Зображення, що містить текст

Автоматично згенерований опис

Міністерство освіти і науки України

Національний технічний університет України

“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

**Лабораторна робота №4**

Аналіз текстів з використанням мови Python

**Тема:** Класифікація текстових даних

**Варіант:** 1

Виконав Перевірив:

студент групи ІП-11: Тимофєєва Ю. С

Панченко С. В.

Київ 2023

Зміст

[1 Мета лабораторної роботи 6](#__RefHeading___Toc272_1253693292)

[2 Завдання 7](#__RefHeading___Toc274_1253693292)

[3 Виконання 8](#__RefHeading___Toc276_1253693292)

[3.1 Створення моделі TD-IDF. 8](#__RefHeading___Toc278_1253693292)

[3.2 KNN 12](#__RefHeading___Toc280_1253693292)

[3.3 Logistic Regression 14](#__RefHeading___Toc282_1253693292)

[3.4 Random Forest 16](#__RefHeading___Toc284_1253693292)

[3.5 SVM 17](#__RefHeading___Toc286_1253693292)

[3.6 Naive Bayes 17](#__RefHeading___Toc288_1253693292)

[3.7 Порівняння результатів 19](#__RefHeading___Toc290_1253693292)

[4 Висновок 21](#__RefHeading___Toc292_1253693292)

# Мета лабораторної роботи

Ознайомитись з класифікацією документів за допомогою моделей машинного навчання.

# Завдання

Створити програму, яка зчитує заданий набір даних, виконує попередню обробку та класифікацію документів відповідно до варіанту. Якщо недостатньо ресурсів для роботи з повним набором даних, можна виділити частину, але таким чином, щоб були присутні усі класи.

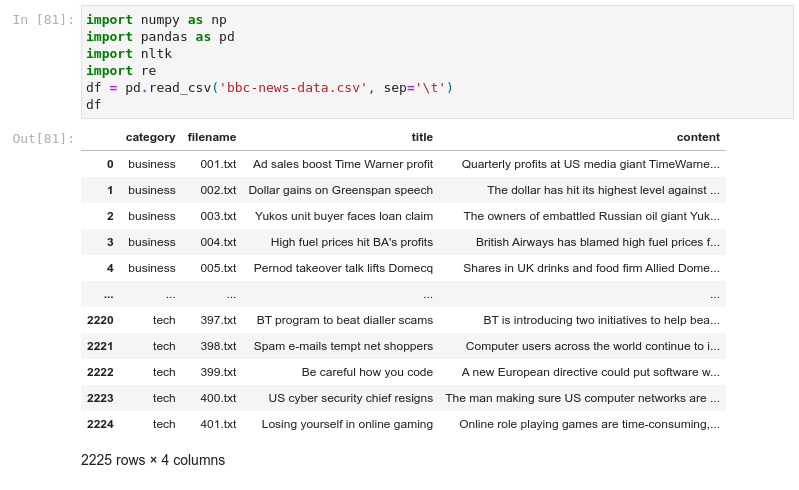
Варіант 1.

Файл bbc-news-data.csv. В якості текстової моделі використати TD-IDF. Виконати класифікацію за допомогою алгоритмів наївний байєсів класифікатор та опорні вектори, порівняти їх точність. Спробувати покращити модель наївний байєсів класифікатор за допомогою GridSearchCV.

