Phase 3: True and Fake News detection using Natural Language Processing

Significant Program:

```
import pandas as pd
import numpy as np
import nltk
from nltk.corpus import stopwords
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix
# Load the dataset
file_path = '/content/true.csv'
data = pd.read_csv(file_path, encoding='latin-1')
# Drop unnecessary columns
data.drop(columns=['Freelance', 'Local/Foreign', 'Source of Fire', 'Impunity
(for Murder)', 'Taken Captive', 'Threatened', 'Tortured'], inplace=True)

# Handle missing values
data.fillna('', inplace=True)  # Fill missing values with empty strings

# Text preprocessing function
def preprocess_text(text):
    text = text.lower()  # Convert to lowercase
    text = ''.join(char for char in text if char.isalnum() or char.isspace())  #
Remove punctuation
    return text

# Apply preprocessing to relevant text columns
data['Job'] = data['Job'].apply(preprocess_text)
data['Organization'] = data['Organization'].apply(preprocess_text)
# Combine text features into a single feature
data['text'] = data['Job'] + ' ' + data['Organization']

# Define features and target variable
X = data['text']
y = data['Type of Death']  # Assuming 'Type of Death' is the target variable
# Initialize TF-IDF Vectorizer
tfidf_vectorizer = TfidfVectorizer(stop_words='english')
```

```
# Fit and transform the text data
X_tfidf = tfidf_vectorizer.fit_transform(X)
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X_tfidf, y, test_size=0.2,
random_state=42)
# Initialize and train the model
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
# Make predictions
y_pred = model.predict(X_test)

# Evaluate the model
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
import joblib

# Save the model to a file
joblib.dump(model, 'journalist_murder_model_nlp.pkl')
```

Obtained Output:

```
<ipython-input-3-4d5415384d3d>:16: FutureWarning: Setting an item of incompatible
dtype is deprecated and will raise an error in a future version of pandas. Value '' has dtype
incompatible with float64, please explicitly cast to a compatible dtype first.
  data.fillna('', inplace=True)  # Fill missing values with empty strings
[[110   0   0   0   0   0   0]
 [  0   0   3   0   3   0   0]
 [  0   0  34   1  22   0   0]
 [  0   0  11   4  14   0   0]
 [  0   0  21   1 148   0   0]
 [  0   0   0   0   0   1   0]
 [  0   0   0   0   0   0   3]]
                    precision    recall  f1-score   support

                         1.00      1.00      1.00       110
             Crossfire       0.00      0.00      0.00         6
Crossfire/Combat-Related       0.49      0.60      0.54        57
   Dangerous Assignment       0.67      0.14      0.23        29
                Murder       0.79      0.87      0.83       170
         Type of Death       1.00      1.00      1.00         1
                    na       1.00      1.00      1.00         3
```

accuracy			0.80	376
macro avg	0.71	0.66	0.66	376
weighted avg	0.79	0.80	0.78	376