Парсер є корисним інструментом для науковців, які працюють із великими об’ємами даних. Завдяки парсингу вебсторінок вони можуть отримувати необхідну інформацію з різних джерел, аналізувати її та використовувати для своїх наукових досліджень.

Основна перевага використання парсера полягає в тому, що він може значно зменшити витрати часу та зусиль на пошук, обробку й аналіз даних. До того ж використання парсера допомагає отримати більш точну та повну інформацію, ніж під час ручного збору даних.

Із допомогою парсера науковці можуть зібрати та проаналізувати великий об’єм даних, що дасть змогу зробити більш точні висновки й отримати нові знання в різних галузях науки. До того ж використання парсера допомагає автоматизувати процес збору даних, що зменшує ймовірність помилок і забезпечує швидкий та ефективний доступ до необхідної інформації.

Отже, користування парсером може допомогти науковцям ефективніше й точніше збирати та аналізувати дані, що дає змогу робити більш досконалі дослідження та отримувати нові знання в різних галузях науки.

**Список використаних джерел**

1. Mitchell R. Web Scraping with Python: Collecting More Data from the Modern Web 2nd Edition, 2018. 306 p.

2. Heydt M. Python Web Scraping Cookbook: Over 90 proven recipes to get you scraping with Python, micro services, Docker and AWS, 2018. 364 p.

3. Beautiful Soup Documentation. URL: https://www.crummy.com/software/BeautifulSoup/bs4/doc/

**УДК 004.85:004.93(043.2)**

*Бевз Д. М., здобувач вищої освіти 2 курсу ОС «Магістр» спеціальності 122 Комп’ютерні науки,*

*Римар П. В., старший викладач кафедри інформаційних технологій*

## ЕФЕКТИВНІСТЬ АЛГОРИТМІВ МАШИННОГО НАВЧАННЯ НА ПРИКЛАДІ ЗАДАЧІ КЛАСИФІКАЦІЇ ТЕКСТІВ

*Донецький національний університет імені Василя Стуса, м. Вінниця*

У сучасному світі обсяги текстових даних зростають із неймовірною швидкістю, що створює потребу в ефективних методах їх обробки та аналізу. Особливо актуальним є завдання класифікації текстів, яке має широке застосування в різних сферах, від виявлення спаму до аналізу настроїв.

Мета дослідження полягає у вивченні та порівнянні ефективності різних алгоритмів машинного навчання для задачі класифікації текстів. Для цього були обрані три алгоритми: наївний класифікатор Баєса, логістична регресія та дерево ухвалення рішень [1, 2].

Методологія дослідження містить збір і підготовку датасету, реалізацію вибраних алгоритмів, їх навчання та тестування. Особлива увага була приділена аналізу точності, швидкості й ефективності кожного з алгоритмів.

## Види та класифікації текстів

Перед початком обробки текстів необхідно спочатку класифікувати їх за різними критеріями. Одним із таких критеріїв є тип тексту або його призначення. Розрізняють такі види текстів:

- інформативні тексти: новини, статті, наукові дослідження тощо;
- літературні тексти: романи, вірші, оповідання;
- технічні тексти: технічні описи, інструкції, технічні звіти;
- рекламні тексти: оголошення, рекламні брошури, рекламні статті;
- соціальні медіатексти: повідомлення в соціальних мережах, коментарі, пости.

Крім типу тексту, текстові дані можна класифікувати за тематикою, мовою, жанром тощо. Це допомагає виконувати більш специфічні завдання обробки та аналізу.

Для тестування алгоритму потрібні дані, на яких можна його навчити. Такі дані було взято з сайта kaggle [3]. Задача класифікації текстів вирішувалась на прикладі датасету із текстами зі статтями BBC [4]. Датасет містить 2 225 текстів, розділених на 5 категорій (рис. 1).

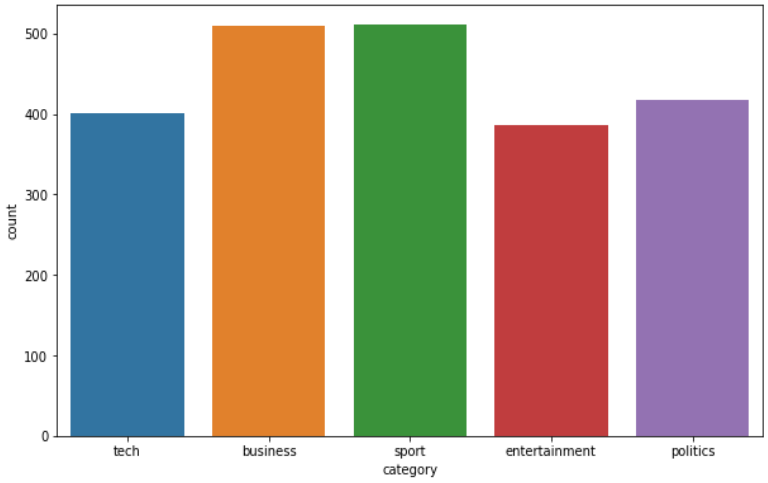


*Рисунок 1. Категорії поділу тексту в датасеті*

Після запуску алгоритму в Colab Notebooks потрібно завантажити датасет та подивитись інформацію про нього. Наступним кроком треба отримати гістограму, в якій показано розподіл текстів за групами. Далі потрібно створити

прогностичну модель із допомогою Word2Vec у вигляді векторного простору. Після цього запустити функцію класифікації зі створеним вектором, яка розділить текст на дві групи: для навчання і для тестування. З цього результату перший вектор відповідатиме за кількість текстів із датасету, що виділені для навчання, другий за кількість текстів для перевірки. Отриманий результат відображено на рис. 2.

