NAAN MUDHALVAN
IBM: AI101
ARTIFICIAL INTELLIGENCE
PHASE 5

Fake News Detection Using NLP

TEAM MEMBERS:

1. *SIVA JEGADEEESH C B*
2. *BABITH SARISH S*
3. *RAJKUMAR M*
4. *LOGESH S*
5. *SHARVESH*

MENTOR

- *Dr. Sudhakar T*

MADRAS INSTITUTE OF TECHNOLOGY, ANNA UNIVERSITY, CHENNAI

Fake News Detection Using NLP

PHASE 5

FINAL DOCUMENT

Problem Statement: The fake news dataset is one of the classic text analytics datasets available on Kaggle. It consists of genuine and fake articles' titles and text from different authors. Our job is to create a model which predicts whether agiven news is real or fake.

Objective: The objective of this project is to develop a machine learning model that can accurately distinguish between genuine and fake news articles based ontheir titles and text content. By doing so, we aim to contribute to the fight against the spread of misinformation and fake news, which can have significant social and political consequences.

Data Source: We will use a *fake news dataset* available on Kaggle. This datasetcontains articles' titles and text, along with their corresponding labels indicatingwhether the news is genuine or fake.

Dataset link: https://www.kaggle.com/datasets/clmentbisaillon/fake-and-real-news-dataset

Project Objective: -

The objective of this project is to develop a robust and accurate machine learningmodel for the detection of fake news using Natural Language Processing (NLP) techniques. In an era where the spread of misinformation and fake news can have far-reaching consequences, our goal is to contribute to the effort of distinguishing between genuine and fake news articles. By harnessing the power of text analysis and deep learning, we aim to create a tool that can aid in combating the dissemination of false information and promote the dissemination of crediblenews sources. This project seeks to leverage a dataset of news articles, theirtitles, and content to design, train, and evaluate a model capable of making informed predictions about the authenticity of news reports.

Introduction

In today's information age, the rapid dissemination of news and information isboth a blessing and a curse. While it allows for quick access to valuable knowledge, it also presents opportunities for the spread of fake news, misinformation, and rumors. Fake news can have dire consequences, influencingpublic opinion, affecting elections, and causing social unrest. Therefore, it is of paramount importance to develop tools that can automatically discern between genuine and fake news.

This project focuses on the application of Natural Language Processing (NLP) techniques and machine learning to tackle the challenge of fake news detection.We will utilize a dataset comprising news articles' titles and content, labeled as either genuine or fake. Our approach involves text preprocessing, feature extraction, and the construction of a deep learning model that combines Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory(BiLSTM) layers. The use of TensorFlow as our framework of choice ensures the model's efficiency and scalability.

Our project's significance lies in its potential to enhance media literacy and empower individuals to make more informed decisions about the information theyconsume. By achieving high accuracy in detecting fake news, we aim to contributeto the broader mission of promoting credible journalism and combatting the spread of false narratives.

KeyChallenges:

1. **Text Preprocessing**: Raw text data often contains noise and irrelevant information. Preprocessing is essential to clean and transform the text datainto a suitable format for analysis and modelling.
2. **Feature Extraction**: Converting text into numerical features is crucial for machine learning models to understand and make predictions. We will explore techniques like TF-IDF and word embeddings for feature extraction.
3. **Model Selection**: Choosing an appropriate machine learning algorithm is critical for achieving high classification accuracy. We plan to use a Convolutional Neural Network (CNN) combined with a Bidirectional Long Short-Term Memory (BiLSTM) architecture, implemented using TensorFlow,to build our fake news detection model.
4. **Model Evaluation:** To assess the model's performance, we will use various evaluation metrics such as accuracy, precision, recall, F1-score, and the Receiver Operating Characteristic Area Under Curve (ROC-AUC). The choiceof metrics will depend on the project's specific requirements and the importance of false positives and false negatives.

Design Thinking

1. **Data Source**

   We will begin by obtaining the fake news dataset from Kaggle, which contains a substantial collection of news articles along with their associated labels (real or fake). This dataset will serve as the foundation forour fake news detection project.

2. **Data Preprocessing**

   Before feeding the text data into our machine learning model, we need topreprocess it to ensure that it is in a clean and standardized format. Datapreprocessing steps will include:

   - Text Cleaning: Removing any HTML tags, special characters, and irrelevant symbols.
   - Tokenization: Splitting the text into individual words or tokens.
   - Stopword Removal: Eliminating common and uninformative wordssuch as "the," "is," and "and."
   - Lemmatization or Stemming: Reducing words to their base or rootform to normalize text.
   - Text Vectorization: Converting the text data into numerical representations for modeling.

3. **Feature Extraction**

   We will explore two common techniques for text feature extraction:

   a) TF-IDF (Term Frequency-Inverse Document Frequency): TF-IDF is a statistical measure that evaluates the importance of a word in a document relative to a collection of documents. We will use it to convert the text data into a matrix of TF-IDF features.

   b) Word Embeddings: Word embeddings, such as Word2Vec or GloVe,can capture semantic relationships between words. We will experiment with pre-trained word embeddings or train custom embeddings on our dataset.

4. **Model Selection**

   Our model choice is a combination of many ML & DL algorithms implemented using TensorFlow. This architecture is well-suited for capturing both local and global patterns in text data, making it suitable forfake news detection.

5. **Model Training**

The model will be trained using the pre-processed and feature-engineeredtext data. We will split the dataset into training, validation, and test sets toensure proper model evaluation. Training will involve optimizing model parameters and monitoring performance using appropriate metrics.

6. **Evaluation**

To evaluate the effectiveness of our fake news detection model, we willemploy a range of evaluation metrics:

- Accuracy: Measures the overall correctness of the model's predictions.
- Precision: Calculates the ratio of true positive predictions to the totalpositive predictions, indicating the model's ability to avoid false positives.
- Recall: Calculates the ratio of true positive predictions to the totalactual positives, indicating the model's ability to capture all positiveinstances.
- F1-Score: Harmonic mean of precision and recall, providing a balanced measure of model performance.
- ROC-AUC: Measures the area under the Receiver Operating Characteristic curve, indicating the model's ability to distinguishbetween real and fake news.

These metrics will help us assess the model's performance comprehensively and make any necessary improvements to achieve ourgoal of accurately detecting fake news articles.

INNOVATIONS:

1. Hybrid Approach:

   Incorporating a hybrid approach that combines content-based and social context-based features to identify fake news. An example is the Transformer-based model proposed by Raza and Ding, which utilizes both news article information and social context to enhancefake news detection. This model utilizes a Transformer architecture, comprising an encoder for learning useful representations from fake news data and a decoder for predicting future behavior based on past observations. It also integrates numerous features from news content and social contexts to improve classification accuracy.

2. Multimodal Approach:

   Employing a multimodal approach that leverages both textual andvisual data for fake news detection. A model like the one proposedby Wang et al. could be adopted, which employs a multimodal deep neural network to merge textual and visual features. This model consists of three key components: a text encoder for extracting textual features from news content, an image encoder for extracting visual features from news images, and a fusion module to combine these features and make a final prediction.

3. Transfer Learning:

   Utilizing transfer learning techniques to improve fake news detection performance. For instance, we can employ pre-trained models like BERT, which is a pre-trained language representation model that can be fine-tuned for various natural language processing tasks, including fake news detection. BERT is proficientat capturing both syntactic and semantic information from extensive text corpora and can be easily adapted to various domains and languages.

4. Ensemble Learning:

   Implementing ensemble learning methods to combine the predictions of multiple models. By leveraging the diversity of different models, we can potentially enhance the accuracy androbustness of our fake news detection system.

5. Explainable AI (XAI):

   Integrating XAI techniques to provide transparency and interpretability in fake news detection. This ensures that the model's decisions can be understood and validated, which is crucial for building trust in the system.

6. Continuous Learning:

   Implementing continuous learning mechanisms to adapt to evolving fake news patterns and emerging disinformation tactics. This involves regularly updating the model with newdata to ensure it remains effective over time.

7. User Feedback Integration:

   Incorporating user feedback mechanisms to gather input fromusers and improve the model's performance based on real- world usage and user perceptions of news credibility.

8. Cross-lingual and Cross-cultural Adaptation:

   Extending the model's capabilities to detect fake news in multiple languages and adapt to different cultural contexts,thereby enhancing its applicability on a global scale.

TOOLS:

Google Colab: Google Colab, a cloud-based Jupyter notebookenvironment, serves as our primary coding platform

Algorithms and Techniques:

1. **TF-IDF (Term Frequency-Inverse Document Frequency)**: Used fortext feature extraction to convert text data into numerical form.
2. **Multinomial Naive Bayes**: A classification algorithm for text dataoften used for spam and fake news detection.
3. **Logistic Regression**: A classification algorithm for binary and multi-class classification tasks.
4. **Random Forest**: An ensemble learning method for classification and regression tasks.
5. **Passive Aggressive Classifier**: A type of online learning algorithm fortext classification.
6. **Decision Tree**: A classification algorithm that uses a tree structurefor decision-making.
7. **Train-Test Split**: A technique to split the dataset into training andtesting sets for model evaluation.
8. **Confusion Matrix**: A tool for evaluating classification model performance.
9. **Precision, Recall, F1-Score**: Metrics for evaluating the performanceof classification models.
10. **ROC Curve (Receiver Operating Characteristic):** Used to assess the performance of binary classification models.
11. **Stopwords Removal**: A text preprocessing technique to remove common words that do not contribute much information.
12. **Lowercasing:** Converting text to lowercase to ensure uniformity.
13. **Tokenization:** Breaking text into words or tokens for analysis.

IMPLEMENTATION STEPS:

1) Import Necessary Libraries:

Start by importing the required Python libraries, such as pandas, numpy, scikit-learn, and natural language processing libraries like NLTK or spaCy.

