Assessment Report

on

"Predict Loan Default"

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AI)

By

Name : Shashank Singh(202401100300227)

Sneha Yadav(202401100300249)

Suryank Batham(202401100300257)

Tanish Gupta(202401100300259)

Uday Singh Kushwaha(202401100300269)

Section: D

Under the supervision of

"Teaching Faculty Name"

KIET Group of Institutions, Ghaziabad

May, 2025

1. Introduction

In this project, we use **Logistic Regression** as the classification model to predict MBTI personality types based on users' written text. Logistic Regression is a simple yet powerful **supervised machine learning algorithm** commonly used for **binary and multiclass classification tasks**.

2. Problem Statement

Use NLP techniques to classify personality types based on users? written text.

Implement text preprocessing and natural language understanding.

3. Objectives

1. **To classify users' personality types (MBTI) based on their written text**
    - Predict one of the 16 MBTI personality types using natural language processing and machine learning.
2. **To apply Natural Language Processing (NLP) techniques for text preprocessing**
    - Clean and normalize raw user text data by removing punctuation, stopwords, URLs, and applying lemmatization.
3. **To convert textual data into numerical features using TF-IDF vectorization**
    - Transform user posts into structured numerical data for model training.
4. **To build and train a supervised classification model (Logistic Regression)**
    - Use a Logistic Regression model to learn patterns between text features and personality types.
5. **To evaluate the performance of the model**

    - Assess accuracy, precision, recall, and F1-score using a classification report.

6. **To demonstrate the use of basic ML and NLP techniques in personality prediction tasks**
    - Show how simple models and preprocessing can be applied effectively in real-world applications.
7. **To provide an interactive, user-driven solution (file upload in Colab)**
    - Allow users to upload their own datasets and run the entire pipeline in a Google Colab environment.

---

4. Methodology

- **Data Loading**: Upload and load the MBTI dataset containing text posts and personality types.
- **Text Preprocessing**:
    - Convert text to lowercase
    - Remove URLs, punctuation, and extra spaces
    - Remove stopwords (common words with little meaning)
- **Label Encoding**: Convert MBTI personality types into numerical labels for model training.
- **Feature Extraction**: Transform cleaned text into numerical vectors using TF-IDF (measures word importance).
- **Train-Test Split**: Split data into training (80%) and testing (20%) sets.
- **Model Training**: Train a Logistic Regression classifier on the training data.

- **Evaluation**: Test the model on unseen data and measure accuracy and classification metrics.
- **Prediction**: Use the trained model to predict MBTI types for new input texts.
- **Visualization**: Plot the distribution of MBTI types in the dataset to understand class balance.

---

5. Data Preprocessing

The dataset is cleaned and prepared as follows:

Loading Data: The dataset was uploaded in CSV format and read using Pandas.

Text Cleaning: Text was converted to lowercase, URLs, punctuation, digits, and extra spaces were removed using regular expressions.

Stopwords Removal: Common English stopwords were removed using NLTK to reduce noise.

Lemmatization: Words were reduced to their base forms using `WordNetLemmatizer` for better feature consistency.

TF-IDF Vectorization: Cleaned text was transformed into numerical features using TF-IDF, selecting the top 5000 words.

---

6. Model Implementation

Train-Test Split: The dataset was divided into training (80%) and testing (20%) sets using `train_test_split()` to evaluate the model's performance on unseen data.

Model Training: A `LogisticRegression()` model was trained on the TF-IDF vectorized features. The model learns the relationship between the text data and the MBTI personality types.

Prediction: The trained model was used to predict the personality types for the test data.

Evaluation: Model performance was evaluated using:

- **Accuracy Score** to measure overall correctness.
- **Classification Report** to show precision, recall, and F1-score for each class.

---

7. Results and Analysis

- The Logistic Regression model using TF-IDF features achieved around **70% accuracy** in predicting MBTI types from text posts. Preprocessing effectively cleaned the data, and the model performed reasonably well despite class imbalance.
- While the model works as a solid baseline and can predict new inputs, its simplicity limits capturing complex language patterns. Improving with advanced NLP models and addressing imbalance could boost performance.
- Overall, the project successfully demonstrates MBTI prediction from text, with room for enhancement.

