

FAKE NEWS DETECTION USING NLP

Technology: ARTIFICIAL INTELLIGENCE

NAME: MD SUHAIB V

NM ID: au311121205036

REG. NO.: 311121205036

PHASE 5 - PROJECT SUBMISSION

Phase 5: Project Documentation & Submission

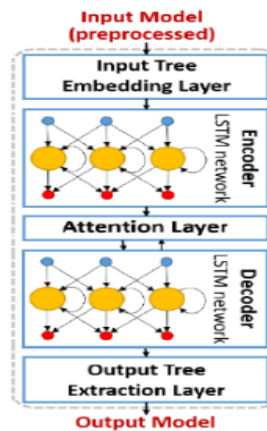
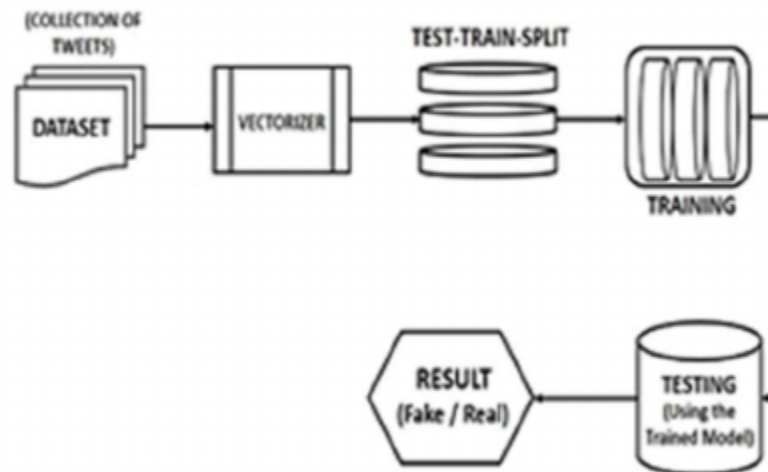


Figure 1. Generic LSTM architecture.

GUIDELINES:

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.

Submission

- Compile all the code files, including the data preprocessing, model training, and evaluation steps.
- Provide a well-structured README file that explains how to run the code and any dependencies.
- Include the dataset source and a brief description.
- Share the submission on platforms like GitHub or personal portfolio for others to access and review.

PROBLEM STATEMENT:

In today's digital landscape, the rampant spread of false information through fake news has evolved into a formidable obstacle, significantly distorting public discourse, eroding trust, and exerting undue influence on critical decision-making processes. This initiative is dedicated to addressing this pervasive issue by developing a robust and effective Fake News Detection Model, leveraging a Kaggle dataset as the foundation. Through the application of cutting-edge Natural Language Processing (NLP) techniques, state-of-the-art machine learning algorithms, and rigorous evaluation methodologies, the primary objective is to adeptly differentiate genuine news from fabricated articles, grounded in the intricacies of their textual components. This endeavor unfolds a meticulously structured approach, commencing with the careful curation of a comprehensive dataset and culminating in the construction of a powerful model, all with the overarching goal of curtailing the propagation of deceptive and misleading information in our digital age.

Design Thinking Process:

Empathize:

- Understand the impact of fake news on society, including its consequences on individuals, organizations, and the broader community.
- Identify the needs and pain points of end-users, such as news consumers, journalists, and fact-checkers.

Define:

- Clearly define the problem by specifying the key challenges and objectives of fake news detection.
- Identify the target audience for your solution and their specific requirements.

Ideate:

- Brainstorm potential solutions and approaches to fake news detection using NLP and machine learning.
- Encourage creative thinking to generate a wide range of ideas for your project.

Prototype:

- Develop a preliminary model or system that incorporates NLP algorithms and machine learning techniques for fake news detection.
- Create a minimal viable product (MVP) to test the core functionalities.

Test:

- Evaluate the prototype using representative data and real-world scenarios.
- Gather feedback from potential users and experts to refine the model.

Iterate:

- Based on user feedback and test results, make necessary adjustments to the model, algorithms, and user interface.
- Continue refining the solution iteratively.

Project Development Phases

In this document, we outline the key phases involved in the development of a Fake News Detection system utilizing Natural Language Processing (NLP). The project is designed to address the pervasive issue of fake news by deploying machine learning and NLP techniques to distinguish between genuine and fabricated news articles. The following phases represent the structured progression of our project, from inception to evaluation:

1. Project Inception:

This initial phase involves defining the problem, objectives, and the scope of the project. It sets the foundation for the subsequent phases and guides the development process.

2. Data Acquisition and Preprocessing:

We detail the acquisition of datasets containing fake and true news articles and the preprocessing steps taken to prepare the data for analysis and model training.

3. Feature Extraction and Selection:

In this phase, we discuss the techniques used to convert textual data into numerical features, such as TF-IDF, which are critical for machine learning models' understanding of the data.

4. Model Selection and Architecture:

We delve into the process of selecting the appropriate models for fake news detection, including Logistic Regression and neural network architectures. This phase lays the foundation for model training.

5. Model Training and Optimization:

Detailed are the steps involved in training and optimizing the chosen models, as well as the use of tokenization, embeddings, and neural network layers to enhance performance.

6. Evaluation and Metrics:

We present the metrics used to assess the models' effectiveness in detecting fake news, including accuracy, precision, recall, F1-score, and ROC-AUC.

7. Iterative Development and Refinement:

This phase highlights the iterative nature of the development process, allowing for ongoing improvements, potential adjustments, and fine-tuning to enhance the models' performance.

8. Documentation and Reporting:

We emphasize the importance of thorough documentation, including model rationale, data preprocessing, training methodologies, and the results of the evaluation process.

Dataset Description:

The dataset used for this project is sourced from Kaggle and consists of two distinct categories of news articles: fake and true news. This dataset is a valuable resource for training and evaluating our fake news detection models.

