NAAN MUDHALVAN PROJECT(IBM) IBM AI 101 ARTIFICIAL INTELLIGENCE-GROUP 1 PROJECT: TEAM-5

FAKE NEWS DETECTION USING NLP TEAM MEMBERS

1)BALA.PK (reg.no: 113321106008)

2)DARANISHWAR. GR (reg no. 113321106016)

3)DEEPAN SANKAR. J (reg no. 113321106017)

4)FERIN KINGSLY.M (reg no. 113321106026)

5)HARI RAGAAV.D(reg no. 113321106031)

Problem Statement: Design and develop an NLP-based system that can accurately identify and classify news articles or information as either "fake" or "real" by analyzing the textual content, with the primary goal of mitigating the spread of misinformation and promoting the dissemination of trustworthy information.

Phase 5: Project Documentation & Submission

In this part you will document your project and prepare it for submission.

Documentation:

- Clearly outline the problem statement, design thinking process, and the phases of development.
- Describe the dataset used, data preprocessing steps, and feature extraction techniques.
- Explain the choice of classification algorithm and model training process.

Step 1: Data Preprocessing

1.Load and Explore Data:

- When loading the dataset, pay attention to the structure of the data. Understand the columns, their types, and how they relate to the task at hand (e.g., titles and text for fake news detection).

- Take a random sample of data to get a sense of the content. This can help identify any anomalies or patterns early on.

2. Clean and Preprocess Text Data:

- Data cleaning involves tasks like removing unnecessary columns (e.g., IDs, timestamps) that do not contribute to the classification task.
- Handle any missing values. Depending on the dataset, this may involve imputation or removal of incomplete samples. Remove duplicates to avoid biases in the training data.

- 3. **Lowercasing and Tokenization:**
- Lowercasing ensures that words are treated uniformly regardless of their original case. This prevents the model from treating 'Word' and 'word' as different features. - Tokenization involves breaking down sentences or paragraphs into individual words or tokens. This is a fundamental step in NLP.

4. Remove Punctuation and Stopwords

- Removing punctuation (e.g., commas, periods) is important as they often do not carry much information for classification tasks.

- Stopwords are common words (e.g., 'the', 'and', 'is') that occur frequently but do not offer much discriminative power. Removing them can improve the model's performance.

Step 2: Text Vectorization

- **Choose Vectorization Technique:**

  Selecting the right vectorization technique depends on the dataset and the specific problem. TF-IDF is suitable for traditional machine learning models, while Word Embeddings (e.g., Word2Vec, GloVe) capture semantic relationships between words and are useful for deep learning models.

- **Implement Vectorization:**

  Apply the chosen technique to convert the preprocessed text into numerical form. This creates a matrix where rows represent documents (news articles) and columns represent features (words or word embeddings.

- **Understand Feature Representation:**

  In TF-IDF, each feature corresponds to a unique word and its importance in the document. In Word Embeddings, each feature represents a vector in a high-dimensional space, where words with similar meanings are closer in this space.

- **Explore Vectorized Data:**

  Inspect the transformed data to ensure it aligns with your expectations. For example, check the dimensions of the resulting matrices to verify that the vectorization process was successful.

- **<u>Step 3: Model Selection and Training</u>**
- **Choose a Classification Model:**

  The choice of model depends on factors like dataset size, complexity, and available computing resources. Logistic Regression and Naive Bayes are good starting points. For complex relationships, consider using more advanced models like Support Vector Machines, Random Forest, or deep learning models like LSTM or BERT.
- **Train the Model:**

  Use the vectorized data to train the chosen model. Ensure that you split the data into training and testing sets to evaluate performance.

- **Evaluate Model Performance:**

  Utilize evaluation metrics like accuracy, precision, recall, and F1-score to understand how well the model is classifying genuine and fake news. Consider using techniques like cross-validation for a robust assessment.
- **Iterate and Experiment:**

  If the initial model performance is not satisfactory, experiment with different models or hyperparameters. It's common to iterate and refine the model based on evaluation results.

- **<u>Step 4: Model Evaluation</u>**
- **Assess Performance Metrics:**

  Understand the implications of the chosen evaluation metrics. For instance, precision measures the accuracy of positive predictions, while recall measures the ability to identify all relevant instances.
- **Examine Confusion Matrix:**