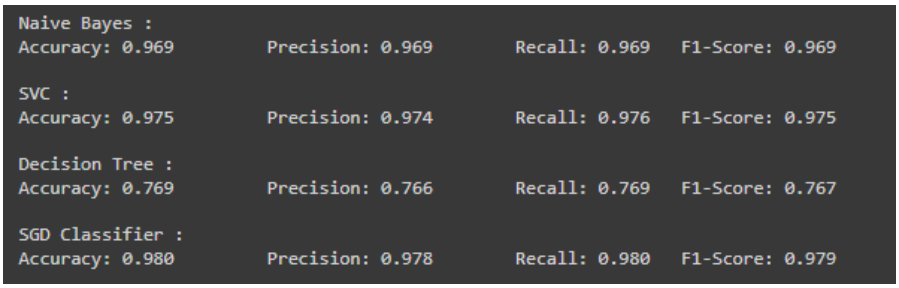


*Рисунок 2. Результат навчання та тестування*

Після цього можна аналізувати отримані результати роботи алгоритму на основі датасету. Після запуску алгоритму отримано такий результат:

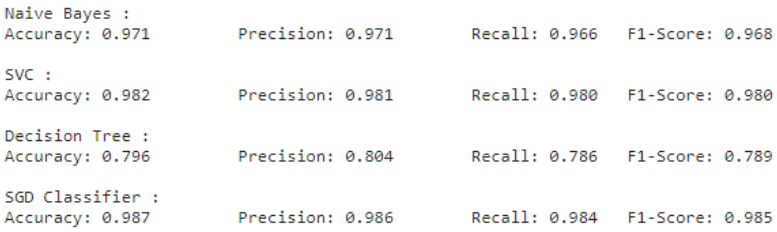


*Рисунок 3. Результат роботи алгоритму*

У цьому результаті є дані про кожен виконаний алгоритм: Accuracy, Precision, Recall та F1-Score.

На підставі проведених досліджень можна дійти висновку, що для цього датасету доцільно використовувати алгоритм логістичної регресії. Отже, користувачі системи можуть самі завантажувати датасети та оцінювати моделі машинного навчання для обрання найбільш ефективної моделі.

Результати дослідження показали, що кожен з алгоритмів має свої переваги й недоліки залежно від специфіки задачі та характеристик датасету. Наївний класифікатор Байєса виявився ефективним у випадках із великою кількістю класів, логістична регресія показала високу точність у бінарній класифікації, а дерево ухвалення рішень було корисним для інтерпретації результатів.

Дослідження підкреслюють важливість вибору відповідного алгоритму машинного навчання для конкретної задачі класифікації текстів. Результати

цього дослідження можуть бути корисними для фахівців у сфері обробки природної мови та машинного навчання.

**Список використаних джерел**

1. Palanivinayagam A., El-Bayeh C. Z., Damaševičius R. A Review of Twenty Years of Machine-Learning-Based Text Classification: A Systematic Review. *Algorithms 2023.* № 16(5). 236 p.

2. Naive Bayes Classifier Explained: Applications and Practice Problems of Naive Bayes Classifier: вебсайт. URL: https://www.analyticsvidhya.com

3. Kaggle: вебсайт. URL: https://www.kaggle.com

4. BBC articles fulltext and category: вебсайт. URL: https://www.kaggle.com

**УДК 004.4’416:004.738.1(043.2)**

*Луцков М. П., здобувач вищої освіти 2 курсу ОС «Магістр» спеціальності 122 Комп’ютерні науки,*

*Римар П. В., старший викладач кафедри інформаційних технологій*

## МЕТОДИ ТЕХНІЧНОЇ ОПТИМІЗАЦІЇ САЙТІВ

*Донецький національний університет імені Василя Стуса, м. Вінниця*

У сучасному інтернет-середовищі важко переоцінити роль вебсайтів, які є ключовим елементом взаємодії між користувачами та інформацією. Зростання конкуренції в електронному просторі підкреслює важливість належної функціональності та продуктивності вебресурсів. Оптимізація сайта виконує важливу роль у покращенні його продуктивності, швидкості завантаження та загальної ефективності.

Методи технічної оптимізації сайта є необхідним інструментарієм для забезпечення ефективної роботи вебресурсів в умовах усе вищих технологічних вимог та різноманітних платформ користувачів. Вони охоплюють широкий спектр заходів, спрямованих на поліпшення архітектури вебсайтів, їх швидкості реакції та адаптивності до різних пристроїв [1].

Ця робота спрямована на вивчення та аналіз методів технічної оптимізації сайтів з метою розкриття їх впливу на користувачів та бізнес-середовище. Вона також розглядає сучасні тенденції в розробці вебресурсів та використання новітніх технологій для досягнення максимального рівня оптимізації.

Вивчення цієї теми дає змогу розглядати важливі аспекти взаємодії між технічними показниками сайта та його користувачами. Отже, вивчення методів