**DEVELOPMENT PART - 1 DOCUMENT**

To begin building the project for sentiment analysis on customer feedback, let’s elaborate on the process of loading and preprocessing the dataset

**DATA COLLECTION**

**DATA SOURCES -**Identify and gather relevant datasets from sources such as online review platforms, social media, or customer feedback forms. Ensure the data is in a format that can be easily imported into your chosen programming environment.

**DATA VOLUME -**Make sure the dataset is sufficiently large to provide meaningful insights and is representative of customer sentiments. Larger datasets often yield more accurate results.

**DATA QUALlITY -**Verify the data quality by checking for missing values, duplicates, and inconsistencies. This may involve data cleaning to handle issues like incorrect formatting or irrelevant information.

**DATA PREPROCESSING**

**TEXT CLEANING-**Remove any special characters, punctuation, and irrelevant information that may not contribute to sentiment analysis. Common cleaning steps include removing HTML tags, emojis, and URLs.

**TOKENZIATION** -Split the text into individual words or tokens for further analysis. Tokenization can be performed using libraries such as NLTK or spaCy.

**STOPWORD REMOVAL**-Eliminate common stopwords (e.g., “and,” “the,” “is”) to reduce noise in the data. You can use predefined lists of stopwords or create your own.

**NORMALISATION -** Normalize text by converting it to lowercase to ensure consistency in text data. This step ensures that “good” and “Good” are treated as the same word.

**DATASET LOADING**

Once we have collected and preprocessed the data, we can load it into the preferred programming environment. We can use libraries like Pandas in Python to load and manipulate the dataset efficiently. Ensure that we have all the necessary dependencies and tools installed.

import numpy as np

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import LabelEncoder

from tensorflow.keras.preprocessing.text import Tokenizer

from tensorflow.keras.preprocessing.sequence import pad\_sequences

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Embedding, LSTM, Dense, Dropout

import optuna

from keras.callbacks import EarlyStopping

from sklearn.preprocessing import LabelEncoder

**LOAD IN DATASET**

Df = pd.read\_csv(‘/kaggle/input/twitter-airline-sentiment/Tweets.csv’)

**EXPLORE DATASET**

from IPython.display import Markdown, display

display(Markdown('#### Preview of the first 7 rows in the DataFrame'))

display(df.head(7))

display(Markdown('#### Shape of the DataFrame'))

display(df.shape)

display(Markdown('#### Summary of the DataFrame'))

display(df.info())

display(Markdown('#### Descriptive statistics of numerical columns in the DataFrame'))

display(df.describe())

display(Markdown('#### Number of null values in each column'))

display(df.isnull().sum())

display(Markdown('#### Correlation between numerical columns in the DataFrame'))

display(df.corr())

**SHAPE OF THE DATAFRAME**

(14640, 15)

**SUMMARY OF THE DATAFRAME**

<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

None

**NUMBER OF NULL VALUES IN EACH COLUMN**

tweet\_id 0

airline\_sentiment 0

airline\_sentiment\_confidence 0

negativereason 5462

negativereason\_confidence 4118

airline 0

airline\_sentiment\_gold 14600

name 0

negativereason\_gold 14608

retweet\_count 0

text 0

tweet\_coord 13621

tweet\_created 0

tweet\_location 4733

user\_timezone 4820

dtype: int64

**PRE-PROCESSING THE TEXT**

def object\_columns\_value\_counts(df, max\_length=200):

*# Create an empty dictionary to store the value counts for each column*

value\_counts = {}

*# Iterate through the columns of the input dataframe*

for col **in** df.columns:

*# Check if the column has an 'object' datatype*

if df[col].dtype == 'object':

*# Check if the maximum string length in the column is less than or equal to max\_length*

if df[col].str.len().max() <= max\_length:

*# Get the value counts for the column as a dataframe*

counts\_df = df[col].value\_counts().reset\_index()

counts\_df.columns = ['Value', 'Count']

*# Store the counts dataframe in the dictionary*

value\_counts[col] = counts\_df

return value\_counts

result = object\_columns\_value\_counts(df, max\_length=200)

