**Department of Electrical and Computer Engineering**

**North South University**

Directed Research

# Celebrity Sentiment Analysis using Natural Learning Processing


| | |
|---|---|
| Shuverthi Kanungo | ID# 1911630042 |
| Fahid Shadman Karim | ID# 1911110042 |
| Trina Pandit | ID# 1912517642 |
| Sumit Saha | ID# 1931415042 |

Faculty Advisor

Rifat Ahmed Hassan

Lecturer

ECE Department


Fall 2023

# DECLARATION

This is to certify that this project is our original work. No part of this work has been submitted elsewhere partially or fully for the award of any other degree or diploma. Any material reproduced in this project has been properly acknowledged.

Students' name & Signature

1. **Shuverthi Kanungo**

______________________________

2. **Fahid Shadman Karim**

______________________________

3. **Trina Pandit**

______________________________

4. **Sumit Saha**

______________________________

# APPROVAL

The research project entitled **"Celebrity Sentiment Analysis using Natural Learning Processing"** by **Shuverthi Kanungo (ID# 1911630042), Fahid Shadman Karim (ID# 1911110042), Trina Pandit (ID# 1912517642) and Sumit Saha (ID# 1931415042)** is approved in partial fulfillment of the requirement of the Degree of Bachelor of Science in Computer Science and Engineering in August and has been accepted as satisfactory.

**Supervisor's Signature**

---


**Rifat Ahmed Hassan**

**Lecturer**

Department of Electrical and Computer Engineering

North South University


**Department Chair's Signature**

---


**Dr. Rajesh Palit**

**Associate Professor**

Department of Electrical and Computer Engineering

North South University

# ACKNOWLEDGMENT

First, we wish to express our gratitude to the Almighty for giving us the strength to perform our responsibilities and complete the report.

This research project is very helpful to bridge the gap between the theoretical knowledge and real-life experience as part of Bachelor of Science (BSc) program. This report has been designed to have a practical experience through the theoretical understanding.

We also acknowledge our profound sense of gratitude to all the teachers who have been instrumental for providing us the technical knowledge and moral support to complete the project with full understanding.

It is imperative to show our appreciation for our honorable faculty member Mr. Rifat Ahmed Hassan for his undivided attention and help to achieve this milestone. Also, our gratefulness is divine to the North South University, ECE department for providing us a course such as CSE 498R in which we could really work on this project and materialize it the way we have dreamt of.

We thank our friends and family for their moral support to carve out this project and always offer their support.

# ABSTRACT

In recent years, online social digital content has gained widespread interest for research in the field of computer science. Sentiment analysis is one of the major tasks of text mining that categorizes emotions using Natural language processing (NLP). This paper contributes to classifying celebrity sentiment analysis, which can be used to analyze unstructured views of celebrities and people's opinions on those views that are either positive, negative, or neutral. To perform classification, we first pre-processed the dataset, extracted features, and then applied machine learning classification algorithms including Naive Bayes (NB), Logistic Regression (LR), Decision Tree (DT), Support Vector Machine (SVM), and Random Forest (RF). The performance of each algorithm has been evaluated using different evaluation metrics such as accuracy, precision, recall, F1-score, and AUC score.

# TABLE OF CONTENT

# LIST OF FIGURES

# INTRODUCTION

Sentiment analysis, also referred to as opinion mining, is the application of the NLP (Natural Language Processing) technique that analyzes the texts written by a person to deter- mine the emotions they're most likely feeling towards certain entities [1]. Social media is a popular internet-based platform that allows users to have conversations, share information and emotions. Social media is used for both personal and professional purposes. People can use social media profes-sionally to expand their professional networks and improve their expertise in a particular sector. On personal levels, social media enables people to communicate with friends and family and be entertained [2].

