Wstęp do sztucznej inteligencji

LABORATORIUM 7 – NAIWNY MODEL BAYESOWSKI

Bartosz Latosek

310 790

1. Wprowadzenie

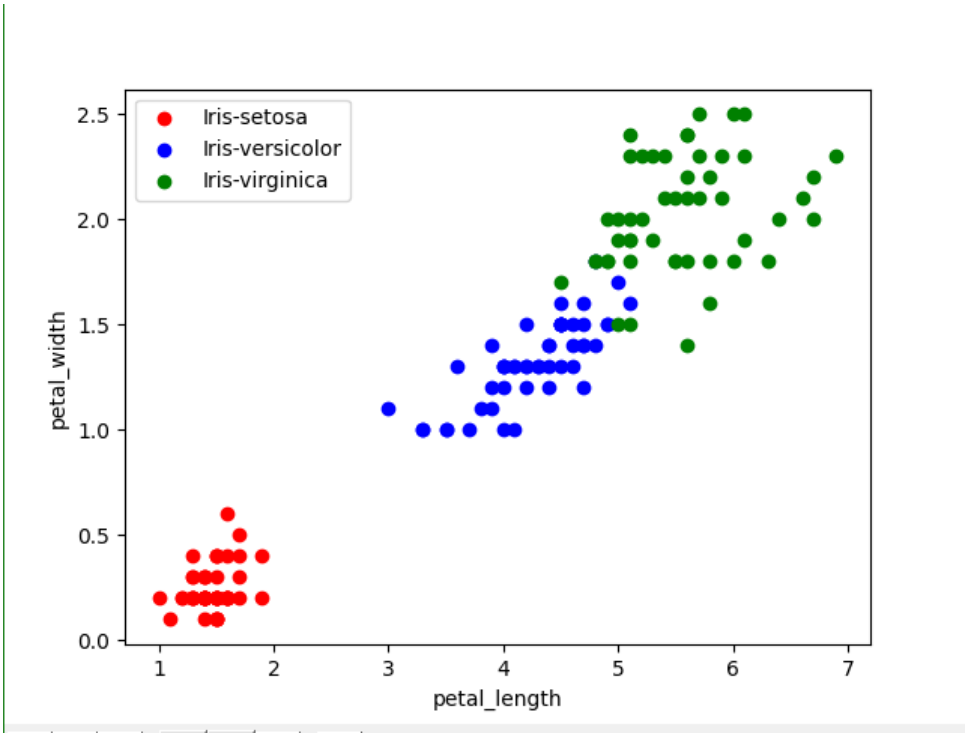
Zadanie polega na stworzeniu programu do klasyfikacji za pomocą modelu Bayesowskiego. Implementowany będzie naiwny klasyfikator Bayesa, umożliwiający klasyfikację ciągłych wartości atrybutów obiektów.

1.1 Opis badanego zbioru danych

Dane reprezentują różne odmiany kwiatu Irysa, dane za pomocą 4 atrybutów: *sepal length, sepal width, petal length i petal width*. Dany osobnik może należeć do jednej z 3 klas: “Iris-setosa”, “Iris-versicolor” i “Iris-virginica”. W testowanym zbiorze danych jest po równo przedstawicieli każdej klasy, jednak program został uogólniony w przypadku podania innego zbioru danych.

1.2 Analiza danych

W tym podpunkcie sprawdzę, czy za pomocą dwóch atrybutów jesteśmy w stanie zdefiniować liniową separowalność pomiędzy osobnikami ze zbioru testowego.



Rys 1. Zależność między petal_width a petal_length dla osobników

Na powyższym wykresie dokładnie widać liniową separowalnosc pomiędzy klasami w zależności od wyżej wymienionych atrybutów.

2. **Analiza Algorytmu**

Analiza algorytmu zostanie przeprowadzona na podstawie 2 różnych przypadków: posortowanego zbioru danych i zbioru danych pomieszanych. Parametr *test_train_proportion* oznacza proporcję zbioru testowego do zbioru trenującego.

2.1 Pomieszany zbiór danych

test_train_proportion = **0.1**

	Iris-versicolor	Iris-virginica	Iris-setosa
Iris-versicolor	6	0	0
Iris-virginica	2	9	0
Iris-setosa	0	0	13

Rec: 93.33% Fall: 3.33% Prec: 93.33% Acc: 95.56% F1: 93.33%

Rys 2. Macierz błędów i pomiary dla test_train_proportion = 0.1

test_train_proportion = **0.3**

	Iris-virginica	Iris-versicolor	Iris-setosa
Iris-virginica	13	5	0
Iris-versicolor	1	13	1
Iris-setosa	0	0	12