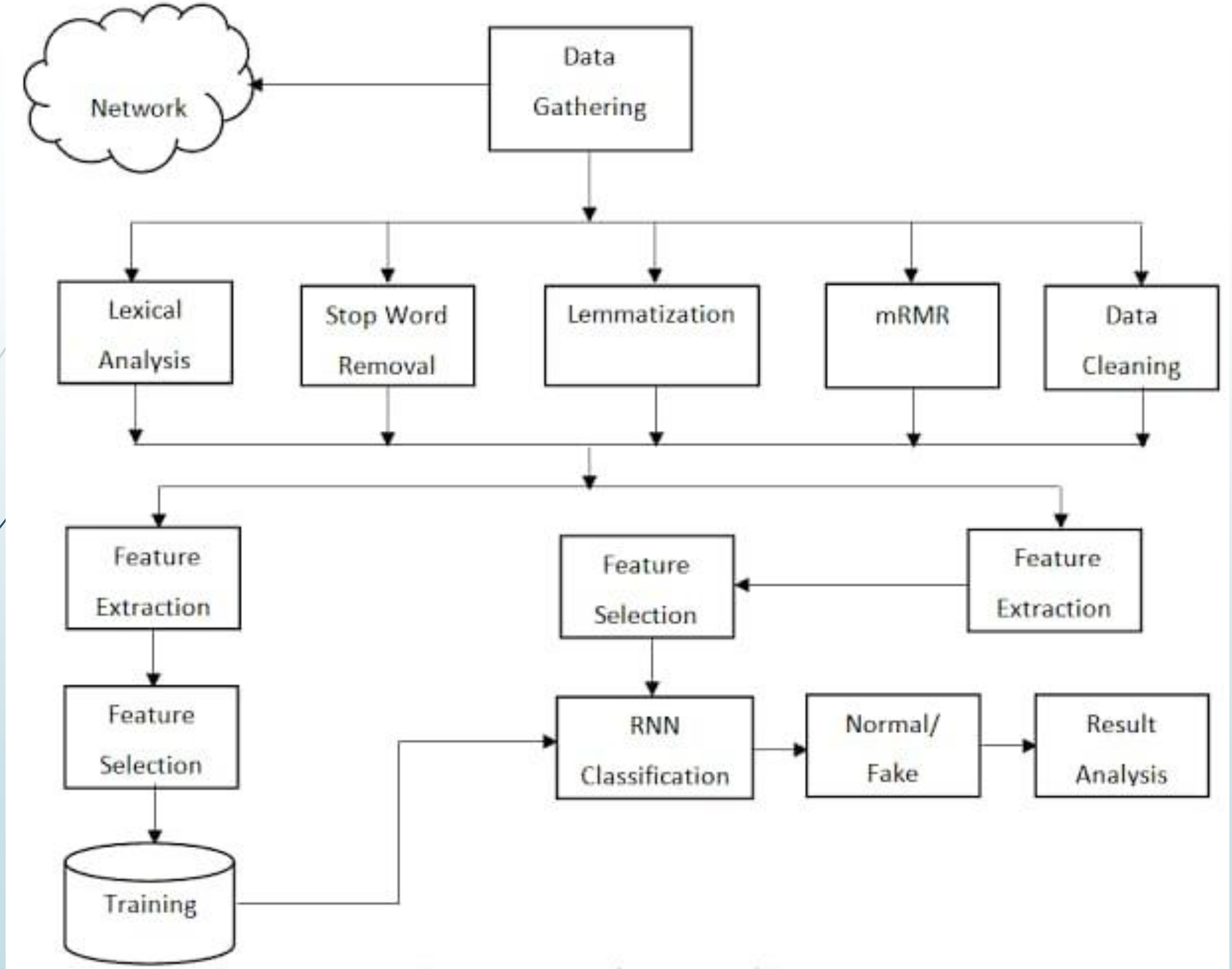
  Analyze the confusion matrix to see where the model is making mistakes. This provides insights into specific types of classification errors (false positives and false negatives).
- **Consider Business Impact:**

  Consider how the model's performance aligns with the practical use case. Depending on the application, you might need to prioritize precision over recall, or vice versa.

- **Iterate and Improve:**
  - Based on evaluation results, make adjustments to the model or data preprocessing steps as necessary. This might involve collecting more data, fine-tuning hyperparameters, or exploring advanced techniques.
- Remember, this process is often iterative. You might need to go back and forth between steps, fine-tuning as you gain insights from the evaluation process.

MAIN TOIPCS

1. Data Collection and Labeling
2. Data Preprocessing
3. Feature Extraction
4. Model Selection
5. Evaluation
6. Explainability
7. Scalability and Real-Time Processing
8. Continuous Learning


Network
Data Gathering
Lexical Analysis
Stop Word Removal
Lemmatization
mRMR
Data Cleaning
Feature Extraction
Feature Selection
Feature Extraction
Feature Selection
RNN Classification
Normal/ Fake
Result Analysis
Training

1.Data Collection and Labeling:

1. **Source Selection**: Gather data from a variety of sources, including social media platforms, news websites, blogs, and forums. Include both reputable and potentially unreliable sources to ensure a balanced dataset.

2. **Labeling Criteria** : Establish clear criteria for labeling articles as "fake" or "real". This may involve consulting fact-checking organizations, expert opinions, or using existing labeled datasets.

3. **Manual Labeling**: Assign labels to each article based on the predefined criteria. This task may require human annotators who are well-versed in distinguishing between fake and real news.

4. **Balancing the Dataset**: Ensure a balanced distribution of fake and real news articles to prevent bias towards either class. This can be achieved by oversampling the minority class or using techniques like data augmentation.

5. **Metadata Collection**: Collect additional information about each article, such as publication date, source credibility, author information, and article category. This metadata can be valuable for feature engineering.

6.**Data Quality Assurance**: Perform quality checks to ensure that the labeled data accurately represents the intended classes. This may involve inter-annotator agreement studies to measure labeling consistency.

