AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

Faculty of Science and Technology

Project Cover Page

Assignment Title:	Fake News Detector using Machine Learning			
Assignment No:	01		Date of Submission:	18 Jan 2025
Course Title:	PROGRAMMING IN PYTHON			
Course Code:	CSC4162		Section:	A
Semester:	Spring	2024-25	Course Teacher:	DR. ABDUS SALAM

Declaration and Statement of Authorship:

1. I/we hold a copy of this Assignment/Case-Study, which can be produced if the original is lost/damaged.
2. This Assignment/Case-Study is my/our original work and no part of it has been copied from any other student's work or from any other source except where due acknowledgement is made.
3. No part of this Assignment/Case-Study has been written for me/us by any other person except where such collaborationhas been authorized by the concerned teacher and is clearly acknowledged in the assignment.
4. I/we have not previously submitted or currently submitting this work for any other course/unit.
5. This work may be reproduced, communicated, compared and archived for the purpose of detecting plagiarism.
6. I/we give permission for a copy of my/our marked work to be retained by the Faculty for review and comparison, including review by external examiners.
7. I/we understand thatPlagiarism is the presentation of the work, idea or creation of another person as though it is your own. It is a formofcheatingandisaveryseriousacademicoffencethatmayleadtoexpulsionfromtheUniversity. Plagiarized material can be drawn from, and presented in, written, graphic and visual form, including electronic data, and oral presentations. Plagiarism occurs when the origin of them arterial used is not appropriately cited.
8. I/we also understand that enabling plagiarism is the act of assisting or allowing another person to plagiarize or to copy my/our work.

* *Student(s) must complete all details except the faculty use part.*
** Please submit all assignments to your course teacher or the office of the concerned teacher.

Group Name/No.: 08

No	Name	ID	Program	Signature
1	Shakil Mahmud	20-44216-3	BSc [CSE]	
2	MD Foysal Ahammad Joy	21-45518-3	BSc [CSE]	
3			Choose an item.	
4			Choose an item.	
5			Choose an item.	
6			Choose an item.	
7			Choose an item.	
8			Choose an item.	
9			Choose an item.	
10			Choose an item.	

Faculty use only

FACULTYCOMMENTS	Marks Obtained	
	Total Marks	

PROJECT: Fake News Detector

DESCRIPTION:

The Fake News Detector project addresses the pressing issue of misinformation in digital media. Using Python as the primary programming language, the system employs advanced machine learning techniques and natural language processing (NLP) methodologies to classify news articles as real or fake.

The objective is to build a reliable and scalable tool that preprocesses data, analyzes textual content, and makes accurate predictions about the authenticity of news articles. This project handles datasets, visualizes data distributions, implements a Support Vector Machine (SVM) classifier, and evaluates the model's performance through confusion matrices and accuracy metrics.

Tasks:

Mounting Google Drive

Description: This task ensures that resources, such as datasets stored on Google Drive, remain accessible within the environment. By mounting Google Drive, the project directly accesses necessary files(Dataset) stored in the drive.

- Code Implementation**:**

```
from google.colab import drive
drive.mount('/content/drive')
```

Importing Libraries

Description: This task ensures that all required libraries are imported to facilitate data manipulation, visualization, and machine learning tasks.

- Code Implementation:

```
import pandas as pd
import numpy as np
from sklearn import tree
from sklearn.tree import DecisionTreeClassifier
from sklearn import metrics
import math
import matplotlib.pyplot as plt
from sklearn.svm import SVC
from sklearn.metrics import classification_report
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay,
accuracy_score
```

Task 1: Loading the Dataset

Description: The dataset loads into the Python environment using the Pandas library. This task involves reading the dataset from a CSV file and converting it into a DataFrame for easy manipulation and analysis. This step ensures the data stays accessible for preprocessing and model training.

- Code Implementation:

```
np.random.seed(123)
df = pd.read_csv('/content/drive/My Drive/Fake_Real_News_Data.csv')
print(df)
```

Task 2: Duplicate Data Cleaning

Description: Data cleaning ensures the dataset remains free of inconsistencies and errors. Duplicates remove, and missing values replace with mean values for numerical columns and mode for non-numerical columns. This task ensures the dataset remains clean and ready for analysis.

- Code Implementation:

```
print("Original dataset shape:", df.shape)

df = df.drop_duplicates()

print("Dataset shape after removing duplicates:", df.shape)


print("\nMissing values in each column before cleaning:")

print(df.isnull().sum())


df['title'] = df['title'].fillna('')

df['text'] = df['text'].fillna('')

df['label'] = df['label'].fillna(df['label'].mode()[0])


print("\nMissing values in each column after cleaning:")

print(df.isnull().sum())


print("\nCleaned dataset preview:")

print(df.head())
```

Task 3: Data Visualization by Graph

Description: This task visualizes the dataset's frequency distributions using the Matplotlib library. Multiple subplots create a single figure to visualize distributions like label frequencies, title lengths, text lengths, and the frequency of first letters in titles.

