# **Sentiment Analysis**

# **Introduction**

This document provides an overview of the sentiment analysis code implemented using Python and popular machine learning libraries such as Pandas, NumPy, Scikit-learn, NLTK, Matplotlib, Seaborn, WordCloud, and Lime. The sentiment analysis is performed on a Twitter dataset using various machine learning models, including Multinomial Naive Bayes, Linear Support Vector Classifier (LinearSVC), and Logistic Regression. Additionally, the code includes data preprocessing, downsampling, text vectorization, model selection, cross-validation, model interpretability using LIME (Local Interpretable Model-agnostic Explanations), and evaluation metrics.

## **Code Overview**

### **1. Data Loading and Preprocessing**

* The code begins with importing necessary libraries and loading the Twitter dataset into a Pandas DataFrame (df).
* The dataset is shuffled for better training and testing.

df= df.sample(frac =1)

* Downsampling is performed to balance the number of positive and negative sentiment tweets.

df\_majority\_downsampled = resample(df\_majority, replace=False,n\_samples=len(df\_minority),random\_state=1234)

df = df\_majority\_downsampled.append(df\_minority)

### **2. Text Preprocessing**

* Text data is preprocessed by removing non-alphabetic characters, converting text to lowercase, removing special characters, and lemmatizing words.
* NLTK's stopwords and WordNetLemmatizer are used for text processing.

*corpus\_f = []*

*corpus\_joined = []*

*for i in df.index:*

*text = re.sub('[^a-zA-Z]', ' ', df['text'][i])*

*text = text.lower()*

*text=re.sub("&lt;/?.\*?&gt;"," &lt;&gt; ",text)*

*text=re.sub("(\\d|\\W)+"," ",text)*

*text = text.split()*

*lem = WordNetLemmatizer()*

*text = [lem.lemmatize(word) for word in text*

*if not word in remove]*

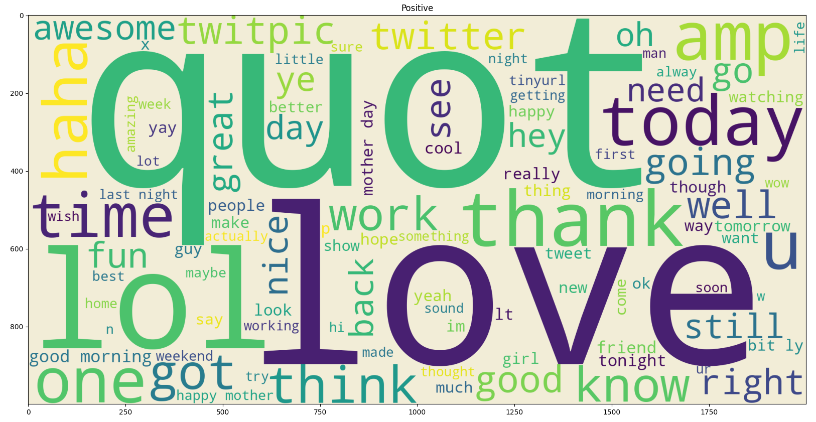
*text1 = " ".join(text)*

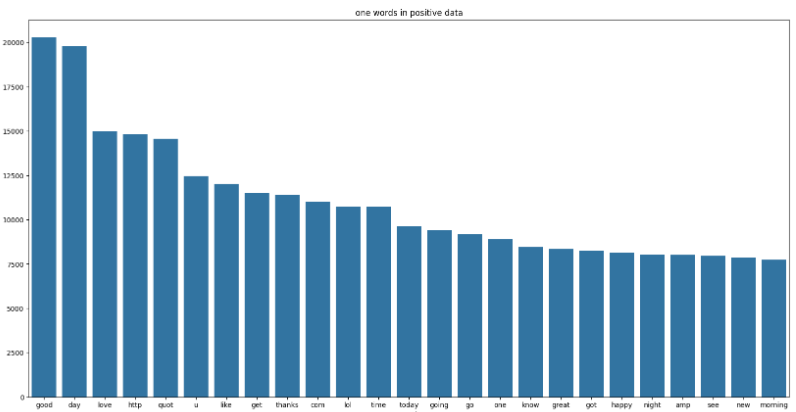
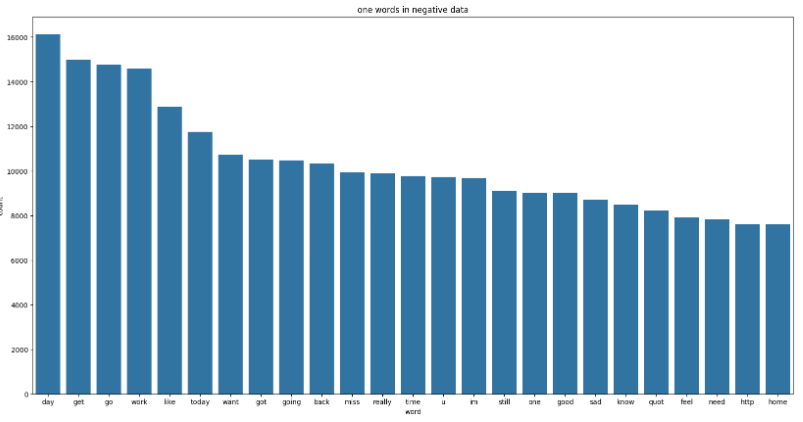
*corpus\_f.append(text)*

*corpus\_joined.append(text1)*

### **3. Exploratory Data Analysis (EDA)**

* Word clouds and bar plots are created to visualize the most frequent words in positive and negative sentiment tweets.

