WSI – Ćwiczenie 7.

Modele bayesowskie

Dawid Bartosiak 318 361

---

1. Treść zadania

Należy zaimplementować naiwny klasyfikator bayesowski bez użycia dodatkowych bibliotek i zastosować go do zbadania załączonego zbioru danych. Konkretnie dla mnie jest to zadanie klasyfikacji 3 odmian wina rosnącego w tym samym rejonie Włoch na podstawie ich chemicznych parametrów. Zbiór tworzy 178 obserwacji – klasa pierwsza (59), klasa druga (71) i klasa trzecia (48). Link: https://archive.ics.uci.edu/ml/datasets/Wine.

2. Przygotowanie danych

Do pracy nad danymi wykorzystałem bibliotekę pandas. Do tabeli z danymi wczytuję zawartość pliku *wine.data*. Do programu można załadować dowolny plik, o ile miałby taki sam format, a klasy były w pierwszej kolumnie.

3. Obliczenie statystyk

Obliczenie statystyk do użycia w klasyfikatorze odbywa się poprzez użycie następujących klas:

1. *split_by_class* – podzielenie całego zbioru danych na podzbiory względem klasy. W ten sposób otrzymujemy tyle podzbiorów, ile występuje różnych klas. Funkcja automatycznie usuwa kolumnę zawierającą klasy z każdego z podzbiorów. Zwracana jest lista z podzbiorami oraz lista klas. Kolejność pozbiorów odpowiada kolejności listy klas, dzięki czemu możemy dopasować zbiór do klasy poprzez podanie jednakowego indeksu.
2. *stats_for_column* – funkcja ta służy do obliczania statystyk dla pojedynczej kolumny, przyjmując jej wartości jako parametr. Na podstawie tych danych, funkcja oblicza średnią, odchylenie standardowe oraz liczbę elementów, zwracając te informacje w formie listy.
3. *stats_for_dataset* – dla każdej kolumny w danym zbiorze, ta funkcja wykorzystuje *stats_for_column* do obliczenia statystyk. Zwraca ona kompletną listę statystyk, gdzie długość listy odpowiada liczbie kolumn w zbiorze. Funkcja ta zakłada, że przekazany zbiór danych nie zawiera już kolumny z etykietami klas, co jest istotne, ponieważ statystyki nie powinny być liczone dla samych etykiet klas.
4. *Stats_for_classes – Łącząc działanie funkcji split_by_class oraz stats_for_dataset, ta funkcja oblicza statystyki dla każdego z podzbiorów utworzonych z głównego zbioru danych. W ten sposób uzyskujemy kompleksowy zestaw statystyk, który odzwierciedla charakterystykę każdej klasy w zbiorze danych.*

4.Obliczanie prawdopodobieństwa

Klasyfikacja za pomocą modelu bayesowskiego polega na obliczeniu prawdopodobieństwa, z jakim podane dane można przypisać do danej klasy. Liczy się to wykorzystując prawdopodobieństwo warunkowe. Wartość prawdopodobieństwa dla danej klasy inicjujemy prawdopodobieństwem jej wystąpienia w zbiorze danych w ogóle (dzielimy liczbę jej wystąpień przez liczbę wszystkich rzędów w tabeli). Następnie mnożymy ją po kolei przez wartości otrzymane ze wzoru:

