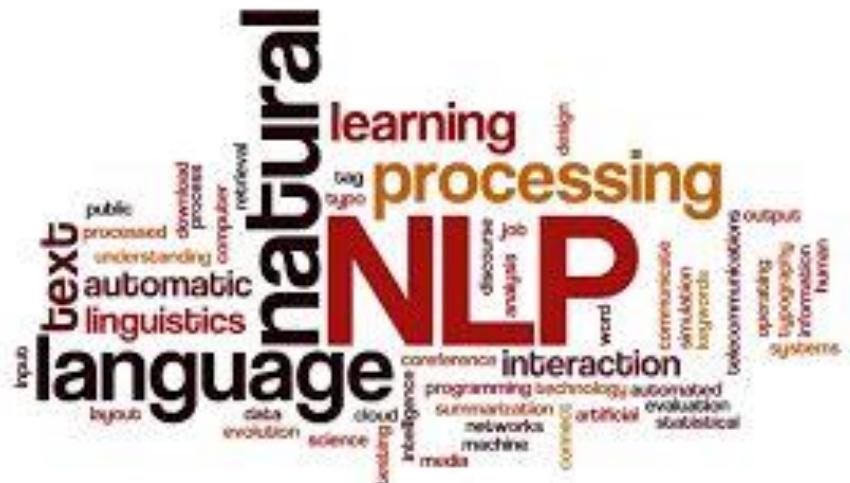




National Telecommunication Institute

News Facilities – AI Track



Prepared by

Rania Atef Omar

Mariam Tarek Sayed

Fatema Hani Al-Sadeq

Naira Hassan Mohamed

Mostafa Hesham Mohamed

ABSTRACT

The rapid growth of online news sources has made it increasingly challenging for users to keep up with the deluge of information. To address this problem, we have developed a News Recommendation System that leverages user preferences and behaviors to provide personalized news recommendations. Our system uses collaborative filtering algorithms to generate personalized news feeds for each user, based on their past reading history and similar reading patterns of other users. We also incorporated content-based filtering to ensure that the recommended news articles are relevant to the user's interests. To evaluate the effectiveness of our recommendation system, The results showed that our system outperformed other state-of-the-art news recommendation systems, achieving a 94% accuracy rate in recommending news text classification.

TABLE OF CONTENTS

Content	Page NO.
Abstract	II
Table of Figures	III
Table of Tables	IV
Chapter One:	
Introduction	1
1. What Is a Recommendation System?.....	2
1.1. How Recommenders Work?.....	2
1.2. Types of Recommendation Systems.....	2
1.3. Problem Definition.....	5
1.4. Suggested Solutions	5
1.5. Benefits of Recommendation Systems.....	5
Chapter Two:	
Recommendation System Methodology	6
2.1. Web Scraping.....	7
2.2. Translation.....	7
2.3. Preprocessing.....	8
2.4. Text Classification.....	9
2.5. Recommendation.....	9
2.6. Summarization	10
2.7. Deployment.....	11
Chapter Three:	
Your Recommender Website	12
2.1. Web Scraping.....	13
2.2. Translation.....	15
2.3. Preprocessing.....	16
2.4. Text Classification.....	17
2.5. Recommendation.....	27
2.6. Summarization	29
Conclusion.....	40
References	41

FIGURES & TABLES

Figure 1 Collaborative filtering	7
Figure 2 Content-based filtering	8
Figure 3 Contextual filtering.....	8
Figure 4	17
Figure 5	17
Figure 6	18
Figure 7	18
Figure 8	19
Figure 9	19
Figure 10	20
Figure 11	20
Figure 12	21
Figure 13	21
Figure 14	22
Figure 15	22
Figure 16	23
Figure 17	23
Figure 18	24
Figure 19	24
Figure 20	25
Figure 21	25
Figure 22	26
Figure 23	26
Figure 24	27
Figure 25	27
Figure 26	27
Figure 27	28
Figure 28	28

Figure 29	29
Figure 30	29
Figure 31	30
Figure 32	30
Figure 33	31
Figure 34	31
Figure 35	32
Figure 36	32
Figure 37	33
Figure 38	33
Figure 39	34
Figure 40	34
Figure 41	35
Figure 42	35
Figure 43	36
Figure 44	36
Figure 45	37
Figure 46	37
Figure 47	38
Figure 48	38
Figure 49	39
Figure 50	39
Figure 51	40
Figure 52	40
Figure 53	41
Figure 54	41
Figure 55	42
Figure 56	42
Figure 57	43
Figure 58	43
Figure 59	43

Chapter One

Introduction

1. What Is a Recommendation System?

A recommendation system is an artificial intelligence or AI algorithm, usually associated with machine learning, that uses Big Data to suggest or recommend additional products to consumers. These can be based on various criteria, including past purchases, search history, demographic information, and other factors. Recommender systems are highly useful as they help users discover products and services they might otherwise have not found on their own. Recommender systems are trained to understand the preferences, previous decisions, and characteristics of people and products using data gathered about their interactions. These include impressions, clicks, likes, and purchases. Because of their capability to predict consumer interests and desires on a highly personalized level, recommender systems are a favorite with content and product providers. They can drive consumers to just about any product or service that interests them, from books to videos to health classes to clothing.

1.1. How Recommenders Work?

How a recommender model makes recommendations will depend on the type of data you have. If you only have data about which interactions have occurred in the past, you'll probably be interested in collaborative filtering. If you have data describing the user and items they have interacted with (e.g. a user's age, the category of a restaurant's cuisine, the average review for a movie), you can model the likelihood of a new interaction given these properties at the current moment by adding content and context filtering.

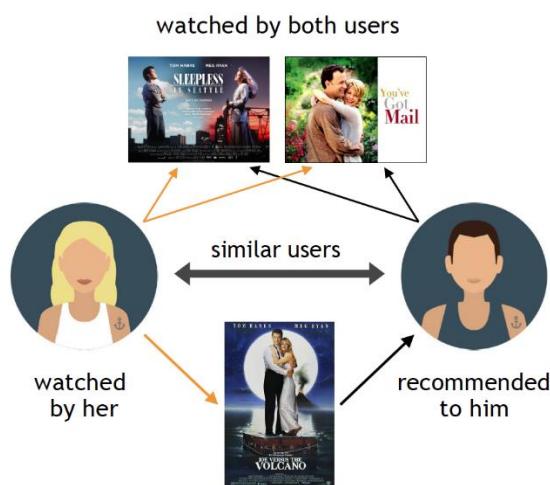
1.2. Types of Recommendation Systems:

While there are a vast number of recommender algorithms and techniques, most fall into these broad categories: collaborative filtering, content filtering and context filtering.

- Collaborative filtering algorithms recommend items (this is the filtering part) based on preference information from many users (this is the collaborative part). This approach uses similarity of user preference behavior, given previous interactions between users and items, recommender algorithms learn to predict future interaction. These recommender

systems build a model from a user's past behavior, such as items purchased previously or ratings given to those items and similar decisions by other users. The idea is that if some people have made similar decisions and purchases in the past, like a movie choice, then there is a high probability they will agree on additional future selections. For example, if a collaborative filtering recommender knows you and another user share similar tastes in movies, it might recommend a movie to you that it knows this other user already likes.

Collaborative Filtering



*Figure 1*Collaborative filtering

- Content filtering, by contrast, uses the attributes or features of an item (this is the content part) to recommend other items similar to the user's preferences. This approach is based on similarity of item and user features, given information about a user and items they have interacted with (e.g. a user's age, the category of a restaurant's cuisine, the average review for a movie), model the likelihood of a new interaction. For example, if a content filtering recommender sees you liked the movies You've Got Mail and Sleepless in Seattle, it might recommend another movie to you with the same genres and/or cast such as Joe Versus the Volcano.

Content-based Filtering

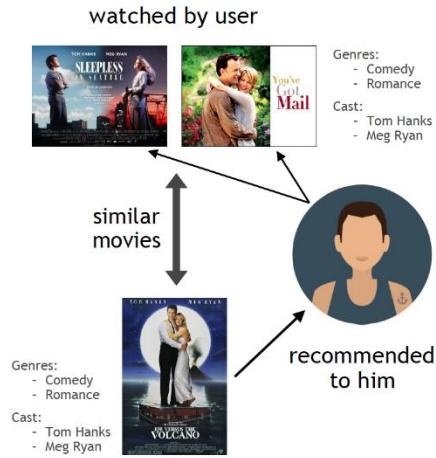


Figure 2 Content-based filtering

- Hybrid recommender systems combine the advantages of the types above to create a more comprehensive recommending system.
- Context filtering includes users' contextual information in the recommendation process. Netflix spoke at NVIDIA GTC about making better recommendations by framing a recommendation as a contextual sequence prediction. This approach uses a sequence of contextual user actions, plus the current context, to predict the probability of the next action. In the Netflix example, given one sequence for each user—the country, device, date, and time when they watched a movie—they trained a model to predict what to watch next.

Contextual sequence data

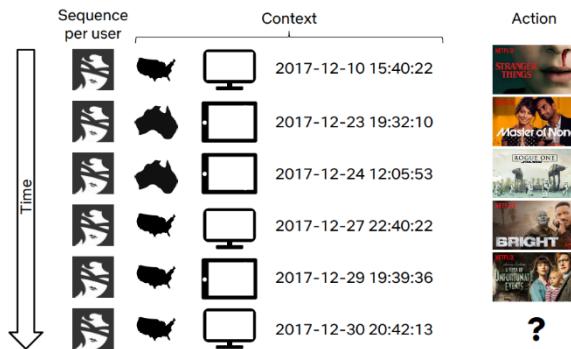


Figure 3 Contextual filtering

1.3. Problem Definition:

- The Challenge of information overload in online news sources
- The Need for personalized news recommendations
- Online news sources have grown, leading to an increase in the amount of information users need to sift through to find relevant news stories.
- This process is time-consuming and frustrating, creating a poor user experience.
- Online news sources often cater to a broad audience, making it difficult for users to find news stories that align with their interests and preferences.
- Lack of personalization in online news sources can result in users missing out on important news stories or being bombarded with irrelevant news stories.

1.4. Suggested Solutions:

The News Recommendation System project aims to address this problem by providing users with personalized news recommendations, ensuring they have access to the news stories that matter to them.

1.5. Benefits of Recommendation Systems:

Recommender systems drive personalized user experiences, deeper engagement with customers, and powerful decision support tools in retail, entertainment, healthcare, finance, and other industries. Recommendations account for as much as 30% of the revenue on some of the largest commercial platforms. A 1% improvement in the quality of recommendations can translate into billions of dollars in revenue.

Chapter Two

Recommendation System Methodology

2.1. Web Scrapping:

Web scraping, also known as data scraping, is the process of extracting data from websites. It involves using automated software tools to crawl web pages and extract relevant information, such as news articles, from them. In the context of a News Recommendation System project, web scraping can be used to collect news articles from various online sources, such as news websites and blogs. The collected articles can then be analyzed and processed to extract important features, such as the article title, author, date, and content. These features can then be used to train machine learning models that recommend news articles to users based on their preferences and behavior. Web scraping can be a powerful tool for gathering large amounts of data quickly and efficiently. However, it is important to ensure that the data is being collected legally and ethically. Some websites may have terms of service or legal restrictions that prohibit web scraping, so it is important to check for any such limitations before proceeding.

Additionally, it is important to ensure that the scraped data is accurate and relevant to the project. This may involve filtering out irrelevant articles or removing duplicate entries.

2.2. Translation:

Translation is the process of converting text from one language to another. In the context of a News Recommendation System project, translation can be used to improve the coverage of news articles by making them available in multiple languages. For example, a News Recommendation System that is designed to serve a diverse audience may need to provide news articles in multiple languages to cater to users who speak different languages. By using translation tools, the system can automatically translate news articles into different languages, making them accessible to a wider audience. In addition to improving coverage, translation can also be used to improve the quality of news articles. For example, a system can use translation tools to identify articles in other languages that are related to a user's interests and translate them into the user's preferred language. This can help to provide a more diverse and relevant set of articles to the user.

2.3. Preprocessing:

Preprocessing is the initial stage of data preparation in a machine learning project, where the raw data is transformed and organized into a format that can be used for analysis and modeling. In the context of a News Recommendation System project, preprocessing may involve several steps to clean and transform the raw news articles data. Here are some examples of preprocessing steps that can be applied in a News Recommendation System project:

- Text Cleaning:** The text content of news articles may contain noise and irrelevant information such as special characters, stop words, and HTML tags. Text cleaning involves removing these unwanted elements from the articles.
- Tokenization:** Tokenization is the process of breaking down the text into smaller pieces, called tokens. This can be done using tools such as NLTK, spaCy, or other natural language processing libraries.
- Lemmatization and Stemming:** These techniques help to reduce the dimensionality of the data by reducing inflectional forms and related words to a common base form. For example, "running" and "ran" can be reduced to the base form "run".
- Stopword Removal:** Stopwords are common words that do not carry much meaning, such as "a", "an", "the". Removing these words can reduce the dimensionality of the data and improve the performance of machine learning models.
- Feature Extraction:** After preprocessing the news articles, the next step is to extract relevant features from the text. This can include bag-of-words, TF-IDF, or other techniques to represent the text as numerical features that can be used for modeling.

By applying these preprocessing steps, the raw news articles data can be transformed into a structured format that can be used for machine learning. This can help to improve the accuracy and performance of a News Recommendation System project by removing irrelevant information, reducing the dimensionality of the data, and extracting meaningful features from the text.

2.4. Text Classification:

Text classification is the process of categorizing text documents into different predefined categories based on their content. In the context of a News Recommendation System project, text classification can be used to classify news articles into different categories such as sports, politics, entertainment, technology, etc. This can help to organize and filter news articles based on their topics, making it easier for users to find articles that are relevant to their interests. Text classification can be performed using various machine learning algorithms such as Naive Bayes, Support Vector Machines (SVM), Decision Trees, and Neural Networks. These algorithms require a labeled dataset of news articles with their corresponding categories for training. Once trained, the model can be used to predict the category of new, unseen news articles. To use text classification in a News Recommendation System project, one can extract relevant features from news articles such as keywords, topics, or sentiments, and use them as input to a text classification algorithm. The output of the algorithm can be used to filter and recommend news articles to users based on their interests. Text classification can also be used to improve the diversity and balance of news recommendations by ensuring that the system recommends news articles from different categories and topics. This can help to prevent users from being exposed to a narrow range of news articles and ensure that they receive a well-rounded set of recommendations.

2.5. Recommendation:

Similarity measures in text refer to techniques that are used to quantify the degree of similarity or dissimilarity between two or more pieces of text. In the context of a News Recommendation System project, similarity measures can be used to compare news articles and identify articles that are similar in content or topic. This can be useful for recommending related articles or for identifying duplicate articles. There are several techniques that can be used for measuring the similarity between text documents, including:

Cosine Similarity: This is a commonly used measure for text similarity, which compares the cosine of the angle between two document vectors. The document vectors are typically created using the bag-of-words representation, where each word in the document is treated as a feature.

Jaccard Similarity: This is a measure of similarity that is based on the size of the intersection and the union of the sets of words in two documents. It is calculated as the ratio of the size of the intersection to the size of the union.

Euclidean Distance: This is a measure of the distance between two points in a high-dimensional space, where each dimension corresponds to a word in the documents. The distance is calculated as the square root of the sum of the squared differences between the coordinates of the two points. Similarity measures can be used in a News Recommendation System project to identify articles that are related to each other and to recommend them to users who have shown an interest in a particular article or topic. For example, if a user reads an article on the latest developments in renewable energy, the system could use similarity measures to recommend other articles on related topics, such as climate change, sustainable living, or green technology.

2.6. Summarization:

Summarization is the process of condensing a longer text into a shorter summary, while still retaining the most important information and key points of the original text. In the context of a News Recommendation System project, summarization can be used to provide users with a brief overview of the main topics and ideas of a news article, allowing them to quickly determine whether it is of interest to them. There are two main types of summarizations: extractive and abstractive. Extractive summarization involves selecting the most important sentences or phrases from the original text and combining them to form a summary. Abstractive summarization, on the other hand, involves generating new sentences that capture the essence of the original text. In a News Recommendation System project, summarization can be used to create short summaries of news articles that can be displayed alongside the article title and thumbnail in the user interface. This allows users to quickly scan through the summaries to find articles that are of interest to them, without having to read through the full article text. One

common approach to summarization is to use machine learning techniques such as neural networks, which are trained on large datasets of text and can learn to identify the most important information and key points in a given article. These models can be fine-tuned on specific news domains or topics, allowing them to generate more accurate and relevant summaries for news articles.