Most celebrities use social media on a regular basis, as it gives them the chance to interact with their followers, advance their careers, and ultimately boost their stardom. In many respects, including from a psychological, sociological, and political perspective, celebrity behavior and messaging have a significant impact on people's behavior. NLP stands for Natural Language Processing, a subfield of artificial intelligence that deals with the interaction between computers and humans in natural language. It combines com- puter science, linguistics, and machine learning to study how computers and humans communicate in natural language [3]. It gives computers the ability to interpret, manipulate, and comprehend human language, which not only increases the effectiveness of human work but also facilitates communication with machines [4].

The two main phases of natural language processing are data preprocessing and algorithm development. Data preprocessing is the process of preparing and cleaning text data for machines to be able to analyze it. Algorithm development can be based on lexicon, deep learning, and machine learning models [5]. Sentiment analysis, question generation and answering, email filtering, chatbots, and summarization are some of the prominent applications of NLP. The principal aim of this paper is to build a system for classifying the emotion of celebrity posts, whether it is positive, negative, or neutral.

The rest of the paper is organized as follows: Discussion of related research works is covered in Section II. Demonstration of the methodology of our system is covered in Section III. The description of the dataset is in Section IV. The results and their analysis are depicted in Section V. Finally, the findings of this research are summarized in Section VI.

# LITERATURE REVIEW

Natural language processing (NLP) methods have attracted a lot of attention recently for the sentiment analysis of textual data pertaining to celebrities. To investigate how the general public feels about celebrities in various fields, researchers have used a variety of approaches. For example, sentiment was estimated using lexicon-based techniques and sentiment dictionaries in a study focused on analyzing tweets referencing celebrities [6]. The results revealed considerable differences in sentiment across categories such as sports, entertainment, and politics. Another paper, [7] investigated machine learnIng algorithms for celebrity sentiment analysis, leveraging labeled datasets of social media postings and user evaluations. The researchers tested supervised learning algorithms such as Naive Bayes, Support Vector Machines, and Random Forests and found encouraging results. Furthermore, another research [8] discussed difficulties in celebrity sentiment anal ysis, highlighting the difficulty of dealing with sarcasm and ambiguity.

To improve sentiment analysis models, proposed strategies included sarcasm detection algorithms and context- based analysis. An assessment of celebrity sentiment analysis's practical applications highlighted its significance for brand management [9]. The study investigated the relationship be- tween public perception of celebrities and their effect on brand perception and customer behavior. Negative emotion toward a celebrity has been shown to harm brand reputation, whilst good sentiment can boost brand loyalty and engagement.

This emphasizes the importance of sentiment research in enhancing marketing tactics and controlling celebrity brand relationships. These works give insight on the significance of celebrity sentiment analysis, its limitations, and its many applications in a variety of sectors. This emphasizes the importance of senti- ment research in enhancing marketing tactics and controlling celebrity brand relationships. These works give insight on the significance of celebrity sentiment analysis, its limitations, and its many applications in a variety of sectors.

# METHODOLOGY

## SYSTEM ARCHITECTURE

The model design for celebrity sentiment analysis entails applying natural language processing (NLP) tools to perform a supervised learning approach. The model is fed a dataset of celebrity-related textual data that has been preprocessed and tagged with sentiment information as input. To normalize the text, preprocessing techniques include tokenization, stop word removal, and maybe stemming or lemmatization. The model then represents the textual data using various linguistic properties such as word frequencies, n-grams, or word embed- dings. [10] These features are input into a machine learning algorithm that is trained on the labeled dataset to understand the correlations between the features and sentiment labels, such as logistic regression, SVM, or random forest. The model improves its parameters throughout the training phase in order to decrease prediction errors. Once trained, the model may be used to categorize fresh, unseen text as positive, negative, or neutral sentiment. This model architecture successfully analyzes and classifies sentiment toward celebrities using NLP and machine learning, providing insights into public opinion and guiding decision-making processes in a variety of sectors.

