

IBM PHASE -3

PROJECT SUBMISSION

PROJECT: SENTIMENT ANALYSIS FOR MARKETING

PHASE 3 : DEVELOPMENT PART

In this part you will begin building your project by loading and preprocessing the dataset. Start building the sentiment analysis solution by loading dataset and preprocessing the data.

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

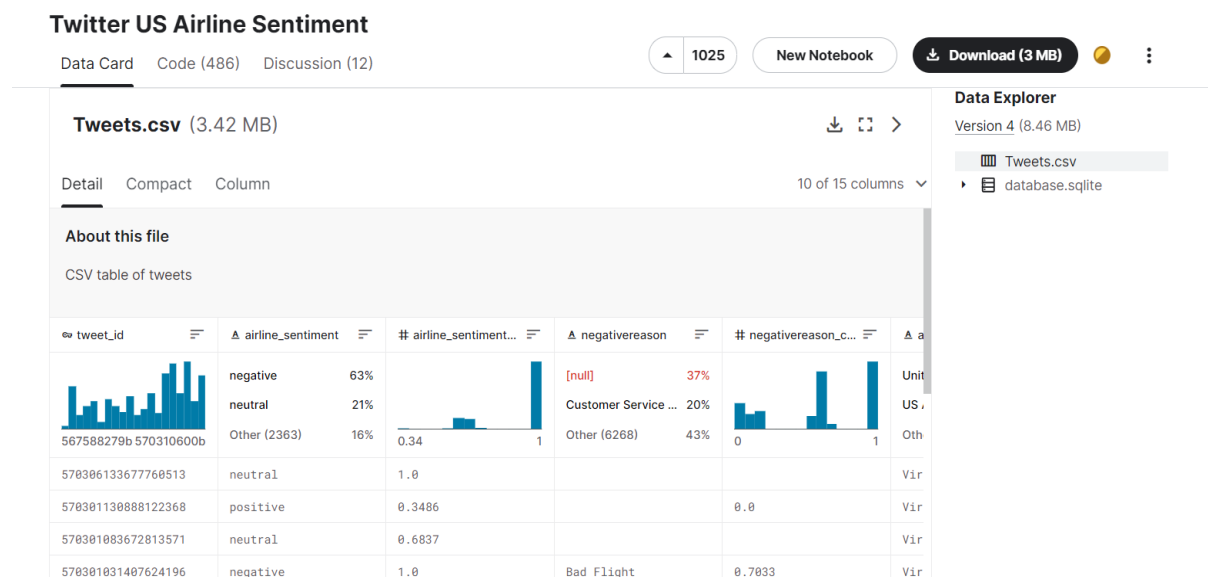
INTRODUCTION:

- In this development part the project have to built by loading and preprocessing the **Twitter US Airline Sentiment** dataset. In this phase the dataset will be load and preprocessed and it is ready for feature engineering, model training, and evaluation.

DATASET :

- The dataset is given in the question. The dataset is imported from Kaggle by the given link.
- The provided link

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



IMPORT THE NECESSARY LIBRARIES:

The necessary libraries like numpy ,pandas , seaborn , matplotlib are imported .

PROGRAM:

```
import numpy as np
```

```
import pandas as pd
```

```
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier, VotingClassifier
```

```
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
```

```
import matplotlib.pyplot as plt
```

```
import seaborn as sns
```

