**MIS581 Critical Thinking Assignment: Capstone GitHub**

Stephan Thrun

Colorado State University – Global Campus

MIS581: Capstone – Business Intelligence and Data Analytics

Dr. Osama Morad

7 April, 2024

**Abstract**

Sentiment analysis is a subset within artificial intelligence and data analytics that has been getting more attention over the past decade. With developments within artificial intelligence and machine learning happening every year, many businesses are taking notice of the potential advantages that sentiment analysis provides for company decision makers. Within this Capstone Project, a data set with 14640 user Tweets reacting to six US based airline companies from February 2015 is explored for already provided sentiment analysis and will look at the relationships between sentiment analysis results and how customers feel about service. This will be done using Python to show various descriptive statistics testing results on the data set.

Findings from analyzing the data set will be discussed to explore the relationship between effects of sentiment analysis and customer responses to company services or products and machine learning using the Naive Bayes Theorem will also be explored to automate any sentiment analysis efforts within a company if findings prove the hypothesis as truthful. Machine learning testing will also be conducted using Python.

If there are any questions about the dataset, please feel free to reach out to me at sutoran.thrun.5@gmail.com

**Introduction**

With expanding developments in artificial intelligence and machine learning every year, individual company business intelligence offices can benefit from these developments to offer new insights to decision makers and company stakeholders. One development in this area is sentiment analysis. Taderhoost and Madanchian define it as “the process of finding, evaluating, and categorizing as negative (-1), neutral (0), or positive (+1) the emotions expressed by individuals in any text data kind (2023).” Using artificial intelligence to harness the power of social media networks with this technique, businesses can quickly see how customer bases may feel about products or may even provide insights on a company social media post if unveiling a new developmental product.

The capstone project is using a dataset provided by Vanjana, which already has sentiment analysis results for 14640 Twitter post entries across six different US based airline companies (2020). The goal and intent of this capstone project is to use artificial intelligence with sentiment analysis to harness the power of social media networks. With the speed businesses capture data for sentiment analysis, it is also important to examine the ethical considerations government entities are considering and implementing so users can be informed about their data.

The following sections will just be dedicated to showing the Python code used for the data analysis as well as the machine learning testing on the dataset.

**Data Analysis Python Coding**

The data has been analyzed using Python to present findings from the Twitter dataset. These results are displayed below in Figures 1-8. Necessary coding will be displayed before each Figure.

import numpy as np

import pandas as pd

import re

import nltk

import matplotlib.pyplot as plt

%matplotlib inline

data\_source\_url = r"C:/Users/sutor/OneDrive/Desktop/Masters/Class/MIS 581/Capstone Project/Tweets.csv"

airline\_tweets = pd.read\_csv("C:/Users/sutor/OneDrive/Desktop/Masters/Class/MIS 581/Capstone Project/Tweets.csv")

airline\_tweets.head()

airline\_tweets.shape

airline\_tweets.airline\_sentiment.value\_counts()

plot\_size = plt.rcParams["figure.figsize"]

print(plot\_size[0])

print(plot\_size[1])