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

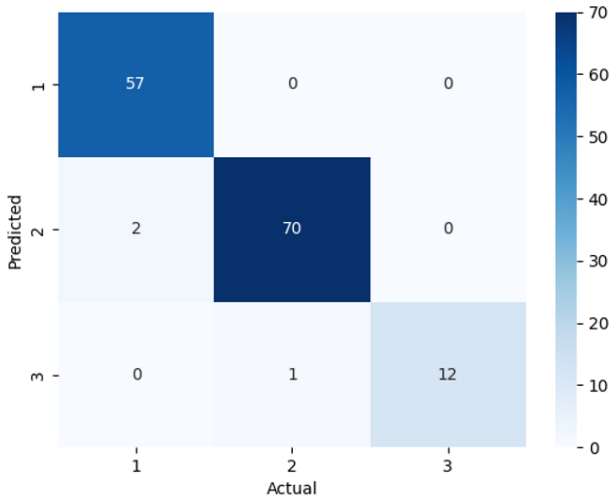
dla statystyk wszystkich pozostałych kolumn. Wykorzystujemy tu statystyki z funkcji *stats_for_classes.*
Funkcja *predict_probabilities* oblicza listę prawdopodobieństw dla podanej listy danych. Funkcja *predict_class* wybiera największe otrzymane prawdopodobieństwo i łączy je z nazwą klasy.

5.Klasyfikacja dla zbiorów danych

Wszystkie wyżej wymienione funkcje zostają połączone w funkcji *predict_dataset.* Jako argumenty przyjmuje 2 zbiory danych: treningowy, na podstawie którego liczy statystyki dla klas, oraz testowy. Kolejne wiersze zbioru testowego dzielone są na wartości poddane klasyfikacji oraz oczekiwane klasy. Lista przewidywanych klas oraz oczekiwanych klas jest wynikiem działania funkcji.

6.Macierz pomyłek

Na podstawie listy przewidywanych klas i listy oczekiwanych klas generuję macierz pomyłek. Wykorzystałem do tego bibliotekę pandas, seaborn oraz matplotlib. Tworzona jest macierz wyświetlana jako wykres.