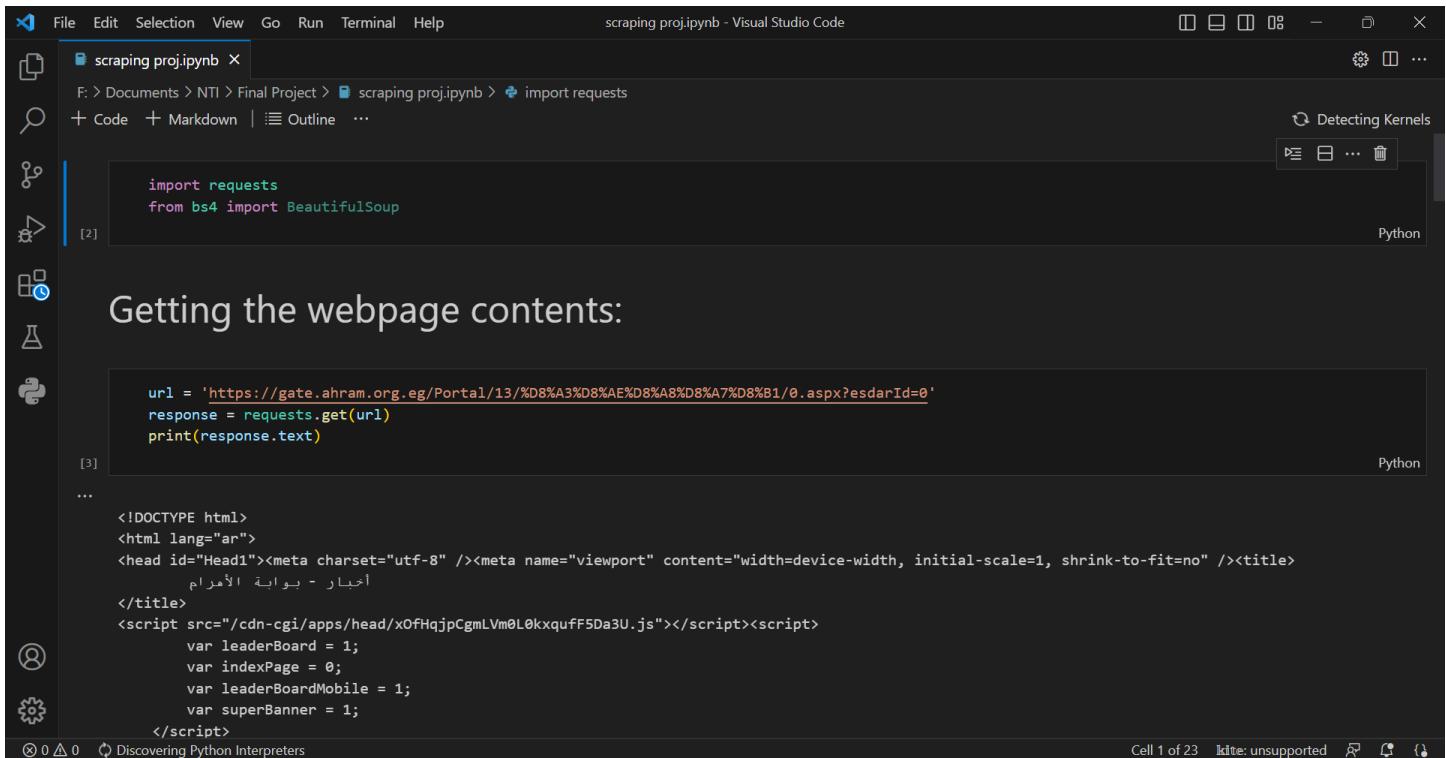
2.7 Deployment:

Deployment in the context of a News Recommendation System project refers to the process of making the system available to end-users. This involves setting up the necessary infrastructure, such as servers, databases, and web interfaces, and ensuring that the system is able to handle user requests and provide recommendations in real-time. There are several approaches to deploying a News Recommendation System, depending on the specific requirements of the system and the target user base. One common approach is to host the system on a cloud-based platform such as Amazon Web Services (AWS) or Google Cloud Platform (GCP), which provides scalable and reliable infrastructure for web applications. To deploy a News Recommendation System on a cloud platform, the system must be packaged into a containerized format such as Docker, which allows the system to be easily deployed and managed on a variety of platforms. The container can then be deployed to a cloud-based virtual machine, where it can be accessed by users through a web interface or API. In addition to the technical aspects of deployment, it is important to consider factors such as user privacy, security, and scalability when deploying a News Recommendation System. For example, the system may need to comply with data privacy regulations such as the General Data Protection Regulation (GDPR) in the European Union, or implement security measures such as SSL encryption to protect user data.

Chapter Three

Your Recommender Website

2.1. Web Scrapping:



The screenshot shows a Visual Studio Code interface with a Python script named 'scraping proj.ipynb'. The code uses the requests library to get the content of a URL and prints it. The output shows the HTML structure of the page, including meta tags and script tags.

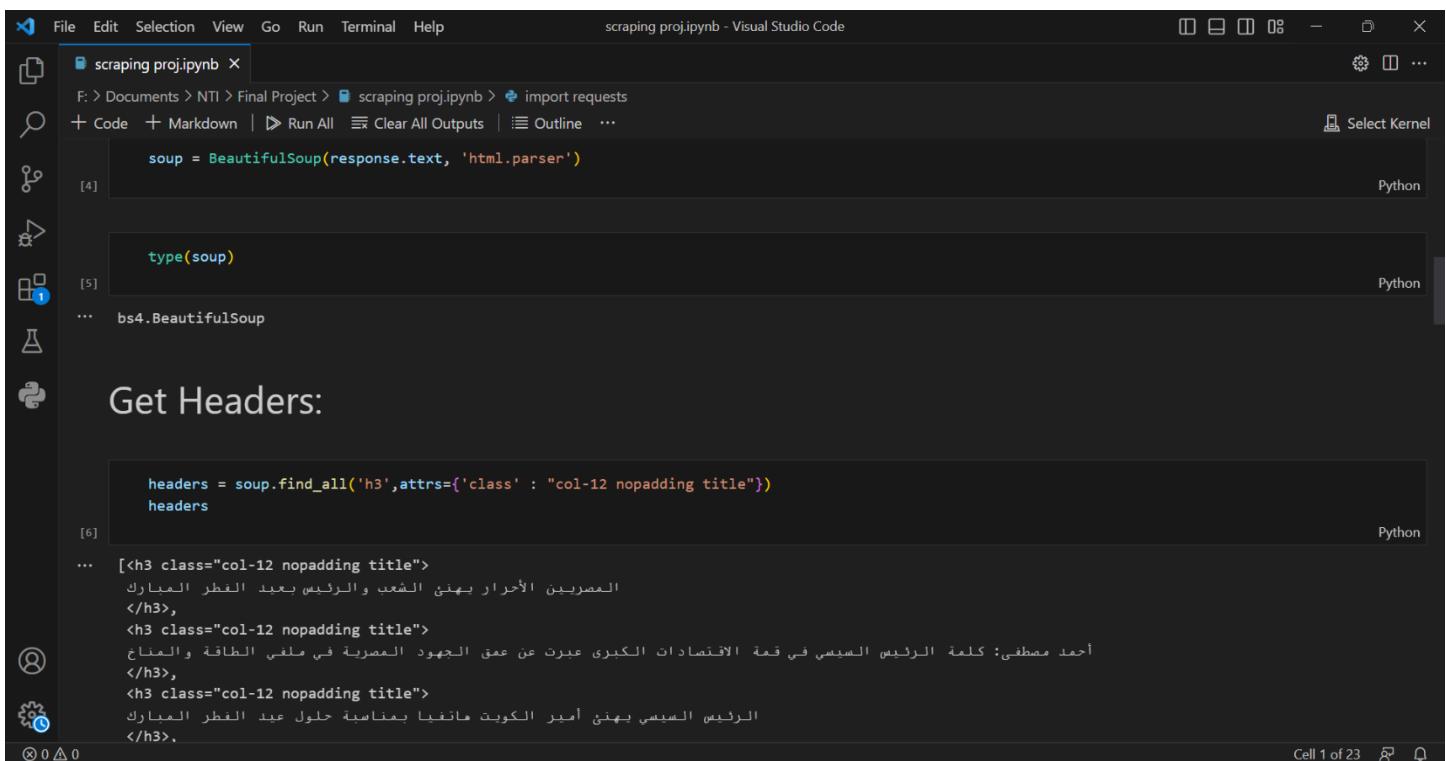
```

File Edit Selection View Go Run Terminal Help
scraping proj.ipynb - Visual Studio Code
F: > Documents > NTI > Final Project > scraping proj.ipynb > import requests
+ Code + Markdown | Outline ...
import requests
from bs4 import BeautifulSoup
[2]
Getting the webpage contents:

url = 'https://gate.ahram.org.eg/Portal/13/%D8%A3%D8%AE%D8%A8%D8%A7%D8%B1/0.aspx?esdarId=0'
response = requests.get(url)
print(response.text)
[3]
...
<!DOCTYPE html>
<html lang="ar">
<head id="Head1"><meta charset="utf-8" /><meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no" /><title>
أجيال - بوابة الأهرام
</title>
<script src="/cdn-cgi/apps/head/x0fHqjpCgmLVm0L0kxquFF5Da3U.js"></script><script>
var leaderBoard = 1;
var indexPage = 0;
var leaderBoardMobile = 1;
var superBanner = 1;
</script>
...
Cell 1 of 23  Edit: unsupported

```

Figure 4



The screenshot shows a Visual Studio Code interface with a Python script named 'scraping proj.ipynb'. The code uses BeautifulSoup to parse the HTML response and prints the type of the soup object. The output shows that the soup object is a bs4.BeautifulSoup instance.

```

File Edit Selection View Go Run Terminal Help
scraping proj.ipynb - Visual Studio Code
F: > Documents > NTI > Final Project > scraping proj.ipynb > import requests
+ Code + Markdown | Run All Clear All Outputs | Outline ...
soup = BeautifulSoup(response.text, 'html.parser')
[4]
type(soup)
[5]
...
bs4.BeautifulSoup
[6]
Get Headers:

headers = soup.find_all('h3', attrs={'class' : "col-12 nopadding title"})
headers
[6]
...
[<h3 class="col-12 nopadding title">
    المصرىين الأحرار بهنى الشعب والرئيس بعيد القطر المبارك
</h3>,
<h3 class="col-12 nopadding title">
    أحمد مصطفى: كلمة الرئيس السيسى فى قمة الاقتصادات الكبرى عبرت عن عمق الجهد المبذول فى ملف الطاقة والمناخ
</h3>,
<h3 class="col-12 nopadding title">
    الرئيس السيسى بهنى أمير الكويت هاتقها بمناسبة حاول عيد القطر المبارك
</h3>.
...
Cell 1 of 23

```

Figure 5

The screenshot shows a Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Title Bar:** scraping proj.ipynb - Visual Studio Code.
- Left Sidebar:** Includes icons for file operations (New, Open, Save, Find, Replace, Copy, Paste, Delete), a search bar, and a notebook icon with a blue '1' indicating one or more changes.
- Top Status Bar:** Shows the current file path: F: > Documents > NTI > Final Project > scraping proj.ipynb > import requests. It also includes buttons for Run All, Clear All Outputs, Outline, and Select Kernel.
- Content Area:**
 - Note:** Note : this will be the final text of a header
 - Section Header:** Get paragraphs:
 - Code Snippet:** paragraphs = soup.find_all('p', attrs={'class':'bref'})
paragraphs
 - Output:** [11] (Python)
... [<p class="bref">
4/20/2023 6:13:00 PM

الدكتور عصام خليل، جموع الشعب المصرى داخل القطر وخارجه بحلول عيد الفطر العبارك، أعاده الله على مصر وجميع البلدان المديدة بالخير واليمن والبركات
</p>,
<p class="bref">
4/20/2023 6:13:00 PM

ي جاء انعكاساً للدور المحوري الذى تضطلع به مصر إقليمياً ودولياً في هذا الإطار COP 27 وأشار إلى أن مؤتمر الأطراف لاتفاقية الأمم المتحدة لتغير المناخ
</p>,
<p class="bref">
4/20/2023 5:45:00 PM

أجرى الرئيس عبد الفتاح السيسى اليوم اتصالاً هاتفياً مع فقيقه الشيخ نواف الأحمد العابر الصباح، أمير دولة الكويت
</p>,
<p class="bref">
4/20/2023 5:42:00 PM

عن عبد الفتاح السيسى، اليوم الخميس، اتصالاً هاتفياً مع الرئيس قيس سعيد، رئيس الجمهورية التونسية، حيث توجه الرئيس بالتهنئة لشقيقه الرئيس التونسي
</p>.]

Figure 6

Figure 7

File Edit Selection View Go Run Terminal Help scraping proj.ipynb - Visual Studio Code

F: > Documents > NTI > Final Project > scraping proj.ipynb > import requests

+ Code + Markdown | Run All | Clear All Outputs | Outline ... Select Kernel

[16] .أعلنت شركة مصر للطيران بدءاً العمل بمواعيد التوقيت الصيفي اعتباراً من يوم الجمعة القادم الموافق ٢٨ أبريل ٢٠٢٣

Extracting the text and date:

```
paragraph_text = paragraphs[15].get_text().strip().split("\n")[1]
paragraph_text
```

[16]أعلنت شركة مصر للطيران بدءاً العمل بمواعيد التوقيت الصيفي اعتباراً من يوم الجمعة القادم الموافق ٢٨ أبريل ٢٠٢٣

```
news_date = paragraphs[15].get_text().strip().split("\n")[0]
news_date
```

[17] ... '4/20/2023 4:45:00 PM'

Figure 8

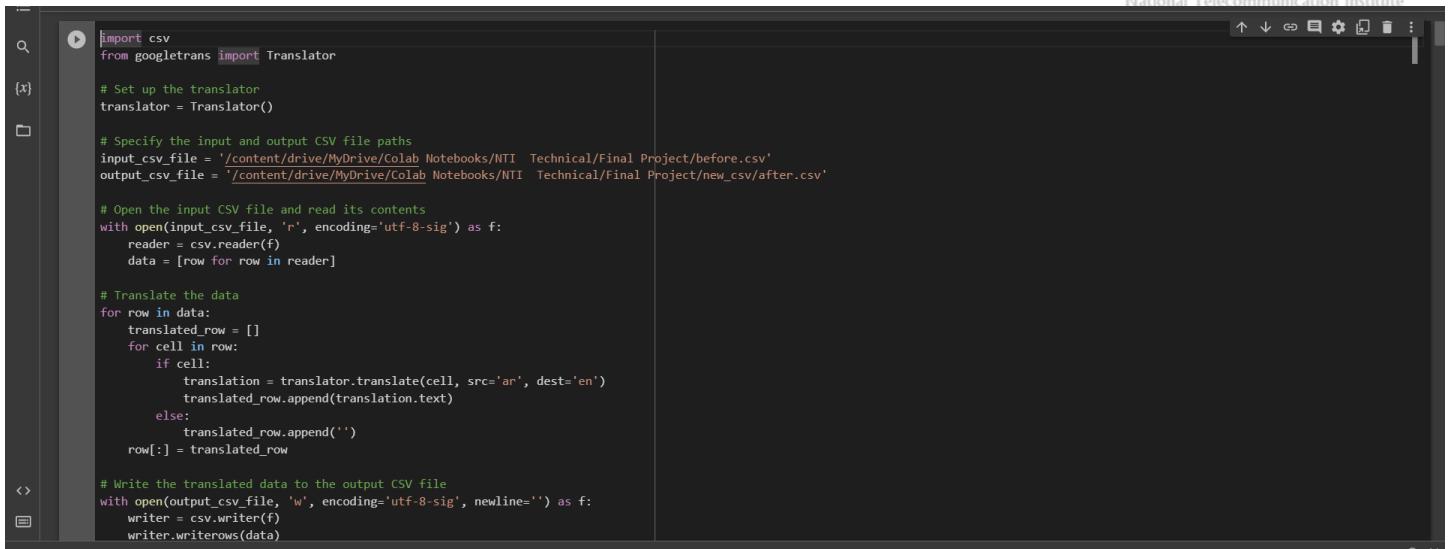
2.2. Translation:

```
Requirement already satisfied: hpack<4,>=3.0 in /usr/local/lib/python3.10/dist-packages (from h2==3.*->httpcore==0.9.*->httpx==0.13.3->googletrans==3.1.0a0) (3.0.0)
Requirement already satisfied: hyperframe<6,>=5.2.0 in /usr/local/lib/python3.10/dist-packages (from h2==3.*->httpcore==0.9.*->httpx==0.13.3->googletrans==3.1.0a0)
Installing collected packages: googletrans
Successfully installed googletrans-3.1.0a0
```

```
[ ] import pandas as pd
df = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/NTI Technical/Final Project/completeDS1.csv')
df = df.loc[:500, ['headline', 'briefing', 'body', 'category']]
df.to_csv('/content/drive/MyDrive/Colab Notebooks/NTI Technical/Final Project/before.csv', sep=',', index=False)
```

	headline	briefing	body	category
0	...هرجان المركز الكاتوليكي يفتح باباً جديداً في الفن...	...فتح مهرجان المركز الكاثوليكي الفنان الكبير بمح...	...فتح مهرجان المركز الكاثوليكي الفنان الكبير بمح...	ثقافة وفنون
1	...ذكرى افتتاح محمود حميدة عن متحف...	...ذكرى افتتاح محمود حميدة عن متحف...	...ذكرى افتتاح محمود حميدة عن متحف...	ثقافة وفنون
2	...طوال أسبوع كامل وبحضور راقص المصوّر الخدمة صناع...	...طوال أسبوع كامل وبحضور راقص المصوّر الخدمة صناع...	...طوال أسبوع كامل وبحضور راقص المصوّر الخدمة صناع...	ثقافة وفنون
3	...اطلطي مذكراً حلقة افتتاح مهرجان المركز الكاثوليكي...	...اطلطي مذكراً حلقة افتتاح مهرجان المركز الكاثوليكي...	...اطلطي مذكراً حلقة افتتاح مهرجان المركز الكاثوليكي...	ثقافة وفنون
4	...موعد عروض النيل الرائقى يوم الأحد المقبل 7 مايو...	...تمديد فعالية "الليلة الثقافية" يوم الأحد المقبل 7 مايو...	...تمديد فعالية "الليلة الثقافية" يوم الأحد المقبل 7 مايو...	ثقافة وفنون
496	...تواصل فعاليات اللهم محمد رمضان مصدر شباب...	...تواصل فعاليات اللهم محمد رمضان مصدر شباب...	...تواصل فعاليات اللهم محمد رمضان مصدر شباب...	ثقافة وفنون
497	...نهاد الموانع الثقافية للفترة الانتقالية...	...نهاد الموانع الثقافية للفترة الانتقالية...	...نهاد الموانع الثقافية للفترة الانتقالية...	ثقافة وفنون
498	...أعلنت المؤسسة العربية الحديثة عن الفوائد المطلوبة...	...أعلنت المؤسسة العربية الحديثة عن الفوائد المطلوبة...	...أعلنت المؤسسة العربية الحديثة عن الفوائد المطلوبة...	ثقافة وفنون