*# Print the value counts for each 'object' datatype column as a dataframe*

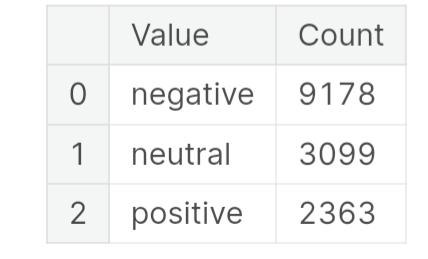
for column, counts\_df **in** result.items():

display(Markdown(f"#### Value counts for column '**{**column**}**':"))

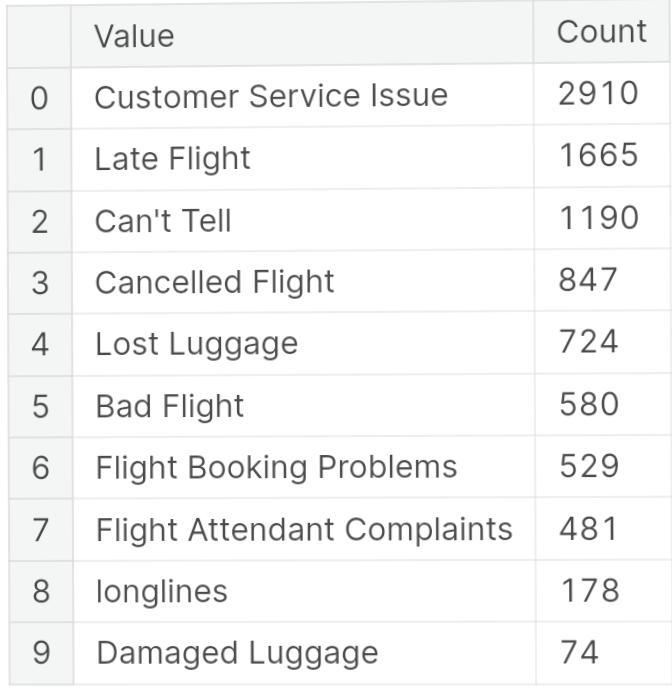
display(counts\_df)

print("**\n**")

