Politechnika Warszawska

Wydział Elektroniki i Technik Informacyjnych

Wstęp do Sztucznej Inteligencji

ćwiczenie 5 - Sztuczne sieci neuronowe

Autor:

Jarosław Jaworski, Arkadiusz Niedzielski

Data oddania ćwiczenia:

27.05.2025

SPIS TREŚCI

1 Cel ćwiczenia

Celem ćwiczenia było utworzenie naiwnego klasyfikatora Bayesa oraz zbadanie jego efektywności poprzez użycie metody k-krotnej walidacji krzyżowej i podziałów losowych.

2 Przegląd zbioru danych i implementacji

Zbiorem danych wykorzystanym w zadaniu był zbiór Spambase, należący do bazy danych *UCI Machine Learning Repository*. Zawierał on liczną ilość cech (features), bo aż 57, oraz pojedynczą kolumnę klasy. W tym wypadku obiektem badań był algorytm realizujący zadanie klasyfikacji binarnej, ponieważ klasa mogła przyjmować dwie wartości - spam/not spam (0/1). Dodatkową trudnością przy pracy z danymi był fakt ich ciągłości, w efekcie zmuszający do przyjęcia pewnego rozkładu dla określenia prawdopodobieństwa wystąpienia cech. W tym celu założono rozkład normalny (zamiast np. rozkładu Laplace'a). Dzięki temu zabiegowi ustandardyzowano dane do postaci umożliwiającej przyporządkowanie wystąpienia wartości z zakresu do wyniku dla klasy (np. wart. 0,14 dla cechy należy do <0,11;0,15> i dla niej klasa=1). Gęstośc prawdopodobieństwa obliczano na podstawie zależności (1).

$$P(x_i|Class) = \frac{1}{\sqrt{2\pi}\cdot\sigma}\cdot e^{\left(-\frac{(x_i-\mu)^2}{2\sigma^2}\right)} \tag{1}$$

,gdzie:

x_i - wartość cechy

μ - średnia wartość danej cechy w zbiorze

σ - odchylenie standardowe

Algorytm naiwnego klasyfikatora oparty został na klasycznej niezmodyfikowanej wersji zależności go realizującej, przedstawionej w zależności (2).

$$\hat{y} = \arg\max_{C} P(C) \prod_{i} P(x_i \mid C) \tag{2}$$

Prawdopodobieństwo wystąpienia klas wyjścia obliczano przy okazji dopasowania zbioru danych do klasyfikatora. Wykorzystano metodę *value_counts()* dostępną dla typu *Series* z biblioteki *pandas*, która zliczała ile razy wystąpiły obie klasy. Równocześnie obliczano średnią oraz odchylenie standardowe każdej z pozostalych cech dla wystąpienia przy nich klasy spam oraz not-spam. Całość przedstawiono w Wycinku kodu 1.

```
def fit(self, X:pd.DataFrame, y:pd.Series)
[...]
# Zliczanie ilosci wystapien klas wyjscia
class_counts = y.value_counts(normalize=True)
self.class_prob = class_counts.to_dict()

# Zliczanie sredniej i odchylenia standardowego dla
   ktorych wystepuje dana klasa wyjscia (dla kazdej z
   cech zbioru)
for class_label in class_counts.index:
    X_class = X[y == class_label]

    self.class_stats[class_label] = {
        'mean': X_class.mean(),
        'std': X_class.std(ddof=0) + 1e-9
    }
```

Wycinek kodu 1: Fragment metody fit z klasy klasyfikatora Bayesa, realizujący przygotowanie danych

Pętla działania programu (generowania przewidywań) pobierała wcześniej obliczone średnie i odchylenia standardowe dla kolejnych cech z danego wiersza i przekazywała je do obliczenia gęstości prawdopodobieństwa (1). Następnie realizowała (2).

3 Porównanie metod pomiaru efektywności algorytmu

W kolejnym etapie ćwiczenia badano różnice między metodami pomiaru efektywności algorytmu. Obiektem badań były k-krotna walidacja krzyżowa oraz losowe podziały zestawów trenującego i testującego. Niestety, strona źródłowa, z której pobrano zbiór danych, zawierała informację na temat oczekiwanych wyników dla wszystkich metod klasyfikacji za wyjątkiem klasyfikatora Bayesa, stąd porównania musiały zostać zrealizowane subiektywnie, bez punktu odniesienia.

Rozpoczęto od zaimplementowania funkcji generującej ocenę podstawowych metryk modelu - czułość, precyzja oraz dokładność. Skorzystano z gotowych rozwiązań biblioteki *scikit-learn*, również dla k-krotnej walidacji krzyżowej. Następnie przeprowadzono serię testów dla obu metod pomiaru. Wyniki przedstawiono w Tabeli 1 oraz Tabeli 2.

Tabela 1. Wyniki pomiarów metryk dla metody pomiaru losowego podziału zbiorów ternującego i testującego

Metryka	Średnia [%]	Odchylenie standardowe [%]
Dokładność	79,78	1,65
Precyzja	67,22	2,04
Czułość	95,26	0,73

Tabela 2. Wyniki pomiarów metryk dla metody pomiaru k-krotną walidacją krzyżową

Metryka	Średnia [%]	Odchylenie standardowe [%]
Dokładność	81,63	2,78
Precyzja	69,4	4,2
Czułość	95,54	2,25

4 Wnioski

Model wykazywał się znaczącą stabilnością na przestrzeni testów. Odchylenia standardowe wyników metryk mieściły się w granicach kilku punktów procentowych, wskazując na niezawodność programu klasyfikatora. Niestety, nie wiązało się to z zadowalającymi wynikami w kwestii poprawności przewidywań. Z wyników ocen metryki precyzji, wynika, że nadmiarowa ilość maili byłaby scharakteryzowana jako spam. Porównując skuteczność oceny spamu przy użyciu naiwnego

klasyfikatora Bayesa i innych modeli uczenia maszynowego (opisanych na stronie źródłowej https://archive.ics.uci.edu/dataset/94/spambase), można zauważyć, jak znacząco gorsze są wyniki precyzji modelu. Różnica między precyzją klasyfikatora Bayesa a pozostałymi modelami (za wyjątkiem SVM) jest rzędu około 20 punktów procentowych.

Wynika to najprawdopodobniej z założeń naiwnego klasyfikatora. W rzeczywistości występuje pewna korelacja między danymi. Przykładowo, w mailach spamowych często można zauważyć występowanie kilku elementów równocześnie, takich jak adresy, numery kont bankowych, itp. itd. Klasyfikator Bayesa w uproszczonej formie nie jest w stanie wziąć pod uwagę tego kontekstu. Analizuje on wyłącznie pradopodobieństwa poszczególnych elementów.

Metody pomiaru wykazały nieznaczne różnice. Pomiary przy losowych podziałach zestawów trenującego i testowego wykazały gorsze wskaźniki metryk, natomiast lepszą stabilność. K-krotna walidacja krzyżowa wygenerowała lepsze wartości ale wykazała się nieznacznie słabszą stabilnoścIa.