plot\_size[0] = 8

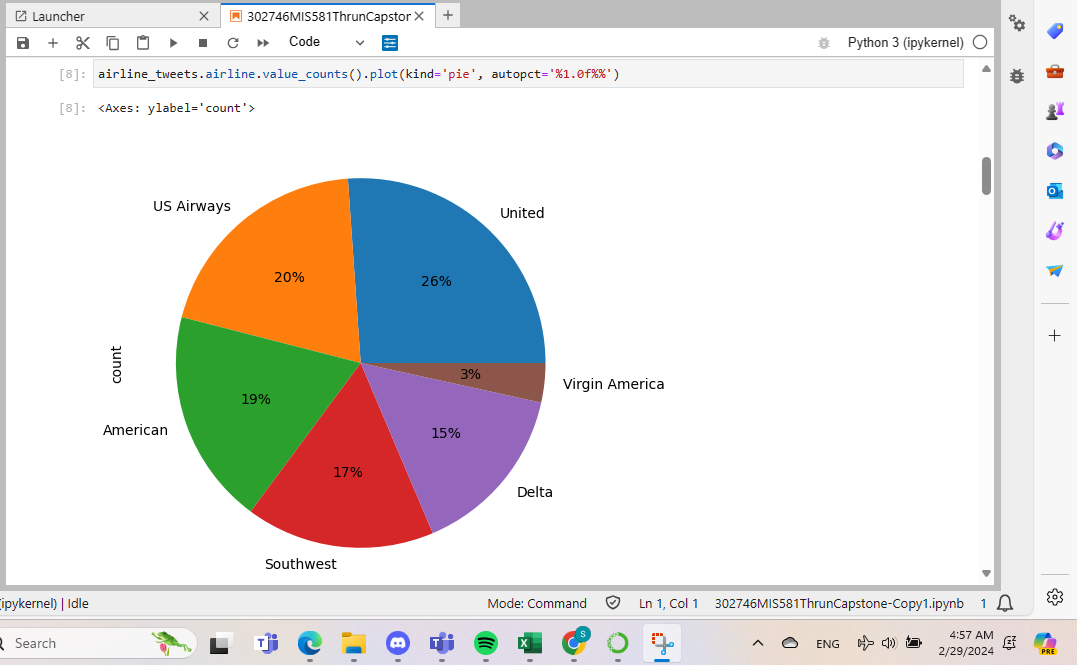
plot\_size[1] = 6

plt.rcParams["figure.figsize"] = plot\_size

airline\_tweets.airline.value\_counts().plot(kind='pie', autopct='%1.0f%%')

**Figure 1**

*Pie Chart Results of Entry Percentage by Airline*

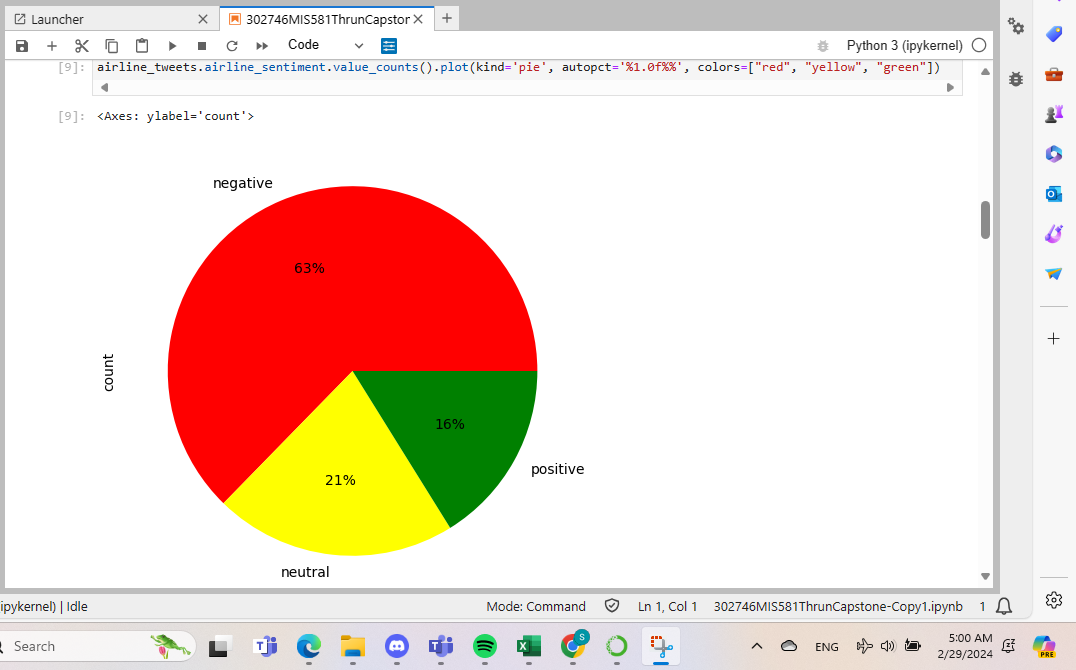


*Note: Performed in Jupyter Lab.*

airline\_tweets.airline\_sentiment.value\_counts().plot(kind='pie', autopct='%1.0f%%', colors=["red", "yellow", "green"])

