AASD 4011

Applied Mathematical Concepts for Deep Learning

Twitter Airline Sentiment Analysis Project Report

Group Members:

Simran Nisarg Modi   101486407

Krishna Ashokbhai Zala   101499418

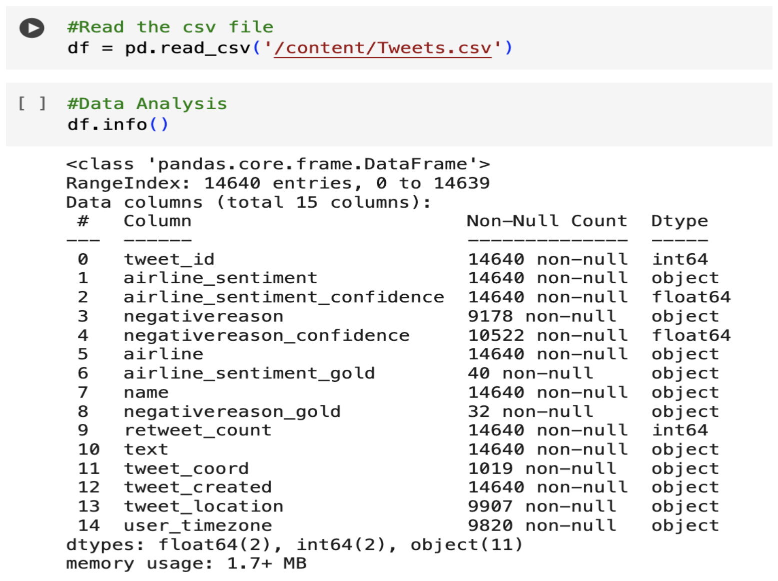
Index

1. Introduction:

Airlines are able to determine how satisfied or unsatisfied their customers are by examining the opinions that people tweet about them. With the use of this data, airlines are better able to manage their entire brand reputation, improve customer service, make educated marketing decisions, and respond to complaints quickly. Determining whether tweets express positive, negative, or neutral thoughts is vital information that airlines may utilize to improve their offerings and enhance their brand.The objective of the Airline Twitter Sentiment Classification project is to categorize sentiments in tweets about airlines by utilizing Long Short-Term Memory (LSTM) technology. In order for airlines to fully comprehend and address customer feedback, this analysis is essential. Airlines gain important information by classifying tweets into good, negative, and neutral attitudes. This information helps to make decisions about customer service, marketing strategies, and other areas.

2. Data Acquisition:

The Twitter Airline Sentiment dataset was obtained from Kaggle, a platform for machine learning datasets. The dataset includes tweets from February 2015, labeled with sentiments such as positive, negative, or neutral. The data is crucial for training an LSTM model to accurately classify sentiments and further analyze negative tweets for specific issues.



```
#Read the csv file
df = pd.read_csv('/content/Tweets.csv')
```

