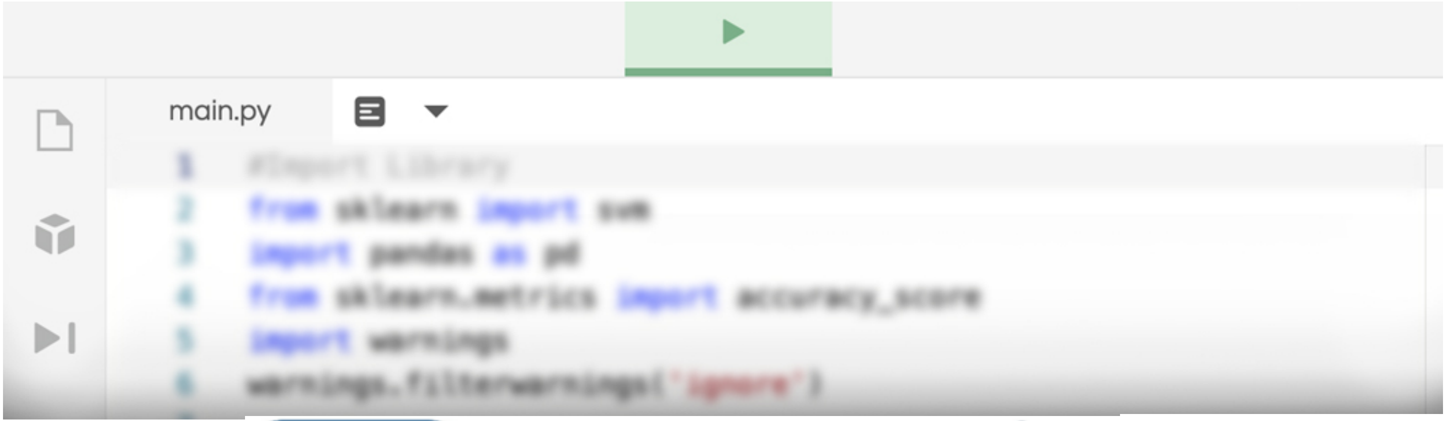
1. Data Collection and Exploration

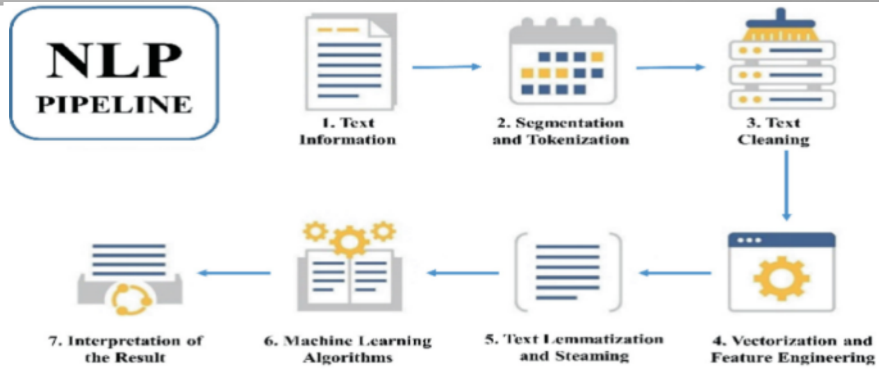
For this project, we will use the Fake and Real News Dataset available on Kaggle. The dataset contains two CSV files: one with real news articles and another with fake news articles. You can download the dataset from this link: https://www.kaggle.com/clmentbisaillon/fake-and-real-news-dataset

Once you have downloaded the dataset, you can load it into a Pandas DataFrame.
The '*real_news*' DataFrame contains real news articles and their labels, and the '*fake_news*' DataFrame contains fake news articles and their labels. Let's take a look at the first few rows of each DataFrame to get an idea of what the data looks like::

Python Code:



```
#Import Library
from sklearn import svm
import pandas as pd
from sklearn.metrics import accuracy_score
import warnings
warnings.filterwarnings('ignore')
```



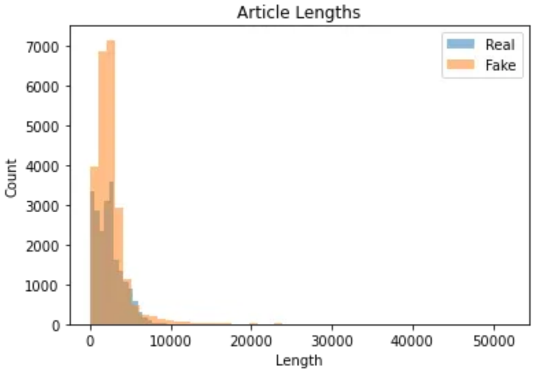
NLP
PIPELINE
1. Text Information
2. Segmentation and Tokenization
3. Text Cleaning
4. Vectorization and Feature Engineering
5. Text Lemmatization and Steaming
6. Machine Learning Algorithms
7. Interpretation of the Result

As we can see, the data contains several columns: the title of the article, the text of the article, the subject of the article, and the date it was published. We will be using the title and text columns to train our model.