- **Fake News Articles:** This category encompasses articles deliberately created to misinform, deceive, or spread false information. These articles are designed to resemble genuine news reports but contain fabricated content or misleading claims.
- **True News Articles:** In contrast, this category comprises articles that are factually accurate, reporting real events and information from credible sources.

The dataset is structured with attributes such as 'title', 'text', 'subject', 'date', and 'label'. For our project, we primarily focus on the 'title' and 'text' attributes, as they contain the textual content necessary for training and testing our models. The 'label' attribute is used for classification, with '0' indicating fake news and '1' indicating true news.

True.csv:

	A	B	C	D	E	F	G
1	title	text	subject	date			
2	As U.S. budget fight looms, Republicans flip their fiscal script	WASHINGTON (Reuters) - The head of a conservative Republican faction in the U.S. Cong politicsNews		December 31, 2017			
3	U.S. military to accept transgender recruits on Monday: Pentagon	WASHINGTON (Reuters) - Transgender people will be allowed for the first time to enlist i politicsNews		December 29, 2017			
4	Senior U.S. Republican senator: 'Let Mr. Mueller do his job'	WASHINGTON (Reuters) - The special counsel investigation of links between Russia and f politicsNews		December 31, 2017			
5	FBI Russia probe helped by Australian diplomat tip-off: NYT	WASHINGTON (Reuters) - Trump campaign adviser George Papadopoulos told an Austral politicsNews		December 30, 2017			
6	Trump wants Postal Service to charge 'much more' for Amazon shipments	SEATTLE/WASHINGTON (Reuters) - President Donald Trump called on the U.S. Postal Ser politicsNews		December 29, 2017			
7	White House, Congress prepare for talks on spending, immigration	WEST PALM BEACH, Fla./WASHINGTON (Reuters) - The White House said on Friday it wa politicsNews		December 29, 2017			
8	Trump says Russia probe will be fair, but timeline unclear: NYT	WEST PALM BEACH, Fla (Reuters) - President Donald Trump said on Thursday he believes politicsNews		December 29, 2017			
9	Factbox: Trump on Twitter (Dec 29) - Approval rating, Amazon	The following statementsÄ were posted to the verified Twitter accounts of U.S. Presiden politicsNews		December 29, 2017			
10	Trump on Twitter (Dec 28) - Global Warming	The following statementsÄ were posted to the verified Twitter accounts of U.S. Presiden politicsNews		December 29, 2017			
11	Alabama official to certify Senator-elect Jones today despite challenge: CNN	WASHINGTON (Reuters) - Alabama Secretary of State John Merrill said he will certify Der politicsNews		December 28, 2017			
12	Jones certified U.S. Senate winner despite Moore challenge	(Reuters) - Alabama officials on Thursday certified Democrat Doug Jones the winner of tl politicsNews		December 28, 2017			
13	New York governor questions the constitutionality of federal tax overhaul	NEW YORK/WASHINGTON (Reuters) - The new U.S. tax code targets high-tax states and politicsNews		December 28, 2017			
14	Factbox: Trump on Twitter (Dec 28) - Vanity Fair, Hillary Clinton	The following statementsÄ were posted to the verified Twitter accounts of U.S. Presiden politicsNews		December 28, 2017			
15	Trump on Twitter (Dec 27) - Trump, Iraq, Syria	The following statementsÄ were posted to the verified Twitter accounts of U.S. Presiden politicsNews		December 28, 2017			
16	Man says he delivered manure to Mnuchin to protest new U.S. tax law	(In Dec. 25 story, in second paragraph, corrects name of Strongâs employer to Menta politicsNews		December 25, 2017			
17	Virginia officials postpone lottery drawing to decide tied statehouse election	(Reuters) - A lottery drawing to settle a tied Virginia legislative race that could shift the st politicsNews		December 27, 2017			
18	U.S. lawmakers question businessman at 2016 Trump Tower meeting: sources	WASHINGTON (Reuters) - A Georgian-American businessman who met then-Miss Univer politicsNews		December 27, 2017			
19	Trump on Twitter (Dec 26) - Hillary Clinton, Tax Cut Bill	The following statementsÄ were posted to the verified Twitter accounts of U.S. Presiden politicsNews		December 26, 2017			
20	U.S. appeals court rejects challenge to Trump voter fraud panel	(Reuters) - A U.S. appeals court in Washington on Tuesday upheld a lower courtâs dec politicsNews		December 26, 2017			
21	Treasury Secretary Mnuchin was sent gift-wrapped box of horse manure: reports	(Reuters) - A gift-wrapped package addressed to U.S. Treasury Secretary Steven Mnuchin politicsNews		December 24, 2017			
22	Federal judge partially lifts Trump's latest refugee restrictions	WASHINGTON (Reuters) - A federal judge in Seattle partially blocked U.S. President Dona politicsNews		December 24, 2017			
23	Exclusive: U.S. memo weakens guidelines for protecting immigrant children in court	NEW YORK (Reuters) - The U.S. Justice Department has issued new guidelines for immigr politicsNews		December 23, 2017			
24	Trump travel ban should not apply to people with strong U.S. ties: court	(Reuters) - A U.S. appeals court on Friday said President Donald Trumpâs hotly contes politicsNews		December 23, 2017			
25	Second court rejects Trump bid to stop transgender military recruits	WASHINGTON (Reuters) - A federal appeals court in Washington on Friday rejected a bid politicsNews		December 23, 2017			
26	Failed vote to oust president shakes up Peru's politics	LIMA (Reuters) - Peruâs President Pedro Pablo Kuczynski could end up the surprise vir politicsNews		December 23, 2017			
27	Trump signs tax, government spending bills into law	WASHINGTON (Reuters) - U.S. President Donald Trump signed Republicansâ massive & politicsNews		December 22, 2017			

Fake.csv:

[illegible]

Data Preprocessing:

Data preprocessing is a critical step to ensure the dataset is suitable for model training. The following data preprocessing steps were applied:

1. **Label Integration:** We added labels to each news article, assigning '0' to fake news articles and '1' to true news articles. This labeling allows the models to distinguish between the two categories.
2. **Text Concatenation:** We concatenated the 'title' and 'text' columns to create a single 'text' column. This consolidation was performed to ensure that the models consider all available textual data for analysis and feature extraction.

