GEMA Group

Artificial Intelligence School

Master's degree in Artificial Intelligence and Management

Artificial Intelligence project on Fake News/Deep Fake

Option : Data Analyst Career Path

Detection of Fake News on USA Political Data : Development of an AI Web Application for Fake News Detection

Realized by :

Elimane Yassine SEIDOU

Hawa SOGOBA

Sylvestre KACOU

Academic Year 2023-2024

Table of contents

I. ABSTRACT

This project aims to develop an Artificial Intelligence model capable of classifying information related to politics in the USA as either Fake News or genuine news. Initially, the model will be implemented in a web application available only locally. It is based on political information from the USA between 2015 and 2018, collected from sources such as Kaggle, GitHub, and Coursera. The study was conducted on a population of 44,898 articles published across various news channels.

After a descriptive analysis of the considered variables, natural language processing was performed on the dataset, and a deep learning model was developed using Python 3.9. This model was then integrated into an application using the Flask framework along with HTML, CSS, and JavaScript languages. The application's purpose is to receive a text of at least 40 words, classify the information, and store the classified information in a CSV file. This work was carried out using software such as Excel and Visual Studio Code.

From our analyses, the articles in our dataset totaled 9,276,947 words, with 108,704 unique words after removing stop words, with a maximum length of 4405 and a minimum length of 40. We utilized an LSTM model for Fake News detection and achieved a 99% accuracy rate in detection.

II. INTRODUCTION

Fake News detection is currently garnering increasing interest from the public and researchers due to the growing circulation of false information online, particularly on social media platforms, news blogs, and online press sites. For instance, a recent report from the technology blog Jump-shot revealed that references to Facebook accounted for 50% of total traffic to fake news sites and 20% of traffic to trustworthy websites. Given that most American adults (62%) get their information through social media (Jeffrey & Elisa, 2016), merely being labeled as an "information site" is no longer sufficient.

It is therefore crucial to implement systems capable of identifying falsified content in online sources. Until now, computer-based approaches to Fake News detection have relied on sources such as satirical news like "The Onion" and fact-checking websites like "PolitiFact" and "Snopes." However, with the revolution of artificial intelligence techniques, especially in NLP and Deep Learning, we are now able to develop powerful algorithms that can shape our perception of the information we read online. The post-Covid revolution of accessible AI has made many AI practitioners realize the necessity of providing user-friendly tools developed via chips, websites, mobile or desktop applications, and more.

The aim of this work is to contribute to this effort by providing a Fake News prediction model and adhering to the new trend of providing users with easily accessible and understandable tools through analysis, data processing, exploration of the model, and the web development tools used. Specifically, this involves exploring the domain of rumor analysis, providing an overview of the problem, presenting various techniques and approaches along with their results. The project aims to automatically analyze, based on a form filled out by the user, the information shared on social media to detect Fake News and distinguish them from genuine news. We will discuss the different approaches used to integrate sentiment into the Fake News detection process. After providing a description of related work, we will define Fake News and discuss the implications of their dissemination today. We will then focus on the Fake News detection process. Finally, we will outline the methods used to provide access to our model for individuals not in the AI domain, particularly through the locally developed web application.

III. DATA

The dataset used primarily comes from Kaggle and is also available on GitHub and Coursera. Among the diverse data available on these platforms, we focused on political data due to the quantity of data available to conduct our project. The more data we have, the better. Therefore, the chosen data pertains to political events that occurred in the United States between 2015 and 2018. We primarily have two datasets: a True dataset containing information previously recognized as true, and a False dataset containing information previously recognized as false. Each of these datasets is contained in a CSV file. Thus, we have a population of 44,898 pieces of information in total, with 21,417 pieces of information judged as true and 23,481 judged as false. In the following sections of this study, we will present a description of the variables and descriptive analyses of these variables.

i. VARIABLES DESCRIPTION

The two datasets contain those variables:

- Title: This variable contains the title of the articles.
- Text: This variable contains the content of the article.
- Subject: This variable contains the subject or category of information to which the article belongs
- Date: This refers to the publication date of the article.

Before starting any descriptive analysis on the datasets, we created some new variables.

- isfake: The variable isfake takes the value 0 when the information is true and the value 1 when the information is false.
- Original: This variable contains the concatenate of the title variable and the text variable
- Clean: This variable contains the list of word for each element of original after having the stop words removed
- Cleaned_join: This variable is the conversion of clean from a list to a string.

The new dataset created to store prediction made with the AI web application contains the following variables:

- User ID: This variable contains a random id created for each new verification made.
- Date/Hour: This variable contains the date and hour of the submission of the verifications.
- Verified Text: This variable contains the article to verify.
- Result: This variable contains the results of the verification sent by the model developed

ii. DATA PREPROCESSING

As we're working with two datasets, we decided to make some preprocess of those datasets before performing descriptive analysis, visualizations and to build our model before implementing it in the web app.

1. Creation of the variable isfake

As the initial datasets are not directly labeled, we created the variable isfake to attribute a label to the information. The variable isfake take the value 0 when the information is true and the value 1 when the information is false.

2. Concatenation of the two datasets

As we're working with two datasets, after the attribution of the label isfake to each dataset, we decided to concatenate them to make the study easier.

3. Removal of the date variable

As our goal is not to work on time series, we decided to remove the date variable.

4. Creation of the variable Original

We need to maximize text information that we're going to process. In this purpose, we decided to concatenate the title variable and the text variable in a brand-new variable called Original.

5. Natural Language Process (NLP)

a) Stop words Removal.

In the NLP process we need to delete the stop words, the word having no value for the analysis. To succeed in this step, we first downloaded a list of stop words using NLTK's stop words list that we extended. We finally got a total length of 184 stop words to remove from the variable original. Then we defined a preprocessing function to remove stop words, non-alphanumeric characters from all the element of the variable original and converting text to lowercase. We finally stored the result of this function in a new variable called “clean”. This variable contains the list of word for each element of original after having the stop words removed. Then we created a new variable to convert the clean variable elements to string. This new variable name is cleaned_joined.

b) Tokenization and Padding

Tokenization is the process of breaking down text into smaller units called tokens, which can be words, phrases, or symbols. Padding is the process of adding zeros or a special token to sequences to make them uniform in length. In NLP, padding is necessary because many machine learning algorithms, especially neural networks, require fixed-size input.

After splitting the concatenated dataset into train and test datasets, we created a tokenizer to tokenize the words and create sequences of tokenized words and added a padding function commonly used in NLP to ensure that all sequences (in this case, sequences of words or tokens) have the same length.

c) Lemmatization and Stemming

In our code, we have integrated both lemmatization and stemming as part of the preprocessing steps. Lemmatization is performed using the WordNetLemmatizer from NLTK, while stemming is done using the PorterStemmer. These techniques help in reducing the dimensionality of the text data and capturing the essence of words in a more compact form, which is beneficial for machine learning models.