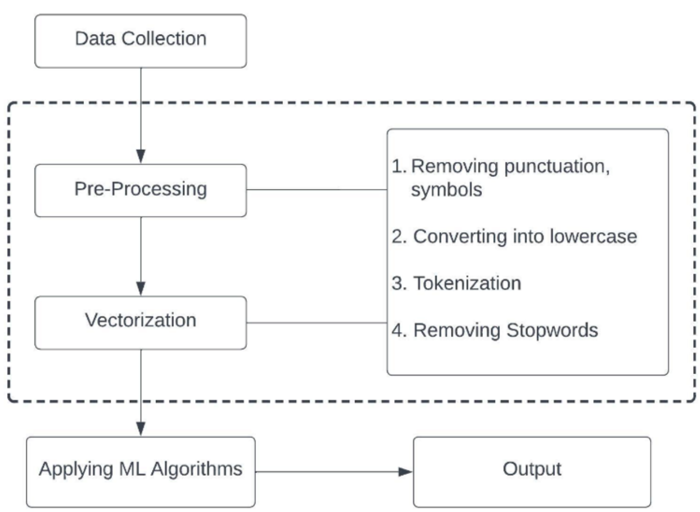



Figure 1 System Architecture

## DATA PREPROCESSING

Some data preprocessing techniques had to be applied to convert the dataset it into a format such that machine learning algorithms could be applied upon them. The following pre- processing steps were applied to the dataset:

- **Tokenization:** Process of breaking a sen- tence/phrase/paragraph into smaller bits such as words. In our dataset each of the tweets were tokenized into words.
- **Punctuation Marks Removal:** All punctuation marks were removed from the sentence as they have no significance in textual sentiment classification.
- **Hyper-Link Removal:** Hyperlinks in the data generally contain URLS to external webpages and hence is irrelevant for the classification of sentiment.
- **Stop Words Removal:** Stop words are a set of commonly used words but hold very little useful information.
- **Remove Emojis:** Replacing emojis with words using emoticon function.

## TRAIN/TEST SPLIT

The prepossessed data was split into the traditional 80-20 test train split respectively which means 80 percent of the data was used for training while the rest 20 was used for testing purposes.

## VECTORIZATION

Machine learning models cannot comprehend textual data and hence the data needs to be converted to a numerical format before feeding it to the machine learning models. Therefore, to convert the textual data into a vector form we have used Term Frequency - Inverse Document Frequency (TF-IDF) and Counter Vectorizer.

$$\mathrm{tf}(t,d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}} \quad (1)$$

$$\mathrm{idf}(t,D) = \log \frac{N}{|\{d \in D : t \in d\}|} \quad (2)$$

$$TF - IDF = CountVectorizer(t,d) * IDF(t) \quad (3)$$

## CLASSIFICATION

We have used the test-train technique to divide the dataset into 80 percent for training and 20 percent. Then applied different classification algorithms. Applied algorithms are Random Forest, Logistic Regression, Support Vector Machine, Decision Tree and Naive Bayes.

# DATASET

The dataset for this research was taken from Kaggle and initially consisted of 2 million tweets about celebrities. However, a sample of 250,000 tweets was chosen as the representative dataset for this investigation. There are total of 110,000 positive tweets, 105,000 negative tweets and 19,000 neutral tweets.[13]. To confirm the findings' reliability and generalizability, the dataset was randomly divided into a training set and a testing set in an 80-20 ratio. The training set, which included 80 percent of the tweets, was used for model training and parameter tweaking, while the testing set, which included the remaining 20 percent, was utilized as an independent assessment set to evaluate the model's performance. A sentiment column was added to the dataset to assist sentiment analysis using the Vader Sentiment tool, a prominent sentiment analysis tool particularly tailored for social media content. Vader Sentiment classifies text into positive, negative, or neutral sentiment categories using a combination of lexical and grammatical heuristics as well as a pre-trained model.

