Project 3

Code-

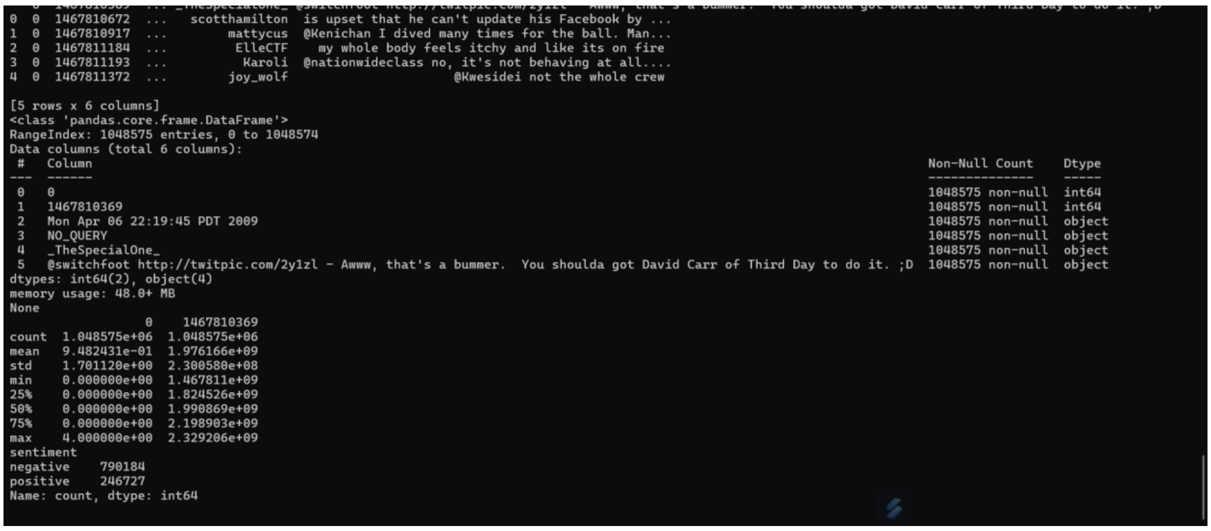
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, f1_score
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer
import re

# Download required NLTK data
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')

# Load dataset
file_path = "C:\\Users\\PIYUSH\\Desktop\\doc.csv"
df = pd.read_csv(file_path, encoding='latin-1')

# Data Exploration
print(df.head())
print(df.info())
print(df.describe())
```

```
# Data Cleaning

df.columns = ['sentiment', 'id', 'date', 'query', 'user', 'text']

df = df[['sentiment', 'text']]

df.drop_duplicates(inplace=True)

df.dropna(inplace=True)

df['sentiment'] = df['sentiment'].apply(lambda x: 'positive' if x == 4 else ('negative' if x == 0 else
'neutral'))


# Exploratory Data Analysis (EDA)

print(df['sentiment'].value_counts())

sns.countplot(x='sentiment', data=df)

plt.title('Sentiment Distribution')

plt.show()


# Word Frequency Analysis

def plot_wordcloud(text):

    wordcloud = WordCloud(width=800, height=400, background_color='white').generate(text)

    plt.figure(figsize=(10, 5))

    plt.imshow(wordcloud, interpolation='bilinear')

    plt.axis('off')

    plt.show()


all_text = ' '.join(df['text'])

plot_wordcloud(all_text)


# Temporal Analysis (Not included due to lack of date parsing)


# Text Preprocessing

def preprocess_text(text):

    text = re.sub(r'http\S+|www\S+|https\S+', '', text, flags=re.MULTILINE)

    text = re.sub(r'\@\w+|\#','', text)
    text = text.lower()
    text = word_tokenize(text)
    text = [WordNetLemmatizer().lemmatize(word) for word in text if word not in
stopwords.words('english') and word.isalnum()]
    return ' '.join(text)

df['clean_text'] = df['text'].apply(preprocess_text)

# Sentiment Prediction Model
X_train, X_test, y_train, y_test = train_test_split(df['clean_text'], df['sentiment'], test_size=0.2,
random_state=42)
vectorizer = CountVectorizer()
X_train_vec = vectorizer.fit_transform(X_train)
X_test_vec = vectorizer.transform(X_test)

model = MultinomialNB()
model.fit(X_train_vec, y_train)
y_pred = model.predict(X_test_vec)

print(f'Accuracy: {accuracy_score(y_test, y_pred)}')
print(f'F1 Score: {f1_score(y_test, y_pred, average="weighted")}')