### **4. Text Vectorization**

* TfidfVectorizer is employed to convert text data into numerical vectors for machine learning model input.

from sklearn.feature\_extraction.text import TfidfVectorizer

tfidf = TfidfVectorizer()

vector = tfidf.fit\_transform(data['text'])

y = data['sentiment']

### **5. Model Selection**

* Three different machine learning models are implemented:
* Multinomial Naive Bayes
* Linear Support Vector Classifier (LinearSVC)
* Logistic Regression
* The models are trained, tested, and evaluated on accuracy and classification reports.

### **6. Cross-Validation**

* Cross-validation is performed using StratifiedKFold to assess the models' performance on different folds of the dataset.

cv\_strategy = StratifiedKFold(n\_splits=5, shuffle=True, random\_state=42)

### **7. Model Interpretability with LIME**

* Logistic Regression model coefficients are interpreted using LIME to explain the predictions.

from sklearn.linear\_model import LogisticRegression

from sklearn.calibration import CalibratedClassifierCV

lr = LogisticRegression(max\_iter=1000)

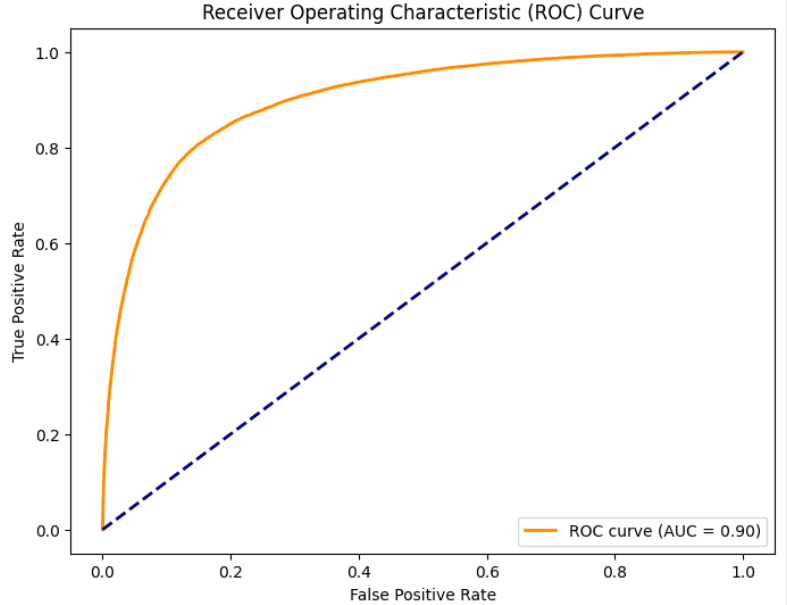
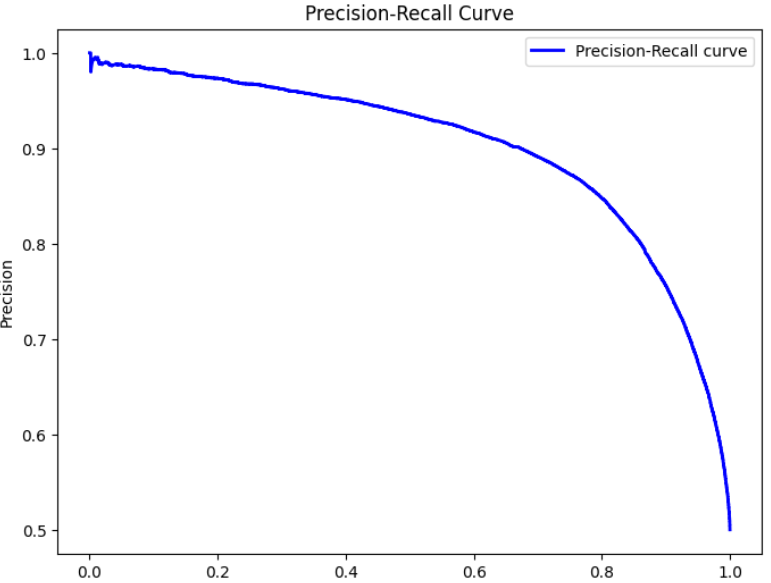
lr.fit(vector, y)

calibrated\_lr = CalibratedClassifierCV(lr, method='sigmoid', cv='prefit')

calibrated\_lr.fit(vector, y)

### **8. Evaluation Metrics**

* Confusion matrix, ROC curve, AUC-ROC, precision-recall curve, and classification report are used to evaluate the models' performance.

## **Results**

### **Naive Bayes Model**

* Accuracy: 76.68%
* Precision, recall, and f1-score for positive and negative sentiments.

### **Linear Support Vector Classifier (LinearSVC) Model**

* Accuracy: 77.55%
* Precision, recall, and f1-score for positive and negative sentiments.

### **Logistic Regression Model**

* Accuracy: 78.23%
* Precision, recall, and f1-score for positive and negative sentiments.

### **Cross-Validation Results**

* Cross-validation shows consistent performance across different folds, with accuracy ranging from 77.61% to 78.19%.

### **Model Interpretability**

* The top 10 important features (words) contributing to the sentiment predictions are displayed.

### **Evaluation Metrics**

* Confusion matrix, ROC curve, AUC-ROC, and precision-recall curve provide a comprehensive understanding of the model's performance.

## **Conclusion**

The sentiment analysis code successfully analyzes Twitter data, preprocesses text, trains machine learning models, and evaluates their performance. The use of cross-validation ensures robust model assessment, and model interpretability with LIME aids in understanding the importance of features. The evaluation metrics provide insights into the strengths and weaknesses of each model. The Logistic Regression model achieved the highest accuracy of 78.23%. The comprehensive analysis presented in this document serves as a valuable resource for sentiment analysis tasks on Twitter data.