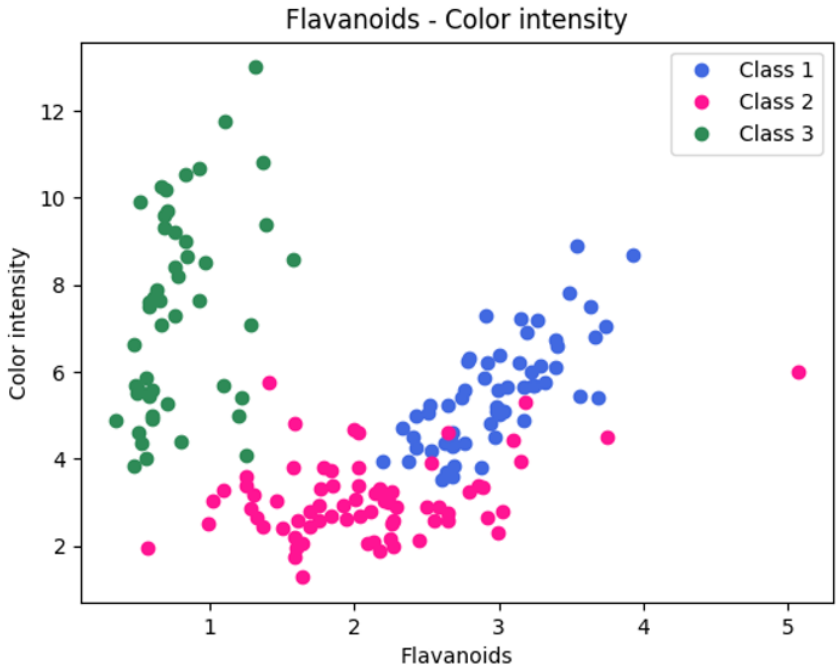
Rysunek 1. Przykładowa macierz pomyłek

7.Sprawdzenie danych

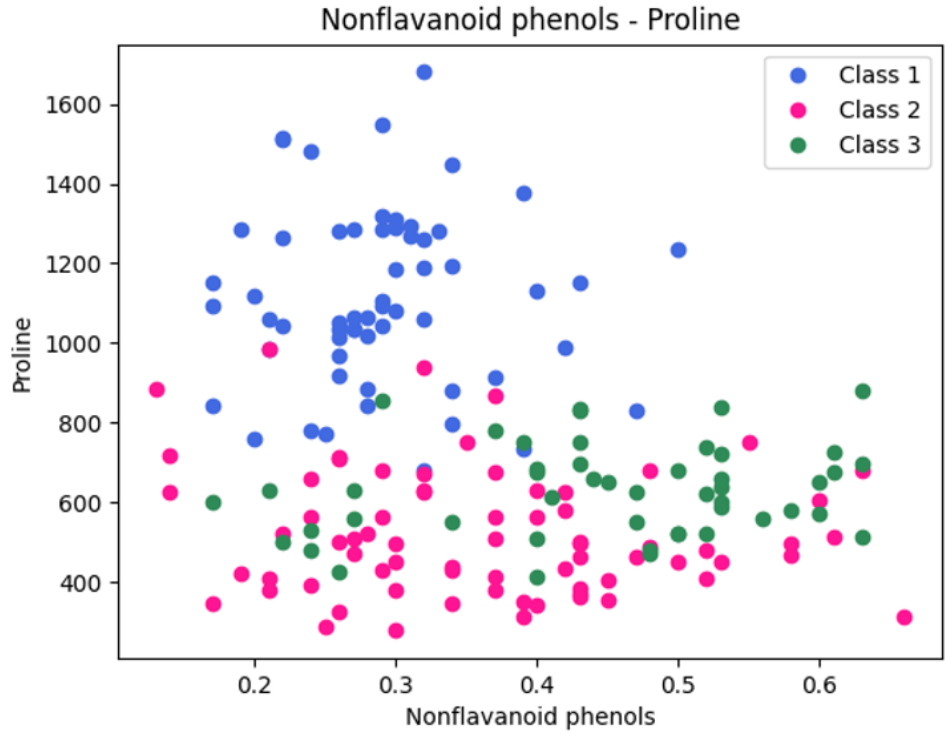
W ramach analizy danych winnych, przeprowadziłem serię wizualizacji, tworząc wykresy zależności między różnymi parametrami chemicznymi. Każdą parę cech zestawiłem ze sobą, tworząc wykresy zależności każdy z każdym. Podczas tej analizy zaobserwowałem interesujące wzorce:

- **Wyraźne Podgrupy**: Dla niektórych par cech, dane wyraźnie grupowały się w odrębne podzbiory, co sugeruje silną korelację między tymi cechami a klasami win. Te wyraźne podziały mogą znacząco przyczynić się do skuteczności klasyfikacji.
- **Zlewające się Dane**: Z drugiej strony, niektóre pary cech nie wykazywały wyraźnego podziału, co skutkowało zlewaniem się danych w jednolitą 'plamę'. Brak rozróżnienia w tych obszarach może stanowić wyzwanie dla procesu klasyfikacji.