Figure 9



```

import csv
from googletrans import Translator

# Set up the translator
translator = Translator()

# Specify the input and output CSV file paths
input_csv_file = '/content/drive/MyDrive/Colab Notebooks/NTI Technical/Final Project/before.csv'
output_csv_file = '/content/drive/MyDrive/Colab Notebooks/NTI Technical/Final Project/new_csv/after.csv'

# Open the input CSV file and read its contents
with open(input_csv_file, 'r', encoding='utf-8-sig') as f:
    reader = csv.reader(f)
    data = [row for row in reader]

# Translate the data
for row in data:
    translated_row = []
    for cell in row:
        if cell:
            translation = translator.translate(cell, src='ar', dest='en')
            translated_row.append(translation.text)
        else:
            translated_row.append('')
    row[:] = translated_row

# Write the translated data to the output CSV file
with open(output_csv_file, 'w', encoding='utf-8-sig', newline='') as f:
    writer = csv.writer(f)
    writer.writerows(data)

```

Figure 10

2.3. Preprocessing:



```

# -----#
# defining a function to remove punctuations
def remove_punc(news_text):
    result = re.sub(r'[0-9]+', '', news_text)
    text = re.compile('[%s]' % re.escape(string.punctuation)).sub('', result)
    return text

# defining a function to remove stop words
def remove_stop(the_text):
    # Tokenize the text into words
    words = word_tokenize(the_text)

    # Filter out the stop words from the text
    filtered_words = [word for word in words if not word in stop_words]

    # Join the filtered words into a string
    filtered_text = ' '.join(filtered_words)

    # return the filtered text
    return(filtered_text)

# prepare spacy model for lemmatization
nlp = spacy.load('en_core_web_lg')
nlp = en_core_web_lg.load()

# defining Lemmatization function
def lemm(text):
    lemme=[]
    for token in nlp(text):
        lemme.append(token.lemma_)

    return " ".join(lemme)

# defining a function that combines all preprocessing steps needed
def preprocessing_text(text):
    text = remove_punc(text)
    text = remove_stop(text)
    text = lemm(text)
    return text

```

Figure 11

2.4. Text Classification:

Figure 12

```
[ ] from transformers import pipeline
model = pipeline('text-classification', model='Ammar-alhaj-ali/arabic-MARBERT-news-article-classification')

Downloading (...)lve/main/config.json: 100% [██████████] 1.22k/1.22k [00:00<00:00, 47.5kB/s]
Downloading pytorch_model.bin: 100% [██████████] 651M/651M [00:11<00:00, 63.0MB/s]
Downloading (...)kenizer_config.json: 100% [██████████] 371/371 [00:00<00:00, 2.31kB/s]
Downloading (...)solve/main/vocab.txt: 100% [██████████] 1.10M/1.10M [00:00<00:00, 4.39MB/s]
Downloading (...)main/tokenizer.json: 100% [██████████] 2.69M/2.69M [00:00<00:00, 8.66MB/s]
Downloading (...)cial_tokens_map.json: 100% [██████████] 125/125 [00:00<00:00, 1.82kB/s]
```

عمل على قناته اليوتيوب الرسمية لمصلحة الشرائب المصرية ، وكذلك على الموقع الإلكتروني للمصلحة من خلال " دليلك للتعامل معمنظومة توحيد معايير احتساب هربة الأجر و المترتبات" ، كما تهدف إلى العمل على الحد من وجود فروق الفحص عن طريق زيادة الدقة والالتزام بحساب وسداد الضريبة ، وتنقيف العيادة الإداري وتقليل تكلفة المعاملات وال الحاجة إلى إرهاص المستندات ورقياً ، القرية المستدقة بما يفهم في تحقيق العدالة بين جميع العاملين ، كما أنها تعمل على تحقيق المستهدفات المننشدة بتعزيز حوكمة المنظومة المالية للدولة واستبدال مستحقات الخزانة العامة

```
[ ] m=model(sentences)
print(m)

[{'label': 'Finance', 'score': 0.996180534362793}]
```

Figure 13

trained model

```
[ ] import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import re
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')
from nltk.stem import PorterStemmer
from nltk.stem import WordNetLemmatizer
nltk.download('wordnet')
from nltk.tokenize import word_tokenize
from nltk.tokenize import sent_tokenize
nltk.download('punkt')
from wordcloud import WordCloud
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.metrics import make_scorer, roc_curve, roc_auc_score
from sklearn.metrics import precision_recall_fscore_support as score
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.multiclass import OneVsRestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC, LinearSVC
from sklearn.tree import DecisionTreeClassifier
```

Figure 14

```
[ ] dataset=pd.read_csv("/content/drive/MyDrive/final_project_NTI/final_data_yarab_b2a")
[ ] dataset=dataset[['body', 'category']]
dataset
```

	body	category
0	Dr. Tariq Rahmi, Governor of Gharbia, met with...	الطباطب
1	The Department of Youth Development at the Dir...	الطباطب
2	Today, Friday, the Follow-up Department of the...	الطباطب
3	Ahmed Samir, President of the Egyptian Federat...	الطباطب
4	The Civil Protection Forces at Al-Hamoul Cente...	الطباطب
...
20225	The price of gold recorded today, Monday, in E...	الطباطب
20226	Today, Monday, the price of the euro witnessed...	الطباطب
20227	Minister of Manpower, Hassan Shehata, said tha...	الطباطب
20228	Mokhtar Tawfiq, Head of the Egyptian Tax Autho...	الطباطب
20229	The Sugar and Integrated Industries Company of...	الطباطب

20230 rows × 2 columns

Figure 15

```
[ ] dataset.shape
[20230, 2]

[ ] target_category = dataset['category'].unique()
print(target_category)
[محافظات, رياضية, عرب وعالم, آخر, ثقافة وفنون, حروات, اقتصاد]

[ ] dataset['CategoryId'] = dataset['category'].factorize()[0]

[ ] dataset
```

	body	category	CategoryId
0	Dr. Tariq Rahmi, Governor of Gharbia, met with...	محافظات	0
1	The Department of Youth Development at the Dir...	محافظات	0
2	Today, Friday, the Follow-up Department of the...	محافظات	0
3	Ahmed Samir, President of the Egyptian Federat...	محافظات	0
4	The Civil Protection Forces at Al-Hamoul Cente...	محافظات	0
...

Figure 16

```
20230 rows × 3 columns

[ ] # Create a new pandas dataframe "category", which only has unique Categories, also sorting this list in order of CategoryId values
category = dataset[['category', 'CategoryId']].drop_duplicates().sort_values('CategoryId')
category
```

	category	CategoryId
0	محافظات	0
2996	رياضية	1
5966	عرب وعالم	2
8937	آخر	3
11663	ثقافة وفنون	4
14494	حروات	5
17484	اقتصاد	6

```
[ ] dataset['category'].value_counts()
```

category	Count
محافظات	2996
حروات	2990
عرب وعالم	2971
رياضية	2970
ثقافة وفنون	2831
اقتصاد	2746
آخر	2726

Name: category, dtype: int64

Figure 17

+ Code + Text

```
[ ] from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import en_core_web_sm
import spacy
import re
import string

[ ] !python -m spacy download en_core_web_sm
nltk.download('punkt')
nltk.download('stopwords')
nlp = spacy.load('en_core_web_sm')

2023-05-09 18:36:17.390583: W tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Could not find TensorRT
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting en-core-web-sm==3.5.0
  Downloading https://github.com/explosion/spacy-models/releases/download/en_core_web_sm-3.5.0/en_core_web_sm-3.5.0-py3-none-any.whl (12.8 MB)
    12.8/12.8 MB 47.5 MB/s eta 0:00:00
Requirement already satisfied: spacy<3.6.0,>=3.5.0 in /usr/local/lib/python3.10/dist-packages (from en-core-web-sm==3.5.0) (3.5.2)
Requirement already satisfied: spacy-legacy<3.1.0,>>=3.0.11 in /usr/local/lib/python3.10/dist-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (3.0.12)
Requirement already satisfied: spacy-loggers<2.0.0,>>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (1.0.4)
Requirement already satisfied: murmurhash<1.1.0,>>=0.28.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (1.0.9)
Requirement already satisfied: cymem<2.1.0,>>=2.0.2 in /usr/local/lib/python3.10/dist-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (2.0.7)
Requirement already satisfied: preshed<3.1.0,>>=3.0.2 in /usr/local/lib/python3.10/dist-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (3.0.8)
Requirement already satisfied: thinc<8.2.0,>>=8.1.8 in /usr/local/lib/python3.10/dist-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (8.1.9)
Requirement already satisfied: wasabi<1.2.0,>>=0.9.1 in /usr/local/lib/python3.10/dist-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (1.1.1)
Requirement already satisfied: srslry<3.0.0,>>=2.4.3 in /usr/local/lib/python3.10/dist-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (2.4.6)
Requirement already satisfied: catalogue<2.1.0,>>=2.0.6 in /usr/local/lib/python3.10/dist-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (2.0.8)
Requirement already satisfied: typer<0.8.0,>>=0.7.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (0.7.0)
Requirement already satisfied: pathy<>0.10.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (0.10.1)
Requirement already satisfied: smart-open<7.0.0,>>=5.2.1 in /usr/local/lib/python3.10/dist-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (6.3.0)
Requirement already satisfied: ruamel<5.0.0,>>=4.38.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.6.0,>=3.5.0->en-core-web-sm==3.5.0) (4.65.0)
```

Figure 18

```
[ ] stop_words = set(stopwords.words('english'))

[ ] nlp = en_core_web_sm.load()

[ ] def remove_punc(news_text):
    result = re.sub(r'[0-9]+', '', news_text)
    text = re.compile('[%s]' % re.escape(string.punctuation)).sub('', result)
    return text

[ ] def remove_stop(the_text):
    # Tokenize the text into words
    words = word_tokenize(the_text)

    # Filter out the stop words from the text
    filtered_words = [word for word in words if not word in stop_words]

    # Join the filtered words into a string
    filtered_text = ' '.join(filtered_words)

    # return the filtered text
    return(filtered_text)

[ ] nlp = spacy.load('en_core_web_sm')
```

Figure 19

```

[ ] nlp = spacy.load('en_core_web_sm')
nlp = en_core_web_sm.load()

[ ] def lemm(text):
    lemme=[]
    for token in nlp(text):
        lemme.append(token.lemma_)

    return " ".join(lemme)

[ ] def preprocessing_text(text):
    text = remove_punc(text)
    text = remove_stop(text)
    text = lemm(text)
    return text

[ ] dataset['text']=dataset['body'].apply(preprocessing_text)

[ ] dataset['text']

0      Dr Tariq Rahmi Governor Gharbia meet Eng Nasse...
1      the Department Youth Development Directorate Y...
2      today Friday Followup Department Directorate H...
3      Ahmed Samir President Egyptian Federation Mini

```

Figure 20

```

[ ] data_prprocessed=dataset[['text','category','CategoryId']]
data_prprocessed.to_csv('/content/drive/MyDrive/final_project_NTI/Data/final data/preprocessed_final_insh2llah.csv')

Vectorization

[ ] !pip install gensim

Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: gensim in /usr/local/lib/python3.10/dist-packages (4.3.1)
Requirement already satisfied: numpy>=1.18.5 in /usr/local/lib/python3.10/dist-packages (from gensim) (1.22.4)
Requirement already satisfied: scipy>=1.7.0 in /usr/local/lib/python3.10/dist-packages (from gensim) (1.10.1)
Requirement already satisfied: smart-open>=1.8.1 in /usr/local/lib/python3.10/dist-packages (from gensim) (6.3.0)

[ ] import gensim
from gensim.models.doc2vec import Doc2Vec, TaggedDocument
from nltk.tokenize import word_tokenize

[ ] # Tokenize documents and create TaggedDocuments for Doc2Vec model
tagged_data = [TaggedDocument(words=word_tokenize(doc), tags=[str(i)]) for i, doc in enumerate(data_prprocessed['text'])]

# Train Doc2Vec model
model = Doc2Vec(tagged_data, vector_size=100, window=5, min_count=1, workers=4, epochs=20)

[ ] # Obtain vector representations for each document
vectors = [model.docvecs[str(i)] for i in range(len(data_prprocessed))]

```

Figure 21

```
[ ] # Tokenize documents and create TaggedDocuments for Doc2Vec model
tagged_data = [TaggedDocument(words=word_tokenize(doc), tags=[str(i)]) for i, doc in enumerate(data_prprocessed['text'])]

[ ] # Train Doc2Vec model
model1 = Doc2Vec(tagged_data, vector_size=100, window=5, min_count=1, workers=4, epochs=20)

[ ] # Obtain vector representations for each document
vectors = [model1.docvecs[str(i)] for i in range(len(data_prprocessed))]

<ipython-input-32-c5b1bbef54e>:2: DeprecationWarning: Call to deprecated `docvecs` (The `docvecs` property has been renamed `dv`).
  vectors = [model1.docvecs[str(i)] for i in range(len(data_prprocessed))]

[ ] vectors
[[array([-1.2097434e+00, 8.7511396e-01, 9.1965395e-01, -1.1184933e+00,
       4.1932475e-02, -3.5396212e-01, -6.5525037e-01, 7.5402391e-01,
      -4.4432843e-01, 1.8710719e-01, 7.1630347e-01, 5.8010572e-01,
      1.4167716e-01, 1.9786741e-01, 2.7465883e-01, 6.7876995e-02,
     -1.1639152e+00, 3.9724788e-01, 4.7962978e-01, 2.7686346e-01,
     -1.3112479e+00, 5.0851792e-01, -2.5025785e-01, 7.0076525e-01,
     -6.4901423e-01, 1.1748614e+00, -1.0963533e+00, 8.2570189e-01,
     -9.4997311e-01, -1.80772277e-01, -2.8372517e-01, -1.2950180e-01,
      7.0535280e-02, 7.5022814e-05, 1.2101258e-01, 9.2961574e-01,
     -1.2946986e+00, 1.7754875e-01, -2.3727182e-01, 2.9393286e-01,
      4.7066835e-01, -3.5687590e-01, -3.5261530e-01, -4.2689260e-02,
     6.1662191e-01, 5.0619924e-01, 7.1438199e-01, -6.9174469e-02,
      7.1927524e-01, -4.6188533e-02, -1.4313592e+00, 3.6452678e-01,
     -3.2074571e-01, 9.8570549e-01, 9.4615698e-01, 8.0956198e-02,
     1.1537598e+00, -4.1953057e-01, 4.7480330e-01, -7.1571702e-01,
     -5.2070241e-01, 6.70771422e-01, -2.07689790e-01, -2.1195691e-01]]
```

Figure 22

		text	category	CategoryId	vector
0	Dr Tariq Rahmi Governor Gharbia meet Eng Nasse...	محافظات	0	-1.2097434, 0.87511396, 0.91965395, -1.118493...	
1	the Department Youth Development Directorate Y...	محافظات	0	-0.26734543, 0.043513965, -0.0036009243, -0.3...	
2	today Friday Followup Department Directorate H...	محافظات	0	[...]	
3	Ahmed Samir President Egyptian Federation Mini...	محافظات	0	[...]	
4	the Civil Protection Forces AlHamoul Center Ka...	محافظات	0	[...]	
...	
20225	the price gold record today Monday Egypt stabl...	اقتصاد	6	[...]	
20226	today Monday price euro witness remarkable sta...	اقتصاد	6	[...]	
20227	Minister Manpower Hassan Shehata say sign join...	اقتصاد	6	[...]	
20228	Mokhtar Tawfiq Head Egyptian Tax Authority ann...	اقتصاد	6	[...]	
20229	the Sugar Integrated Industries Company Holdin...	اقتصاد	6	[...]	

