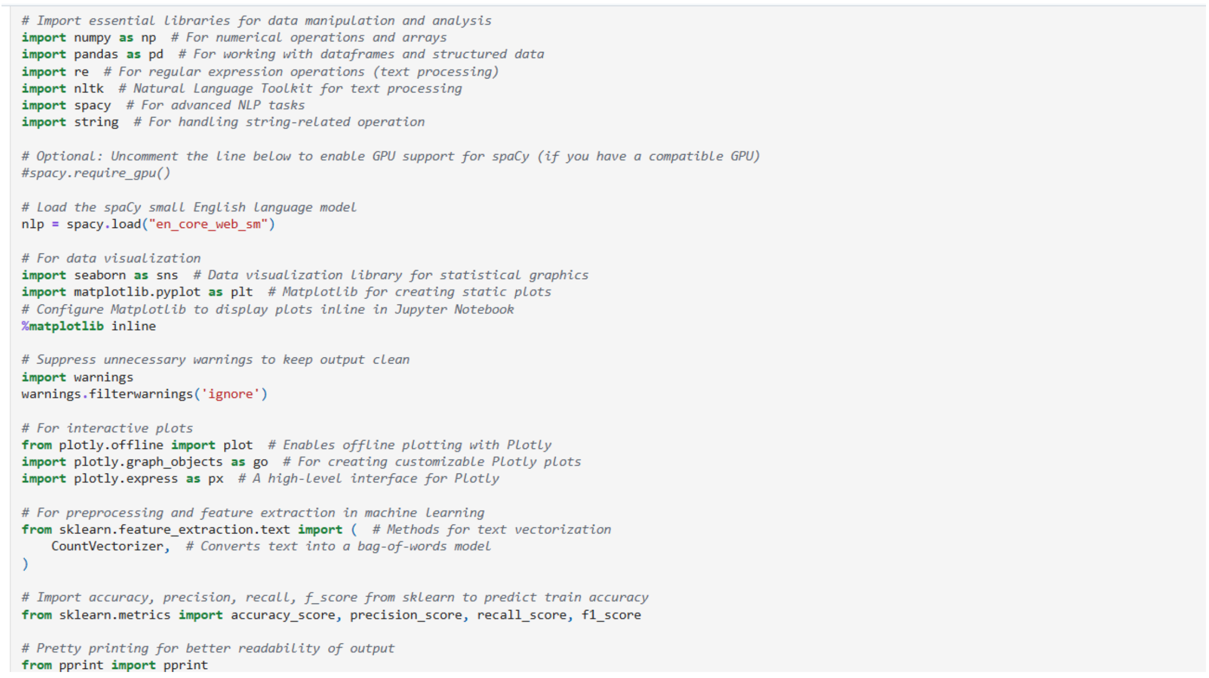
Fake News Detection

Submitted by -Anjali Shaw

Business Objective: The spread of fake news has become a significant challenge in today's digital world. With the massive volume of news articles published daily, it's becoming harder to distinguish between credible and misleading information. This creates a need for systems that can automatically classify news articles as true or fake, helping to reduce misinformation and protect public trust.

In this assignment, you will develop a Semantic Classification model that uses the Word2Vec method to detect recurring patterns and themes in news articles. Using supervised learning models, the goal is to build a system that classifies news articles as either fake or true.

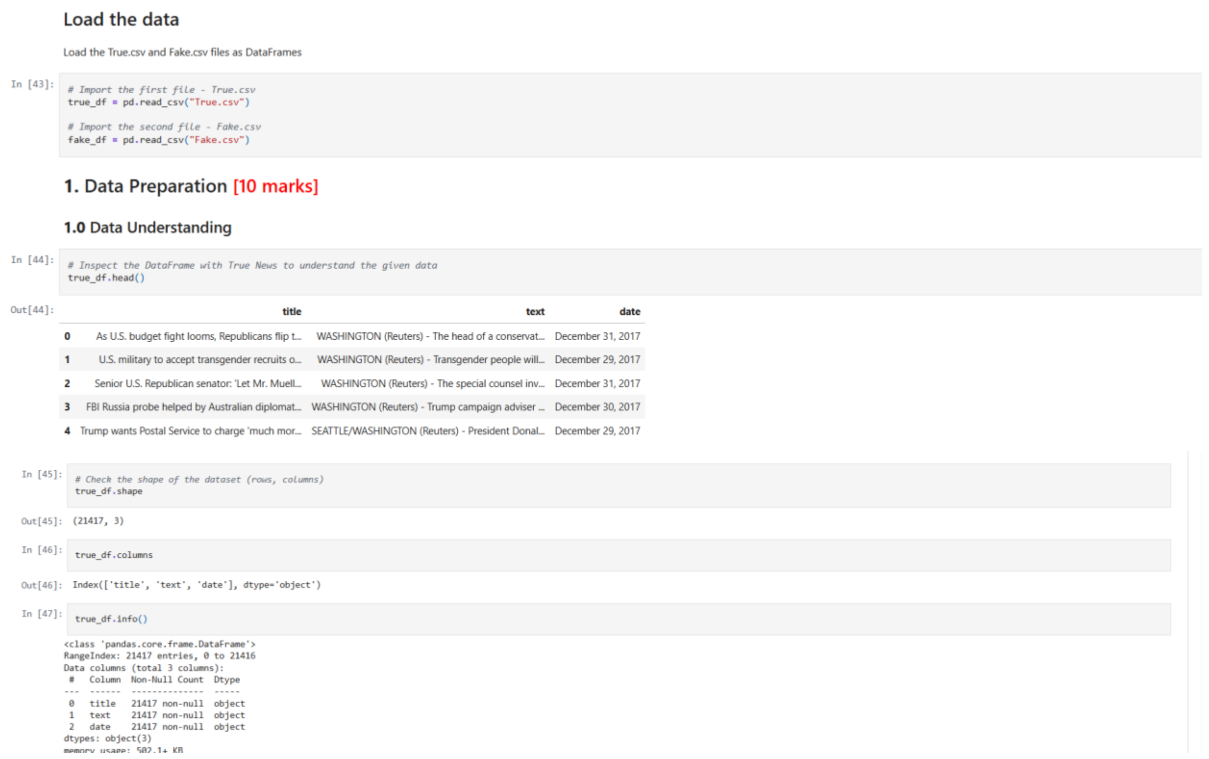
Imported Necessary Libraries



```python
# Import essential libraries for data manipulation and analysis
import numpy as np  # For numerical operations and arrays
import pandas as pd  # For working with dataframes and structured data
import re  # For regular expression operations (text processing)
import nltk  # Natural Language Toolkit for text processing
import spacy  # For advanced NLP tasks
import string  # For handling string-related operation

# Optional: Uncomment the line below to enable GPU support for spaCy (if you have a compatible GPU)
#spacy.require_gpu()

# Load the spaCy small English language model
nlp = spacy.load("en_core_web_sm")

# For data visualization
import seaborn as sns  # Data visualization library for statistical graphics
import matplotlib.pyplot as plt  # Matplotlib for creating static plots
# Configure Matplotlib to display plots inline in Jupyter Notebook
%matplotlib inline

# Suppress unnecessary warnings to keep output clean
import warnings
warnings.filterwarnings('ignore')

# For interactive plots
from plotly.offline import plot  # Enables offline plotting with Plotly
import plotly.graph_objects as go  # For creating customizable Plotly plots
import plotly.express as px  # A high-level interface for Plotly

# For preprocessing and feature extraction in machine learning
from sklearn.feature_extraction.text import (  # Methods for text vectorization
    CountVectorizer,  # Converts text into a bag-of-words model
)

# Import accuracy, precision, recall, f_score from sklearn to predict train accuracy
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

# Pretty printing for better readability of output
from pprint import pprint
```