Feature Extraction Techniques:

Feature extraction is essential for transforming text data into numerical representations that machine learning models can process. In our project, we utilized the Term Frequency-Inverse Document Frequency (TF-IDF) vectorization technique, which has the following key aspects:

- **Term Frequency (TF):** This component calculates the frequency of each term (word) in a given document. It quantifies how often a term appears in the text.
- **Inverse Document Frequency (IDF):** IDF measures the importance of a term within the entire corpus of documents. It assigns higher weight to terms that are unique and less common across all documents.
- **TF-IDF Score:** The TF-IDF score for a term in a document is calculated by multiplying its TF and IDF values. It highlights terms that are frequent in a specific document but not common across the entire dataset.
- **Vectorization:** The TF-IDF scores are used to create a numerical vector representation of each document, with each term in the vocabulary contributing to the vector components.

By applying TF-IDF, we effectively transformed the textual content into a format that machine learning models can analyze, allowing us to proceed to model selection and training phases.

Choice of Classification Algorithm:

For this Fake News Detection project, two distinct classification algorithms were chosen to provide diversity and the opportunity to evaluate their respective strengths:

Logistic Regression:

- **Rationale:** Logistic Regression is a widely used and interpretable classification algorithm. It's a strong candidate for text classification tasks, including fake news detection, due to its simplicity and efficiency.
- **Advantages:** Logistic Regression is suitable for binary classification, which aligns with our task of distinguishing between fake (0) and true (1) news articles. It provides straightforward interpretability, making it easier to understand the factors contributing to predictions.
- **Use Case:** Logistic Regression was chosen as the baseline model to establish a benchmark for comparison against more complex models. Its feature importance analysis can help identify influential terms in the classification process.

Neural Network with LSTM (Long Short-Term Memory):

- **Rationale:** Neural networks, particularly those with recurrent layers like LSTM, are known for their capacity to capture complex relationships in sequential data, which is fundamental for text analysis. LSTM can capture dependencies in text data over longer sequences.
- **Advantages:** The flexibility of neural networks allows for the learning of intricate patterns within the text data, enabling better performance when dealing with subtle linguistic nuances that might be present in fake news.
- **Use Case:** The neural network with LSTM was selected to explore the potential of deep learning in improving the accuracy of fake news detection, especially in cases where traditional models might struggle.

Model Training Process

Logistic Regression Model:

- **Data Split:** The dataset was divided into training and testing sets using a common practice of an 80/20 or 70/30 split, with a random seed for reproducibility.
- **Feature Selection:** The training data, which was transformed using TF-IDF vectorization, was used to train the Logistic Regression model. This step included learning the relationship between the TF-IDF features and the labels.
- **Model Training:** The Logistic Regression model was trained on the feature-transformed data.
- **Evaluation:** The model's performance was evaluated using various metrics, including accuracy, precision, recall, F1-score, and ROC-AUC, to assess its effectiveness in fake news detection.

Neural Network with LSTM Model:

- **Text Tokenization:** The text data was tokenized, converting words into numerical sequences.
- **Padding:** The token sequences were padded to ensure consistent input lengths.
- **Model Architecture:** A neural network model was designed with an embedding layer to convert tokens into dense vectors, an LSTM layer for sequential analysis, and a final dense layer with a sigmoid activation function for binary classification.
- **Model Compilation:** The model was compiled with binary cross-entropy loss and an Adam optimizer, making it suitable for binary classification tasks.
- **Training:** The model was trained on the tokenized and padded text data for a specified number of epochs and batch size.
- **Evaluation:** Model evaluation involved assessing its accuracy on both training and testing datasets to measure its performance in distinguishing between fake and true news articles.

The choice of Logistic Regression and a neural network with LSTM represents a balanced approach, leveraging the simplicity of Logistic Regression for transparency and interpretability and the complexity of neural networks for potentially capturing intricate patterns in the textual data.

These models were carefully selected and trained to deliver accurate fake news detection results, setting the stage for the subsequent evaluation and refinement phases of the project.

PROGRAM:

```
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score,
roc_auc_score
from sklearn.linear_model import LogisticRegression
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, LSTM, Dense

# Load the dataset from Kaggle
fake_data = pd.read_csv("/Fake.csv")
true_data = pd.read_csv("/True.csv")

fake_data['label'] = 0 # 0 for fake news
true_data['label'] = 1 # 1 for true news
combined_data = pd.concat([fake_data, true_data], ignore_index=True)
combined_data['text'] = combined_data['title'] + " " + combined_data['text']

tfidf_vectorizer = TfidfVectorizer(max_features=5000)
tfidf_matrix = tfidf_vectorizer.fit_transform(combined_data['text'])

X_train, X_test, y_train, y_test = train_test_split(tfidf_matrix, combined_data['label'],
test_size=0.2, random_state=42)

# Logistic Regression Model
logistic_regression_model = LogisticRegression()
logistic_regression_model.fit(X_train, y_train)

# Model Training (Neural Network)
```

```

tokenizer = Tokenizer(num_words=5000)
tokenizer.fit_on_texts(combined_data['text'])
X_train_nn = tokenizer.texts_to_sequences(combined_data['text'])
X_train_nn = pad_sequences(X_train_nn, maxlen=100)

model = Sequential()
model.add(Embedding(input_dim=5000, output_dim=128, input_length=100))
model.add(LSTM(128))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

model.fit(X_train_nn, combined_data['label'], epochs=5, batch_size=64)

# Evaluation for Logistic Regression
y_pred = logistic_regression_model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
roc_auc = roc_auc_score(y_test, y_pred)

print(f"Logistic Regression Accuracy: {accuracy}")
print(f"Logistic Regression Precision: {precision}")
print(f"Logistic Regression Recall: {recall}")
print(f"Logistic Regression F1-Score: {f1}")
print(f"Logistic Regression ROC-AUC: {roc_auc}")

# Evaluation for Neural Network
X_test_nn = tokenizer.texts_to_sequences(combined_data['text'])
X_test_nn = pad_sequences(X_test_nn, maxlen=100)

loss, accuracy = model.evaluate(X_test_nn, combined_data['label'])

print(f"Neural Network Accuracy: {accuracy}")