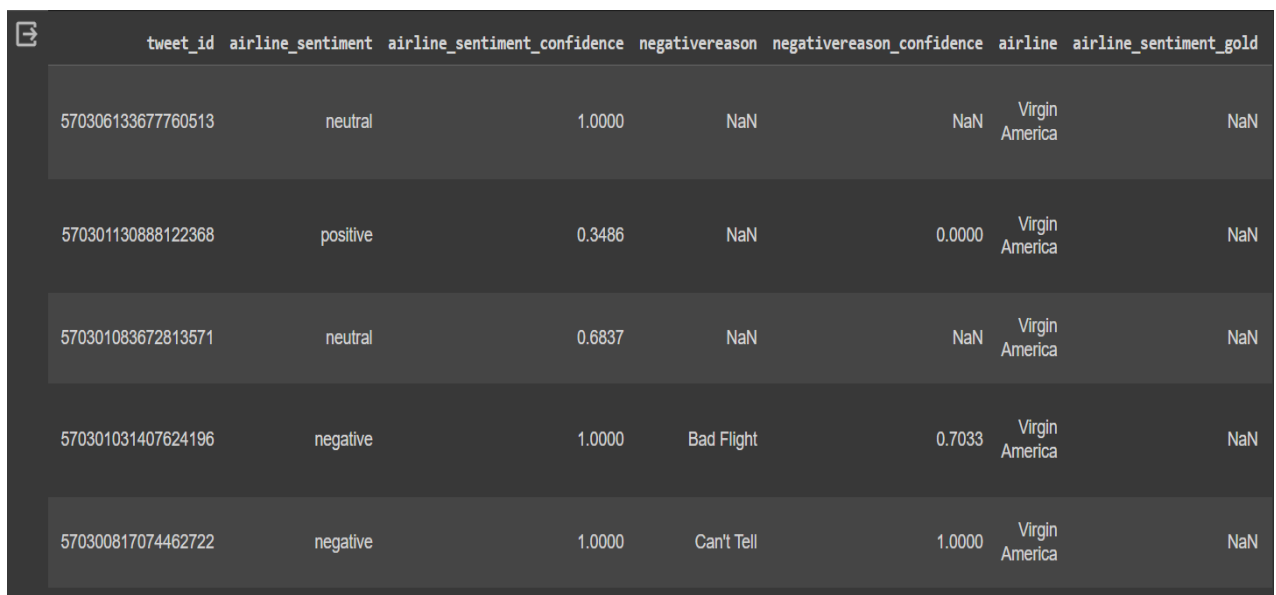
LOAD THE DATASET:

Program :

```
df = pd.read_csv('/content/Tweets.csv')
```

```
print(df.head())
```

Output:



The screenshot shows a Jupyter Notebook interface with a dark theme. The output of the `df.head()` command is displayed as a table with 7 columns: `tweet_id`, `airline_sentiment`, `airline_sentiment_confidence`, `negativereason`, `negativereason_confidence`, `airline`, and `airline_sentiment_gold`. The table contains 5 rows of data, representing the first 5 rows of the dataset. The `airline` column consistently shows 'Virgin America' for all rows. The `airline_sentiment_gold` column shows 'NaN' for all rows.

tweet_id	airline_sentiment	airline_sentiment_confidence	negativereason	negativereason_confidence	airline	airline_sentiment_gold
570306133677760513	neutral	1.0000	NaN	NaN	Virgin America	NaN
570301130888122368	positive	0.3486	NaN	0.0000	Virgin America	NaN
570301083672813571	neutral	0.6837	NaN	NaN	Virgin America	NaN
570301031407624196	negative	1.0000	Bad Flight	0.7033	Virgin America	NaN
570300817074462722	negative	1.0000	Can't Tell	1.0000	Virgin America	NaN

PREPROCESSING:

- Preprocessing the Twitter US Airline Sentiment dataset is an essential step before using it for any machine learning task. This dataset contains tweets about different US airlines, along with their sentiments (positive, negative, or neutral)

The step by step preprocessing is

1.HANDLE MISSING VALUES:

- Handling missing values is an essential step in data preprocessing. Missing data can cause issues during analysis or modeling, so it's important to address them appropriately. Here are some common strategies for handling missing values:

Identify Missing Values:

- Begin by identifying which columns or features in your dataset contain missing values. You can use functions like `isna()` or `isnull()` to check for missing values.

CODE:

```
df.isna().sum()
```

Remove Rows with Missing Values:

- If the number of missing values is relatively small and removing them won't significantly impact the analysis, you can choose to drop the rows containing missing values.

CODE:

```
df_clean = df.dropna()
```

2.TEXT CLEANING:

- Twitter data often contains noisy elements such as special characters, URLs, hashtags, mentions, and emojis. It's crucial to remove or handle these to focus on the actual text content. Use regular expressions or specialized libraries for this purpose.