#### **Value counts for column 'airline\_sentiment'**

****

**Value counts for column ‘negativereason’**

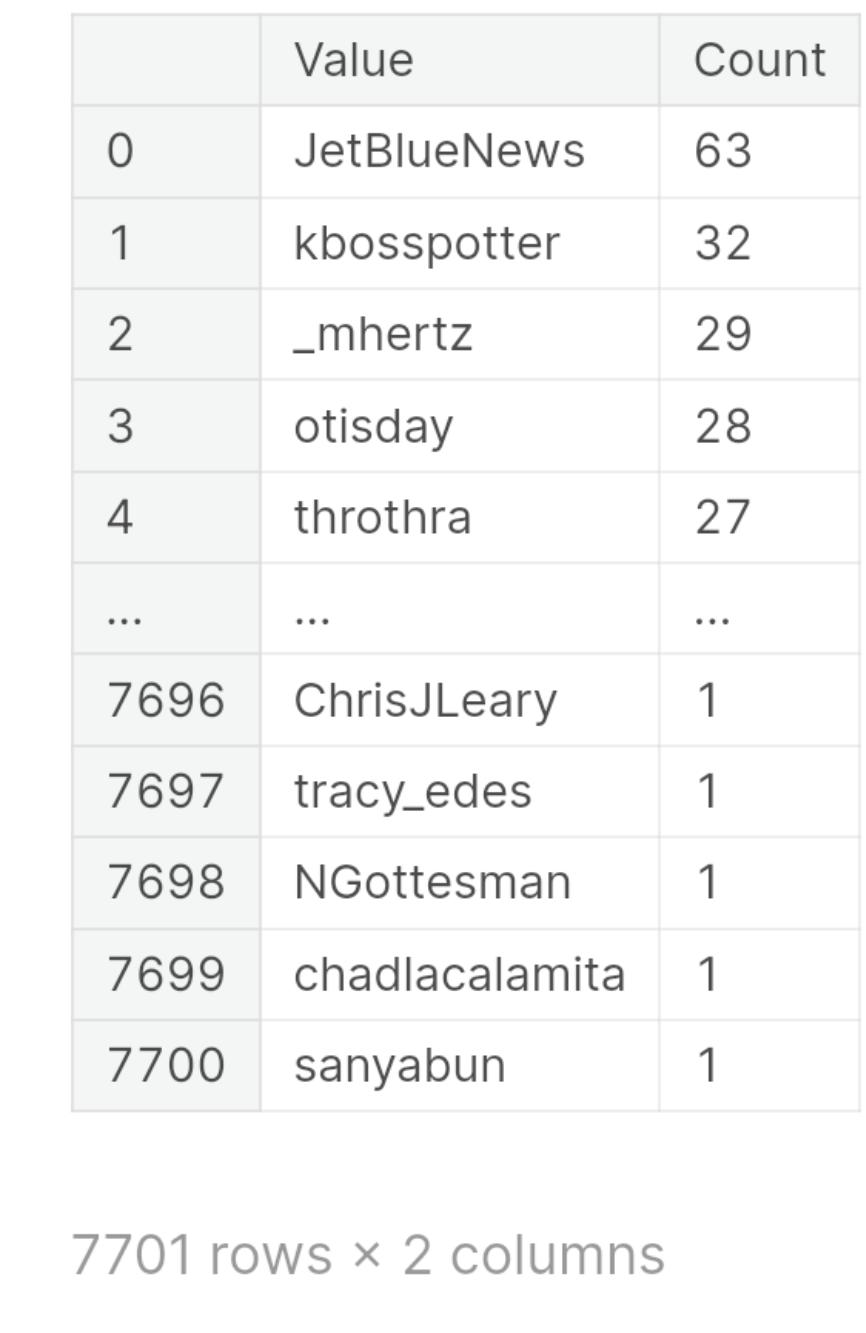
****

#### **Value counts for column 'airline':**



#### **Value counts