# Feature Importance
def plot_feature_importance(model, vectorizer):
    feature_names = vectorizer.get_feature_names_out()
    coef = model.feature_log_prob_[1] - model.feature_log_prob_[0]
    importance = np.exp(coef)
    top_features = np.argsort(importance)[::-1][:20]

    plt.figure(figsize=(10, 8))
    plt.barh(range(len(top_features)), importance[top_features], align='center')
    plt.yticks(range(len(top_features)), np.array(feature_names)[top_features])
    plt.xlabel('Feature Importance')
    plt.title('Top 20 Features for Positive Sentiment')
    plt.show()


plot_feature_importance(model, vectorizer)
```

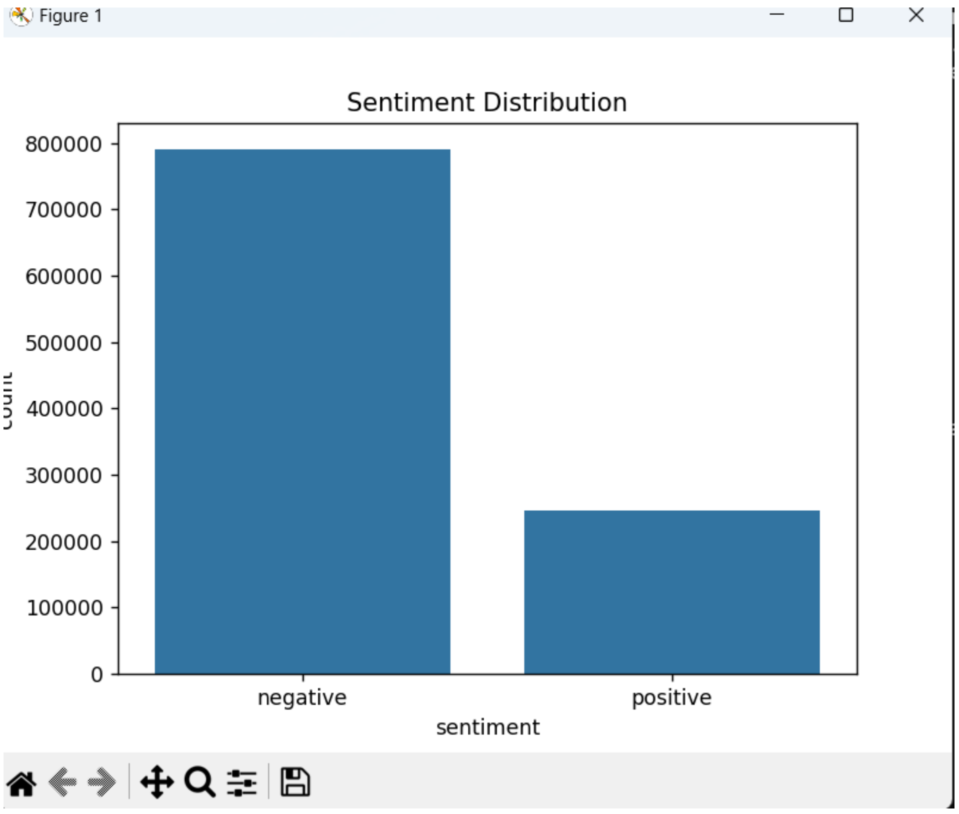
SUMMARY

- **Data Loading: Loading the dataset using pandas.**
- **Data Exploration: Exploring the dataset with basic descriptive statistics.**
- **Data Cleaning: Renaming columns, removing duplicates, dropping missing values, and mapping sentiment values.**
- **Text Preprocessing: A function to clean, tokenize, remove stopwords, and lemmatize the text.**
- **EDA: Displaying the distribution of sentiments using seaborn.**
- **Word Cloud Generation: Visualizing the most frequent words using WordCloud.**
- **Feature Engineering: Converting text into numeric features using CountVectorizer.**
- **Model Training: Splitting data, training a Naive Bayes classifier, and evaluating its performance using accuracy and F1 score.**

output and analysis



```
0  0  1467810672  ...   scotthamilton  is upset that he can't update his Facebook by ...
1  0  1467810917  ...        mattycus  @Kenichan I dived many times for the ball. Man...
2  0  1467811184  ...         ElleCTF    my whole body feels itchy and like its on fire
3  0  1467811193  ...          Karoli  @nationwideclass no, it's not behaving at all....
4  0  1467811372  ...        joy_wolf                      @Kwesidei not the whole crew

