SENTIMENT ANALYSIS IN MARKETING

PROJECT TITLE : SENTIMENT ANALYSIS IN MARKETING.

***Phase 5:* overall submission of the project.**

TOPIC: Developing SENTIMENT ANALYSIS model respectively.

SYNOPSIS FOR SENTIMENTAL ANALYSIS FOR MARKETING :

1. Introduction for project submission
2. Design thinking
3. Sentimental Analysis-Innovation
4. Sentimental Analysis-Development Part-1
5. Sentimental Analysis-Development Part-2
6. Conclusion for project submission

SENTIMENT ANALYSIS IN MARKETING

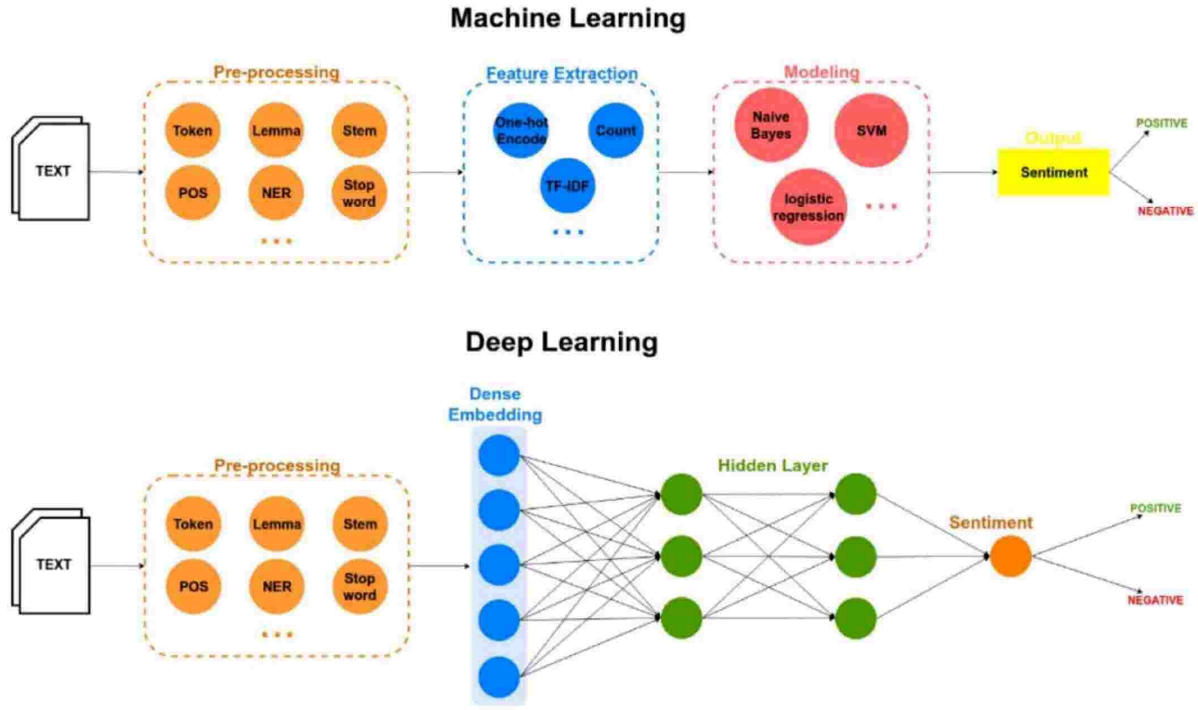
INTRODUCTION:

- **With advancements in technology and fields like deep learning, sentiment analysis is becoming more and more common for companies that want to gauge their customers' sentiments.**
- **Today, businesses use natural language processing, statistical analysis, and text analysis to identify the sentiment and classify words into positive, negative, and neutral categories.**
- **The best companies understand the importance of understanding their customers' sentiments – what they are saying, what they mean and how they are saying. You can use xsentiment analysis to identify customer sentiment in comments, reviews, tweets, or social media platforms where people mention your brand.**

As sentiment analysis is the domain of understanding emotions using software,

we have prepared a complete guide to understand ‘what is sentiment analysis?’,

its tools, and different classifications and use cases.



PHASE - I

Problem Definition and Design Thinking

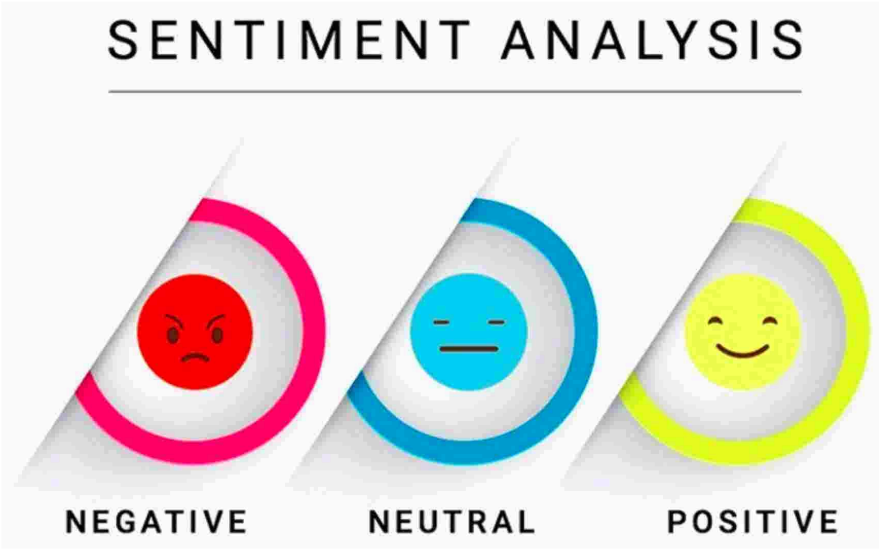
SENTIMENTAL ANALYSIS FOR MARKETING

Project No.	5
Project Name	Sentimental Analysis for marketing

ABSTRACT :

Sentiment analysis plays a pivotal role in modern marketing strategies by extracting valuable insights from consumer-generated content across various digital platforms. This abstract introduces a comprehensive framework comprising modules designed to facilitate sentiment analysis for marketing professionals.

Sentiment analysis, a crucial component of modern marketing strategies, has witnessed a transformative evolution with the integration of Natural Language Processing (NLP) techniques. This abstract presents an overview of the application of NLP in sentiment analysis for marketing, highlighting its significance, key methodologies, and implications for enhancing customer engagement and decision-making.



Problem Statement:

-Clearly define the problem statement, such as "How might we effectively analyze customer sentiment to enhance marketing strategies and customer

engagement? "

- User Personas: Create user personas representing marketing professionals and stakeholders to keep their perspectives in mind throughout the design process.

DATA SOURCE:

A good data source for Sentimental analysis for marketing using nlp should be Accurate, Complete, Covering the reviews of customers from all possible ways like Social Media, Direct review and trends of products.

Dataset Link: https://www.kaggle.com/datasets/crowdflower/twitter-airline-sentiment

DESIGN THINKING:

1. **Empathize:**

- Understand the Business Context: Begin by gaining a deep understanding of the marketing objectives, target audience, and the specific challenges or goals related to sentiment analysis.

- User Research: Conduct interviews and surveys to gather insights from marketing professionals, stakeholders, and end-users about their needs, pain points, and expectations regarding sentiment analysis.

2. **Ideate:**

- Brainstorm Solutions: Engage cross-functional teams in brainstorming sessions to generate innovative ideas for sentiment analysis tools and techniques.

- Ideation Workshops: Host workshops to encourage creative thinking, considering both technical and non-technical solutions.

3. **Data Collection and Pre-processing Module:**

- Efficient data collection from diverse sources, including social media, reviews, forums, and surveys.

- Text pre-processing techniques for cleaning and normalization, including tokenization, stemming, and stop-word removal.

- Integration of natural language processing (NLP) libraries for language detection and encoding conversion.

4. **Sentiment Analysis Module:**

- Utilization of state-of-the-art machine learning and deep learning models for sentiment classification, such as LSTM, BERT, and Transformer models.

- Fine-tuning of pre-trained models on domain-specific data to improve accuracy and relevance.

- Integration of sentiment lexicons and dictionaries for rule-based sentiment analysis, enhancing interpretability.

- Bag of Words (BoW) is a Natural Language Processing technique of text modeling. It is a method of feature extraction with text data. Whenever we apply any algorithm in NLP, it works on numbers. We cannot directly feed our text into that algorithm. Hence, Bag of Words model is used to preprocess the text by converting it into a bag of words, which keeps a count of the total occurrences of most frequently used words.

- Word embeddings are a way to represent words as dense vectors in a lower-dimensional space. They are used in natural language processing (NLP) to encode the meaning of words such that similar words have similar vector representations1. Word embeddings are typically real-valued vectors that capture semantic and syntactic information2. They are learned from large text corpora using algorithms like Word2Vec and GloVe.

5. **Visualization and Reporting Module:**

- Creation of interactive dashboards and visualizations to present sentiment insights in a user-friendly manner.

- Real-time monitoring of sentiment trends and sentiment scores over time.

- Generation of actionable reports and recommendations to guide marketing strategies based on sentiment analysis outcomes.

This framework offers marketing professionals a structured approach to harness the power of sentiment analysis for data-driven decision-making. It empowers them to gauge consumer sentiment accurately, identify areas of improvement, and optimize marketing campaigns, product development, and customer engagement strategies accordingly. The integration of cutting-edge NLP techniques ensures the framework's adaptability to evolving language patterns and sentiment nuances. By employing this modular approach, marketing teams can enhance their competitive edge in a dynamic and data-intensive environment.

6. Labeling

Assign sentiment labels to your data, typically as positive, negative, or neutral sentiments. This can be done manually or using pre-labeled datasets if available.

7. Feature Extraction

Convert the text data into numerical features that machine learning models can understand. Common techniques include TF-IDF (Term Frequency-Inverse Document Frequency) or word embeddings like Word2Vec or GloVe.

8. Model Selection

Choose a suitable machine learning or deep learning model for sentiment analysis. Common models include logistic regression, support vector machines (SVM), recurrent neural networks (RNNs), or transformer-based models like BERT.

9. Training

Train your selected model on the labeled data. Use techniques like cross-validation to optimize model performance and prevent overfitting.

10. Evaluation

Assess the model's performance using metrics such as accuracy, precision, recall, F1-score, or ROC-AUC, depending on your specific goals and dataset characteristics.

Methodologies:

The application of NLP in sentiment analysis for marketing encompasses several methodologies and techniques:

1. **Text Preprocessing:**

NLP techniques are employed to clean and preprocess textual data, including tokenization, stop-word removal, and stemming, ensuring the data's quality and consistency.