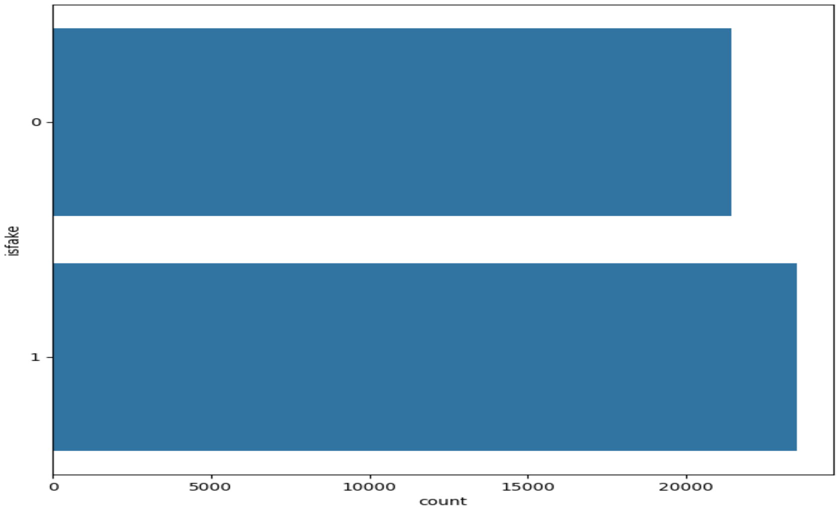
Lemmatization is the process of reducing words to their base or dictionary form, known as a lemma. For example, "running" becomes "run", "better" becomes "good", and so on. Lemmatization usually involves dictionary lookup and morphological analysis to determine the lemma of a word.

Stemming, on the other hand, is a more heuristic process that chops off suffixes from words to reduce them to their stems. For example, "running" might be stemmed to "run", "better" to "better", and so on. Stemming is less precise than lemmatization but can still be effective in many cases.

iii. DATA VISUALIZATION AND DESCRIPTIVE ANALYSIS

1. Count of fake news

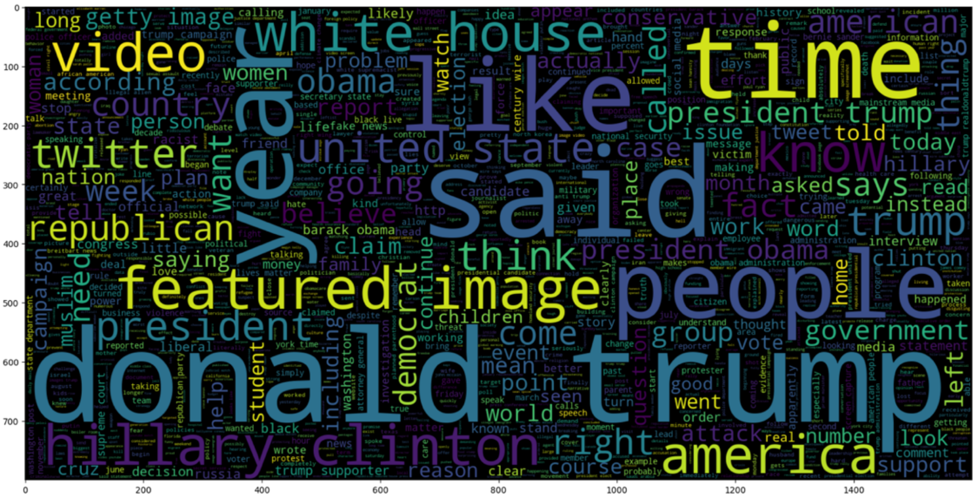
This graph shows the number of fake news and the number of information not classified as fake news.



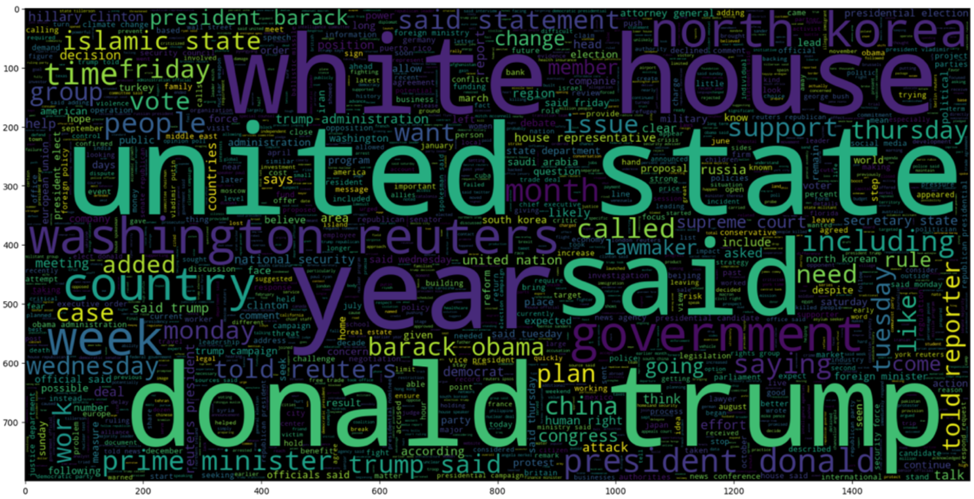
Img 1: Number of fake news and true news

2. Word cloud

A word cloud is a visual representation of text data where the size of each word indicates its frequency or importance within the given text. In a word cloud, words are typically displayed in different sizes and colors, with more frequent or important words appearing larger and more prominent. Here're the word clouds for fake and true news.



