WOJSKOWA AKADEMIA TECHNICZNA

im. Jarosława Dąbrowskiego

WYDZIAŁ CYBERNETYKI

SPRAWOZDANIE
Metody Eksploracji Danych

Temat laboratorium: **KLASYFIKACJA NA PODSTAWIE KLASYFIKATORA BAYESOWSKIEGO I NAJBLIŻSZEGO SĄSIEDZTWA**

INFORMATYKA

(kierunek studiów)

INŻYNIERIA SYSTEMÓW – ANALIZA DANYCH

(specjalność)

Zespół:

Michał ŚLĘZAK
Szymon OLEŚKIEWICZ

Prowadzący laboratorium:

Dr inż. Romuald Hoffmann, prof. WAT

Warszawa 2025

Spis treści

Wstęp

Celem niniejszego sprawozdania jest zaprezentowanie praktycznego zastosowania naiwnego klasyfikatora Bayesa do problemu detekcji spamu na podstawie występowania słów kluczowych w wiadomościach. Przeprowadzono klasyfikację testowej wiadomości oraz dokonano krytycznej oceny przydatności metody do tego typu zadań.

Naiwny klasyfikator Bayesa należy do najczęściej stosowanych algorytmów w systemach filtracji spamu, ze względu na swoją prostotę implementacyjną, efektywność obliczeniową oraz zaskakująco wysoką skuteczność w praktycznych zastosowaniach. Metoda ta została z powodzeniem wykorzystana m.in. w systemach pocztowych oraz aplikacjach do komunikacji natychmiastowej.

Rozdział I. Podstawy teoretyczne

I.1. Naiwny klasyfikator Bayesa

Naiwny klasyfikator Bayesa jest probabilistyczną metodą klasyfikacji opartą na twierdzeniu Bayesa z założeniem niezależności warunkowej cech. Zasada działania klasyfikatora opiera się o rozwiązanie zadania optymalizacyjnego w postaci:

$$C^* = arg \max_{\{C_i:\, i \geq 1\}} \left(Pr\{C_i\} \prod_{k=1}^{n} Pr\{x_k | C_i\} \right) \tag{1}$$

, gdzie

- $Pr\{C_i\}$ – prawdopodobieństwo wystąpienia klasy i, które można estymować poprzez iloczyn liczby wystąpień danej klasy do liczność zbioru testowego,
- $Pr\{x_k | C_i\}$ – prawdopodobieństwo warunkowe wystąpienia atrybutu x_k pod warunkiem wystąpienia klasy C_i.

Dla każdego atrybutu prawdopodobieństwo $Pr\{x_k | C_i\}$ można estymować w oparciu o zbiór uczący Z, wykorzystując wzór:

$$Pr\{x_k | C_i\} = \frac{\left| C_i^{x_k} \right|}{|C_i|} \tag{2}$$

, gdzie

- $\left| C_i^{x_k} \right|$ – oznacza liczbę „przykładów” (w zbiorze Z) z klasy C_i, dla których atrybut o numerze $k = 1, \ldots, n$ przyjmuje wartość x_k. [1]

I.2. Naiwny klasyfikator Bayesa w filtracji spamu

Naiwny klasyfikator Bayesa jest jedną z najpopularniejszych metod stosowanych w automatycznej filtracji spamu. Jego popularność wynika z kilku kluczowych zalet: prostoty implementacji, niewielkich wymagań obliczeniowych, możliwości inkrementalnego uczenia oraz wysokiej skuteczności praktycznej pomimo upraszczających założeń teoretycznych.

W kontekście filtracji spamu zadanie klasyfikacji polega na przypisaniu każdej wiadomości do jednej z dwóch klas: spam lub nie-spam. Decyzja ta opiera się na analizie cech charakterystycznych wiadomości, najczęściej występowania określonych słów kluczowych.