1. Data Ingestion and Preparation

Load file and convert it to a dataframe



Load the data

Load the True.csv and Fake.csv files as DataFrames

```
In [43]:  # Import the first file - True.csv
          true_df = pd.read_csv("True.csv")

          # Import the second file - Fake.csv
          fake_df = pd.read_csv("Fake.csv")
```

1. Data Preparation [10 marks]

1.0 Data Understanding

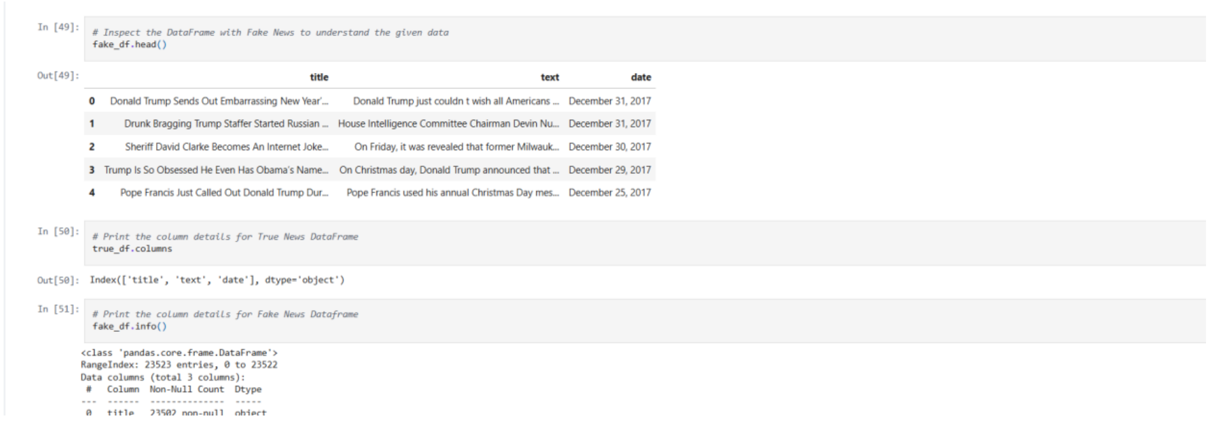
```
In [44]:  # Inspect the DataFrame with True News to understand the given data
          true_df.head()
```

Out[44]:

	title	text	date
0	As U.S. budget fight looms, Republicans flip t...	WASHINGTON (Reuters) - The head of a conservat...	December 31, 2017
1	U.S. military to accept transgender recruits o...	WASHINGTON (Reuters) - Transgender people will...	December 29, 2017
2	Senior U.S. Republican senator: 'Let Mr. Muell...	WASHINGTON (Reuters) - The special counsel inv...	December 31, 2017
3	FBI Russia probe helped by Australian diplomat...	WASHINGTON (Reuters) - Trump campaign adviser ...	December 30, 2017
4	Trump wants Postal Service to charge 'much mor...	SEATTLE/WASHINGTON (Reuters) - President Donal...	December 29, 2017

```
In [45]:  # Check the shape of the dataset (rows, columns)
          true_df.shape

Out[45]:  (21417, 3)

In [46]:  true_df.columns

Out[46]:  Index(['title', 'text', 'date'], dtype='object')

In [47]:  true_df.info()

         <class 'pandas.core.frame.DataFrame'>
         RangeIndex: 21417 entries, 0 to 21416
         Data columns (total 3 columns):
          #   Column  Non-Null Count  Dtype
         ---  ------  --------------  -----
          0   title   21417 non-null  object
          1   text    21417 non-null  object
          2   date    21417 non-null  object
         dtypes: object(3)
         memory usage: 502.1+ KB
```

The true_news file has 21417 records and 3 columns namely title, text and date where title is the news headline and text is the news content



```
In [49]:  # Inspect the DataFrame with Fake News to understand the given data
          fake_df.head()
```

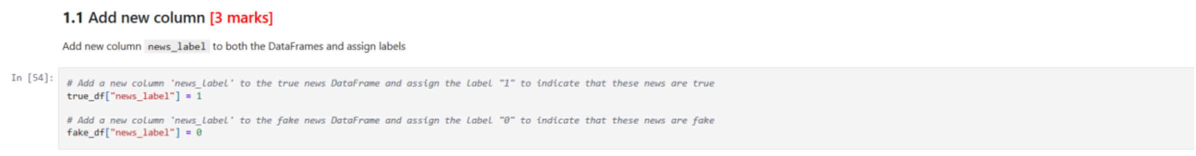
Out[49]:

	title	text	date
0	Donald Trump Sends Out Embarrassing New Year'...	Donald Trump just couldn t wish all Americans ...	December 31, 2017
1	Drunk Bragging Trump Staffer Started Russian ...	House Intelligence Committee Chairman Devin Nu...	December 31, 2017
2	Sheriff David Clarke Becomes An Internet Joke...	On Friday, it was revealed that former Milwauk...	December 30, 2017
3	Trump Is So Obsessed He Even Has Obama's Name...	On Christmas day, Donald Trump announced that ...	December 29, 2017
4	Pope Francis Just Called Out Donald Trump Dur...	Pope Francis used his annual Christmas Day mes...	December 25, 2017

```
In [50]:  # Print the column details for True News DataFrame
          true_df.columns

Out[50]:  Index(['title', 'text', 'date'], dtype='object')

In [51]:  # Print the column details for Fake News Dataframe
          fake_df.info()

         <class 'pandas.core.frame.DataFrame'>
         RangeIndex: 23523 entries, 0 to 23522
         Data columns (total 3 columns):
          #   Column  Non-Null Count  Dtype
         ---  ------  --------------  -----
          0   title   23502 non-null  object
```

The true_news file has 23523 records and 3 columns namely title, text and date where title is the news headline and text is the news content

Add a new column 'news_label' to the DataFrame and assign the label "1" to indicate that these news are true and “0” for fake news

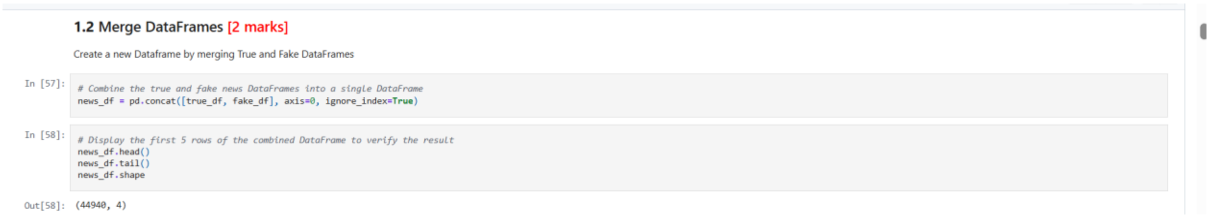


1.1 Add new column [3 marks]

Add new column `news_label` to both the DataFrames and assign labels

```
In [54]:  # Add a new column 'news_label' to the true news DataFrame and assign the label "1" to indicate that these news are true
          true_df["news_label"] = 1

          # Add a new column 'news_label' to the fake news DataFrame and assign the label "0" to indicate that these news are fake
          fake_df["news_label"] = 0
```