for column 'airline\_sentiment\_gold'**

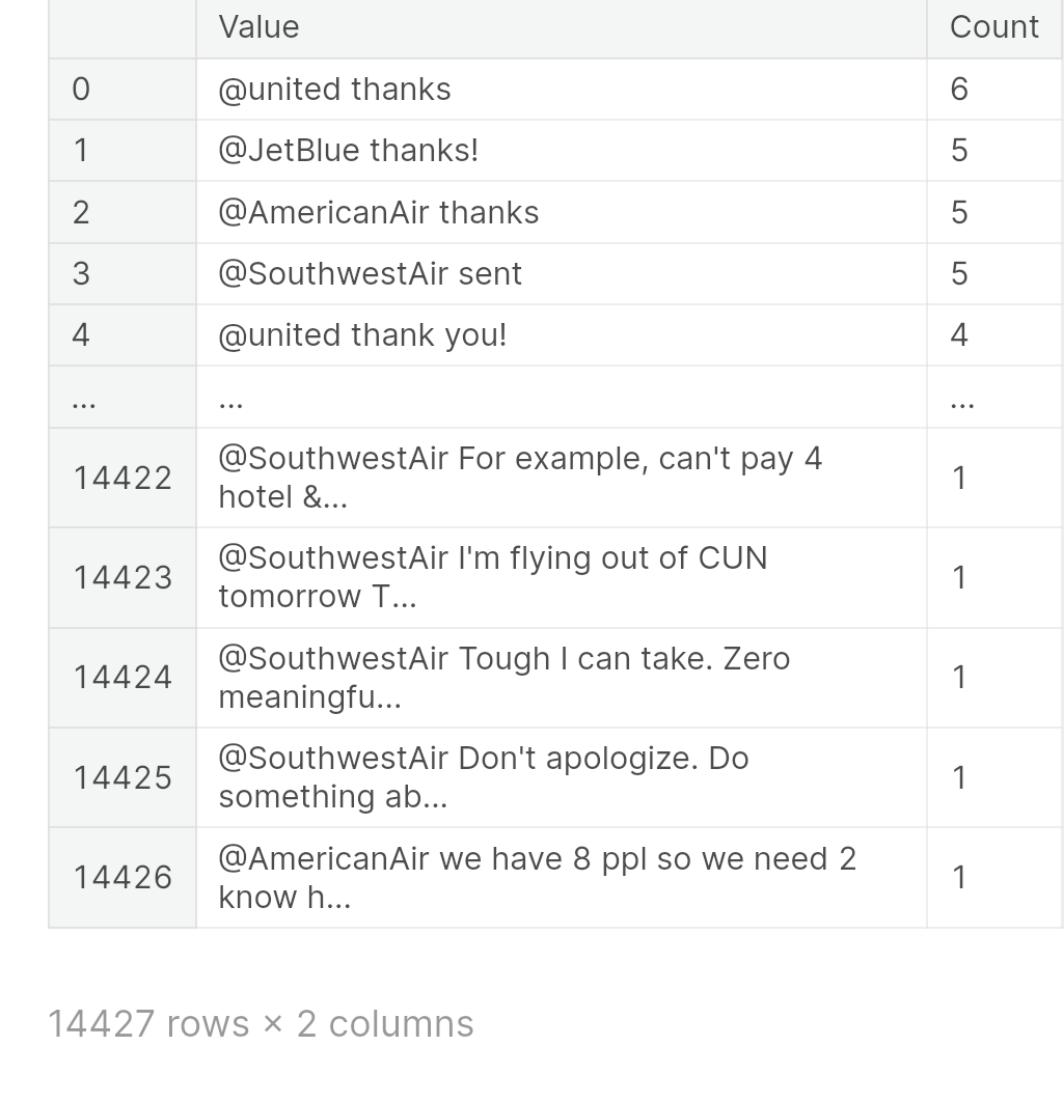
**Values counts for column ‘name’**



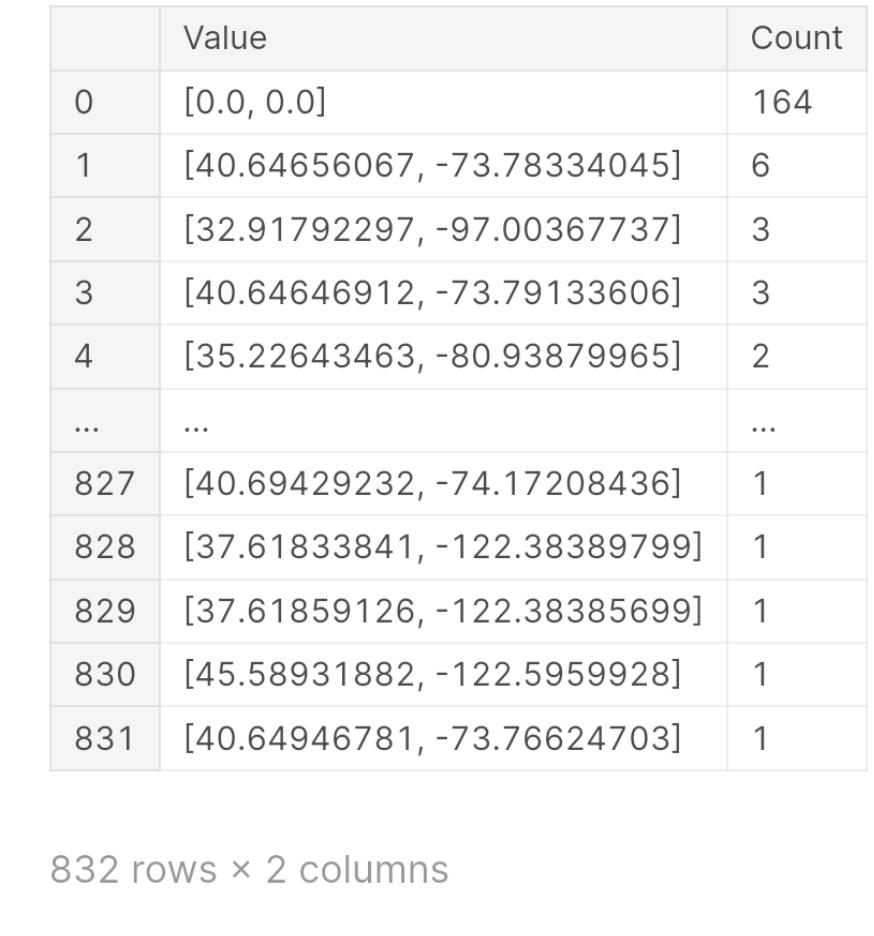
#### **Value counts for column 'negativereason\_gold'**