I.2.1. Założenie o niezależności atrybutów

Kluczowym uproszczeniem w naiwnym klasyfikatorze Bayesa jest założenie o warunkowej niezależności cech przy danej klasie. Dla wiadomości opisanej przez n słów kluczowych $x_1, x_2, \ldots, x_{\mathrm{n}}$, gdzie każde $x_{\mathrm{i}} \in \{tak, nie\}$ oznacza obecność lub brak danego słowa, stąd można wykorzystać własność:

$$\Pr\{\mathrm{X}|C_i\} = \Pr\{x_1, x_2, \ldots, x_{\mathrm{n}}|C_i\} = \prod_{k=1}^{n} Pr\{x_k|C_i\} \tag{3}$$

To założenie jest określane jako "naiwne", ponieważ w praktyce słowa w tekście często są skorelowane. Przykładowo, występowanie słowa "pieniądz" może być silnie skorelowane z występowaniem słowa "darmowy" w kontekście spamu. Pomimo tego teoretycznego ograniczenia, w praktyce naiwny klasyfikator Bayesa wykazuje zaskakująco dobrą skuteczność. [1]

Rozdział II. Opis problemu

II.1. Treść zadania

Załóżmy, że chcemy zbudować algorytm odfiltrowania spamu z otrzymanych wiadomości pocztą lub za pomocą czatu. W spamie występują treści wg. określonych słów kluczowych, których statystyka występowania została zawarta w tabeli wraz z dokonaną klasyfikacją wg. pewnego algorytmu.

Tab. 1. Statystyka występowania słów kluczowych w wiadomościach

Nr wiad.	Słowa kluczowe					Klasyfikacja: spam
	pieniądz	darmowy	bogaty	nieprzyzwoicie	tajny	
1	nie	nie	tak	nie	tak	tak
2	tak	tak	tak	nie	nie	tak
3	nie	nie	nie	nie	nie	nie
4	nie	tak	nie	nie	nie	tak
5	tak	nie	nie	nie	nie	nie
6	nie	tak	nie	tak	tak	tak
7	nie	tak	nie	tak	nie	tak
8	nie	nie	nie	tak	nie	tak
9	nie	tak	nie	nie	nie	nie
10	nie	nie	nie	nie	tak	nie
11	tak	tak	tak	nie	tak	tak
12	tak	nie	nie	nie	tak	tak
13	nie	tak	tak	nie	nie	nie
14	tak	nie	tak	nie	tak	???

II.2. Opis problemu badawczego

Problem badawczy dotyczy klasyfikacji wiadomości elektronicznych jako spam lub nie-spam na podstawie występowania określonych słów kluczowych. Zadanie polega na zbudowaniu klasyfikatora, który na podstawie historycznych danych o wiadomościach i ich klasyfikacji będzie w stanie automatycznie kategoryzować nowe, nieznane wiadomości.

Zebrano dane dotyczące 13 wiadomości treningowych, dla których znana jest klasyfikacja oraz informacja o występowaniu pięciu słów kluczowych charakterystycznych dla spamu:

Słowa kluczowe:

- **pieniądz** – słowo często występujące w spamie oferującym szybki zarobek
- **darmowy** – typowe dla ofert promocyjnych i oszustw
- **bogaty** – związane z obietnicami wzbogacenia się
- **nieprzyzwoicie** – związane z treściami dla dorosłych
- **tajny** – używane w kontekście tajemniczych ofert i teorii spiskowych

Każde słowo kluczowe może występować (tak) lub nie występować (nie) w danej wiadomości. Klasyfikacja docelowa również ma charakter binarny: spam lub nie-spam.

Dla wiadomości testowej nr 14 o następujących cechach:

- pieniądz: **tak**
- darmowy: **nie**
- bogaty: **tak**
- nieprzyzwoicie: **nie**
- tajny: **tak**

należy określić, czy wiadomość jest spamem, wykorzystując naiwny klasyfikator Bayesa.

Rozdział III. Implementacja algorytmów

III.1. Struktura danych

Implementację przeprowadzono w języku Python z wykorzystaniem biblioteki pandas do zarządzania danymi. Dane zostały zorganizowane w strukturze DataFrame, co umożliwia efektywne operacje na zbiorze treningowym i testowym.