2. **Feature Extraction:**

NLP enables the extraction of meaningful features from text, such as n-grams, word embeddings, and semantic representations, which are essential for sentiment analysis models.

3. **Sentiment Lexicons:**

NLP-driven sentiment lexicons, dictionaries, and word embeddings are leveraged to assign sentiment scores to words and phrases, facilitating the classification of text into positive, negative, or neutral sentiments.

4. **Machine Learning Models:**

NLP-powered machine learning models, including Natural Language Processing models (e.g., LSTM, BERT), are employed for sentiment classification tasks, achieving state-of-the-art accuracy.

5. **Aspect-Based Analysis:**

NLP allows for aspect-based sentiment analysis, enabling businesses to dissect customer feedback and sentiments related to specific product features or attributes, thereby informing product development and marketing strategies.

Applications:

Sentiment analysis in marketing using NLP finds applications across various domains, including but not limited to:

- Product Reviews: Analyzing customer reviews on e-commerce platforms to gauge product satisfaction and identify areas for improvement.

- Social Media Monitoring: Tracking brand mentions and customer sentiment on social media platforms to assess the success of marketing campaigns.

- Customer Service: Assessing the sentiment of customer service interactions to improve the customer experience.

- Market Research: Analyzing sentiment in market research surveys and open-ended responses to gain insights into consumer preferences and trends.

SENTIMENT ANALYSIS FOR MARKETING

PHASE 2

Project : Sentient Analysis For Marketing

Introduction:

Sentiment analysis using BERT and RoBERTa models is a powerful approach to extract sentiment information from text data. These models, based on transformer architecture, have achieved state-of-the-art performance on various NLP tasks, including sentiment analysis. In this introduction, I'll walk you through the steps to perform sentiment analysis using the Hugging Face Transformers library, which provides pre-trained BERT and RoBERTa models.

Data Collection and Preprocessing:

- Importing the dataset : Obtain a comprehensive dataset containing relevant features such as tweet count, tweet timezone, tweet id, etc. ,
- Data pre-processing : Clean the data by handling missing values, outliers and categorical variables. Standardize or normalize numerical features

Exploratory Data Analysis(EDA):

- Visualize and analysis the dataset to gain insights into the relationship between variables.
- Identify correlations and patterns that can inform features selected and engineering

ADVANCED TECHNIQUES:

- BERT or RoBERTa for Text Embeddings:

First, you can use BERT or RoBERTa to generate text embeddings (vectors) for your text data. These embeddings capture the semantic information of the text, which you can then use as input to a regression model.

- Random Forest Regressor:

Random Forest is an ensemble learning method that can handle both

regression and classification tasks effectively. It's known for its ability to capture complex relationships in the data.

- Gradient Boosting Regressor (e.g., XGBoost, LightGBM, or CatBoost):

Gradient boosting algorithms often provide excellent predictive performance by combining the predictions of multiple weak learners. Each of these libraries (XGBoost, LightGBM, and CatBoost) has its advantages and can be fine-tuned for optimal results.

DATA SOURCE:

A good data source for Sentimental analysis for marketing using nlp should be Accurate, Complete, Covering the reviews of customers from all possible ways like Social Media, Direct review and trends of products.

Dataset Link: https://www.kaggle.com/datasets/crowdflower/twitter-airline-sentiment

PROGRAM:

SENTIMENT ANALYSIS FOR MARKETING

IMPORTING DEPENDENCIES:

```
import pandas as pd
import numpy as np
import torch
import tokenize
import seaborn as sns
import matplotlib.pyplot as plt
import nltk
import tensorflow as tf
from sklearn.model_selection import train_test_split
```

```
from sklearn. metrics import accuracy_score, classification_report
from transformers import BertTokenizer, BertForSequenceClassification, Trainer, TrainingArguments
from skearn. linear_model import LinearRegression
from sklearn. ensemble import RandomForestRegressor
import xgboost as xg
```

Loading Data:

```
dataset=pd. read_csv('Tweets. csv')
dataset. info()
print(dataset. shape)
print(dataset['airline_sentiment']. value_counts())
```

Out[1]:

```
<class 'pandas. core. frame. DataFrame'>
RangeIndex: 14640 entries, 0 to 14639
Data columns (total 15 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   tweet_id                     14640 non-null  int64
 1   airline_sentiment            14640 non-null  object
 2   airline_sentiment_confidence 14640 non-null  float64
 3   negativereason               9178 non-null   object
 4   negativereason_confidence    10522 non-null  float64
 5   airline                      14640 non-null  object
 6   airline_sentiment_gold       40 non-null     object
 7   name                         14640 non-null  object
 8   negativereason_gold          32 non-null     object
 9   retweet_count                14640 non-null  int64
 10  text                         14640 non-null  object
 11  tweet_coord                  1019 non-null   object
 12  tweet_created                14640 non-null  object
 13  tweet_location               9907 non-null   object
 14  user_timezone                9820 non-null   object
dtypes: float64(2), int64(2), object(11)
memory usage: 1. 7+ MB
(14640, 15)
```

```
negative    9178
neutral     3099
positive    2363
Name: airline_sentiment, dtype: int64
```

Pre-Process the Data:

```
def preprocess_text(text):
    # Remove punctuations and numbers
    text = re.sub('[^a-zA-Z]', ' ', text)

    # Single character removal
    text = re.sub(r'\s+[a-zA-Z]\s+', ' ', text)

    # Removing multiple spaces
    text = re.sub(r'\s+', ' ', text)

    # Converting to Lowercase
    text = text.lower()

    # Lemmatization
    #text = text.split()
    #lemmatizer = WordNetLemmatizer()
    #text = [lemmatizer.lemmatize(word) for word in text if not word in
set(stopwords.words('english'))]
    #text = ' '.join(text)

    return text

# Apply the preprocessing to the 'text' column
df['text'] = df['text'].apply(preprocess_text)

# Display the first 5 rows of the dataframe after preprocessing
df.head()
output:
```

S.no	airline_sentiment	text

```
----
0       neutral         virginamerica what dhepburn said
1       positive              virginamerica plus you ve added commercials t. . .
2       neutral         virginamerica didn today must mean need to ta. . .
3       negative              virginamerica it really aggressive to blast o. . .
4       negative              virginamerica and it a really big bad thing a. . .
```

DATA CLEANING:

```
data = data[['airline_sentiment', 'text']]

data['airline_sentiment'] = data['airline_sentiment']. map({'positive': 2, 'neutral': 1, 'negative': 0})
```

SPLIT THE DATA INTO TRAINING AND TESTING SETS:

```
X = data['text']

y = data['airline_sentiment']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0. 2, random_state=42)
```

REGRESSION MODELS:

LOGISTIC REGRESSION:

```
model=LogisticRegression(max_iter=10000)

model. fit(train_vec, train_labels)

Output : LogisticRegression(max_iter=10000)
```

RANDOM FORESTING:

```
rf_classifier = RandomForestClassifier(n_estimators=100, random_state=42)

rf_classifier. fit(X_train_tfidf, y_train)

rf_predictions = rf_classifier. predict(X_test_tfidf)
```

Output:

Classification Report for Random Forest:

	precision	recall	f1-score	support
negative	0.79	0.93	0.85	1889
neutral	0.58	0.36	0.44	580
positive	0.73	0.56	0.64	459
accuracy			0.76	2928
macro avg	0.70	0.62	0.64	2928
weighted avg	0.74	0.76	0.74	2928

```
r_train_accuracy, r_test_accuracy, r_train_auc, r_test_auc=
check_scores(RandomForestClassifier(random_state=0).fit(x_train, y_train),
x_train, x_test, y_train, y_test)
```

Output:

```
Train confusion matrix is:
[[6829   26]
 [   5 1795]]

Test confusion matrix is:
[[2215  108]
 [ 238  325]]
```

	precision	recall	f1-score	support
0	0.90	0.95	0.93	2323
1	0.75	0.58	0.65	563
accuracy			0.88	2886
macro avg	0.83	0.77	0.79	2886
weighted avg	0.87	0.88	0.87	2886

```
Train accuracy score: 0.996418255343732
Test accuracy score: 0.8801108801108801

Train ROC-AUC score: 0.9982442661479861
Test ROC-AUC score: 0.8956867344777572

Are under Precision-Recall curve: 0.6526104417670683
```

Area under ROC-AUC: 0.7441899264879837

GRADIANT BOOSTING CLASSIFICATION:

```
gb_classifier = GradientBoostingClassifier(n_estimators=100, random_state=42)
gb_classifier.fit(X_train_tfidf, y_train)
gb_predictions = gb_classifier.predict(X_test_tfidf)
```

Output:

Classification Report for Gradient Boosting:

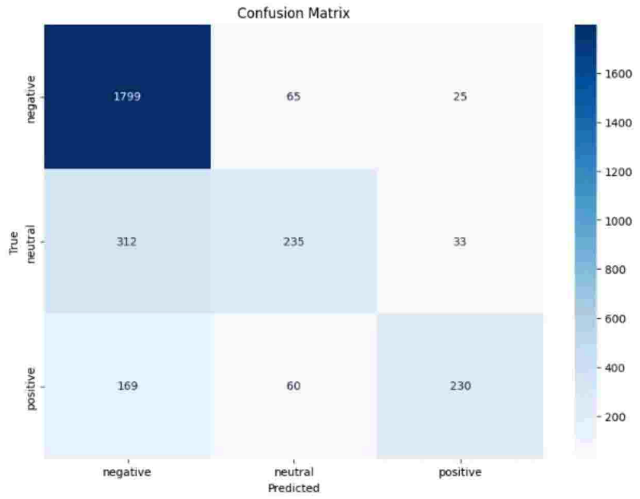
	precision	recall	f1-score	support
negative	0.76	0.96	0.85	1889
neutral	0.67	0.24	0.35	580
positive	0.74	0.54	0.63	459
accuracy			0.75	2928
macro avg	0.72	0.58	0.61	2928
weighted avg	0.74	0.75	0.71	2928

PLOTING THE REGRESSION MODELS:

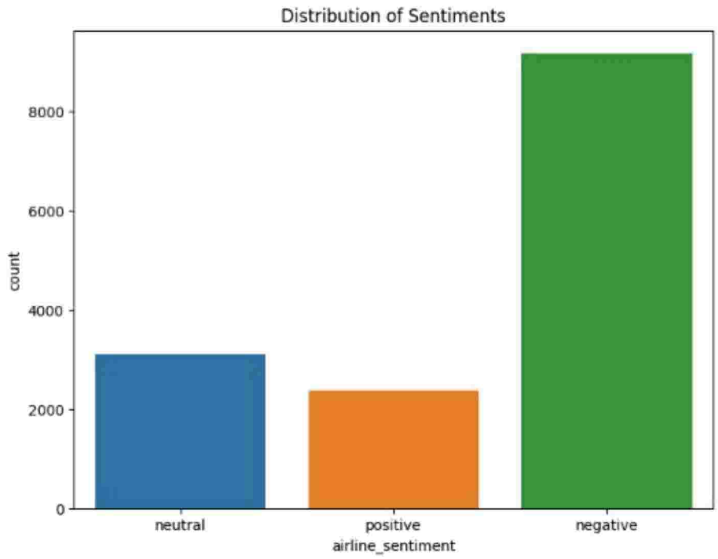
CONFUSION MATRIX

```
def plot_confusion_matrix(y_test, y_pred):
    cm = confusion_matrix(y_test, y_pred)
    df_cm = pd.DataFrame(cm, index = [i for i in ['negative', 'neutral', 'positive']],
                         columns = [i for i in ['negative', 'neutral', 'positive']])
    plt.figure(figsize = (10, 7))
    sns.heatmap(df_cm, annot=True, fmt='d', cmap='Blues')
    plt.title('Confusion Matrix')
```

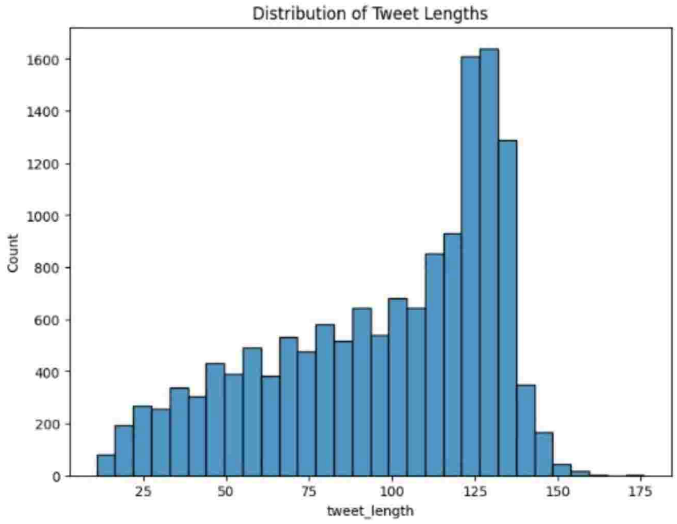
```
    plt. xlabel('Predicted')
    plt. ylabel('True')
    plt. show()
plot_confusion_matrix(y_test, y_pred)
```




```
# Creating  column 'tweet_length'
df['tweet_length'] = df['text']. apply(len)
# distribution of sentiments
plt. figure(figsize=(8, 6))
sns. countplot(x='airline_sentiment', data=df)
plt. title('Distribution of Sentiments')
plt. show()
```

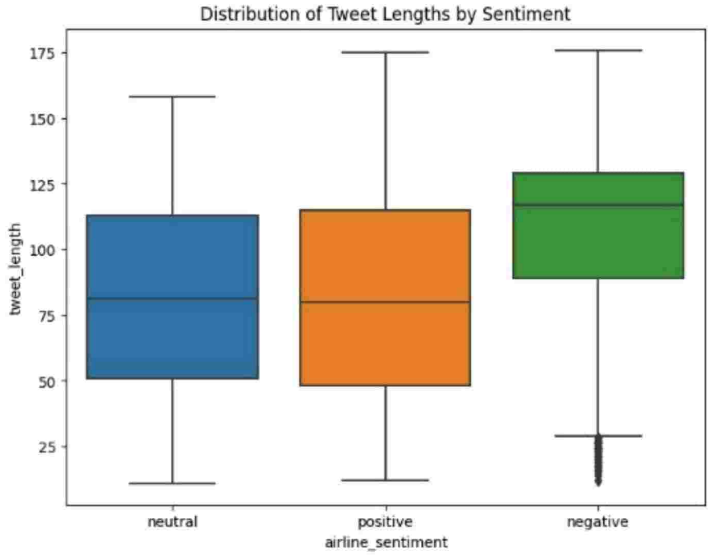



```
# Histogram of tweet lengths
plt. figure(figsize=(8, 6))
sns. histplot(df['tweet_length'], bins=30)
plt. title('Distribution of Tweet Lengths')
plt. show()
```




```
# Boxplot of tweet lengths
plt. figure(figsize=(8, 6))
```

```
sns. boxplot(x='airline_sentiment', y='tweet_length', data=df)
plt. title('Distribution of Tweet Lengths by Sentiment')
plt. show()
```



CONCLUSION:

- In the phase 2 conclusion, I summarize the key findings and insights from the advanced techniques. We will reiterate the impact of these techniques on the improving the accuracy and robustness of Sentiment analysis.

SENTIMENT ANALYSIS FOR MARKETING

PHASE 3

Project : Sentient Analysis For Marketing

DATA VISUALIZATION:

Data visualization in sentiment analysis is the combination of these two processes, where the results of sentiment analysis are displayed in a visual form that can facilitate analysis and decision making. For example, data visualization in sentiment analysis can help to

- Compare the overall sentiment (positive, negative, or neutral) of different groups of customers, products, topics, or time periods.
- Identify the most common words or phrases that are associated with positive or negative sentiment.
- Explore the distribution and variation of sentiment scores across different categories or dimensions.
- Track the changes and trends of sentiment over time

PROGRAM:

SENTIMENTAL ANALSIS FOR MARKETING

Importing Libraries:

```
import pandas as pd
import seaborn as sns
import re, nltk
nltk.download('punkt')
import matplotlib.pyplot as plt
from sklearn import model_selection, naive_bayes, svm
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.model_selection import GridSearchCV
from matplotlib import pyplot
import string
from nltk.corpus import stopwords
```

```
nltk.download('stopwords')
import numpy as np
from lime import lime_tabular
from tensorflow.keras.layers import Embedding
from tensorflow.keras.layers import LSTM, Bidirectional
from tensorflow.keras.layers import Dense, Dropout

import warnings
warnings.filterwarnings('ignore')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\ELCOT\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\ELCOT\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
#DATA LOADING

tweets_df =pd.read_csv('Tweets.csv')
tweets= tweets_df.copy()
tweets_df.head()
```

	tweet_id	airline_sentiment	airline_sentiment_confidence	\
0	570306133677760513	neutral	1.0000	
1	570301130888122368	positive	0.3486	
2	570301083672813571	neutral	0.6837	
3	570301031407624196	negative	1.0000	
4	570300817074462722	negative	1.0000	

	negativereason	negativereason_confidence	airline	\
0	NaN	NaN	Virgin America	
1	NaN	0.0000	Virgin America	
2	NaN	NaN	Virgin America	
3	Bad Flight	0.7033	Virgin America	
4	Can't Tell	1.0000	Virgin America	

	airline_sentiment_gold	name	negativereason_gold	retweet_count	\
0	NaN	cairdin	NaN	0	
1	NaN	jnardino	NaN	0	
2	NaN	yvonnalynn	NaN	0	
3	NaN	jnardino	NaN	0	
4	NaN	jnardino	NaN	0	

```
                                                text tweet_coord   \
0                 @VirginAmerica What @dhepburn said.         NaN
1  @VirginAmerica plus you've added commercials t...         NaN
2  @VirginAmerica I didn't today... Must mean I n...         NaN
3  @VirginAmerica it's really aggressive to blast...         NaN
4  @VirginAmerica and it's a really big bad thing...         NaN

              tweet_created tweet_location               user_timezone
0  2015-02-24 11:35:52 -0800            NaN  Eastern Time (US & Canada)
1  2015-02-24 11:15:59 -0800            NaN  Pacific Time (US & Canada)
2  2015-02-24 11:15:48 -0800      Lets Play  Central Time (US & Canada)
3  2015-02-24 11:15:36 -0800            NaN  Pacific Time (US & Canada)
4  2015-02-24 11:14:45 -0800            NaN  Pacific Time (US & Canada)
```

```
#Data columns

tweets_df.columns
```

```
Index(['tweet_id', 'airline_sentiment', 'airline_sentiment_confidence',
       'negativereason', 'negativereason_confidence', 'airline',
       'airline_sentiment_gold', 'name', 'negativereason_gold',
       'retweet_count', 'text', 'tweet_coord', 'tweet_created',
       'tweet_location', 'user_timezone'],
      dtype='object')
```

```
tweets_df['airline_sentiment'].unique()
```

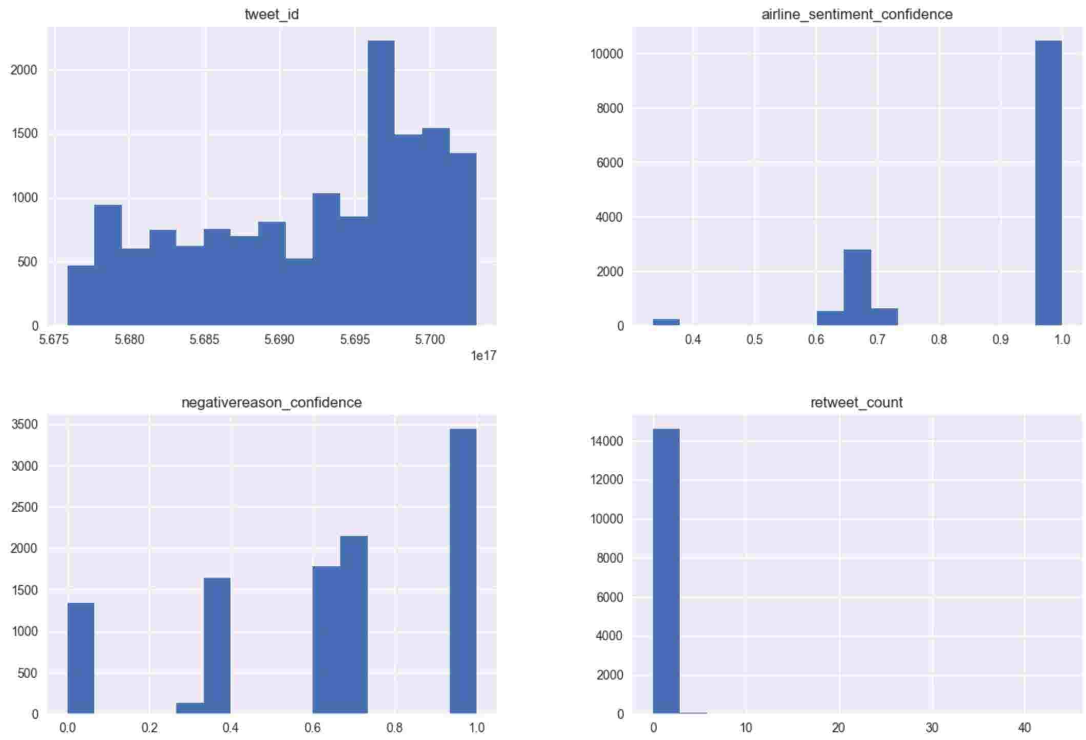
```
array(['neutral', 'positive', 'negative'], dtype=object)
```

```
tweets_df['airline_sentiment'].value_counts()
```

```
negative    9178
neutral     3099
positive    2363
Name: airline_sentiment, dtype: int64
```

```
#Data Visualization
plt.style.use("seaborn")
tweets_df.hist(figsize=(15, 10), bins=15)
```