CODE:

```
import re
```

```
def clean_text(text):
```

```
    # Remove URLs
```

```
    text = re.sub(r'http\S+', "", text)
```

```
    # Remove mentions
```

```
    text = re.sub(r'@\w+', "", text)
```

```
# Remove hashtags
```

```
text = re.sub(r'#\w+', "", text)
```

```
# Remove special characters and numbers
```

```
text = re.sub(r'^a-zA-Z\s]', "", text)
```

```
# Convert to lowercase
```

```
text = text.lower()
```

```
return text
```

3.TOKENIZATION:

- Tokenization involves breaking down the text into individual words or tokens. This can be done using simple space-based splitting or more advanced techniques like spaCy or NLTK.

CODE:

```
from nltk.tokenize import word_tokenize
```

```
def tokenize(text):
```

```
    return word_tokenize(text)
```

4.STOPWORD REMOVAL:

- Stopwords are common words (e.g., "and", "the") that don't contribute much to the meaning of a sentence. Removing them can reduce noise in the data.

CODE:

```
from nltk.corpus import stopwords

def remove_stopwords(tokens):
    stop_words = set(stopwords.words('english'))
    return [word for word in tokens if word not in stop_words]
```

5.LEMMATIZATION OR STEMMING:

- Reducing words to their base or root form can help in further reducing noise and standardizing the text.

CODE:

```
from nltk.stem import WordNetLemmatizer

def lemmatize(tokens):
    lemmatizer = WordNetLemmatizer()
    return [lemmatizer.lemmatize(word) for word in tokens]
```

6.JOINING TOKENS:

- After all the preprocessing steps, you may want to join the tokens back into a single string.

CODE:

```
def join_tokens(tokens):
    return ' '.join(tokens)
```

7.APPLYING THE PREPROCESSING PIPELINE:

- Apply the defined functions to the text data.

CODE:

```
df['text'] = df['text'].apply(clean_text)
df['tokens'] = df['text'].apply(tokenize)
df['tokens'] = df['tokens'].apply(remove_stopwords)
df['tokens'] = df['tokens'].apply(lemmatize)
df['clean_text'] = df['tokens'].apply(join_tokens)
```

8.SAVED THE PREPROCESSED DATA:

- Once the preprocessing is complete, save the preprocessed dataset for future use.

CODE:

```
df.to_csv('preprocessed_data.csv', index=False)
```

9.PREPROCESSING THE DATASET:

- The given Twitter US Airline Sentiment dataset is preprocessed using the above mentioned steps .It is essential to preprocess the dataset before using it for machine learning task.

PROGRAM:

```
import pandas as pd

import re

from nltk.tokenize import word_tokenize

from nltk.corpus import stopwords

from nltk.stem import WordNetLemmatizer

# Load the dataset

df = pd.read_csv('Tweets.csv')

# Define functions for preprocessing

def clean_text(text):

    text = re.sub(r'http\S+', '', text)

    text = re.sub(r'@\w+', '', text)

    text = re.sub(r'#\w+', '', text)

    text = re.sub(r'^a-zA-Z\s', '', text)

    text = text.lower()

    return text

def tokenize(text):
```

```
    return word_tokenize(text)

def remove_stopwords(tokens):

    stop_words = set(stopwords.words('english'))

    return [word for word in tokens if word not in stop_words]

def lemmatize(tokens):

    lemmatizer = WordNetLemmatizer()

    return [lemmatizer.lemmatize(word) for word in tokens]

def join_tokens(tokens):

    return ' '.join(tokens)

# Apply the preprocessing steps

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

df['tokens'] = df['clean_text'].apply(tokenize)

df['tokens'] = df['tokens'].apply(remove_stopwords)

df['tokens'] = df['tokens'].apply(lemmatize)

df['clean_text'] = df['tokens'].apply(join_tokens)

# Save the preprocessed data

df.to_csv('preprocessed_tweets.csv', index=False)
```

OUTPUT:

```
7s [nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data] Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data] Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data] Package wordnet is already up-to-date!

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
```

CONCLUSION:

- By completing these preprocessing steps, the original Twitter US Airline Sentiment dataset is transformed into a format that is more amenable for sentiment analysis. The cleaned and tokenized data is now ready for feature engineering, model training, and evaluation.