WSI Laboratorium 7
Naiwny klasyfikator Bayesa

Filip Misztal 310276

24 stycznia 2024

1 Opis Implementowanych Algorytmów

Działanie naiwnego klasyfikatora Bayesa (pseudokod)

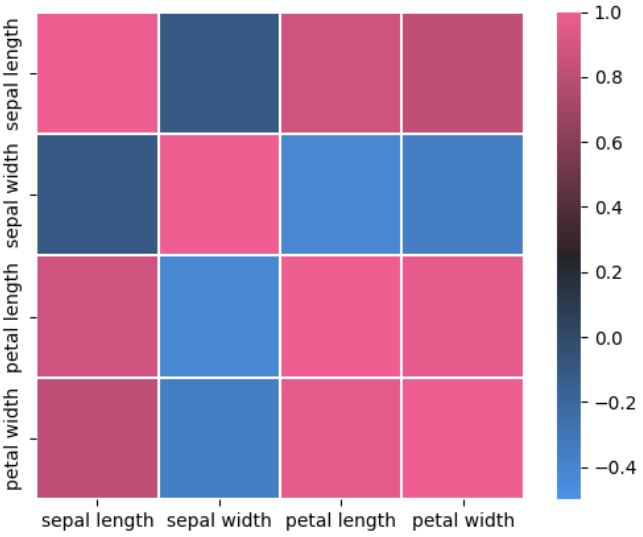
1. Inicjalizuj unikalne klasy oraz prawdopodobieństwa klas:
   $P(C_i) = \frac{N_i}{N}$, gdzie N_i to liczba przykładów klasy C_i, a N to ogólna liczba przykładów
2. Dla każdej klasy:
   (a) Wyodrębnij dane przypisane do danej klasy z danych treningowych X i etykiet y
   (b) Dla każdej cechy w danych treningowych:
      i. Oblicz parametry rozkładu (mean, b) za pomocą wybranego rozkładu (gaussian/laplace) w danej klasie
      ii. Zapisz parametry rozkładu jako parametry dla danej cechy w danej klasie
3. Zapisz unikalne klasy, prawdopodobieństwa klas ($P(C_i)$), i parametry rozkładów jako wytrenowane parametry klasyfikatora
4. Dla nowego przykładu X:
   (a) Dla każdej klasy C_i:
      i. Inicjalizuj wyniki klas na 1.0
      ii. Dla każdej cechy x_j w X: - Oblicz $P(x_j|C_i)$ za pomocą odpowiedniego rozkładu (gaussian/laplace) - Pomnóż wynik klasy przez $P(x_j|C_i)$
      iii. Pomnóż wynik klasy przez $P(C_i)$
   (b) Wybierz klasę C_i z najwyższym wynikiem jako przewidywaną klasę dla X