20230 rows × 4 columns

```
[ ] x=np.array(data_prprocessed['vector'])
y=np.array(data_prprocessed['CategoryId'])
```

Figure 23

```
[ ] # Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(x,y, test_size=0.2, random_state=42, shuffle=True, stratify=y)

DecisionTree with doctovec

[ ] # Train decision tree classifier on training set
clf = DecisionTreeClassifier(random_state=42)
clf.fit(list(X_train), y_train)

*      DecisionTreeClassifier
DecisionTreeClassifier(random_state=42)

[ ] # Predict labels for testing set using trained classifier
y_pred = clf.predict(list(X_test))

[ ] # Calculate accuracy of predictions
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy}")

Accuracy: 0.3319327731092437

RandomForest with doctovec
```

Figure 24

```
RandomForest with doctovec

[ ] # Train random forest classifier on training set
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rf.fit(list(X_train), y_train)

*      RandomForestClassifier
RandomForestClassifier(random_state=42)

[ ] # Predict labels for testing set using trained random forest classifier
y_pred_rf = rf.predict(list(X_test))

# Calculate accuracy of predictions for random forest classifier
accuracy_rf = accuracy_score(y_test, y_pred_rf)
print(f"Accuracy of random forest classifier: {accuracy_rf}")

Accuracy of random forest classifier: 0.6178942165101334

Logistic Regression with doctovec

[ ] # Train logistic regression classifier on training set
lr = LogisticRegression(random_state=42)
lr.fit(list(X_train), y_train)
```

Figure 25

```
Logistic Regression with doctovec

[ ] # Train logistic regression classifier on training set
lr = LogisticRegression(random_state=42)
lr.fit(list(X_train), y_train)

*      LogisticRegression
LogisticRegression(random_state=42)

[ ] # Predict labels for testing set using trained logistic regression classifier
y_pred_lr = lr.predict(list(X_test))

# Calculate accuracy of predictions for logistic regression classifier
accuracy_lr = accuracy_score(y_test, y_pred_lr)
print(f"Accuracy of logistic regression classifier: {accuracy_lr}")

Accuracy of logistic regression classifier: 0.4587246663371231
```

Figure 26

↳ TFI-DF Embedding

```
[ ] from sklearn.feature_extraction.text import TfidfVectorizer
[ ] preprocess_df=pd.read_csv('/content/drive/MyDrive/final_project_NTI/Data/final_data/preprocessed_final_insh2llah.csv')
[ ] preprocess_df.drop('Unnamed: 0', axis=1,inplace=True)
[ ] preprocess_df
```

	text	category	CategoryId
0	Dr Tariq Rahmi Governor Gharbia meet Eng Nasse...	محافظات	0
1	the Department Youth Development Directorate Y...	محافظات	0
2	today Friday Followup Department Directorate H...	محافظات	0
3	Ahmed Samir President Egyptian Federation Mini...	محافظات	0
4	the Civil Protection Forces AlHamoul Center Ka...	محافظات	0
...
20225	the price gold record today Monday Egypt stabl...	الاقتصاد	6
20226	today Monday price euro witness remarkable sta...	الاقتصاد	6

Figure 27

```
[ ] # Preprocess data using TfidfVectorizer
vectorizer = TfidfVectorizer()
x_tf = vectorizer.fit_transform(preprocess_df['text'])

y_tf=preprocess_df['CategoryId']

[ ] # Split data into training and testing sets
X_trainn, X_testtt, y_trainn, y_testt = train_test_split(x_tf, y_tf, test_size=0.2, random_state=42,shuffle=True, stratify=y_tf)

[ ] # Train random forest classifier on training set
rf_tf = RandomForestClassifier(n_estimators=100, random_state=42)
rf_tf.fit(X_trainn, y_trainn)

RandomForestClassifier(random_state=42)

[ ] # Predict labels for testing set using trained random forest classifier
y_pred_TF_Random = rf_tf.predict(X_testtt)

[ ] # Calculate accuracy of predictions
accuracy = accuracy_score(y_testt, y_pred_TF_Random)
print(f"Accuracy: {accuracy}")
```

Figure 28

XGBOOST with TFIDF

```
[ ] df_preprocessed_edit=pd.read_csv('/content/drive/MyDrive/final_project_NTI/Data/final_data/preprocessed_final_insh2llah.csv')

[ ] df_preprocessed_edit
```

	Unnamed: 0	text	category	CategoryId
0	0	Dr Tariq Rahmi Governor Gharbia meet Eng Nasse...	الطبخ	0
1	1	the Department Youth Development Directorate Y...	الطبخ	0
2	2	today Friday Followup Department Directorate H...	الطبخ	0
3	3	Ahmed Samir President Egyptian Federation Mini...	الطبخ	0
4	4	the Civil Protection Forces AlHamoul Center Ka...	الطبخ	0
...
20225	20225	the price gold record today Monday Egypt stabl...	الاقتصاد	6
20226	20226	today Monday price euro witness remarkable sta...	الاقتصاد	6
20227	20227	Minister Manpower Hassan Shehata say sign join...	الاقتصاد	6
20228	20228	Mokhtar Tawfiq Head Egyptian Tax Authority ann...	الاقتصاد	6
20229	20229	the Sugar Integrated Industries Company Holdin...	الاقتصاد	6

20230 rows × 4 columns

Figure 29

```
[ ] x_g=df_preprocessed_edit['text']
y_g=df_preprocessed_edit['CategoryId']

[ ] # Split data into training and testing sets
X_trainxg, X_testxg, y_trainxg, y_testxg = train_test_split(x_g, y_g, test_size=0.05, random_state=42,shuffle=True, stratify=y_g)

[ ] # Preprocess data using TfIdfVectorizer
vectorizer_tf = TfIdfVectorizer(max_features=5000,ngram_range=(1,3))
X_train_gb = vectorizer_tf.fit_transform(X_trainxg)
X_test_gb = vectorizer_tf.transform(X_testxg)

[ ] import xgboost as xgb

[ ] # Initialize XGBClassifier object
xg = xgb.XGBClassifier()

# Train the XGBoost model
xg.fit(X_train_gb, y_trainxg)
```

```
[ ] XGBClassifier
XGBClassifier(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, early_stopping_rounds=None,
```

Figure 30

```

Q [ ] # Preprocess data using TfidfVectorizer
vectorizer = TfidfVectorizer(max_features=5000,ngram_range=(1,3))
x = vectorizer.fit_transform(x_g)
{x}

D [ ] xg.fit(x, y_g)

XGBClassifier
[XGBClassifier(base_score=None, booster=None, callbacks=None,
   colsample_bylevel=None, colsample_bynode=None,
   colsample_bytree=0.8, early_stopping_rounds=None,
   enable_categorical=False, eval_metric='mlogloss',
   feature_types=None, gamma=None, gpu_id=None, grow_policy=None,
   importance_type=None, interaction_constraints=None,
   learning_rate=0.1, max_bin=None, max_cat_threshold=None,
   max_cat_to_onehot=None, max_delta_step=None, max_depth=6,
   max_leaves=None, min_child_weight=None, missing=nan,
   monotone_constraints=None, n_estimators=100, n_jobs=None,
   num_parallel_tree=None, objective='multi:softmax', predictor=None, ...)]
```

```

[ ] xg = xgb.XGBClassifier(
   max_depth=6,
   learning_rate=0.1,
   n_estimators=100,
   subsample=0.8,
   colsample_bytree=0.8,
   objective='multi:softmax',
   eval_metric='mlogloss')
```

Figure 31

```

Q [ ] n_estimators=100,
subsample=0.8,
colsample_bytree=0.8,
objective='multi:softmax',
eval_metric='mlogloss')

{x}

D [ ] xg.fit(X_train_gb, y_traingxg)

XGBClassifier
[XGBClassifier(base_score=None, booster=None, callbacks=None,
   colsample_bylevel=None, colsample_bynode=None,
   colsample_bytree=0.8, early_stopping_rounds=None,
   enable_categorical=False, eval_metric='mlogloss',
   feature_types=None, gamma=None, gpu_id=None, grow_policy=None,
   importance_type=None, interaction_constraints=None,
   learning_rate=0.1, max_bin=None, max_cat_threshold=None,
   max_cat_to_onehot=None, max_delta_step=None, max_depth=6,
   max_leaves=None, min_child_weight=None, missing=nan,
   monotone_constraints=None, n_estimators=100, n_jobs=None,
   num_parallel_tree=None, objective='multi:softmax', predictor=None, ...)]
```

```

[ ] # Make predictions on the testing set
y_pred_xg = xg.predict(X_testgb)

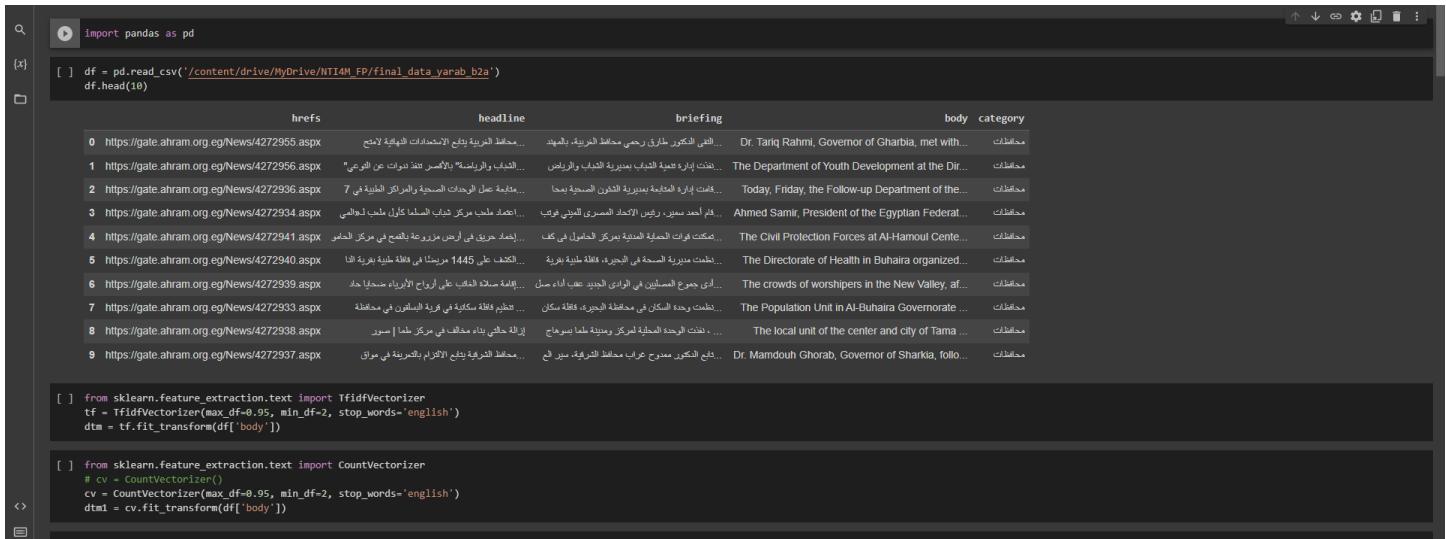
# Evaluate the accuracy of the model
accuracy_xg = accuracy_score(y_testgb, y_pred_xg)
print("Accuracy: %.2f%%" % (accuracy_xg * 100.0))

Accuracy: 93.08%
```

Figure 32

2.5. Recommendation:

LDA:



The screenshot shows a Jupyter Notebook interface with several code cells and a table of news articles.

```

import pandas as pd
df = pd.read_csv('/content/drive/MyDrive/NTI4M_FP/final_data_yarab_b2a')
df.head(10)

```

hrefs	headline	briefing	body	category
0 https://gate.ahram.org.eg/News/4272955.aspx	الى، النكورة، مارق، رحبي، محافظ، الغربية، الهاشمية، بهاء الدين، لامج	محافظ الغربية يتابع الاستعدادات الهاشمية لامتحان...	Dr. Tariq Rahmi, Governor of Gharbia, met with...	محافظات
1 https://gate.ahram.org.eg/News/4272956.aspx	النادل، والبرادعي، الدباب، بدرية، إقبال، والبرادعي	يختتم، برادعي، الدباب، بدرية، إقبال، والبرادعي...	The Department of Youth Development at the Dir...	محافظات
2 https://gate.ahram.org.eg/News/4272936.aspx	الإمامية، بيبريرية، الشفرون، المصوحة، بمحـا	افتـ، إدارـة، إيمـاـة، بـيـبـرـيـرـيـة، الشـفـرـونـ، المصـوـحـةـ، بـمـحـاـ...	Today, Friday, the Follow-up Department of the...	محافظات
3 https://gate.ahram.org.eg/News/4272934.aspx	الـأـصـدـيـرـ، وـرـيـسـ، الـأـسـدـيـرـ، الـصـيـرـ، الـلـيـهـ، وـرـيـسـ	ـاـصـدـيـرـ، مـرـكـ، شـيـبـ، الـلـيـهـ، كـاـلـيـ، طـبـ، الـلـيـهـ...	Ahmed Samir, President of the Egyptian Federat...	محافظات
4 https://gate.ahram.org.eg/News/4272941.aspx	ـاـصـدـيـرـ، حـرـيـ، فـيـ، اـصـدـيـرـ، مـرـكـ، الـلـيـهـ، بـلـاـجـ، فـيـ، الـلـيـهـ	ـاـصـدـيـرـ، حـرـيـ، فـيـ، اـصـدـيـرـ، مـرـكـ، الـلـيـهـ، بـلـاـجـ، فـيـ، الـلـيـهـ...	The Civil Protection Forces at Al-Hamouli Cente...	محافظات
5 https://gate.ahram.org.eg/News/4272940.aspx	ـاـصـدـيـرـ، مـرـيـضـ، فـيـ، الـلـيـهـ، بـلـاـجـ، الـلـيـهـ	ـاـصـدـيـرـ، مـرـيـضـ، فـيـ، الـلـيـهـ، بـلـاـجـ، الـلـيـهـ...	The Directorate of Health in Buhaira organized...	محافظات
6 https://gate.ahram.org.eg/News/4272939.aspx	ـاـصـدـيـرـ، مـوـعـ، الـصـلـيـونـ، فـيـ، الـلـيـهـ، عـقـبـ، اـمـ، سـلـ	ـاـصـدـيـرـ، مـوـعـ، الـصـلـيـونـ، فـيـ، الـلـيـهـ، عـقـبـ، اـمـ، سـلـ...	The crowds of worshipers in the New Valley, af...	محافظات
7 https://gate.ahram.org.eg/News/4272933.aspx	ـاـصـدـيـرـ، مـكـاـنـ، فـيـ، مـهـافـقـةـ، الـجـيـرـ، فـكـلـاـتـ، مـكـاـنـ	ـاـصـدـيـرـ، مـكـاـنـ، فـيـ، مـهـافـقـةـ، الـجـيـرـ، فـكـلـاـتـ، مـكـاـنـ...	The Population Unit in Al-Buhaira Governorate ...	محافظات
8 https://gate.ahram.org.eg/News/4272938.aspx	ـاـصـدـيـرـ، مـكـاـنـ، فـيـ، مـهـافـقـةـ، الـجـيـرـ، فـكـلـاـتـ، مـكـاـنـ	ـاـصـدـيـرـ، مـكـاـنـ، فـيـ، مـهـافـقـةـ، الـجـيـرـ، فـكـلـاـتـ، مـكـاـنـ...	The local unit of the center and city of Tama...	محافظات
9 https://gate.ahram.org.eg/News/4272937.aspx	ـاـصـدـيـرـ، مـكـاـنـ، فـيـ، مـهـافـقـةـ، الـجـيـرـ، فـكـلـاـتـ، مـكـاـنـ	ـاـصـدـيـرـ، مـكـاـنـ، فـيـ، مـهـافـقـةـ، الـجـيـرـ، فـكـلـاـتـ، مـكـاـنـ...	Dr. Mamdouh Ghorab, Governor of Sharqia, follo...	محافظات

```

from sklearn.feature_extraction.text import TfidfVectorizer
tf = TfidfVectorizer(max_df=0.95, min_df=2, stop_words='english')
dtm = tf.fit_transform(df['body'])

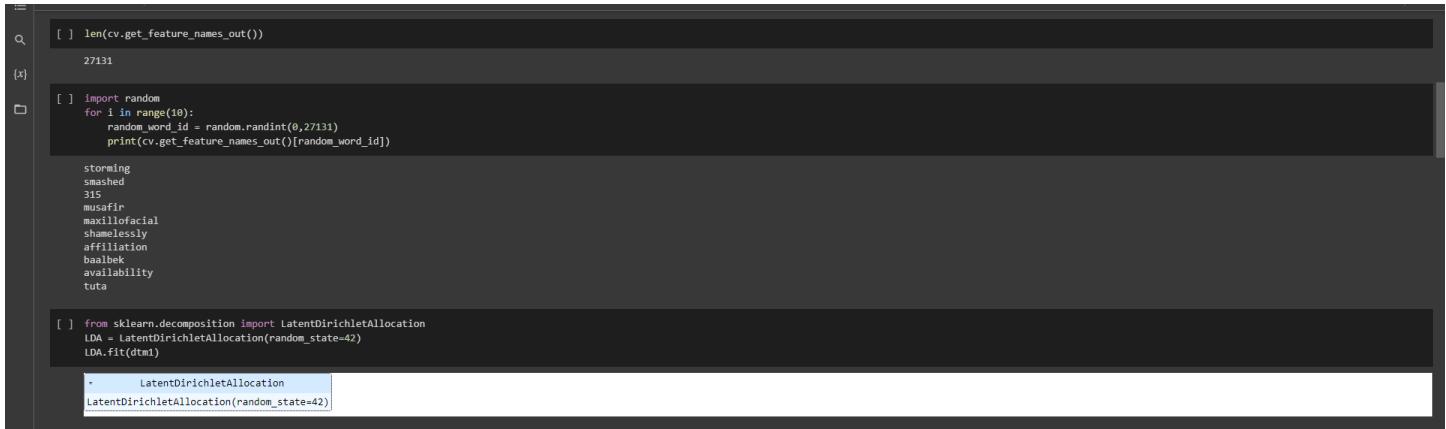
```

```

from sklearn.feature_extraction.text import CountVectorizer
# cv = CountVectorizer()
cv = CountVectorizer(max_df=0.95, min_df=2, stop_words='english')
dtm1 = cv.fit_transform(df['body'])

```

Figure 33



The screenshot shows a Jupyter Notebook interface with several code cells.

```

len(cv.get_feature_names_out())
27131

```

```

import random
for i in range(10):
    random_word_id = random.randint(0,27131)
    print(cv.get_feature_names_out()[random_word_id])