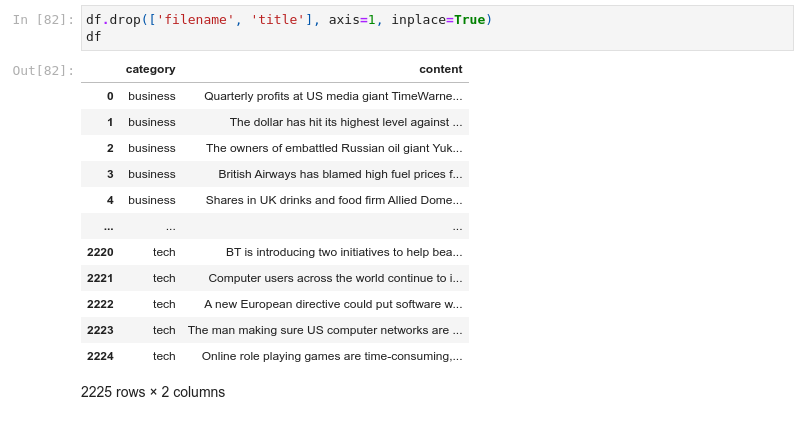
# Виконання

## Створення моделі TD-IDF.

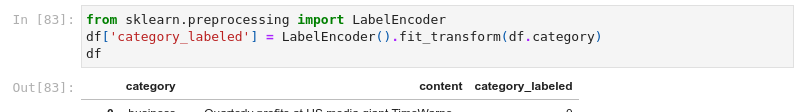
Для початку імпортуємо модулі та зчитаємо файл.

  
  
Рисунок 3.1 - Зчитування файлу

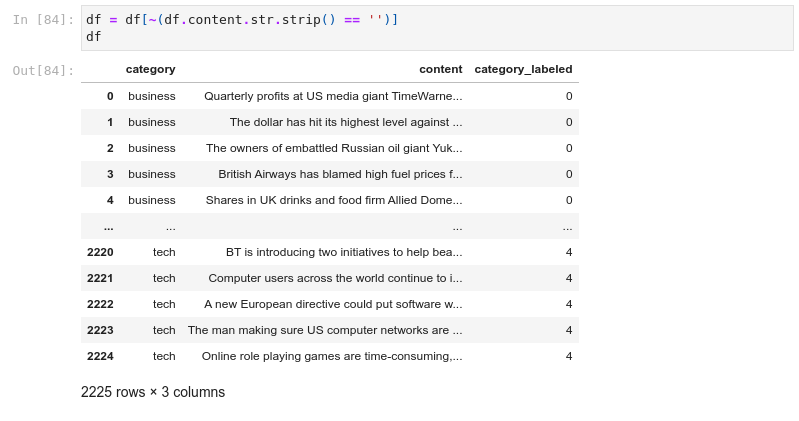
Видалимо колонки 'filename' та 'title', щоб класифікувати модуль лише за категоріями.

  
  
Рисунок 3.2 - Видалення колонок 'filename' та 'title'

Перетворимо категоріальні змінні у числові за допомогою класу LabelEncoder з модуля sklearn.preprocessing.

  
  
Рисунок 3.3 - Перетворення категоріальних змінних в числові

Видалимо порожні документи, якщо вони є.

  
  
Рисунок 3.4 - Видалення порожніх документів

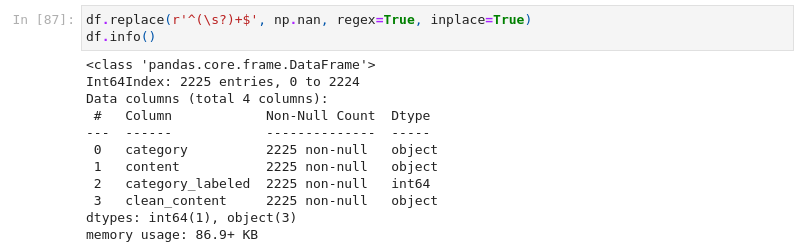
Визначимо стоп-слова англійської мови.

  
  
Рисунок 3.5 - Стоп-слова

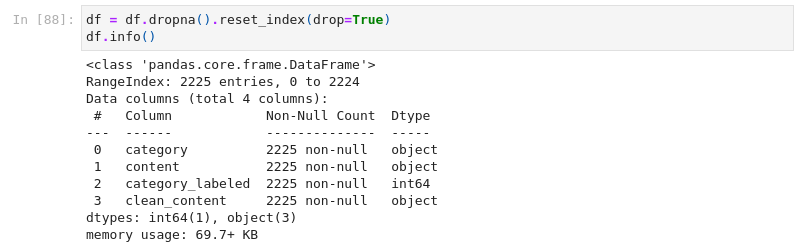
Визначимо функцію, що виконує попередню обробку документу. Застосуємо декоратор np.vectorize для того, щоб функція могла працювати з корпусами.