Before we can start training our model, we need to do some exploratory data analysis to get a sense of the data. For example, we can plot the distribution of article lengths in each dataset using the following code:

```
import matplotlib.pyplot as plt real_lengths = real_news['text'].apply(len) fake_lengths = fake_news['text'].apply(len) plt.hist(real_lengths, bins=50, alpha=0.5, label='Real') plt.hist(fake_lengths, bins=50, alpha=0.5, label='Fake') plt.title('Article Lengths') plt.xlabel('Length') plt.ylabel('Count') plt.legend() plt.show()
```

The output should look something like this:



As we can see, the length of the articles is highly variable, with some articles being very short (less than 1000 characters) and others being quite long (more than 40,000 characters). We will need to take this into account when preprocessing the text.

We can also look at the most common words in each dataset using the following code:

```
from collections import Counter import nltk #downloading stopwords and punkt nltk.download('stopwords') nltk.download('punkt') def get_most_common_words(texts, num_words=10): all_words = [] for text in texts: all_words.extend(nltk.word_tokenize(text.lower())) stop_words = set(nltk.corpus.stopwords.words('english')) words = [word for word in all_words if word.isalpha() and word not in stop_words] word_counts = Counter(words) return word_counts.most_common(num_words) real_words = get_most_common_words(real_news['text']) fake_words = get_most_common_words(fake_news['text']) print('Real News:', real_words) print('Fake News:', fake_words)
```

The output should look something like this:

```
Real News: [('trump', 32505), ('said', 15757), ('us', 15247), ('president', 12788), ('would', 12337), ('people', 10749), ('one', 10681), ('also', 9927), ('new', 9825), ('state', 9820)] Fake News: [('trump', 10382), ('said', 7161), ('hillary', 3890), ('clinton', 3588), ('one', 3466), ('people', 3305), ('would', 3257), ('us', 3073), ('like', 3056), ('also', 3005)]
```

As we can see, some of the most common words in both datasets are related to politics and the current US president, Donald Trump. However, there are some differences between the two datasets, with the fake news dataset containing more references to Hillary Clinton and a greater use of words like “like”.

Model Performance without removing stopwords(used logistic regression)

```
Accuracy: 0.9953 Precision: 0.9940 Recall: 0.9963 F1 Score: 0.9951
```

2. Text Preprocessing

Before we can start training our model, we need to preprocess the text data. The preprocessing steps we will perform are:

1. Lowercasing the text
2. Removing punctuation and digits
3. Removing stop words
4. Stemming or lemmatizing the text

Lowercasing the Text

Lowercasing the text refers to converting all the letters in a piece of text to lowercase. This is a common text preprocessing step that can be useful for improving the accuracy of text classification models. For example, “Hello” and “hello” would be considered two different words by a model that does not account for case, whereas if the text is converted to lowercase, they would be treated as the same word.

Removing Punctuation and Digits

Removing punctuation and digits refers to removing non-alphabetic characters from a text. This can be useful for reducing the complexity of the text and making it easier for a model to analyze. For example, the words “Hello,” and “Hello!” would be considered different words by a text analysis model if it doesn’t account for the punctuation.

Removing Stop Words

Stop words are words that are very common in a language and do not carry much meaning, such as “the”, “and”, “in”, etc. Removing stop words from a piece of text can help reduce the dimensionality of the data and focus on the most important words in the text. This can also help improve the accuracy of a text classification model by reducing noise in the data.

Stemming or Lemmatizing the Text

Stemming and lemmatizing are common techniques for reducing words to their base form. Stemming involves removing the suffixes of words to produce a stem or root word. For example, the word “jumping” would be stemmed to “jump.” This technique can be useful for reducing the dimensionality of the data, but it can sometimes result in stems that are not actual words.

Conversely, Lemmatizing involves reducing words to their base form using a dictionary or morphological analysis. For example, the word “jumping” would be lemmatized to “jump”, which is an actual word. This technique can be more accurate than stemming but also more computationally expensive.