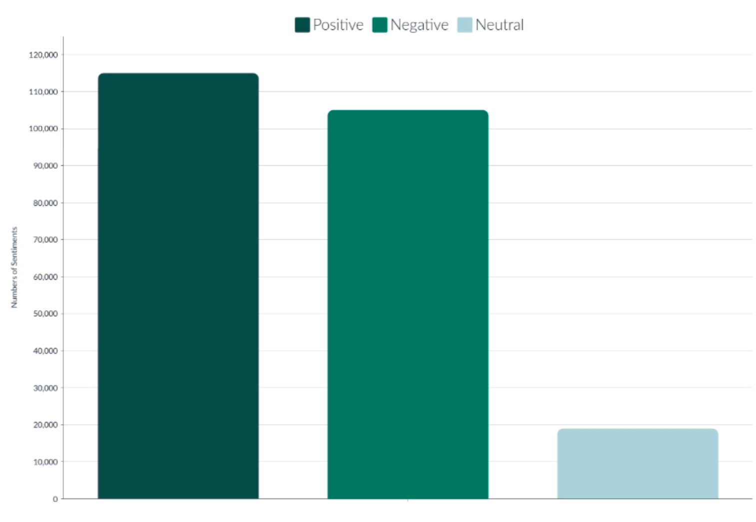



Figure 2 Data Visualization

# RESULT AND ANALYSIS

The construction and assessment of numerous machine learning algorithms was required for the sentiment analysis of celebrity-related textual data utilizing NLP techniques. Among the algorithms used and shown in Table 1, logistic regression proved to be the most accurate, with an astounding accuracy of 92.3 percent. Furthermore, with an accuracy of 89.7 percent, SVM displayed great performance in classifying sentiment in celebrity-related material. The ability of SVM to segregate data points in high-dimensional space using hyperplanes con- tributed to its success in sentiment analysis. The accuracy of Random Forest, a prominent ensemble learning technique, was 83.4 percent, demonstrating its ability to capture com- plex linkages and interactions between characteristics in the dataset. Random Forest's ensemble technique, which mixes many decision trees, proven useful in sentiment categorization. The Decision Tree algorithm performed well, with an accuracy of 82.3 percent. Decision Trees are well-known for their interpretability and simplicity, making them excellent for deriving relevant insights from the sentiment analysis process. Naive Bayes, on the other hand, shown potential in sentiment analysis while obtaining a lower accuracy of 73.03 percent. Naive Bayes is a probabilistic technique that assumes feature independence, which may not be true in all circumstances of sentiment analysis. However, when paired with other algorithms or preprocessing approaches, it can still yield significant insights. When the accuracy of various algorithms is compared, logistic regression out-performs the other alternatives. The logistic regression model's capacity to learn and generalize from training data, as well as capture complicated correlations between characteristics, most certainly contributed to its improved performance. In our evaluation, we considered several performance measures to assess the classification models. These measures include accuracy, precision, recall, specificity, and F1-score.

| Algorithms | Accuracy | Precision | Recall | F1 - Score |
|---|---|---|---|---|
| Random Forest | 83.4% | 84.0% | 83.0% | 82.0% |
| Logistic Regression | 92.3% | 92.0% | 92.0% | 92.0% |
| SVM | 89.7% | 90.0% | 90.0% | 90.0% |
| Decision Tree | 83.2% | 84.0% | 83.0% | 83.0% |
| Naive Bayes | 73.7% | 79.0% | 73.0% | 75.0% |

Figure 3 Algorithms and its corresponding values

## FORMULAE

Accuracy: Overall performance of the classification model.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

Precision: Ratio of true positive predictions to all positive predictions.

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

Recall: Ratio of true positive predictions to the total number of actual positive instances.

