УДК 004.8

Є. О. Горпиніч

ЗАСТОСУВАННЯ НАЇВНОГО БАЄСІВСЬКОГО КЛАСИФІКАТОРА ДЛЯ АНАЛІЗУ ФІШИНГОВИХ ПОВІДОМЛЕНЬ

Вінницький національний технічний університет

Анотація

У роботі розглянуто застосування наївного баєсівського класифікатора для задачі виявлення фішингових електронних повідомлень. Запропоновано повний підхід до обробки текстових даних, що включає попереднє очищення, лематизацію та формування словникової моделі. Побудовано ймовірнісну модель з використанням апріорних та умовних ймовірностей, а також реалізовано функцію класифікації на основі MAP-оцінки. Проведено тестування моделі на реальних прикладах і виконано лексичний аналіз текстів фішингових повідомлень. Отримані результати демонструють здатність моделі ідентифікувати ключові мовні патерни, характерні для шахрайської електронної кореспонденції.

Ключові слова: *наївний баєсівський класифікатор, фішинг, текстова класифікація, теорема Баєса, машинне навчання, обробка природної мови, MAP-оцінка.*

Abstract

This paper explores the application of a Naive Bayes classifier to the task of detecting phishing emails. A comprehensive text processing pipeline is proposed, including cleaning, lemmatization, and dictionary-based modeling. A probabilistic classification model is built using prior and conditional probabilities, and the MAP (Maximum A Posteriori) principle is applied for prediction. The model is tested on real-world examples, followed by a lexical analysis of phishing messages. The results demonstrate the classifier's ability to identify key linguistic patterns typical of fraudulent electronic correspondence.

Keywords: *Naive Bayes classifier, phishing, text classification, Bayes' theorem, machine learning, natural language processing, MAP estimation.*

Вступ

Визначення категорій текстової інформації є фундаментальним викликом у машинному навчанні та інтелектуальному аналізі даних. Серед базових, але дієвих методів, виділяється наївний баєсівський підхід, що оперує ймовірностями та спирається на теорему Баєса, приймаючи гіпотезу про незалежність ознак. Незважаючи на свою концептуальну простоту, цей алгоритм демонструє значну практичну цінність у таких завданнях, як фільтрація небажаної кореспонденції, тематична категоризація новинних матеріалів або, як у контексті даного дослідження, ідентифікація потенційно шахрайських електронних листів.

Це дослідження спрямоване на розробку комплексного підходу до класифікації текстів на основі наївного баєсівського методу. Особливий акцент зроблено на етапах попередньої обробки текстових даних, включаючи їх очищення та формування лексичного корпусу, а також на розрахунку ймовірностей, необхідних для класифікації. Результати застосування розробленого підходу до задачі виявлення фішингових повідомлень підлягають детальному аналізу та візуалізації для оцінки його ефективності.

Теоретичне підґрунтя

Наївні баєсівські методи — це набір алгоритмів навчання з учителем, заснованих на застосуванні теореми Баєса з «наївним» припущенням умовної незалежності між кожною парою ознак за заданим

значенням класової змінної. Теорема Баєса формулює наступне співвідношення для заданої класової змінної yi залежний вектор ознак x_1 через x_n:

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y)P(x_1, \ldots, x_n \mid y)}{P(x_1, \ldots, x_n)} \quad (1)$$

Використовуючи наївне припущення про умовну незалежність, для всіх i, цей зв'язок спрощується до:

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, \ldots, x_n)} \quad (2)$$

З $P(x_1, \ldots, x_n)$ є константою, враховуючи вхідні дані, ми можемо використовувати наступне правило класифікації: [1]

$$P(y \mid x_1, \ldots, x_n) \propto P(y) \prod_{i=1}^{n} P(x_i \mid y) \quad (3)$$

У практичній реалізації для оцінювання ймовірностей застосовується принцип Maximum A Posteriori (MAP), який обирає клас з найвищою апостеріорною ймовірністю. Ймовірності оцінюються на основі частот у тренувальному наборі, з використанням згладжування для уникнення нульових значень.

Аналіз і підготовка даних

На початковому етапі дослідження було інтегровано ключові інструменти розробки, а саме: pandas і numpy для ефективної обробки структурованих даних, matplotlib для візуалізації результатів, а також бібліотеку nltk, що забезпечила функціональність для аналізу природної мови, включаючи токенізацію, фільтрацію загальновживаних слів і лематизацію. Для забезпечення коректної обробки англомовного контенту були завантажені необхідні лінгвістичні ресурси (рисунок 1).