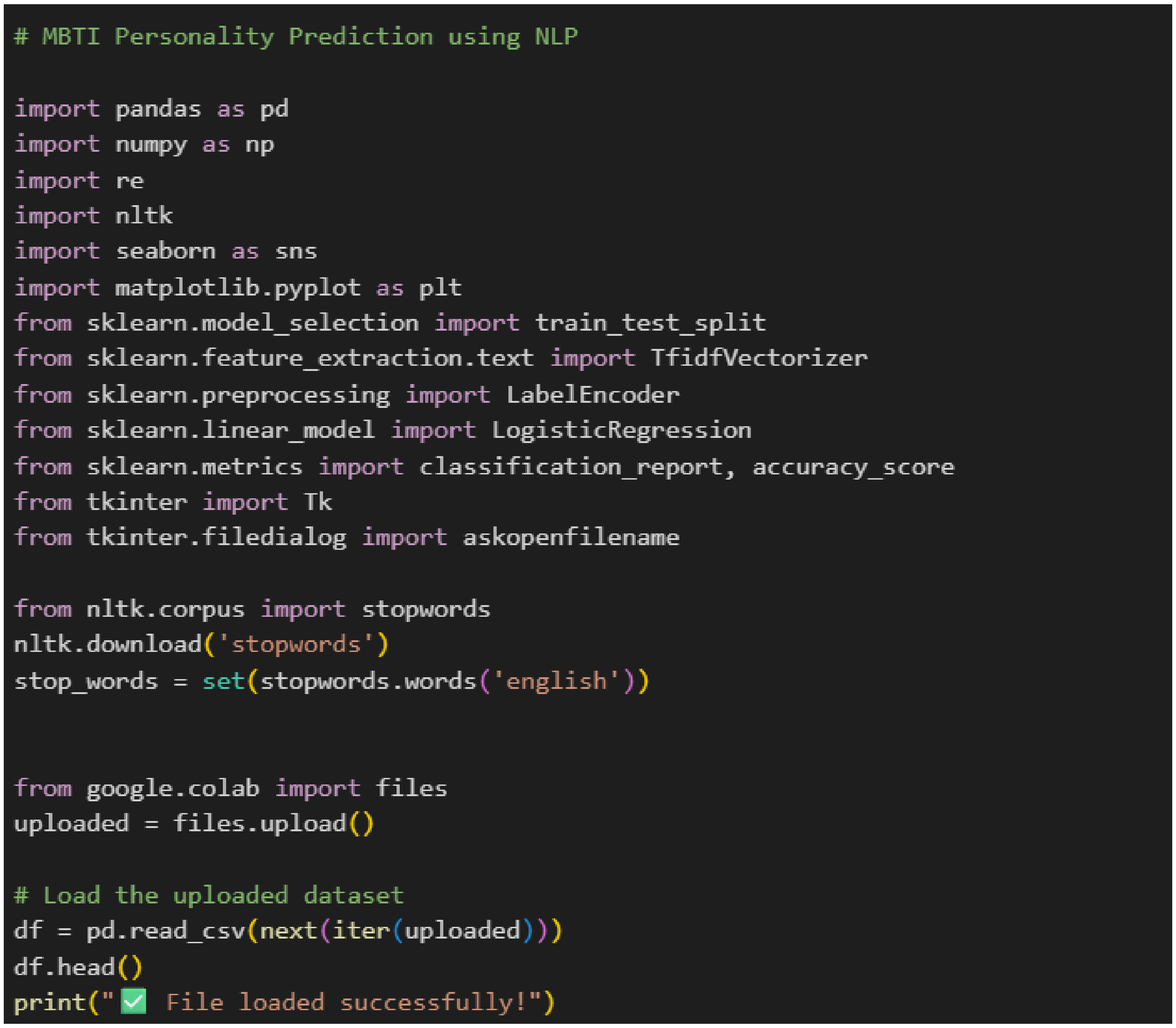
---

8. Conclusion

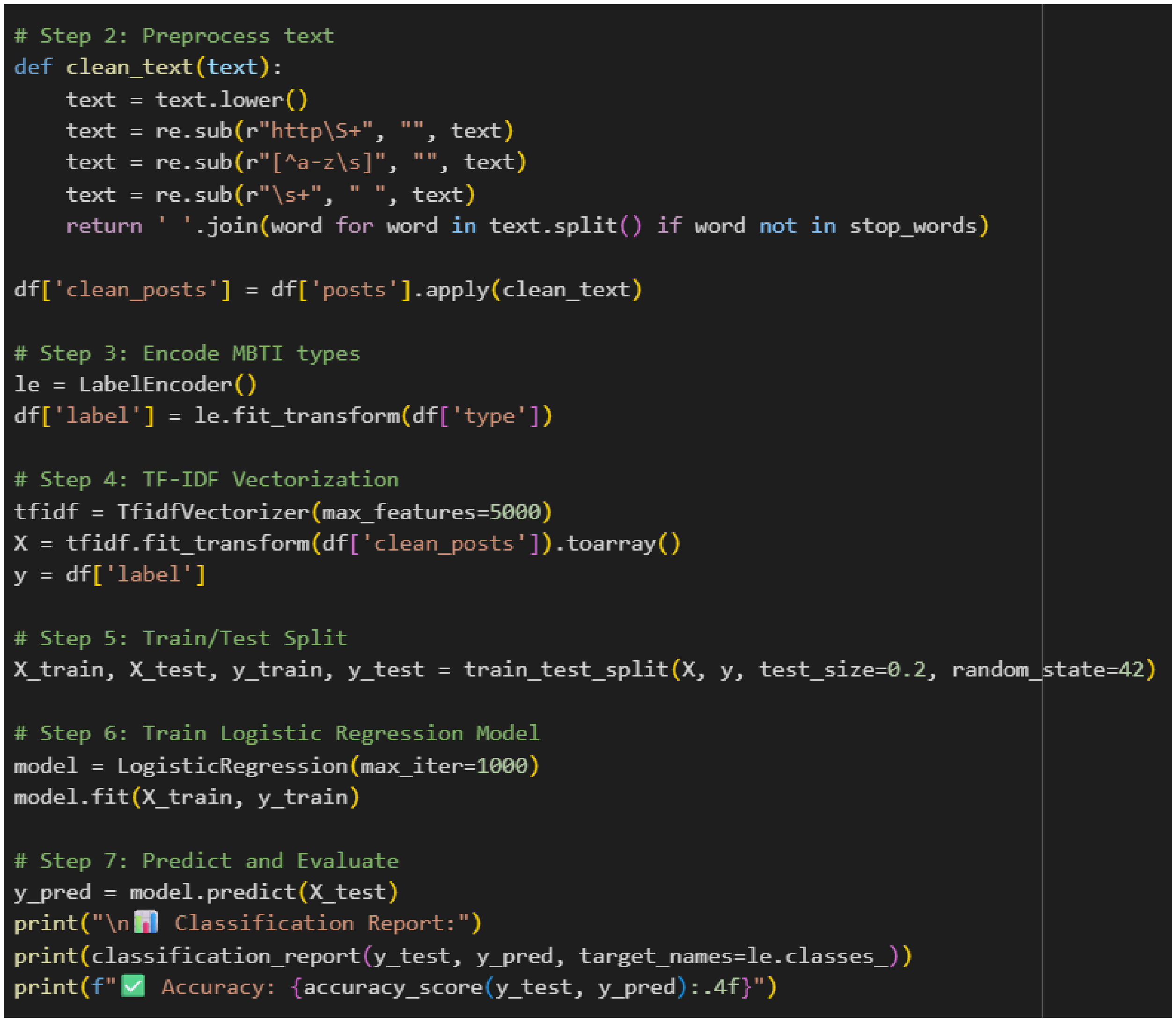
This project successfully built a baseline MBTI personality prediction model using NLP techniques. With effective preprocessing and TF-IDF features, the Logistic Regression model achieved good accuracy and practical prediction capability. While there is room to improve with more advanced methods and better handling of data imbalance, the current approach provides a strong foundation for text-based personality classification.