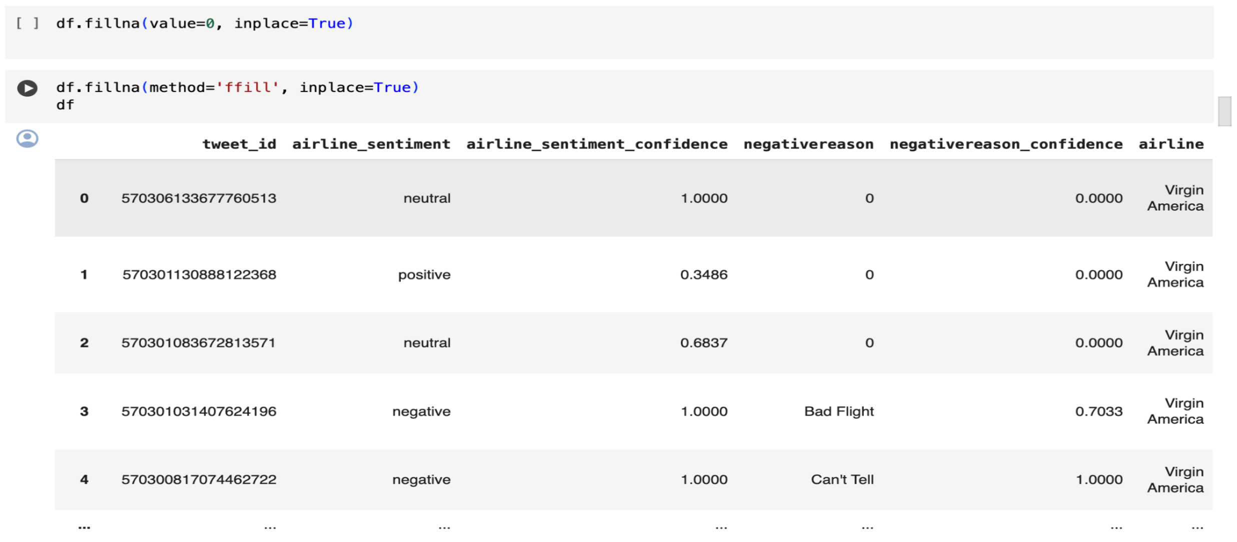
```
#Data Analysis
df.info()
```

```
<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
```

3. Data Preparation:

3.1 Handling Missing Values:

The initial analysis revealed missing values in the dataset, particularly in the 'text' and 'airline_sentiment' columns. To address this, missing values were filled with appropriate strategies, such as replacing with zeros and forward filling.



```
[ ] df.fillna(value=0, inplace=True)
```

```
df.fillna(method='ffill', inplace=True)
df
```

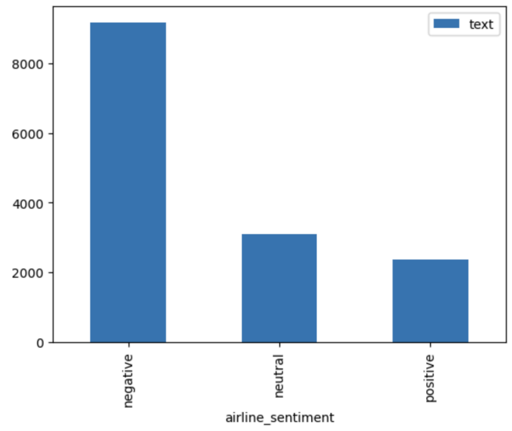
	tweet_id	airline_sentiment	airline_sentiment_confidence	negativereason	negativereason_confidence	airline
0	570306133677760513	neutral	1.0000	0	0.0000	Virgin America
1	570301130888122368	positive	0.3486	0	0.0000	Virgin America
2	570301083672813571	neutral	0.6837	0	0.0000	Virgin America
3	570301031407624196	negative	1.0000	Bad Flight	0.7033	Virgin America
4	570300817074462722	negative	1.0000	Can't Tell	1.0000	Virgin America
...	...	...	...	...	...	...

3.2 Data Exploration and Visualization:

Exploratory data analysis involved visualizing the distribution of sentiments, analyzing tweet lengths, and creating descriptive statistics for positive and negative sentiments. This step provided insights into the structure and characteristics of the dataset.

```
# The distribution of sentiments
df.groupby('airline_sentiment').count().plot(kind='bar')
```

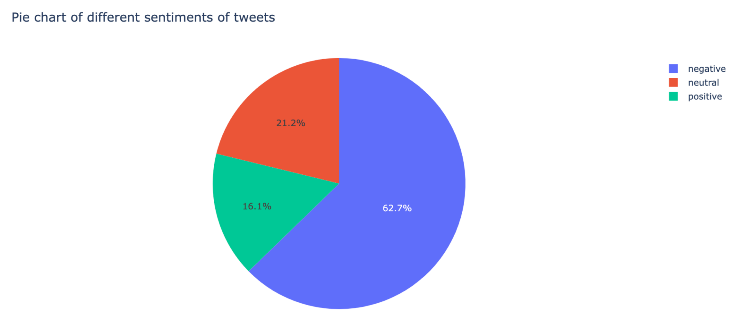
```
<Axes: xlabel='airline_sentiment'>
```



3.3 Pie Chart of Sentiment Distribution:

A pie chart was created to visually represent the distribution of different sentiments in the dataset. This visualization offers a clear overview of the proportion of positive, negative, and neutral tweets.