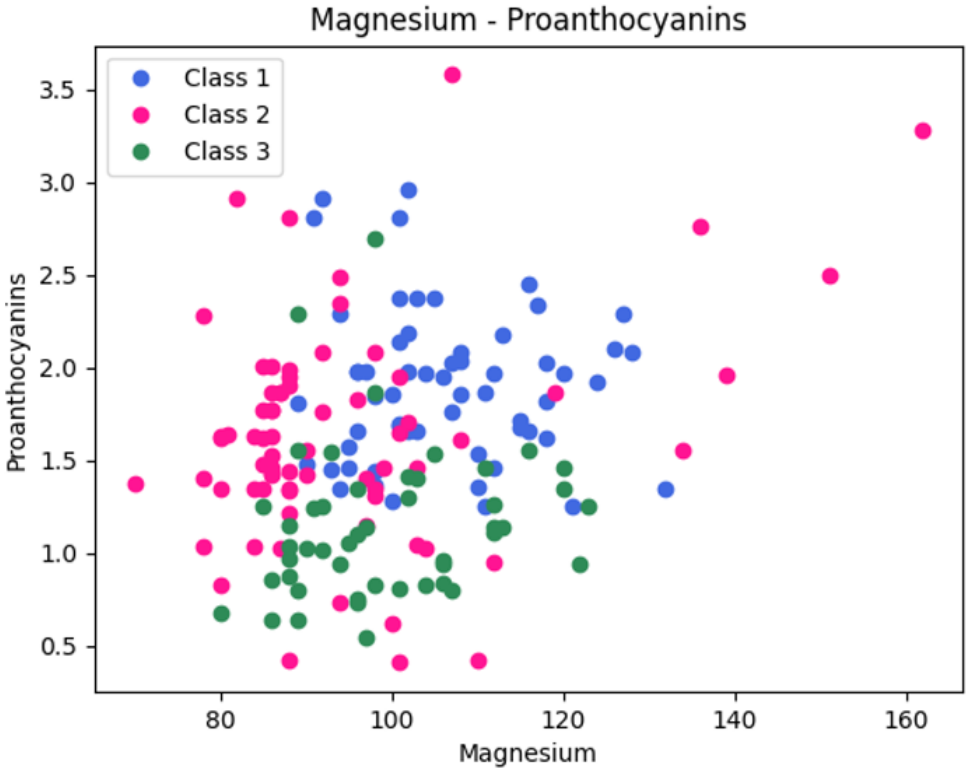
Kluczowym wnioskiem z tej analizy jest to, że klasyfikator Bayesa, poprzez łączenie informacji z różnych kategorii, jest w stanie wykorzystać nawet subtelne różnice między klasami. Dzięki temu, nawet w przypadkach, gdy poszczególne cechy indywidualnie nie pozwalają na jednoznaczne rozróżnienie klas, klasyfikator może osiągnąć wysoką skuteczność poprzez analizę kombinacji różnych cech.



Rysunek 2. Klasy łatwo rozróżnialne.



Rysunek 3. Klasa 2 i 3 podobne, 1 łatwo odróżnić



Rysunek 4. Wszystkie klasy są podobne.

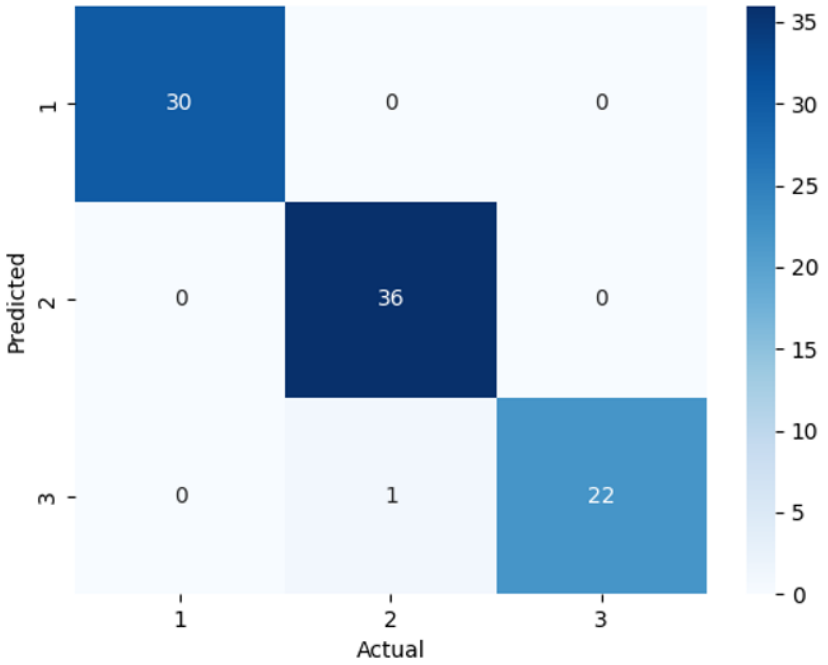
8.Testowanie algorytmu

Zdecydowałem się przeprowadzić testy w zależności od dwóch parametrów: proporcji wielkości zbioru testowego i treningowego oraz tego, czy zbiór danych jest początkowo pomieszany.