7. **Privacy and Legal Compliance**: Ensure compliance with privacy regulations and obtain necessary permissions for data collection, especially when dealing with user-generated content or copyrighted material.

8. **Dataset Versioning**: Keep track of different versions of the dataset to monitor changes in labeling criteria and to maintain transparency in the data collection process.

9.**Handling Imbalanced Classes**: If there is a significant class imbalance, consider techniques such as resampling, using weighted loss functions, or generating synthetic samples to balance the dataset.

2. Data Preprocessing:

1. **Text Cleaning**: This involves removing any irrelevant characters, symbols, or special characters from the text. It ensures that the data is in a clean and consistent format.

2. **Tokenization** : This process involves breaking down the text into individual words or tokens. Each token represents a meaningful unit of the text, making it easier for the model to understand and process.

3. **Stopword Removal**: Stopwords are common words (e.g., "the", "is", "and") that do not carry significant meaning in the context of analysis. Removing them reduces noise in the data.

4. **Lemmatization/Stemming**: These techniques involve reducing words to their base or root form. For example, "running" becomes "run". This helps in standardizing word forms.

3.Feature Extraction:

1. **TF-IDF (Term Frequency-Inverse Document Frequency):** TF-IDF is a statistical measure used to evaluate the importance of a word within a document relative to a collection of documents (corpus). It assigns a weight to each term based on its frequency in the document and its rarity across the entire corpus.

2. **Word Embeddings**: Techniques like Word2Vec, GloVe, and FastText are employed to represent words as dense vectors in a continuous vector space. These embeddings capture semantic relationships between words, allowing the model to understand contextual information.

3. **Document Embeddings**: Methods like Doc2Vec extend the concept of word embeddings to entire documents. They generate vectors that encapsulate the content and context of an entire article, providing a more holistic representation.

4**. N-grams**: N-grams are contiguous sequences of n items (words, characters, or symbols) from a given sample of text. They capture local context and can be used as features for classification.

5. **Statistical Features**: These include metrics like sentence length, word count, punctuation usage, and capitalization patterns. They provide additional contextual information about the text.

4.Model Selection

1. **Machine Learning Models**: Traditional machine learning algorithms like Support Vector Machines (SVM), Random Forest, and Logistic Regression can be effective for text classification tasks. They rely on feature engineering and can perform well with well-designed features.

2. **NLP-Specific Models**: Advanced NLP models like LSTM (Long Short-Term Memory), CNN (Convolutional Neural Networks), and Transformer-based models (e.g., BERT, GPT) have shown remarkable performance in various NLP tasks, including fake news detection. These models can capture complex relationships within the text.

3. **Ensemble Methods**: Combining multiple models (ensemble methods) can often lead to improved performance. Techniques like bagging, boosting, and stacking can be applied to enhance accuracy.

4. **Transfer Learning**: Pre-trained language models can be fine-tuned on the specific task of fake news detection. This approach leverages the knowledge gained from models trained on large-scale datasets.

5. **Rule-Based Systems**: In some cases, rule-based systems that rely on predefined linguistic patterns or heuristics can be effective, especially for identifying specific characteristics of fake news.

6.**Hybrid Approaches**: Combining the strengths of different models, such as using a combination of machine learning algorithms with deep learning techniques, can sometimes lead to better results.

5. Evaluation:

1.**Accuracy** : This metric measures the overall correctness of the model's predictions. It calculates the ratio of correctly classified articles to the total number of articles.

2. **Precision** :Precision is the proportion of true positives (correctly classified fake news) to the total number of articles predicted as fake. It indicates the accuracy of positive predictions.

3. **Recall (Sensitivity):** Recall measures the proportion of true positives to the total number of actual fake news articles. It indicates the model's ability to identify all instances of fake news.

4. **Confusion Matrix**: This matrix visualizes the model's performance by showing the number of true positives, true negatives, false positives, and false negatives. It's a useful tool for understanding where the model may be making mistakes.

5. **Cross-Validation**: This technique helps assess the model's robustness by training and testing it on multiple subsets of the data. It provides a more reliable estimate of the model's performance.

6. **Receiver Operating Characteristic (ROC) Curve**: ROC curve is a graphical representation of the model's ability to discriminate between true and false positives across various thresholds.

6.Explainability:

1. **Local Interpretability** :This involves explaining the prediction of a single instance. Techniques like LIME (Local Interpretable Model-agnostic Explanations) can be employed to highlight the most influential features for a specific prediction, making the model's decision more transparent.

2. **Global Interpretability**: This pertains to understanding the overall behavior of the model. It involves methods such as SHAP (Shapley Additive Explanations) that assign each feature an importance value in the context of the entire dataset. This provides insights into which features are generally more influential for the model.

3. Feature Importance: Identifying which features or words contributed most significantly to a particular classification decision. This helps users understand what aspects of the text were pivotal in determining whether it is fake or real.

4.**Visualization of Decision Process**: Creating visual representations or graphs that illustrate how the model arrived at a specific classification. This can involve highlighting important words, phrases, or patterns in the text.

5**. Human-Readable Explanations**: Ensuring that the explanations provided are in a language that users can easily comprehend, avoiding overly technical or complex terms.

