NPL&NAIVE BAYES

Submission Structure

1. Comprehensive Report

Title: Text Classification and Sentiment Analysis of Blog Posts

Introduction:

- Briefly explain the objective of the project: to classify blog posts using a Naive Bayes model and analyze their sentiments.
- Introduce the dataset and its significance.

Dataset Overview:

- Describe the structure of the /content/blogs.csv dataset.
- Highlight the key columns (`Data` and `Labels`).

Data Exploration:

- Include insights from exploratory data analysis (EDA).
- Visualizations of category distributions and any notable observations.

Data Preprocessing:

- Explain the preprocessing steps taken (e.g., cleaning, tokenization, stopword removal).
- Discuss the importance of these steps in the context of NLP.

Feature Extraction:

- Describe the TF-IDF vectorization process.
- Justify the choice of using TF-IDF for feature extraction.

Naive Bayes Classification:

- Detail the model training process, including the train-test split and training the Naive Bayes classifier.
- Present performance metrics (accuracy, precision, recall, F1-score).
- Discuss the results and any challenges encountered.

Sentiment Analysis:

- Explain the choice of the VADER sentiment analysis tool.
- Present the sentiment analysis results, including distribution of sentiments and insights across categories.
- Discuss the implications of sentiment results on content strategy.

Conclusion:

- Summarize key findings and potential future work, such as exploring other models or deeper sentiment analysis techniques.

References:

- Cite any literature or resources you referred to during the project.

2. Codebase Organization

Directory Structure:

```
text_classification_project/
│
├── data/
│   └── blogs_categories.csv
├── notebooks/
│   └── text_classification_analysis.ipynb
├── src/
│   ├── preprocess.py
│   ├── train_model.py
│   ├── sentiment_analysis.py
│   └── evaluate_model.py
└── report/
    └── text_classification_report.md
```

Code Documentation:

- Each script in the `src` folder should include docstrings at the beginning of the file and comments throughout the code to explain the logic.

Example Code Snippet (train_model.py):

```
import pandas as pd
```

```
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.feature_extraction.text import
TfidfVectorizer
from sklearn.metrics import accuracy_score,
classification_report

def load_data(filepath):
    """Load the dataset from a CSV file."""
    return pd.read_csv(filepath)

def preprocess_data(data):
    """Preprocess the text data for model training."""
    # Add your preprocessing steps here
    pass

def train_naive_bayes(X_train, y_train):
    """Train a Naive Bayes classifier."""
    model = MultinomialNB()
    model.fit(X_train, y_train)
    return model

def evaluate_model(model, X_test, y_test):
    """Evaluate the model using accuracy and
classification report."""
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    report = classification_report(y_test, y_pred)
    print(f"Accuracy: {accuracy}\nClassification
Report:\n{report}")

if __name__ == "__main__":
    # Main execution flow
    data = load_data('data/blogs_categories.csv')
    # Preprocessing and feature extraction would be
called here
```

Final Submission Checklist

1. **Report:**
    - Ensure clarity and thoroughness.

- Include visualizations where applicable.
- Proofread for grammar and formatting consistency.

2. **Code:**
   - Ensure that all code is organized and well-documented.
   - Verify that each module runs correctly and produces expected results.
   - Test for edge cases in preprocessing and model evaluation.
3. **Readme File (optional but recommended):**
   - Provide an overview of the project, instructions for running the code, and any dependencies needed.