CODE:

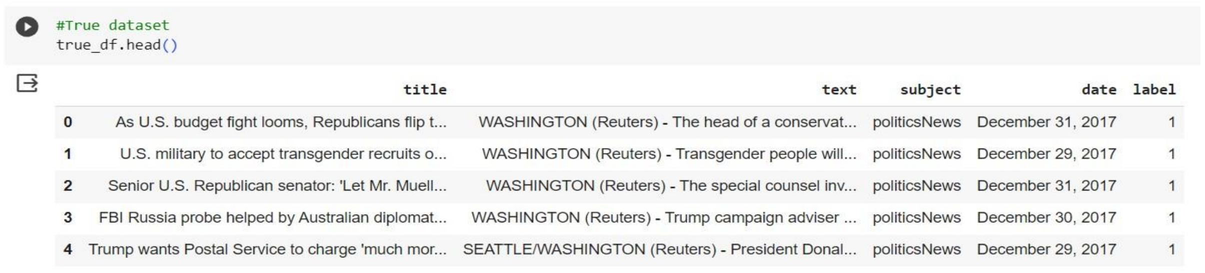
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
import nltk
from nltk.corpus import stopwords
nltk.download('punkt')
nltk.download('stopwords')
```

2) Load and Explore the Dataset:

Load the CSV files 'true.csv' and 'false.csv' using pandas and explore the dataset to understand its structure.

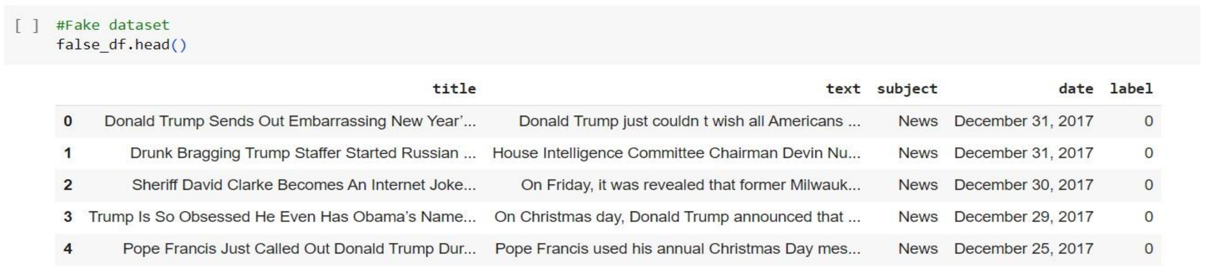
CODE:

```
# Load the dataset
true_df = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/fake-news-detective/dataset/True.csv')
false_df = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/fake-news-detective/dataset/Fake.csv')

# Add labels to indicate real and fake news
true_df['label'] = 1
false_df['label'] = 0

# Concatenate both datasets
data = pd.concat([true_df, false_df])
```



```
#True dataset
true_df.head()
```

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



```
[ ] #Fake dataset
    false_df.head()
```

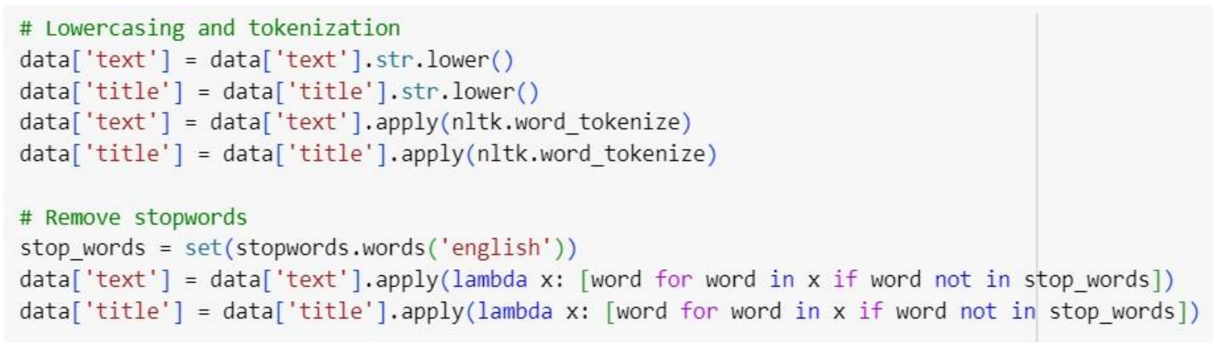
	title	text	subject	date	label
0	Donald Trump Sends Out Embarrassing New Year'...	Donald Trump just couldn t wish all Americans ...	News	December 31, 2017	0
1	Drunk Bragging Trump Staffer Started Russian ...	House Intelligence Committee Chairman Devin Nu...	News	December 31, 2017	0
2	Sheriff David Clarke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

3) Data Preprocessing:

Data preprocessing is essential for text data. Perform the followingpreprocessing steps:

- Lowercasing: Convert text to lowercase. Tokenization: Split
- text into words or tokens.
- Stopword Removal: Remove common words like 'and', 'the', etc.
- Text Vectorization: Convert text into numerical format (e.g.,using TF-IDF or Count Vectorization).

CODE:

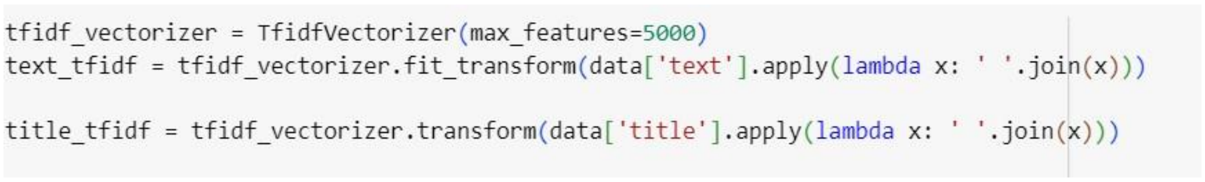


```python
# Lowercasing and tokenization
data['text'] = data['text'].str.lower()
data['title'] = data['title'].str.lower()
data['text'] = data['text'].apply(nltk.word_tokenize)
data['title'] = data['title'].apply(nltk.word_tokenize)

# Remove stopwords
stop_words = set(stopwords.words('english'))
data['text'] = data['text'].apply(lambda x: [word for word in x if word not in stop_words])
data['title'] = data['title'].apply(lambda x: [word for word in x if word not in stop_words])
```

4) Feature Extraction (TF-IDF):

Choose a text vectorization method. You can use either Count Vectorization or TF-IDF Vectorization.

CODE:

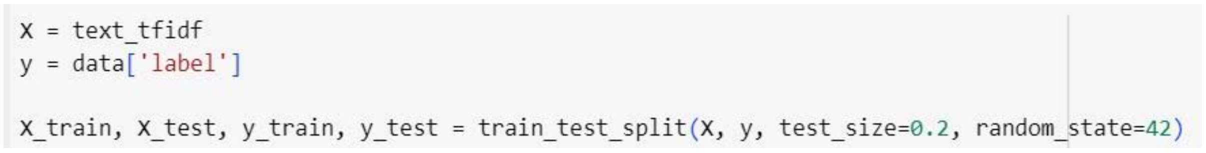


```python
tfidf_vectorizer = TfidfVectorizer(max_features=5000)
text_tfidf = tfidf_vectorizer.fit_transform(data['text'].apply(lambda x: ' '.join(x)))

title_tfidf = tfidf_vectorizer.transform(data['title'].apply(lambda x: ' '.join(x)))
```

5) **Split the Data into Training and Testing Sets:**

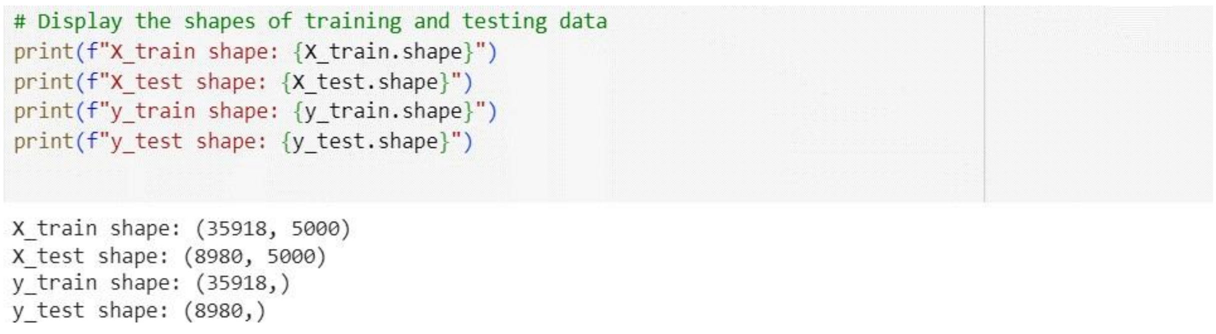
Split the dataset into training and testing sets to evaluate the model'sperformance.

CODE:



```
X = text_tfidf
y = data['label']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

SAMPLE OUTPUT (Data Preprocessing and splitting):



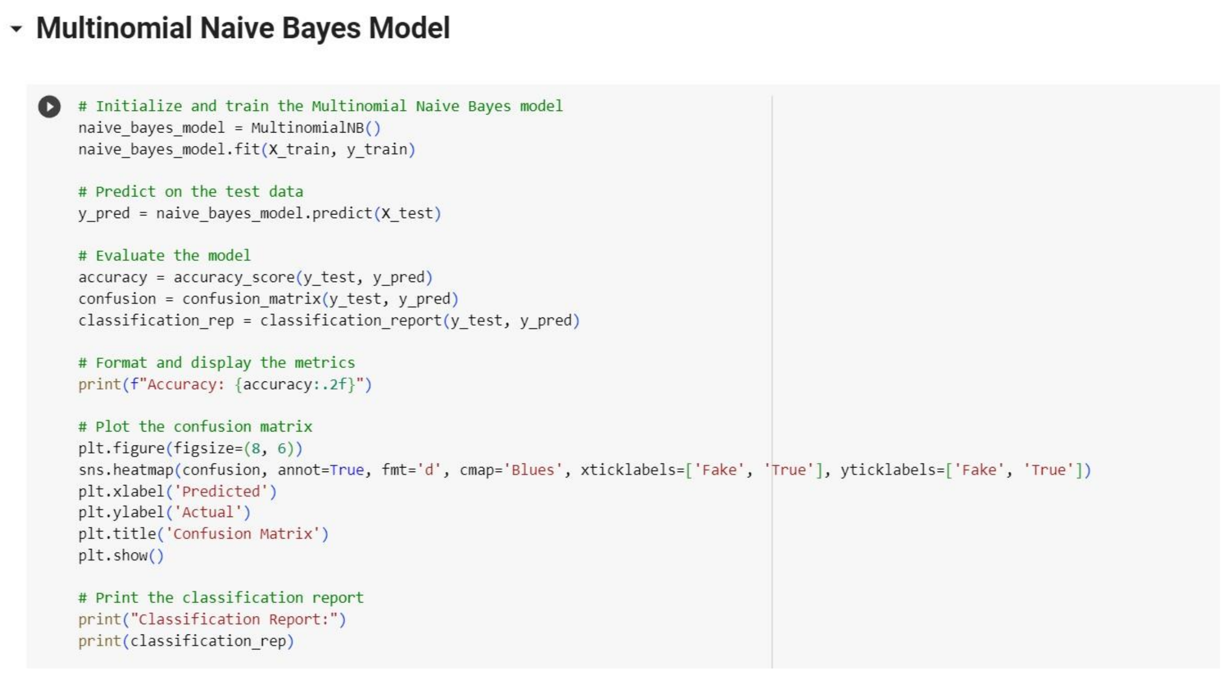
```
# Display the shapes of training and testing data
print(f"X_train shape: {X_train.shape}")
print(f"X_test shape: {X_test.shape}")
print(f"y_train shape: {y_train.shape}")
print(f"y_test shape: {y_test.shape}")
```

```
X_train shape: (35918, 5000)
X_test shape: (8980, 5000)
y_train shape: (35918,)
y_test shape: (8980,)
```

6) **Model Training**:

During this stage, we will proceed to train the Multinomial Naive Bayes model utilizing the designated training dataset. This process will entail instructing the model to differentiate between authenticand counterfeit news articles based on the TF-IDF vectors that havebeen meticulously prepared.