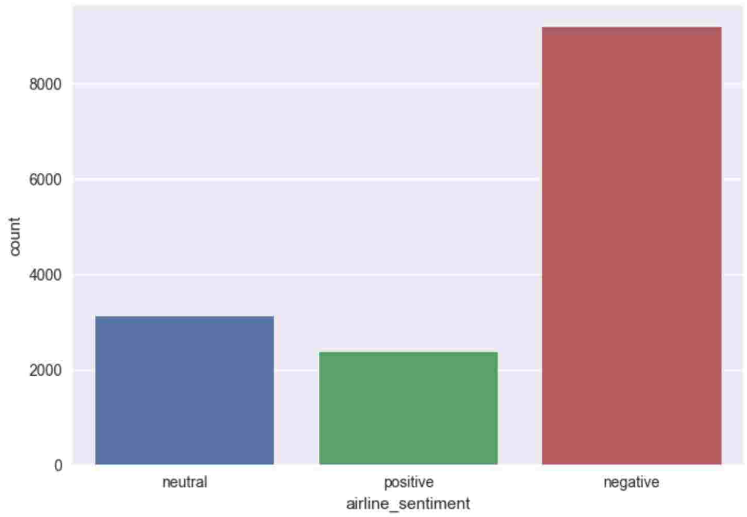
```
array([[<AxesSubplot: title={'center': 'tweet_id'}>,
        <AxesSubplot: title={'center': 'airline_sentiment_confidence'}>],
       [<AxesSubplot: title={'center': 'negativereason_confidence'}>,
        <AxesSubplot: title={'center': 'retweet_count'}>]], dtype=object)
```



#COUNT PLOT

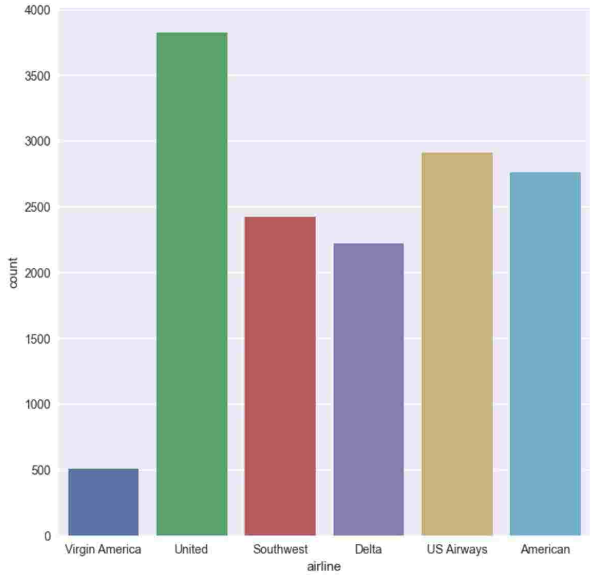
```
sns. countplot(x="airline_sentiment", data=tweets_df)
```

```
<AxesSubplot: xlabel='airline_sentiment', ylabel='count'>
```



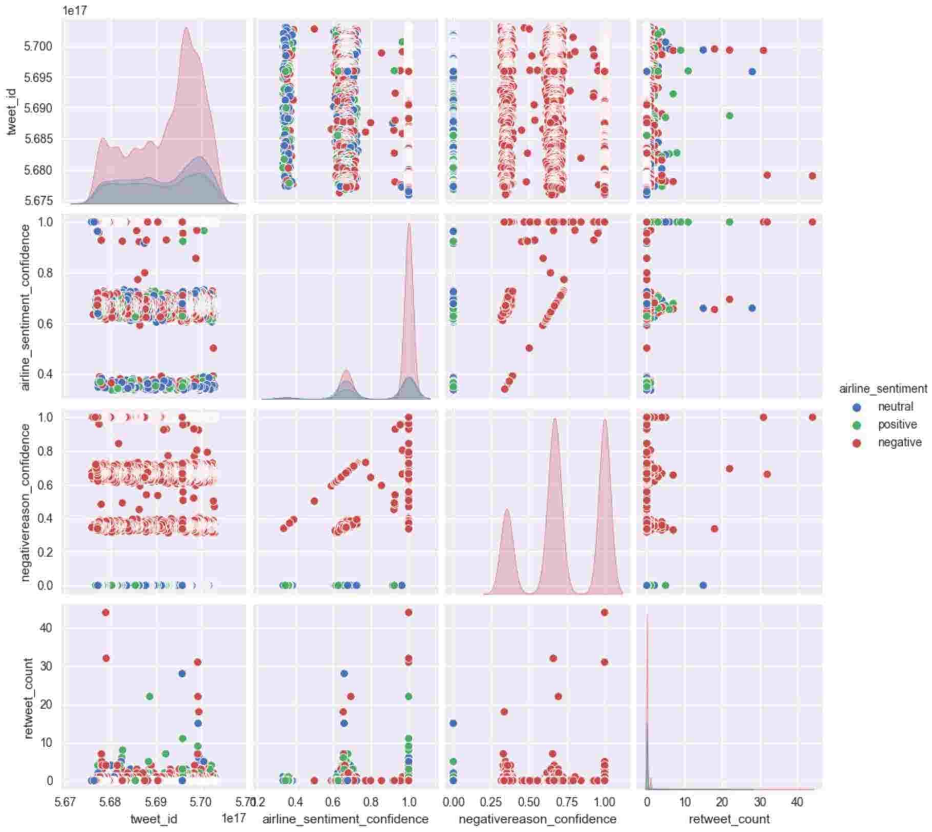

```
plt. figure(figsize=(8, 8))
sns. countplot(x="airline", data=tweets_df)
```

```
<AxesSubplot: xlabel='airline', ylabel='count'>
```




```
sns. pairplot(tweets_df, hue='airline_sentiment')
```

```
<seaborn. axisgrid. PairGrid at 0x211c88598d0>
```




```
print("Total number of tweets for each airline \n
", tweets_df. groupby('airline')['airline_sentiment']. count(). sort_values(ascending=False))
```

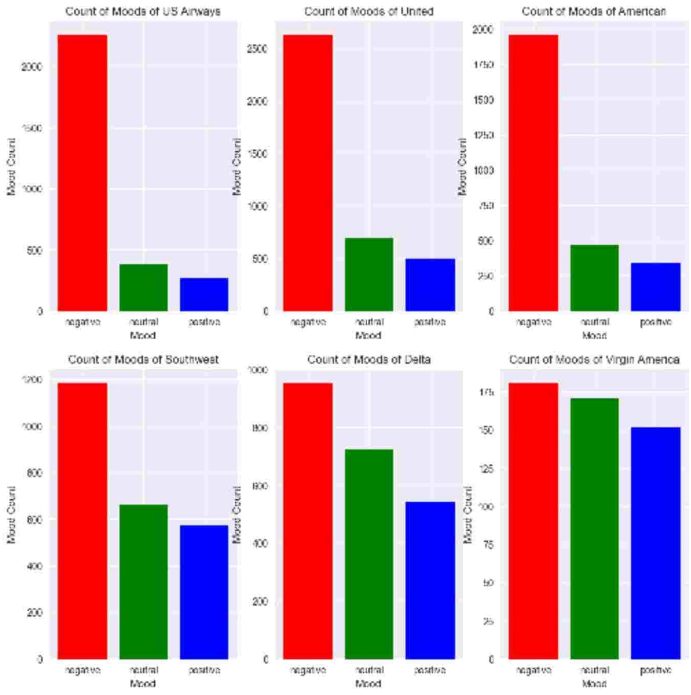
```
airlines= ['US Airways', 'United', 'American', 'Southwest', 'Delta', 'Virgin America']
plt. figure(1, figsize=(12,  12))
for i in airlines:
        indices= airlines. index(i)
        plt. subplot(2, 3, indices+1)
        new_df=tweets_df[tweets_df['airline']==i]
        count=new_df['airline_sentiment']. value_counts()
        Index = [1, 2, 3]
        plt. bar(Index, count,  color=['red',  'green',  'blue'])
        plt. xticks(Index, ['negative', 'neutral', 'positive'])
        plt. ylabel('Mood Count')
        plt. xlabel('Mood')
        plt. title('Count of Moods of '+i)
```

Out:

Total number of tweets for each airline

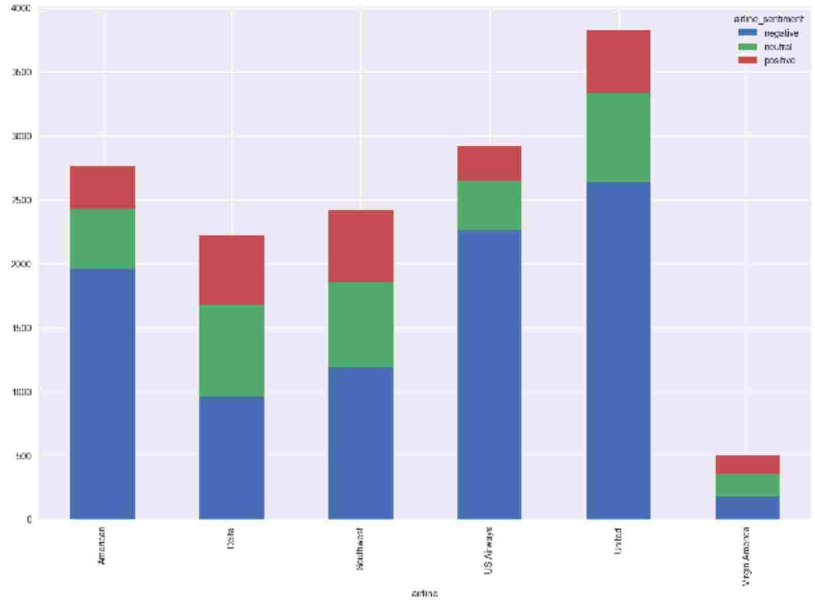
airline	
United	3822
US Airways	2913
American	2759
Southwest	2420
Delta	2222
Virgin America	504