```

```

stomping
smashed
315
musafir
maxillofacial
shamelessly
affiliation
baalbek
availability
tuta

```

```

from sklearn.decomposition import LatentDirichletAllocation
LDA = LatentDirichletAllocation(random_state=42)
LDA.fit(dtm)

```

```

LatentDirichletAllocation(random_state=42)

```

Figure 34

```

Q len(LDA.components_)

[ ] 10

[X] LDA.components_
array([[7.36080413e+01, 1.88888260e+00, 1.00691154e-01, ...,
       2.7099585e+01, 1.00000001e-01, 1.00089853e-01, ...,
       1.00491409e+03, 6.42346171e+00, 1.00000000e-01, ...,
       1.00000000e-01, 1.00000000e-01, 1.00000000e-01, ...,
       1.00000000e-01, 1.44823408e+00, 1.00000000e-01, ...,
       1.00028957e-01, 1.00000001e-01, 1.00002494e-01, ...,
       ...,
       [1.00000000e-01, 1.03995156e+02, 1.00000000e-01, ...,
       1.00000000e-01, 2.00017504e+00, 1.00000000e-01, ...,
       1.01163195e-01, 2.58025148e+02, 1.00000000e-01, ...,
       1.00000000e-01, 1.00000000e-01, 1.00000000e-01, ...,
       [2.5764363e+01, 5.60702767e+02, 1.00000000e-01, ...,
       1.00012506e-01, 1.00000001e-01, 2.69090765e+00]]]

[X] len(LDA.components_) , len(LDA.components_[0])

[ ] (10, 27131)

[X] for index,topic in enumerate(LDA.components_):
    print("THE TOP 15 WORDS FOR TOPIC #"+str(index))
    print((cv.get_feature_names_out()[i] for i in topic.argsort()[-15:]))
    print("\n")

THE TOP 15 WORDS FOR TOPIC #0
['stadium', 'matches', 'minute', 'goals', 'scored', 'goal', 'round', 'ahly', 'place', 'second', 'team', 'league', 'match', 'points', 'al']

<> THE TOP 15 WORDS FOR TOPIC #1
['el', 'god', 'air', 'work', 'departure', 'train', 'cairo', 'muhammad', 'ramadan', 'abdel', 'artist', 'time', 'ahmed', 'series', 'al']

```

Figure 35

```

Q topic_results = LDA.transform(dtmi)

[X] topic_results.shape
(20230, 10)

[X] topic_results[0].round(2)
array([0.02, 0.01, 0. , 0.62, 0. , 0.19, 0.03, 0.04, 0.05, 0.04])

[X] topic_results[0].argmax()
3

[X] topic_results.argmax(axis=1)
array([3, 3, 3, ..., 5, 5, 8])

[X] ex = df['body'][11]
array([0. , 0. , 0. , 0.15, 0. , 0.14, 0. , 0.06, 0.64, 0. ])

[X] ex.round(2)
array([[0. , 0. , 0. , 0.15, 0. , 0.14, 0. , 0.06, 0.64, 0. ]])

[X] ex.argmax()

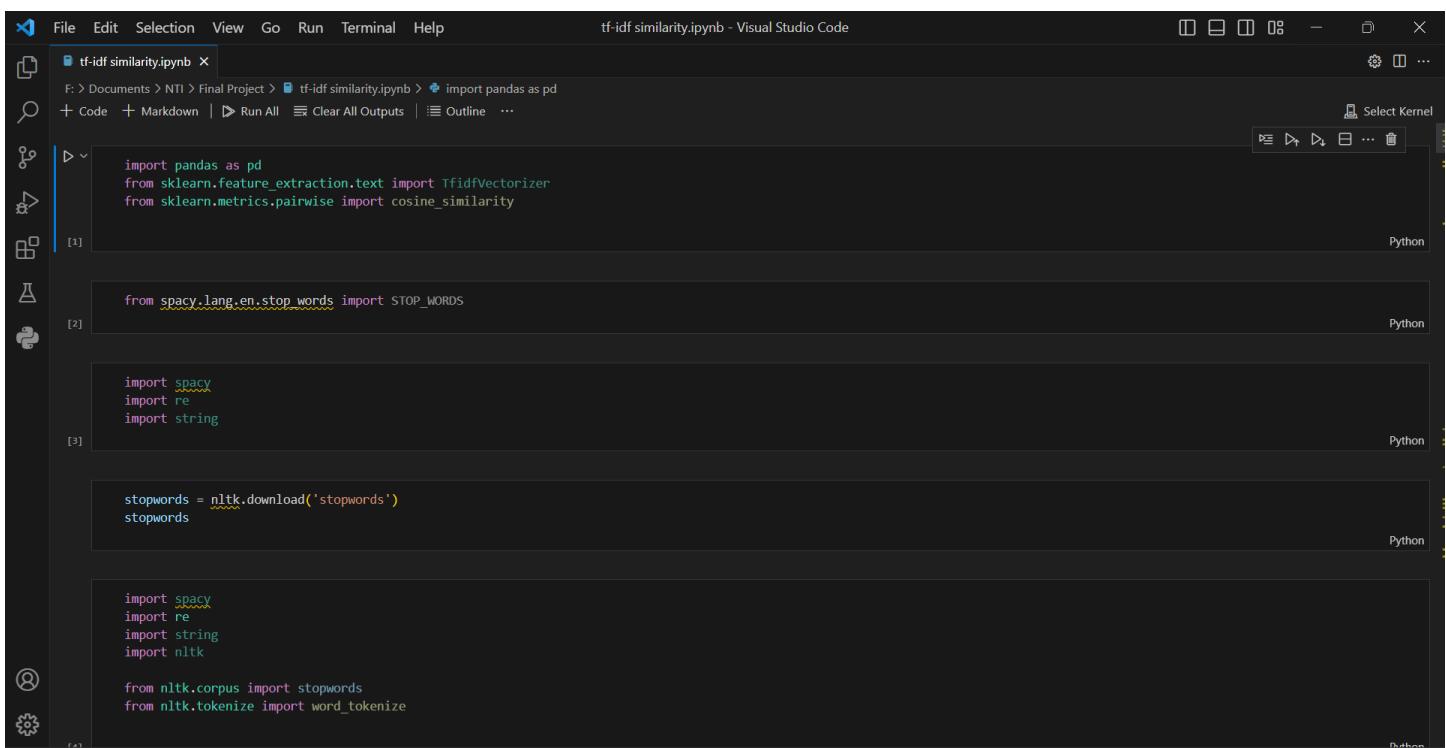
```

Figure 36

Topic	Category	Count
السيسي	محافظات	3
السيسي	محافظات	7
السيسي	محافظات	3
السيسي	محافظات	3
السيسي	محافظات	1
السيسي	محافظات	3
السيسي	محافظات	7
السيسي	محافظات	7
السيسي	محافظات	3
السيسي	محافظات	3
السيسي	محافظات	3
السيسي	محافظات	7

Figure 37

Tf-idf similarity:



```

File Edit Selection View Go Run Terminal Help tf-idf similarity.ipynb - Visual Studio Code
tf-idf similarity.ipynb × Select Kernel
F: > Documents > NTI > Final Project > tf-idf similarity.ipynb import pandas as pd
+ Code + Markdown | Run All Clear All Outputs Outline ...
[1]
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity

[2]
from spacy.lang.en.stop_words import STOP_WORDS

[3]
import spacy
import re
import string
stopwords = nltk.download('stopwords')
stopwords

import spacy
import re
import string
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

```

Figure 38



The screenshot shows a Visual Studio Code interface with a Jupyter Notebook open. The notebook has the title "tf-idf similarity.ipynb". The code cells contain Python code related to spaCy and pandas, including downloading a language model, importing spaCy, reading a CSV file, and performing data manipulation. The code is run in a "Python" kernel.

```
!python -m spacy download en_core_web_lg
import en_core_web_sm
import en_core_web_lg
news_df = pd.read_csv('final_data.csv')
news_df.columns
... Index(['Unnamed: 0', 'hrefs', 'headline', 'briefing', 'body', 'category'], dtype='object')
news_df.drop('Unnamed: 0', axis=1, inplace=True)
news_df.tail(10)
```

Figure 39



File Edit Selection View Go Run Terminal Help tf-idf similarity.ipynb - Visual Studio Code

F: > Documents > NTI > Final Project > tf-idf similarity.ipynb > import pandas as pd

+ Code + Markdown | Run All Clear All Outputs Outline ...

news_df.tail(10)

... hrefs headline briefing body category

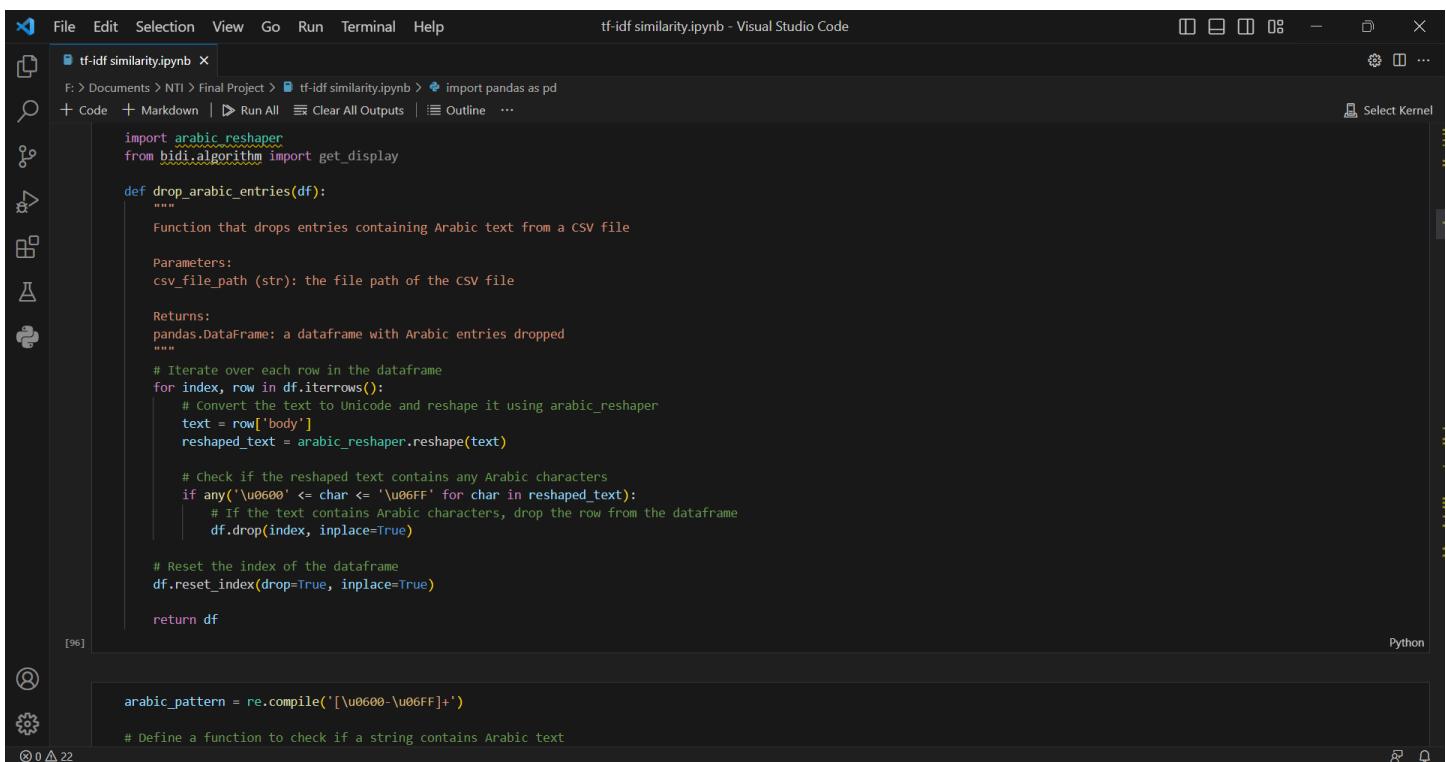
20994 https://gate.ahram.org.eg/News/4122898.aspx ... قال رامي الدكани رئيس البورصة المصرية، إن سوق ... Rami El Dokani, Chairman of the Egyptian Stock... اقتصاد
20995 https://gate.ahram.org.eg/News/4122861.aspx ... عقد الوفد المصري الذي يضم القادات التنفيذية ل ... The Egyptian delegation, which includes the ex... اقتصاد
20996 https://gate.ahram.org.eg/News/4122808.aspx ... اهتمام حاولي كبير بالشركات المصرية المطروفة ... Dr. Rania Al-Mashat, Minister of Internation... اقتصاد
20997 https://gate.ahram.org.eg/News/4122801.aspx ... استقبلت الدكتورة رانيا المشاط، وزيرة التعاون ا ... الفيومي: الرقة الإلكترونية على المساع تشهد في ... Representative Dr. Mohamed Attia Al-Fayoumi, P... اقتصاد
20998 https://gate.ahram.org.eg/News/4122730.aspx ... وزيرة التخطيط: معظم مشروعات الدولة تستهدف تنمي ... شاركت الدكتورة هالة السعيد، وزيرة التخطيط والت ... اقتصاد
20999 https://gate.ahram.org.eg/News/4122711.aspx ... سعر الذهب، اليوم الاثنين، في مصر، عيار 21 يسجل ... The price of gold recorded today, Monday, in E... اقتصاد
21000 https://gate.ahram.org.eg/News/4122703.aspx ... شهد سعر اليورو اليوم الإثنين، 20 فبراير 2023 في الس ... Today, Monday, the price of the euro witnessed... اقتصاد
21001 https://gate.ahram.org.eg/News/4122674.aspx ... ووفد من العرف التجاري بيحدث بطلب توقيع ... قال وزير القوى العمالة حسين شحاته، إنه ي Trident of Manpower, Hassan Shehata, said tha... اقتصاد
21002 https://gate.ahram.org.eg/News/4122669.aspx ... الصارثي، ورش العمل لتقدير الدعم الفني للممول ... مأمور مختار توفيق، رئيس مصلحة الضرائب المصرية ... Mokhtar Tawfiq, Head of the Egyptian Tax Autho... اقتصاد
21003 https://gate.ahram.org.eg/News/4122662.aspx ... وزارة التقويم: ارتفاع معدل توريد القصب إلى 3 ... أكدت شركة السكر والصناعات التكاملية التابعة لل ... The Sugar and Integrated Industries Company of... اقتصاد

news_df.category.value_counts()

import arabic_reshaper
from bidi.algorithm import get_display

def drop_arabic_entries(df):

Figure 40



The screenshot shows a Visual Studio Code interface with a Python script named `tf-idf similarity.ipynb`. The code defines a function `drop_arabic_entries` that iterates over rows of a DataFrame, converts the 'body' column to Unicode, reshapes it using `arabic_reshaper`, and then checks if any character is an Arabic character (U0600-U06FF). If found, the row is dropped. Finally, the index is reset. A regular expression pattern is also defined to identify Arabic characters.

```

import arabic_reshaper
from bidi.algorithm import get_display

def drop_arabic_entries(df):
    """
    Function that drops entries containing Arabic text from a CSV file

    Parameters:
    csv_file_path (str): the file path of the csv file

    Returns:
    pandas.DataFrame: a dataframe with Arabic entries dropped
    """
    # Iterate over each row in the dataframe
    for index, row in df.iterrows():
        # Convert the text to Unicode and reshape it using arabic_reshaper
        text = row['body']
        reshaped_text = arabic_reshaper.reshape(text)

        # Check if the reshaped text contains any Arabic characters
        if any('\u0600' <= char <= '\u06FF' for char in reshaped_text):
            # If the text contains Arabic characters, drop the row from the dataframe
            df.drop(index, inplace=True)

    # Reset the index of the dataframe
    df.reset_index(drop=True, inplace=True)

    return df

```

[96]

```

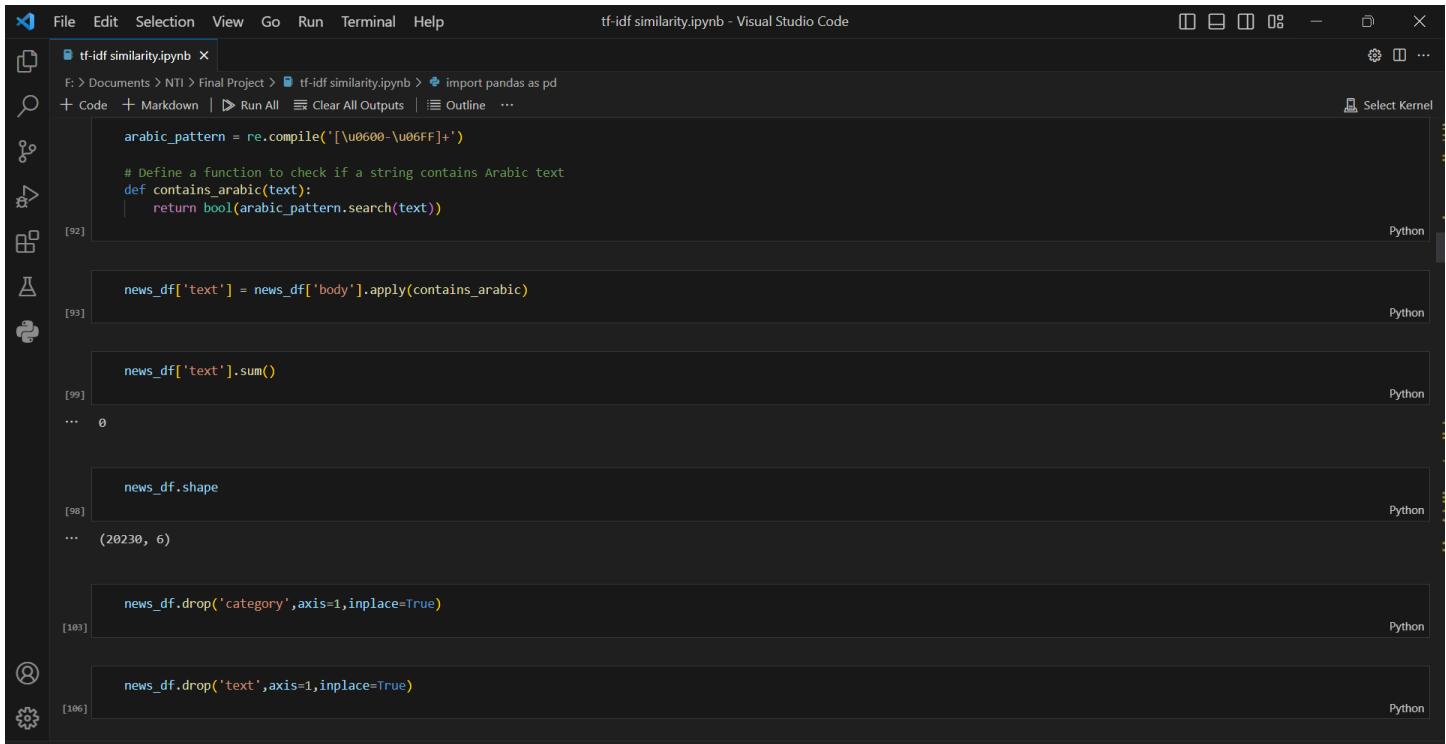
arabic_pattern = re.compile('[\u0600-\u06FF]+')

# Define a function to check if a string contains Arabic text