Merged dataframe into news_df with total of 44940 rows and 3 columns

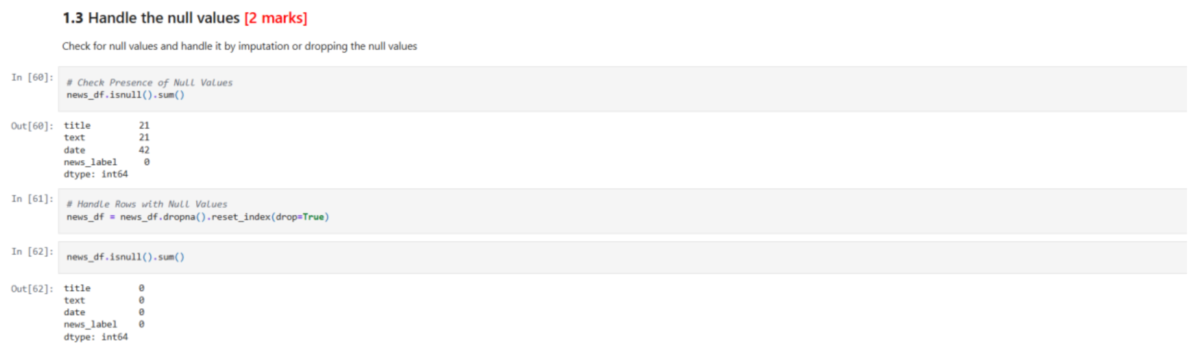


1.2 Merge DataFrames [2 marks]

Create a new Dataframe by merging True and Fake DataFrames

```
In [57]:  # Combine the true and fake news DataFrames into a single DataFrame
          news_df = pd.concat([true_df, fake_df], axis=0, ignore_index=True)

In [58]:  # Display the first 5 rows of the combined DataFrame to verify the result
          news_df.head()
          news_df.tail()
          news_df.shape

Out[58]:  (44940, 4)
```

Dropped rows with null text content

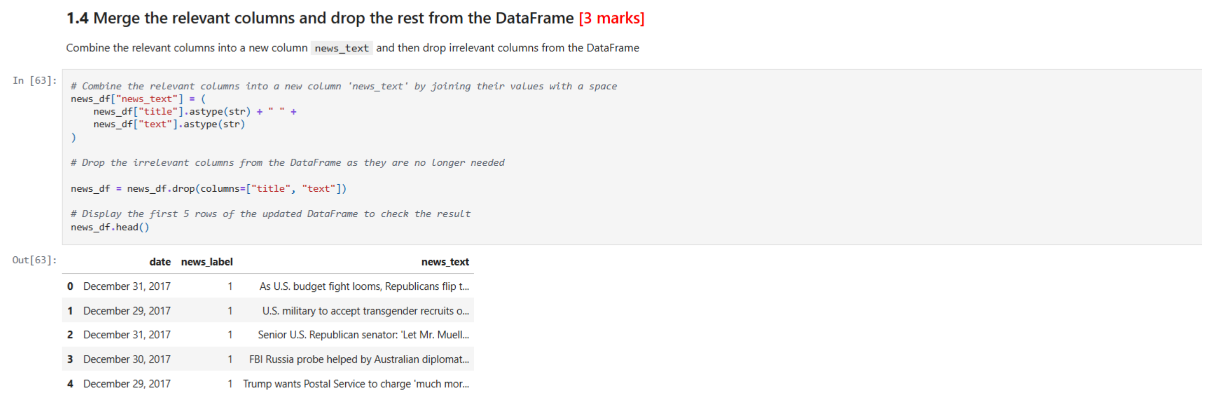


1.3 Handle the null values [2 marks]

Check for null values and handle it by imputation or dropping the null values

```
In [60]:  # Check Presence of Null Values
          news_df.isnull().sum()

Out[60]:  title         21
          text          21
          date          42
          news_label     0
          dtype: int64

In [61]:  # Handle Rows with Null Values
          news_df = news_df.dropna().reset_index(drop=True)

In [62]:  news_df.isnull().sum()

Out[62]:  title         0
          text          0
          date          0
          news_label    0
          dtype: int64
```

Merged title and text for news+text



1.4 Merge the relevant columns and drop the rest from the DataFrame [3 marks]

Combine the relevant columns into a new column `news_text` and then drop irrelevant columns from the DataFrame

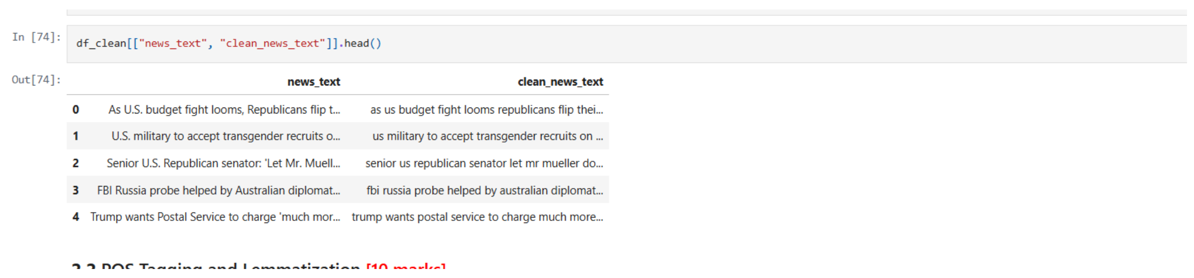
In [63]:
```
# Combine the relevant columns into a new column 'news_text' by joining their values with a space
news_df["news_text"] = (
    news_df["title"].astype(str) + " " +
    news_df["text"].astype(str)
)

# Drop the irrelevant columns from the DataFrame as they are no longer needed

news_df = news_df.drop(columns=["title", "text"])

# Display the first 5 rows of the updated DataFrame to check the result
news_df.head()
```

Out[63]:

	date	news_label	news_text
0	December 31, 2017	1	As U.S. budget fight looms, Republicans flip t...
1	December 29, 2017	1	U.S. military to accept transgender recruits o...
2	December 31, 2017	1	Senior U.S. Republican senator: 'Let Mr. Muell...
3	December 30, 2017	1	FBI Russia probe helped by Australian diplomat...
4	December 29, 2017	1	Trump wants Postal Service to charge 'much mor...

2. Text Preprocessing

Create a DataFrame('df_clean') that will have only the cleaned news text and the lemmatized news text with POS tags removed

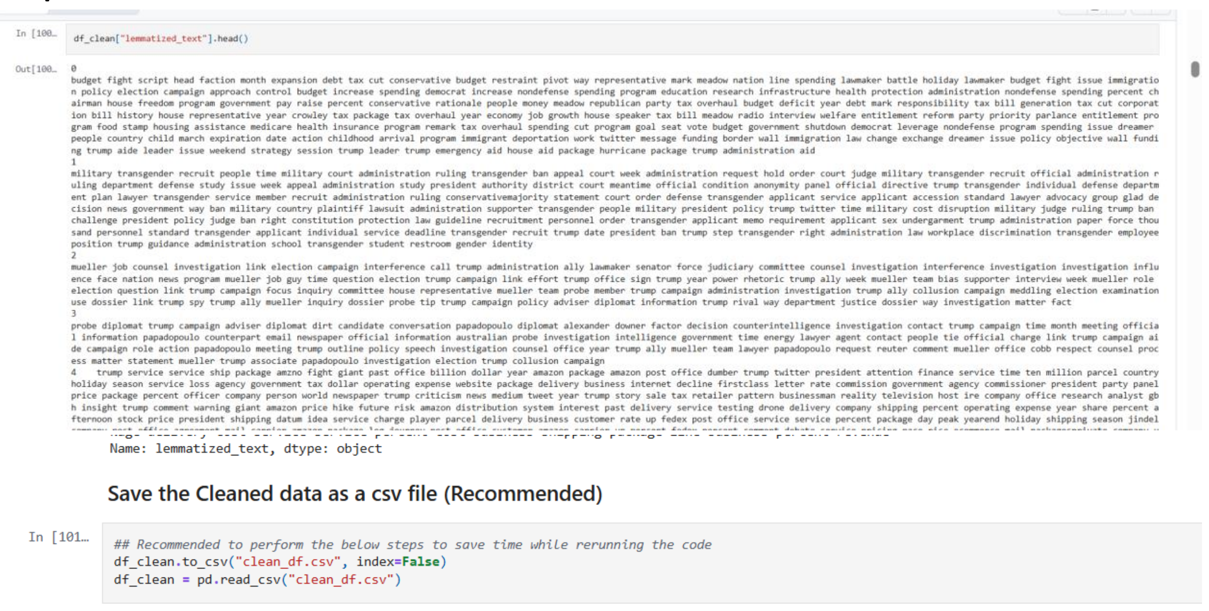


In [74]:
```
df_clean[["news_text", "clean_news_text"]].head()
```

Out[74]:

	news_text	clean_news_text
0	As U.S. budget fight looms, Republicans flip t...	as us budget fight looms republicans flip thei...
1	U.S. military to accept transgender recruits o...	us military to accept transgender recruits on ...
2	Senior U.S. Republican senator: 'Let Mr. Muell...	senior us republican senator let mr mueller do...
3	FBI Russia probe helped by Australian diplomat...	fbi russia probe helped by australian diplomat...
4	Trump wants Postal Service to charge 'much mor...	trump wants postal service to charge much more...