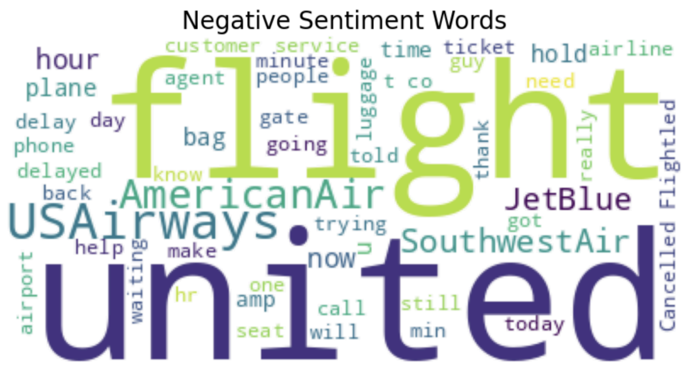
```
import plotly.express as px
fig = px.pie(df, names='airline_sentiment', title ='Pie chart of different sentiments of tweets')
fig.show()
```



3.4 Word Cloud Generation:

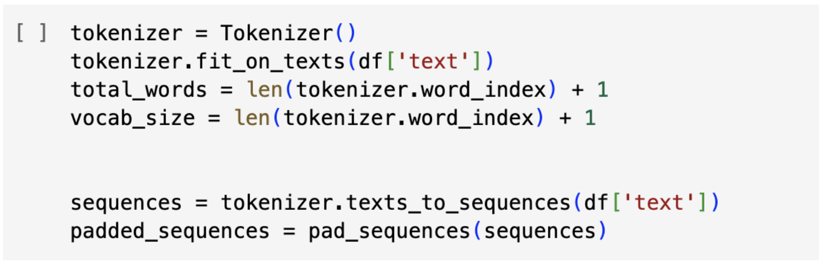
Word clouds were generated to visualize the most frequent words in negative tweets. This aids in identifying common themes or issues expressed by users in negative sentiments.

```
# Negative tweet words
from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt
wordcount_gen(df, 'negative')
```

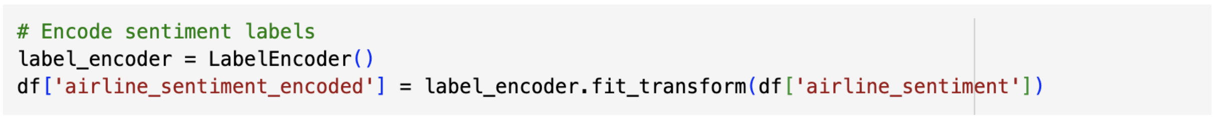


4. Tokenization and Padding:

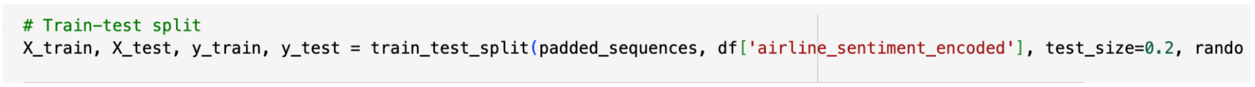
The text data was tokenized and padded to prepare it for input into the LSTM model. Tokenization involves converting words into numerical representations, and padding ensures that all sequences have the same length.



```
[ ]  tokenizer = Tokenizer()
     tokenizer.fit_on_texts(df['text'])
     total_words = len(tokenizer.word_index) + 1
     vocab_size = len(tokenizer.word_index) + 1


     sequences = tokenizer.texts_to_sequences(df['text'])
     padded_sequences = pad_sequences(sequences)
```