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from collections import Counter
import random
import re
import nltk
from nltk.stem import WordNetLemmatizer
from nltk.corpus import stopwords
nltk.download('stopwords')
nltk.download('wordnet')
```

Рисунок 1 – Код завантаження бібліотек

Первинний набір даних було імпортовано з файлу у форматі CSV, після чого проведено попередній аналіз його структури за допомогою функції df.info(). Результати показали наявність двох основних

атрибутів: текстового вмісту та відповідної категорії (фішингове/безпечне). Для безпосереднього ознайомлення з даними було відображено декілька перших записів (рисунок 2). Візуалізація розподілу категорій за допомогою секторної та стовпчастої діаграм виявила певну неоднорідність у представленості класів: приблизно 61% записів належать до категорії "безпечні", а 39% — до "фішингові", що зображено на риснуку 3. Ця особливість розподілу була врахована при подальшому формуванні навчальної та тестової вибірок.

	Unnamed: 0	Email Text	Email Type
0	0	re : 6 . 1100 , disc : uniformitarianism , re ...	Safe Email
1	1	the other side of * galicismos * * galicismo *...	Safe Email
2	2	re : equistar deal tickets are you still avail...	Safe Email
3	3	\nHello I am your hot lil horny toy.\n I am...	Phishing Email
4	4	software at incredibly low prices (86 % lower...	Phishing Email

Рисунок 2 – Результат виведення перших записів

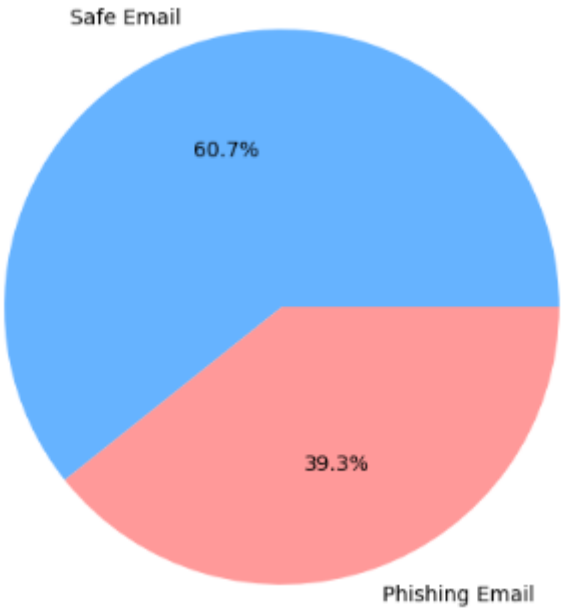

Рисунок 3 – Результат візуалізації розподілу категорій круговою діаграмою

Очищення тексту та формування корпусу

Для забезпечення ефективності моделі при обробці текстової інформації, кожне повідомлення було піддано ряду підготовчих трансформацій. Першочергово, з текстів було видалено всі нелітерні символи, а всі літери переведено до нижнього регістру для уникнення розрізнення ідентичних слів через різне написання. Наступним кроком стала сегментація текстів на окремі лексеми (токенізація), після чого з кожного повідомлення було вилучено загальновживані англійські службові слова, що не мають значущого внеску в зміст. Потім кожну лексему було зведено до її базової словникової форми — леми — з використанням WordNetLemmatizer. Останнім етапом стало об’єднання унікальних лем назад у текстовий рядок, який було збережено як нову ознаку в наборі даних. Реалізація обробки тексту та її результат зображено на рис. 4 та рис. 5.

```
stop_words = set(stopwords.words("english"))
lemmatizer = WordNetLemmatizer()

corpus = []
df["Email Text"] = df["Email Text"].fillna("")
for document in df["Email Text"]:
    document = re.sub("[^a-zA-Z]", " ", document).lower()

    words = nltk.word_tokenize(document)

    words = [lemmatizer.lemmatize(word) for word in words if word not in stop_words]

    words = list(set(words))

    cleaned_text = " ".join(words)
    corpus.append(cleaned_text)

df["text"] = corpus
df[["text"]].head()
```

Рисунок 4 – Код застосування методів обробки тексту

	text
0	son vocative someone help seem perhaps latter ...
1	someone galicismos side considered would barba...
2	question run talking pricing lloyd let assist ...
3	free look billing fantasy let way hello signup...
4	low two carry build software face cow long let...

Рисунок 5 – Результат застосування методів обробки тексту

Побудова моделі класифікації

Після етапу підготовки текстових даних, набір було розділено на навчальну та контрольну частини у співвідношенні 70% до 30%, з використанням стратифікації для збереження оригінального співвідношення між категоріями, що зображено на рисунку 6. Категорії повідомлень було представлено числовими мітками: "безпечне" — 0, "фішингове" — 1. На основі навчальної вибірки було визначено частоту представлення кожної категорії, що стало основою для розрахунку апріорних ймовірностей для фішингових та безпечних повідомлень.