After removal of stopwords, pos tags of NN and NNS kept and lemmatization



In [100…
```
df_clean["lemmatized_text"].head()
```

Out[100…
```
0
budget fight script head faction month expansion debt tax cut conservative budget restraint pivot way representative mark meadow nation line spending lawmaker battle holiday lawmaker budget fight issue immigratio
n policy election campaign approach control budget increase spending democrat increase nondefense spending program education research infrastructure health protection administration nondefense spending percent ch
airman house freedom program government pay raise percent conservative rationale people money meadow republican party tax overhaul budget deficit year debt mark responsibility tax bill generation tax cut corporat
ion bill history house representative year crowley tax package tax overhaul year economy job growth house speaker tax bill meadow radio interview welfare entitlement reform party priority parlance entitlement pro
gram food stamp housing assistance medicare health insurance program remark tax overhaul spending cut program goal seat vote budget government shutdown democrat leverage nondefense program spending issue dreamer
people country child march expiration date action childhood arrival program immigrant deportation work twitter message funding border wall immigration law change exchange dreamer issue policy objective wall fundi
ng trump aide leader issue weekend strategy session trump leader trump emergency aid house aid package hurricane package trump administration aid
1
military transgender recruit people time military court administration ruling transgender ban appeal court week administration request hold order court judge military transgender recruit official administration r
uling department defense study issue week appeal administration study president authority district court meantime official condition anonymity panel official directive trump transgender individual defense departm
ent plan lawyer transgender service member recruit administration ruling conservativemajority statement court order defense transgender applicant service applicant accession standard lawyer advocacy group glad de
cision news government way ban military country plaintiff lawsuit administration supporter transgender people military president policy trump twitter time military cost disruption military judge ruling trump ban
challenge president policy judge ban right constitution protection law guideline recruitment personnel order transgender applicant memo requirement applicant sex undergarment trump administration paper force thou
sand personnel standard transgender applicant individual service deadline transgender recruit trump date president ban trump step transgender right administration law workplace discrimination transgender employee
position trump guidance administration school transgender student restroom gender identity
2
mueller job counsel investigation link election campaign interference call trump administration ally lawmaker senator force judiciary committee counsel investigation interference investigation investigation influ
ence face nation news program mueller job guy time question election trump campaign link effort trump office sign trump year power rhetoric trump ally week mueller team bias supporter interview week mueller role
election question link trump campaign focus inquiry committee house representative mueller team probe member trump campaign administration investigation trump ally collusion campaign meddling election examination
use dossier link trump spy trump ally mueller inquiry dossier probe tip trump campaign policy adviser diplomat information trump rival way department justice dossier way investigation matter fact
3
probe diplomat trump campaign adviser diplomat dirt candidate conversation papadopoulo diplomat alexander downer factor decision counterintelligence investigation contact trump campaign time month meeting officia
l information papadopoulo counterpart email newspaper official information australian probe investigation intelligence government time energy lawyer agent contact people tie official charge link trump campaign ai
de campaign role action papadopoulo meeting trump outline policy speech investigation counsel office year trump ally mueller team lawyer papadopoulo request reuter comment mueller office cobb respect counsel proc
ess matter statement mueller trump associate papadopoulo investigation election trump collusion campaign
4    trump service service ship package amzno fight giant past office billion dollar year amazon package amazon post office dumber trump twitter president attention finance service time ten million parcel country
holiday season service loss agency government tax dollar operating expense website package delivery business internet decline firstclass letter rate commission government agency commissioner president party panel
price package percent officer company person world newspaper trump criticism news medium tweet year trump story sale tax retailer pattern businessman reality television host ire company office research analyst gb
h insight trump comment warning giant amazon price hike future risk amazon distribution system interest past delivery service testing drone delivery company shipping percent operating expense year share percent a
fternoon stock price president shipping datum idea service charge player parcel delivery business customer rate up fedex post office service service percent package day peak yearend holiday shipping season jindel
Name: lemmatized_text, dtype: object
```

Save the Cleaned data as a csv file (Recommended)

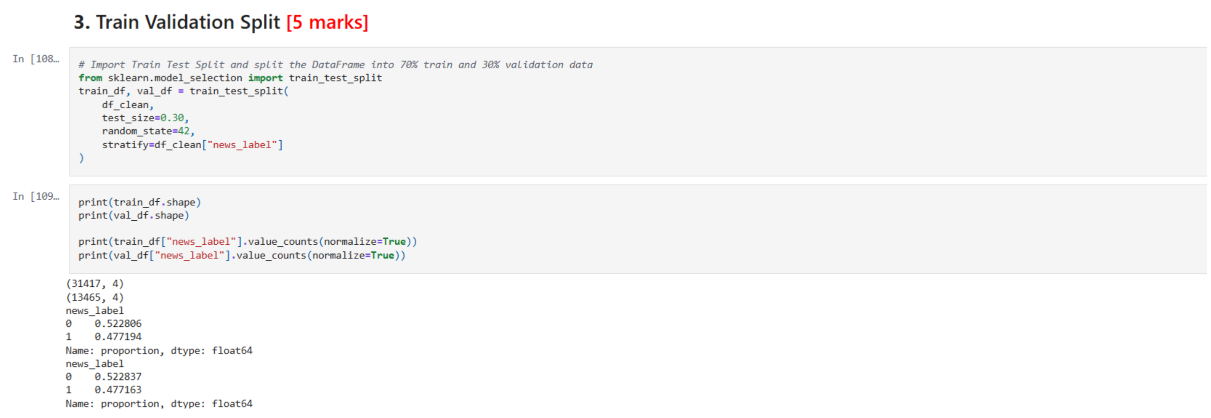
In [101…
```
## Recommended to perform the below steps to save time while rerunning the code
df_clean.to_csv("clean_df.csv", index=False)
df_clean = pd.read_csv("clean_df.csv")
```

df_clean.shape

(44898, 4)

44898 rows and 4 columns, after preprocessing text.