**Figure 2**

*Pie Chart Results of Entry Percentage by Sentiment Response*



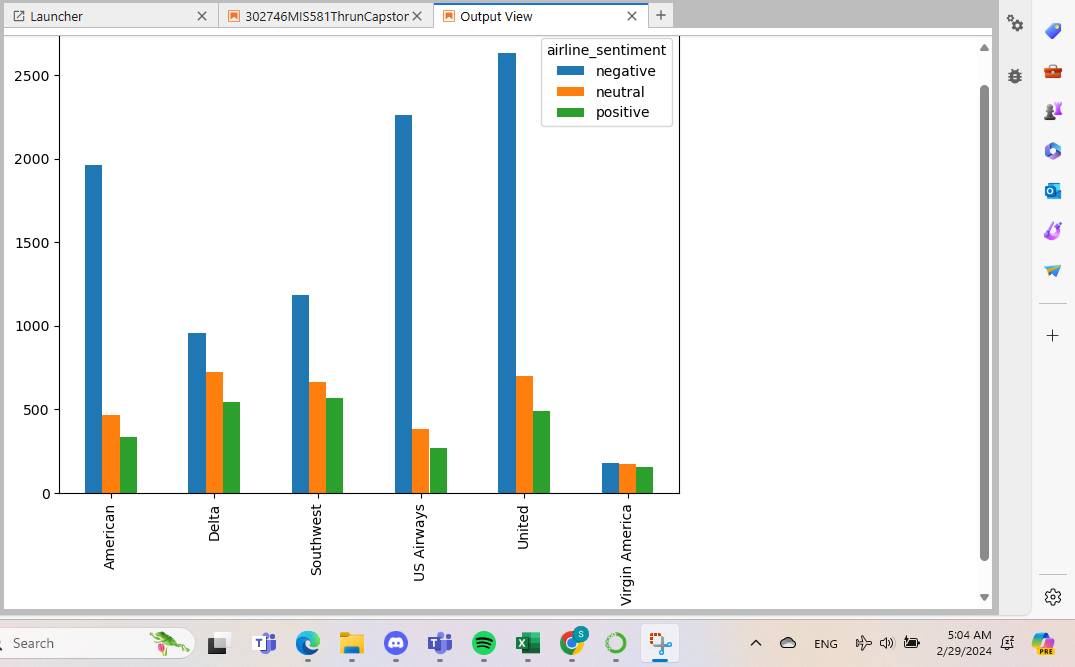
*Note: Performed in Jupyter Lab.*

airline\_sentiment = airline\_tweets.groupby(['airline', 'airline\_sentiment']).airline\_sentiment.count().unstack()

airline\_sentiment.plot(kind='bar')