  
  
Рисунок 3.6 - Обробка документів

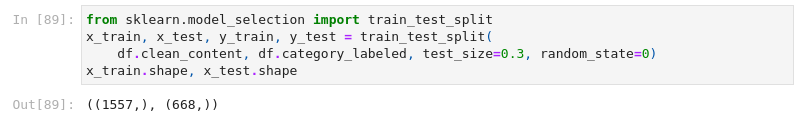
У результаті попередньої обробки деякі документи могли стати порожніми. Тому потрібно це перевірити.

  
  
Рисунок 3.7 - Перевірка на присутність пустих значень

Видалимо пусті значення.

  
  
Рисунок 3.8 - Видалення пустих значень

Розділимо дані на навчальні та тестові, отримаємо чотири масиви: документи для навчання, документи для тестування, мітки для навчання, мітки для тестування.

  
  
Рисунок 3.9 - Розділення даних на навчальні та тестові

Використаємо TF-IDF модель для представлення тексту у векторному вигляді.

  
  
Рисунок 3.10 - TF-IDF модель

## KNN

Оберемо декілька методів класифікації: KNN, Logistic Regression, SVC, Random Forest, Naive Bayes, Gridient Boosting.

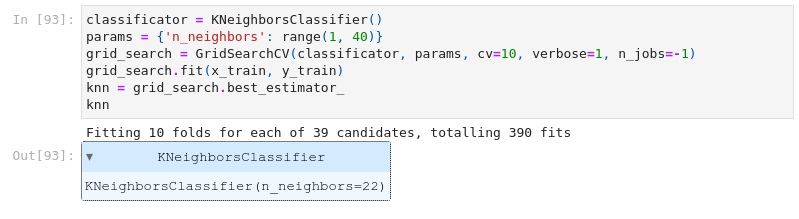
Зберігатимемо результати тестування моделей у списку results.

  
  
Рисунок 3.11 - Список результатів

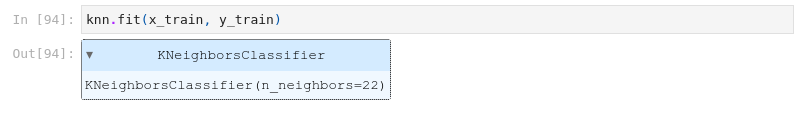
Для виконання роботи методу KNN імпортуємо sklearn.neighbors.KNeighborsClassifier та sklearn.model\_selection.GridSearchCV.

  
  
Рисунок 3.12 - Імпортування модулів

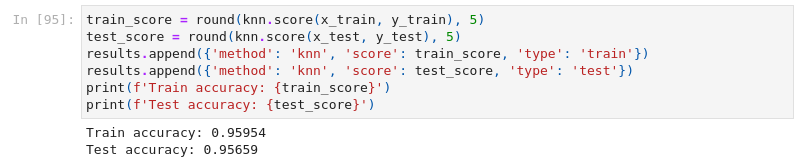
Визначимо, які варіанти параметрів найкраще вирішують дану задачу.

  
  
Рисунок 3.13 - Визначення найкращого параметра

Натренуємо модель з найкращим параметром.

  
  
Рисунок 3.14 - Тренування моделі K-Nearest Neighbors

Визначимо точність моделі на тренувальних та тестових даних.

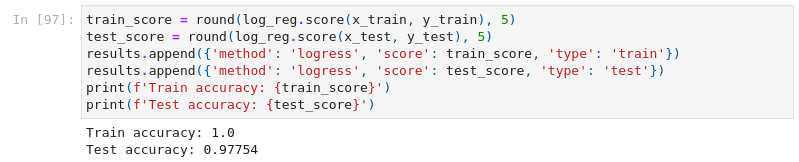
  
  
Рисунок 3.15 - Точність моделі K-Nearest Neighbors

## Logistic Regression

Для виконання роботи методу Logistic Regression імпортуємо sklearn.linear\_model.LogisticRegression. Визначимо найкращі параметри моделі, передавши в неї параметри регуляризації.