Name: airline_sentiment, dtype: int64

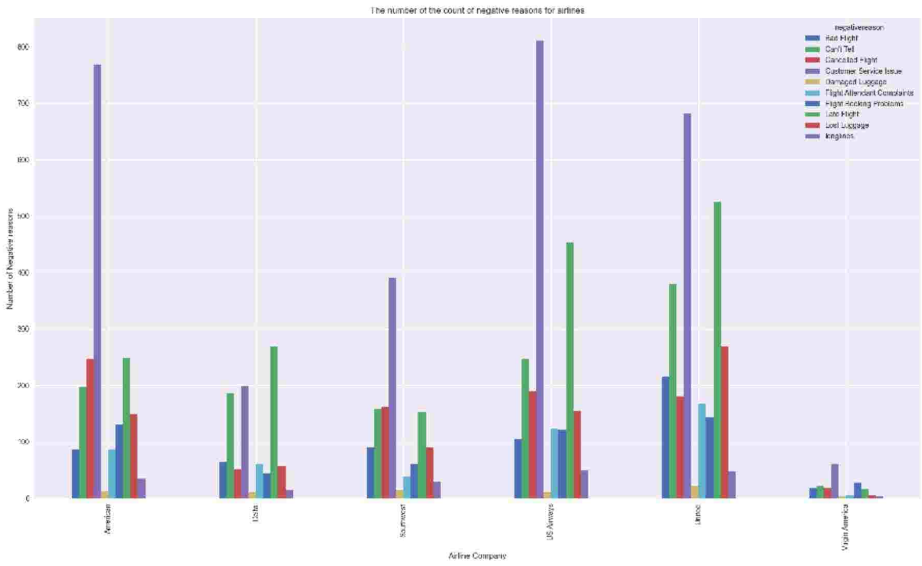


```
figure_2 = tweets_df.groupby(['airline', 'airline_sentiment']).size()
figure_2.unstack().plot(kind='bar', stacked=True, figsize=(15, 10))
```

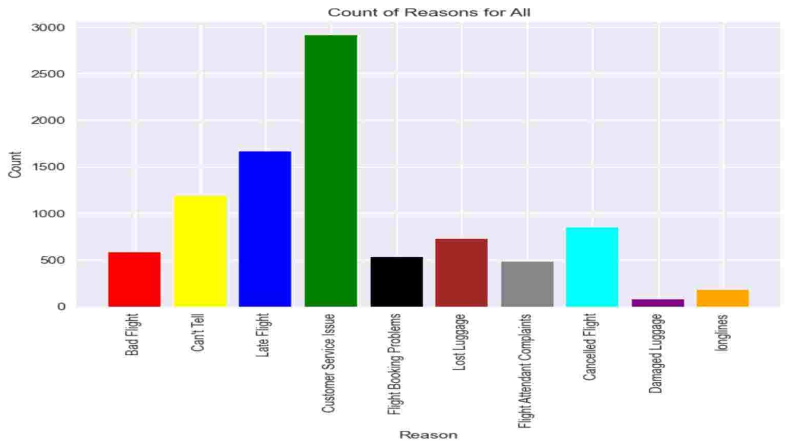
```
<AxesSubplot: xlabel='airline'>
```

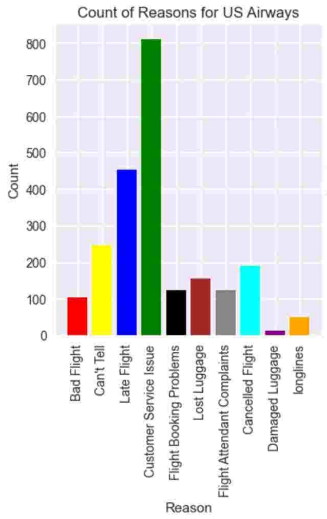


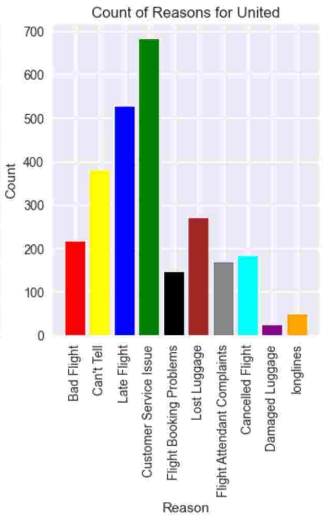

```
negative_reasons = tweets_df.groupby('airline')['negativereason'].value_counts(ascending=True)
negative_reasons.groupby(['airline', 'negativereason']).sum().unstack().plot(kind='bar', figsize=(22, 12))
plt.xlabel('Airline Company')
plt.ylabel('Number of Negative reasons')
plt.title("The number of the count of negative reasons for airlines")
plt.show()
```

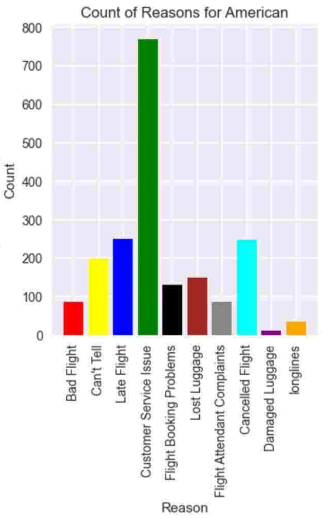


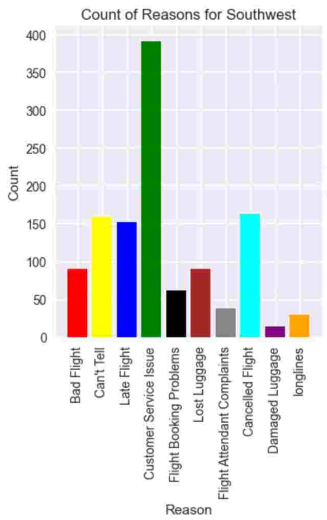
```
tweets_df['negativereason'].nunique()

NR_Count=dict(tweets_df['negativereason'].value_counts(sort=False))
def NR_Count(Airline):
    if Airline=='All':
        a=tweets_df
    else:
        a=tweets_df[tweets_df['airline']==Airline]
    count=dict(a['negativereason'].value_counts())
    Unique_reason=list(tweets_df['negativereason'].unique())
    Unique_reason=[x for x in Unique_reason if str(x) != 'nan']
    Reason_frame=pd.DataFrame({'Reasons':Unique_reason})
    Reason_frame['count']=Reason_frame['Reasons'].apply(lambda x: count[x])
    return Reason_frame
def plot_reason(Airline):

    a=NR_Count(Airline)
    count=a['count']
    Index = range(1,(len(a)+1))
    plt.bar(Index,count, color=['red','yellow','blue','green','black','brown','gray','cyan','purple','orange'])
    plt.xticks(Index,a['Reasons'],rotation=90)
    plt.ylabel('Count')
    plt.xlabel('Reason')
    plt.title('Count of Reasons for '+Airline)

plot_reason('All')
plt.figure(2,figsize=(13, 13))
for i in airlines:
    indices= airlines.index(i)
    plt.subplot(2,3,indices+1)
    plt.subplots_adjust(hspace=0.9)
    plot_reason(i)
```

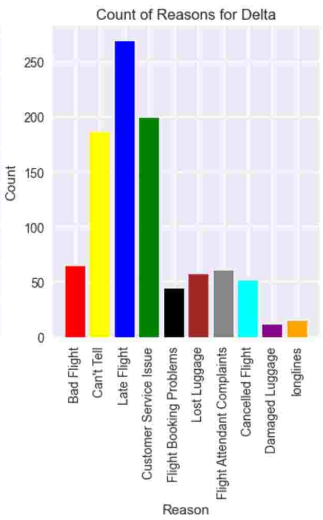


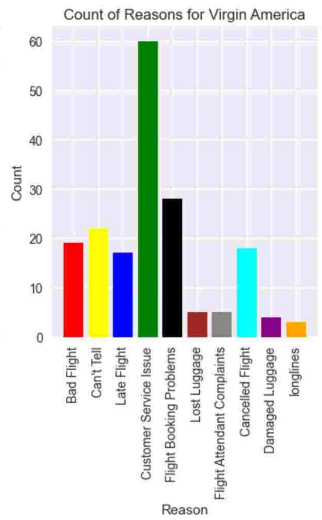










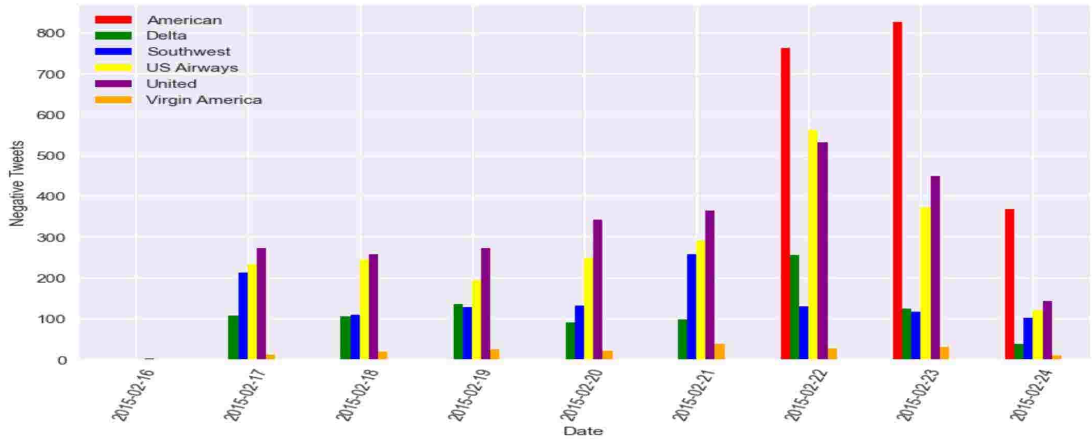



```
date = tweets_df.reset_index()
#convert the Date column to pandas datetime
```