```

⑧ 0 △ 22

Figure 41



The screenshot shows a Visual Studio Code interface with a Python script named `tf-idf similarity.ipynb`. The code defines a function `contains_arabic` that uses a regular expression to search for Arabic characters in a string. It then applies this function to the 'body' column of a DataFrame named `news_df`. The DataFrame is then summed, resulting in a value of 0. The shape of the DataFrame is checked, showing dimensions of (20230, 6). Finally, the 'category' and 'text' columns are dropped from the DataFrame.

```

arabic_pattern = re.compile('[\u0600-\u06FF]+')

# Define a function to check if a string contains Arabic text
def contains_arabic(text):
    return bool(arabic_pattern.search(text))

news_df['text'] = news_df['body'].apply(contains_arabic)

news_df['text'].sum()

... 0

news_df.shape
... (20230, 6)

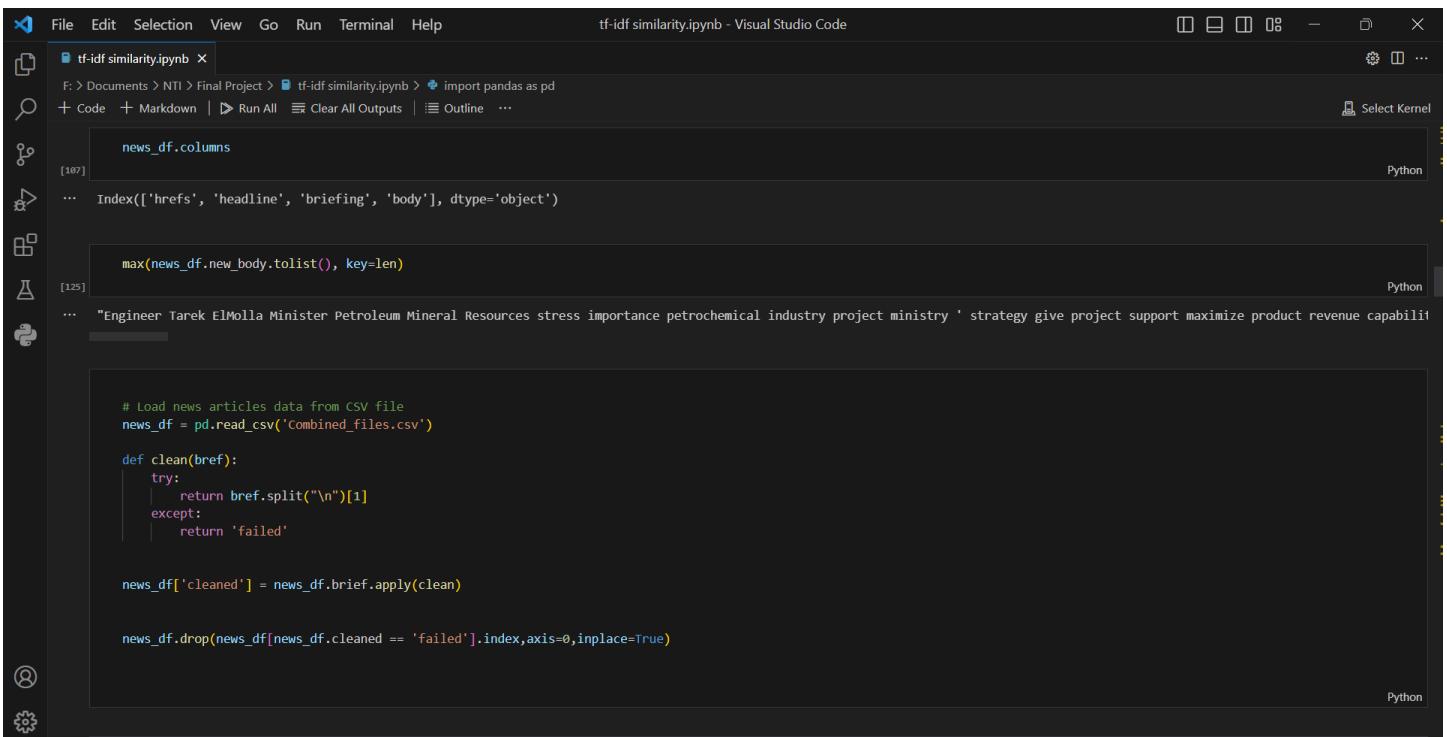
news_df.drop('category', axis=1, inplace=True)

news_df.drop('text', axis=1, inplace=True)

```

⑧ 106

Figure 42



```

File Edit Selection View Go Run Terminal Help tf-idf similarity.ipynb - Visual Studio Code
tf-idf similarity.ipynb × F: > Documents > NTI > Final Project > tf-idf similarity.ipynb > import pandas as pd
+ Code + Markdown | ▶ Run All ⌂ Clear All Outputs | ⌂ Outline ...
news_df.columns [107] Select Kernel Python
... Index(['hrefs', 'headline', 'briefing', 'body'], dtype='object')

max(news_df.new_body.tolist(), key=len) [125] Python
... "Engineer Tarek ElMolla Minister Petroleum Mineral Resources stress importance petrochemical industry project ministry ' strategy give project support maximize product revenue capabiliti

# Load news articles data from CSV file
news_df = pd.read_csv('Combined_files.csv')

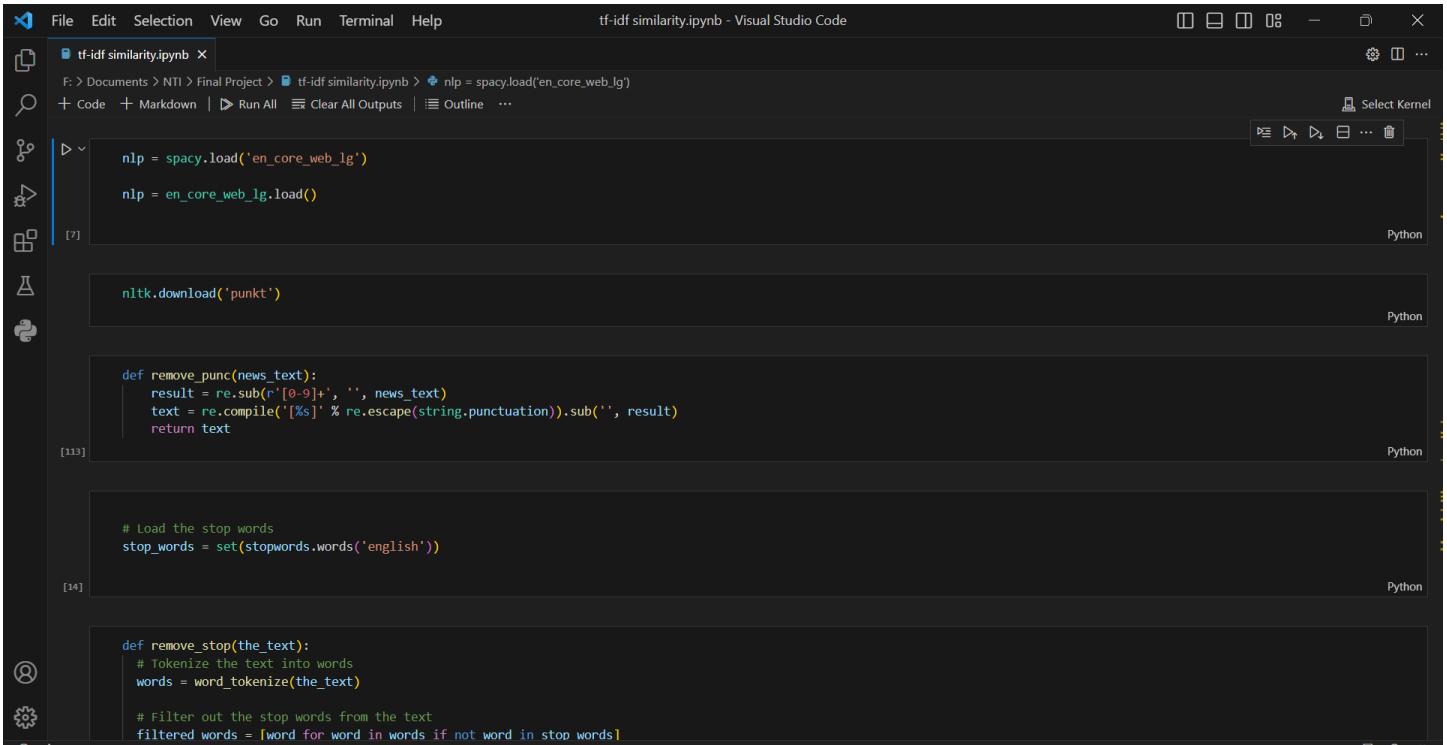
def clean(bref):
    try:
        return bref.split("\n")[1]
    except:
        return 'failed'

news_df['cleaned'] = news_df.brief.apply(clean)

news_df.drop(news_df[news_df.cleaned == 'failed'].index, axis=0, inplace=True)

```

Figure 43



```

File Edit Selection View Go Run Terminal Help tf-idf similarity.ipynb - Visual Studio Code
tf-idf similarity.ipynb × F: > Documents > NTI > Final Project > tf-idf similarity.ipynb > nlp = spacy.load('en_core_web_lg')
+ Code + Markdown | ▶ Run All ⌂ Clear All Outputs | ⌂ Outline ...
nlp = spacy.load('en_core_web_lg') [7] Select Kernel Python
nlp = en_core_web_lg.load()

nltk.download('punkt')

def remove_punc(news_text):
    result = re.sub(r'[0-9]+', '', news_text)
    text = re.compile('[%s]' % re.escape(string.punctuation)).sub('', result)
    return text

# Load the stop words
stop_words = set(stopwords.words('english'))

def remove_stop(the_text):
    # Tokenize the text into words
    words = word_tokenize(the_text)

    # Filter out the stop words from the text
    filtered_words = [word for word in words if not word in stop_words]

```

Figure 44



The screenshot shows a Visual Studio Code interface with the following details:

- Title Bar:** tf-idf similarity.ipynb - Visual Studio Code
- File Path:** F:\Documents > NTI > Final Project > tf-idf similarity.ipynb > def preprocessing_text(text):
- Code Editor:** The main pane displays Python code:

```
def preprocessing_text(text):
    text = remove_punc(text)
    text = remove_stop(text)
    text = lemm(text)
    return text
```
- Output Panel:** Shows three code cells with their execution numbers and results:
 - [116] Python:

```
news_df['new_body'] = news_df['body'].apply(preprocessing_text)
```
 - [122] Python:

```
news_df.head(5)
```
 - [123] Python:

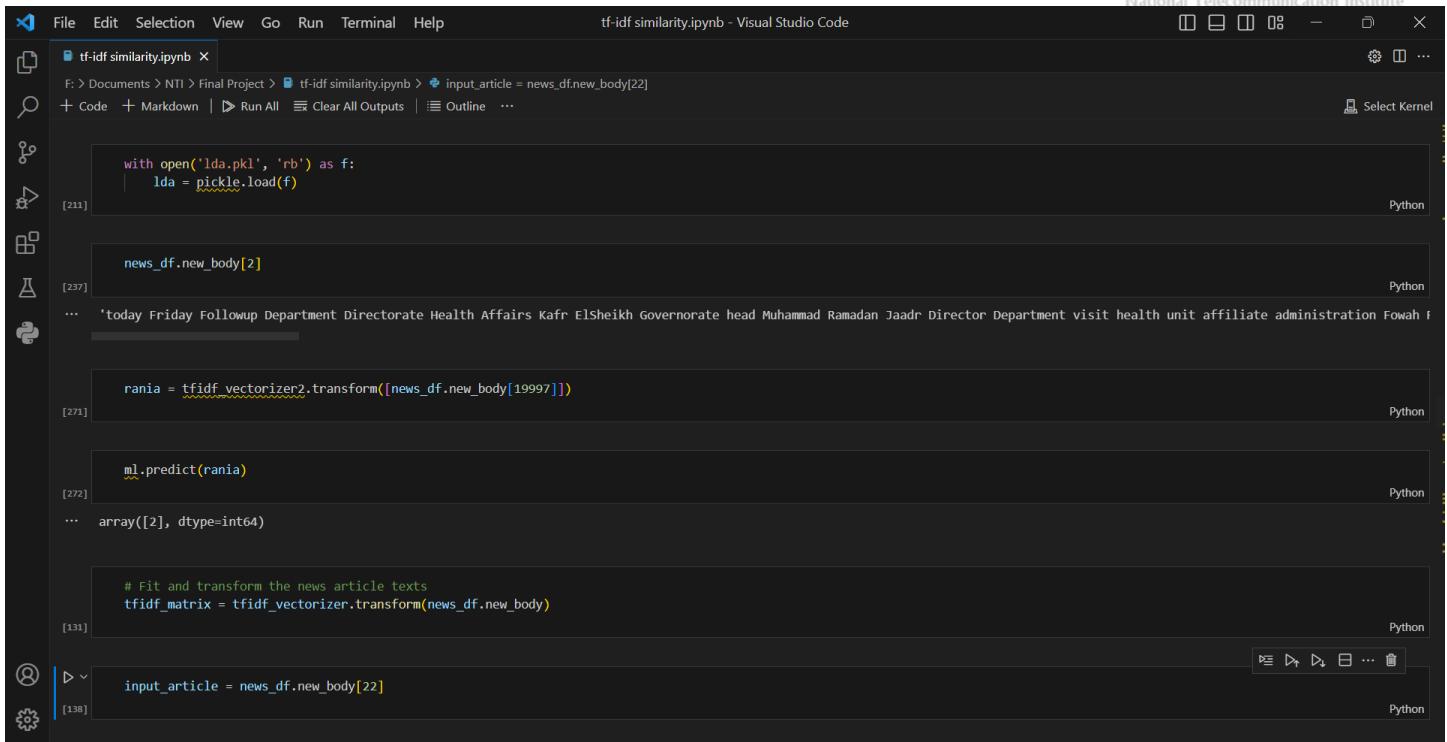
```
... hrefs headline briefing body new_body
0 https://gate.ahram.org.eg/News/427295.aspx محافظ الغربية يتابع الاستعدادات النهائية... التony... Dr. Tariq Rahmi, Governor of Gharbia, met with... Dr Tariq Rahmi Governor Gharbia meet Eng Nasse...
1 https://gate.ahram.org.eg/News/4272956.aspx الشاب والرياضة" بالآخر تنفذ ندوات عن"... نفذت إدارة تنمية الشباب بمديرية الشباب... The Department of Youth Development at the Dir...
2 https://gate.ahram.org.eg/News/4272936.aspx متابعة عمل الوحدات الصناعية والمراكز الطبية... قاتت إدارة المتابعة بمديرية الشئون... Today, Friday, the Follow-up Department of the...
3 https://gate.ahram.org.eg/News/4272934.aspx اعتماد ملعب مركز شباب الصالوة كأول ملعب... قات أحمد سمير, رئيس الاتحاد المصري... Ahmed Samir, President of the Egyptian Federat...
4 https://gate.ahram.org.eg/News/4272941.aspx إنجام حريق في أرض مزروعة بالقمح في... تمتクト قوات الحماية المدنية بمراكز الجامول... The Civil Protection Forces at Al-Hamoul Cente...
... فى كفر...
```
- Bottom Status Bar:** 0 △ 22
- Bottom Right:** linter: unsupported

Figure 45

The screenshot shows the Visual Studio Code interface with a Jupyter Notebook open. The title bar reads "tf-idf similarity.ipynb - Visual Studio Code". The left sidebar has icons for file operations like Open, Save, and Close, as well as a search icon. The main area displays the notebook code in a dark theme:

```
File Edit Selection View Go Run Terminal Help tf-idf similarity.ipynb - Visual Studio Code F: > Documents > NTI > Final Project > tf-idf similarity.ipynb > news_df.drop('body', axis=1, inplace=True) + Code + Markdown | Run All | Clear All Outputs | Outline ... Select Kernel Python news_df.drop('body', axis=1, inplace=True) [126] news_df.columns [127] ... Index(['hrefs', 'headline', 'briefing', 'new_body'], dtype='object') news_df.to_csv('app_data.csv') [147] # Initialize a TfidfVectorizer tfidf_vectorizer = TfidfVectorizer(max_features=5000, ngram_range=(1, 3)) [128] tfidf_vectorizer.fit(news_df.new_body) [130] ... TfidfVectorizer(max_features=5000, ngram_range=(1, 3)) from joblib import dump, load [214] with open('lda.pkl', 'rb') as f:
```

Figure 46



File Edit Selection View Go Run Terminal Help

tf-idf similarity.ipynb - Visual Studio Code

F: > Documents > NTI > Final Project > tf-idf similarity.ipynb > # input_article = news_df.new_body[22]

+ Code + Markdown | ▶ Run All ⌘ Clear All Outputs | ⌥ Outline ...