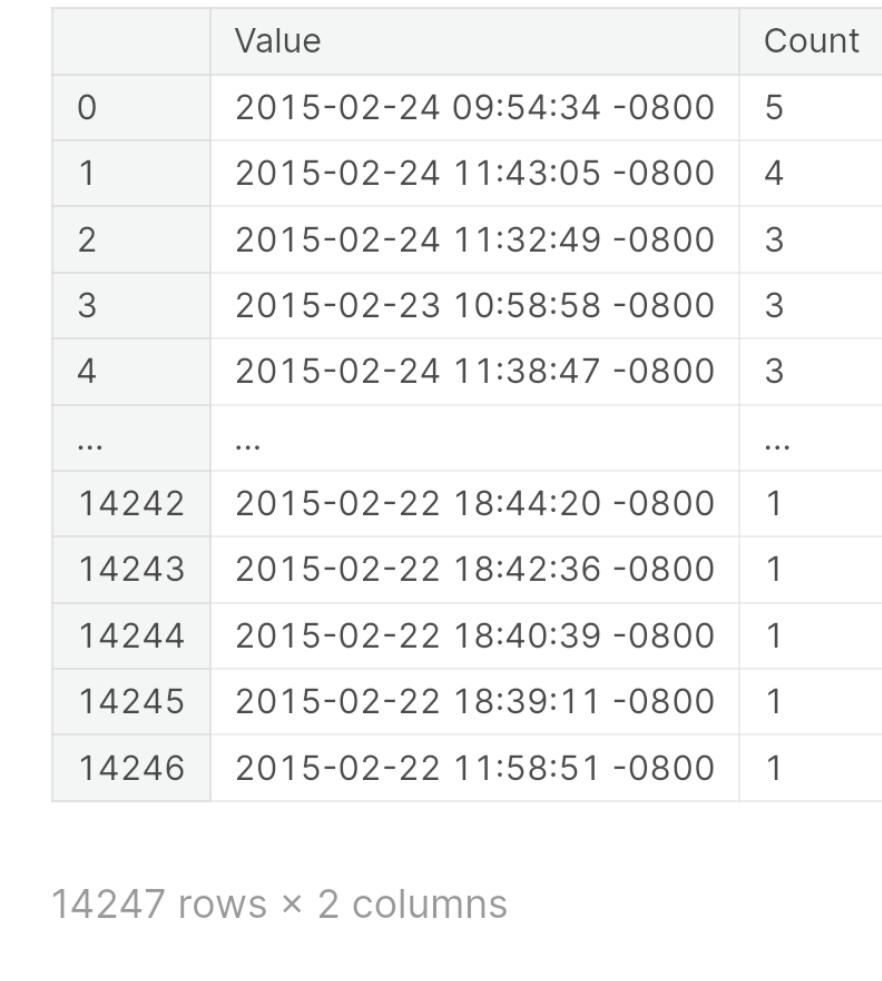


#### **Value counts for column ‘ text’**



**Value counts for column ‘tweet\_coord’**

****

**Value counts for column ‘tweet\_created’**

#### **Value counts for column 'tweet\_location'**

**Value counts for column ‘user\_timezone’:**

**PREPARE DATASET FOR ANALYSIS**

##### ***Tokenizing text column which contains the tweet***

In [7]:

max\_features = 10000

max\_length = 100

tokenizer = Tokenizer(num\_words=max\_features, split=' ')

tokenizer.fit\_on\_texts(df['text'].values)

X = tokenizer.texts\_to\_sequences(df['text'].values)

X = pad\_sequences(X, maxlen=max\_length)

y = df['airline\_sentiment'].values

##### ***Encoding Airline Sentiment Column which contains 'positive', 'negative' or 'neutral'***

In [8]:

encoder = LabelEncoder()

encoder.fit(y)

y = encoder.transform(y)

##### ***Train Test Split the data***

In [9]:

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=365)

##### ***Turn y variables into categorical***

In [10]:

from keras.utils import to\_categorical

y\_train = to\_categorical(y\_train)

y\_test = to\_categorical(y\_test)

By following these steps, we will have a clean and loaded dataset ready for sentiment analysis. From there, we can proceed with applying sentiment analysis techniques, feature extraction, visualization, and insights generation as outlined in our design thinking approach.