Tabela 1: Porównanie macierzy błędu dla różnych wielkości zbioru testowego w podziale na posortowanie

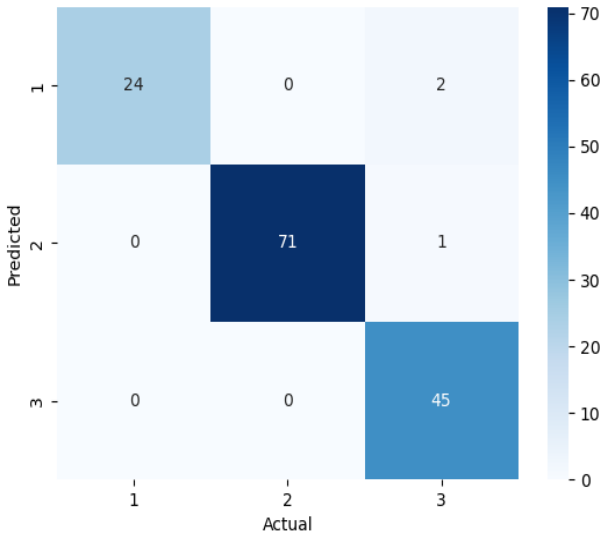
Shuffled	Test Data Proportion	Recall	Fall_out	Precision	Accuracy	F1_score
0	0,1	0,9906	0,0094	0,9906	0,9813	0,9906
1	0,1	1	0	1	1	1
0	0,2	0,9894	0,0106	0,9894	0,9790	0,9894
1	0,2	1	0	1	1	1
0	0,3	0,7624	0,2376	0,7624	0,6160	0,7624
1	0,3	0,9930	0,0070	0,9930	0,9861	0,9930
0	0,4	0,7108	0,2761	0,7239	0,5654	0,7173
1	0,4	0,9947	0,0053	0,9947	0,9897	0,9947
0	0,5	0,6308	0,3594	0,6407	0,4719	0,6357
1	0,5	0,9953	0,0047	0,9953	0,9907	0,9953
0	0,6	0,3256	0,3	0,7	0,5139	0,4444
1	0,6	0,9953	0,0047	0,9953	0,9907	0,9953
0	0,7	0,0	0,0	0,0	0,5	0,0
1	0,7	0,9919	0,0081	0,9919	0,9840	0,9919
0	0,8	0,0	0,0	0,0	0,5	0,0
1	0,8	0,9894	0,0106	0,9894	0,9790	0,9894
0	0,9	0,0	0,0	0,0	0,5	0,0
1	0,9	0,9338	0,0662	0,9338	0,8758	0,9338

Możemy zauważyć że klasyfikator osiąga fenomenalne wyniki, jeżeli pracuje na pomieszanych danych. Nawet jeżeli zbiór treningowy ma wielkość 10% całego zbioru, osiągana precyzja oscyluje na poziomie 93%. Oczywiście im większy zbiór treningowy, tym lepiej wyuczony klasyfikator i tym lepsze osiągi.

