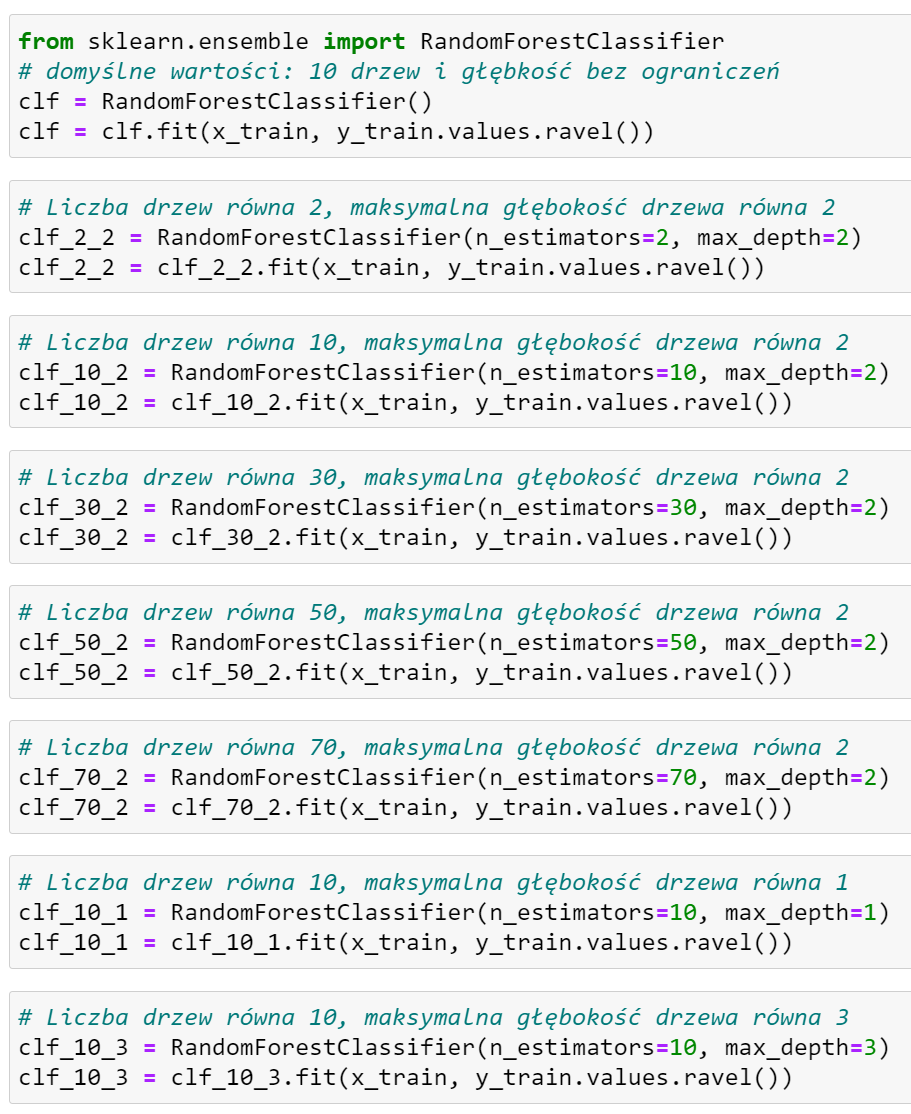


Obraz 17: Fragment kodu przedstawiający liczność zbiorów po podziale dla test\_size = 0.9

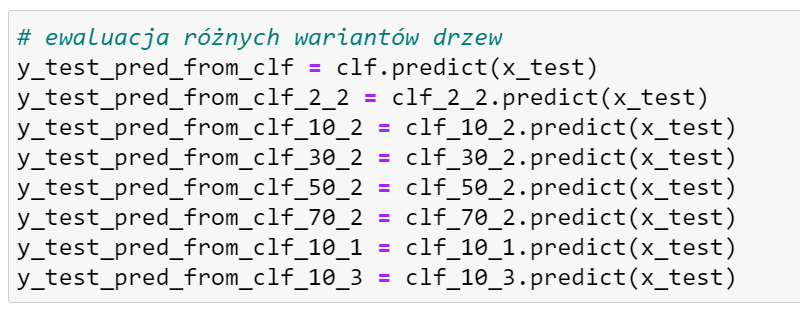
#### Budowanie lasów losowych

Zbudowane lasy:

* 2 drzewa o maksymalnej głębokości równej 2,
* 10 drzew o maksymalnej głębokości równej 2,
* 30 drzew o maksymalnej głębokości równej 2,
* 50 drzew o maksymalnej głębokości równej 2,
* 70 drzew o maksymalnej głębokości równej 2
* 10 drzew o maksymalnej głębokości równej 1,
* 10 drzew o maksymalnej głębokości równej 3.