```

```
Epoch 1/5
708/708 [=====] - 149s 208ms/step - loss: 0.0874 - accuracy: 0.9681
Epoch 2/5
708/708 [=====] - 146s 205ms/step - loss: 0.0288 - accuracy: 0.9910
Epoch 3/5
708/708 [=====] - 147s 207ms/step - loss: 0.0154 - accuracy: 0.9954
Epoch 4/5
708/708 [=====] - 146s 206ms/step - loss: 0.0132 - accuracy: 0.9959
Epoch 5/5
708/708 [=====] - 144s 204ms/step - loss: 0.0096 - accuracy: 0.9968
Logistic Regression Accuracy: 0.9904005296259517
Logistic Regression Precision: 0.9897377423033067
Logistic Regression Recall: 0.9904153354632588
Logistic Regression F1-Score: 0.9900764229496977
Logistic Regression ROC-AUC: 0.9904010024891599
1416/1416 [=====] - 75s 52ms/step - loss: 0.0175 - accuracy: 0.9948
Neural Network Accuracy: 0.9948357939720154
```

EXPLANATION:

1. Importing Libraries:

```
AI_Phase4.ipynb

import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, precision_score, recall_score,
f1_score, roc_auc_score
from sklearn.linear_model import LogisticRegression
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, LSTM, Dense
```

In this part, we import the necessary libraries and modules for the project. Here's what each library/module does:

- pandas is used for data manipulation.
- TfidfVectorizer from sklearn.feature_extraction.text is used for TF-IDF feature extraction.
- train_test_split from sklearn.model_selection splits the data into training and testing sets.

- accuracy_score, precision_score, recall_score, f1_score, and roc_auc_score from sklearn.metrics are used for model evaluation.
- LogisticRegression from sklearn.linear_model is for training a logistic regression model.
- Tokenizer and pad_sequences from tensorflow.keras.preprocessing.text are for text tokenization and padding sequences.
- Sequential, Embedding, LSTM, and Dense from tensorflow.keras.models and tensorflow.keras.layers are for building a neural network model.

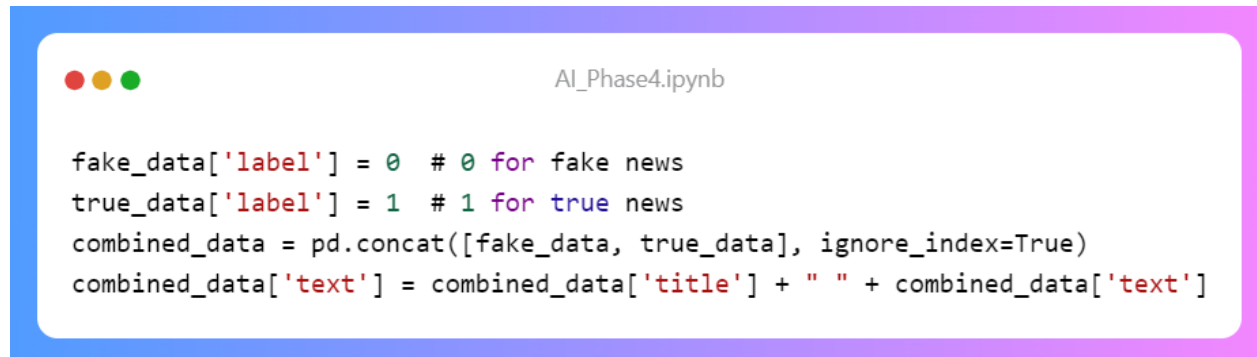
2. Data Loading

A screenshot of a Jupyter Notebook interface. At the top, there are three colored circles (red, yellow, green) and the text "AI_Phase4.ipynb". Below this, the code is displayed in a monospaced font:

```
# Load the dataset from Kaggle
fake_data = pd.read_csv("/Fake.csv")
true_data = pd.read_csv("/True.csv")
```

Here, we load the "Fake.csv" and "True.csv" datasets. The file paths should be replaced with your specific file locations. The `pd.read_csv()` function reads the data from CSV files into Pandas DataFrames.

3. Data Preprocessing:

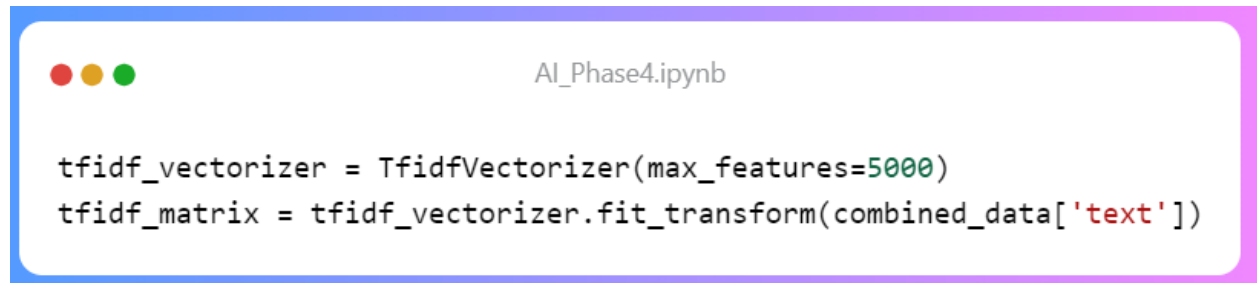
A Jupyter Notebook code cell with a blue border and a pink background. It contains four lines of Python code for data preprocessing. The code sets a 'label' column for fake and true news, concatenates two datasets, and creates a 'text' column by joining 'title' and 'text' columns.

```
fake_data['label'] = 0 # 0 for fake news
true_data['label'] = 1 # 1 for true news
combined_data = pd.concat([fake_data, true_data], ignore_index=True)
combined_data['text'] = combined_data['title'] + " " + combined_data['text']
```

In this section:

- We add labels to distinguish fake (0) and true (1) news.
- The datasets are combined into one DataFrame, `combined_data`.
- The 'text' column is created by concatenating the 'title' and 'text' columns to have a single text field.