3. Train Validation Split (70 train - 30 val)



3. Train Validation Split [5 marks]

In [108…

```
# Import Train Test Split and split the DataFrame into 70% train and 30% validation data
from sklearn.model_selection import train_test_split
train_df, val_df = train_test_split(
    df_clean,
    test_size=0.30,
    random_state=42,
    stratify=df_clean["news_label"]
)
```

In [109…

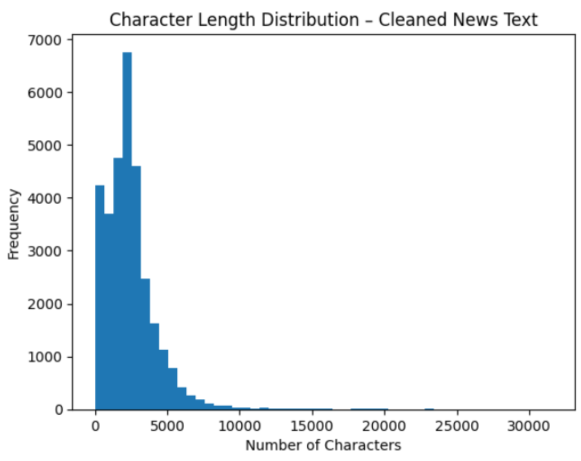
```
print(train_df.shape)
print(val_df.shape)

print(train_df["news_label"].value_counts(normalize=True))
print(val_df["news_label"].value_counts(normalize=True))
```

```
(31417, 4)
(13465, 4)
news_label
0    0.522806
1    0.477194
Name: proportion, dtype: float64
news_label
0    0.522837
1    0.477163
Name: proportion, dtype: float64
```

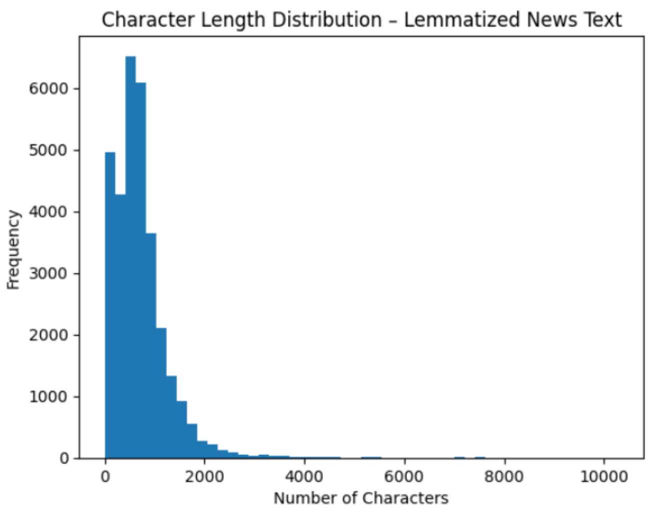
The train data set has 31417
rows and val dataset has 13465
rows respectively using 70-30 split.

4. Exploratory Data Analysis on Training Data

Character length distribution of cleaned news text:



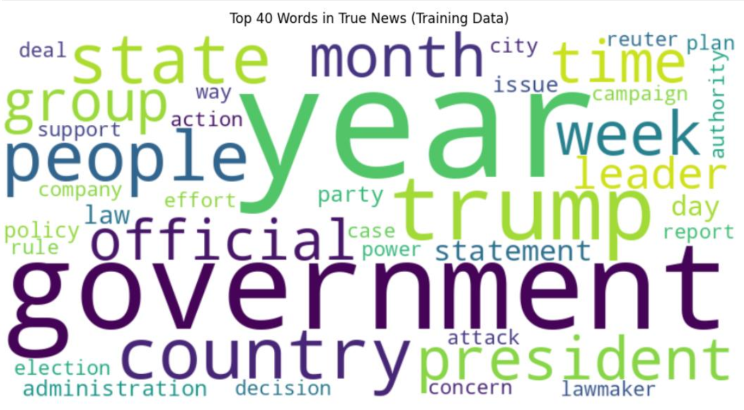
Character length distribution of lemmatized text:



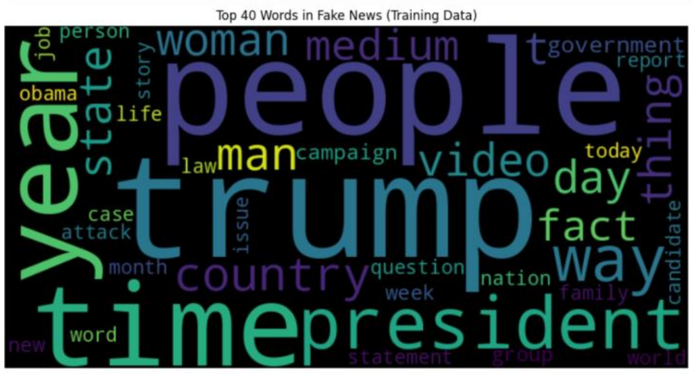
The histogram shows a right-skewed distribution where most lemmatized news articles fall between 200–1500 characters, with a few significantly longer outliers. This indicates consistent preprocessing and a generally balanced text length suitable for semantic embedding and model training.

Top 10 frequent ingredientss in Training dataset:

Top 40 Words in True News



Top 40 words in Fake news:



Top 10 unigrams in true news training data:

Top 10 Unigrams in True News:

trump: 25395

state: 14363

government: 13886

year: 13313

people: 10862

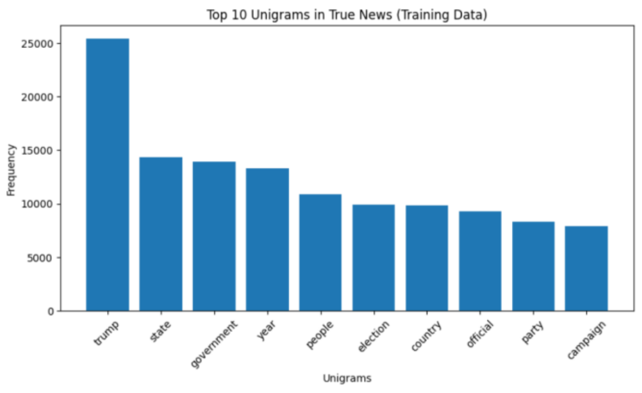
election: 9934

country: 9810

official: 9307

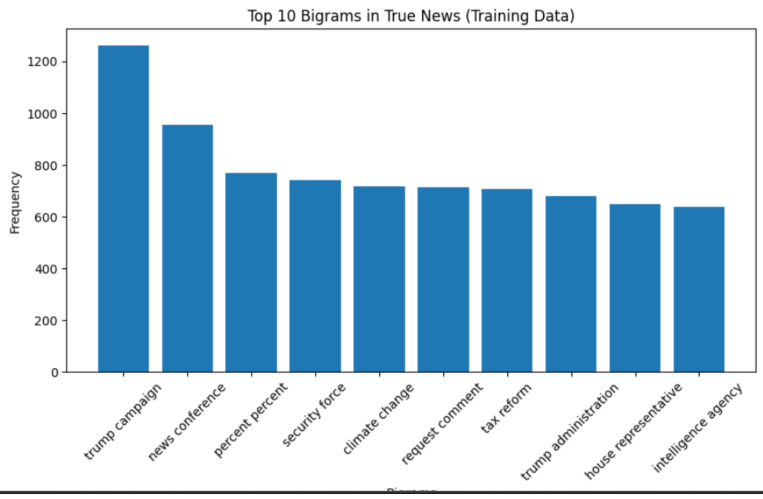
party: 8292

campaign: 7912



Top 10 bigrams in true news training data:

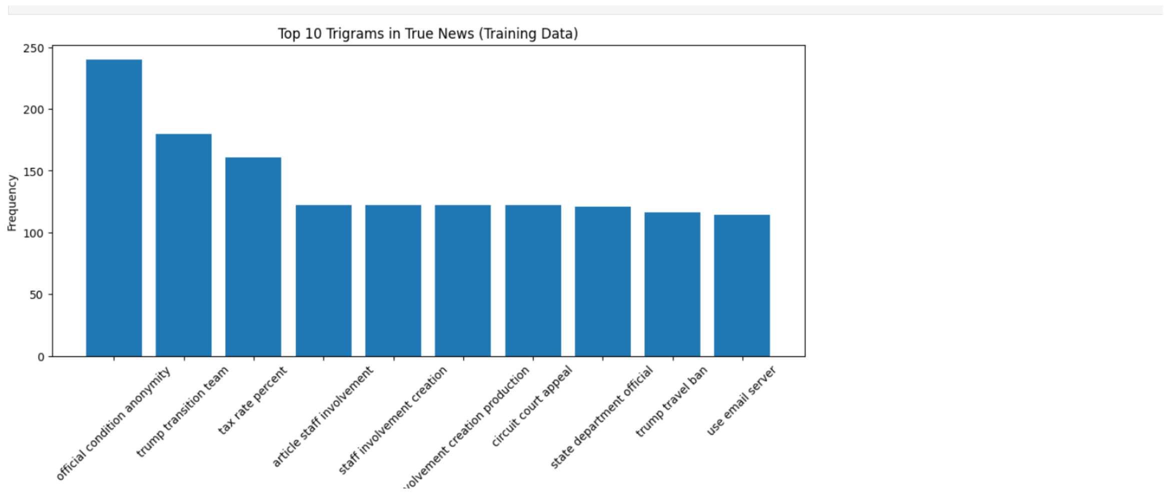
Top 10 Bigrams in True News:
('trump', 'campaign'): 1263
('news', 'conference'): 954
('percent', 'percent'): 768
('security', 'force'): 743
('climate', 'change'): 717
('request', 'comment'): 715
('tax', 'reform'): 708
('trump', 'administration'): 680
('house', 'representative'): 647
('intelligence', 'agency'): 639



top 10 trigrams by frequency in true news

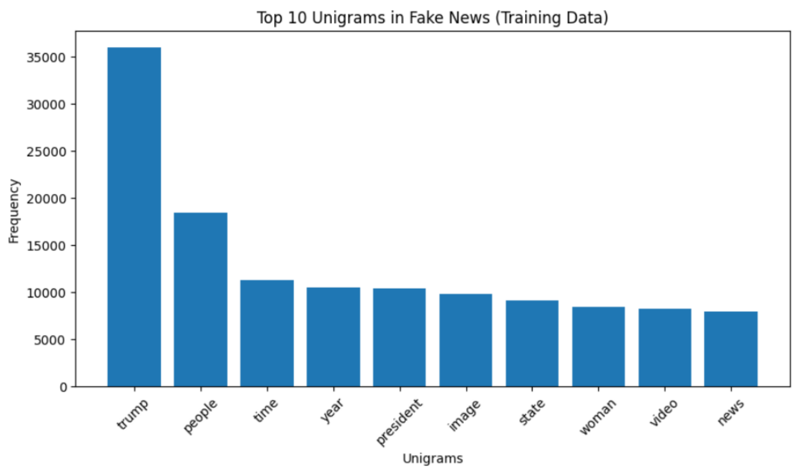
Top 10 Trigrams in True News:
('official', 'condition', 'anonymity'): 240
('trump', 'transition', 'team'): 180
('tax', 'rate', 'percent'): 161

('article', 'staff', 'involvement'): 122
('staff', 'involvement', 'creation'): 122
('involvement', 'creation', 'production'): 122
('circuit', 'court', 'appeal'): 122
('state', 'department', 'official'): 121
('trump', 'travel', 'ban'): 116
('use', 'email', 'server'): 114



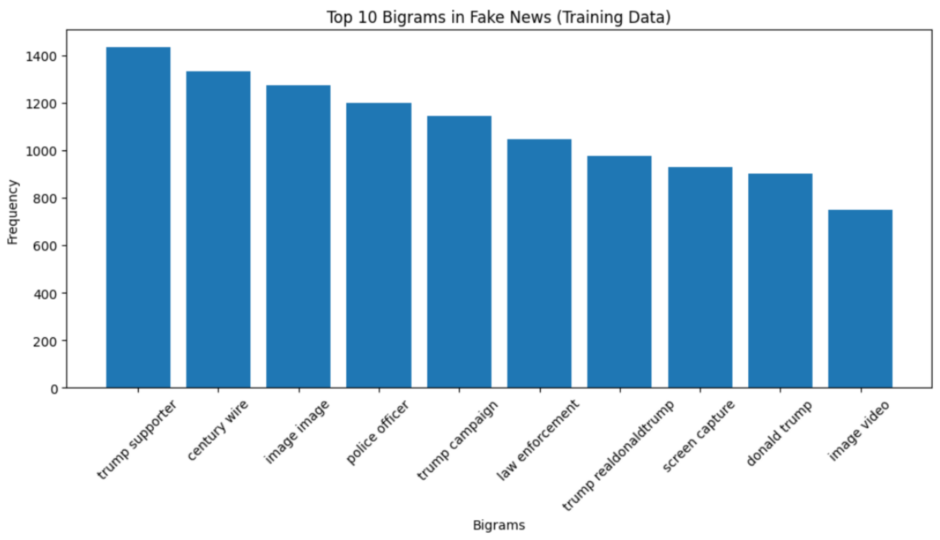
Top 10 Unigrams in Fake News:

trump: 35984
people: 18432
time: 11224
year: 10428
president: 10324
image: 9789
state: 9031
woman: 8398
video: 8152
news: 7903



Top 10 Bigrams in Fake News:

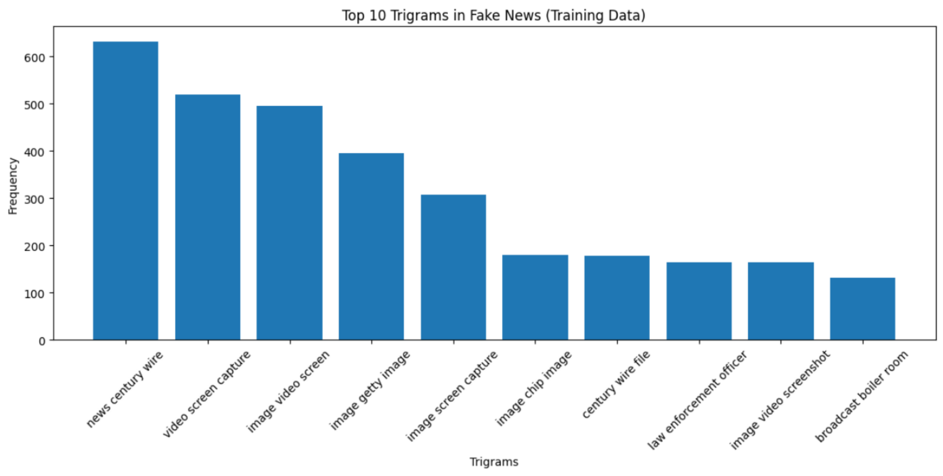
('trump', 'supporter'): 1434
('century', 'wire'): 1330
('image', 'image'): 1274
('police', 'officer'): 1200
('trump', 'campaign'): 1145
('law', 'enforcement'): 1046
('trump', 'realdonaldtrump'): 974
('screen', 'capture'): 928
('donald', 'trump'): 903
('image', 'video'): 750



Top 10 Trigrams in Fake News:

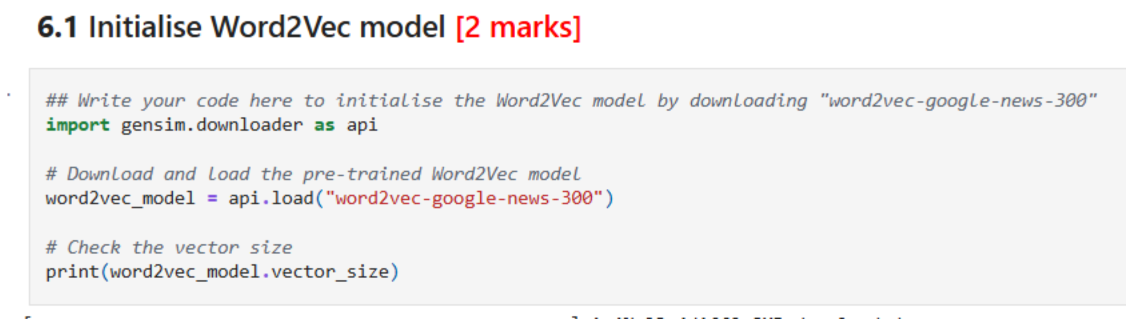
('news', 'century', 'wire'): 632
('video', 'screen', 'capture'): 520
('image', 'video', 'screen'): 495
('image', 'getty', 'image'): 396