Select Kernel

```

with open('lda.pkl', 'rb') as f:
    lda = pickle.load(f)

news_df.new_body[2]
... 'today Friday Followup Department Directorate Health Affairs Kafr Elsheikh Governorate head Muhammad Ramadan Jaadr Director Department visit health unit affiliate administration Fowah i

rania = tfidf_vectorizer2.transform([news_df.new_body[19997]])
... ml.predict(rania)
... array([2], dtype=int64)

# Fit and transform the news article texts
tfidf_matrix = tfidf_vectorizer.transform(news_df.new_body)

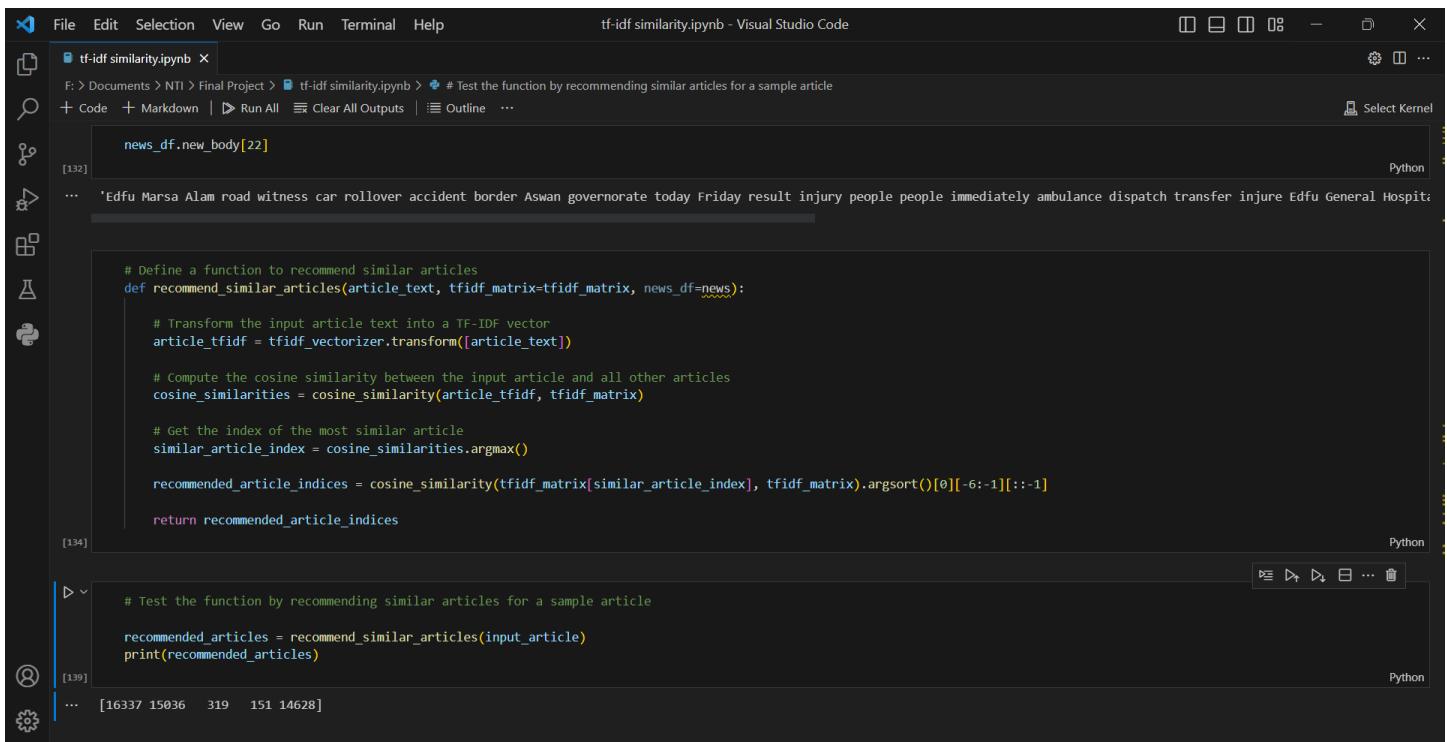
```

[131]

input_article = news_df.new_body[22]

[138]

Figure 47



File Edit Selection View Go Run Terminal Help

tf-idf similarity.ipynb - Visual Studio Code

F: > Documents > NTI > Final Project > tf-idf similarity.ipynb > # Test the function by recommending similar articles for a sample article

+ Code + Markdown | ▶ Run All ⌘ Clear All Outputs | ⌥ Outline ...

Select Kernel

```

news_df.new_body[22]
... 'Edfu Marsa Alam road witness car rollover accident border Aswan governorate today Friday result injury people people immediately ambulance dispatch transfer injure Edfu General Hospital

# Define a function to recommend similar articles
def recommend_similar_articles(article_text, tfidf_matrix=tfidf_matrix, news_df=news):

    # Transform the input article text into a TF-IDF vector
    article_tfidf = tfidf_vectorizer.transform([article_text])

    # Compute the cosine similarity between the input article and all other articles
    cosine_similarities = cosine_similarity(article_tfidf, tfidf_matrix)

    # Get the index of the most similar article
    similar_article_index = cosine_similarities.argmax()

    recommended_article_indices = cosine_similarity(tfidf_matrix[similar_article_index], tfidf_matrix).argsort()[0][-6:-1][::-1]

    return recommended_article_indices

```

[134]

Test the function by recommending similar articles for a sample article

```

recommended_articles = recommend_similar_articles(input_article)
print(recommended_articles)
... [16337 15036 319 151 14628]

```

[139]

Figure 48

File Edit Selection View Go Run Terminal Help tf-idf similarity.ipynb - Visual Studio Code

tf-idf similarity.ipynb X F: > Documents > NTI > Final Project > tf-idf similarity.ipynb > # Define the two texts to compare

+ Code + Markdown | ▶ Run All ⌘ Clear All Outputs | ⌥ Outline ... Select Kernel

```
[142] news_df.news_body[319]
... 'today Suez Road witness microbus accident result injury people separate injury injure transfer emergency department Suez General Hospital receive necessary treatment in Suez microbus c
```

```
[143] news.headline[recommended_articles[0]]
```

```
[45] import googletrans
```

```
[39] from googletrans import Translator
```

```
[36] # Set up the translator
    translator = Translator()
```

```
[40] translation = translator.translate(cell, src='ar', dest='en')
print(translation.text)
```

```
[40] ... The boy was playing ball in the field with his friends until midnight, then his mother called him and told him to come home.
```

Figure 49

File Edit Selection View Go Run Terminal Help tf-idf similarity.ipynb - Visual Studio Code

tf-idf similarity.ipynb X F: > Documents > NTI > Final Project > tf-idf similarity.ipynb > # Define the two texts to compare

+ Code + Markdown | ▶ Run All ⌘ Clear All Outputs | ⌥ Outline ... Select Kernel

```
[1] # Define the two texts to compare
text1 = news.body[22]
text2 = news.iloc[369]['body']
# Initialize a TfidfVectorizer
tfidf_vectorizer5 = TfidfVectorizer()

# Fit and transform the texts
tfidf_matrix5 = tfidf_vectorizer5.fit_transform([text1, text2])

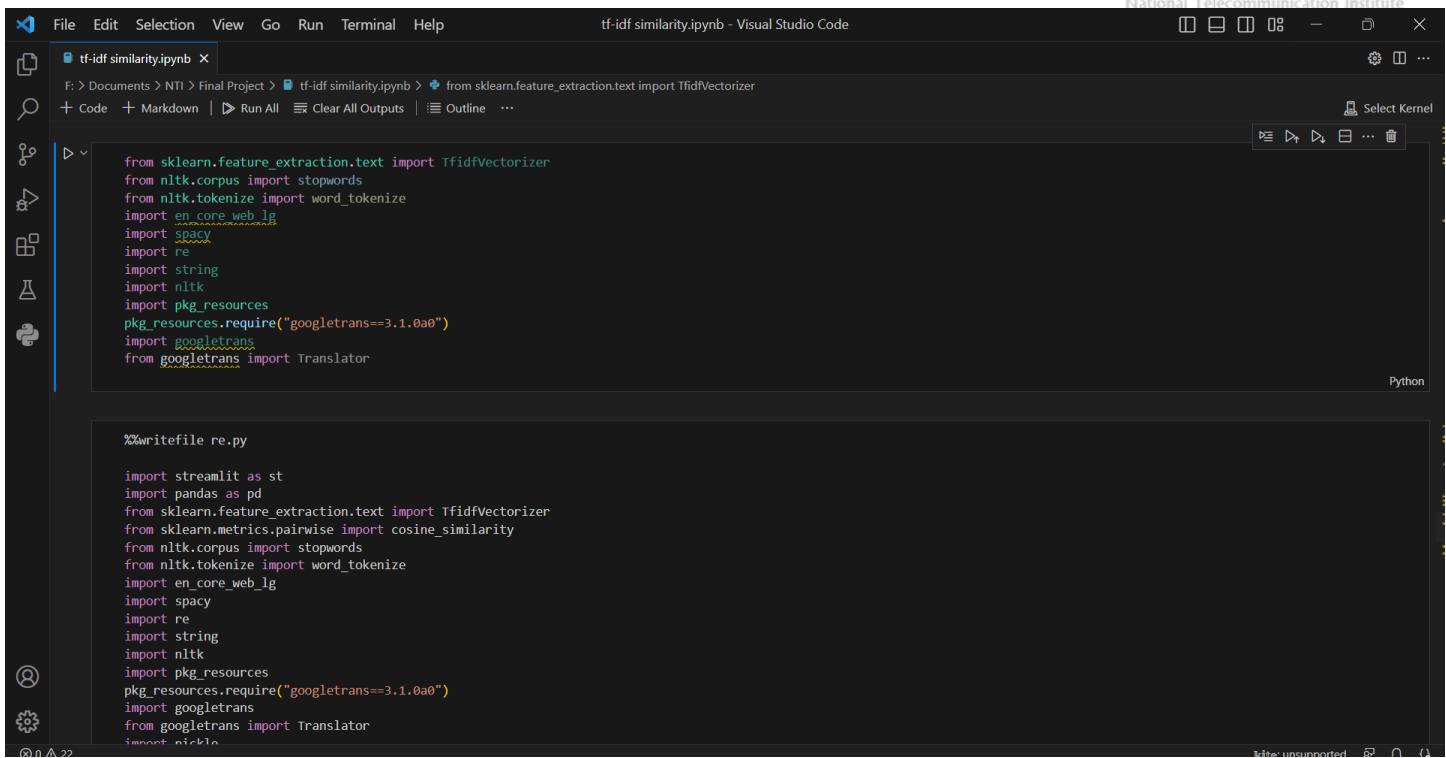
# Compute the cosine similarity between the two texts
cosine_similarities5 = cosine_similarity(tfidf_matrix5[0], tfidf_matrix5[1])

# Print the cosine similarity score
print("Cosine similarity score:", cosine_similarities5[0][0])
```

```
[2] 
```

```
[3] from sklearn.feature_extraction.text import TfidfVectorizer
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import en_core_web_lg
import spacy
import re
import string
import nltk
import pkg_resources
pkg_resources.require("googletrans==3.1.0a0")
```

Figure 50



```

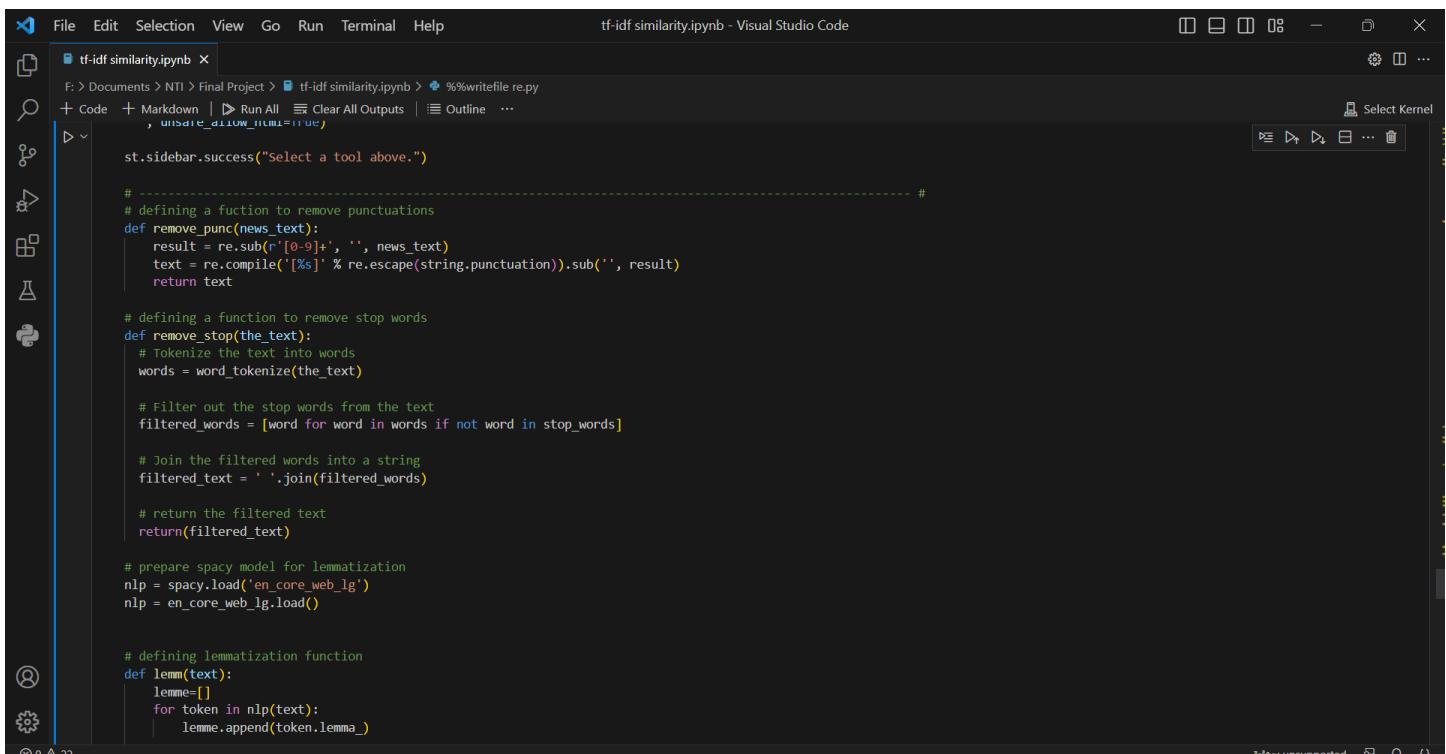
File Edit Selection View Go Run Terminal Help
tf-idf similarity.ipynb - Visual Studio Code
F: > Documents > NTI > Final Project > tf-idf similarity.ipynb > from sklearn.feature_extraction.text import TfidfVectorizer
+ Code + Markdown | ▶ Run All ⌘ Clear All Outputs | ⌥ Outline ...
Select Kernel
from sklearn.feature_extraction.text import TfidfVectorizer
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import en_core_web_lg
import spacy
import re
import string
import nltk
import pkg_resources
pkg_resources.require("googletrans==3.1.0a0")
import googletrans
from googletrans import Translator

%%writefile re.py

import streamlit as st
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import en_core_web_lg
import spacy
import re
import string
import nltk
import pkg_resources
pkg_resources.require("googletrans==3.1.0a0")
import googletrans
from googletrans import Translator
import pickle