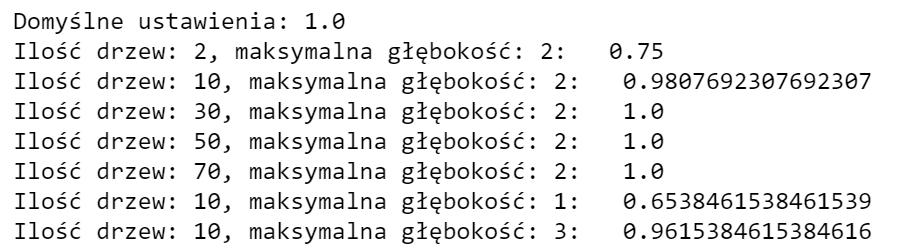


Obraz 18: Fragment kodu przedstawiający sposób tworzenia lasów losowych o różnych parametrach

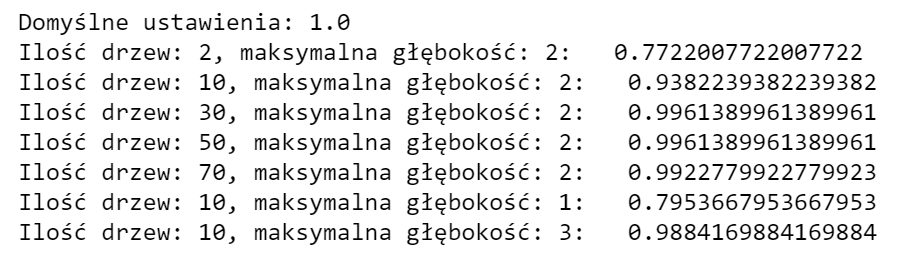


Obraz 19: Fragment kodu przedstawiający ewaluację utworzonych lasów losowych

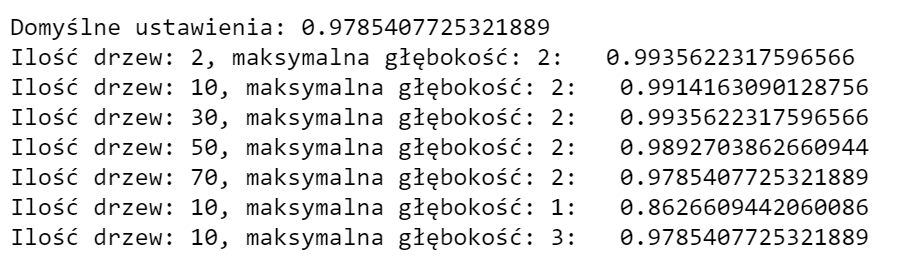
#### Porównanie wyników



Obraz 20: Fragment kodu przedstawiający wyniki dla test\_size = 0.1



Obraz 21: Fragment kodu przedstawiający wyniki dla test\_size = 0.5



Obraz 22: Fragment kodu przedstawiający wyniki dla test\_size = 0.9

Tabela 2: Wyniki dokładności modelu w zależności od ilości drzew  
 w lesie losowym oraz wielkości zbioru testowego przy **maksymalnej głębokości równej 2**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test\_size = 0.1** | **Test\_size = 0.5** | **Test\_size = 0.9** |
| **Domyślnie  (100 drzew i nieograniczona głębokość)** | 1.0 | 1.00 | 0.98 |
| **10 drzew** | 0.98 | 0.938 | 0.99 |
| **30 drzew** | 1.0 | 0.996 | 0.99 |
| **50 drzew** | 1.0 | 0.996 | 0.99 |
| **70 drzew** | 1.0 | 0.992 | 0.98 |

Tabela 3: Wyniki dokładności modelu w zależności od maksymalnej głębokości drzew w lesie losowym oraz wielkości zbioru testowego przy **ilości drzew w lasach losowych równej 10**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test\_size = 0.1** | **Test\_size = 0.5** | **Test\_size = 0.9** |
| **Domyślnie  (100 drzew i nieograniczona głębokość)** | 1.0 | 1.00 | 0.98 |
| **Głębokość = 1** | 0.65 | 0.80 | 0.86 |
| **Głębokość = 2** | 0.98 | 0.94 | 0.99 |
| **Głębokość = 3** | 0.96 | 0.99 | 0.98 |

## Wnioski

Podczas korzystania z klasyfikatora lasu losowego duże znaczenie ma wielkość zbioru testującego i maksymalna głębokość drzew decydujących. Na podstawie zebranych wyników można wywnioskować, że najbardziej optymalne jest użycie jak największego zbioru testującego i jak największej głębokości drzew decyzyjnych. Warto również zauważyć, że domyślne ustawienia klasyfikatora lasu losowego otrzymały najlepsze wyniki.