2 Opis planowanych eksperymentów numerycznych

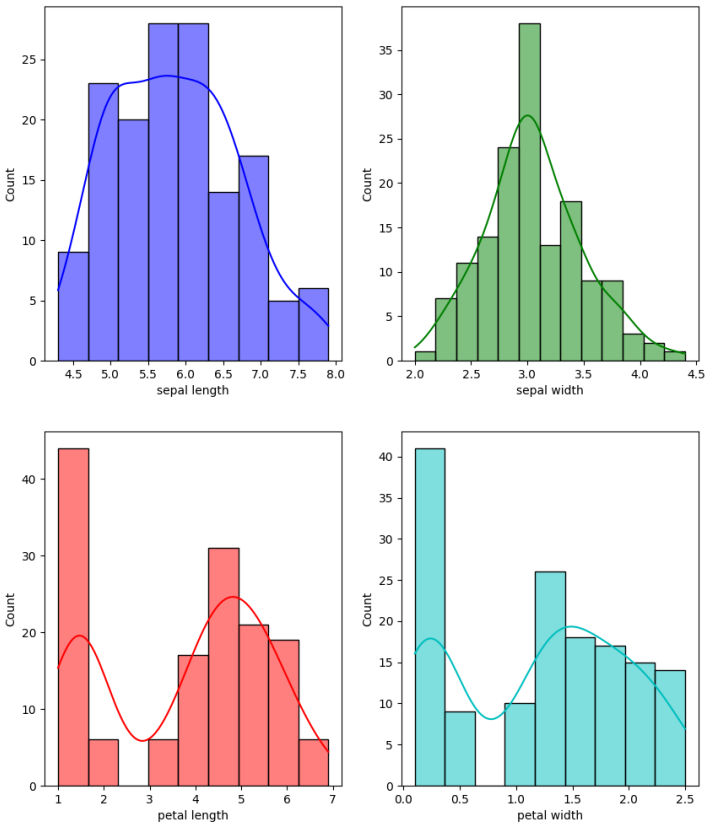
W celu przetestowania możliwości zaimplementowanego algorytmu przeprowadziłem test badający dokładność, czułość i specyficzność wytrenowanego klasyfikatora w zależności od zastosowanego rozkładu. Aby zmniejszyć wagę czynnika losowego przy podziale zbioru na dane treningowe i testowe, przeprowadziłem analizę także metodą corss-validation. Wcześniej sprawdziłem również, czy zbiór danych nadaje się do użycia przez tego typu algorytm. Porównałem róznież działanie zaimplementowanego algorytmu z gotowym rozwiązaniem z biblioteki sklearn oraz zbadałem jak na jakość klasyfikacji wpłynęło usunięcie silnie korelującej cechy ze zbioru.

3 Opis uzyskanych Wyników

Głównym wymogiem co do danych treningowych dla naiwnego klasyfikatora Bayesa jest to, żeby cechy nie były ze sobą silnie skorelowane. Dlatego też na początku sprawdziłem zależności między cechami w badanym zbiorze iris.