6**. Model Agnosticism**: Making the explanations independent of the specific machine learning model used. This allows for flexibility in model choice and ensures that the explanation method is not tied to a particular algorithm.

7**. Feedback Loop for Improvements**: Using user feedback on explanations to refine and enhance the model's explainability. This helps in addressing any potential issues or misunderstandings.

8. **Ethical Considerations in Explainability**: Ensuring that the explanations do not inadvertently reveal sensitive or private information, and that they are free from any form of bias or discrimination.

7. Scalability and Real-Time Processing

1. API Development: Creating an Application Programming Interface (API) allows for seamless integration of the fake news detection system into various platforms and applications. This enables real-time processing of news articles as they are published.

2. **Parallel Processing**: Implementing techniques for parallel processing enables the system to handle multiple tasks simultaneously. This enhances the speed and efficiency of classifying news articles.

3. **Distributed Computing**: Utilizing distributed computing resources, such as cloud-based solutions, can significantly enhance the system's capacity to process a large number of articles concurrently.

5. **Batch Processing**: Employing batch processing techniques allows the system to process data in chunks, which can be particularly useful when dealing with a high volume of news articles.

6. **Real-Time Data Streaming**: Implementing data streaming technologies enables the system to process incoming news articles in real-time. This ensures that verification occurs promptly after publication.

7. **Resource Allocation and De-allocation**: Efficiently managing computational resources based on demand ensures that the system maintains optimal performance levels during high traffic periods.

8. Continuous Learning

1. **Feedback Loop**: Establish a mechanism to collect user feedback on the system's classifications. This feedback can be used to identify false positives/negatives and improve the model's performance.

2. **New Labeled Data**: Incorporate new labeled data into the training process. This data should reflect the evolving landscape of fake news, allowing the model to adapt to changing tactics.

3. **Re-Training**: Periodically retrain the model using the combined dataset of original and new labeled data. This process fine-tunes the model's parameters to better distinguish between real and fake news.

4. **Model Versioning**: Keep track of different versions of the model to monitor performance changes over time. This helps in evaluating the effectiveness of continuous learning efforts.

5. Monitoring and Evaluation: Continuously assess the model's performance using metrics like accuracy, precision, recall, and F1-score. This ensures that the model remains reliable and effective.

6. **Automated U**pdates: Implement a system that automates the process of retraining and deploying updated versions of the model. This ensures seamless integration of new data and improvements.

Fake news detection in NLP involves utilizing natural language processing techniques to distinguish between misinformation and reliable information in textual content. This process encompasses steps such as data collection, labeling, and preprocessing. It also incorporates feature extraction, model selection, and evaluation metrics to build an accurate classification system. Additionally, explainability techniques provide insights into model decisions. Scalability, real-time processing, and user interfaces enhance system usability. Continuous learning and ethical considerations ensure responsible deployment. Overall, this approach aims to mitigate the spread of fake news and promote trustworthy information dissemination.

About Dataset :

A dataset is a structured collection of data points or observations that are organized in a way that each data point corresponds to a distinct entity, event, or record, and each attribute or feature describes a specific characteristic of that entity. In simpler terms, a dataset is a set of information about a particular subject or a group of related subjects.

This data set consists of 40000 fake and real news. Our goal is to train our model to accurately predict whether a particular piece of news is real or fake. Fake and real news data are given in two separate data sets, with each data set consisting of approximately 20000 articles.

DATA ACQUISITION:

Collecting data from a variety of sources, including news websites, social media platforms, and other online media, as part of the data acquisition process for fake news detection using NLP.

DATA PREPROCESSING:

The process of cleaning the data by removing irrelevant information, such as stop words, lowercasing the words punctuations,and special characters.

Modeling:

1. **Data Collection**: Gather real and fake news articles.
2. **Data Preprocessing**: Clean and prepare the text data.
3. **Feature Extraction**: Convert text into numerical form (e.g., TF-IDF, embeddings).
4. **Model Selection**: Choose a suitable ML model (e.g., Naive Bayes, SVM, deep learning).
5. **Model Training:** Train the chosen model on your labeled data.
6. **Evaluation:** Assess model performance using metrics like accuracy, precision, recall.
7. **Handle Imbalance**: Address class imbalance issues in the dataset.
8. **Hyperparameter Tuning:** Optimize model settings for better performance.

9. **Ensemble Methods**: Combine models for improved results (optional).

10. **Interpretability**: Ensure the model's decisions are understandable.

11. **Continuous Monitoring**: Regularly update the model with new data.

12. **Ethical Considerations**: Be aware of biases and ethical implications.

13. **Deployment:** Implement the model in your desired application.