CODE: (MULTINOMIAL NAIVE BAYES ALGORITHM)

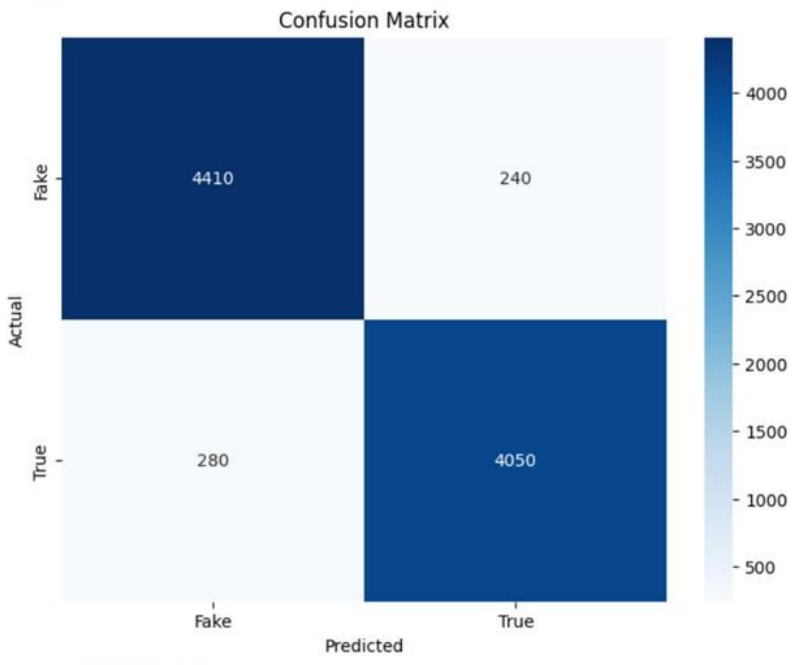


Multinomial Naive Bayes Model

```python
# Initialize and train the Multinomial Naive Bayes model
naive_bayes_model = MultinomialNB()
naive_bayes_model.fit(X_train, y_train)

# Predict on the test data
y_pred = naive_bayes_model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
confusion = confusion_matrix(y_test, y_pred)
classification_rep = classification_report(y_test, y_pred)

# Format and display the metrics
print(f"Accuracy: {accuracy:.2f}")

# Plot the confusion matrix
plt.figure(figsize=(8, 6))
sns.heatmap(confusion, annot=True, fmt='d', cmap='Blues', xticklabels=['Fake', 'True'], yticklabels=['Fake', 'True'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()

# Print the classification report
print("Classification Report:")
print(classification_rep)
```

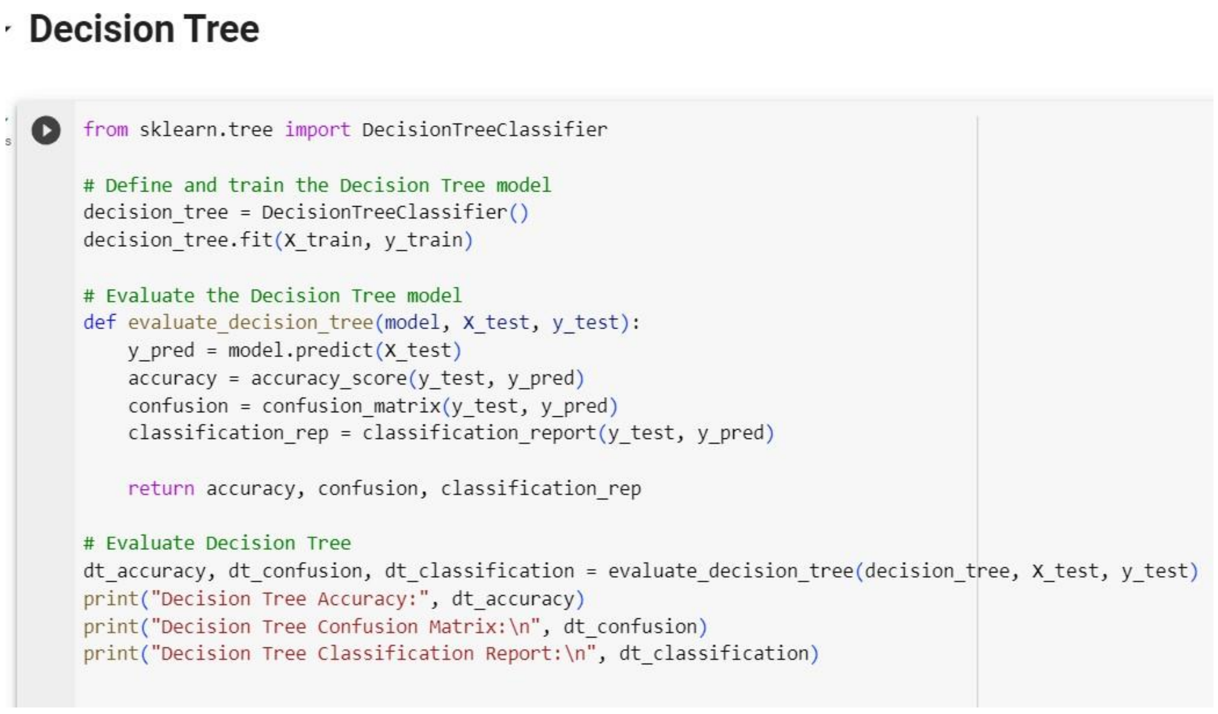
OUTPUT: (MULTINOMIAL NAIVE BAYES ALGORITHM)

```
Accuracy: 0.94
```



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

           0       0.94      0.95      0.94      4650
           1       0.94      0.94      0.94      4330

    accuracy                           0.94      8980
   macro avg       0.94      0.94      0.94      8980
weighted avg       0.94      0.94      0.94      8980
```

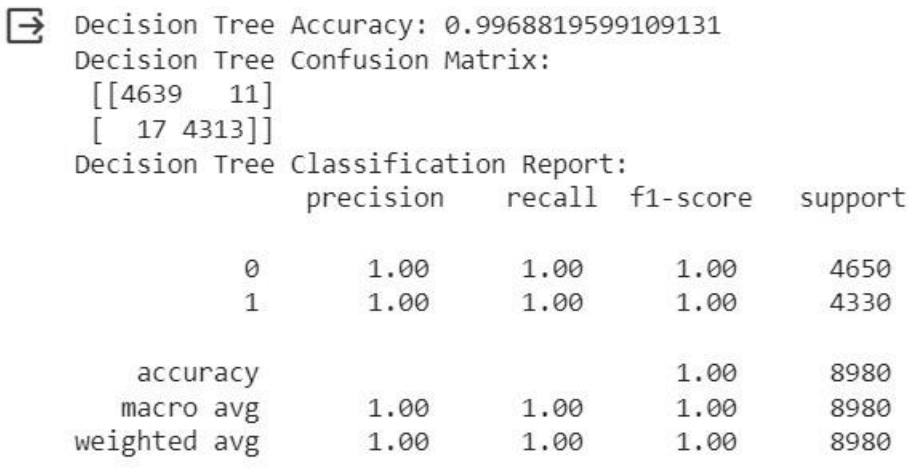
CODE: (DECISION TREE)



Decision Tree

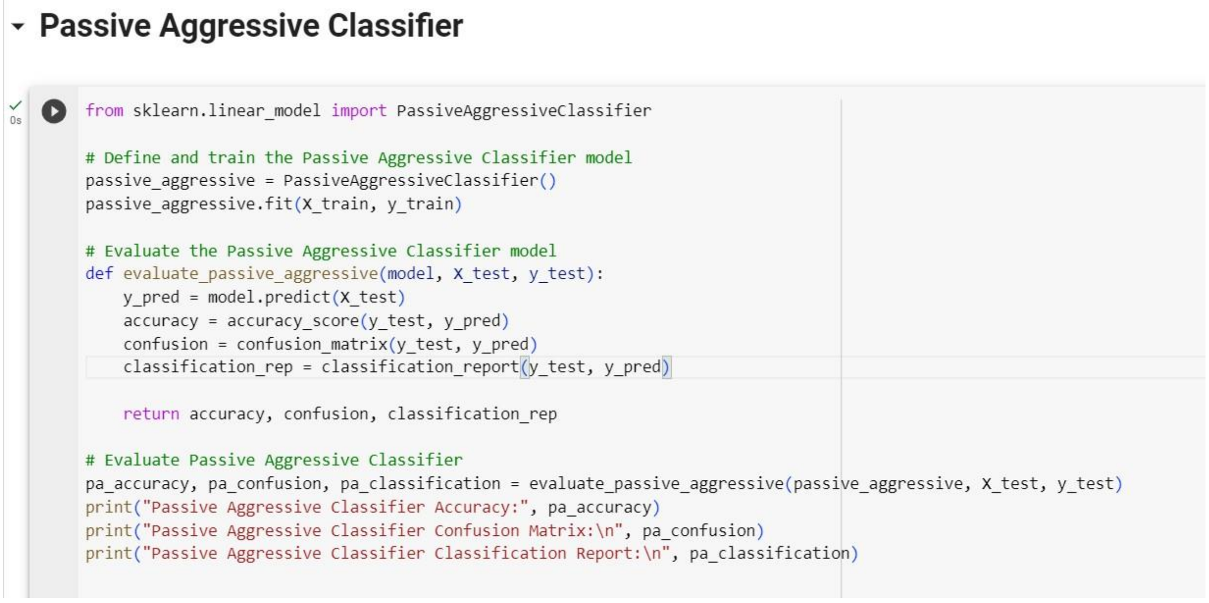
```python
from sklearn.tree import DecisionTreeClassifier

# Define and train the Decision Tree model
decision_tree = DecisionTreeClassifier()
decision_tree.fit(X_train, y_train)

# Evaluate the Decision Tree model
def evaluate_decision_tree(model, X_test, y_test):
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    confusion = confusion_matrix(y_test, y_pred)
    classification_rep = classification_report(y_test, y_pred)

    return accuracy, confusion, classification_rep

# Evaluate Decision Tree
dt_accuracy, dt_confusion, dt_classification = evaluate_decision_tree(decision_tree, X_test, y_test)
print("Decision Tree Accuracy:", dt_accuracy)
print("Decision Tree Confusion Matrix:\n", dt_confusion)
print("Decision Tree Classification Report:\n", dt_classification)
```

OUTPUT:



```
Decision Tree Accuracy: 0.9968819599109131
Decision Tree Confusion Matrix:
 [[4639   11]
 [  17 4313]]
Decision Tree Classification Report:
               precision    recall  f1-score   support