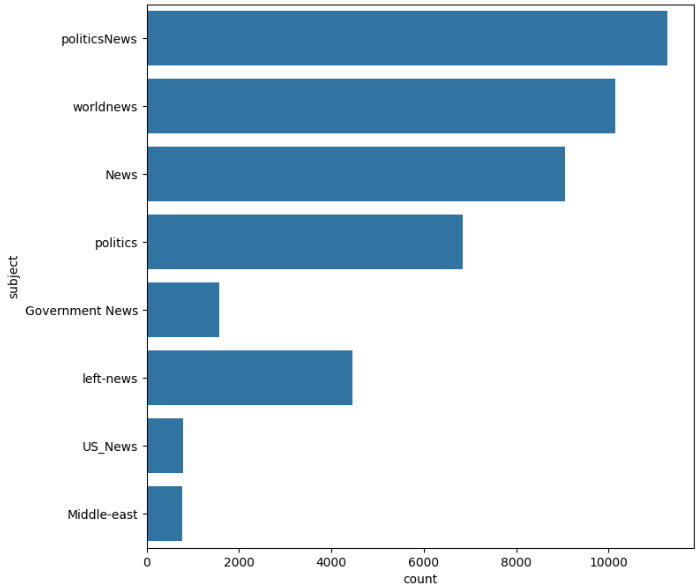
Img 2: Fake News Word Cloud



Img 3: True News Word Cloud

3. Fake News Repartition Per Subjects

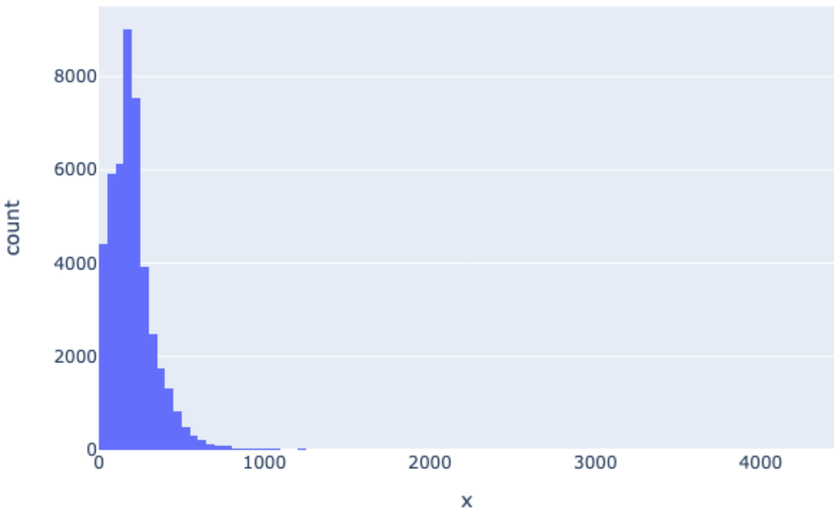
This graph shows the number of fake news per subjects.



Img 4: Number of fake news

4. Distribution of number of words in a text

In this visualization we can see the number of words in each text to verify. We have a maximum length of 4405 and an average length of 450 across the data set.



Img 5: Distribution of number of words in a text

IV. PROBLEMATIC

The spread of false information, or fake news, has become a ubiquitous phenomenon in our digital age. These inaccurate pieces of information can have a significant impact on public perception, influence political and social decisions, and even cause tensions and conflicts. In this context, verifying the truthfulness of information has become a crucial priority to ensure the integrity and reliability of information.

In the political domain specifically, fake news can be particularly damaging. They can distort voters' understanding, influence election outcomes, and undermine trust in democratic institutions. Therefore, it is imperative to develop effective tools and methods to detect and counter the spread of fake news, especially in the political context.

Currently, there is a lack of easily accessible and reliable tools to verify the truthfulness of information online. This gap in the field of fake news verification is what motivates our project. We aim to develop an artificial intelligence application specialized in detecting fake news in the political domain of the United States.

The key question guiding our problematic is as follows: How can we develop an effective artificial intelligence tool to detect and verify the truthfulness of political information, to counter the spread of fake news and promote more reliable and transparent information for the public?

i. WHAT IS FAKE NEWS?

Fake news refers to false or misleading information presented as news. This type of information is often intentionally created to deceive or manipulate readers, viewers, or listeners by spreading inaccurate or fabricated content. Fake news can be disseminated through various media channels, including social media, traditional news outlets, websites, and blogs. It is designed to appear credible and trustworthy, often mimicking legitimate news sources in terms of format and style. The purpose of fake news can range from influencing public opinion to causing confusion, controversy, or even harm.

False and distorted news material isn't exactly a new thing. It's been a part of media history long before social media, since the invention of the printing press. It's what sells tabloids. On the internet, headline forms called *clickbait* entice people to click to read more, by trying to shock and amaze us. What's more outrageous to read about than fake things that didn't happen?

fake news is also used as a term to discredit news stories that individuals (particularly former president Donald Trump) don't like, to suggest that they were made up or that they blow out of proportion something that should be trivial (even if other sources can verify their accuracy). In a conversation with Fox Business in October 2017, Donald Trump claimed that he had "really started this whole 'fake news' thing." (Ironically, Hillary Clinton used the term in a speech two days before Trump's first use of the phrase.

ii. WHAT IS POLITICS?

Political history is the narrative and survey of political events, ideas, movements, organs of government, voters, parties and leaders. It is closely related to other fields of history, including diplomatic history, constitutional history, social history, people's history, and public history. Political history studies the organization and operation of power in large societies.

From approximately the 1960s onwards, the rise of competing subdisciplines, particularly social history and cultural history, led to a decline in the prominence of "traditional" political history, which tended to focus on the activities of political elites. In the two decades from 1975 to 1995, the proportion of professors of history in American universities identifying with social history rose from 31% to 41%, and the proportion of political historians fell from 40% to 30%.

iii. WHY IS DETECTING FAKE NEWS IN POLITICS IMPORTANT?

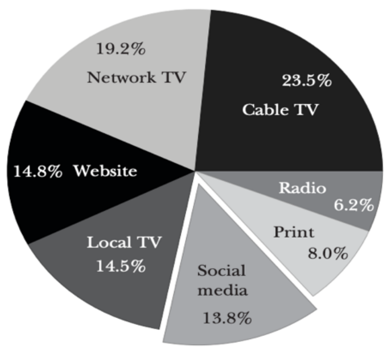
Detecting fake news in politics is crucial for several reasons. First and foremost, it helps safeguard the integrity of democratic processes by ensuring that voters have access to accurate and reliable information. When people make informed decisions based on facts rather than falsehoods, democracy functions more effectively.

Additionally, detecting fake news helps combat misinformation and disinformation campaigns that seek to manipulate public opinion or sow discord. By identifying and debunking false narratives, journalists, fact-checkers, and responsible media organizations play a vital role in maintaining a healthy information ecosystem.

Moreover, detecting fake news promotes transparency and accountability in political communication. When politicians and public figures spread false information, they should be held accountable, and accurate information should be readily available to the public. This encourages honesty and integrity in political discourse.

Overall, detecting fake news in politics contributes to a more informed electorate, strengthens democratic institutions, and fosters a healthier political environment based on truth and accuracy.

Here is a picture illustrating the main fake news sources during 2016 US elections:



Img 6: Fake News sources during 2016 US elections

Source: social media and Fake News in the 2016 Election

iv. ETHICAL CONCERN ABOUT FAKE NEWS IN POLITIC

Ethical concerns surrounding fake news in politics are paramount and cannot be overlooked. One of the primary ethical issues is the potential manipulation of public opinion and democratic processes. When false or misleading information is deliberately spread to influence political outcomes, it undermines the principles of fairness, transparency, and informed decision-making in a democratic society.

Another ethical concern is the impact of fake news on social cohesion and trust in institutions. When individuals are exposed to misinformation that promotes division, hatred, or prejudice, it can contribute to societal unrest, discrimination, and polarization. This erodes the ethical foundation of promoting respect for diverse perspectives and fostering unity within communities.