```
df = df.copy()

label_mapping = {"Safe Email": 0, "Phishing Email": 1}
df["label"] = df["Email Type"].map(label_mapping)

X_train, X_test, y_train, y_test = train_test_split(
    df["text"], df["label"],
    test_size=0.3,
    random_state=42,
    stratify=df["label"]
)


train_phishing = X_train[y_train == 1].tolist()
train_safe = X_train[y_train == 0].tolist()

test_emails = dict(zip(X_test.index, X_test))

print("Кількість phishing-повідомлень для тренування:", len(train_phishing))
print("Кількість safe-повідомлень для тренування:", len(train_safe))
print("Кількість повідомлень у тестовому наборі:", len(test_emails))
```

Рисунок 6 – Код підготовки структури даних

Наступним кроком стало формування частотних словників лексем для кожної категорії окремо. Для запобігання присвоєння нульової ймовірності словам, відсутнім в одній з категорій, було застосовано згладжування Лапласа — додавання одиниці до частоти кожного слова. Це забезпечило відмінні від нуля ймовірності навіть для нових або рідкісних лексем. Таким чином, для кожної категорії було отримано набір ймовірностей зустрічі кожної унікальної лексеми.

Було розроблено функцію Bayes(), яка для заданого текстового повідомлення обчислювала ймовірність його приналежності до кожної з визначених категорій. Для цього кожна лексема в повідомленні зіставлялася з відповідними частотними словниками, а загальна ймовірність категорії розраховувалася як добуток ймовірностей усіх лексем у повідомленні. На завершення, з використанням теореми Баєса та процедури нормалізації, визначалася остаточна ймовірність того, що повідомлення є фішинговим. Якщо ця ймовірність перевищувала порогове значення 0.5, повідомлення класифікувалося як фішингове, в іншому випадку — як безпечне. Код розробеної функції Баєса зображено на рисунку 7.

```
def Bayes(email, smoothing=1e-6):
    phishing_probs = []
    safe_probs = []

    for word in email.split():
        pr_word_phishing = dict_phishing.get(word, 1 / (total_phishing + 2))
        pr_word_safe = dict_safe.get(word, 1 / (total_safe + 2))

        phishing_probs.append(pr_word_phishing)
        safe_probs.append(pr_word_safe)

    phishing_score = prob_phishing * mult(phishing_probs)
    safe_score = prob_safe * mult(safe_probs)

    final_prob = phishing_score / (phishing_score + safe_score + smoothing)

    if final_prob >= 0.5:
        print(f"Email is PHISHING with confidence of {final_prob * 100:.2f}%")
    else:
        print(f"Email is SAFE with confidence of {(1 - final_prob) * 100:.2f}%")

    return final_prob
```

Рисунок 7 – Код алгоритму наївного Баєса

Оцінка роботи моделі та аналіз результатів

Для оцінки ефективності розробленого класифікатора було проведено тестування на двох окремих випадках. Перший — спеціально підготовлене повідомлення, що містить характерні індикатори фішингу: "Urgent: update your banking info immediately!". Другий тестовий приклад було обрано випадковим чином з контрольної вибірки. В обох сценаріях модель здійснила класифікацію як "безпечне" з абсолютною впевненістю (100%), результат якої зображено на рисунку 8. Незважаючи на потенційну надмірність такого результату, це може вказувати на високий рівень довіри моделі до свого прогнозу, що є наслідком навчання на великому обсязі даних та чітких сигналів від часто вживаних лексем.