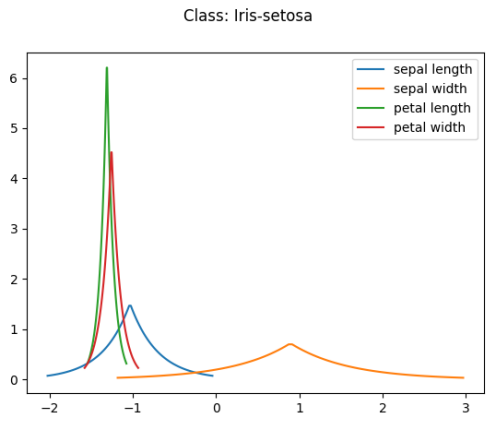


Rysunek 1: Macierz korelacji cech

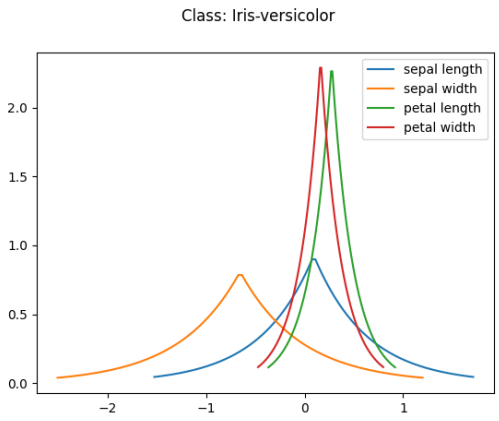


Rysunek 2: Rozkład wartości cech

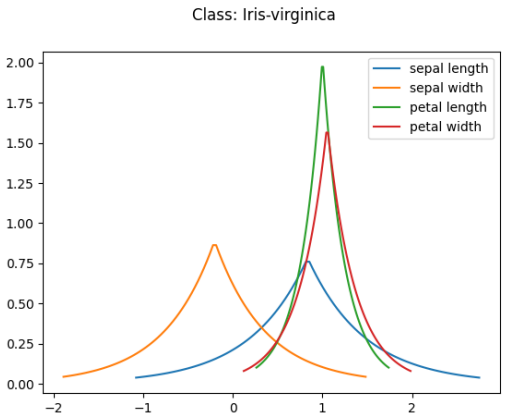
Jak widać na powyższych wykresach, chechy *petallength* i *petalwidth* dosyć mocno ze sobą korelują. Trudno jest także dopasować do tych klas jakikolwiek typ rozkładu, który by je dobrze reprezentował. Mimo to przeszedłem do dalszych testów aby zbadać, czy będzie to miało negatywny wpływ na działanie klasyfikatora. W tym celu podzieliłem zbiór na dane treningowe i testowe, a następnie wytrenowałem modele przy zasosowaniu rozkładu normalnego Gaussa, Laplace'a i Cauchy'ego oraz wyznaczyłem ich czułość, dokładność i specyficzność.
Prezentacja wyników osiąganych przez wytrenowane modele dla obu dostępnych rozkładów, przy przeznaczeniu na dane treningowe 80% zbioru:



(a)

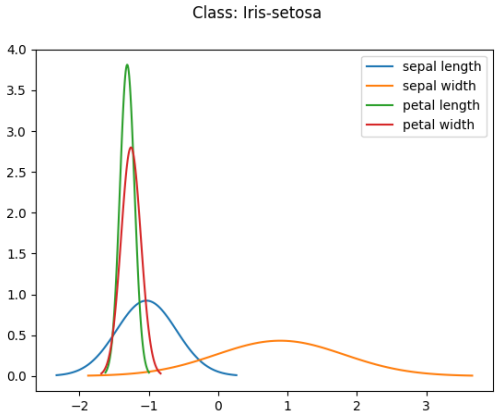


(b)

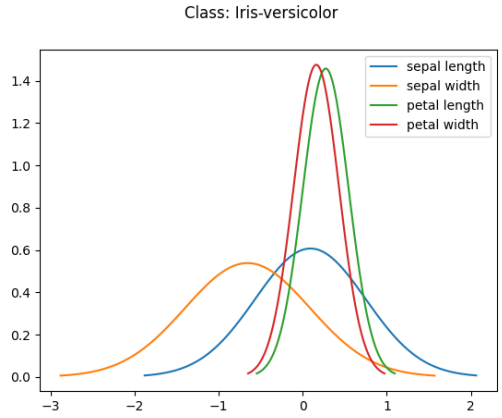


(c)

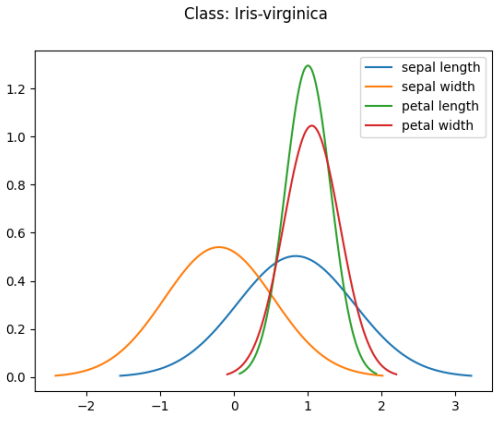
Rysunek 3: Rozkłady Laplace'a poszczególnych cech dla każdej klasy



(a)

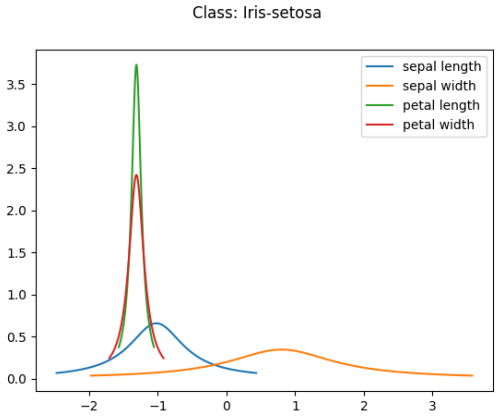


(b)