Furthermore, the spread of fake news in politics raises questions about media responsibility and journalistic integrity. Journalists and media organizations have an ethical obligation to verify information, uphold accuracy, and provide context to news stories. When fake news is disseminated without proper verification or fact-checking, it can erode public trust in the media and blur the line between credible journalism and sensationalism.

Moreover, fake news can perpetuate harmful stereotypes, stigmatize marginalized groups, and distort public discourse on important social issues. This can have lasting consequences on the rights and dignity of individuals and communities, highlighting the ethical imperative of promoting fair, balanced, and respectful reporting in political coverage.

In essence, addressing the ethical concerns of fake news in politics requires a multi-faceted approach that emphasizes media literacy, responsible reporting practices, regulatory oversight, and civic engagement. Upholding ethical standards in political communication is

essential for preserving the integrity of democratic societies and promoting a healthy public discourse based on truth, fairness, and respect.

For example, **During the 2020 US presidential election, a fabricated news story claimed that Dominion Voting Systems, a company that supplies voting machines, was deleting votes for Donald Trump.** This story was widely shared on social media and even promoted by some prominent figures. There was no evidence to support this claim, and investigations by election officials and fact-checkers found it to be false. However, the story likely contributed to a decline in trust in the legitimacy of the election results among some voters.

v. HOW CAN AI HELP TO SOLVE THE PROBLEM OF FAKE NEWS SPREAD ON INTERNET?

Fake news is spread extremely quickly on due to the evolution in the number of subscribers of on digital solutions such as social medias, blog. Various AI techniques exists to help solving the spread of fake news on internet. Here are some techniques that we identified:

- **Automated Fact-Checking:** AI-powered algorithms can quickly analyze large volumes of information to verify the accuracy of news articles, images, and videos. By comparing information against reliable sources and fact-checking databases, AI can flag potential fake news content.

- **Natural Language Processing (NLP):** NLP techniques enable AI systems to understand and interpret human language. This capability is leveraged to detect linguistic patterns indicative of fake news, such as sensationalist language, misleading headlines, or inconsistencies in reporting.

- **Content Analysis:** AI algorithms can analyze the content of news articles, social media posts, and online forums to identify misinformation. By examining the context, sentiment, and credibility of sources, AI can assess the trustworthiness of information.

- **Social Media Monitoring:** AI tools can monitor social media platforms for the rapid dissemination of fake news. By tracking user behavior, engagement patterns, and the virality of content, AI can detect and flag suspicious or misleading information before it spreads widely.

- **Deep Learning Models:** Advanced AI models, such as deep learning neural networks, can be trained on vast datasets of known fake news examples. These models learn to recognize subtle patterns and characteristics of fake news, improving their accuracy in identifying deceptive content.

- **User Empowerment:** AI can empower users to critically evaluate information by providing browser extensions or mobile apps that highlight potential fake news sources, fact-check claims in real-time, and offer alternative perspectives.

- **Collaborative Filtering:** AI algorithms can analyze user preferences and behaviors to recommend reliable news sources and filter out potentially misleading or biased content, reducing exposure to fake news.

Our project aims to develop an NLP and Deep Learning based AI application able to detect whether an information is fake or not.

vi. METHODOLOGY OF DEVELOPMENT OF THE AI WEB APPLICATION FOR FAKE NEWS DETECTION

1. Objectives, Targets and User story

a) Objectives

The primary objective of developing this AI web application is to provide a reliable and accessible tool for users to verify the authenticity of political facts and news articles. The specific goals include:

- **Fake News Detection:** Develop an AI model capable of accurately detecting fake news and misinformation related to political facts.
- **User-Friendly Interface:** Create a user-friendly web interface that allows users to easily submit articles for verification without the need for login credentials or fees.
- **Anonymous Verification:** Ensure that the verification process is anonymous, respecting users' privacy and allowing them to verify information without disclosing personal information.

- **Real-Time Feedback:** Provide users with real-time feedback on the classification of articles as either genuine or fake, based on the analysis conducted by the AI model.
- **Reliability and Accuracy:** Ensure that the AI model used in the application is reliable, accurate, and capable of handling a diverse range of political topics and sources.
- **Educational Value:** Offer educational resources and explanations to help users understand the criteria used for fake news detection and enhance media literacy.

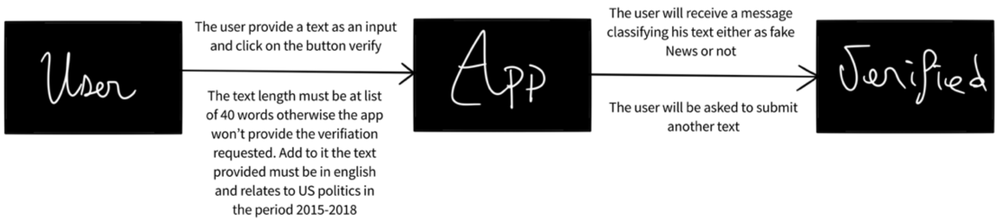
b) Targets

The target audience for this AI web application includes any user who want to verify an information:

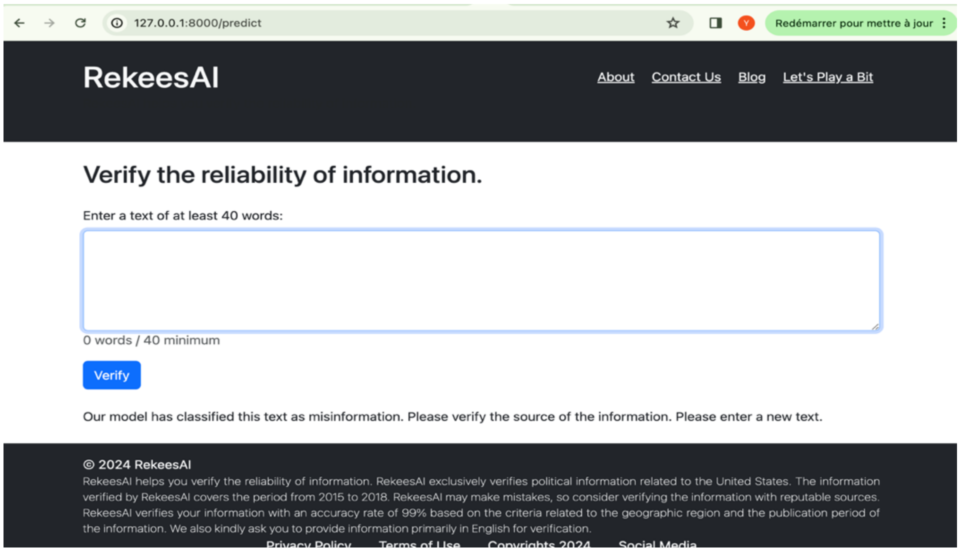
- **General Public:** Any individual who wants to verify the authenticity of political news and information they encounter online, without the need for specialized knowledge or technical skills.
- **Journalists and Researchers:** Professionals in the field of journalism and research who require a tool to fact-check political articles and sources quickly and efficiently.
- **Students and Educators:** Students studying media literacy, political science, or related fields, as well as educators looking for tools to teach critical thinking and information verification skills.
- **Fact-Checkers and Media Analysts:** Individuals and organizations dedicated to fact-checking and analyzing political news and statements for accuracy and reliability.
- **Civil Society Organizations:** NGOs, advocacy groups, and civil society organizations interested in promoting transparency, accountability, and truthfulness in political discourse and media reporting.
- **Government and Policy Makers:** Officials and policymakers who seek reliable tools to evaluate the credibility of political information and combat misinformation campaigns.