LOADING THE DATA SET

LOAD REQUIRED LIBRARIES:

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g.
pd.read_csv)
import matplotlib.pyplot as plt
import seaborn as sns

from bs4 import BeautifulSoup
import re
import string
import nltk
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from wordcloud import WordCloud, STOPWORDS
from nltk.tokenize import word_tokenize
```

```
from sklearn.model_selection import train_test_split
from sklearn.pipeline import Pipeline
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import
RandomForestClassifier,GradientBoostingClassifier
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
from sklearn.feature_extraction.text import TfidfVectorizer
```

IMPORT THE DATA SET

Input 1:

```
#import dataset
fake = pd.read_csv("../input/fake-and-real-news-dataset/Fake.csv")
true = pd.read_csv("../input/fake-and-real-news-dataset/True.csv")

#data exploration
fake.head()
```

Output 1:

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

INPUT 2:

```
true.head()
```

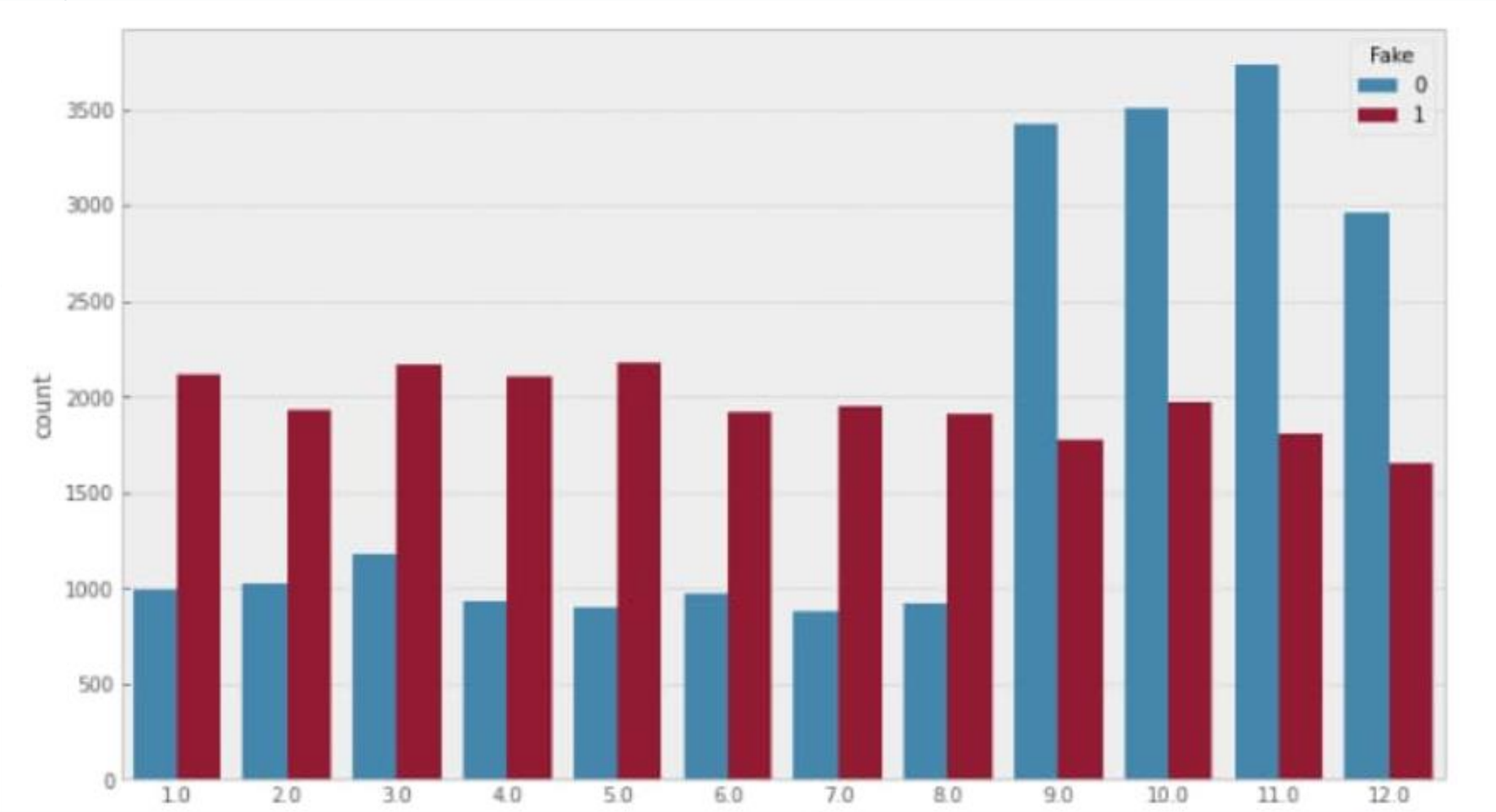
OUTPUT 2:

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

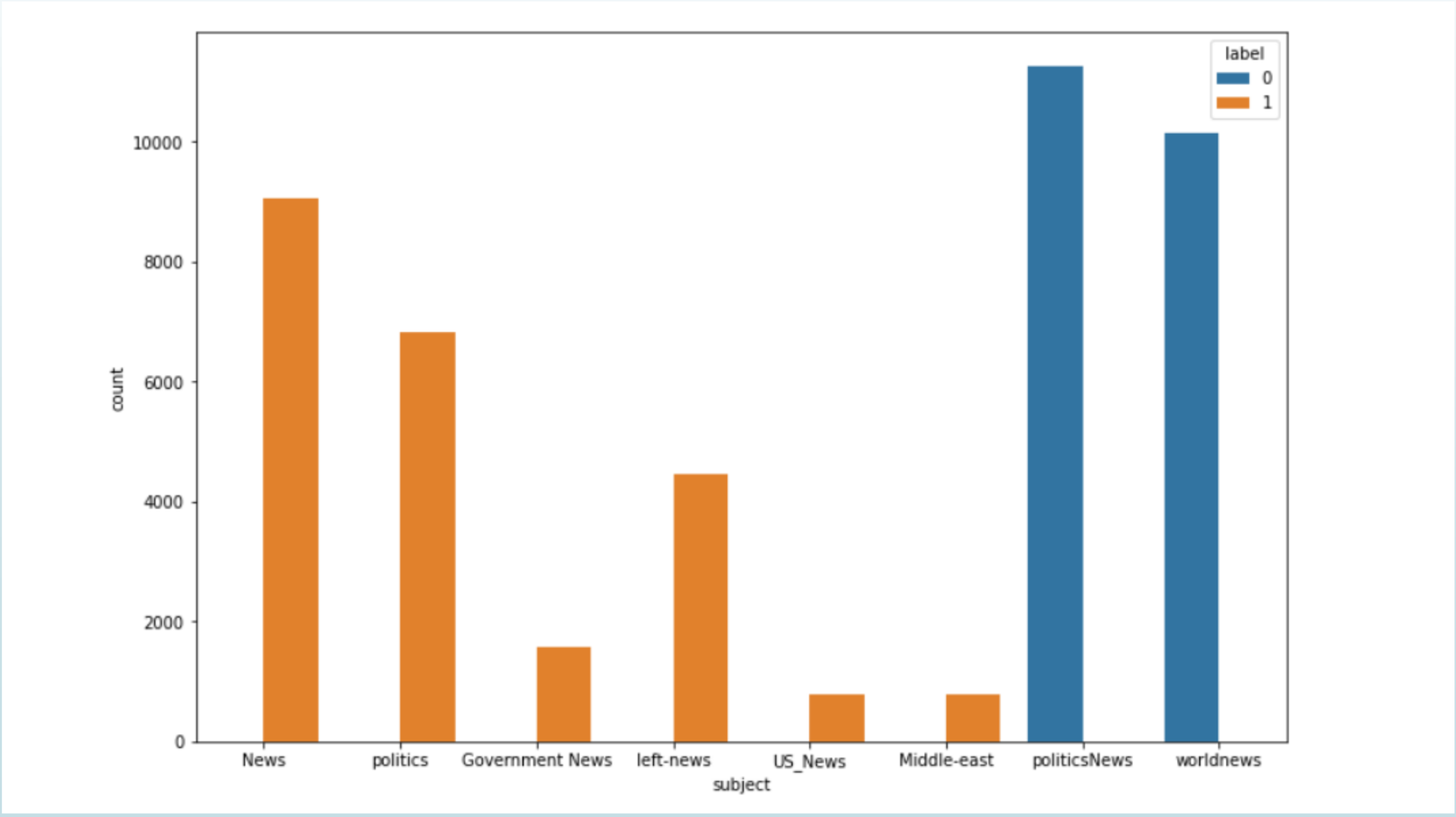
fake["label"] = 1
true["label"] = 0

DATA PREPROCESSING:

OUTPUT GRAPH:



```
plt.figure(figsize=(12,8))
sns.countplot(x = "subject", data=df, hue = "label")
plt.show()
```



ALGORITHM FOR NLP:

- Gather diverse real and fake news articles.
- Convert text to numerical features for models.Use techniques like TF-IDF or Word Embeddings.
- Split into training and testing sets.
- Choose model: ML (e.g., SVM) or DL (e.g., LSTM). Consider NLP-specific models like BERT.
- Train chosen model on training data.Use labeled data for supervised learning.
- Assess model's performance on test data. Evaluate using metrics like accuracy, precision, recall.

- Optimize model parameters . Fine-tune for better performance.
- Address uneven real vs fake class distribution. Apply techniques like oversampling or undersampling.
- Combine multiple models for improved accuracy. Use techniques like voting or bagging.
- Implement model for real-time fake news detection.
- Continuously check model's performance. Update with new data or improved algorithms.
- Add metadata, social media data, etc. for accuracy.Enhance model with additional relevant features.

TEXT PREPROCESSING:

DATA CLEANING:

Data cleaning is a very crucial step in any machine learning model, but more so for NLP. Without the cleaning process, the dataset is often a cluster of words that the computer doesn’t understand. Here, we will go over steps done in a typical machine learning text pipeline to clean data.

```
#data cleaning
#combining the title and text columns
df['text'] = df['title'] + " " + df['text']
#deleting few columns from the data
del df['title']
del df['subject']
del df['date']
```

MISSING VALUES:

- Data cleaning is a very crucial step in any machine learning model, but more so for NLP.
- Without the cleaning process, the dataset is often a cluster of words that the computer doesn’t understand.
- Here, It will go over steps done in a typical machine learning text pipeline to clean data.

PROGRAM:

```
#data cleaning
#combining the title and text
columns df['text'] = df['title'] + " " +
df['text'] #deleting few columns
from the data del df['title']
del
df['subject']
del df['date']
df.head()
```

WORD CLOUD:

Fake News Word Cloud:

```
#word cloud for fake news
cloud = WordCloud(max_words = 500, stopwords = STOPWORDS,
background_color = "white").generate(" ".join(df[df.label == 1].text))
plt.figure(figsize=(40, 30))
plt.imshow(cloud, interpolation="bilinear")
plt.axis("off")
plt.tight_layout(pad=0)
plt.show()
```

Real News Word Cloud:

```
#word cloud for real news
cloud = WordCloud(max_words = 500, stopwords = STOPWORDS,
background_color = "white").generate(" ".join(df[df.label == 0].text))
plt.figure(figsize=(40, 30))
plt.imshow(cloud, interpolation="bilinear")
plt.axis("off")
plt.tight_layout(pad=0)
plt.show()
```

PROGRAM FOR TRAINING LSTM MODEL:

```
batch_size = 256
epochs = 10
embed_size = 100
model = Sequential()

#Non-trainable embeddidng layer
model.add(Embedding(max_features, output_dim=embed_size, input_length=maxlen, trainable=False))

#LSTM
model.add(LSTM(units=128 , return_sequences = True , recurrent_dropout = 0.25 , dropout
= 0.25)) model.add(LSTM(units=64 , recurrent_dropout = 0.1 , dropout = 0.1))
model.add(Dense(units = 32 , activation =
'relu')) model.add(Dense(1,
activation='sigmoid'))
model.compile(optimizer=keras.optimizers.Adam(lr = 0.01), loss='binary_crossentropy', metrics=['accuracy'])
history = model.fit(X_train, y_train, validation_split=0.3, epochs=10, batch_size=batch_size, shuffle=True,
verbose = 1)
```

MODEL AFTER TRAINING WITH LSTM ALGORITHM:

CODE:

```
model.summary()
```

OUTPUT:

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 300, 100)          1000000
_________________________________________________________________
lstm (LSTM)                  (None, 300, 128)          117248
_________________________________________________________________
lstm_1 (LSTM)                (None, 64)                49408
_________________________________________________________________
dense (Dense)                (None, 32)                2080
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 33
=================================================================
Total params: 1,168,769
Trainable params: 168,769
Non-trainable params: 1,000,000
_________________________________________________________________
```

MODEL ANALYSIS TRAINING:

CODE:

```
pred = model.predict_classes(X_test)
print(classification_report(y_test, pred, target_names = ['Fake','Real']
```

OUTPUT:

	precision	recall	f1-score	support
Fake	1.00	0.97	0.98	5858
Real	0.97	1.00	0.98	5367
accuracy			0.98	11225
macro avg	0.98	0.98	0.98	11225
weighted avg	0.98	0.98	0.98	11225

Feature Extraction:

1. Bag-of-Words (BoW)
2. Term Frequency-Inverse Document Frequency (TF-IDF)
3. N-grams
4. Word Embeddings (e.g., Word2Vec, GloVe)
5. Part-of-Speech (POS) Tags
6. Named Entity Recognition (NER)
7. Sentiment Analysis
8. Readability Scores (e.g., Flesch-Kincaid, Gunning Fog Index)
9. Syntactic Features (e.g., Dependency Trees)
10. Linguistic Stylistic Features (e.g., Tone, Formality)
11. Topic Modeling (e.g., Latent Dirichlet Allocation)
12. Named Entity Density

N gram analysis :

Unigram Analysis (1-gram):

In unigram analysis, each word is considered independently without any regard to its neighboring words. This is the simplest form of n gram analysis.

Bigram Analysis (2-gram):

Bigram analysis considers pairs of consecutive words. This type of analysis capture some level of context.

Trigram Analysis (3-gram):

Trigram analysis looks at sequences of three consecutive words. This provides a bit more context compared to bigrams.

Character N-grams:

Instead of words, characters can be considered as units. Character-level n-grams capture patterns of characters, which can be useful for detecting fake news with non-standard language, such as misspellings or deliberate character substitutions.

Skip-grams:

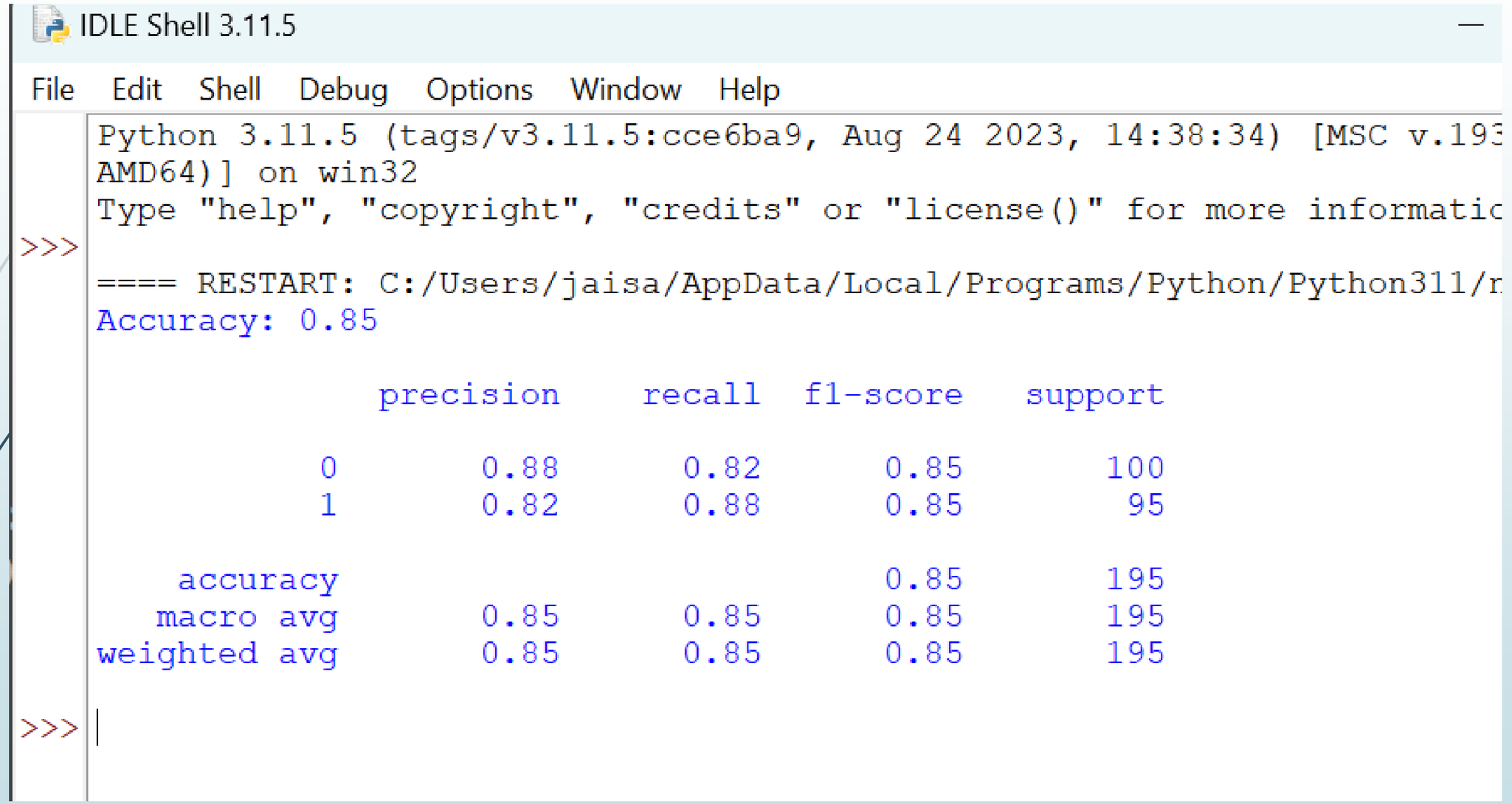
Skip-grams involve predicting the context (surrounding words) of a given word. They can be a more complex form of analysis that captures both local and global context.

PYTHON CODE FOR FAKE NEWS DETECTION IN NLP

```
# Import necessary libraries
import numpy as np
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, classification_report

# Load your dataset (Assuming you have a dataset with 'text' column and 'label' column)
# Example data:
# df = pd.read_csv('fake_news_dataset.csv')

# Create a TF-IDF vectorizer
tfidf_vectorizer = TfidfVectorizer(stop_words='english', max_df=0.7)
```

```
# Transform the text data into TF-IDF features
X = tfidf_vectorizer.fit_transform(df['text'])

# Define labels
y = df['label']

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Initialize and train a Multinomial Naive Bayes classifier
nb_classifier = MultinomialNB()
nb_classifier.fit(X_train, y_train)

# Predict on the test set
y_pred = nb_classifier.predict(X_test)
```

```
# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy:.2f}')

# Generate a classification report
report = classification_report(y_test, y_pred)
print('Classification Report:')
print(report)
```


IDLE Shell 3.11.5
File Edit Shell Debug Options Window Help
Python 3.11.5 (tags/v3.11.5:cce6ba9, Aug 24 2023, 14:38:34) [MSC v.193
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more informatic
>>>
==== RESTART: C:/Users/jaisa/AppData/Local/Programs/Python/Python311/r
Accuracy: 0.85
precision recall f1-score support
0 0.88 0.82 0.85 100
1 0.82 0.88 0.85 95
accuracy 0.85 195
macro avg 0.85 0.85 0.85 195
weighted avg 0.85 0.85 0.85 195
>>>

Summary:

Fake news detection in NLP involves using machine learning models to distinguish between genuine and deceptive news articles. It typically employs techniques like text vectorization and classification algorithms. By training on labeled datasets, models learn to identify patterns indicative of fake or real news. Evaluating the model's performance using metrics like accuracy, precision, recall, and F1-score helps gauge its effectiveness. Advanced approaches, such as deep learning and transformer models, can enhance detection accuracy. The goal is to provide users with a tool to discern trustworthy information from potentially misleading or fabricated content in the digital landscape.

THANK YOU