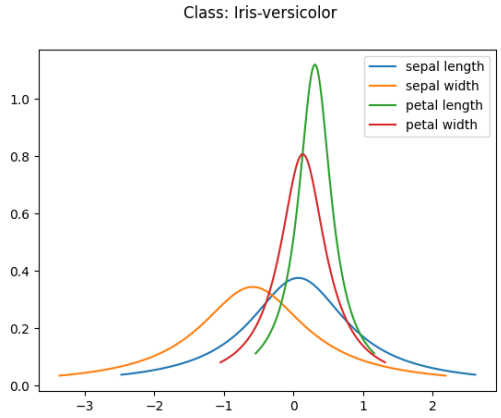


(c)

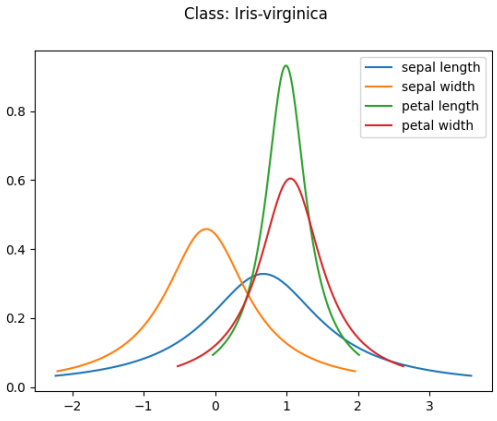
Rysunek 4: Rozkłady Gaussa poszczególnych cech dla każdej klasy



(a)

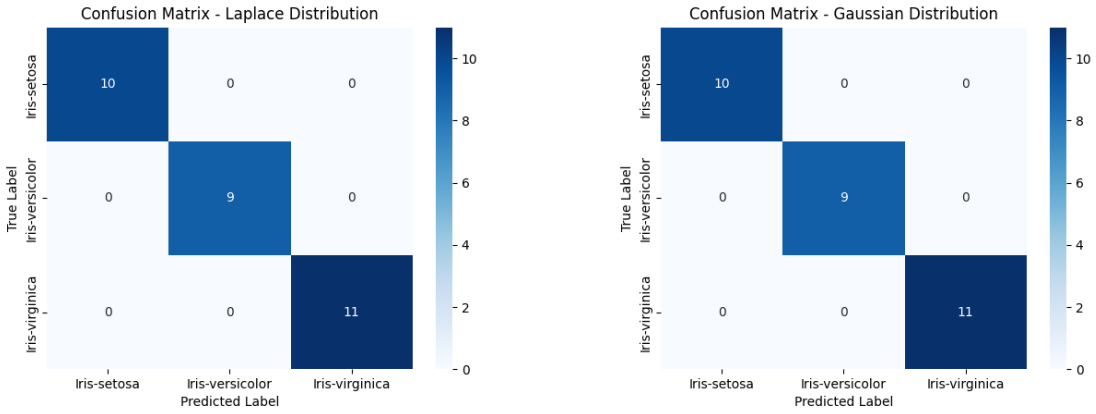


(b)



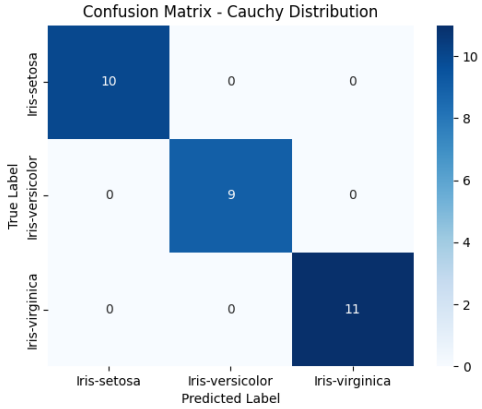
(c)

Rysunek 5: Rozkłady Cauchy'ego poszczególnych cech dla każdej klasy



(a)