[5 rows x 6 columns]
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1048575 entries, 0 to 1048574
Data columns (total 6 columns):
 #   Column                                                                                                              Non-Null Count    Dtype
---  ------                                                                                                              --------------    -----
 0   0                                                                                                                   1048575 non-null  int64
 1   1467810369                                                                                                          1048575 non-null  int64
 2   Mon Apr 06 22:19:45 PDT 2009                                                                                        1048575 non-null  object
 3   NO_QUERY                                                                                                            1048575 non-null  object
 4   _TheSpecialOne_                                                                                                     1048575 non-null  object
 5   @switchfoot http://twitpic.com/2y1zl - Awww, that's a bummer.  You shoulda got David Carr of Third Day to do it. ;D  1048575 non-null  object
dtypes: int64(2), object(4)
memory usage: 48.0+ MB
None
                  0    1467810369
count  1.048575e+06  1.048575e+06
mean   9.482431e-01  1.976166e+09
std    1.701120e+00  2.300580e+08
min    0.000000e+00  1.467811e+09
25%    0.000000e+00  1.824526e+09
50%    0.000000e+00  1.990869e+09
75%    0.000000e+00  2.198903e+09
max    4.000000e+00  2.329206e+09
sentiment
negative    790184
positive    246727
Name: count, dtype: int64
```



The bar chart shows the sentiment distribution for a dataset of tweets, indicating the number of tweets classified as either "negative" or "positive."

Key Observations:

- **Imbalance in Sentiment Distribution: The chart reveals a significant imbalance in sentiment, with a much larger number of negative tweets compared to positive ones. This indicates that the dataset may be biased toward negative sentiments.**

Analytical Insights:

1. **Class Imbalance: Since the negative class is dominant, any sentiment prediction model developed using this dataset might be biased toward predicting negative sentiment. It's essential to address this imbalance either by resampling techniques (oversampling the minority class or under sampling the majority class) or by using techniques like SMOTE.**

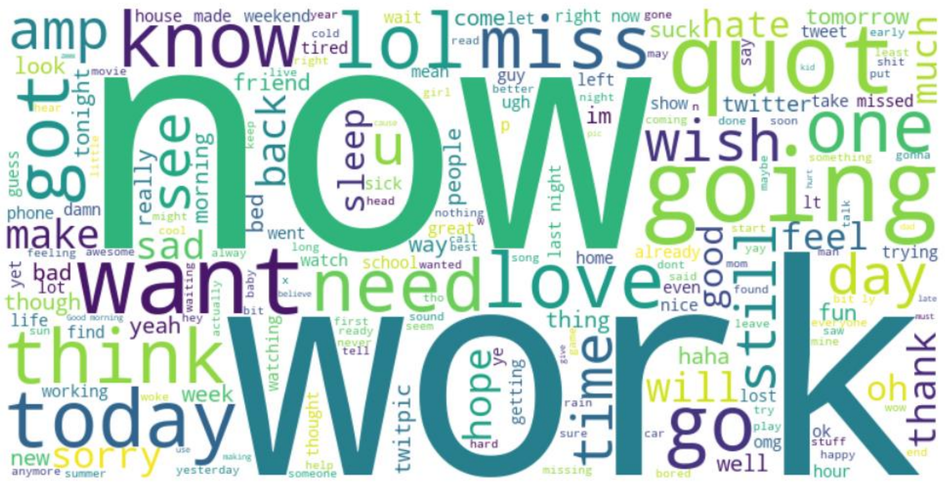
2. **Potential Impact on Model Performance: With such an imbalance, the model could achieve high accuracy by predominantly predicting the majority class (negative sentiment). However, this would lead to poor performance in predicting positive sentiments, as seen through metrics like F1 score, precision, and recall for the minority class.**

3. **Exploratory Data Analysis Next Steps:**

    - **Perform further analysis to explore the distribution of words or phrases in each sentiment category.**

- Investigate temporal trends to see if there are specific periods where one sentiment dominates more than the other.

This sentiment imbalance should be carefully considered when developing any machine learning model or drawing insights from this dataset.



The word cloud provides a visual representation of the most frequently occurring words in the Twitter dataset. Larger words indicate higher frequency, while smaller words represent less common terms.

Key Insights:

1. Common Words: The most prominent words include "work," "now," "want," "need," "think," and "today." These words suggest topics related to daily activities, desires, emotions, and time-related contexts.
2. Sentiment Implications:
   - Positive Words: Words like "love," "lol," "thank," and "fun" suggest positive sentiment.
   - Negative Words: Words like "hate," "sad," and "sorry" imply negative emotions.
   - Neutral Words: Terms like "work," "time," "day," and "today" could be context-dependent, conveying neutral, positive, or negative sentiments based on usage.
3. Common Expressions: Words such as "going," "miss," "wish," and "hope" reflect personal reflections and aspirations, which are typical in tweets expressing daily thoughts or feelings.
4. Trivial Terms: Words like "quot" and "amp" might be artifacts from processing, likely representing HTML character codes or formatting issues that should be filtered out during data cleaning.