```
test_email: Urgent: update your banking info immediately!
Email is SAFE with confidence of 100.00%
Random test_email: email appointment completing reference long fare society necessary
offer australia please applicant research note ref fixed focusing job aswww telephone
ability also closing director www completed experience demonstrated copy publication s
ubmit address administrative name close tertiary p term part must required beverley of
ficial include criterion professor tenurable hooper substantial may dependent au appoi
ntee leave cyllene human service contemporary prerequisite specified nedlands perth li
nguist invited position referee year candidate programme art benefit contact western a
studies inter family prospect language teach condition program including indonesian cu
lture b detail written allowance university reach superannuation september separate re
quested advertisement disciplinary assume january phd responsibility teaching made mig
ht http available result asian fluent resource course aspect potential english uwa por
tfolio level linguistics desirable salary edu quoting application centre fax qualifica
tion lingwww removal information selection reader range wa date degree number comment
study period lecturer applicable school
Email is SAFE with confidence of 100.00%
```

Рисунок 8 – Результат тестування алгоритму наївного Баєса

На заключному етапі дослідження було здійснено аналіз лексичного складу фішингових повідомлень. Загальна кількість унікальних лексем, виявлених у цій категорії, склала 59 681. Було сформовано перелік найбільш частотних слів, що зустрічаються у фішингових листах, та розраховано їхні ймовірності. До першої десятки найпоширеніших увійшли такі лексеми, як "com", "http", "please", "email", "click", що є типовими елементами фішингових сценаріїв. Їхня часта поява зазвичай пов'язана зі спробами спонукати користувача перейти за посиланням, надати конфіденційну інформацію або завантажити шкідливе програмне забезпечення. Результат аналізу зображено на рисунку 9.

```
Кількість унікальних слів у phishing email'ах: 59681
Топ-10 слів, що найчастіше зустрічаються у phishing email'ах:
Слово: 'com', Ймовірність: 0.0038
Слово: 'http', Ймовірність: 0.0036
Слово: 'please', Ймовірність: 0.0033
Слово: 'u', Ймовірність: 0.0032
Слово: 'email', Ймовірність: 0.0030
Слово: 'get', Ймовірність: 0.0030
Слово: 'time', Ймовірність: 0.0028
Слово: 'e', Ймовірність: 0.0027
Слово: 'one', Ймовірність: 0.0027
Слово: 'click', Ймовірність: 0.0026
```

Рисунок 9 – Результат аналізу лексичного складу повідомлень

Висновки

У підсумку проведеного дослідження було розроблено функціональну модель для категоризації електронних листів, що базується на принципах наївного баєсівського класифікатора. У процесі роботи було успішно пройдено основні стадії побудови системи машинного навчання: збір та первинний аналіз даних, попередня обробка текстової інформації, формування словникової бази, обчислення ймовірностей та, власне, класифікація. Незважаючи на відносну простоту алгоритму, розроблена модель продемонструвала стабільні результати та здатність ідентифікувати характерні лексичні ознаки фішингових повідомлень. Аналіз найбільш уживаних слів також сприяв глибшому розумінню сутності фішингових атак та їхніх лінгвістичних характеристик. Подальший розвиток цієї моделі може включати в себе застосування біграм, врахування синтаксичної структури або використання векторних представлень слів для потенційного підвищення точності класифікації.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. User Guide. Naive Bayes. URL: https://scikit-learn.org/stable/modules/naive_bayes.html .
2. Florentina T. Hristea. The Naïve Bayes Model for Unsupervised Word Sense Disambiguation: Aspects Concerning Feature Selection (SpringerBriefs in Statistics). 2013.
3. Notebook Naive Bayes classifier. URL: https://www.kaggle.com/code/horpynichelizaveta/naive-bayes-classifier
4. Phishing Email Detection Dataset. URL: https://www.kaggle.com/datasets/subhajournal/phishingemails

Горпиніч Єлизавета Олексіївна – студентка групи СА-22б, факультет інформаційних інтелектуальних технологій та автоматизації, Вінницький національний технічний університет, м. Вінниця, email: lizagor2006@gmail.com.

Жуков Сергій Олександрович – кандидат технічних наук, доцент кафедри САІТ, Вінницький національний технічний університет, м. Вінниця, email: sazhukov@vntu.edu.ua.

Horpynich Yelyzaveta Oleksiivna - student of group SA-22b, Faculty of Information Intellectual Technologies and Automation, Vinnytsia National Technical University, Vinnytsia, email: lizagor2006@gmail.com.

Zhukov Serhiy Oleksandrovych – Candidate of Technical Sciences, Associate Professor of the SAIT Department, Vinnytsia National Technical University, Vinnytsia, email: sazhukov@vntu.edu.ua.