           0       1.00      1.00      1.00      4650
           1       1.00      1.00      1.00      4330

    accuracy                           1.00      8980
   macro avg       1.00      1.00      1.00      8980
weighted avg       1.00      1.00      1.00      8980
```

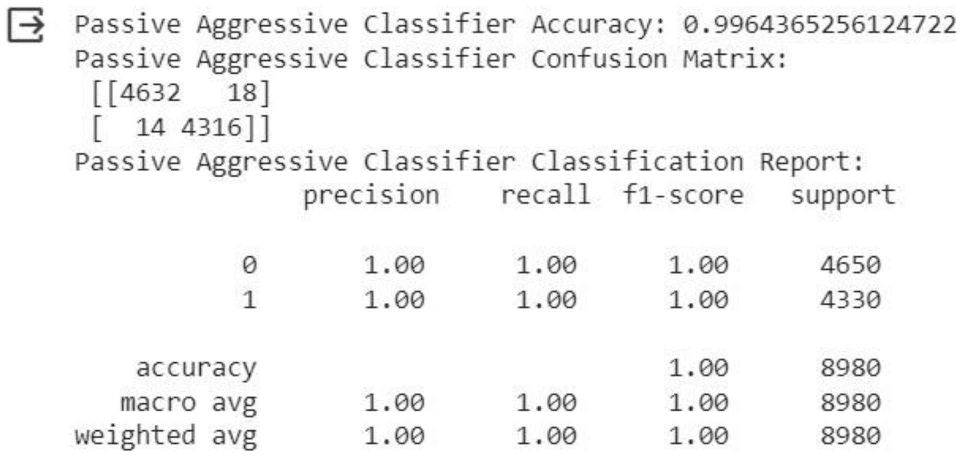
CODE: (PASSIVE AGGRESSIVE CLASSIFIER)



Passive Aggressive Classifier

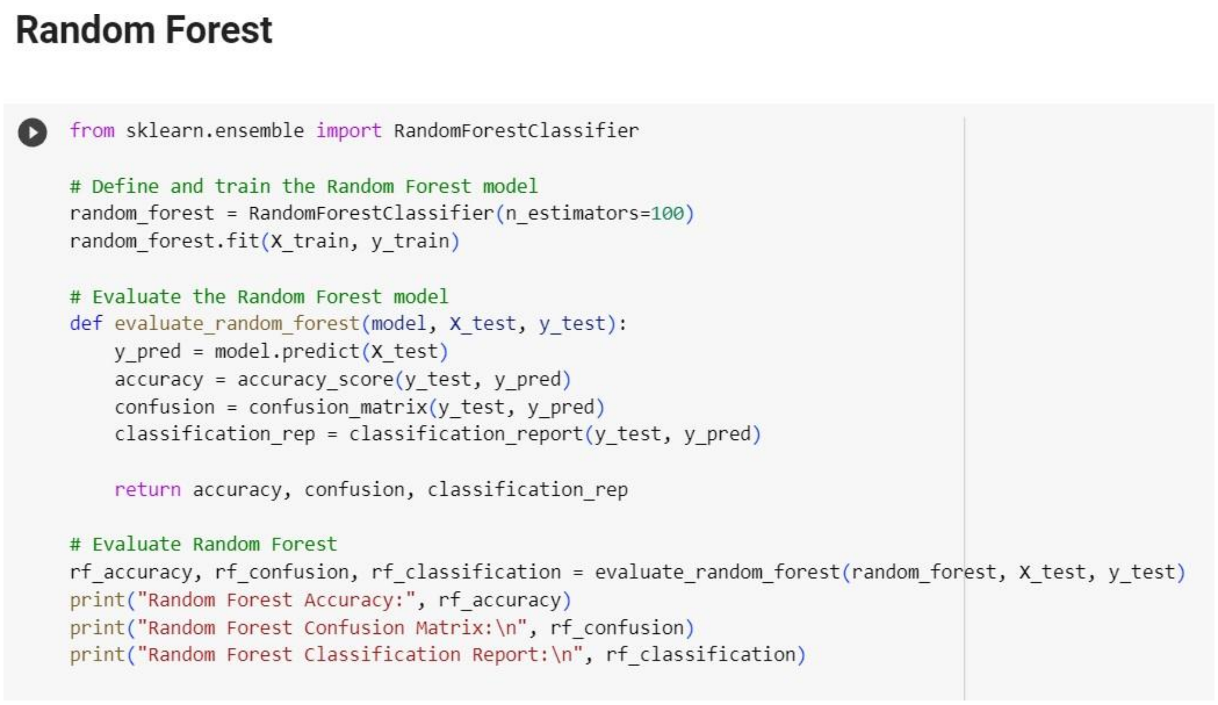
```
from sklearn.linear_model import PassiveAggressiveClassifier

# Define and train the Passive Aggressive Classifier model
passive_aggressive = PassiveAggressiveClassifier()
passive_aggressive.fit(X_train, y_train)

# Evaluate the Passive Aggressive Classifier model
def evaluate_passive_aggressive(model, X_test, y_test):
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    confusion = confusion_matrix(y_test, y_pred)
    classification_rep = classification_report(y_test, y_pred)

    return accuracy, confusion, classification_rep

# Evaluate Passive Aggressive Classifier
pa_accuracy, pa_confusion, pa_classification = evaluate_passive_aggressive(passive_aggressive, X_test, y_test)
print("Passive Aggressive Classifier Accuracy:", pa_accuracy)
print("Passive Aggressive Classifier Confusion Matrix:\n", pa_confusion)
print("Passive Aggressive Classifier Classification Report:\n", pa_classification)
```

OUTPUT: (PASSIVE AGGRESSIVE CLASSIFIER)



```
Passive Aggressive Classifier Accuracy: 0.9964365256124722
Passive Aggressive Classifier Confusion Matrix:
 [[4632   18]
 [  14 4316]]
Passive Aggressive Classifier Classification Report:
               precision    recall  f1-score   support

           0       1.00      1.00      1.00      4650
           1       1.00      1.00      1.00      4330

    accuracy                           1.00      8980
   macro avg       1.00      1.00      1.00      8980
weighted avg       1.00      1.00      1.00      8980
```

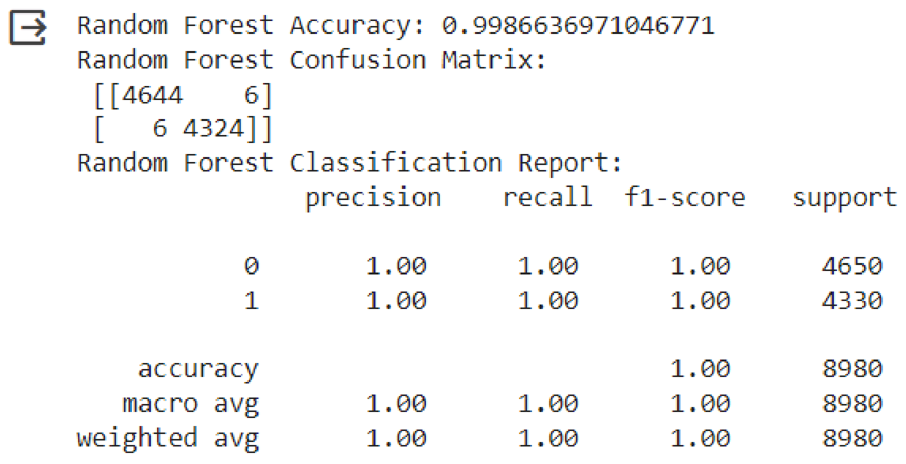
CODE: (RANDOM FOREST)



Random Forest

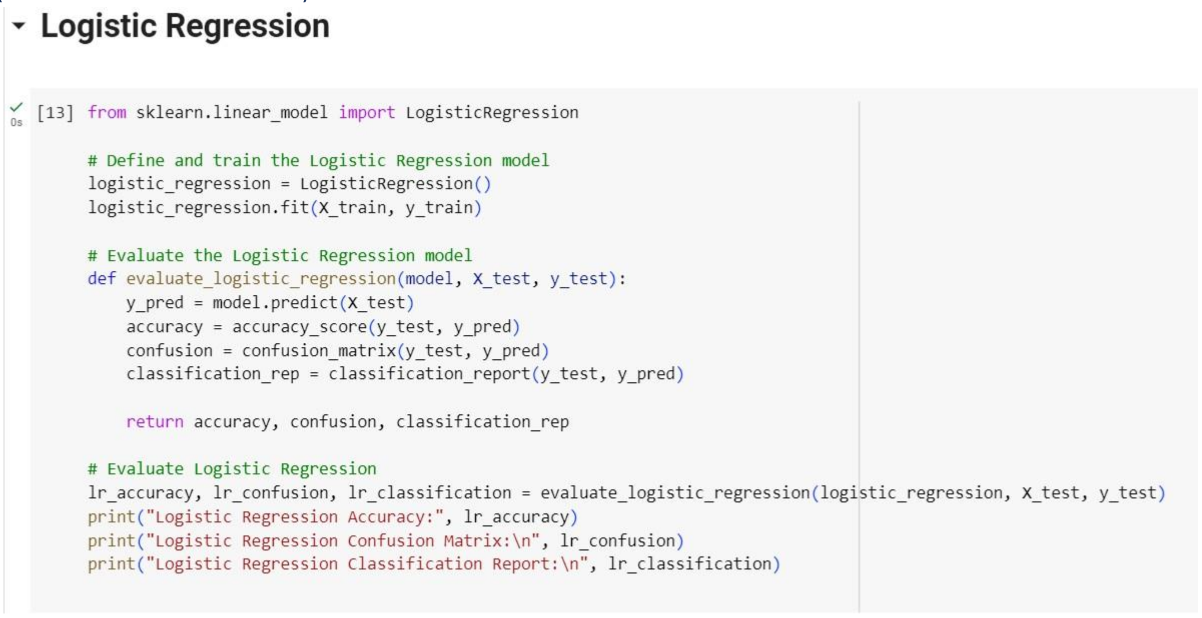
```python
from sklearn.ensemble import RandomForestClassifier

# Define and train the Random Forest model
random_forest = RandomForestClassifier(n_estimators=100)
random_forest.fit(X_train, y_train)

# Evaluate the Random Forest model
def evaluate_random_forest(model, X_test, y_test):
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    confusion = confusion_matrix(y_test, y_pred)
    classification_rep = classification_report(y_test, y_pred)

    return accuracy, confusion, classification_rep

# Evaluate Random Forest
rf_accuracy, rf_confusion, rf_classification = evaluate_random_forest(random_forest, X_test, y_test)
print("Random Forest Accuracy:", rf_accuracy)
print("Random Forest Confusion Matrix:\n", rf_confusion)
print("Random Forest Classification Report:\n", rf_classification)
```

OUTPUT: (RANDOM FOREST)



```
Random Forest Accuracy: 0.9986636971046771
Random Forest Confusion Matrix:
 [[4644    6]
 [   6 4324]]
Random Forest Classification Report:
               precision    recall  f1-score   support