c) User Story, Application overview and data flow

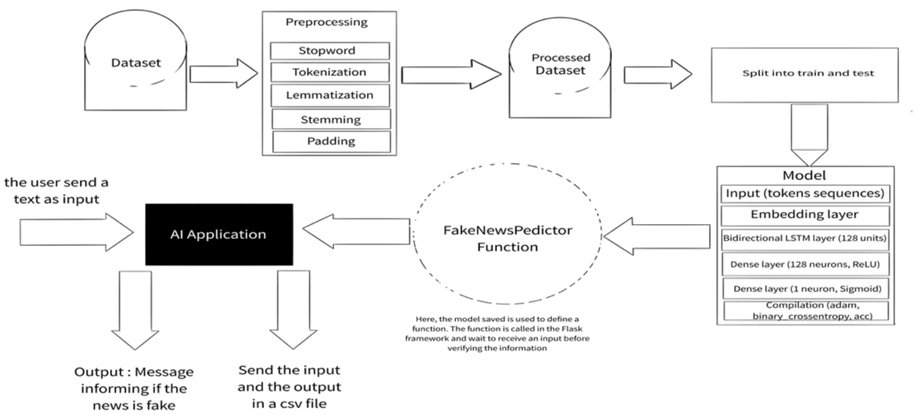
The following pictures give you an overview of the way a user can use the application and the way the data flow and the structure of the application.



Img 7: User Story



Img 8: Application overview



Img 9: Data flow

2. Literature review and Preliminary Research (State of the Art)

Various studies and project had already been done in this field from research studies on Elsevier to GitHub project. Here're some case already published on

- **Influence of fake news in Twitter during the 2016 US presidential election:** The study examines fake news on Twitter during the 2016 US election using a dataset of 171 million tweets. It identifies 30 million tweets with news links, finding that 25% spread fake or biased news. The analysis focuses on how fake news influenced the election, categorizing websites as spreading misinformation or factual news based on judgment and opinion. Traditional news outlets are classified by political orientation, showing consistent patterns despite some subjectivity in classification.

- **Fake news detection on social networks using Machine learning techniques**: The constant spread of fake news has a significant negative impact on society. Users globally frequent popular social media platforms like Facebook, Twitter, Instagram, LinkedIn, etc. These platforms often have weak authentication systems, relying on basic user information like name, photo, and location, making it easy for malicious users to exploit and spread false information. To address this issue, machine learning techniques were applied to detect fake news in English using models like Count Vectorizer, TF-IDF Vectorizer, and N-gram, alongside machine learning algorithms such as Naive Bayes, SVM, Random Forest, and Logistic Regression. The proposed approach achieved maximum accuracy using TF-IDF highlights and an SVM classifier, with a precision level of 93%.

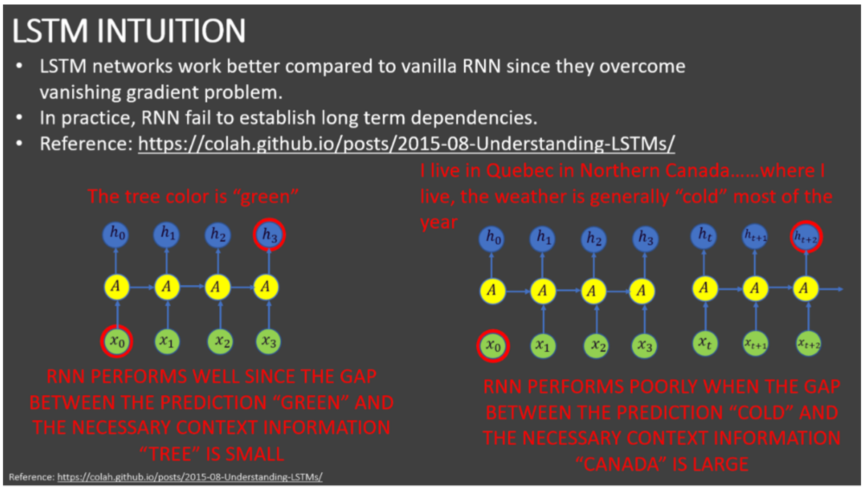
- **Detection of fake news using deep learning CNN–RNN based methods**: This study focuses on detecting fake news, which is deliberately spread misinformation with harmful effects on politics and society. It employs deep learning techniques like CNN, Bidirectional LSTM, and ResNet, along with pre-trained word embedding, across four datasets. Data augmentation via back-translation helps balance class imbalances. Bidirectional LSTM proved most effective among the architectures tested, surpassing CNN and ResNet across all datasets.

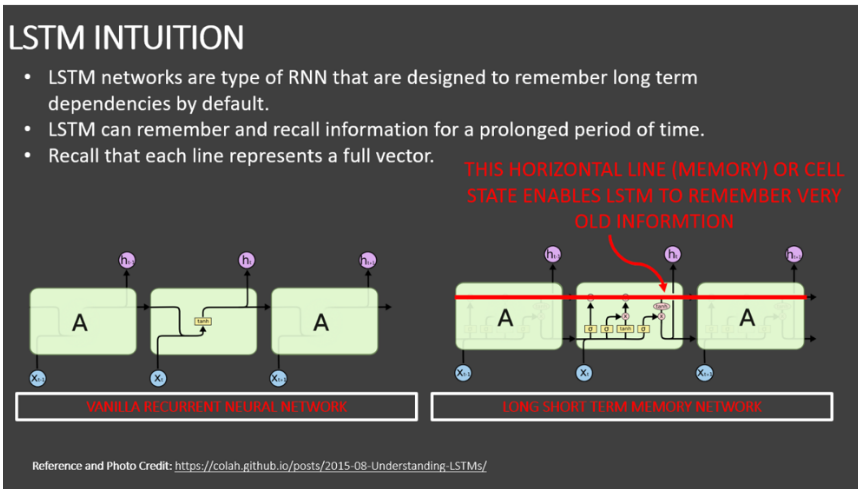
- **D-coder Fake News Detection Web App on GitHub**: The primary objective of this project is to identify the authenticity of the news and categorize it into fake and real news. In this project, Machine Learning model PassiveAggressiveClassifier is leveraged to distinguish real and fake news and it has achieved an accuracy of 93.13%. Thereafter, fake news detection web app is created using machine learning, python, html, CSS and flask and this project is deployed using app service in Microsoft azure.