Kod. 1. Struktura danych

```
import pandas as pd

dane_uczace = pd.DataFrame({
    "pieniądz": ['nie', 'tak', 'nie', 'nie', 'tak', 'nie', 'nie',
                 'nie', 'nie', 'nie', 'tak', 'tak', 'nie'],
    "darmowy": ['nie', 'tak', 'nie', 'tak', 'nie', 'tak', 'tak',
                'nie', 'tak', 'nie', 'tak', 'nie', 'tak'],
    "bogaty": ['tak', 'tak', 'nie', 'nie', 'nie', 'nie', 'nie',
               'nie', 'nie', 'nie', 'tak', 'nie', 'tak'],
    "nieprzyzwoicie": ['nie', 'nie', 'nie', 'nie', 'nie', 'tak',
'tak',
                       'tak', 'nie', 'nie', 'nie', 'nie', 'nie'],
    "tajny": ['tak', 'nie', 'nie', 'nie', 'nie', 'tak', 'nie',
              'nie', 'nie', 'tak', 'tak', 'tak', 'nie'],
    "spam": ['tak', 'tak', 'nie', 'tak', 'nie', 'tak', 'tak',
             'tak', 'nie', 'nie', 'tak', 'tak', 'nie'],
})

dane_testowe = pd.DataFrame({
    "pieniądz": ["tak"],
    "darmowy": ["nie"],
    "bogaty": ["tak"],
    "nieprzyzwoicie": ["nie"],
    "tajny": ["tak"],
})
```

III.1.1. Zbiór treningowy

Zbiór treningowy składa się z 13 wiadomości, z których:

- 9 wiadomości sklasyfikowano jako spam (≈ 69.2%)
- 4 wiadomości sklasyfikowano jako nie-spam (≈ 30.8%)

Ta dystrybucja klas wskazuje na pewną nierównowagę w zbiorze treningowym, z przewagą przykładów spamu.

III.2. Implementacja naiwnego klasyfikatora Bayesa

Funkcja naive_bayess implementuje algorytm naiwnego klasyfikatora Bayesa zgodnie z przedstawionymi podstawami teoretycznymi. Algorytm działa następująco:

1. Wyznaczenie unikalnych klas decyzyjnych ze zbioru treningowego
2. Obliczenie prawdopodobieństw a priori $Pr\{C_i\}$ dla każdej klasy
3. Dla każdego atrybutu obliczenie prawdopodobieństw warunkowych $Pr\{x_k|C_i\}$
4. Wyznaczenie klasy maksymalizującej iloczyn $Pr\{C_i\}\prod_{k=1}^{n} Pr\{x_k|C_i\}$

Kod. 2. Funkcja implementująca algorytm naiwnego klasyfikatora Bayesa

```
1           def naive_bayess(X_train: pd.DataFrame, y_train: pd.Series, X_test:
            pd.DataFrame):
2               y_uniques = y_train.unique()
3               prob = [len(y_train[y_train == y]) / len(y_train) for y in
            y_uniques]
4
5               for col in X_train:
6                   matching = X_train[col][X_train[col] == X_test[col][0]]
7                   ys = (y_train[matching.index])
8
9                   for i in range(len(prob)):
10                      prob[i] *= len(ys[ys == y_uniques[i]]) / len(ys)
11
12              result = y_uniques[argmax(prob)]
13
14              return result
```

Rozdział IV. Wyniki eksperymentu

IV.1. Klasyfikacja wiadomości

IV.1.1. Prawdopodobieństwo *a priori*

Na podstawie zbioru treningowego wyznaczono prawdopodobieństwa a priori dla obu klas:

$$Pr\{spam = \boldsymbol{tak}\} = {}^{9}/_{13} \approx 0.6923$$
$$Pr\{spam = \boldsymbol{nie}\} = {}^{4}/_{13} \approx 0.3077$$

Te wartości wskazują, że w zbiorze treningowym większość wiadomości (około 69%) została sklasyfikowana jako spam.

IV.1.2. Prawdopodobieństwa warunkowe

Dla wiadomości testowej o cechach:

pieniądz = **tak**,
darmowy = **nie**,
bogaty = **tak**,
nieprzyzwoicie = **nie**,
tajny = **tak**,

obliczono prawdopodobieństwa warunkowe dla każdego atrybutu.

Tab. 2. Prawdopodobieństwa warunkowe dla klasy "spam = nie"

Atrybut	Wartość	$Pr\{\boldsymbol{atrybut} = \boldsymbol{wartość} \mid \boldsymbol{spam} = \boldsymbol{nie}\}$
pieniądz	tak	$4/9 = 0.444$
darmowy	nie	$2/9 = 0.222$
bogaty	tak	$3/9 = 0.333$
nieprzyzwoicie	nie	$6/9 = 0.667$
tajny	tak	$4/9 = 0.444$

Tab. 3. Prawdopodobieństwa warunkowe dla klasy "spam = tak"