**Figure 3**

*Bar Graph Results of Sentiment by Airline*



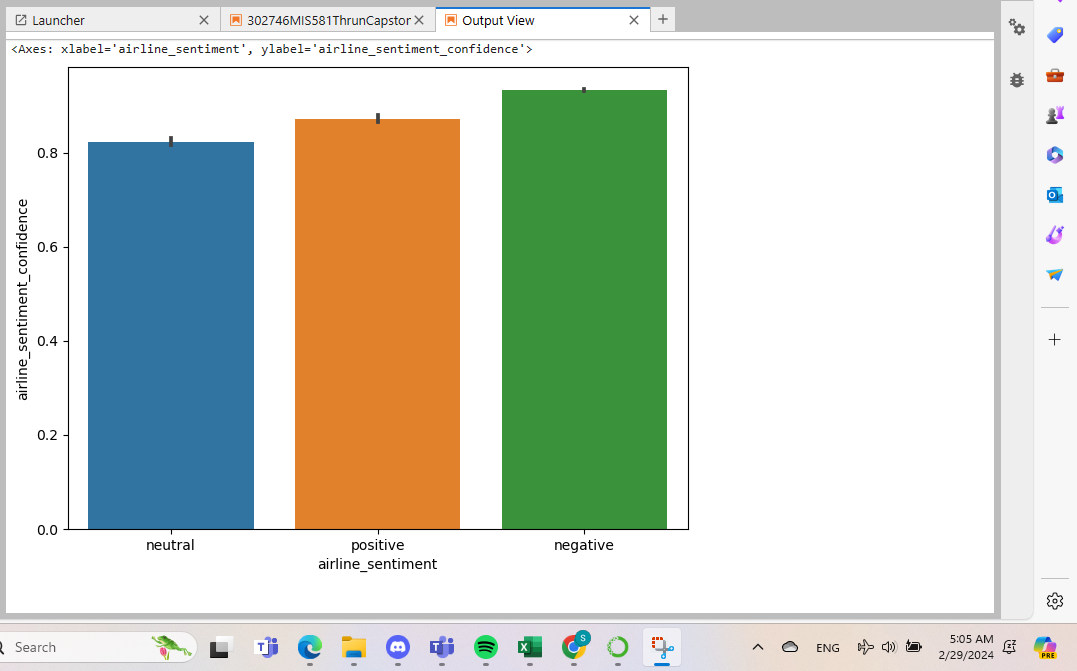
*Note: Performed in Jupyter Lab.*

import seaborn as sns

sns.barplot(x='airline\_sentiment', y='airline\_sentiment\_confidence' , data=airline\_tweets)

**Figure 4**

*Bar Graph Dataset Results by Sentiment*



*Note: Performed in Jupyter Lab.*

def plot\_sub\_sentiment(Airline):

pdf = airline\_tweets[airline\_tweets['airline']==Airline]

count = pdf['airline\_sentiment'].value\_counts()

Index = [1,2,3]

color=sns.color\_palette("husl", 10)

plt.bar(Index,count,width=0.5,color=color)

plt.xticks(Index,['Negative','Neutral','Positive'])

plt.title('Sentiment Summary of' + " " + Airline)

airline\_name = airline\_tweets['airline'].unique()

plt.figure(1,figsize=(12,12))

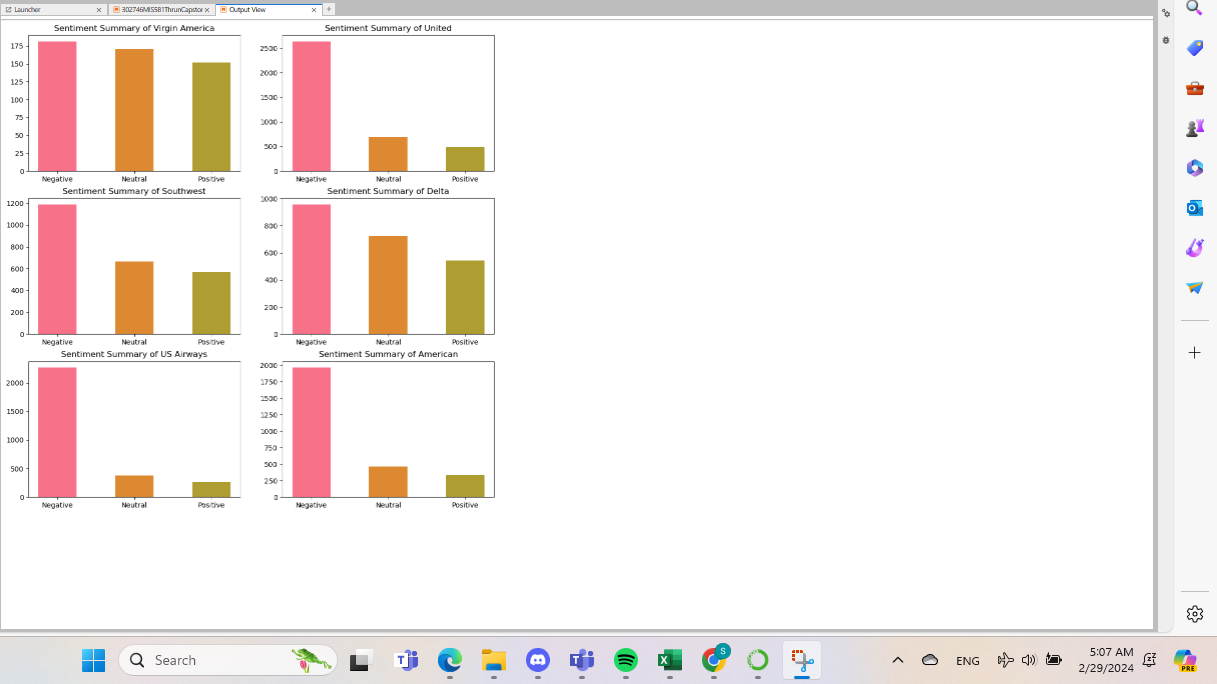
for i in range(6):