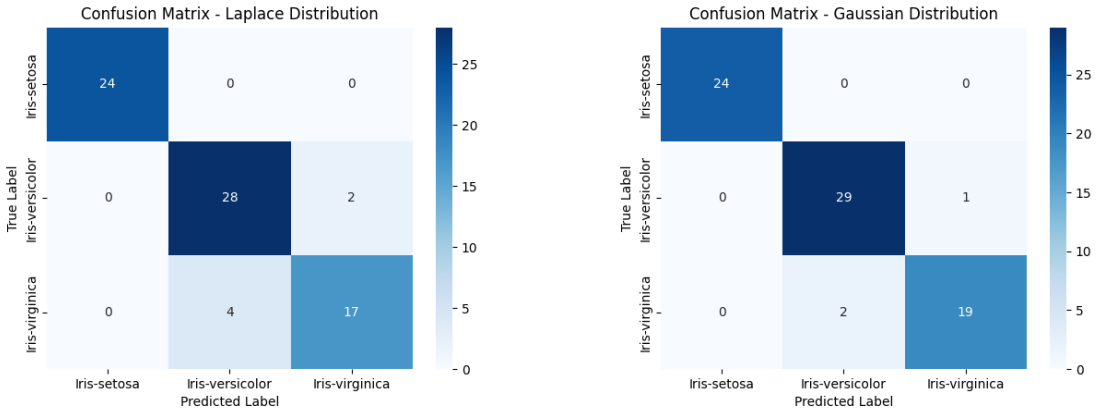
(b)



(c)

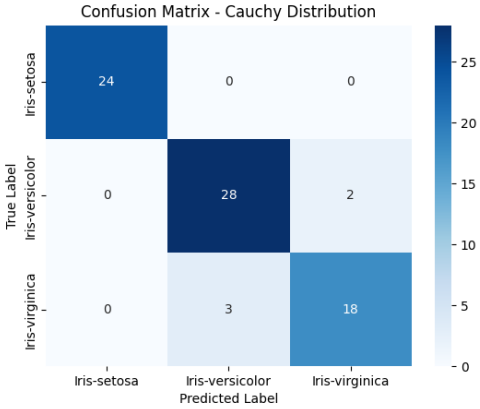
Rysunek 6: Macierze pomyłek wytrenowanych modeli przy rozmiarze zbioru treningowego 80% całości

Jak widać, modele w obu wariantach poradziły sobie z zadaniem ze 100% skutecznością. Aby je im utrudnić, zmniejszyłem rozmiar zbioru treningowego do 50% całości:



(a)

(b)



(c)

Rysunek 7: Macierze pomyłek wytrenowanych modeli przy rozmiarze zbioru treningowego 50% całości

Tutaj możemy już zaobserwować pewne różnice w modelach w zależności od użytego rozkładu. Wyliczone współczynniki:

1. Model korzystający z rozkładu Laplace'a:
   (a) Dokładność: 92%
   (b) Czułość: [100%, 93.33%, 80.95%]
   (c) Specyficzność: [100%, 91.11%, 96.3%]
2. Model korzystający z rozkładu Gaussa:
   (a) Dokładność: 96%
   (b) Czułość: [100%, 96.67%, 90.48%]
   (c) Specyficzność: [100%, 95.56%, 98.15%]
3. Model korzystający z rozkładu Cauchy'ego:
   (a) Dokładność: 93.33%
   (b) Czułość: [100%, 93.33%, 85.71%]
   (c) Specyficzność: [100%, 93.33%, 96.3%]

Z powyższych danych wynika, że z klasyfikacją zadanego zbioru najlepiej radzi sobie model wytrenowany w oparciu o rozkład normalny Gaussa. Nadal jednak brakowało pewności, czy nie jest to spowodowane jedynie fortunnym dla niego podziałem zbioru na dane treningowe i testowe. Zdecydowałem się więc przeprowadzić badanie cross-validation wyciągając wartości średnie współczynników z 20 prób, każda dla innego podziału zbioru danych:

1. Model korzystający z rozkładu Laplace'a:
   (a) Dokładność: 94.64%
   (b) Czułość: [100%, 91.67%, 91.67%]
   (c) Specyficzność: [100%, 95.83%, 96%]
2. Model korzystający z rozkładu Gaussa:
   (a) Dokładność: 95.36%
   (b) Czułość: [100%, 93.33%, 91.67%]
   (c) Specyficzność: [100%, 95.83%, 97%]
3. Model korzystający z rozkładu Cauchy'ego:
   (a) Dokładność: 95.36%
   (b) Czułość: [100%, 93.33%, 91.67%]
   (c) Specyficzność: [100%, 95.83%, 97%]

Na podstawie tego badania można wyciągnąć wniosek, że zastosowanie rozkładu Gaussa i Cauchy'ego daje bardzo zbliżone wyniki, natomiast rozkład Laplace'a wypada od nich nieznacznie gorzej.