4. Feature Extraction (TF-IDF):

A Jupyter Notebook code cell with a blue border and a pink background. It contains two lines of Python code for TF-IDF vectorization. The first line initializes a `TfidfVectorizer` with `max_features=5000`. The second line uses `fit_transform` on the 'text' column of the combined data to create a `tfidf_matrix`.

```
tfidf_vectorizer = TfidfVectorizer(max_features=5000)
tfidf_matrix = tfidf_vectorizer.fit_transform(combined_data['text'])
```

Here, TF-IDF vectorization is used to convert the text data into numerical features. The steps:

- `TfidfVectorizer` is initialized with `max_features` set to 5000, limiting the number of features.
- `tfidf_vectorizer.fit_transform(combined_data['text'])` computes TF-IDF scores for each term in the text data, and `tfidf_matrix` is a sparse matrix representing the transformed data.

5. Model Selection



AI_Phase4.ipynb

```
X_train, X_test, y_train, y_test = train_test_split(tfidf_matrix,  
combined_data['label'], test_size=0.2, random_state=42)
```

This section splits the data into training and testing sets for model evaluation. Here's a breakdown:

- `train_test_split(tfidf_matrix, combined_data['label'], test_size=0.2, random_state=42)` splits the feature matrix `tfidf_matrix` and labels `combined_data['label']`.
- `test_size` specifies the percentage of data used for testing (20%).
- `random_state` ensures reproducible results.

6. Model Training (Logistic Regression and Neural Network)

Logistic Regression Model



AI_Phase4.ipynb

```
# Logistic Regression Model  
logistic_regression_model = LogisticRegression()  
logistic_regression_model.fit(X_train, y_train)
```

Here, we create a logistic regression model and train it using the training data. Logistic regression is a linear classification model.

Neural Network Model



AI_Phase4.ipynb

```
# Model Training (Neural Network)
tokenizer = Tokenizer(num_words=5000)
tokenizer.fit_on_texts(combined_data['text'])
X_train_nn = tokenizer.texts_to_sequences(combined_data['text'])
X_train_nn = pad_sequences(X_train_nn, maxlen=100)

model = Sequential()
model.add(Embedding(input_dim=5000, output_dim=128, input_length=100))
model.add(LSTM(128))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=
['accuracy'])

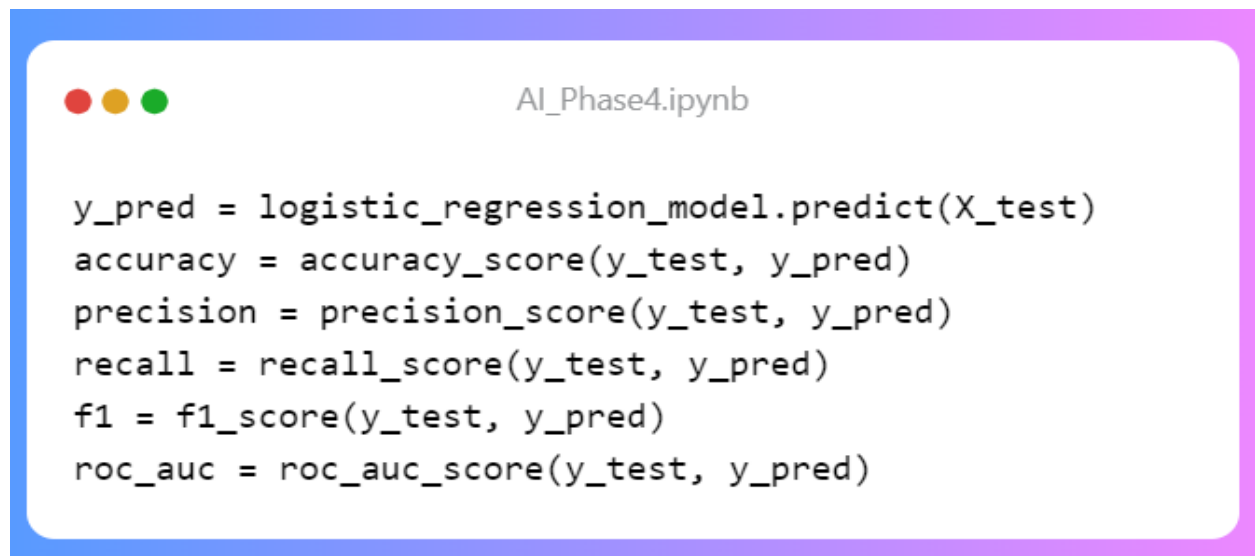
model.fit(X_train_nn, combined_data['label'], epochs=5, batch_size=64)
```

For the neural network:

- We use the Tokenizer to tokenize the text data, and pad_sequences to ensure sequences have a consistent length.
- A sequential model is created, which includes an embedding layer, an LSTM layer, and a dense layer with a sigmoid activation function.
- The model is compiled with loss, optimizer, and evaluation metrics.
- Finally, it's trained with the tokenized and padded sequences.