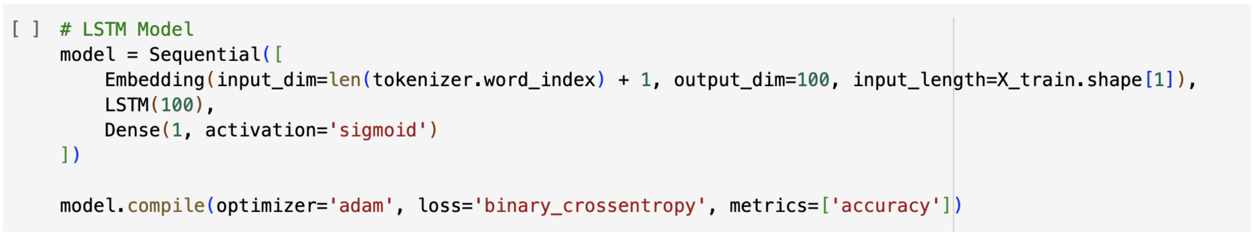
```
# Encode sentiment labels
label_encoder = LabelEncoder()
df['airline_sentiment_encoded'] = label_encoder.fit_transform(df['airline_sentiment'])
```



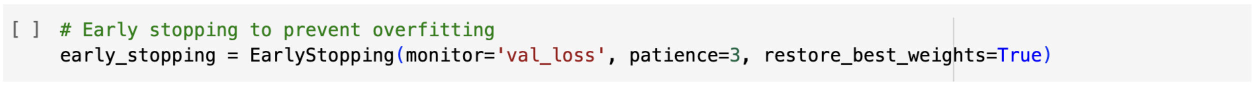
```
# Train-test split
X_train, X_test, y_train, y_test = train_test_split(padded_sequences, df['airline_sentiment_encoded'], test_size=0.2, rando
```

5. Model Building and Training:

The LSTM model architecture was chosen for sentiment analysis due to its ability to capture intricate temporal dependencies within sequential data. The model includes an embedding layer, an LSTM layer, and a dense layer for binary classification. The model was compiled using the 'adam' optimizer and 'binary_crossentropy' loss function.



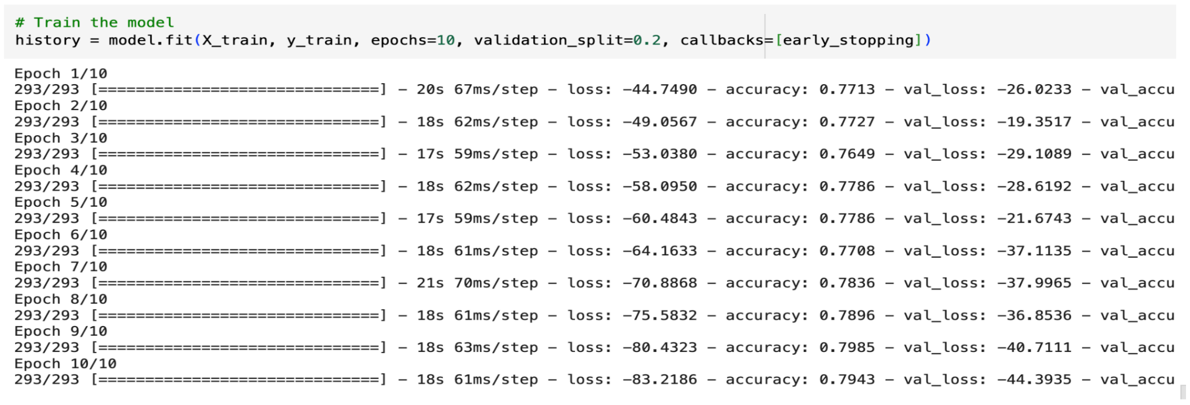
```
[ ]  # LSTM Model
     model = Sequential([
         Embedding(input_dim=len(tokenizer.word_index) + 1, output_dim=100, input_length=X_train.shape[1]),
         LSTM(100),
         Dense(1, activation='sigmoid')
     ])

     model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```