           0       1.00      1.00      1.00      4650
           1       1.00      1.00      1.00      4330

    accuracy                           1.00      8980
   macro avg       1.00      1.00      1.00      8980
weighted avg       1.00      1.00      1.00      8980
```

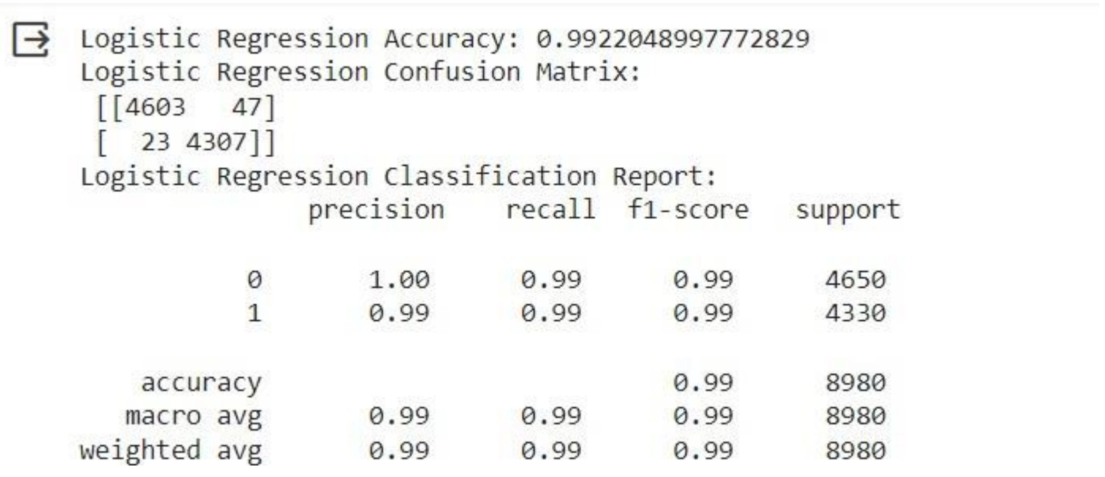
CODE: (LOGISTIC REGRESSION)



Logistic Regression

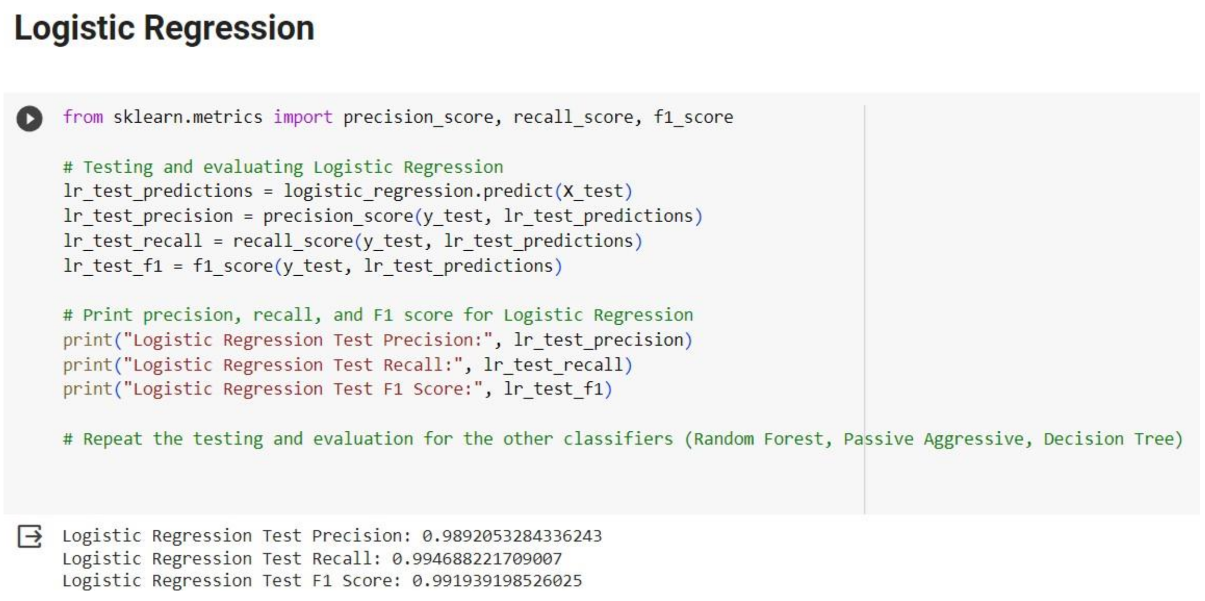
```
[13] from sklearn.linear_model import LogisticRegression

     # Define and train the Logistic Regression model
     logistic_regression = LogisticRegression()
     logistic_regression.fit(X_train, y_train)

     # Evaluate the Logistic Regression model
     def evaluate_logistic_regression(model, X_test, y_test):
         y_pred = model.predict(X_test)
         accuracy = accuracy_score(y_test, y_pred)
         confusion = confusion_matrix(y_test, y_pred)
         classification_rep = classification_report(y_test, y_pred)

         return accuracy, confusion, classification_rep

     # Evaluate Logistic Regression
     lr_accuracy, lr_confusion, lr_classification = evaluate_logistic_regression(logistic_regression, X_test, y_test)
     print("Logistic Regression Accuracy:", lr_accuracy)
     print("Logistic Regression Confusion Matrix:\n", lr_confusion)
     print("Logistic Regression Classification Report:\n", lr_classification)
```

OUTPUT: (LOGISTIC REGRESSION)



```
Logistic Regression Accuracy: 0.9922048997772829
Logistic Regression Confusion Matrix:
 [[4603   47]
 [  23 4307]]
Logistic Regression Classification Report:
               precision    recall  f1-score   support

           0       1.00      0.99      0.99      4650
           1       0.99      0.99      0.99      4330

    accuracy                           0.99      8980
   macro avg       0.99      0.99      0.99      8980
weighted avg       0.99      0.99      0.99      8980
```

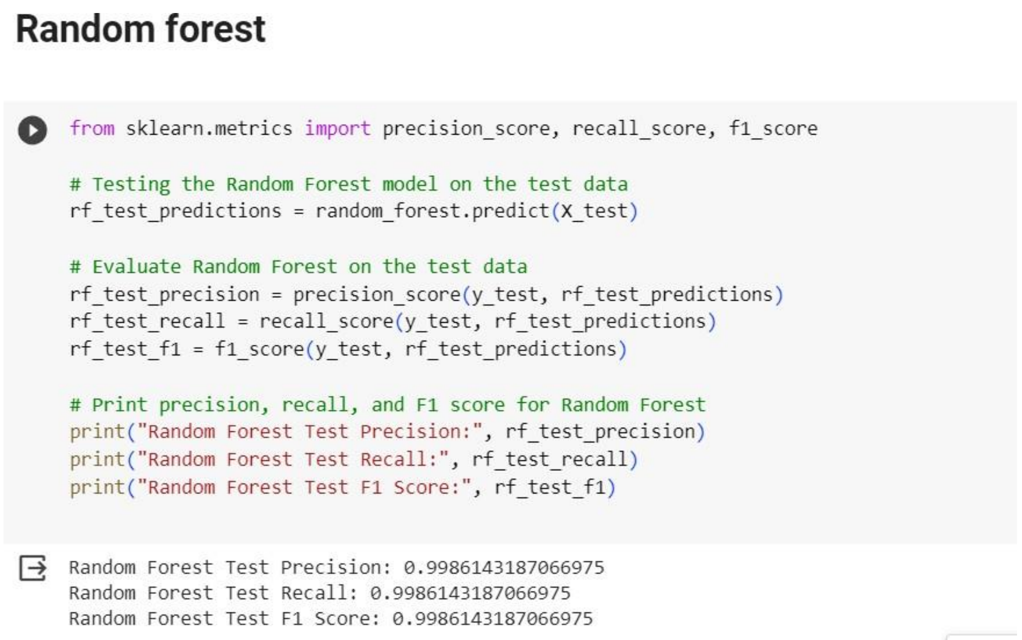
7) **Model Evaluation**:
Following the training process, we will conduct an evaluation of themodel's performance using the designated testing dataset. This evaluation is essential to gauge the model's efficacy in accuratelyclassifying news articles as either genuine or fraudulent. Standardevaluation metrics, such as accuracy, a confusion matrix, and a classification report, will be employed to provide comprehensive insights into the model's classification capabilities.



Logistic Regression

```
from sklearn.metrics import precision_score, recall_score, f1_score

# Testing and evaluating Logistic Regression
lr_test_predictions = logistic_regression.predict(X_test)
lr_test_precision = precision_score(y_test, lr_test_predictions)
lr_test_recall = recall_score(y_test, lr_test_predictions)
lr_test_f1 = f1_score(y_test, lr_test_predictions)

# Print precision, recall, and F1 score for Logistic Regression
print("Logistic Regression Test Precision:", lr_test_precision)
print("Logistic Regression Test Recall:", lr_test_recall)
print("Logistic Regression Test F1 Score:", lr_test_f1)

# Repeat the testing and evaluation for the other classifiers (Random Forest, Passive Aggressive, Decision Tree)
```

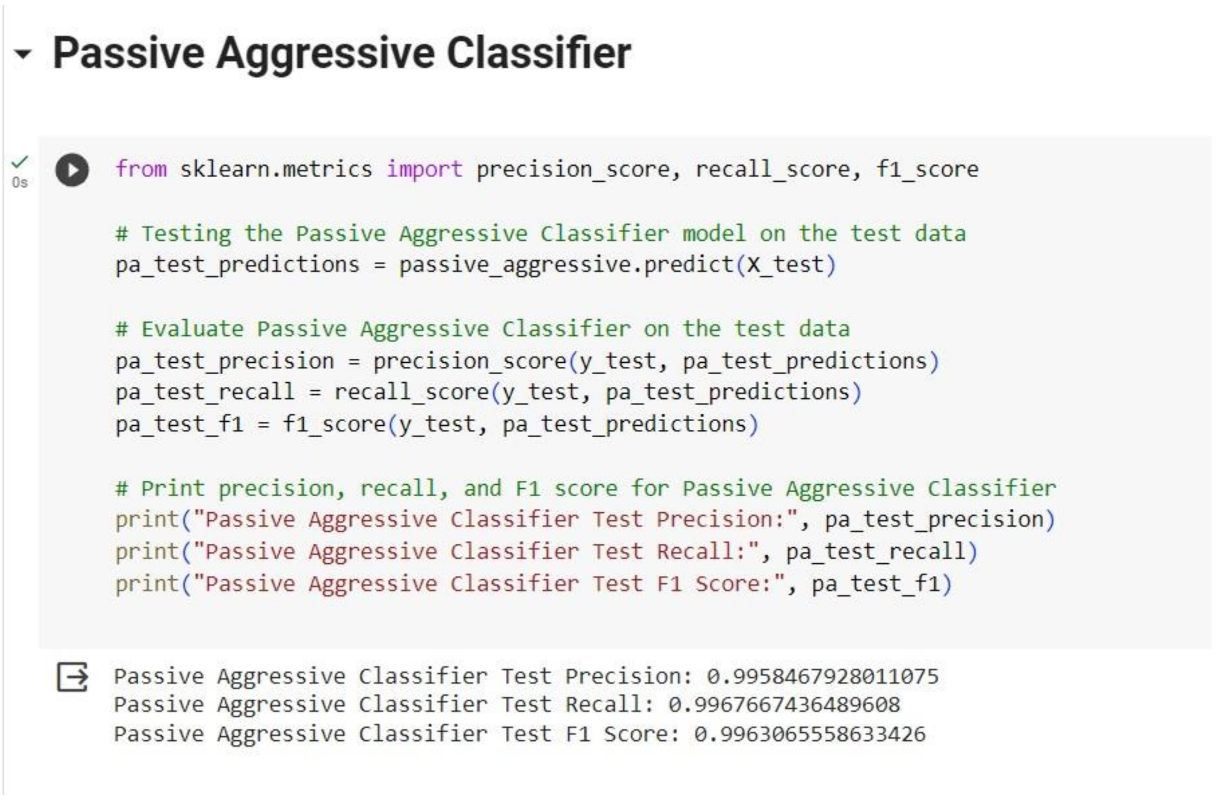
```
Logistic Regression Test Precision: 0.9892053284336243
Logistic Regression Test Recall: 0.994688221709007
Logistic Regression Test F1 Score: 0.991939198526025
```



Random forest