Atrybut	Wartość	$Pr\{\boldsymbol{atrybut} = \boldsymbol{wartość} \mid \boldsymbol{spam} = \boldsymbol{tak}\}$
pieniądz	tak	$1/4 = 0.25$
darmowy	nie	$3/4 = 0.75$
bogaty	tak	$1/4 = 0.25$
nieprzyzwoicie	nie	$4/4 = 1.00$
tajny	tak	$1/4 = 0.25$

IV.1.3. Klasyfikacja wiadomości tekstowej

Decyzja klasyfikacji opiera się o wybraniu klasy, dającej największą wartość ilorazu prawdopodobieństw warunkowych i prawdopodobieństwa *a priori*.

Dla klasy „spam = tak”:

$$Pr\{spam = tak\} \prod_{k=1}^{n} Pr\{x_k | spam = tak\}$$
$$= 0{,}6923 \cdot 0{,}444 \cdot 0{,}222 \cdot 0{,}333 \cdot 0{,}667 \cdot 0{,}444 \approx 0.0060$$

Dla klasy „spam = nie”:

$$Pr\{spam = nie\} \prod_{k=1}^{n} Pr\{x_k | spam = nie\}$$
$$= 0{,}3077 \cdot 0{,}250 \cdot 0{,}750 \cdot 0{,}250 \cdot 1{,}000 \cdot 0{,}250 \approx 0.0036$$

Wartość łącznego prawdopodobieństwa jest większa dla klasy „spam = tak”, stąd wiadomość tekstowa zostaje zakwalifikowana jako spam.

IV.2. Dyskusja wyników

IV.2.1. Interpretacja wyniku klasyfikacji

Wiadomość testowa została sklasyfikowana jako spam z przewagą prawdopodobieństwa około 1,67:1. Analiza poszczególnych cech pozwala zrozumieć przyczyny tej decyzji:

Cechy wspierające klasyfikację jako spam:

- **pieniądz = tak** – słowo "pieniądz" występuje częściej w spamie (4/9 ≈ 44%) niż w nie-spamie (1/4 = 25%)
- **bogaty = tak** – słowo "bogaty" również częściej w spamie (3/9 ≈ 33%) niż w nie-spamie (1/4 = 25%)
- **tajny = tak** – słowo "tajny" częściej w spamie (4/9 ≈ 44%) niż w nie-spamie (1/4 = 25%)

Cechy wspierające klasyfikację jako nie-spam:

- **darmowy = nie** – brak słowa "darmowy" jest bardziej charakterystyczny dla nie-spamu (3/4 = 75%) niż dla spamu (2/9 ≈ 22%)

Cechy neutralne:

- **nieprzyzwoicie = nie** – brak tego słowa występuje zarówno w spamie (6/9 ≈ 67%), jak i w nie-spamie (4/4 = 100%)

Kluczowym czynnikiem decydującym o klasyfikacji jest również wyższe prawdopodobieństwo a priori klasy spam (≈69% vs 31%), które wzmacnia wpływ cech charakterystycznych dla spamu.

IV.2.2. Ocena skuteczności metody

Naiwny klasyfikator Bayesa w zastosowaniu do filtracji spamu charakteryzuje się następującymi właściwościami:

Mocne strony metody:

1. **Efektywność obliczeniowa** – algorytm wymaga jedynie prostych operacji arytmetycznych (mnożenia i porównywania prawdopodobieństw), co umożliwia szybką klasyfikację dużej liczby wiadomości.
2. **Łatwość interpretacji** – wynik klasyfikacji można wytłumaczyć przez analizę wkładu poszczególnych słów kluczowych, co jest istotne z perspektywy debugowania i zrozumienia decyzji systemu.
3. **Inkrementalne uczenie** – model można łatwo aktualizować w miarę napływu nowych danych treningowych bez konieczności przetwarzania całego zbioru od początku.

4. **Odporność na nieistotne cechy** – metoda automatycznie przypisuje mniejszą wagę cechom, które nie różnicują klas (jak "nieprzyzwoicie = nie" w naszym przypadku).
5. **Działanie przy małych zbiorach danych** – nawet przy 13 przykładach treningowych algorytm jest w stanie dokonać klasyfikacji, choć z ograniczoną wiarygodnością.