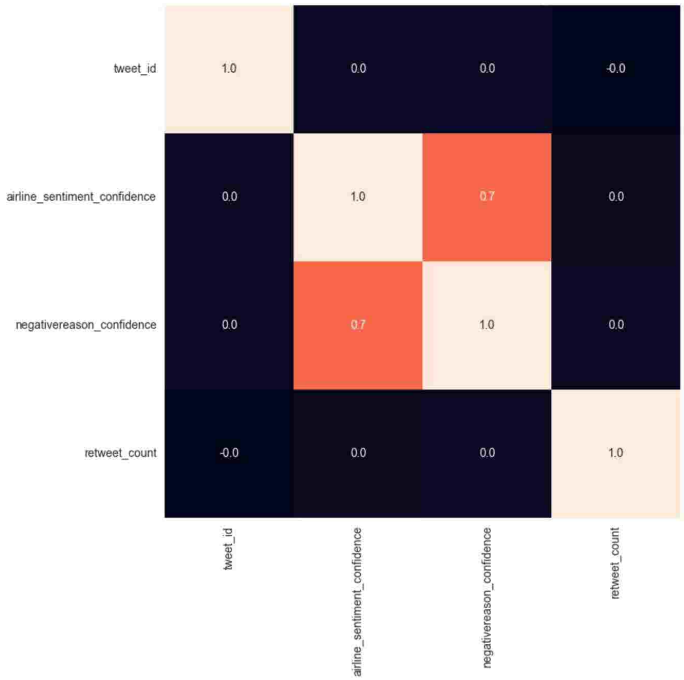
```
date.tweet_created = pd.to_datetime(date.tweet_created)
#Reduce the dates in the date column to only the date and no time stamp using the 'dt.date' method
date.tweet_created = date.tweet_created.dt.date
date.tweet_created.head()
df = date
day_df = df.groupby(['tweet_created', 'airline', 'airline_sentiment']).size()
day_df
```

```
tweet_created  airline         airline_sentiment
2015-02-16     Delta           negative            1
                               neutral             1
               United          negative            2
2015-02-17     Delta           negative          108
                               neutral            86
                                                  ...
2015-02-24     United          neutral            49
                               positive           25
               Virgin America  negative           10
                               neutral             6
                               positive           13
Length: 136, dtype: int64
```

```
day_df = day_df.loc(axis=0)[:, :, 'negative']

#groupby and plot data
ax2 = day_df.groupby(['tweet_created', 'airline']).sum().unstack().plot(kind = 'bar', color=['red', 'green',
'blue', 'yellow', 'purple', 'orange'], figsize = (10, 6), rot = 70)
labels = ['American', 'Delta', 'Southwest', 'US Airways', 'United', 'Virgin America']
ax2.legend(labels = labels)
ax2.set_xlabel('Date')
ax2.set_ylabel('Negative Tweets')
plt.show()
```

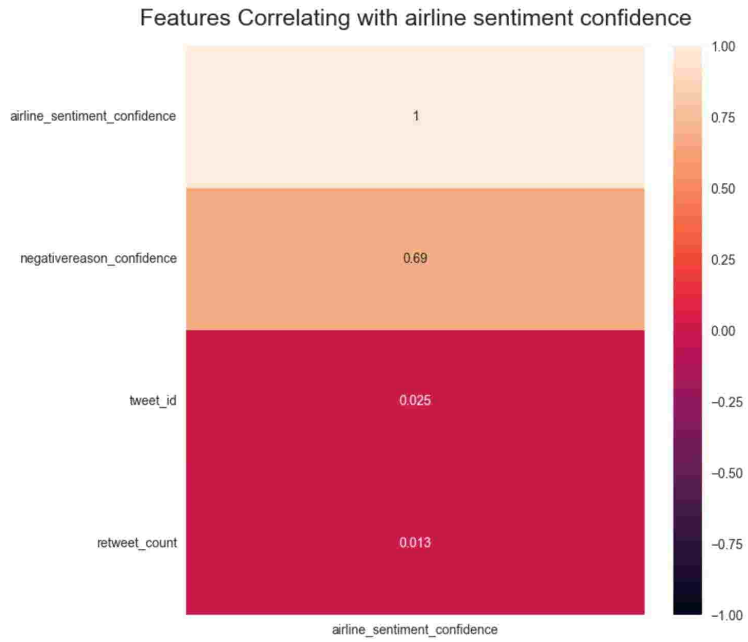


```
#Heatmap
plt. figure(figsize=(8, 8))
sns. heatmap(tweets_df. corr(), annot=True, cbar=False, fmt='. 1f')
plt. show()
```

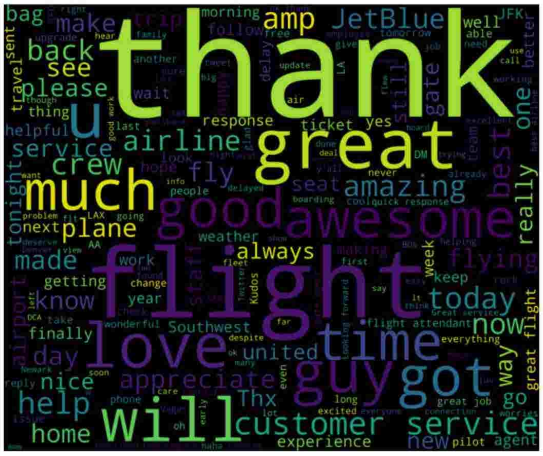



```
plt. figure(figsize=(8, 8))
heatmap =
sns. heatmap(tweets_df. corr()[['airline_sentiment_confidence']]. sort_values(by='airline_sentiment_confidence
', ascending=False), vmin=-1, vmax=1, annot=True)
heatmap. set_title('Features Correlating with airline sentiment confidence', fontdict={'fontsize':18}, pad=16)
```

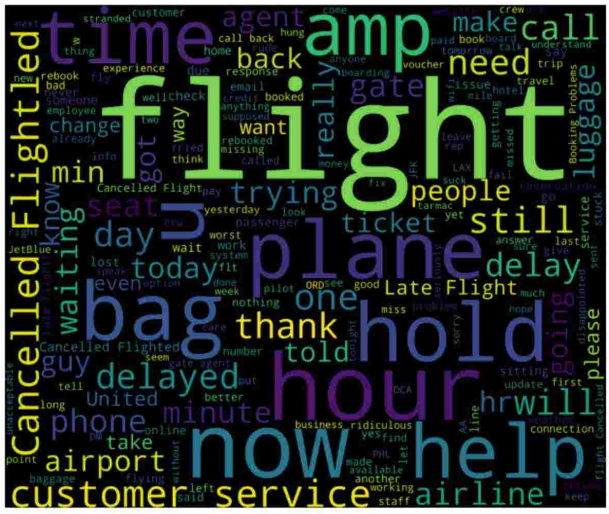
```
Text(0. 5, 1. 0, 'Features Correlating with airline sentiment confidence')
```




```
from wordcloud import WordCloud, STOPWORDS
new_df=tweets_df[tweets_df['airline_sentiment']=='positive']
words = ' '. join(new_df['text'])
cleaned_word = '' ''. join([word for word in words. split()
                                    if 'http' not in word
                                        and not word. startswith('@')
                                        and word ! = 'RT'
                                    ])
wordcloud = WordCloud(stopwords=STOPWORDS,
                    background_color='black',
width=3000, height=2500). generate(cleaned_word)
plt. figure(1, figsize=(8,  8))
plt. imshow(wordcloud)
plt. axis('off')
plt. show()
```

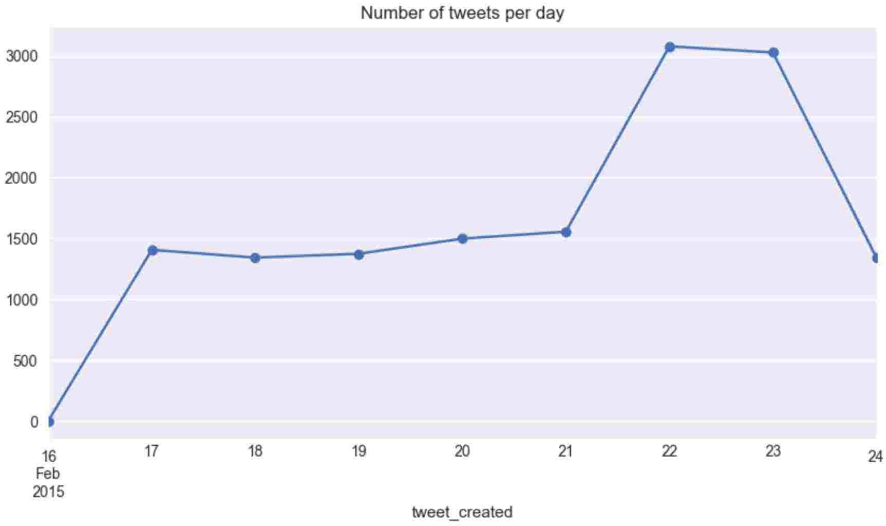



```
new_df=tweets_df[tweets_df['airline_sentiment']=='negative']
words = ' '. join(new_df['text'])
cleaned_word = " ". join([word for word in words. split()
                                    if 'http' not in word
                                        and not word. startswith('@')
                                        and word ! = 'RT'
                                    ])
wordcloud = WordCloud(stopwords=STOPWORDS,
                                background_color='black',
width=3000, height=2500). generate(cleaned_word)
plt. figure(1, figsize=(8, 8))
plt. imshow(wordcloud)
plt. axis('off')
plt. show()
```



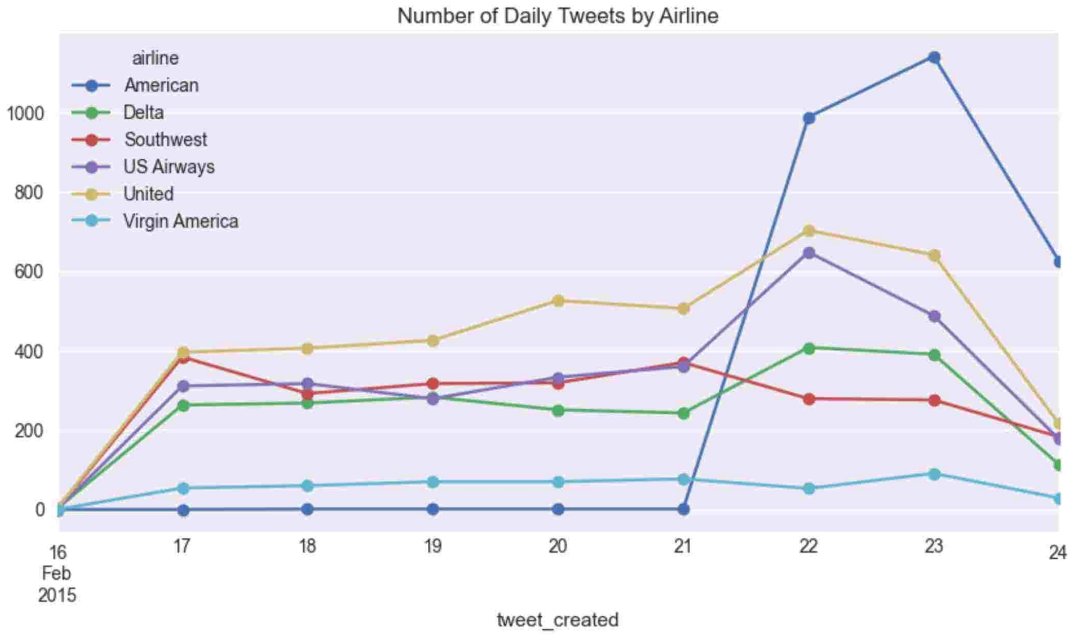
```
tweets['tweet_created'] = pd.to_datetime(tweets['tweet_created'])
tweets_time_index = tweets.copy()
tweets_time_index.set_index("tweet_created", inplace=True)

tweets_time_index.resample("D")['tweet_id'].count().plot(style="-o", figsize=(8, 5), title="Number of tweets per day")
```

```
<AxesSubplot: title={'center': 'Number of tweets per day'}, xlabel='tweet_created'>
```




```
tweets_time_index = tweets_time_index.pivot_table(index="tweet_created", columns="airline", values="tweet_id", aggfunc=np.count_nonzero, fill_value=0)
tweets_time_index.resample("D").sum().plot(style="-o", figsize=(10, 5), title="Number of Daily Tweets by Airline")
```

```
<AxesSubplot: title={'center': 'Number of Daily Tweets by Airline'}, xlabel='tweet_created'>
```



Conclusion:

In the quest to build a sentiment analysis for marketing, we have embarked on a critical journey that begins with loading and preprocessing the dataset. We have traversed through essential steps, starting with importing the necessary libraries to facilitate data manipulation and analysis.