```

Figure 51



```

File Edit Selection View Go Run Terminal Help
tf-idf similarity.ipynb - Visual Studio Code
F: > Documents > NTI > Final Project > tf-idf similarity.ipynb > %%writefile re.py
+ Code + Markdown | ▶ Run All ⌘ Clear All Outputs | ⌥ Outline ...
Select Kernel
st.sidebar.success("Select a tool above.")

# defining a function to remove punctuations
def remove_punc(news_text):
    result = re.sub(r'[0-9]+', '', news_text)
    text = re.compile('[%s]' % re.escape(string.punctuation)).sub('', result)
    return text

# defining a function to remove stop words
def remove_stop(the_text):
    # Tokenize the text into words
    words = word_tokenize(the_text)

    # Filter out the stop words from the text
    filtered_words = [word for word in words if not word in stop_words]

    # Join the filtered words into a string
    filtered_text = ' '.join(filtered_words)

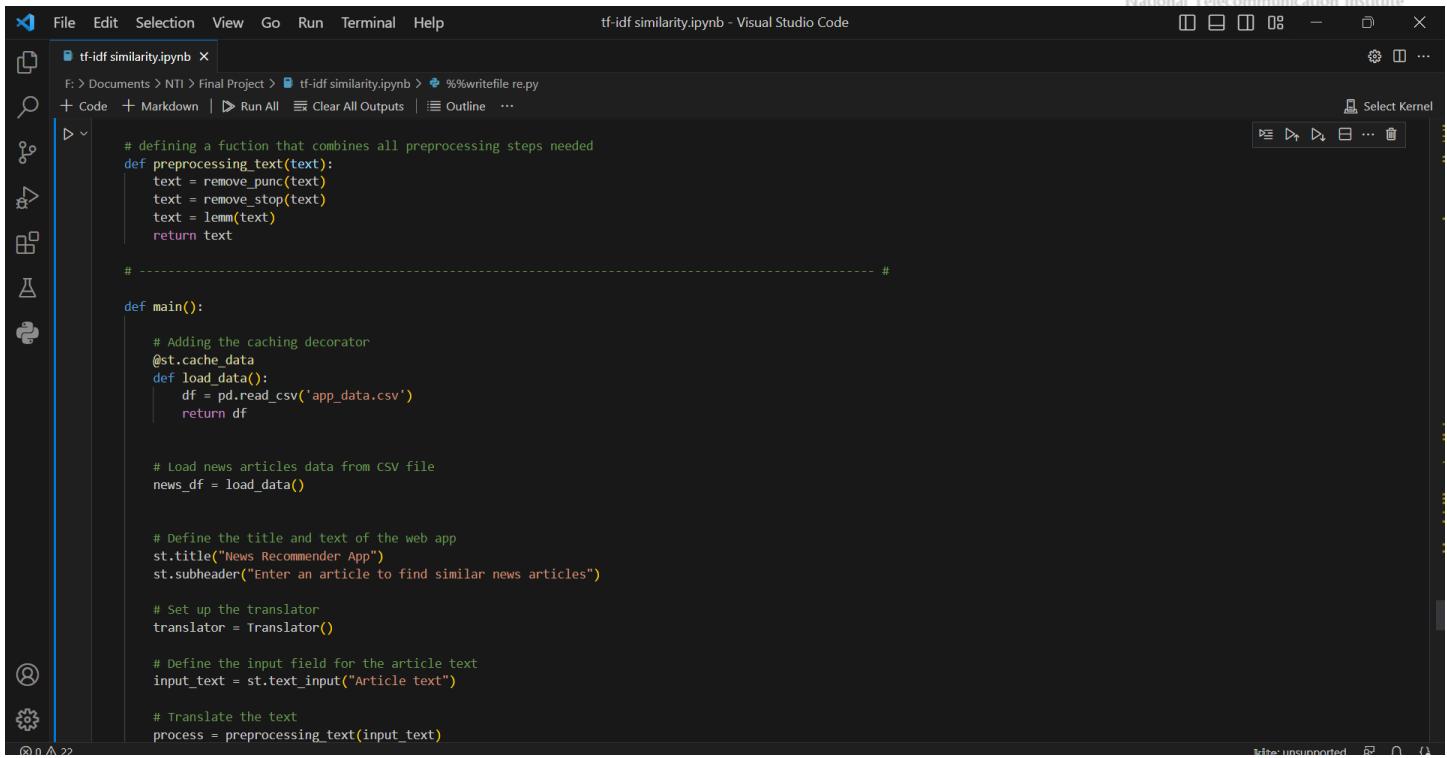
    # return the filtered text
    return(filtered_text)

# prepare spacy model for lemmatization
nlp = spacy.load('en_core_web_lg')
nlp = en_core_web_lg.load()

# defining lemmatization function
def lemm(text):
    lemm=[]
    for token in nlp(text):
        lemm.append(token.lemma_)


```

Figure 52



File Edit Selection View Go Run Terminal Help

tf-idf similarity.ipynb - Visual Studio Code

F: > Documents > NTI > Final Project > tf-idf similarity.ipynb > %%writefile re.py

+ Code + Markdown | Run All Clear All Outputs | Outline ...

```
# defining a function that combines all preprocessing steps needed
def preprocessing_text(text):
    text = remove_punc(text)
    text = remove_stop(text)
    text = lemm(text)
    return text

# ----- #

def main():

    # Adding the caching decorator
    @st.cache_data
    def load_data():
        df = pd.read_csv('app_data.csv')
        return df

    # Load news articles data from CSV file
    news_df = load_data()

    # Define the title and text of the web app
    st.title("News Recommender App")
    st.subheader("Enter an article to find similar news articles")

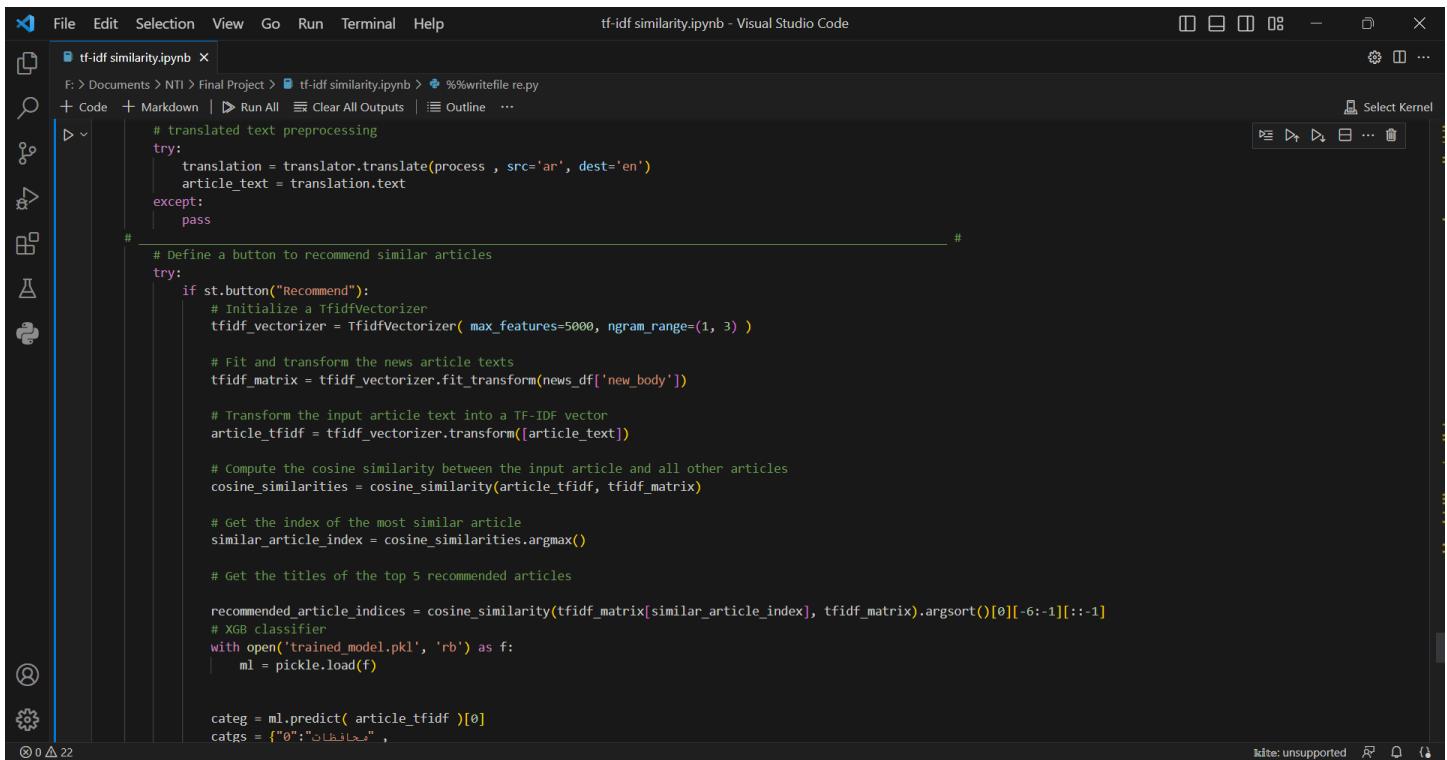
    # Set up the translator
    translator = Translator()

    # Define the input field for the article text
    input_text = st.text_input("Article text")

    # Translate the text
    process = preprocessing_text(input_text)
```

Kite: unsupported

Figure 53



File Edit Selection View Go Run Terminal Help

tf-idf similarity.ipynb - Visual Studio Code

F: > Documents > NTI > Final Project > tf-idf similarity.ipynb > %%writefile re.py

+ Code + Markdown | Run All Clear All Outputs | Outline ...

```
# translated text preprocessing
try:
    translation = translator.translate(process , src='ar', dest='en')
    article_text = translation.text
except:
    pass

# Define a button to recommend similar articles
try:
    if st.button("Recommend"):
        # Initialize a TfidfVectorizer
        tfidf_vectorizer = TfidfVectorizer( max_features=5000, ngram_range=(1, 3) )

        # Fit and transform the news article texts
        tfidf_matrix = tfidf_vectorizer.fit_transform(news_df['new_body'])

        # Transform the input article text into a TF-IDF vector
        article_tfidf = tfidf_vectorizer.transform([article_text])

        # Compute the cosine similarity between the input article and all other articles
        cosine_similarities = cosine_similarity(article_tfidf, tfidf_matrix)

        # Get the index of the most similar article
        similar_article_index = cosine_similarities.argmax()

        # Get the titles of the top 5 recommended articles
        recommended_article_indices = cosine_similarities[similar_article_index].argsort()[0][-6:-1][::-1]

        # XGB classifier
        with open('trained_model.pkl', 'rb') as f:
            ml = pickle.load(f)

        categ = ml.predict( article_tfidf )[0]
        catgs = {"0": "ا\u0628\u0627\u062f\u06cc\u0628\u06cc",
```

Kite: unsupported

Figure 54



The screenshot shows a Visual Studio Code window with the following details:

- Title Bar:** tf-idf similarity.ipynb - Visual Studio Code
- File Explorer:** Shows a file tree with 'tf-idf similarity.ipynb' as the active file.
- Code Editor:** Displays Python code for news classification using TF-IDF and XGBoost. The code includes:
 - Importing necessary libraries: `re` and `pickle` from `open('trained_model.pkl', 'rb')`.
 - Calculating cosine similarity between the input article and a matrix of similar articles.
 - Predicting the category of the input article based on the predicted TF-IDF values.
 - Defining a dictionary `catgs` mapping category numbers (0-6) to Arabic labels: 0 ("محافطة"), 1 ("الإسكندرية"), 2 ("الإسكندرية و محافظة"), 3 ("أخبار"), 4 ("ثقافة وفنون"), 5 ("جودة"), 6 ("اقتصاد").
 - Printing the recommended articles.
 - A loop to write news details to a file, including headline, brief, go-to URL, and hrefs.
 - Handling exceptions and prompting for news text if an error occurs.
 - A conditional block to run the main function if the script is executed directly.
- Terminal:** Not visible in the screenshot.
- Status Bar:** Shows standard status icons.

Figure 55

2.6. Summarization:

Figure 56

```
[ ] from transformers import AutoTokenizer, AutoModelWithLMHead
import torch
import nltk
nltk.download('punkt')
from nltk.tokenize import sent_tokenize

[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]  Unzipping tokenizers/punkt.zip.

[ ] device = torch.device("cuda:0" if torch.cuda.is_available() else "cpu")
m_name = "marcfa-nlp/summarization-arabic-english-news"

[ ] tokenizer = AutoTokenizer.from_pretrained(m_name)
model = AutoModelWithLMHead.from_pretrained(m_name).to(device)

Downloading (...)okenizer_config.json: 100% [██████████] 432/432 [00:00<00:00, 22.0kB/s]
Downloading spiece model: 100% [██████████] 847k/847k [00:00<00:00, 2.78MB/s]
Downloading (...)main/tokenizer.json: 100% [██████████] 1.40M/1.40M [00:00<00:00, 5.17MB/s]
Downloading (...)cial_tokens_map.json: 100% [██████████] 65.0/65.0 [00:00<00:00, 3.94kB/s]
/usr/local/lib/python3.10/dist-packages/transformers/models/auto/modeling_auto.py:1322: FutureWarning: The class `AutoModelWithLMHead` is deprecated and will be removed in a future version. Pl
warnings.warn(
Downloading (...)ve/main/config.json: 100% [██████████] 765/765 [00:00<00:00, 35.9kB/s]
Downloading pytorch_model.bin: 100% [██████████] 3.16G/3.16G [00:53<00:00, 61.2MB/s]
```

Figure 57

```
[ ] def get_summary(text, tokenizer, model, device="cpu", num_beams=2):
    if len(text.strip()) < 50:
        return ["Please provide more longer text"]

    text = "summarize: <paragraph> " + " <paragraph> ".join([ s.strip() for s in sent_tokenize(text) if s.strip() != "" ]) + " </s>"
    text = text.strip().replace("\n", "")

    tokenized_text = tokenizer.encode(text, return_tensors="pt").to(device)

    summary_ids = model.generate(
        tokenized_text,
        max_length=512,
        num_beams=num_beams,
        repetition_penalty=1.5,
        length_penalty=1.0,
        early_stopping=True
    )

    output = tokenizer.decode(summary_ids[0], skip_special_tokens=True)
    return [ s.strip() for s in output.split("<hl>") if s.strip() != "" ]
```

Figure 58

```
[ ] ## Get summaries if there are a lot of lines in string

print("Original Article:", text)
print("\n=====\\nSummary: \\n")
hls = get_summary(text, tokenizer, model, device)
for hl in hls:
    print("\\t-", hl)
```

قال محيط في مقالة مع مي إن بي بي عرب، إن مصر انطلاق مدنات لى منكوك دولية حتى نهاية يونيو المقبل، وهي حالة الخفاف اللائدة عالمياً من الممكن أن تطرح مدنات أو منكوك دولية.

Original Article:

اـكـدـ انـ مـصـرـ مـنـ الـلـوـلـ الـقـلـيلـ الـتـيـ خـفـتـ مـدـلـاتـ نـمـوـ بـلـتـ 3.6% حـالـ آرـمـةـ كـوـروـنـاـ، مـتـبـرـ إـلـيـ أـنـ نـسـيـنـ إـلـىـ النـاقـحـ الـمـلـحـيـ مـنـ الـمـقـرـضـنـ أـنـ تـكـونـ 78% فـالـإـقـتصـادـ الـمـصـرـيـ تـحـلـ الصـمـدـاتـ وـاسـتـمـرـ فـيـ تـحـقـيقـ مـدـلـاتـ نـمـوـ رـسـمـ الـأـزـمـاتـ - عـلـيـ حدـ تـحـقـيقـهـ.

واـكـدـ مـعـيـطـ اـنـ اـرـقـاعـ اـسـمـارـ الـخـاءـ عـالـمـاـ زـادـ كـلـفـةـ اـمـتـرـادـ الـلـلـغـ منـ الـخـارـجـ، مـتـبـرـ إـلـيـ أـنـ الـإـقـتصـادـ الـمـصـرـيـ أـصـبحـ تـحـضـيـتـ بـسـبـبـ تـعـذـيـتـ الـحـربـ الـرـوـسـيـةـ الـأـيـرـكـيـةـ ، مـوـضـيـاـ أـنـ 80% مـنـ وـارـدـاتـ مـصـرـ مـنـ الصـفـحـ ظـاهـيـ مـنـ جـوـاتـ روـسـاـ وأـوـكـراـنـاـ.

=====

Summary:

وـقـدـ قـالـ وزـيرـ الـإـقـتصـادـ الـمـصـرـيـ أـنـ مصرـ انـ تـطـرـحـ مـدـلـاتـ أوـ منـكـوكـ دـولـيـةـ حتـىـ نـهاـيـةـ يونيوـ الـمـقـلـلـ -

Figure 59

CONCLUSION

In conclusion, the News Recommendation System project has demonstrated the feasibility and effectiveness of using machine learning techniques to recommend personalized news articles to users. By analyzing user behavior patterns and preferences, we were able to create a system that delivers relevant news articles in real-time, improving the user experience and increasing engagement. The project also highlighted the importance of incorporating ethical considerations into the development of such systems. We ensured that our system does not perpetuate filter bubbles or bias, and that user data is kept secure and private. Moving forward, we recommend further exploration of potential applications and improvements to the system, as discussed in the previous section. We also encourage the adoption of similar ethical considerations in the development of news recommendation systems. Overall, the News Recommendation System project has provided valuable insights into the possibilities of using machine learning to personalize the news and improve the way we stay informed. We hope that our work can inspire further innovation in this field and contribute to a more informed and diverse society.

REFERENCES

- 1- What is a recommendation system? NVIDIA Data Science Glossary. Available at: <https://www.nvidia.com/en-us/glossary/data-science/recommendation-system/> (Accessed: 09 May 2023).
- 2- *What is scraping: About price & web scraping tools: Imperva* (2019) *Learning Center*. Available at: <https://www.imperva.com/learn/application-security/web-scraping-attack/> (Accessed: 10 May 2023).
- 3- *Text classification: What it is and why it matters* (no date) *MonkeyLearn*. Available at: <https://monkeylearn.com/text-classification/> (Accessed: 09 May 2023).
- 4- *What is summarization? - hugging face* (no date) *What is Summarization? - Hugging Face*. Available at: <https://huggingface.co/tasks/summarization> (Accessed: 09 May 2023).
- 5- *Streamlit docs* (no date) *Streamlit documentation*. Available at: <https://docs.streamlit.io/> (Accessed: 09 May 2023).
- 6- <https://www.edlitera.com/en/blog/posts/text-summarization-nlp-how-to>