Both stemming and lemmatizing can reduce the dimensionality of text data and make it easier for a model to analyze. However, it is important to note that they can sometimes result in loss of information, so it is important to experiment with both techniques and determine which works best for a particular text classification problem.

We will perform these steps using the NLTK library, which provides various text-processing tools.

```
from nltk.corpus import stopwords from nltk.tokenize import word_tokenize from nltk.stem import
PorterStemmer, WordNetLemmatizer import string nltk.download('wordnet') stop_words =
set(stopwords.words('english')) stemmer = PorterStemmer() lemmatizer = WordNetLemmatizer() def
preprocess_text(text): # Lowercase the text text = text.lower() # Remove punctuation and digits text =
text.translate(str.maketrans('', '', string.punctuation + string.digits)) # Tokenize the text words =
word_tokenize(text) # Remove stop words words = [word for word in words if word not in stop_words] # Stem or
lemmatize the words words = [stemmer.stem(word) for word in words] # Join the words back into a string text =
' '.join(words) return text
```

```
# This Python 3 environment comes with many helpful analytics libraries
installed
# It is defined by the kaggle/python Docker image:
https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load
import warnings
warnings.filterwarnings('ignore')
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

import nltk
from nltk.sentiment import SentimentIntensityAnalyzer
import warnings
warnings.filterwarnings("ignore")
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will
list all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/)
that gets preserved as output when you create a version using "Save & Run
All"
# You can also write temporary files to /kaggle/temp/, but they won't be
saved outside of the current session
```

```
/kaggle/input/fake-and-real-news-dataset/True.csv
/kaggle/input/fake-and-real-news-dataset/Fake.csv
```

Loading Data

In [2]:

```
true = pd.read_csv('/kaggle/input/fake-and-real-news-dataset/True.csv')
fake = pd.read_csv('/kaggle/input/fake-and-real-news-dataset/Fake.csv')
```

In [3]:

```
fake['Category'] = 'fake'
fake
```

Out[3]:

	title	text	subject	date	Category
0	Donald Trump Sends Out Embarrassing New Year’...	Donald Trump just couldn t wish all Americans ...	News	December 31, 2017	fake
1	Drunk Bragging Trump Staffer Started Russian ...	House Intelligence Committee Chairman Devin Nu...	News	December 31, 2017	fake
2	Sheriff David Clarke Becomes An Internet Joke...	On Friday, it was revealed that former Milwauk...	News	December 30, 2017	fake
3	Trump Is So Obsessed He Even Has Obama’s Name...	On Christmas day, Donald Trump announced that ...	News	December 29, 2017	fake
4	Pope Francis Just Called Out Donald Trump Dur...	Pope Francis used his annual Christmas Day mes...	News	December 25, 2017	fake
...	...	...	...	...	...
23476	McPain: John McCain Furious That Iran Treated ...	21st Century Wire says As 21WIRE reported earl...	Middle-east	January 16, 2016	fake
23477	JUSTICE? Yahoo Settles E-mail Privacy Class-ac...	21st Century Wire says It s a familiar theme. ...	Middle-east	January 16, 2016	fake
23478	Sunnistan: US and Allied ‘Safe Zone’ Plan to T...	Patrick Henningsen 21st Century WireRemember ...	Middle-east	January 15, 2016	fake
23479	How to Blow $700 Million: Al Jazeera America F...	21st Century Wire says Al Jazeera America will...	Middle-east	January 14, 2016	fake

23480	10 U.S. Navy Sailors Held by Iranian Military ...	21st Century Wire says As 21WIRE predicted in ...	Middle-east	January 12, 2016	fake

```
23481 rows × 5 columns
```

In [4]:

```
true['Category'] = 'true'
true
```

Out[4]:

	title	text	subject	date	Category
0	As U.S. budget fight looms, Republicans flip t...	WASHINGTON (Reuters) - The head of a conservat...	politicsNews	December 31, 2017	true
1	U.S. military to accept transgender recruits o...	WASHINGTON (Reuters) - Transgender people will...	politicsNews	December 29, 2017	true
2	Senior U.S. Republican senator: 'Let Mr. Muell...	WASHINGTON (Reuters) - The special counsel inv...	politicsNews	December 31, 2017	true
3	FBI Russia probe helped by Australian diplomat...	WASHINGTON (Reuters) - Trump campaign adviser ...	politicsNews	December 30, 2017	true
4	Trump wants Postal Service to charge 'much mor...	SEATTLE/WASHINGTON (Reuters) - President Donal...	politicsNews	December 29, 2017	true
...	...	...	...	...	...
21412	'Fully committed' NATO backs new U.S. approach...	BRUSSELS (Reuters) - NATO allies on Tuesday we...	worldnews	August 22, 2017	true
21413	LexisNexis withdrew two products from Chinese ...	LONDON (Reuters) - LexisNexis, a provider of l...	worldnews	August 22, 2017	true
21414	Minsk cultural hub becomes haven from authorities	MINSK (Reuters) - In the shadow of disused Sov...	worldnews	August 22, 2017	true
21415	Vatican upbeat on possibility of Pope Francis ...	MOSCOW (Reuters) -	worldnews	August 22, 2017	true