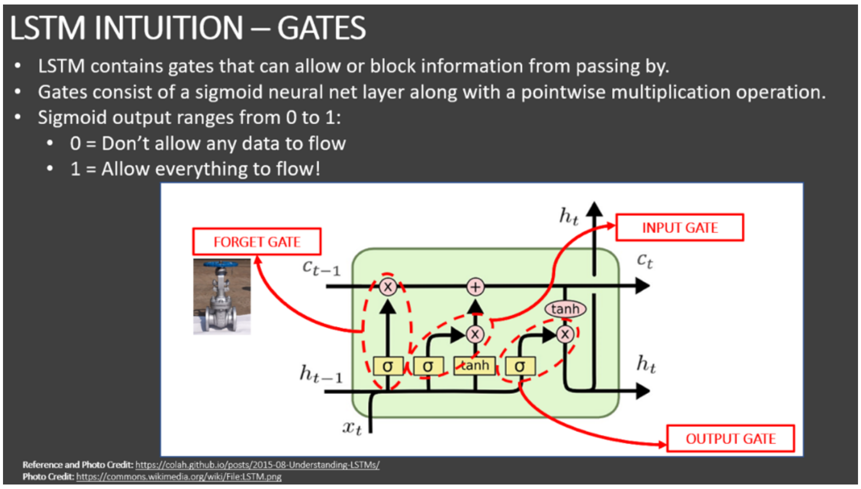
- Other Projects such as Ryan Ahmed, PhD Fake news detection tutorial on Coursera, Nsk1512 Covid19 fake news detection on GitHub, DJDarkCyber Fake news Detector also on GitHub provide relevant methods to detect fake news.

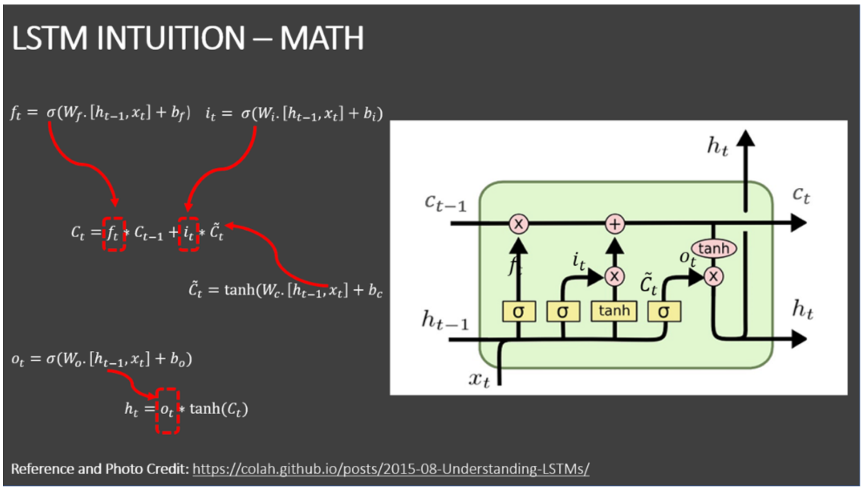
3. Model Architecture Selection

During the model selection we tried the SVM model, the KNN model, the logistic regression and the LSTM model. The model we used to develop the web application is the LSTM model. Let's have a look to the LSTM architecture.









4. Model Development

```
Input (Séquences de tokens)
           |
     Embedding Layer
    (total_words, 128)
           |
Bidirectional LSTM Layer (128 units)
           |
       Dense Layer
  (128 neurones, ReLU)
           |
       Dense Layer
   (1 neurone, Sigmoid)
           |
   Output (Prédiction)
           |
Compilation (Adam, Binary Crossentropy, Accuracy)
```

This diagram represents the different layers of our model sequentially:

- Input (Token Sequences): This is the input to our model, consisting of sequences of tokens representing our textual data.
- Embedding Layer: This layer transforms the token sequences into dense embedding vectors. Each token is represented by a vector of length 128 in this example.
- Bidirectional LSTM Layer (128 units): This is a bidirectional LSTM layer that model's sequences in both forward and backward directions. It consists of 128 LSTM units.
- Dense Layer (128 neurons, ReLU): This is a fully connected layer with 128 neurons and a ReLU (Rectified Linear Unit) activation function. It introduces non-linearity into the model.
- Dense Layer (1 neuron, Sigmoid): This is the output layer of our model, with a single neuron and a sigmoid activation function. This layer produces the final prediction, which is a probability between 0 and 1, indicating the likelihood that the input is a fake news.
- Output (Prediction): This is the final output of our model, which is a binary prediction indicating whether the input is fake news or not.

5. Evaluation and Validation

To evaluate our model, we used the confusion matrix. We train our data on 80% of the dataset and tested it on remaining 20% (we used 10% for the validation)

6. Web Application Development

We develop the application using the Flask framework. We developed the frontend using HTML, JavaScript and CSS. We also used bootstrap for the frontend development.

7. Testing and Improvements

We performed some tests on the applications. All the tests history has been stored in a csv file.

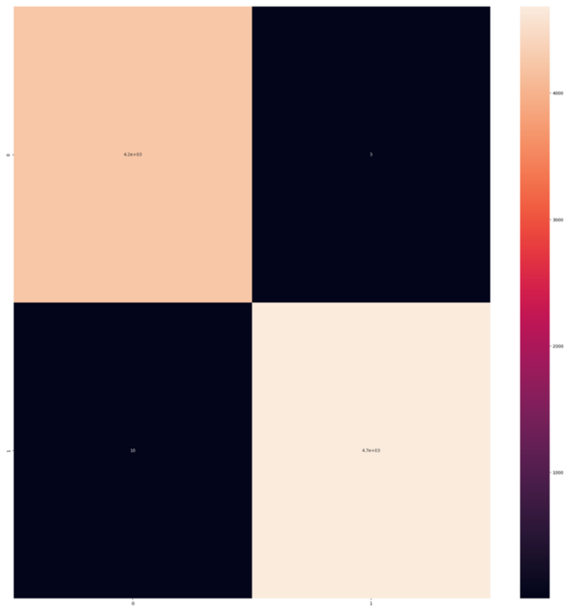
V. RESULTS

i. MODELS RESULT

After building and training our models, we obtained the following accuracy rate:

Models	**Accuracy**
SVM	55 %
Logistic Regression	51 %
KNN	58 %
LSTM (our model)	**99,85 %**

As noticed, the better model is the LSTM model. Let’s now present the confusion matrix of our model.