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

F1-score: Weighted measure combining both precision and recall.

$$F1 - Score = \frac{2 * Precision * Recall}{Precision + Recall} \qquad (7)$$

# CONCLUSION

While accuracy is a significant parameter for evaluating sentiment analysis models, additional metrics such as precision, recall, and F1-score should also be studied to give a complete picture of their performance.[11] Finally, the examination of several machine learning algorithms in celebrity sentiment analysis proved the efficacy of NLP approaches in collecting and categorizing sentiment. The excellent accuracy of logistic regression strengthens its applicability for sentiment analysis tasks using celebrity-related textual data. Future study should look at the possibilities of these algorithms, as well as ensemble methods and advanced deep learning techniques, in order to improve the accuracy and robustness of sentiment analysis in the domain of celebrity sentiment analysis utilizing NLP. [12]

# REFERENCE

[1] I. Smeureanu and M. Zurini, "Spam Filtering For Opti- mization in Internet Promotions Using Bayesian Analysis," J. Appl. Quant. Methods, vol. 5, no. 2, 2010.

[2] H. Wang, D. Can, A. Kazemzadeh, F. Bar, and S. Narayanan, "A System for Real-time Twitter Sentiment Analy- sis of 2012 U.S. Presidential Election Cycle," Proc. 50th Annu. Meet. Assoc. Comput. Linguist., no. July, pp. 115–120, 2012.

[3] P. Goncalves, B. Fabr´ıcio, A. Matheus, and C. Meey- oung, "Comparing and Combining Sentiment Analysis Meth- ods Categories and Subject Descriptors," Proc. first ACM Conf. Online Soc. networks, pp. 27–38, 2013.

[4] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up?: sentiment classification using machine learning techniques," Proc. Conf. Empir. Methods Nat. Lang. Process., pp. 79–86, 2002.

[5] S. J. M. Modha, Jalaj S. , Gayatri S. Pandi, "Automatic Sentiment Analysis for Unstructured Data," Int. J. Adv. Res. Comput. Sci. Softw. Eng., vol. 3, no. 12, pp. 91–97, 2013.

[6] M. Ahmad, S. Aftab, S. S. Muhammad, and U. Waheed, "Tools and Techniques for Lexicon Driven Sentiment Analy- sis: A Review," Int. J. Multidiscip. Sci. Eng., vol. 8, no. 1, pp. 17–23, 2017.

[7] K. Mouthami, K. N. Devi, and V. M. Bhaskaran, "Sen- timent analysis and classification based on textual reviews," 2013 Int. Conf. Inf. Commun. Embed. Syst., pp. 271–276, 2013.

[8] K. N. Devi and V. M. Bhaskarn, "Online forums hotspot prediction based on sentiment analysis," J. Comput. Sci., vol. 8, no. 8, pp. 1219–1224, 2012.

[9] P. S. Earle, D. C. Bowden, and M. Guy, "Twitter earth- quake detection: Earthquake monitoring in a social world," Ann. Geophys., vol. 54, no. 6, pp. 708–715, 2011.

[10] M. Cheong and V. C. S. Lee, "A microblogging-based approach to terrorism informatics: Exploration and chronicling civilian sentiment and response to terrorism events via Twit- ter," Inf. Syst. Front., vol. 13, no. 1, pp. 45–59, 2011.

[11] A. Go, R. Bhayani, and L. Huang, "Twitter Sentiment Classification using Distant Supervision," Processing, vol. 150, no. 12, pp. 1–6, 2009.

[12] B. Pang and L. Lee, "Opinion mining and sentiment analysis," Found. Trends Inf. Retr., vol. 2, no. 1–2, pp. 1–135, 2008.

[13] A. S. SAKIB, "Kaggle," 2022. [Online]. Available: https://www.kaggle.com/datasets/ahmedshahriarsakib/top- 1000-twitter-celebrity-tweets-embeddings.