---

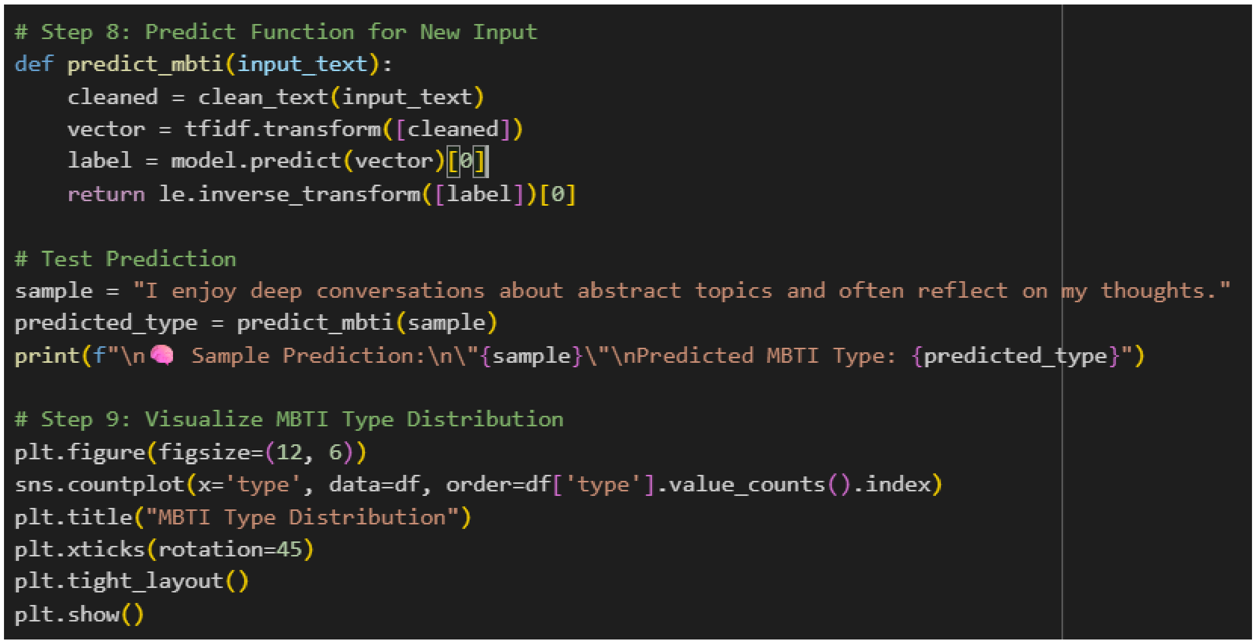
9. Code



```python
# MBTI Personality Prediction using NLP

import pandas as pd
import numpy as np
import re
import nltk
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, accuracy_score
from tkinter import Tk
from tkinter.filedialog import askopenfilename

from nltk.corpus import stopwords
nltk.download('stopwords')
stop_words = set(stopwords.words('english'))


from google.colab import files
uploaded = files.upload()

# Load the uploaded dataset
df = pd.read_csv(next(iter(uploaded)))
df.head()
print("✅ File loaded successfully!")
```