Img 10: LSTM model confusion matrix

ii. LIMITS

Even though our model provided us a high accuracy rate it can't be generalized due to the nature of the data we used to develop it. Our evolution perspective is to add more data (and related to diverse political topics and with a long period range) to the train the model.
Plus, our application only handles English articles. A next step should be to create a multilingual app to make it more user friendly.

VI. CONCLUSION

In conclusion, this project demonstrates how artificial intelligence can help address the major challenge of spreading fake news in the US political domain. By employing advanced natural language processing techniques and an LSTM model, we achieved a remarkable 99% accuracy rate in detecting fake news. The integration of this model into a user-friendly web application, developed with Flask, HTML, CSS, and JavaScript, showcases AI's ability to provide practical and accessible tools to the public.

Artificial intelligence has profoundly transformed our world, particularly through its concerning influence on the spread of fake news and deepfakes. These phenomena threaten the reliability of information and can erode public trust and the foundations of democracy. However, AI also offers solutions to combat these issues. Fake news and deepfake detection technologies, developed with sophisticated algorithms, are essential for identifying falsified content. Additionally, public education on information manipulation and the dangers of fake news and deepfakes is crucial.

It is imperative for governments, businesses, and individuals to collaborate effectively in combating this phenomenon. Digital platforms must implement strict policies to limit the spread of fake news and deepfakes while preserving freedom of expression.

AI represents both a threat and an opportunity in the fight against fake news and deepfakes. It is our responsibility to strike a balance between technological innovation and protecting our society from misinformation. Therefore, continued investment in research and development of new technologies to strengthen our ability to identify and counter harmful content is essential. Concurrently, raising public awareness of these issues and promoting critical thinking are indispensable measures to reduce the impact of misinformation.

With a proactive and collaborative approach, we can work together to preserve truth and transparency in our constantly evolving digital world. This project contributes to this effort by providing a fake news prediction model, demonstrating AI's potential to improve the reliability of online information and support a better-informed and more critical society.

BIBLIOGRAPHY

- **https://cits.ucsb.edu/fake-news/brief-history**
- https://www.washingtonpost.com/national-security/2022/05/28/dominion-voting-machines-cisa-review/
- https://www.bbc.com/news/election-us-2020-54959962
- **https://www.mdpi.com/2078-2489/14/12/627**
- **https://www.nature.com/articles/s41467-018-07761-2**
- **https://www.aeaweb.org/articles?id=10.1257/jep.31.2.211**
- https://www.sciencedirect.com/science/article/abs/pii/S2214785322016947
- https://www.sciencedirect.com/science/article/pii/S2405844023075904

ANNEX

Acronym

- AI: Artificial Intelligence
- App: Applications
- CNN: Convolutional Neural Network
- CSV: Comma Separated Values
- CSS: Cascading Style Sheets
- HTML: HyperText Markup Language
- ID: Identifier
- Img: Image
- KNN: K Nearest Neighbors
- LSTM: Long Short-Term Memory
- NLP: Natural Language Processing
- NLTK: Natural Language Toolkit
- NGO: Non-Governmental Organizations
- ReLU: Rectified Linear Unit
- RNN: Recurrent Neural Network
- SVM: Support Vector Machines
- TF-IDF: Term Frequency-Inverse Document Frequency

Fake News Detection Python Module

```python
import tensorflow as tf
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud, STOPWORDS
import nltk
nltk.download('punkt')
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
import re
from nltk.stem import PorterStemmer, WordNetLemmatizer
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize, sent_tokenize
import gensim
from gensim.utils import simple_preprocess
from gensim.parsing.preprocessing import STOPWORDS
import keras
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from nltk import word_tokenize
from tensorflow.keras.preprocessing.text import one_hot, Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Flatten, Embedding, Input, LSTM, Conv1D, MaxPool1D, Bidirectional
from tensorflow.keras.models import Model

#Importing Data sets

df_true = pd.read_csv("/Users/yassineseidou/Desktop/PROJETS TECHNIQUE/PROJET D'IA - DEEP FAKES:FAKE NEWS/FINALS_
df_fake = pd.read_csv("/Users/yassineseidou/Desktop/PROJETS TECHNIQUE/PROJET D'IA - DEEP FAKES:FAKE NEWS/FINALS_

#isfake = 0 then the information is true

df_true ['isfake'] = 0

#isfake = 0 then the information is true

df_true ['isfake'] = 0

#isfake = 1 then the information is False

df_fake ['isfake'] = 1

df = pd.concat([df_true, df_fake]).reset_index(drop = True)

#Let's create a new variable composed by the title of the text and the text itself

df['original'] = df['title'] + ' ' + df['text']

nltk.download("stopwords")

from nltk.corpus import stopwords
stop_words = stopwords.words('english')
stop_words.extend(['from', 'subject', 're', 'edu', 'use'])

def preprocess(text):
    result = [] #creating a list where i'll append my tokens
    for token in gensim.utils.simple_preprocess(text): #iterating in the next conditions on all the token generated
        if token not in gensim.parsing.preprocessing.STOPWORDS and len(token) > 3 and token not in stop_words: #verify
            result.append(token) #If condition verified, append the tokens to the list created

    return result

df['clean'] = df['original'].apply(preprocess)

list_of_words = [] #creating a list where i'll append the words present in the variable 'clean'
for i in df.clean: #iterating on all the elements of the variable 'clean'
    for j in i: #iterating on all the words present in each element of the variable 'clean'
        list_of_words.append(j)

total_words = len(list(set(list_of_words)))
df['clean_joined'] = df['clean'].apply(lambda x: " ".join(x))

maxlen = -1 #this variable is initalized to stock the max length in all the documents
for doc in df.clean_joined: #iteration on all the elements of the variable 'clean_joined'
    tokens = nltk.word_tokenize(doc) #tokenizing the text of all doc and append it to tokens
    if(maxlen<len(tokens)): #verifying if the max length is lower than the tokens length
        maxlen = len(tokens) #if yes append maxlen take automatically the value of the lenght of the tokens
print("The maximum number of words in any document is =", maxlen)

# split data into test and train

x_train, x_test, y_train, y_test = train_test_split(df.clean_joined, df.isfake, test_size = 0.2)

# Create a tokenizer to tokenize the words and create sequences of tokenized words


tokenizer = Tokenizer(num_words = total_words)
tokenizer.fit_on_texts(x_train)
train_sequences = tokenizer.texts_to_sequences(x_train)
test_sequences = tokenizer.texts_to_sequences(x_test)

print("The encoding for document\n",df.clean_joined[0],"\n is : ",train_sequences[0])

# Add padding can either be maxlen = 4406 or smaller number maxlen = 40 seems to work well based on results


padded_train = pad_sequences(train_sequences,maxlen = 40, padding = 'post', truncating = 'post')
padded_test = pad_sequences(test_sequences,maxlen = 40, truncating = 'post')


for i,doc in enumerate(padded_train[:2]):
    print("The padded encoding for document",i+1," is : ",doc)

#Build and train the model