# Naiwny klasyfikator bayesowski

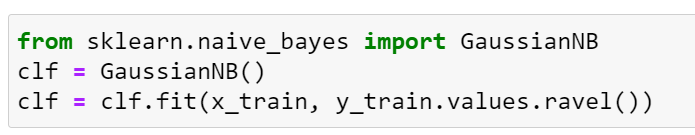
## Zadanie #4

### Treść polecenia

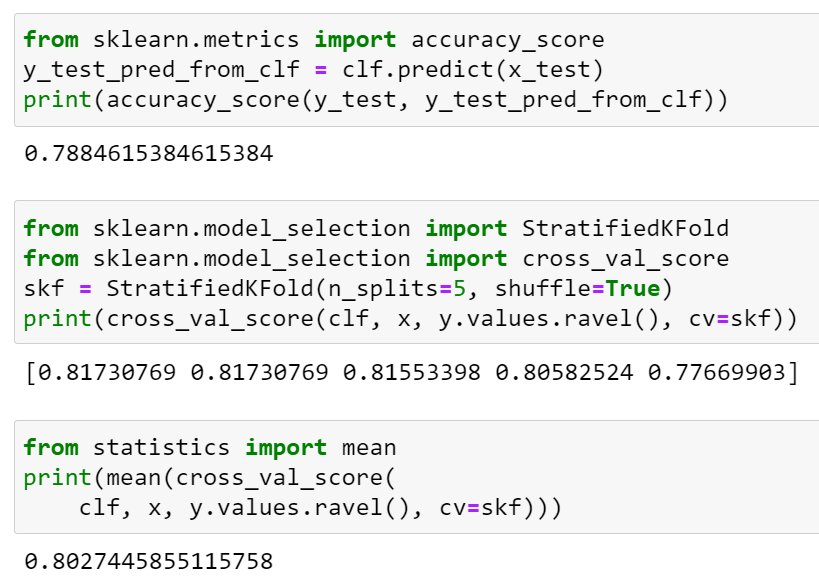
Proszę pobrać dowolny zbiór danych ze strony <https://archive.ics.uci.edu/ml/index.php> Następnie proszę podzielić zbiór na dane trenujące i testujące, wytrenować i przetestować klasyfikator GaussianNB. Proszę o sporządzenie sprawozdania z wnioskami.

### Rozwiązanie

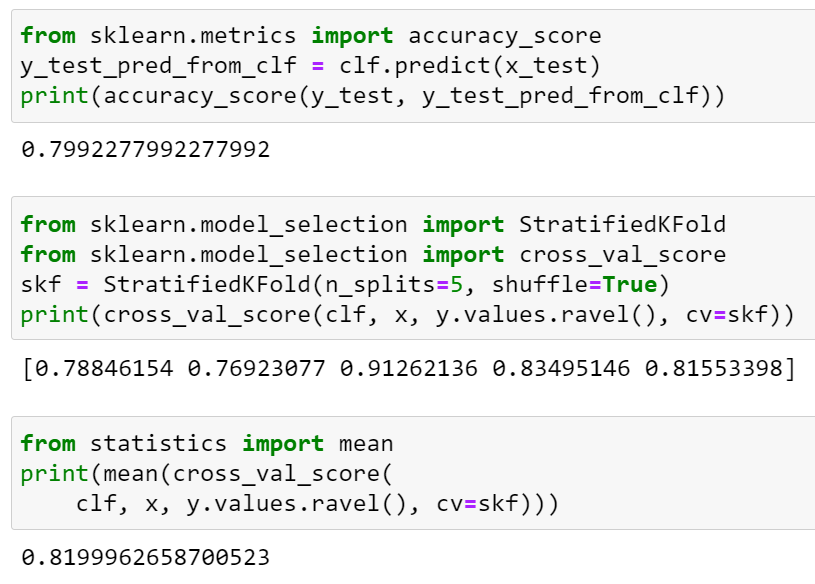
Podczas dokonywania obliczeń dla naiwnego klasyfikatora bayesowski również przetestowano 3 podziały na zbiór testujący i trenujący, jak w przypadku pomiarów dla [Zadanie #3](#_Zadanie #3).