plt.subplot(3,2,i+1)

plot\_sub\_sentiment(airline\_name[i])

**Figure 5**

*Bar Graph Sentiment Summary Results by Airline*



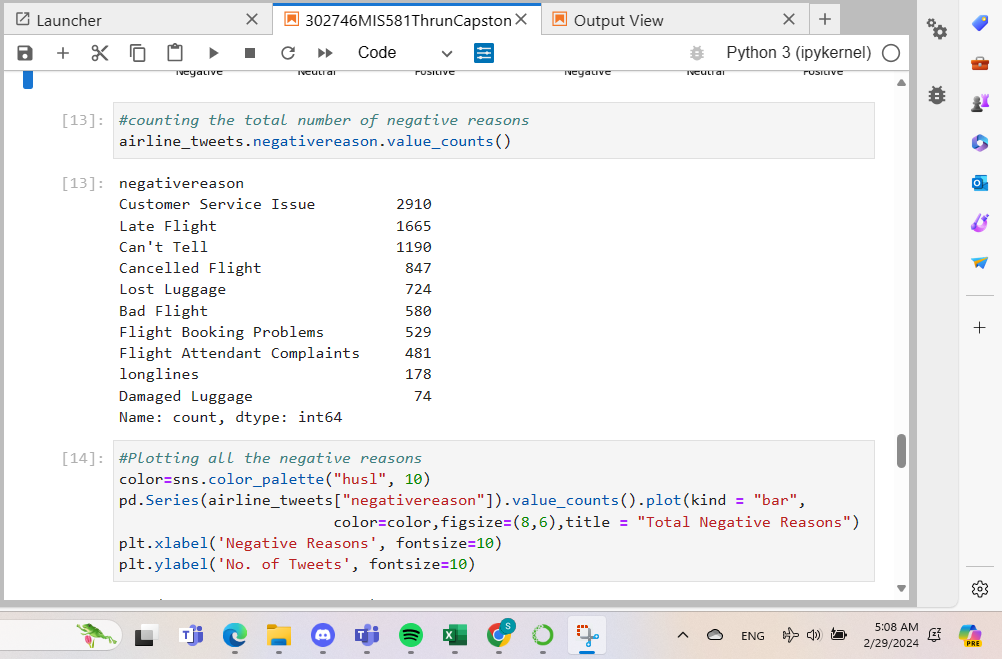
*Note: Performed in Jupyter Lab.*

#counting the total number of negative reasons

airline\_tweets.negativereason.value\_counts()