Understanding the data's structure, characteristics, and any potential issues through exploratory data analysis (EDA) is essential for informed decision-making.

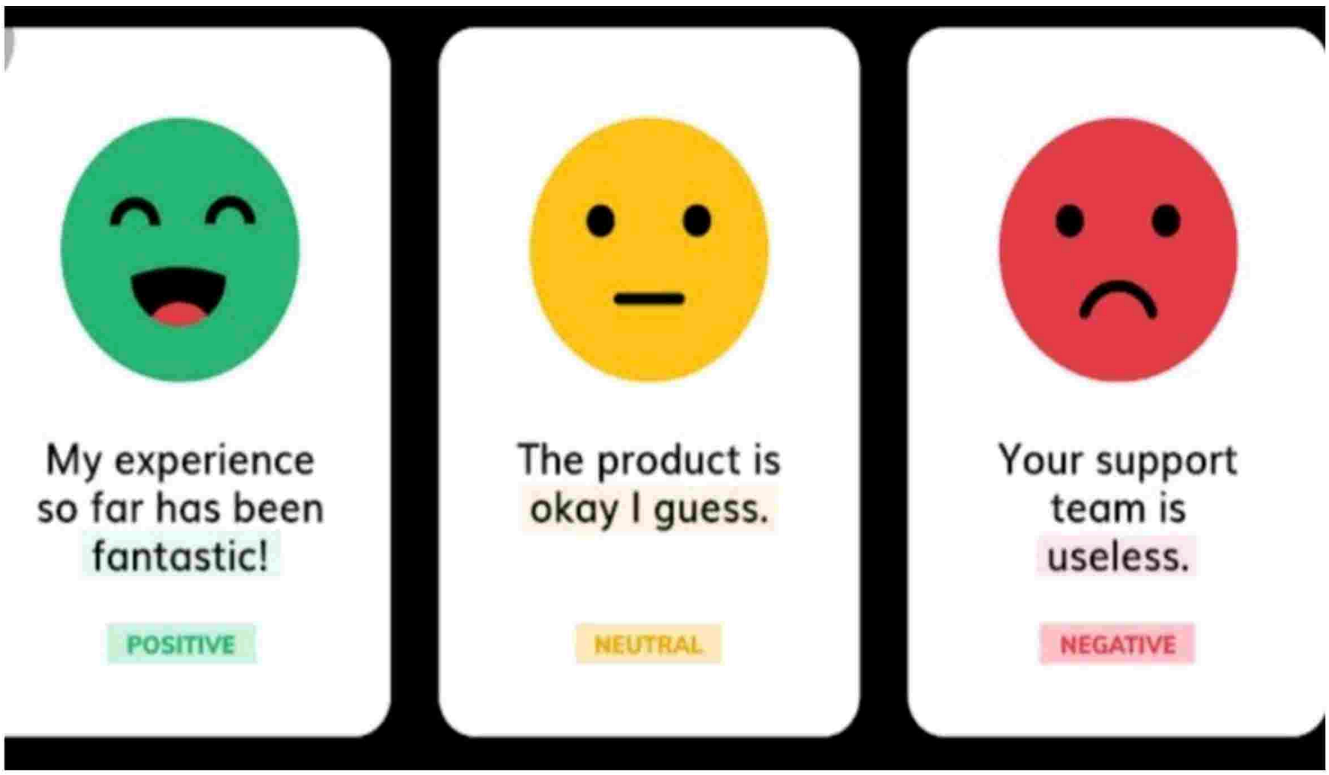
Data preprocessing emerged as a pivotal aspect of this process. It involves cleaning, transforming, and refining the dataset to ensure that it aligns with the requirements of machine learning algorithms.

SENTIMENT ANALYSIS IN MARKETING USING MACHINE LEARNING.

PROJECT TiTle : SENTIMENT ANALYSIS IN MARKETING.

Phase 4: Development Part-2.

Topic :Continue developing the sentiment analysis model by feature engineering,model training and evaluation.



SENTIMENT ANALYSIS IN MARKETING.

Introduction:

- With advancements in technology and fields like deep learning, sentiment analysis is becoming more and more common for companies that want to gauge their customers' sentiments.
- Today, businesses use natural language processing, statistical analysis, and text analysis to identify the sentiment and classify words into positive, negative, and neutral categories.
- The best companies understand the importance of understanding their customers' sentiments – what they are saying, what they mean and how they are saying. You can use xsentiment analysis to identify customer sentiment in comments, reviews, tweets, or social media platforms where people mention your brand.
- As sentiment analysis is the domain of understanding emotions using software, we have prepared a complete guide to understand 'what is sentiment analysis?', its tools, and different classifications and use cases.
- Sentiment analysis can be defined as analyzing the positive or negative sentiment of the customer in text. The contextual analysis of identifying information helps businesses understand their customers' social sentiment by monitoring online conversations.

The sentiment analysis process mainly focuses on polarity, i.e., positive, negative, or neutral. Apart from polarity, it also considers the feelings and emotions(happy, sad, angry, etc.), intentions(interested or not interested), or urgency(urgent or not urgent) of the text.

Overview of the process:

- ✓ Prepare the data.
- ✓ Perform feature selection.
- ✓ Train the model.
- ✓ Evaluate the model.
- ✓ Deploy the model.

OVERVIEW OF THE PROCESS:

The following is an overview of the process of building a house price prediction model by feature selection, model training, and evaluation:

1.Prepare the data: This includes cleaning the data, removing outliers, and handling missing values.

2.*Perform feature selection:* This can be done using a variety of methods, such as correlation analysis, information gain, and recursive feature elimination.

3.Train the model: There are many different machine learning algorithms that can be used for house price prediction. Some popular choices include linear regression, random forests, and gradient boosting machines.

4.Evaluate the model: This can be done by calculating the mean squared error (MSE) or the root mean squared error (RMSE) of the model's predictions on the held-out test set.

5.Deploy the model: Once the model has been evaluated and found to be performing well, it can be deployed to production so that it can be used to predict the house prices of new houses.

PROCEDURE:

1. **Identify the target variable**. This is the variable that you want to predict, such as house price.

2. **Explore the data**. This will help you to understand the relationships between the different features and the target variable. You can use data visualization and correlation analysis to identify feature that are highly correlated with the target variable.

3. **Remove redundant features**. If two features are highly correlated with each other, then you can remove one of the features, as they are likely to contain redundant information.

4. **Remove irrelevant features**. If a feature is not correlated with the target variable, then you can remove it, as it is unlikely to be useful for prediction.

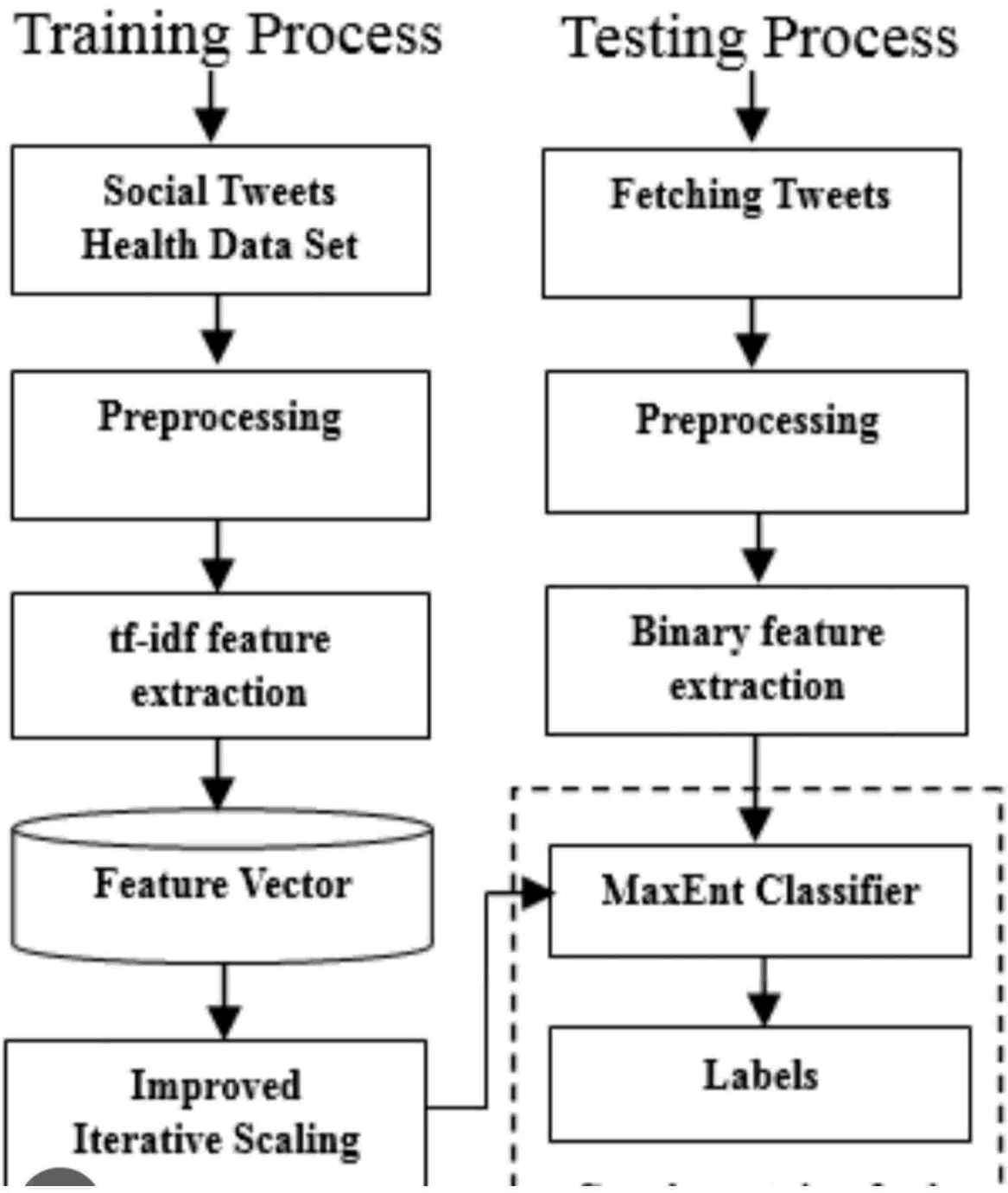
FEATURE SELECTION:

A list of possible features and their variations was produced by analyzing available literature. The following features were selected for testing, since they proved useful in previous research The different types of features used were unigram bag-of-words,bigram bag-of-words, special characters counts, and other

textual properties. The motivation of using these is presented in this section.