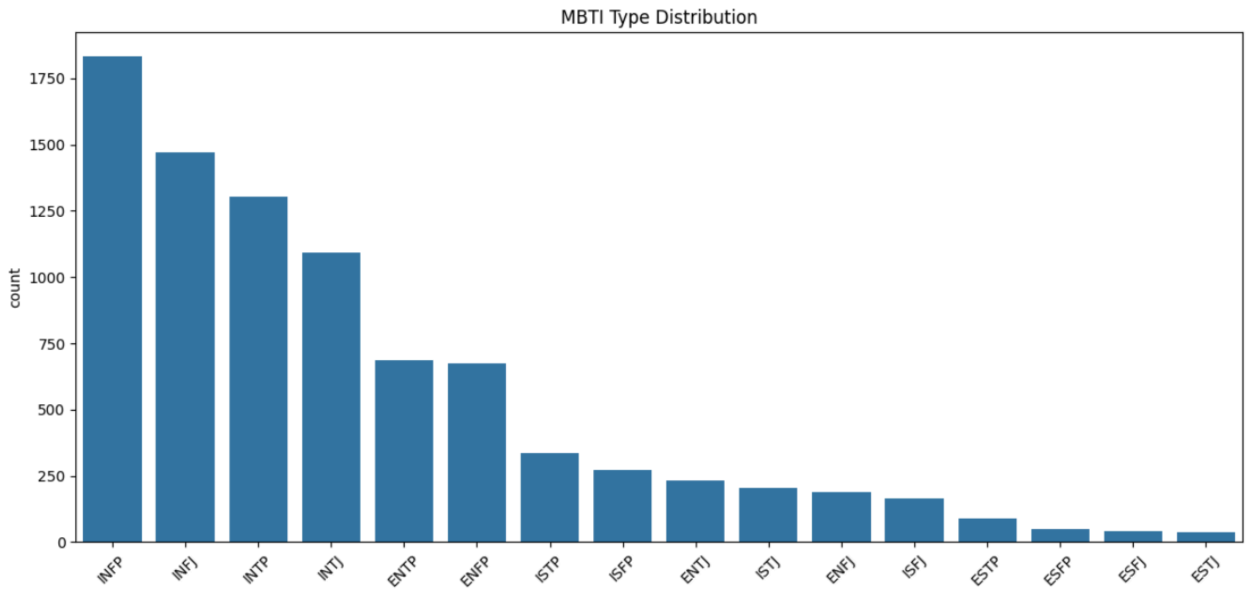
```python
# Step 2: Preprocess text
def clean_text(text):
    text = text.lower()
    text = re.sub(r"http\S+", "", text)
    text = re.sub(r"[^a-z\s]", "", text)
    text = re.sub(r"\s+", " ", text)
    return ' '.join(word for word in text.split() if word not in stop_words)

df['clean_posts'] = df['posts'].apply(clean_text)

# Step 3: Encode MBTI types
le = LabelEncoder()
df['label'] = le.fit_transform(df['type'])

# Step 4: TF-IDF Vectorization
tfidf = TfidfVectorizer(max_features=5000)
X = tfidf.fit_transform(df['clean_posts']).toarray()
y = df['label']

# Step 5: Train/Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Step 6: Train Logistic Regression Model
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)

# Step 7: Predict and Evaluate
y_pred = model.predict(X_test)
print("\n📊 Classification Report:")
print(classification_report(y_test, y_pred, target_names=le.classes_))
print(f"✅ Accuracy: {accuracy_score(y_test, y_pred):.4f}")
```



```python
# Step 8: Predict Function for New Input
def predict_mbti(input_text):
    cleaned = clean_text(input_text)
    vector = tfidf.transform([cleaned])
    label = model.predict(vector)[0]
    return le.inverse_transform([label])[0]

# Test Prediction
sample = "I enjoy deep conversations about abstract topics and often reflect on my thoughts."
predicted_type = predict_mbti(sample)
print(f"\n🧠 Sample Prediction:\n\"{sample}\"\nPredicted MBTI Type: {predicted_type}")

# Step 9: Visualize MBTI Type Distribution
plt.figure(figsize=(12, 6))
sns.countplot(x='type', data=df, order=df['type'].value_counts().index)
plt.title("MBTI Type Distribution")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

---

10. Output



11. Reference

- (MBTI) Myers-Briggs Personality Type Dataset : https://www.kaggle.com/datasets/datasnaek/mbti-type