**Figure 6**

*Total Count of Different Types of Negative Responses*



*Note: Performed in Jupyter Lab.*

#Plotting all the negative reasons

color=sns.color\_palette("husl", 10)

pd.Series(airline\_tweets["negativereason"]).value\_counts().plot(kind = "bar",

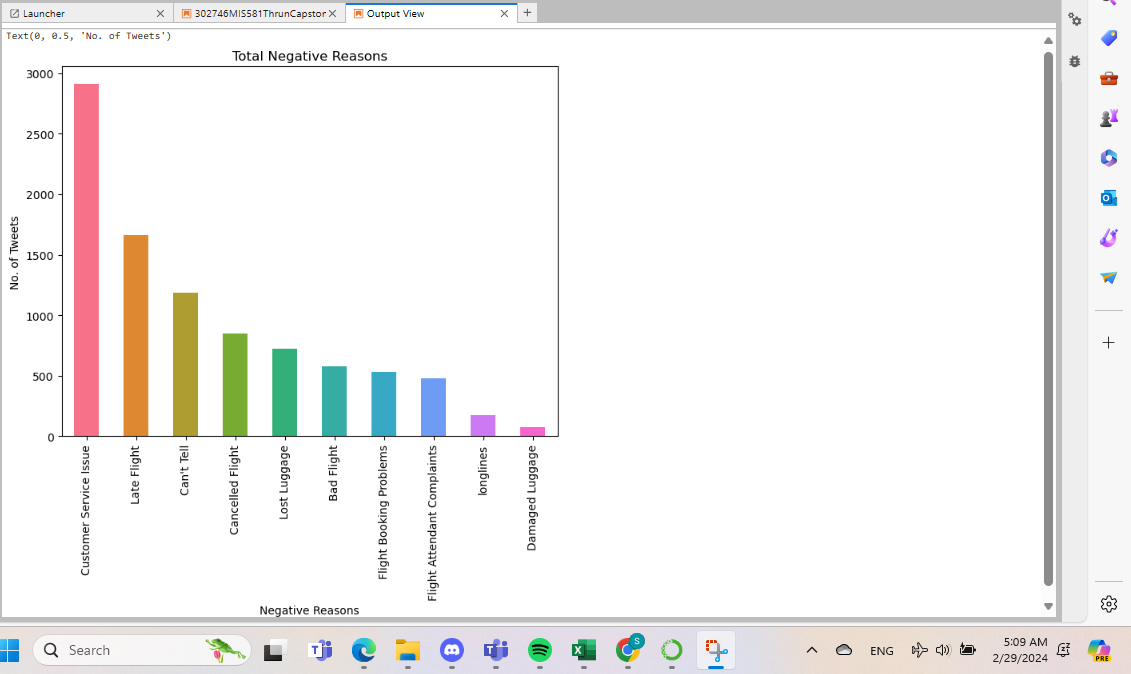
color=color,figsize=(8,6),title = "Total Negative Reasons")

plt.xlabel('Negative Reasons', fontsize=10)

plt.ylabel('No. of Tweets', fontsize=10)

**Figure 7**

*Bar Graph Results of Total Negative Responses*



*Note: Performed in Jupyter Lab.*

!pip install wordcloud

from wordcloud import WordCloud,STOPWORDS

airline\_tweets=airline\_tweets [airline\_tweets ['airline\_sentiment']=='negative']

words = ' '.join(airline\_tweets ['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=(12, 12))