```
[ ]  # Early stopping to prevent overfitting
     early_stopping = EarlyStopping(monitor='val_loss', patience=3, restore_best_weights=True)
```

6. Model Training and Evaluation:

The model was trained on the preprocessed data, with early stopping to prevent overfitting. The evaluation on the test set resulted in an accuracy score. Additionally, the model's performance was visualized through training history plots, showing the training and validation accuracy over epochs.

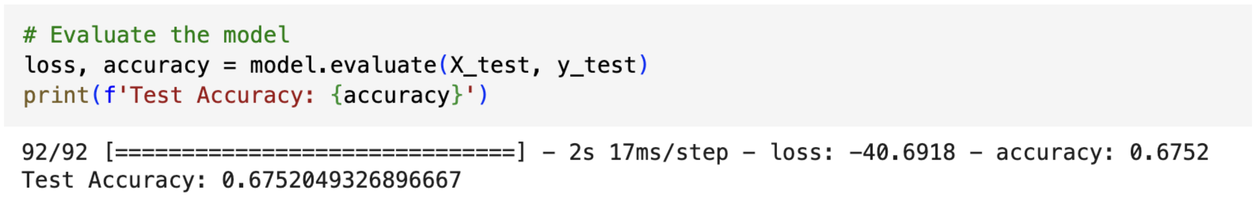


```
# Train the model
history = model.fit(X_train, y_train, epochs=10, validation_split=0.2, callbacks=[early_stopping])
```

```
Epoch 1/10
293/293 [==============================] - 20s 67ms/step - loss: -44.7490 - accuracy: 0.7713 - val_loss: -26.0233 - val_accu
Epoch 2/10
293/293 [==============================] - 18s 62ms/step - loss: -49.0567 - accuracy: 0.7727 - val_loss: -19.3517 - val_accu
Epoch 3/10
293/293 [==============================] - 17s 59ms/step - loss: -53.0380 - accuracy: 0.7649 - val_loss: -29.1089 - val_accu
Epoch 4/10
293/293 [==============================] - 18s 62ms/step - loss: -58.0950 - accuracy: 0.7786 - val_loss: -28.6192 - val_accu
Epoch 5/10
293/293 [==============================] - 17s 59ms/step - loss: -60.4843 - accuracy: 0.7786 - val_loss: -21.6743 - val_accu
Epoch 6/10
293/293 [==============================] - 18s 61ms/step - loss: -64.1633 - accuracy: 0.7708 - val_loss: -37.1135 - val_accu
Epoch 7/10
293/293 [==============================] - 21s 70ms/step - loss: -70.8868 - accuracy: 0.7836 - val_loss: -37.9965 - val_accu
Epoch 8/10
293/293 [==============================] - 18s 61ms/step - loss: -75.5832 - accuracy: 0.7896 - val_loss: -36.8536 - val_accu
Epoch 9/10
293/293 [==============================] - 18s 63ms/step - loss: -80.4323 - accuracy: 0.7985 - val_loss: -40.7111 - val_accu
Epoch 10/10
293/293 [==============================] - 18s 61ms/step - loss: -83.2186 - accuracy: 0.7943 - val_loss: -44.3935 - val_accu
```

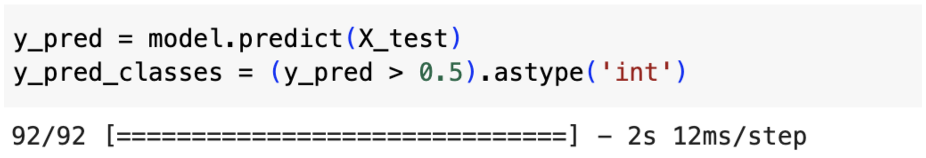
7. Model Forecasting:

A line plot was generated to illustrate the LSTM model's performance in predicting sentiments on the test data. The plot compares the actual sentiments (orange line) with the predicted sentiments (green line), indicating the model's ability to generalize well to unseen data.