Żeby ostatecznie przekonać się o poprawności implementacji algorytmu, porównałem jego osiągi z wbudowanym algorytmem GaussianNB z biblioteki sklearn. Przetestowałem go w ten sam sposób co poprzednio mój algorytm używający rozkładu Gaussa, zarówno w teście po wytrenowaniu na 50% danych treningowych jak i teście cross-validation. Otrzymałem identyczne rezultaty, co uznaję za potwierdzenie, że algorytm działa w pełni poprawnie.

Na koniec przetestowałem jeszcze działanie algorytmu na zbiorze po usunięciu jednej z silnie korelujących cech (*petal length*):

```
======= DISTRIBUTION COMPARISON TEST =======
Results for Laplace Distribution:
Accuracy: 0.8933
Sensitivity: [1.     0.8333 0.8571]
Specificity: [1.     0.9333 0.9074]

Results for Gaussian Distribution:
Accuracy: 0.9333
Sensitivity: [1.     0.9333 0.8571]
Specificity: [1.     0.9333 0.963 ]

Results for Cauchy Distribution:
Accuracy: 0.9333
Sensitivity: [1.     0.9    0.9048]
Specificity: [1.     0.9556 0.9444]
```

(a)

```
========== CROSS-VALIDATION TEST ==========
Results for Laplace Distribution:
Mean Accuracy: 0.9402
Mean Sensitivity: [1.     0.8917 0.9167]
Mean Specificity: [1.     0.9583 0.95  ]

Results for Gaussian Distribution:
Mean Accuracy: 0.9402
Mean Sensitivity: [1.     0.8917 0.9167]
Mean Specificity: [1.     0.9583 0.95  ]

Results for Cauchy Distribution:
Mean Accuracy: 0.9402
Mean Sensitivity: [1.     0.8917 0.9167]
Mean Specificity: [1.     0.9583 0.95  ]
```

(b)

Rysunek 8: Wyniki badań analogicznych do wykonywanych poprzedno z wyłączeniem silnie korelującej cechy *petal length* ze zbioru

Jak można zauważyć z powyższych pomiarów, we wszsystkich wariantach modele wytrenowane na zbiorze z wyłączeniem korelującej cechy poradziły sobie gorzej.

4 Wnioski z przeprowadzonych badań

1. Mimo korelacji między cechami, naiwny klasyfikator Bayesa z rozkładem prawdopodobieństwa wykazał się bardzo dobrą skutecznością.

2. Pracując na zbiorze danych iris ciężko było zaobserwować bardzo znaczące różnice w skuteczności modeli w zależności od użytego rozkładu. Możliwe, że zbiór o większej ilości danych dałby większe możliwości w tej kwestii.

3. Usunięcie silnie korelującej cechy ze zbioru w celu spełnienia warunków poprawnego działania algorytmu naiwnego nie zawsze musi przynosić pozytywne skutki. W tym przypadku taka zmiana wpłyneła negatywnie na osiągi algorytmu. Mogło to wynikać np. z utraty istotnych informacji albo zaburzeniem równowagi między klasami.

4. W przypadku, gdy zbiór danych treningowych jest stosunkowo niewielki, Naiwny klasyfikator Bayesa w oparciu o rozkład Gaussa radzi sobie najlepiej.

5. W oparciu o zebrane statystyki rekomendowałbym korzystanie z rozkładu Gaussa w przypadku zbiorów danych o podobnej charakterystyce do iris.