		Vatican Secretary of State ...			
21416	Indonesia to buy $1.14 billion worth of Russia...	JAKARTA (Reuters) - Indonesia will buy 11 Sukh...	worldnews	August 22, 2017	true

```
21417 rows × 5 columns
```

In [5]:

```
#Now let's combine the whole dataset into one
data = pd.concat([fake, true], ignore_index = True)
data
```

Out[5]:

	title	text	subject	date	Category
0	Donald Trump Sends Out Embarrassing New Year’...	Donald Trump just couldn t wish all Americans ...	News	December 31, 2017	fake
1	Drunk Bragging Trump Staffer Started Russian ...	House Intelligence Committee Chairman Devin Nu...	News	December 31, 2017	fake
2	Sheriff David Clarke Becomes An Internet Joke...	On Friday, it was revealed that former Milwauk...	News	December 30, 2017	fake
3	Trump Is So Obsessed He Even Has Obama’s Name...	On Christmas day, Donald Trump announced that ...	News	December 29, 2017	fake
4	Pope Francis Just Called Out Donald Trump Dur...	Pope Francis used his annual Christmas Day mes...	News	December 25, 2017	fake
...	...	...	...	...	...
44893	'Fully committed' NATO backs new U.S. approach...	BRUSSELS (Reuters) - NATO allies on Tuesday we...	worldnews	August 22, 2017	true
44894	LexisNexis withdrew two products from Chinese ...	LONDON (Reuters) - LexisNexis, a provider of l...	worldnews	August 22, 2017	true

44895	Minsk cultural hub becomes haven from authorities	MINSK (Reuters) - In the shadow of disused Sov...	worldnews	August 22, 2017	true
44896	Vatican upbeat on possibility of Pope Francis ...	MOSCOW (Reuters) - Vatican Secretary of State ...	worldnews	August 22, 2017	true
44897	Indonesia to buy $1.14 billion worth of Russia...	JAKARTA (Reuters) - Indonesia will buy 11 Sukh...	worldnews	August 22, 2017	true

```
44898 rows × 5 columns
```

In [6]:

```
data.shape
```

Out[6]:

```
(44898, 5)
```

Preprocessing

In [7]:

```
data['Category'].value_counts()
```

Out[7]:

```
Category
fake    23481
true    21417
Name: count, dtype: int64
```

In [8]:

```
#Transforming category values to numerical
from sklearn.preprocessing import LabelEncoder
encoder = LabelEncoder()
data['Category'] = encoder.fit_transform(data['Category'])
```

In [9]:

```
data['Category']
```

Out[9]:

```
0         0
1         0
2         0
3         0
```

```
4        0
        ..
44893    1
44894    1
44895    1
44896    1
44897    1
Name: Category, Length: 44898, dtype: int64
```

In [10]:

```
vectorizer = TfidfVectorizer()
title = vectorizer.fit_transform(data['title'])
title
```

Out[10]:

```
<44898x20896 sparse matrix of type '<class 'numpy.float64'>'
	with 546512 stored elements in Compressed Sparse Row format>
```

Modeling

In [11]:

```
from sklearn.model_selection import train_test_split
X = title
y = data['Category']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,
random_state = 42)
```

In [12]:

```
model = SVC()
model.fit(X_train, y_train)
```

Out[12]:

SVC

```
SVC()
```

In [13]:

```
y_pred = model.predict(X_test)
print('Classification Report: ')
print(classification_report(y_test, y_pred))
```

```
Classification Report:
              precision    recall  f1-score   support
```

```
           0       0.97      0.96      0.96      4733
           1       0.95      0.97      0.96      4247

    accuracy                           0.96      8980
   macro avg       0.96      0.96      0.96      8980
weighted avg       0.96      0.96      0.96      8980
```

Thankyou so much for your valuable time :)