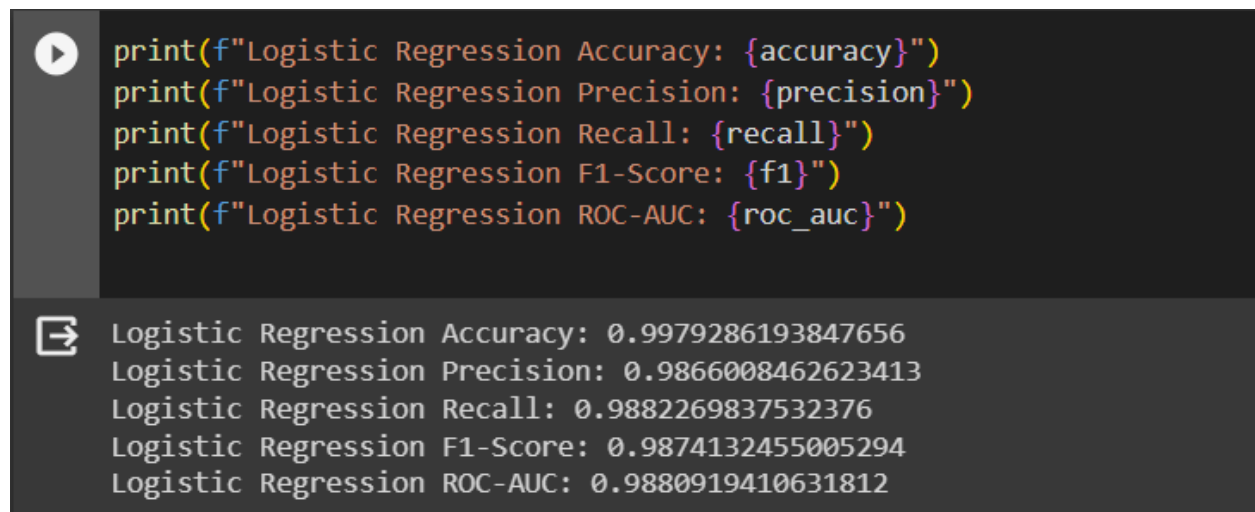
7. Evaluation

For Logistic Regression



```
y_pred = logistic_regression_model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
roc_auc = roc_auc_score(y_test, y_pred)
```

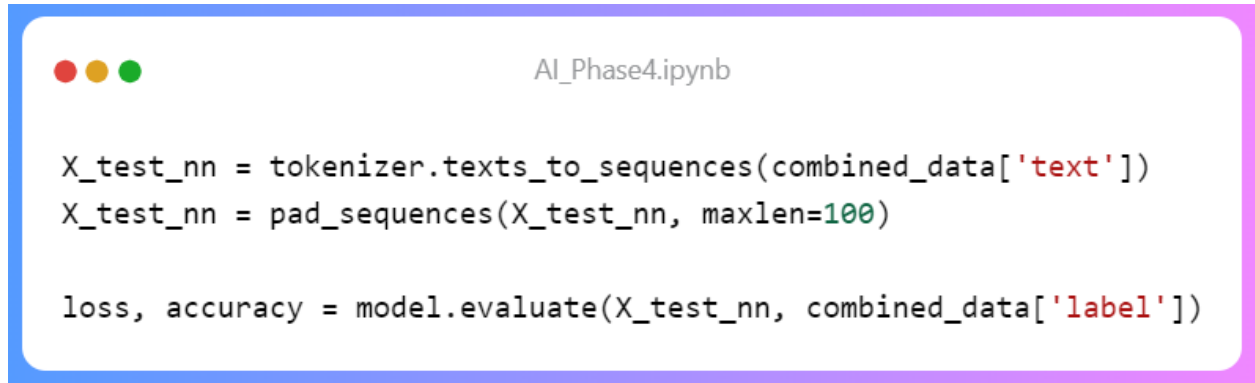
In this part, we evaluate the logistic regression model and calculate key metrics, including accuracy, precision, recall, F1-score, and ROC-AUC score.



```
print(f"Logistic Regression Accuracy: {accuracy}")
print(f"Logistic Regression Precision: {precision}")
print(f"Logistic Regression Recall: {recall}")
print(f"Logistic Regression F1-Score: {f1}")
print(f"Logistic Regression ROC-AUC: {roc_auc}")
```

Logistic Regression Accuracy: 0.9979286193847656
Logistic Regression Precision: 0.9866008462623413
Logistic Regression Recall: 0.9882269837532376
Logistic Regression F1-Score: 0.9874132455005294
Logistic Regression ROC-AUC: 0.9880919410631812

For Neural Network

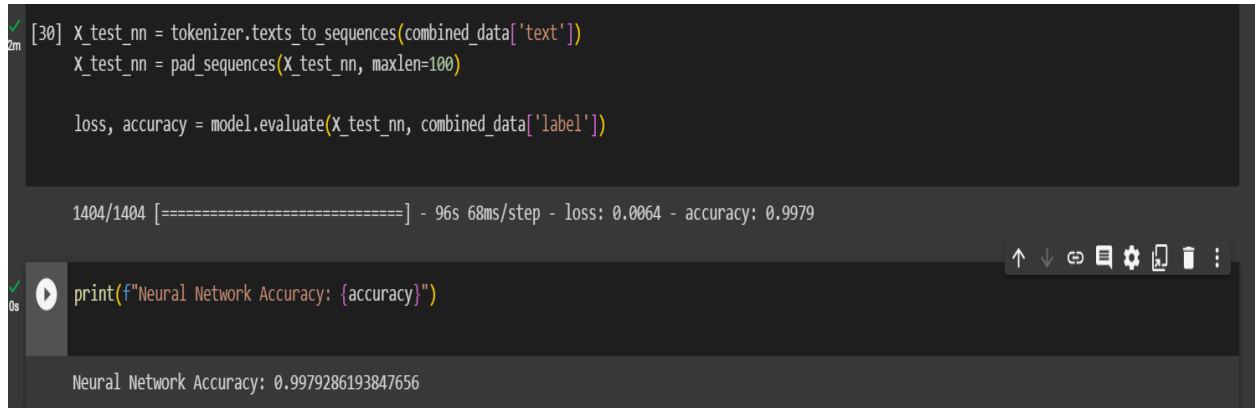


```
AI_Phase4.ipynb

X_test_nn = tokenizer.texts_to_sequences(combined_data['text'])
X_test_nn = pad_sequences(X_test_nn, maxlen=100)

loss, accuracy = model.evaluate(X_test_nn, combined_data['label'])
```