('image', 'screen', 'capture'): 308
('image', 'chip', 'image'): 180
('century', 'wire', 'file'): 178
('law', 'enforcement', 'officer'): 165
('image', 'video', 'screenshot'): 164
('broadcast', 'boiler', 'room'): 131



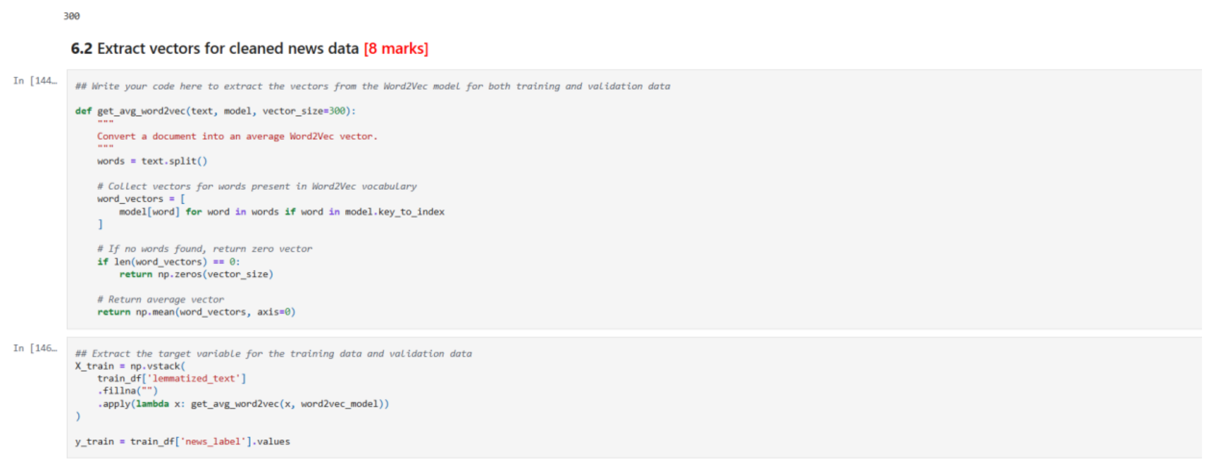
5. Feature Extraction

initialised the Word2Vec model by downloading "word2vec-google-news-300"



6.1 Initialise Word2Vec model [2 marks]

```
## Write your code here to initialise the Word2Vec model by downloading "word2vec-google-news-300"
import gensim.downloader as api

# Download and load the pre-trained Word2Vec model
word2vec_model = api.load("word2vec-google-news-300")

# Check the vector size
print(word2vec_model.vector_size)
```



```
300
```

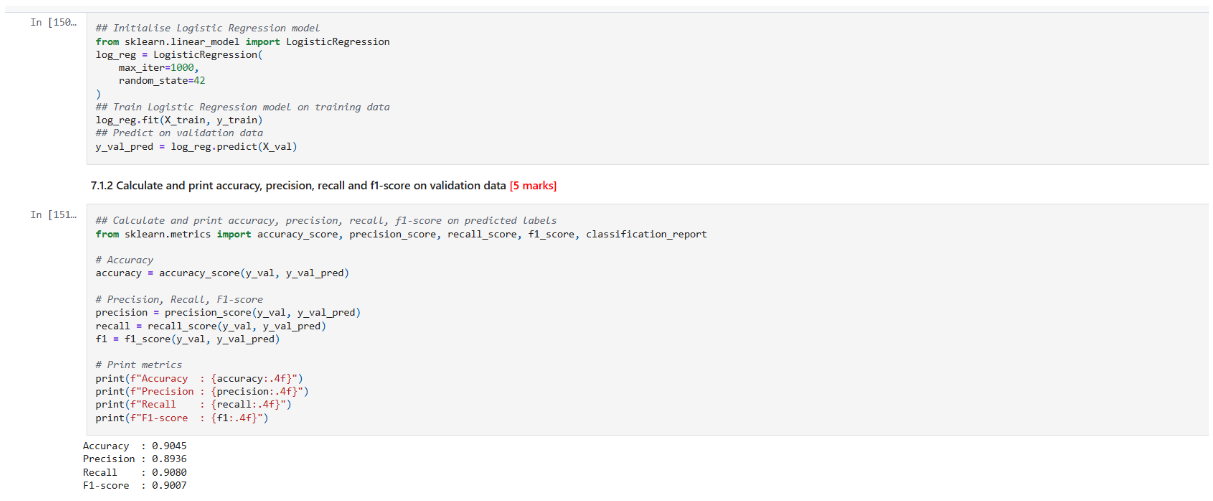
6.2 Extract vectors for cleaned news data [8 marks]

```
In [144…
## Write your code here to extract the vectors from the Word2Vec model for both training and validation data

def get_avg_word2vec(text, model, vector_size=300):
    """
    Convert a document into an average Word2Vec vector.
    """
    words = text.split()

    # Collect vectors for words present in Word2Vec vocabulary
    word_vectors = [
        model[word] for word in words if word in model.key_to_index
    ]

    # If no words found, return zero vector
    if len(word_vectors) == 0:
        return np.zeros(vector_size)

    # Return average vector
    return np.mean(word_vectors, axis=0)
```

```
In [146…
## Extract the target variable for the training data and validation data
X_train = np.vstack(
    train_df['lemmatized_text']
    .fillna("")
    .apply(lambda x: get_avg_word2vec(x, word2vec_model))
)

y_train = train_df['news_label'].values
```

6. Model Building and Training

Logistic Regression Model



```
In [150…
## Initialise Logistic Regression model
from sklearn.linear_model import LogisticRegression
log_reg = LogisticRegression(
    max_iter=1000,
    random_state=42
)
## Train Logistic Regression model on training data
log_reg.fit(X_train, y_train)
## Predict on validation data
y_val_pred = log_reg.predict(X_val)
```