Słabe strony metody:

1. **Naiwne założenie niezależności** – w rzeczywistości słowa w tekstach są silnie skorelowane. Przykładowo, jeśli wiadomość zawiera słowo "pieniądz", zwiększa się prawdopodobieństwo wystąpienia również słów "darmowy" czy "bogaty". Ignorowanie tych zależności może prowadzić do błędnych oszacowań prawdopodobieństw.
2. **Problem zerowych prawdopodobieństw** – jeśli w zbiorze treningowym dane słowo nigdy nie występuje w kontekście określonej klasy, prawdopodobieństwo warunkowe wynosi 0, co powoduje, że całe prawdopodobieństwo a posteriori również wynosi 0. W praktyce stosuje się wygładzanie Laplace'a, aby temu zapobiec. [2]
3. **Wrażliwość na nierównowagę klas** – przewaga klasy spam w zbiorze treningowym (69% vs 31%) może prowadzić do nadmiernej tendencji klasyfikatora do przypisywania wiadomości do klasy większościowej.
4. **Ograniczony zbiór cech** – w naszym przypadku analizowanych jest tylko 5 słów kluczowych, podczas gdy w praktycznych systemach analizuje się tysiące cech. Ograniczony zestaw cech może nie wystarczyć do dokładnej klasyfikacji.
5. **Brak uwzględnienia kontekstu** – metoda analizuje jedynie obecność lub brak słów, ignorując ich kontekst, kolejność występowania czy częstotliwość.

Z uwagi na wymienione cechy naiwny klasyfikator Bayesa, w większości przypadków, jest dobrym sposobem na wykrywanie spamu, ponieważ jest tani w implementacji oraz łatwo się adaptuje na podstawie danych treningowych.

Podsumowanie

W ramach niniejszego laboratorium przeprowadzono praktyczną analizę zastosowania naiwnego klasyfikatora Bayesa do problemu automatycznej detekcji spamu w wiadomościach elektronicznych. Na podstawie zbioru 13 wiadomości treningowych zbudowano model klasyfikacyjny, który następnie zastosowano do klasyfikacji nowej, nieznanej wiadomości.

Kluczowe wnioski:

1. **Skuteczność metody** – naiwny klasyfikator Bayesa, pomimo upraszczającego założenia o niezależności cech, stanowi efektywną metodę detekcji spamu, co potwierdzają zarówno wyniki eksperymentu, jak i szerokie zastosowanie praktyczne w komercyjnych systemach.
2. **Wpływ prawdopodobieństw *a priori*** – rozkład klas w zbiorze treningowym (69% spam vs 31% nie-spam) istotnie wpływa na decyzje klasyfikacyjne, co podkreśla znaczenie reprezentatywności danych treningowych.
3. **Interpretacja cech** – analiza prawdopodobieństw warunkowych pozwala na zrozumienie, które słowa kluczowe są najbardziej charakterystyczne dla spamu: "pieniądz", "bogaty" i "tajny" silnie wspierały klasyfikację jako spam, podczas gdy brak słowa "darmowy" sugerował legalną wiadomość.
4. **Ograniczenia małego zbioru danych** – zbiór 13 wiadomości treningowych jest niewystarczający do zbudowania niezawodnego systemu produkcyjnego. Praktyczne systemy wymagają tysięcy przykładów treningowych i setek lub tysięcy cech.

Bibliografia

[1] Hoffmann R. Metody eksploracji danych - Wykład 5. Klasyfikacja Naive Bayes slajdy. Materiały dydaktyczne WAT, 2025.

[2] Leung, K.M., 2007. Naive bayesian classifier. *Polytechnic University Department of Computer Science/Finance and Risk Engineering*, *2007*, pp.123-156.

Spis tabel

Załączniki

1. Plik źródłowy Lab-3-Zadanie-2-Klasyfikatory.py – implementacja algorytmóu naiwnej klasyfikacji Bayesa zawierająca funkcje:
    a. `naive_bayes()` – implementacja naiwnego klasyfikatora Bayesa;
    b. `print_result()` – funkcja pomocnicza do wyświetlania wyników.
2. Plik notebook Lab-3-Zadanie-2-Obliczenia.ipynb – kompletny kod przeprowadzonych eksperymentów obejmujący:
    a. Definicję zbioru danych treningowych (13 wiadomości)
    b. Definicję danych testowych (wiadomość nr 14)
    c. Wywołanie funkcji klasyfikującej
    d. Wyświetlenie wyników klasyfikacji