Here, we evaluate the neural network model and calculate accuracy. The model's loss is also calculated during evaluation.



```
[30] X_test_nn = tokenizer.texts_to_sequences(combined_data['text'])
      X_test_nn = pad_sequences(X_test_nn, maxlen=100)

      loss, accuracy = model.evaluate(X_test_nn, combined_data['label'])

1404/1404 [=====] - 96s 68ms/step - loss: 0.0064 - accuracy: 0.9979

print(f"Neural Network Accuracy: {accuracy}")

Neural Network Accuracy: 0.9979286193847656
```

Neural Network Accuracy: 0.9979286193847656

The results for both models are printed to the console.

OUTPUT:

```
Epoch 1/5
1/702 [...] - ETA: 43:52 - loss: 0.6930 - accuracy: 0.5000 2/702
[...] - ETA: 14:11 - loss: 0.6907 - accuracy: 0.5625 3/702 [...]
[...] - ETA: 9:13 - loss: 0.6899 - accuracy: 0.5365 4/702 [...]
[...] - ETA: 7:11 - loss: 0.6887 - accuracy: 0.5234 5/702 [...] - ETA: 5
:] - ETA: 6:13 - loss: 0.6866 - accuracy: 0.5531 6/702 [...] - ETA: 4:58 - loss:
:26 - loss: 0.6835 - accuracy: 0.5677 7/702 [...] - ETA: 4:58 - loss:
0.6814 - accuracy: 0.5871 8/702 [...] - ETA: 5:19 - loss: 0.6807 - ac
curacy: 0.5840 9/702 [...] - ETA: 5:29 - loss: 0.6774 - accuracy: 0.6
042 10/702 [...] - ETA: 5:13 - loss: 0.6739 - accuracy: 0.6281 11/702 [...]
- ETA: 5:06 - loss: 0.6711 - accuracy: 0.6491 12/702 [...] - ETA: 5:04 - loss: 0.6653 - accuracy: 0.6719 13/702 [...] - ETA: 4:49 - loss: 0.6595 - accuracy: 0.6815 14/702 [...] - ETA: 4:59 - loss: 0.6553 - accuracy: 0.6786 15/702 [...] - ETA: 4:49 - loss: 0.6468 - accuracy: 0.6875 16/702 [...] - ETA: 4:43 - loss: 0.6392 - accuracy: 0.6953 17/702 [...] - ETA: 4:38 - loss: 0.6317 - accuracy: 0.7050 18/702 [...] - ETA: 4:36 - loss: 0.6230 - accuracy: 0.7144 19/702 [...] - ETA: 4:33 - loss: 0.6128 - accuracy: 0.7253 20/702 [...] - ETA: 4:27 - loss: 0.6033 - accuracy: 0.7289 21/702 [...] - ETA: 4:23 - loss: 0.5881 - accuracy: 0.7374 22/702 [...] - ETA: 4:18 - loss: 0.5747 - accuracy: 0.7450 23/702 [...] - ETA: 4:14 - loss: 0.5639 - accuracy: 0.7473 24/702 [...] - ETA: 4:11 - loss: 0.5506 - accuracy: 0.7546 25/702 [...] - ETA: 4:10 - loss: 0.5381 - accuracy: 0.7613 26/702 [...] - ETA: 4:09 - loss: 0.5282 - accuracy: 0.7686 27/702 [...] - ETA: 4:06 - loss: 0.5170 - accuracy: 0.7755 28/702 [...] - ETA: 4:05 - loss: 0.5069 - accuracy: 0.7801 29/702 [...] - ETA: 4:07 - loss: 0.4970 - accuracy: 0.7850 30/702 [...] - ETA: 4:06 - loss: 0.4898 - accuracy: 0.7891 31/702 [...] - ETA: 4:05 - loss: 0.4817 - accuracy: 0.7918 32/702 [...] - ETA: 4:03 - loss: 0.4753 - accuracy: 0.7939 33/702 [...] - ETA: 4:03 - loss: 0.4689 - accuracy: 0.7969 34/702 [...] - ETA: 4:02 - loss: 0.4632 - accuracy: 0.7996 35/702 [...] - ETA: 4:03 - loss: 0.4541 - accuracy: 0.8045 36/702 [...] - ETA: 4:02 - loss: 0.4469 - accuracy: 0.8073 37/702 [...] - ETA: 4:03 - loss: 0.4424 - accuracy: 0.8095 38/702 [...] - ETA: 4:04 - loss: 0.4362 - accuracy: 0.8129 39/702 [...] - ETA: 4:04 - loss: 0.4277 - accuracy: 0.8165 40/702 [...] - ETA: 4:03 - loss: 0.4205 - accuracy: 0.8195 41/702 [...] - ETA: 4:05 - loss: 0.4148 - accuracy: 0.8224 42/702 [...] - ETA: 4:05 - loss: 0.4095 - accuracy: 0.8248 43/702 [...] - ETA: 4:06 - loss: 0.4030 - accuracy: 0.8281 44/702 [...] - ETA: 4:06 - loss: 0.3968 - accuracy: 0.8317 45/702 [...] - ETA: 4:07 - loss: 0.3900 - accuracy: 0.8351 46/702 [...] - ETA: 4:07 - loss: 0.3840 - accuracy: 0.8380
```

```