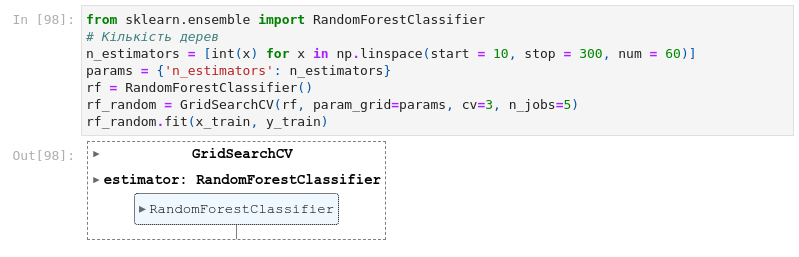
  
  
Рисунок 3.16 - Тренування моделі Logistic Regression

Визначимо точність моделі на тренувальних та тестових даних.

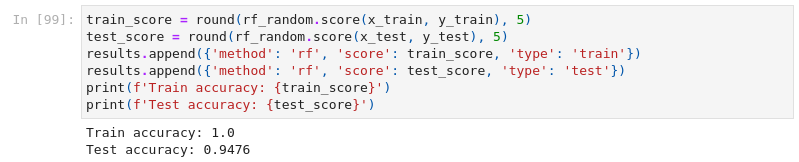
  
  
Рисунок 3.17 - Точність моделі Logistic Regression

## Random Forest

Для виконання роботи методу Random Forest імпортуємо sklearn.ensemble.RandomForestClassifier. Визначимо найкращі параметри для моделі. У випадку Random Forest параметри включають кількість дерев рішень та кількість характеристик, які враховуються кожним деревом під час поділу вузла і використовуються для поділу кожного вузла, отриманого під час навчання. Імпортуємо sklearn.model\_selection.RandomizedSearchCV.

  
  
Рисунок 3.18 - Тренування моделі Random Forest

Визначимо точність моделі на тренувальних та тестових даних.

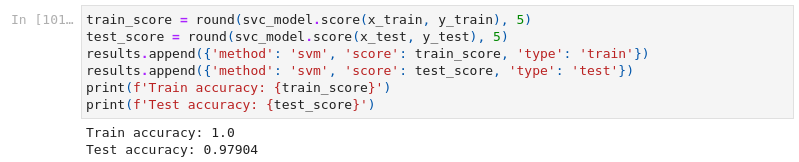
  
  
Рисунок 3.19 - Точність моделі Random Forest

## SVM

Для виконання роботи методу SVM імпортуємо sklearn.svm.SVC.