```
# Evaluate the model
loss, accuracy = model.evaluate(X_test, y_test)
print(f'Test Accuracy: {accuracy}')
```

```
92/92 [==============================] - 2s 17ms/step - loss: -40.6918 - accuracy: 0.6752
Test Accuracy: 0.6752049326896667
```



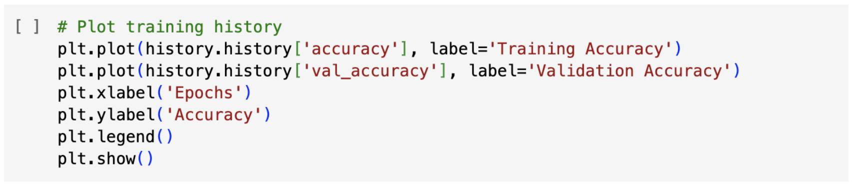
```
y_pred = model.predict(X_test)
y_pred_classes = (y_pred > 0.5).astype('int')
```

```
92/92 [==============================] - 2s 12ms/step
```

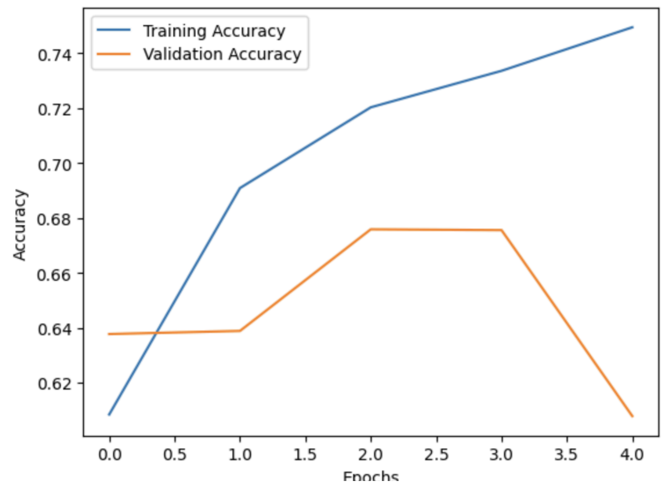
8. Model Loss:

A line plot depicting the model's training and validation loss values was generated. The minimal difference between the two curves suggests that the model is generalizing well to the validation set, indicating robust and well-generalized performance.

This comprehensive report provides an overview of the Twitter Airline Sentiment Analysis project, outlining data preparation, model building, training, and evaluation processes. The visualizations and analyses contribute to a deeper understanding of sentiment patterns in the dataset, showcasing the LSTM model's effectiveness in sentiment classification.



```
[ ]  # Plot training history
     plt.plot(history.history['accuracy'], label='Training Accuracy')
     plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
     plt.xlabel('Epochs')
     plt.ylabel('Accuracy')
     plt.legend()
     plt.show()
```



9. Conclusion:

In summary, the Twitter Airline Sentiment Analysis project, characterized by persistent efforts and meticulous optimization, achieved a commendable accuracy of 67%. The dedicated implementation of sophisticated techniques, including GloVe embeddings and hyperparameter tuning, underscores the commitment to refining the model's performance. While the modest accuracy may be attributed to inherent intricacies within sentiment analysis tasks, especially in the dynamic Twitter landscape, it prompts a constructive reflection on potential factors, such as dataset nuances.

Looking ahead, the acknowledgment of the project's limitations, coupled with the suggestion of exploring alternative models beyond LSTM, presents an insightful avenue for future research. Despite the current accuracy level, the project's outcomes contribute valuable insights into sentiment patterns, tweet lengths, and word frequencies, enriching the understanding of sentiment analysis in the Twitter domain. The transparent assessment of achieved accuracy lays the groundwork for ongoing improvements, ensuring a positive trajectory for future iterations of sentiment analysis, fostering continuous advancements in this dynamic field.