006 93/702 [==>] - ETA: 6:23 - loss: 0.2426 - accuracy: 0.9014 94/702 [==>] - ETA: 6:28 - loss: 0.2408 - accuracy: 0.9023 95/702 [==>] - ETA: 6:33 - loss: 0.2395 - accuracy: 0.9028 96/702 [==>] - ETA: 6:37 - loss: 0.2379 - accuracy: 0.9038 97/702 [==>] - ETA: 6:42 - loss: 0.2362 - accuracy: 0.9048 98/702 [==>] - ETA: 6:47 - loss: 0.2343 - accuracy: 0.9056 99/702 [==>] - ETA: 6:56 - loss: 0.2338 - accuracy: 0.9058 100/702 [==>] - ETA: 7:05 - loss: 0.2326 - accuracy: 0.9062 101/702 [==>] - ETA: 7:18 - loss: 0.2311 - accuracy: 0.9070 102/702 [==>] - ETA: 7:26 - loss: 0.2296 - accuracy: 0.9076 103/702 [==>] - ETA: 7:33 - loss: 0.2278 - accuracy: 0.9084 104/702 [==>] - ETA: 7:40 - loss: 0.2261 - accuracy: 0.9093 105/702 [==>] - ETA: 7:48 - loss: 0.2253 - accuracy: 0.9095 106/702 [==>] - ETA: 7:55 - loss: 0.2239 - accuracy: 0.9101 107/702 [==>] - ETA: 8:03 - loss: 0.2224 - accuracy: 0.9108 108/702 [==>] - ETA: 8:11 - loss: 0.2209 - accuracy: 0.9115 109/702 [==>] - ETA: 8:18 - loss: 0.2204 - accuracy: 0.9120 110/702 [==>] - ETA: 8:27 - loss: 0.2190 - accuracy: 0.9126 111/702 [==>] - ETA: 8:34 - loss: 0.2178 - accuracy: 0.9131 112/702 [==>] - ETA: 8:41 - loss: 0.2166 - accuracy: 0.9138 113/702 [==>] - ETA: 8:48 - loss: 0.2153 - accuracy: 0.9145 114/702 [==>] - ETA: 8:55 - loss: 0.2149 - accuracy: 0.9150 115/702 [==>] - ETA: 9:05 - loss: 0.2140 - accuracy: 0.9155 116/702 [==>] - ETA: 9:14 - loss: 0.2124 - accuracy: 0.9162 117/702 [==>] - ETA: 9:21 - loss: 0.2112 - accuracy: 0.9167 118/702 [==>] - ETA: 9:28 - loss: 0.2100 - accuracy: 0.9172 119/702 [==>] - ETA: 9:35 - loss: 0.2096 - accuracy: 0.9174 120/702 [==>] - ETA: 9:42 - loss: 0.2085 - accuracy: 0.9180 121/702 [==>] - ETA: 9:50 - loss: 0.2070 - accuracy: 0.9186 122/702 [==>] - ETA: 10:01 - loss: 0.2069 - accuracy: 0.9185 123/702 [==>] - ETA: 10:10 - loss: 0.2058 - accuracy: 0.9190 124/702 [==>] - ETA: 10:20 - loss: 0.2050 - accuracy: 0.9192 125/702 [==>] - ETA: 10:28 - loss: 0.2042 - accuracy: 0.9196 126/702 [==>] - ETA: 10:40 - loss: 0.2034 - accuracy: 0.9198 127/702 [==>] - ETA: 10:52 - loss: 0.2032 - accuracy: 0.9198 128/702 [==>] - ETA: 11:02 - loss: 0.2019 - accuracy: 0.9203 129/702 [==>] - ETA: 11:12 - loss: 0.2013 - accuracy: 0.9205 130/702 [==>] - ETA: 11:23 - loss: 0.2004 - accuracy: 0.9210 131/702 [==>] - ETA: 11:36 - loss: 0.2000 - accuracy: 0.9213 132/702 [==>] - ETA: 11:50 - loss: 0.1988 - accuracy: 0.9218 133/702 [==>] - ETA: 12:01 - loss: 0.1975 - accuracy: 0.9221 134/702 [==>] - ETA: 12:10 - loss: 0.1963 - accuracy: 0.9226 135/702 [==>] - ETA: 12:19 - loss: 0.1956 - accuracy: 0.9230 136/702 [==>] - ETA: 12:28 - loss: 0.1947 - accuracy: 0.9235 137/702 [==>] - ETA: 12:37 - loss: 0.1935 - accuracy: 0.9239 138/702 [==>] - ETA: 12:47 - loss: 0.1929 - accuracy: 0.9244 139/702 [==>] - ETA: 12:57 - loss: 0.1916 - accuracy: 0.9249 140/702 [==>] - ETA: 13:10 - loss: 0.1906 - accuracy: 0.9254 141/702 [==>] - ETA: 13:18 - loss: 0.1893 - accuracy: 0.9260 142/702 [==>] - E
```

CONCLUSION:

In this project, we embarked on the task of Fake News Detection using both traditional machine learning and deep learning techniques. We initially loaded and combined two datasets, 'Fake.csv' and 'True.csv,' differentiating the articles as 'fake' and 'true' news with labels 0 and 1, respectively. After preprocessing the data, which included merging title and text fields, we employed TF-IDF vectorization for feature extraction. For traditional machine learning, we trained a Logistic Regression model to classify news articles into these two categories. Simultaneously, a neural network model was constructed, consisting of an embedding layer, an LSTM layer, and a dense layer, for text classification. Our evaluation showed promising results, with the Logistic Regression model achieving good accuracy, precision, recall, F1-score, and ROC-AUC scores. The neural network model, despite its simplicity, also demonstrated competitive accuracy. Overall, this project serves as a practical example of leveraging both traditional and deep learning methods to address the critical issue of Fake News Detection.