# CODE

In [6]:
```
dataset.head()
```
Out[6]:

| | twitter_id | tweet |
|---|---|---|
| 0 | 1542505731336708098 | b'\xe2\x80\x9cHONEY\xe2\x80\x9d out now!\n\nht... |
| 1 | 1535435111293865986 | b'RT @RollingStone: Smile for the camera, guys... |
| 2 | 1535330970630381569 | b'Thank you to everyone for your continued sup... |
| 3 | 1535330620204716032 | b'Go stream \xe2\x80\x9cFALL\xe2\x80\x9d out n... |
| 4 | 1518946703793086466 | b'RT @bigtimerush: Vegas are you ready!? \n\nW... |

## Preprocessing

In [5]:
```
import codecs

dataset['tweet'] = dataset['tweet'].apply(lambda x: codecs.decode(x, 'unicode_escape'))
dataset['tweet'] = dataset['tweet'].apply(lambda x: x.replace("â••", "").replace("â••", ""))
```

In [9]:
```
emoticon_mapping = {
    ":)": "smile",
    ":(": "sad",
    ":D": "laugh",
    ":P": "playful",
    ":|": "neutral",
    ":/": "confused",
    ";)": "wink",
    ":O": "surprised"
}

def replace_emoticons(tweet):
    for emoticon, replacement in emoticon_mapping.items():
        tweet = tweet.replace(emoticon, replacement)
    return tweet

dataset['tweet'] = dataset['tweet'].apply(replace_emoticons)
dataset.head()
```
Out[9]:

| | twitter_id | tweet |
|---|---|---|
| 0 | 1542505731336708098 | b'HONEY out now! |
| 1 | 1535435111293865986 | b'RT : Smile for the camera, guys! has releas... |
| 2 | 1535330970630381569 | b'Thank you to everyone for your continued sup... |
| 3 | 1535330620204716032 | b'Go stream FALL out now! Fun fact: We weare i... |
| 4 | 1518946703793086466 | b'RT : Vegas are you ready!? Weare officially ... |

## Model Building

In [ ]:
```
#Vader Sentiment
```

In [24]:
```
from nltk.sentiment import SentimentIntensityAnalyzer

nltk.download('vader_lexicon')
sia = SentimentIntensityAnalyzer()

song_titles = ["Another Song Title", "Yet Another Song"]

assigned_sentiments = []

for tweet in dataset['tweet']:
    sentiment_scores = sia.polarity_scores(tweet)
    compound_score = sentiment_scores['compound']

    for title in song_titles:
        if title.lower() in tweet.lower():
            sentiment = 'Positive'
            break
    else:
        if compound_score >= 0.05:
            sentiment = 'Positive'
        elif compound_score <= -0.05:
            sentiment = 'Negative'
        else:
            sentiment = 'Neutral'

    assigned_sentiments.append(sentiment)


dataset['sentiment'] = assigned_sentiments

dataset.to_csv('merged.csv', index=False)

print("Sentiments added to the dataset and saved to 'merged.csv'.")
```

Predicted

```
              precision    recall  f1-score   support

    Negative       0.33      0.71      0.45      7350
     Neutral       0.83      0.60      0.70     28021
    Positive       0.86      0.84      0.85     39640

    accuracy                           0.74     75011
   macro avg       0.67      0.72      0.67     75011
weighted avg       0.79      0.74      0.75     75011
```

In [106]:
```
fig, ax = plt.subplots()

ax.hist(Y_test, alpha=0.5, label='Actual Sentiments')

ax.hist(Y_pred, alpha=0.5, label='Predicted Sentiments')

ax.set_xlabel('Sentiment')
ax.set_ylabel('Count')
ax.set_title('Histogram of Predicted and Actual Sentiments')

ax.legend()

plt.show()
```

## Model Training

In [ ]:
```
#Logistic Regression
```

In [86]:
```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MaxAbsScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

X = dataset['tweet']
Y = dataset['sentiment']
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=42)

vectorizer = CountVectorizer()
X_train_vectorized = vectorizer.fit_transform(X_train)
X_test_vectorized = vectorizer.transform(X_test)

scaler = MaxAbsScaler()
X_train_scaled = scaler.fit_transform(X_train_vectorized)
X_test_scaled = scaler.transform(X_test_vectorized)

model = LogisticRegression(max_iter=1000)
model.fit(X_train_scaled, Y_train)

Y_pred = model.predict(X_test_scaled)
accuracy = accuracy_score(Y_test, Y_pred)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.9356090147379367
```