```
from sklearn.metrics import precision_score, recall_score, f1_score

# Testing the Random Forest model on the test data
rf_test_predictions = random_forest.predict(X_test)

# Evaluate Random Forest on the test data
rf_test_precision = precision_score(y_test, rf_test_predictions)
rf_test_recall = recall_score(y_test, rf_test_predictions)
rf_test_f1 = f1_score(y_test, rf_test_predictions)

# Print precision, recall, and F1 score for Random Forest
print("Random Forest Test Precision:", rf_test_precision)
print("Random Forest Test Recall:", rf_test_recall)
print("Random Forest Test F1 Score:", rf_test_f1)
```

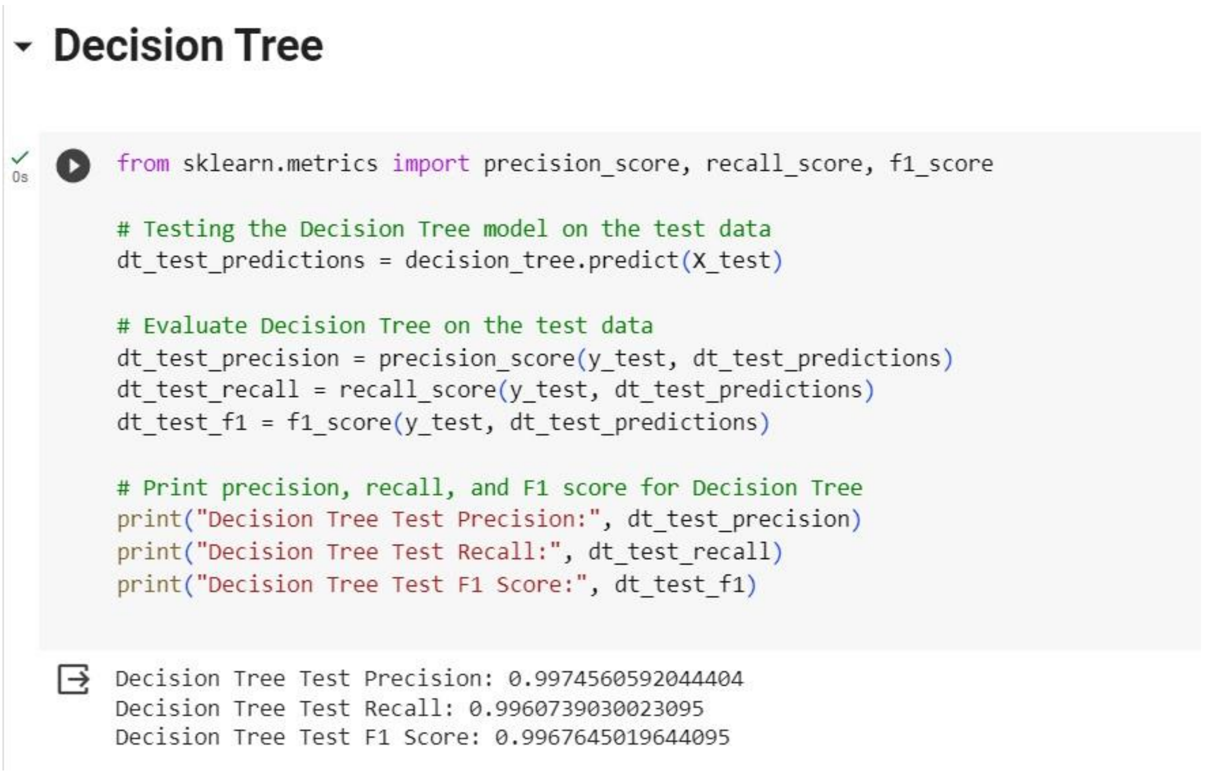
```
Random Forest Test Precision: 0.9986143187066975
Random Forest Test Recall: 0.9986143187066975
Random Forest Test F1 Score: 0.9986143187066975
```



Passive Aggressive Classifier

```python
from sklearn.metrics import precision_score, recall_score, f1_score

# Testing the Passive Aggressive Classifier model on the test data
pa_test_predictions = passive_aggressive.predict(X_test)

# Evaluate Passive Aggressive Classifier on the test data
pa_test_precision = precision_score(y_test, pa_test_predictions)
pa_test_recall = recall_score(y_test, pa_test_predictions)
pa_test_f1 = f1_score(y_test, pa_test_predictions)

# Print precision, recall, and F1 score for Passive Aggressive Classifier
print("Passive Aggressive Classifier Test Precision:", pa_test_precision)
print("Passive Aggressive Classifier Test Recall:", pa_test_recall)
print("Passive Aggressive Classifier Test F1 Score:", pa_test_f1)
```

```
Passive Aggressive Classifier Test Precision: 0.9958467928011075
Passive Aggressive Classifier Test Recall: 0.9967667436489608
Passive Aggressive Classifier Test F1 Score: 0.9963065558633426
```

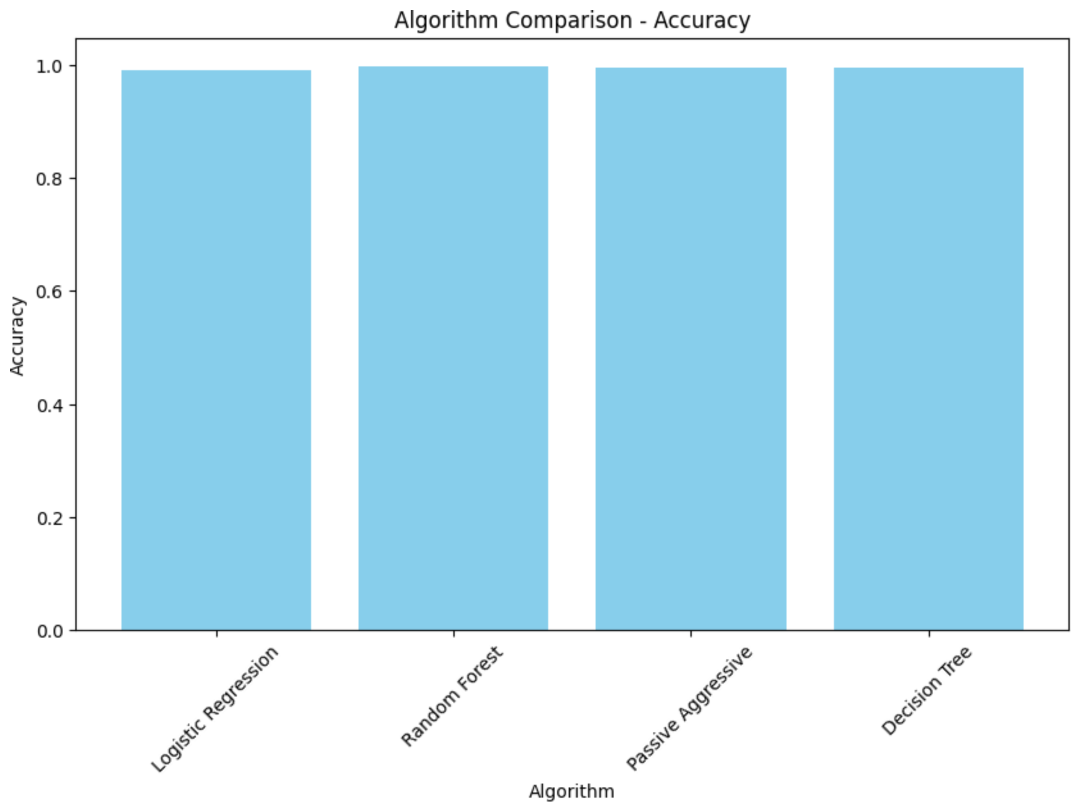


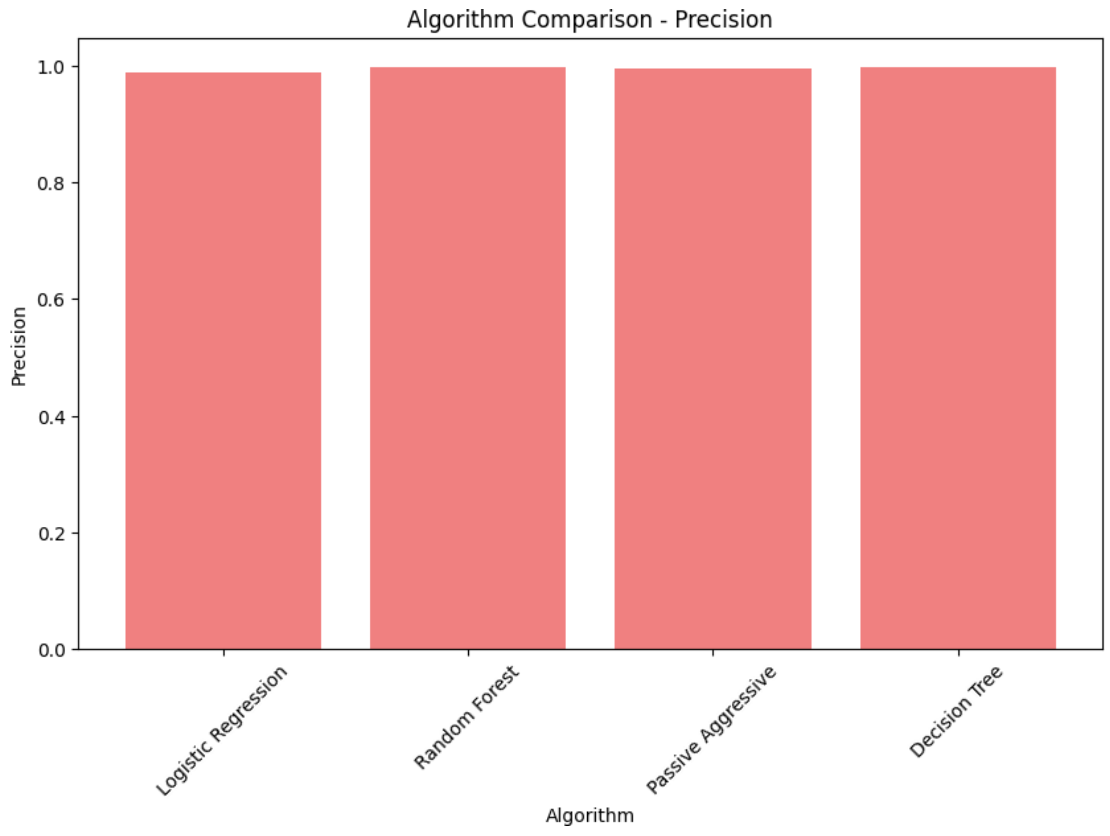
Decision Tree

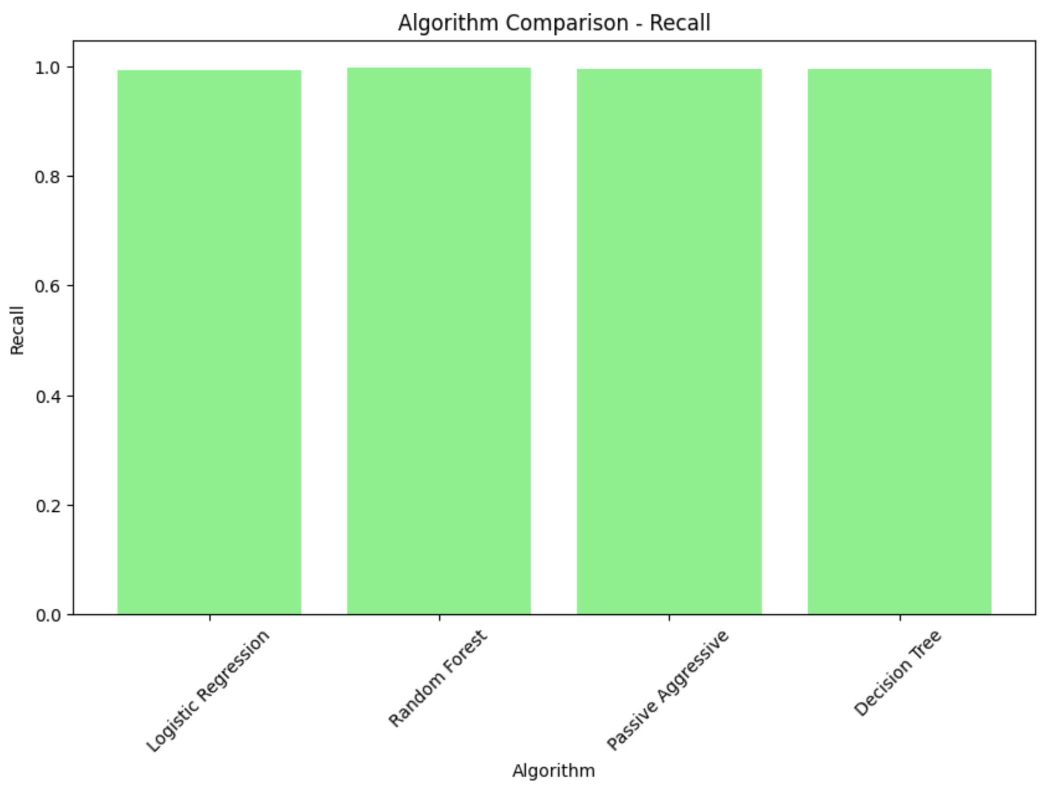
```python
from sklearn.metrics import precision_score, recall_score, f1_score