7.1.2 Calculate and print accuracy, precision, recall and f1-score on validation data [5 marks]

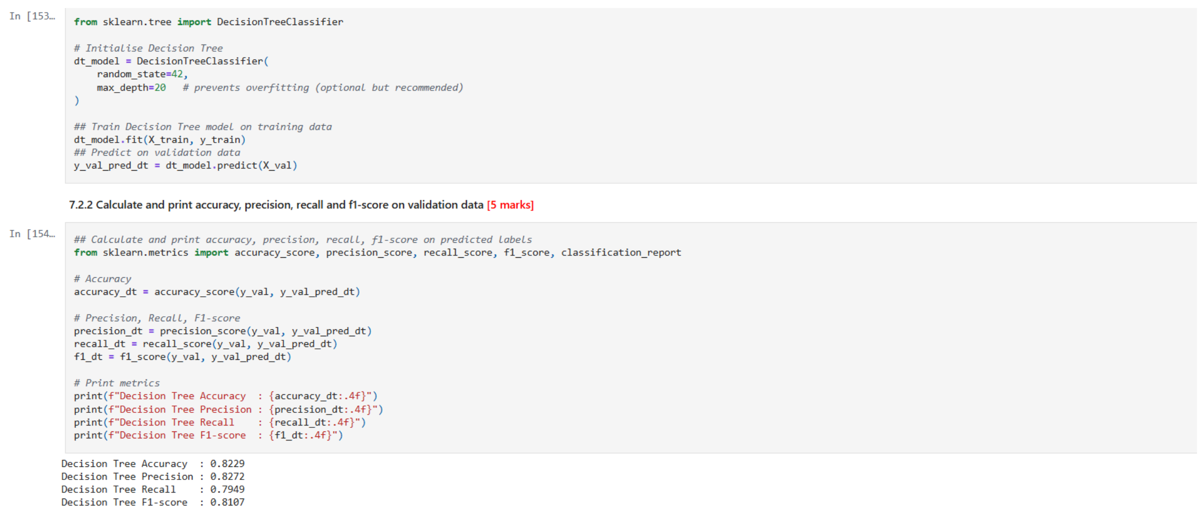
```
In [151…
## Calculate and print accuracy, precision, recall, f1-score on predicted labels
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, classification_report

# Accuracy
accuracy = accuracy_score(y_val, y_val_pred)

# Precision, Recall, F1-score
precision = precision_score(y_val, y_val_pred)
recall = recall_score(y_val, y_val_pred)
f1 = f1_score(y_val, y_val_pred)

# Print metrics
print(f"Accuracy  : {accuracy:.4f}")
print(f"Precision : {precision:.4f}")
print(f"Recall    : {recall:.4f}")
print(f"F1-score  : {f1:.4f}")
```

```
Accuracy  : 0.9045
Precision : 0.8936
Recall    : 0.9080
F1-score  : 0.9007
```

```
         precision    recall  f1-score   support
```

```
   Fake News       0.91      0.90      0.91      7040
  True News       0.89      0.91      0.90      6425

   accuracy                           0.90     13465
  macro avg       0.90      0.90      0.90     13465
weighted avg       0.90      0.90      0.90     13465
```

Decision Tree Model



```
In [153…
from sklearn.tree import DecisionTreeClassifier

# Initialise Decision Tree
dt_model = DecisionTreeClassifier(
    random_state=42,
    max_depth=20   # prevents overfitting (optional but recommended)
)

## Train Decision Tree model on training data
dt_model.fit(X_train, y_train)
## Predict on validation data
y_val_pred_dt = dt_model.predict(X_val)
```

7.2.2 Calculate and print accuracy, precision, recall and f1-score on validation data [5 marks]

```
In [154…
## Calculate and print accuracy, precision, recall, f1-score on predicted labels
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, classification_report

# Accuracy
accuracy_dt = accuracy_score(y_val, y_val_pred_dt)

# Precision, Recall, F1-score
precision_dt = precision_score(y_val, y_val_pred_dt)
recall_dt = recall_score(y_val, y_val_pred_dt)
f1_dt = f1_score(y_val, y_val_pred_dt)

# Print metrics
print(f"Decision Tree Accuracy  : {accuracy_dt:.4f}")
print(f"Decision Tree Precision : {precision_dt:.4f}")
print(f"Decision Tree Recall    : {recall_dt:.4f}")
print(f"Decision Tree F1-score  : {f1_dt:.4f}")
```

```
Decision Tree Accuracy  : 0.8229
Decision Tree Precision : 0.8272
Decision Tree Recall    : 0.7949
Decision Tree F1-score  : 0.8107
```

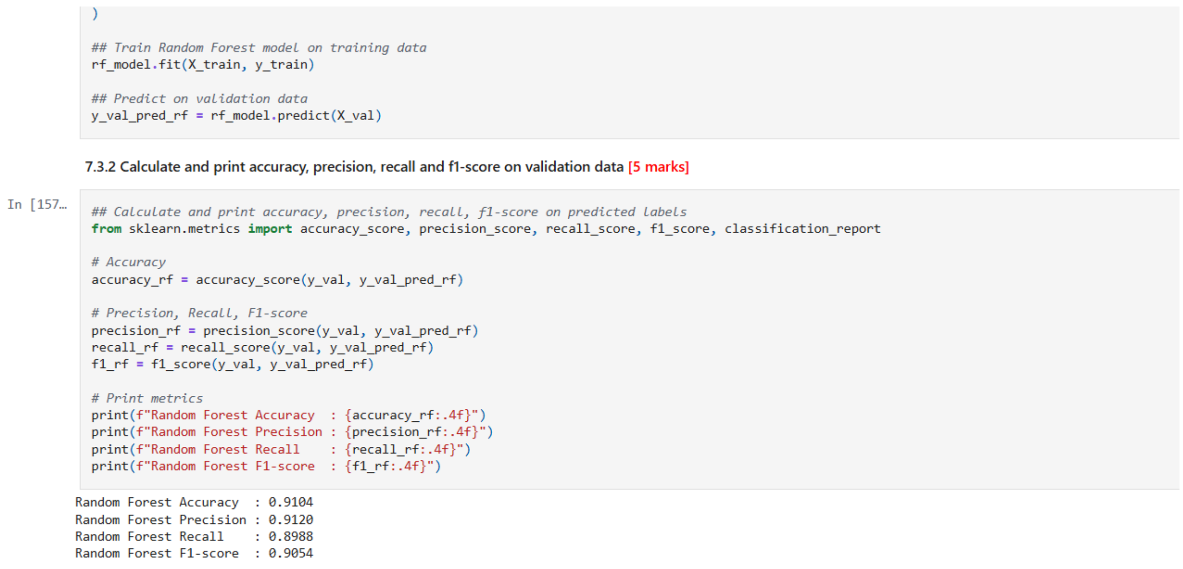
```
Decision Tree Accuracy  : 0.8229
Decision Tree Precision : 0.8272
Decision Tree Recall    : 0.7949
Decision Tree F1-score  : 0.8107
```

Classification report:

```
              precision    recall  f1-score   support

   Fake News       0.82      0.85      0.83      7040
```

```
  True News     0.83     0.79     0.81     6425


   accuracy                       0.82    13465
  macro avg     0.82     0.82     0.82    13465
weighted avg     0.82     0.82     0.82    13465
```

Random Forest Model



```
)

## Train Random Forest model on training data
rf_model.fit(X_train, y_train)

## Predict on validation data
y_val_pred_rf = rf_model.predict(X_val)
```

7.3.2 Calculate and print accuracy, precision, recall and f1-score on validation data [5 marks]

In [157…

```
## Calculate and print accuracy, precision, recall, f1-score on predicted labels
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, classification_report

# Accuracy
accuracy_rf = accuracy_score(y_val, y_val_pred_rf)

# Precision, Recall, F1-score
precision_rf = precision_score(y_val, y_val_pred_rf)
recall_rf = recall_score(y_val, y_val_pred_rf)
f1_rf = f1_score(y_val, y_val_pred_rf)

# Print metrics
print(f"Random Forest Accuracy  : {accuracy_rf:.4f}")
print(f"Random Forest Precision : {precision_rf:.4f}")
print(f"Random Forest Recall    : {recall_rf:.4f}")
print(f"Random Forest F1-score  : {f1_rf:.4f}")
```

```
Random Forest Accuracy  : 0.9104
Random Forest Precision : 0.9120
Random Forest Recall    : 0.8988
Random Forest F1-score  : 0.9054
```

```
              precision    recall  f1-score   support


  Fake News       0.91      0.92      0.91      7040
  True News       0.91      0.90      0.91      6425


   accuracy                           0.91     13465
  macro avg       0.91      0.91      0.91     13465
weighted avg       0.91      0.91      0.91     13465
```

7. Conclusion:

This study demonstrates that semantic classification using Word2Vec embeddings is highly effective for fake news detection. Random Forest emerged as the best-performing model with 91% accuracy and F1-score, showing strong balance between precision and recall. Overall, leveraging semantic representations significantly improved the model's ability to distinguish between true and fake news, highlighting the practical value of NLP-based approaches in combating misinformation.