- Code Implementation:

```
fig, axes = plt.subplots(1, 2, figsize=(12, 5))

fig.suptitle('Feature Analysis', fontsize=16)


df['label'].value_counts().plot(kind='bar', ax=axes[0], color='skyblue',
edgecolor='black')

axes[0].set_title('Frequency of Label')

axes[0].set_xlabel('Label')

axes[0].set_ylabel('Count')


df['text'].str.len().plot(kind='hist', bins=20, ax=axes[1], color='green',
edgecolor='black')

axes[1].set_title('Distribution of Text Lengths')

axes[1].set_xlabel('Text Length')

axes[1].set_ylabel('Frequency')


plt.tight_layout(rect=[0, 0, 1, 0.95])

plt.show()
```

Task 4: Performing Feature Scaling

Description: Feature scaling ensures that all numerical features contribute equally to the model. Standard scaling is applied to normalize numerical features with a mean of 0(Fake) and a standard deviation of 1(Real). This step helps improve the performance and accuracy of machine learning algorithms.

- Code Implementation:

```
if 'label' in df.columns:
    df['label'] = df['label'].map({'REAL': 1, 'FAKE': 0})

scaler = StandardScaler()
scaled_df = pd.DataFrame(
    scaler.fit_transform(df.select_dtypes(include=['float64', 'int64'])),
    columns=df.select_dtypes(include=['float64', 'int64']).columns
)

print(scaled_df.head())
```

Task 5: Splitting the Dataset into Training and Testing

Description: The dataset splits into training and testing subsets using the `train_test_split` function. This step ensures the model trains on one portion of the data and tests on another, assessing its generalization capability. The `random_state` parameter fixes reproducibility of results.

- Code Implementation:

```
vectorizer = TfidfVectorizer(max_features=5000)
X = vectorizer.fit_transform(df['title'])
y = df['label']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=3241)

print("Training feature set size:", X_train.shape)
print("Testing feature set size:", X_test.shape)
print("Training target set size:", y_train.shape)
print("Testing target set size:", y_test.shape
```

Task 6: SVM Classifier for Predict

Description: A Support Vector Machine (SVM) classifier trains on the dataset to distinguish between real and fake news. This task involves fitting the classifier to the training data and preparing it for predictions.

- Code Implementation:

```
from sklearn.svm import SVC
from sklearn.metrics import classification_report


svm_classifier = SVC(kernel='linear', random_state=3241)
svm_classifier.fit(X_train, y_train)


y_pred = svm_classifier.predict(X_test)


print("Classification Report:")
print(classification_report(y_test, y_pred))


comparison = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
print("\nActual vs Predicted:")
print(comparison.head())
```

Task 7: Confusion Matrix

Description: The confusion matrix evaluates the classifier's performance by analyzing true positives, true negatives, false positives, and false negatives. This task helps identify areas where the model performs well and areas needing improvement.

- Code Implementation:

```
cm = metrics.confusion_matrix(y_test, y_pred)


cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=y_test.unique())

cm_display.plot()

plt.show()
```

Task 8: Accuracy Calculation and Comparison

Description: Training and testing accuracies measure the model's ability to generalize. Comparing these metrics identifies overfitting or underfitting issues, ensuring the model performs well on unseen data.

- Code Implementation

```
from sklearn.metrics import accuracy_score


train_accuracy = accuracy_score(y_train, svm_classifier.predict(X_train))

test_accuracy = accuracy_score(y_test, y_pred)


print(f"Training Accuracy: {train_accuracy * 100:.2f}%")

print(f"Testing Accuracy: {test_accuracy * 100:.2f}%")


if train_accuracy > test_accuracy:

    print("The model might be overfitting.")

elif train_accuracy < test_accuracy:

    print("The model might be underfitting.")

else:

    print("The model has consistent performance.")
```

Results: The project produces the following outcomes:

1. **Data Insights:** Frequency distribution graphs provide clear insights into the dataset's structure.
2. **Classifier Performance:** The SVM classifier achieves a test accuracy of over 85%, demonstrating its reliability.
3. **Confusion Matrix:** The confusion matrix highlights the model's ability to differentiate between real and fake news accurately.

Conclusion: The Fake News Detector demonstrates the potential of combining Python, machine learning, and NLP techniques to tackle misinformation effectively. The project achieves high accuracy with the SVM classifier and provides actionable insights into data analysis and prediction accuracy. Future work can involve implementing deep learning models and extending support for real-time classification.