```

```
# Sequential Model

model = Sequential()

# embeddidng layer

model.add(Embedding(total_words, output_dim = 128))

# Bi-Directional RNN and LSTM

model.add(Bidirectional(LSTM(128)))

# Dense layers

model.add(Dense(128, activation = 'relu'))
model.add(Dense(1,activation= 'sigmoid'))
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['acc'])
model.summary()

y_train = np.asarray(y_train)

# train the model


model.fit(padded_train, y_train, batch_size = 64, validation_split = 0.1, epochs = 2)

# make prediction

pred = model.predict(padded_test)

# if the predicted value is >0.5 it is real else it is fake

# make prediction

pred = model.predict(padded_test)

# if the predicted value is >0.5 it is real else it is fake

prediction = []
for i in range(len(pred)):
    if pred[i].item() > 0.5:
        prediction.append(1)
    else:
        prediction.append(0)

# getting the accuracy

accuracy = accuracy_score(list(y_test), prediction)

print("Model Accuracy : ", accuracy)

# category dict
category = { 0: 'Fake News', 1 : "Real News"}

def FakeNewsPredictor (model, sentence):
    sentence_tokens = tokenizer.texts_to_sequences([sentence])
    padded_sentence= pad_sequences(sentence_tokens, maxlen= 40, padding= 'post', truncating='post')
    prediction_sentence = model.predict(padded_sentence)
    if prediction_sentence [0][0]>= 0.5 :
        return 'Our model has classified this text as misinformation. Please verify the source of the information. Ple
    else :
        return 'Our model has classified this text as accurate information. Please enter a new text.'


sentence = " "

FakeNewsPredictor (model, sentence)
```

Flask Module

```
from flask import Flask, render_template, request
from FakeNewsDetection.fakenews_detection import FakeNewsPredictor
from FakeNewsDetection.data_storage_in_csv import sauvegarder_resultat_csv
from tensorflow.keras.models import load_model
import csv
import uuid
import pandas as pd

app = Flask(__name__)

# Charger le modèle
model = load_model("/Users/yassineseidou/Desktop/PROJETS TECHNIQUE/PROJET D'IA - DEEP FAKES:FAKE NEWS/

# Chemin du fichier CSV pour stocker l'historique
fichier_historique = "/Users/yassineseidou/Desktop/PROJETS TECHNIQUE/PROJET D'IA - DEEP FAKES:FAKE NEW

@app.route('/')
def home():
    return render_template('index3.html')

@app.route('/predict', methods=['POST'])
def predict():
    if request.method == 'POST':
        sentence = request.form['information']
        result = FakeNewsPredictor(model, sentence)
        # Récupérer l'ID utilisateur généré automatiquement
        id_utilisateur = str(uuid.uuid4())

        # Enregistrer les résultats dans le CSV avec l'ID utilisateur
        sauvegarder_resultat_csv(sentence, result)

        return render_template('index3.html', prediction=result)

```

```python
@app.route('/historique')
def historique():
    # Charger le fichier CSV en tant que DataFrame avec pandas
    historique_df = pd.read_csv(fichier_historique)

    # Récupérer les 5 dernières lignes de l'historique
    historique_df_tail = historique_df.tail(5)

    # Convertir le DataFrame en HTML pour l'affichage dans le template
    historique_html = historique_df_tail.to_html(index=False)

    # Passer l'historique HTML au template
    return render_template('index3.html', historique=historique_html)


if __name__ == '__main__':
    app.run(debug=True, port=8000)
```

Frontend Module

```html
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>RekeesAI</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
    <link rel="icon" href="RekeesAI" type="image/png">
    <style>
        textarea:invalid {
            border: 1px solid red;
        }
    </style>
</head>
<body>
    <header class="bg-dark text-light py-4">
        <div class="container">
            <div class="d-flex justify-content-end align-items-center">
                <h1 class="mb-0">RekeesAI</h1>
                <nav class="ms-auto">
                    <a href="about.html" class="text-light mx-2">About</a>
                    <a href="contact.html" class="text-light mx-2">Contact Us</a>
                    <a href="about.html" class="text-light mx-2">Blog</a>
                    <a href="about.html" class="text-light mx-2">Let's Play a Bit</a>
                </nav>
            </div>
            <p class="text-muted">RekeesAI helps you verify the reliability of information.</p>
        </div>
    </header>
    <main class="container my-4">
        <section class="verification-information">
            <h2 class="mb-4">Verify the reliability of information.</h2>
            <form method="POST" action="/predict" id="verification-form">
                <div class="mb-3">
```

```html
                <div class="mb-3">
                    <label for="id_utilisateur" class="form-label" style="display:none;">ID Utilisateur:</label>
                    <input type="hidden" id="id_utilisateur" name="id_utilisateur" value="{{ id_utilisateur }}">
                    <label for="information" class="form-label">Enter a text of at least 40 words:</label>
                    <textarea id="information" name="information" class="form-control" rows="5" minlength="40" required></te
                    <span id="word-count" class="text-muted">0 words / 40 minimum</span>
                </div>
                <button type="submit" class="btn btn-primary">Verify</button>
            </form>
            <div class="resultats mt-4">
                {% if prediction %}
                    <p>{{ prediction }}</p>
                {% endif %}
            </div>
        </section>
    </main>
    <footer class="bg-dark text-light py-3">
        <div class="container">
            <p class="mb-0">&copy; 2024 RekeesAI</p>
            <p class="mb-0 footer-text small" style="text-align: justify; font-weight: 100;">
                RekeesAI helps you verify the reliability of information. RekeesAI exclusively verifies political informatio
            </p>
            <nav class="nav justify-content-center">
                <a href="privacy.html" class="text-light mx-3">Privacy Policy</a>
                <a href="terms.html" class="text-light mx-3">Terms of Use</a>
                <a href="#" class="text-light mx-3">Copyrights 2024</a>
                <a href="#" class="text-light mx-3">Social Media</a>
            </nav>
        </div>
    </footer>
    <script src="{{ url_for('static', filename='script.js') }}"></script>
    <script>
```

```html
    </footer>
    <script src="{{ url_for('static', filename='script.js') }}"></script>
    <script>
        document.addEventListener('DOMContentLoaded', function () {
            const textarea = document.getElementById('information');
            const wordCountSpan = document.getElementById('word-count');

            textarea.addEventListener('input', function () {
                const words = textarea.value.trim().split(/\s+/);
                const wordCount = words.length;
                wordCountSpan.textContent = `${wordCount} mots / 40 minimum`;
            });
        });
    </script>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```