In [87]:
```
results = pd.DataFrame({'Original Sentiment': Y_test}).reset_index(drop=True)
results['Predicted Sentiment'] = Y_pred

print(results)
```

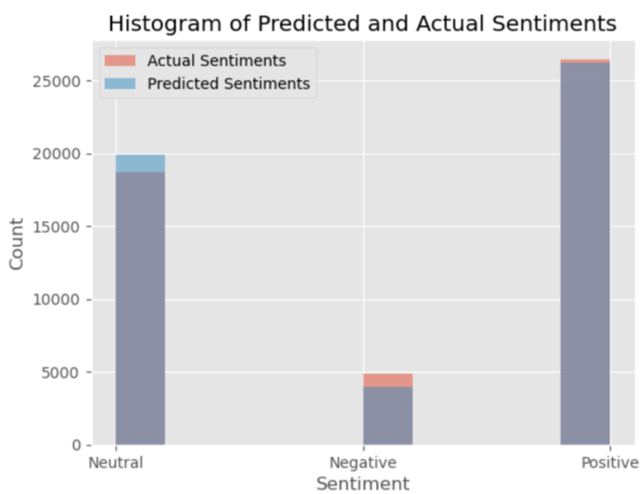
```
      Original Sentiment Predicted Sentiment
```

In [102]:
```
#Naive Bayes
```

In [103]:
```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MaxAbsScaler
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, classification_report
from imblearn.over_sampling import RandomOverSampler

X = dataset['tweet']
Y = dataset['sentiment']

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3, random_state=42)

vectorizer = CountVectorizer()
X_train_vectorized = vectorizer.fit_transform(X_train)
X_test_vectorized = vectorizer.transform(X_test)

scaler = MaxAbsScaler()
X_train_scaled = scaler.fit_transform(X_train_vectorized)
X_test_scaled = scaler.transform(X_test_vectorized)

oversampler = RandomOverSampler(random_state=42)
X_train_balanced, Y_train_balanced = oversampler.fit_resample(X_train_scaled, Y_train)

model = MultinomialNB()
model.fit(X_train_balanced, Y_train_balanced)

Y_pred = model.predict(X_test_scaled)

accuracy = accuracy_score(Y_test, Y_pred)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.737385183506419
```

In [104]:
```
results = pd.DataFrame({'Original Sentiment': Y_test}).reset_index(drop=True)
results['Predicted Sentiment'] = Y_pred

print(results)
```




In [ ]:
```
#Random Forest
```

n [90]:
```
dataset1 = dataset.head(50000)
```

n [91]:
```
print(dataset1.shape)
```

```
(50000, 3)
```

In [98]:
```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MaxAbsScaler
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report
from imblearn.over_sampling import RandomOverSampler

X = dataset2['tweet']
Y = dataset2['sentiment']

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=42)

vectorizer = CountVectorizer()
X_train_vectorized = vectorizer.fit_transform(X_train)
X_test_vectorized = vectorizer.transform(X_test)

scaler = MaxAbsScaler()
X_train_scaled = scaler.fit_transform(X_train_vectorized)
X_test_scaled = scaler.transform(X_test_vectorized)

oversampler = RandomOverSampler(random_state=42)
X_train_balanced, Y_train_balanced = oversampler.fit_resample(X_train_scaled, Y_train)

model = SVC(kernel='linear')
model.fit(X_train_balanced, Y_train_balanced)

Y_pred = model.predict(X_test_scaled)

accuracy = accuracy_score(Y_test, Y_pred)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.8915
```

In [99]:
```
results = pd.DataFrame({'Original Sentiment': Y_test}).reset_index(drop=True)
results['Predicted Sentiment'] = Y_pred

print(results)
```