Rec: 84.44% Fall: 7.78% Prec: 84.44% Acc: 89.63% F1: 84.44%

Rys 3. Macierz błędów i pomiary dla test_train_proportion = 0.3

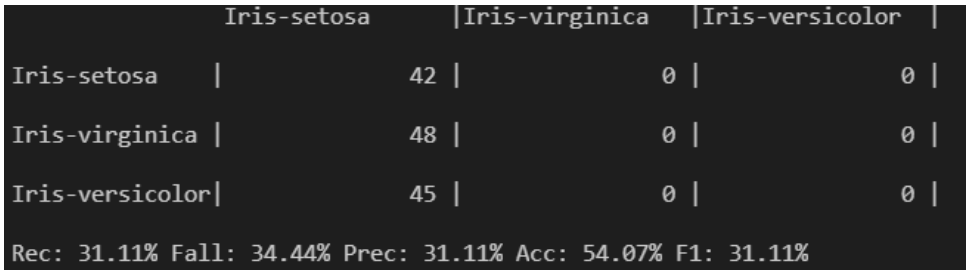
test_train_proportion = **0.5**

	Iris-setosa	Iris-versicolor	Iris-virginica
Iris-setosa	22	0	0
Iris-versicolor	5	20	1
Iris-virginica	1	8	18

Rec: 80.0% Fall: 10.0% Prec: 80.0% Acc: 86.67% F1: 80.0%

Rys 4. Macierz błędów i pomiary dla test_train_proportion = 0.5

test_train_proportion = **0.9**



Rys 5. Macierz błędów i pomiary dla test_train_proportion = 0.9

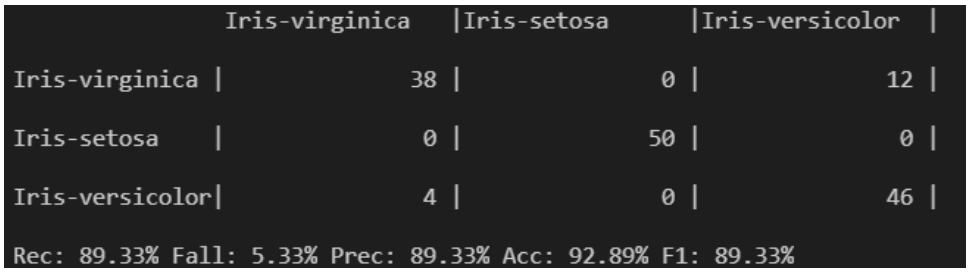
Wnioski:

Dla pomieszanego zbioru danych, proporcja zbioru testowego do zbioru trenującego nie ma aż tak dużego znaczenia. Widzimy, że dopiero przy wartości **0.9** można zaobserwować znaczny spadek jakości algorytmu. Jest to spowodowane tym, że do zbioru trenującego wpada mało lub prawie wcale osobników danej klasy. W ogólności im więcej danych trenujących, tym lepsze działanie algorytmu.

2.2 Posortowany zbiór danych testowych

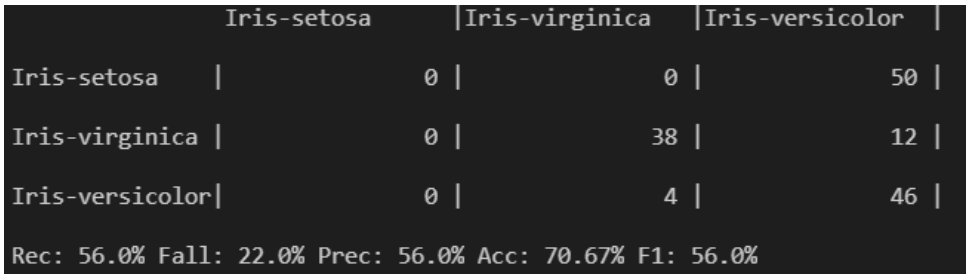
(W celu wiarygodnej analizy dla tego przypadku, zbiorem testowym będzie cały zbiór danych, w przeciwnym wypadku w większości przypadków testowana będzie tylko jedna klasa)

test_train_proportion = **0.1**



Rys 6. Macierz błędów i pomiary dla test_train_proportion = 0.1

test_train_proportion = **0.3**



Rys 7. Macierz błędów i pomiary dla test_train_proportion = 0.3

test_train_proportion = **0.5**

	Iris-virginica	Iris-versicolor	Iris-setosa
Iris-virginica	50	0	0
Iris-versicolor	42	8	0
Iris-setosa	0	50	0

Rec: 38.67% Fall: 30.67% Prec: 38.67% Acc: 59.11% F1: 38.67%

Rys 8. Macierz błędów i pomiary dla test_train_proportion = 0.5

test_train_proportion = **0.9**

	Iris-versicolor	Iris-virginica	Iris-setosa
Iris-versicolor	0	50	0
Iris-virginica	0	50	0
Iris-setosa	0	50	0

Rec: 33.33% Fall: 33.33% Prec: 33.33% Acc: 55.56% F1: 33.33%

Rys 9. Macierz błędów i pomiary dla test_train_proportion = 0.9

Wnioski

Wyniki analizy tego przypadku są dość intuicyjne. W przypadku posortowania danych ze względu na klasę, do której przynależy osobnik, w niektórych przypadkach do zbioru trenującego w ogóle nie trafiają osobniki poszczególnych klas. Algorytm nie posiada wiedzy o takich klasach przez co nie jest w stanie ich poprawnie zidentyfikować.

3. Wnioski ogólne

Algorytm naiwnej klasyfikacji Bayesa jest niesamowicie prosty w implementacji, a przy tym skuteczny. Jest to godne uwagi rozwiązanie w wielu przypadkach zważając właśnie na proporcję czasu implementacji do jego wydajności.