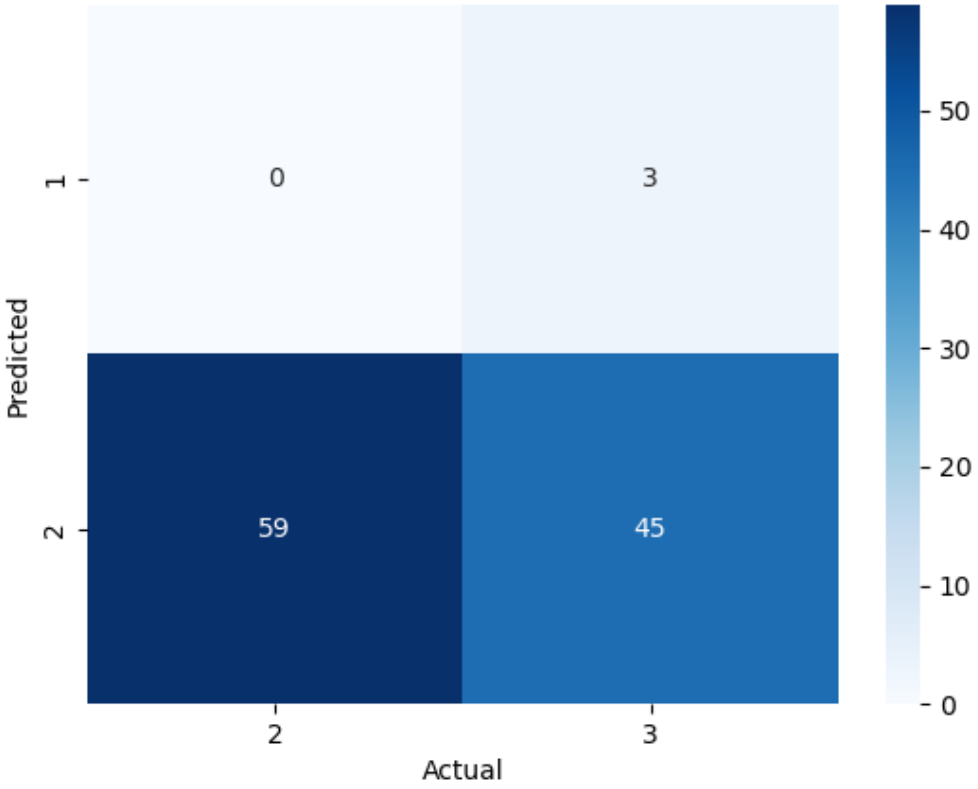


Rysunek 5. Macierz pomyłek dla klasyfikacji danych pomieszanych przy proporcji zbioru testowego do treningowego 1:1

Wyniki klasyfikacji okazują się być znacznie słabsze, gdy klasyfikatorowi przedstawiony jest zbiór danych, który nie został wcześniej pomieszany. W takim przypadku, dane są uporządkowane według klas, co oznacza, że początkowe wiersze reprezentują pierwszą klasę, po nich następują wiersze klasy drugiej, a na końcu trzeciej. W sytuacji, gdy zbiór testowy jest niewielki, klasyfikator nadal wykazuje zadowalającą skuteczność. Problem pojawia się jednak w przypadku małego zbioru treningowego. Zmniejszając jego rozmiar, stopniowo tracimy reprezentację klas drugiej i trzeciej, co prowadzi do obniżenia precyzji klasyfikacji. Brak wystarczającej różnorodności w zbiorze treningowym sprawia, że klasyfikator nie jest w stanie efektywnie nauczyć się charakterystyk wszystkich klas, co negatywnie wpływa na jego zdolność do poprawnego przewidywania klasyfikacji.



Rysunek 6. Macierz pomyłek klasyfikacji przy niepomieszanych danych i proporcji zbioru treningowego do testowego 4:1



Rysunek 7. Macierz pomyłek klasyfikacji przy niepomieszanych danych i proporcji zbioru treningowego do testowego 3:2

Na powyższym wykresie mamy przykład klasyfikacji, kiedy w zbiorze treningowym znajdowały się wiersze tylko i wyłącznie klas 1 i 2, a w zbiorze testowym: 2 i 3. Klasyfikator nie ma problemu z rozpoznaniem drugiej klasy. Trzeciej klasy nigdy nie spotkał, ale wyniki nie pasują mu do klasy numer jeden, więc wiersze trzeciej klasy rozpoznaje jako wiersze drugiej klasy.