Unigram bag of words:

The bag-of-words feature is a word counter since it is implemented in the unigram case. Each feature in the input vector represents the count of a certain word in the input sentence. Stemming and lower casing was used to decrease the number of unique words, and join different inflections of the same primitive together. However, the information loss of removing the end of words must be taken into consideration.


Training Process
Testing Process
Social Tweets Health Data Set
Fetching Tweets
Preprocessing
Preprocessing
tf-idf feature extraction
Binary feature extraction
Feature Vector
MaxEnt Classifier
Improved Iterative Scaling
Labels

Model training:

```
>>> import pandas as pd
>>> df =
pd.read_csv('/datasets/sentiment/amazon_cells_labelled.txt',
names=['review', 'sentiment'], sep='\t')
>>> df.head()0 So there is no way for me to plug it in here i… 0
1 Good case, Excellent value. 1
2 Great for the jawbone. 1
3 Tied to charger for conversations lasting more… 0
4 The mic is great.
```

Splitting the Data Set into a Training Set and aTest Set:

```
>>> from sklearn.model_selection import train_test_split
>>> reviews = df['review'].values
>>> labels = df['sentiment'].values
>>> reviews_train, reviews_test, y_train, y_test =
train_test_split(reviews, labels, test_size=0.2, random_state=1000)
```

Transforming Text into Numerical Feature Vectors:

```
>>> from sklearn.feature_extraction.text import CountVectorizer
>>> vectorizer = CountVectorizer(tokenizer = spacy_tokenizer,
ngram_range=(1,1))
>>> #By default, the vectorizer might be created as follows:
>>> #vectorizer = CountVectorizer()
>>> vectorizer.fit(reviews_train)
```

Training the Model:

```
>>> from sklearn.linear_model import LogisticRegression
>>> classifier = LogisticRegression()
>>> classifier.fit(X_train, y_train)
```

Evaluation of the Model:

```
>>> accuracy = classifier.score(X_test, y_test)
>>> print("Accuracy:", accuracy)Accuracy: 0.785
```

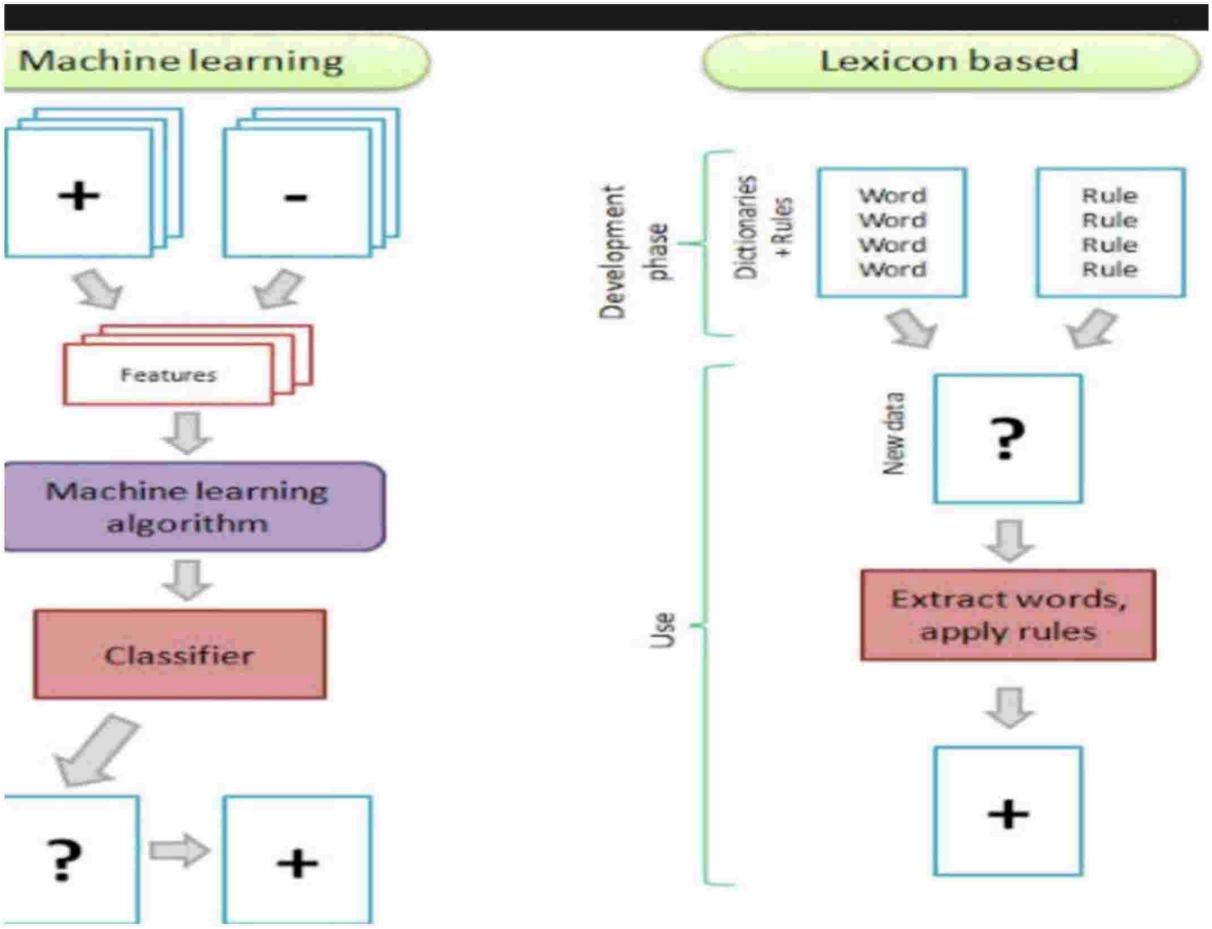
Predictions on New Data:

```
>>> new_reviews = ['Old version of python useless', 'Very good
effort, but not five stars', 'Clear and concise']
>>> X_new = vectorizer.transform(new_reviews)
>>> classifier.predict(X_new)array([0, 1, 1])
```

MODEL EVALUATION :

Consumer feedback is highly valuable in business to assess their performance and is also beneficial to customers as it gives them an idea of what to expect from new products. In this research, the aim is to evaluate different deep learning approaches to accurately predict the opinion of customers based on mobile phone reviews . The prediction is based on analysing these reviews and categorizing them as positive, negative, or neutral. Different deep learning algorithms have been implemented and evaluated such as simple RNN with its four variants, namely, Long Short-Term Memory Networks (LRNN), Group Long Short-Term Memory Networks (GLRNN), gated recurrent unit (GRNN), and update recurrent unit (UGRNN). All evaluated algorithms are combined with word embedding as feature extraction approach for sentiment analysis including Glove, word2vec, and FastText by Skip-grams. The five different algorithms with the three feature extraction methods are evaluated based on accuracy, recall, precision, and F1-score for both balanced and unbalanced datasets. For the unbalanced dataset, it was found that the GLRNN algorithms with FastText feature extraction scored the highest accuracy of 93.75%. This result achieved the highest accuracy on this dataset when compared with other methods mentioned in the literature. For the balanced dataset, the highest achieved accuracy was 88.39% by the LRNN algorithm..

FEATURE TO PERFORM MODEL TRAINING:



Feature engineering is the process of transforming raw data into features that are more informative and predictive for machine learning models. By using a variety of feature engineering techniques, you can create a set of features that will help your model to predicts more accurately.Use

cross-validation.Cross-validation is a technique for evaluating the performance of a

machine learning model on unseen data. It is important to use cross- validation to evaluate the performance of your model during the evaluation process. This will help you to avoid overfitting and to ensure that the model will generalize well to new data.

- Use cross validation.
- Use ensemble methods.
- Compare model to a damper.
- Use a hold out test set.
- Analysis model for prediction.

  Use cross-validat:ion.

  Cross-validation is a technique for evaluating the performance of a machine learning model on unseen data. It is important to use cross-validation to

evaluate the performance of your model during the trainingprocess. This will help you to avoid overfitting and to ensure that your model will generalize well to new data.

Use ensemble methods:

Ensemble methods are machine learning methods that combine the predictions of multiple models to produce a more accurate prediction.Ensemble methods can often achieve better performance than individual machine learning models.

Use cross-validation.

Cross-validation is a technique for evaluating the performance of a machine learning model on unseen data. It is important to use cross- validation to evaluate the performance of your model during the

evaluation process. This will help you to avoid overfitting and to ensure that the model will generalize well to new data.

Use a **holdout test set**.

A holdout test set is a set of data that is not used to train or evaluate the model during the training process. This data is used toevaluate the performance of the model on unseen data after the training process is complete.

Compare the model to a baseline.

A baseline is a simple model that is used to compare the performance of your model to. For example, you could use the means a baseline.

Analyze the model's predictions.

Once you have evaluated the performance of the model, you can analyze the model's predictions to

identify any patterns or biases. Thiswill help you to understand the strengths and weaknesses of the model and to improve it

CONCLUSION:

Sentiment analysis deals with the classification of texts based on the sentiments they contain. This article focuses on a typical sentiment analysis model consisting of three
core steps, namely data preparation, review analysis and sentiment classification, and
describes representative techniques involved in those steps.
Sentiment analysis is an emerging research area in text mining and computational

linguistics, and has attracted considerable research attention in the past few years.
Future research shall explore sophisticated methods for opinion and product feature
extraction, as well as new classification models that can address the ordered labels
property in rating inference. Applications that utilize results from sentiment analysis
is also expected to emerge in the near future.