# Testing the Decision Tree model on the test data
dt_test_predictions = decision_tree.predict(X_test)

# Evaluate Decision Tree on the test data
dt_test_precision = precision_score(y_test, dt_test_predictions)
dt_test_recall = recall_score(y_test, dt_test_predictions)
dt_test_f1 = f1_score(y_test, dt_test_predictions)

# Print precision, recall, and F1 score for Decision Tree
print("Decision Tree Test Precision:", dt_test_precision)
print("Decision Tree Test Recall:", dt_test_recall)
print("Decision Tree Test F1 Score:", dt_test_f1)
```

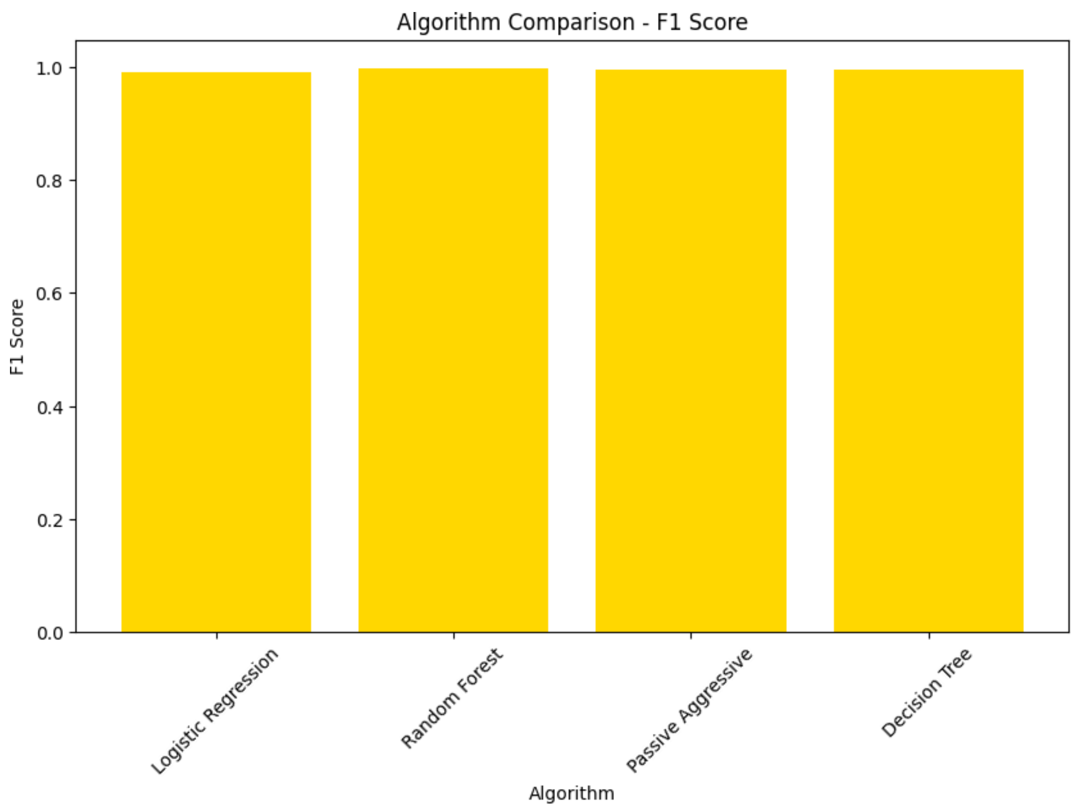
```
Decision Tree Test Precision: 0.9974560592044404
Decision Tree Test Recall: 0.9960739030023095
Decision Tree Test F1 Score: 0.9967645019644095
```

RESULT ANALYSIS GRAPH:

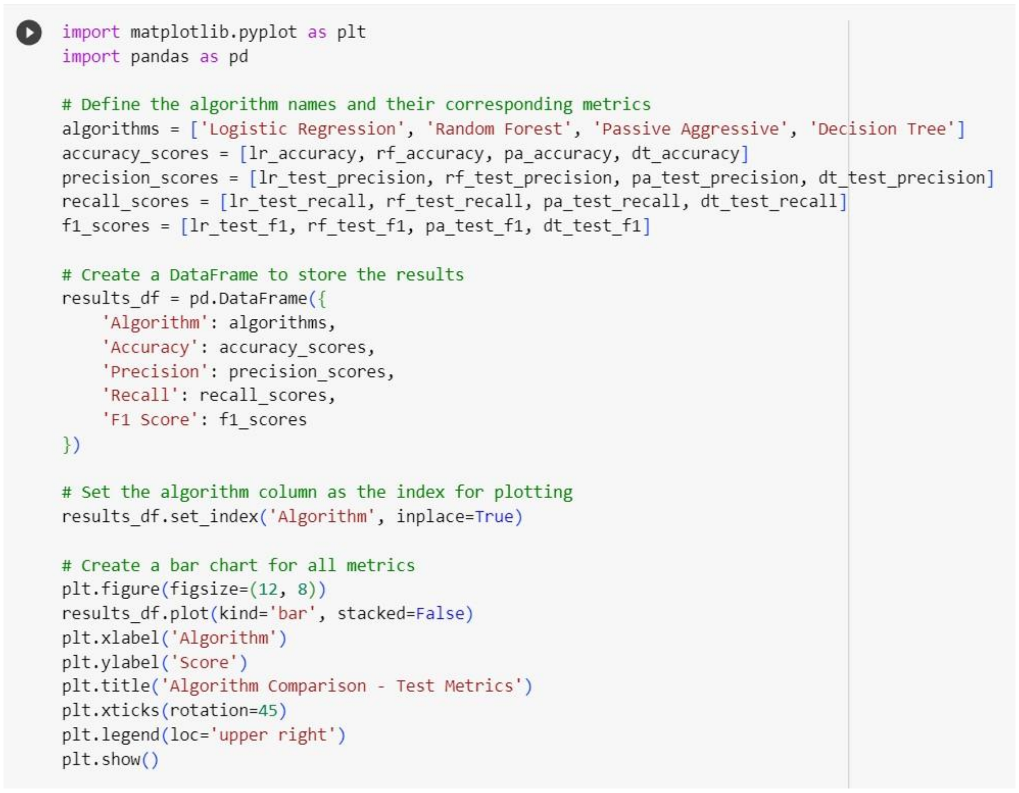




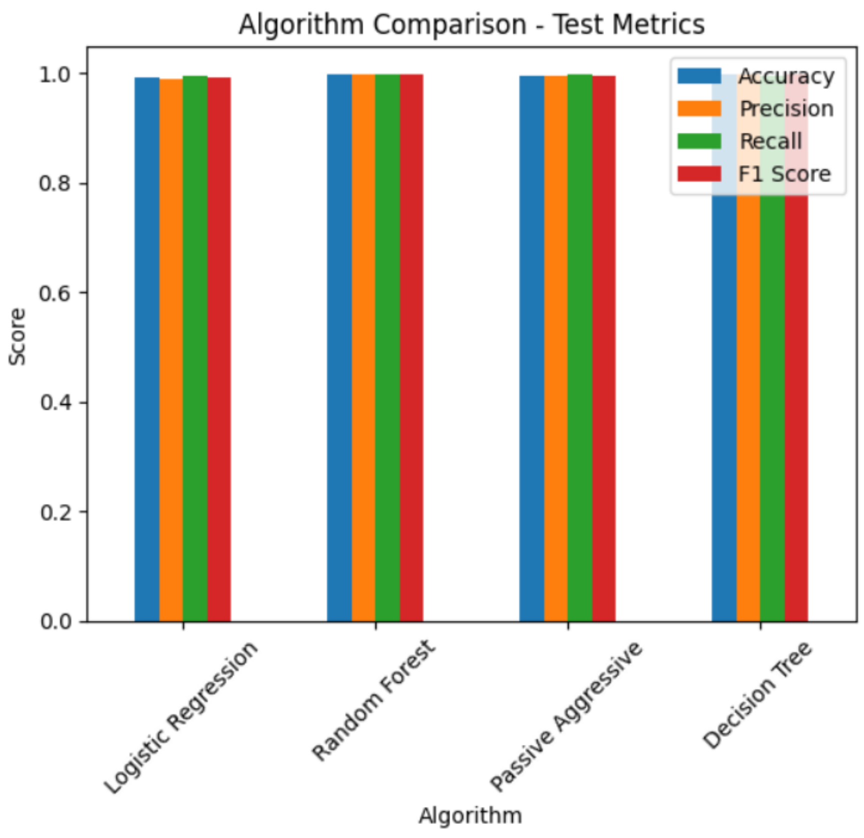

Algorithm Comparison - Recall
Recall
1.0
0.8
0.6
0.4
0.2
0.0
Logistic Regression
Random Forest
Passive Aggressive
Decision Tree
Algorithm


Algorithm Comparison - F1 Score
F1 Score
1.0
0.8
0.6
0.4
0.2
0.0
Logistic Regression
Random Forest
Passive Aggressive
Decision Tree
Algorithm

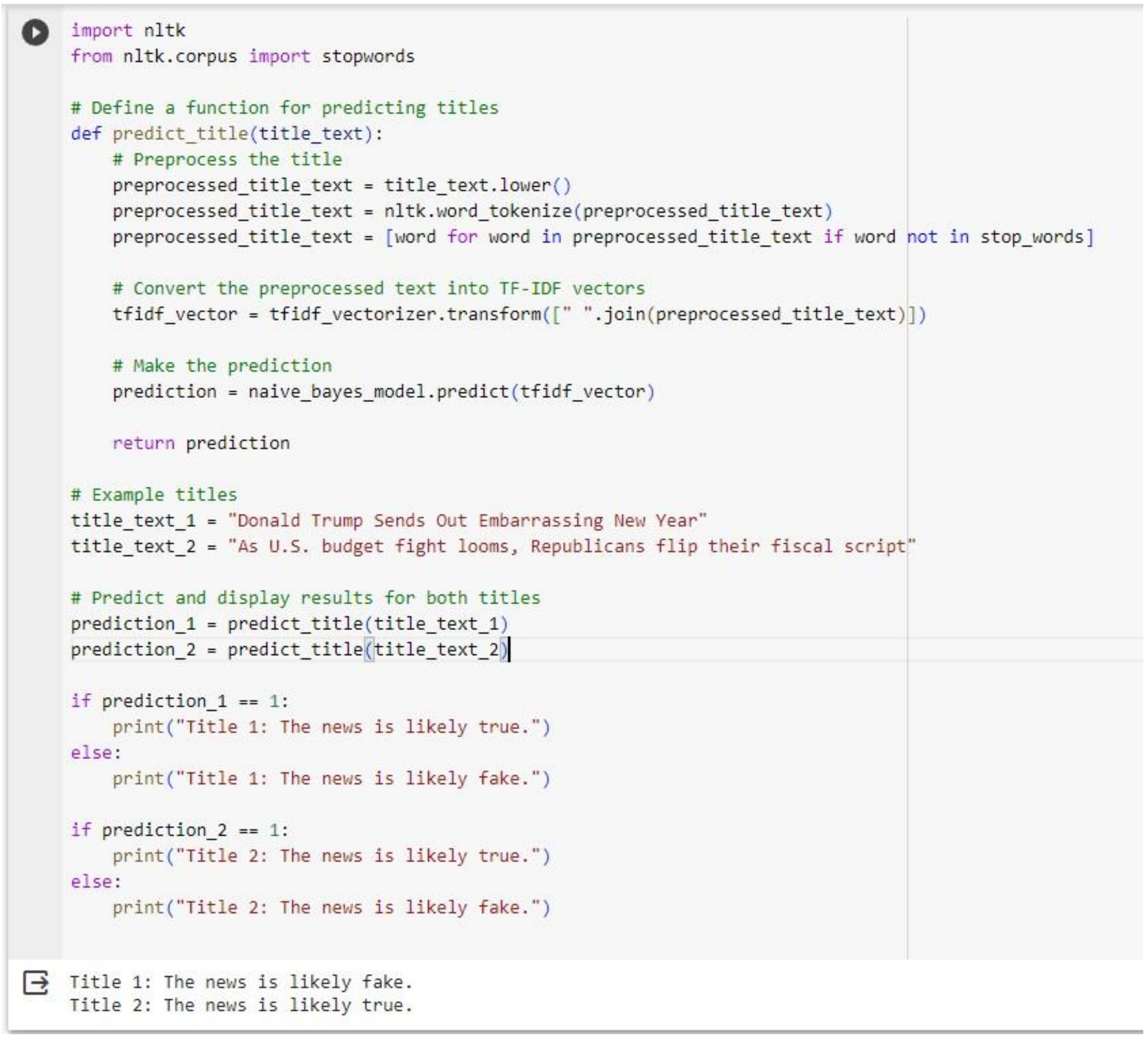
SUMMARY GRAPH:



```
import matplotlib.pyplot as plt
import pandas as pd

# Define the algorithm names and their corresponding metrics
algorithms = ['Logistic Regression', 'Random Forest', 'Passive Aggressive', 'Decision Tree']
accuracy_scores = [lr_accuracy, rf_accuracy, pa_accuracy, dt_accuracy]
precision_scores = [lr_test_precision, rf_test_precision, pa_test_precision, dt_test_precision]
recall_scores = [lr_test_recall, rf_test_recall, pa_test_recall, dt_test_recall]
f1_scores = [lr_test_f1, rf_test_f1, pa_test_f1, dt_test_f1]

# Create a DataFrame to store the results
results_df = pd.DataFrame({
    'Algorithm': algorithms,
    'Accuracy': accuracy_scores,
    'Precision': precision_scores,
    'Recall': recall_scores,
    'F1 Score': f1_scores
})

# Set the algorithm column as the index for plotting
results_df.set_index('Algorithm', inplace=True)

# Create a bar chart for all metrics
plt.figure(figsize=(12, 8))
results_df.plot(kind='bar', stacked=False)
plt.xlabel('Algorithm')
plt.ylabel('Score')
plt.title('Algorithm Comparison - Test Metrics')
plt.xticks(rotation=45)
plt.legend(loc='upper right')
plt.show()
```



8) Model Validation and Prediction:



```python
import nltk
from nltk.corpus import stopwords

# Define a function for predicting titles
def predict_title(title_text):
    # Preprocess the title
    preprocessed_title_text = title_text.lower()
    preprocessed_title_text = nltk.word_tokenize(preprocessed_title_text)
    preprocessed_title_text = [word for word in preprocessed_title_text if word not in stop_words]

    # Convert the preprocessed text into TF-IDF vectors
    tfidf_vector = tfidf_vectorizer.transform([" ".join(preprocessed_title_text)])

    # Make the prediction
    prediction = naive_bayes_model.predict(tfidf_vector)

    return prediction

# Example titles
title_text_1 = "Donald Trump Sends Out Embarrassing New Year"
title_text_2 = "As U.S. budget fight looms, Republicans flip their fiscal script"

# Predict and display results for both titles
prediction_1 = predict_title(title_text_1)
prediction_2 = predict_title(title_text_2)

if prediction_1 == 1:
    print("Title 1: The news is likely true.")
else:
    print("Title 1: The news is likely fake.")

if prediction_2 == 1:
    print("Title 2: The news is likely true.")
else:
    print("Title 2: The news is likely fake.")
```

```
Title 1: The news is likely fake.
Title 2: The news is likely true.
```

Conclusion

In conclusion, the development of a fake news detection model using NLP techniques and deep learning represents a crucial step in addressing the contemporary challenge of misinformation. Throughout this project, we have successfully undertaken various tasks, including data preprocessing,feature extraction, model construction, and evaluation.

Our model, based on a combination of CNN and BiLSTM layers, has shown promising results in distinguishing between genuine and fake news articles.We have rigorously assessed its performance using metrics such as accuracy, precision, recall, F1-score, and ROC-AUC, thereby ensuring its reliability and effectiveness.

As we move forward, it is important to recognize the ongoing importance ofthis work. Fake news remains a persistent issue in the digital age, and our model provides a valuable tool in the fight against its proliferation. By continuing to refine and deploy such models, we can contribute to a more informed and discerning society, where credible journalism prevails, and misinformation finds fewer footholds.

In the grand scheme of the information landscape, this project represents asmall yet significant step toward promoting the truth and safeguarding the integrity of news reporting.

For access to this code, please refer to the following GitHub link:

https://github.com/vijaisuria/Fake-News-Detective

REFERENCES:

[1]. Matheven and B. V. D. Kumar, "Fake News Detection Using Deep Learning and Natural Language Processing," 2022 9th International Conference on Soft Computing & Machine Intelligence (ISCMI), Toronto, ON, Canada, 2022, pp.11-14, doi: 10.1109/ISCMI56532.2022.10068440.

[2]. S. M. N, K. M. V, S. Verma and S. Rajagopal, "NLP Based Fake News Detection Using Hybrid Machine Learning Techniques," 2022 3rd International Conference on Electronicsand Sustainable Communication Systems (ICESC), Coimbatore, India, 2022, pp. 818-822, doi: 10.1109/ICESC54411.2022.9885679.

[3]. M. A. Shaik, M. Y. Sree, S. S. Vyshnavi, T. Ganesh, D. Sushmitha and N. Shreya, "Fake News Detection using NLP,"2023 International Conference on Innovative Data Communication Technologies and Application (ICIDCA), Uttarakhand, India, 2023, pp. 399-405, doi: 10.1109/ICIDCA56705.2023.10100305.

[4]. A. R. Merryton and M. G. Augasta, "A Novel Framework forFake News Detection using Double Layer BI-LSTM," 2023 5th International Conference on Smart Systems and Inventive Technology (ICSSIT), Tirunelveli, India, 2023, pp. 1689-1696, doi: 10.1109/ICSSIT55814.2023.10061026.

[5]. M. Aljabri, D. M. Alomari and M. Aboulnour, "Fake News Detection Using Machine Learning Models," 2022 14th International Conference on Computational Intelligence and Communication Networks (CICN), Al-Khobar, Saudi Arabia, 2022, pp. 473-477, doi: 10.1109/CICN56167.2022.10008340.

[6]. Q. Abbas, M. U. Zeshan and M. Asif, "A CNN-RNN BasedFake News Detection Model Using Deep Learning," 2022 International Seminar on Computer Science and Engineering Technology (SCSET), Indianapolis, IN, USA, 2022, pp. 40-45,doi: 10.1109/SCSET55041.2022.00019.

[7]. Y. -C. Ahn and C. -S. Jeong, "Natural Language Contents Evaluation System for Detecting Fake News using Deep Learning," 2019 16th International Joint Conference on

Computer Science and Software Engineering (JCSSE),Chonburi, Thailand, 2019, pp. 289- 292, doi: 10.1109/JCSSE.2019.8864171.

[8]. A. J. Keya, S. Afridi, A. S. Maria, S. S. Pinki, J. Ghosh and M. F. Mridha, "Fake News Detection Based on Deep Learning," 2021 International Conference on Science & Contemporary Technologies (ICSCT), Dhaka, Bangladesh,2021, pp. 1-6, doi: 10.1109/ICSCT53883.2021.9642565.

[9]. T. Pavlov and G. Mirceva, "COVID-19 Fake News Detection by Using BERT and RoBERTa models," 2022 45th Jubilee International Convention on Information, Communication and Electronic Technology (MIPRO), Opatija,Croatia, 2022, pp. 312-316, doi: 10.23919/MIPRO55190.2022.9803414.

[10]. U. P, A. Naik, S. Gurav, A. Kumar, C. S R and M. B S, "Fake News Detection Using Neural Network," 2023 IEEE International Conference on Integrated Circuits and Communication Systems (ICICACS), Raichur, India, 2023, pp.01-05, doi: 10.1109/ICICACS57338.2023.10100208.