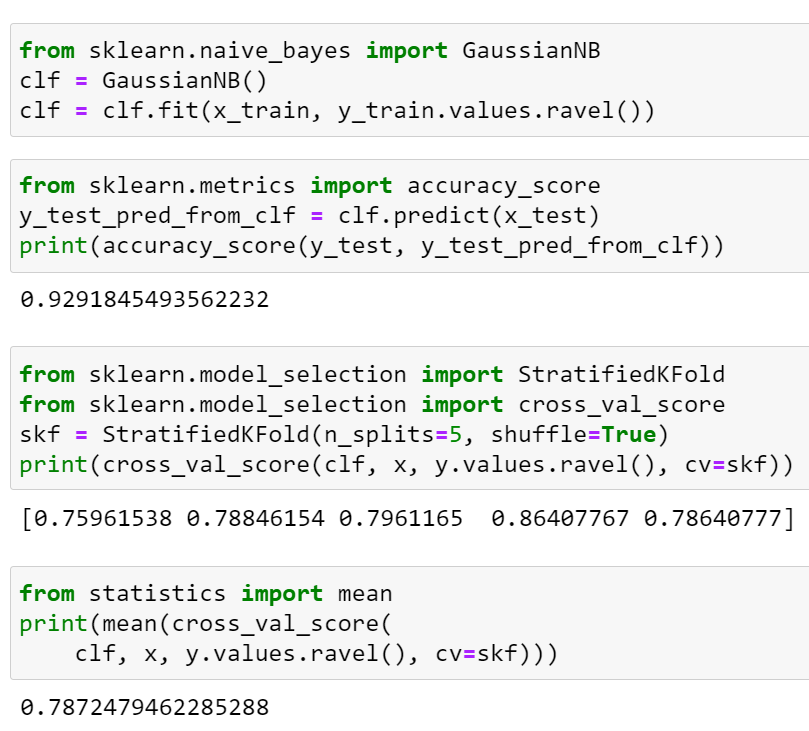
Obraz 23: Fragment kodu przedstawiający użycie naiwnego klasyfikatora bayesowskiego



Obraz 24: Fragment kodu przedstawiający wynik dokładności modelu dla test\_size - 0.1



Obraz 25: Fragment kodu przedstawiający wynik dokładności modelu dla test\_size - 0.5



Obraz 26: Fragment kodu przedstawiający wynik dokładności modelu dla test\_size - 0.9

Tabela 4: Wyniki dokładności modelu dla naiwnego klasyfikatora bayesowskiego   
w zależności od wielkości zbioru testującego

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test\_size = 0.1** | **Test\_size = 0.5** | **Test\_size = 0.9** |
| **Naiwny klasyfikator bayesowski** | 0.79 | 0.80 | 0.93 |

## Wnioski

Naiwny klasyfikator bayesowski jest szczególnie czuły na wielkość zbioru testującego, najlepiej sobie radzi gdy zbiór trenujący jest bardzo mały.

# Porównanie klasyfikatorów

Do porównania klasyfikatorów użyto najlepsze otrzymane wyniki dla poszczególnych klasyfikatorów.

Tabela 5: Zebranie wyników dokładności modelu dla wszystkich przetestowanych klasyfikatorów w zależności od wielkości zbioru testowego

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test\_size = 0.1** | **Test\_size = 0.5** | **Test\_size = 0.9** |
| **Klasyfikator binarny ze stratyfikacją i walidacją krzyżową** | 1.0 | 1.0 | 1.0 |
| **Klasyfikator lasu losowego (100 drzew i nieograniczona głębokość)** | 1.0 | 1.00 | 0.98 |
| **Naiwny klasyfikator bayesowski** | 0.79 | 0.80 | 0.93 |

# Wnioski końcowe

Dla wybranego zbioru danych najlepszym rozwiązaniem okazał się klasyfikator binarny ze stratyfikacją i walidacją krzyżową, bo uzyskał dokładność 100% niezależnie od wielkości zbioru testowego. Najgorszy wynik otrzymał naiwny klasyfikator bayesowski. Warto jednak mieć na uwadze, że sposób przypisywania klas w algorytmie klasyfikującym był losowy, co mogło mieć duży wpływ na działanie tych klasyfikatorów.