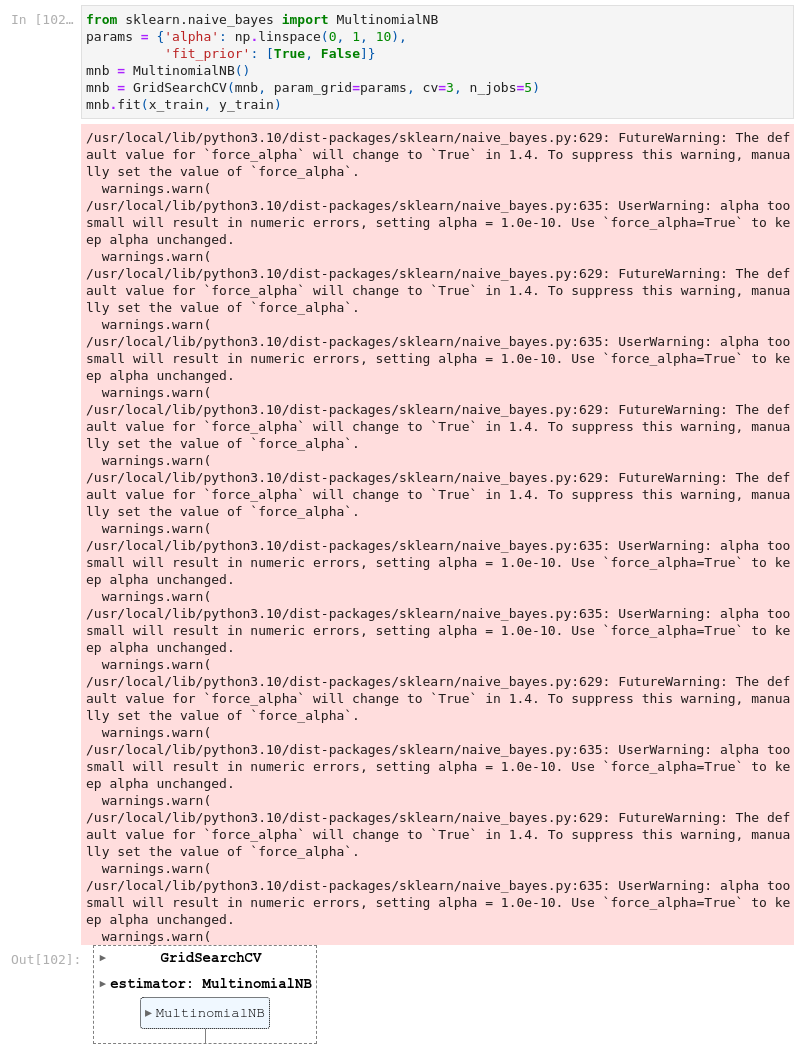
  
  
Рисунок 3.20 - Тренування моделі SVM

Визначимо точність моделі на тренувальних та тестових даних.

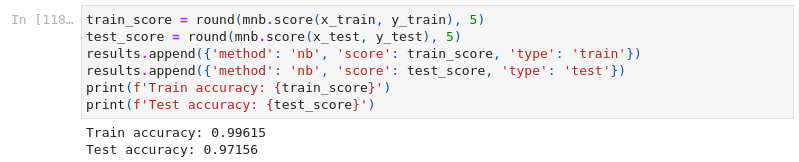
  
  
Рисунок 3.21 - Точність моделі SVM

## Naive Bayes

Для використання алгоритму імпортуємо клас MultinomialNB з модуля sklearn.naive\_bayes.

  
  
Рисунок 3.22 - Тренування моделі Naive Bayes

Визначимо точність моделі на тренувальних та тестових даних.

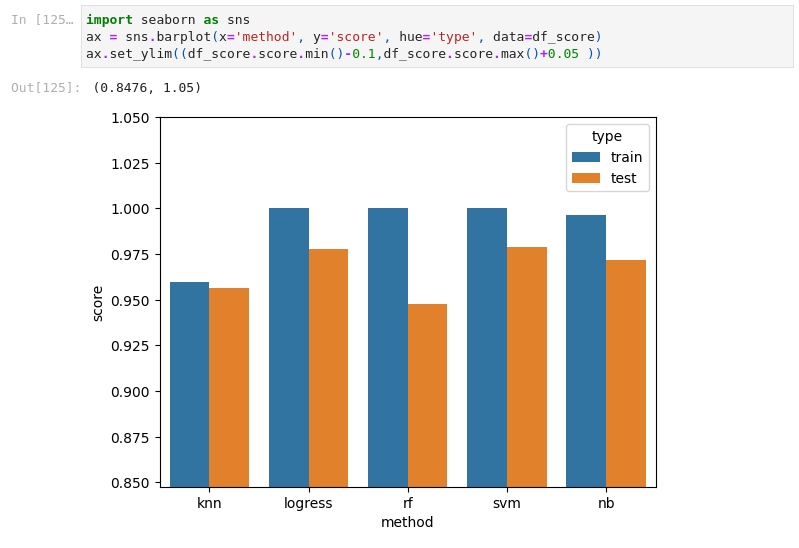
  
  
Рисунок 3.23 - Точність моделі Naive Bayes

## Порівняння результатів

Проаналізувавши окремо кожен із методів, проведемо порівняння даних методів.

  
  
Рисунок 3.24 - Датафрейм результатів

Для наочності побудуємо гістограму.

  
  
Рисунок 3.25 - Результати моделей

# Висновок

Під час виконання даної лабораторної роботи я ознайомитись з класифікацією документів за допомогою моделей машинного навчання.

Для початку я обробив набір даних: перетворив категоріальні змінні до числових, нормалізував тексти документів, використав TD-IDF модель для представлення тексту у векторному вигляді.

Після цього використав більше алгоритмів ніж було прописано у завданні, як-от: KNN, Logistic Regression, Random Forest, SVM, Naive Bayes.

Під час тренування моделей використовував GridSearch для підбору найкращих гіперпараметрів.

У результаті отримав, що на як на тестових, так і тренувальних даних SVM показала себе найкраще, однак до неї близько підібралися Logistic Regression та Naive Bayes. KNN гірше відпрацював на обох наборах ніж перші три, а Random Forest — найгірше на тестовому.