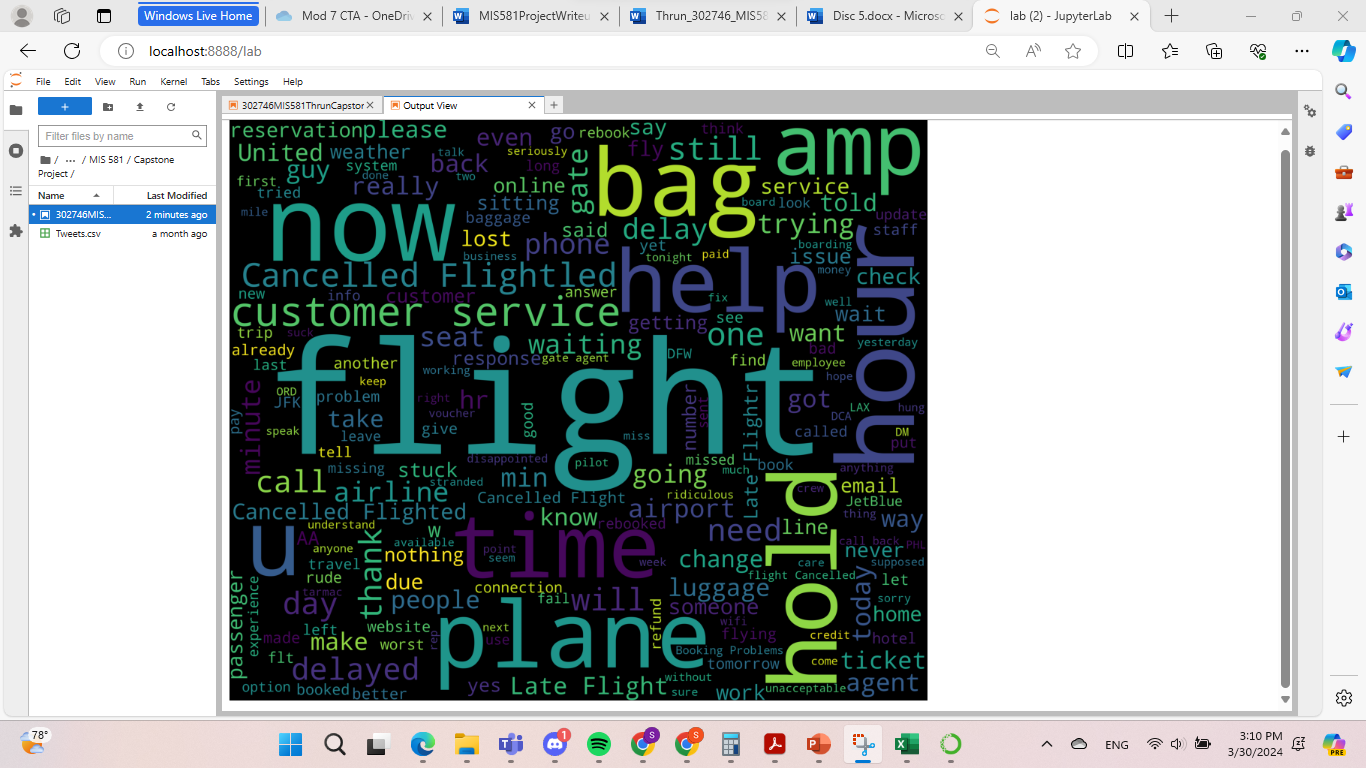
plt.imshow(wordcloud)

plt.axis('off')

plt.show()

**Figure 8**

*Word Cloud Results of Total Negative Responses*



*Note: Performed in Jupyter Lab.*

**Data Cleaning and Machine Learning Result Coding**

The dataset was investigated further to get ready for data cleaning and machine learning scenarios. This was a bit of an undertaking because there was no previous experience attempting to go through and clean data for 14640 Twitter posts and attempt machine learning on a bigger data set like this before.

As for the data cleaning, the data set was examined to remove characteristics and standardize characters to improve machine learning results. The data cleaning involved removing special characters in each Twitter post as well as removing any extra spaces in the Tweets. This proved relatively successful and was carried out for each data entry using the Python coding below.

air\_senti=pd.crosstab(airline\_tweets.airline, airline\_tweets.airline\_sentiment)

air\_senti

percent=air\_senti.apply(lambda a: a / a.sum() \* 100, axis=1)

percent

pd.crosstab(index = airline\_tweets["airline"],columns = airline\_tweets["airline\_sentiment"]).plot(kind='bar',

figsize=(10, 6),alpha=0.5,rot=0,stacked=True,title="Airline Sentiment")

airline\_tweets['tweet\_created'] = pd.to\_datetime(airline\_tweets['tweet\_created'])

airline\_tweets["date\_created"] = airline\_tweets["tweet\_created"].dt.date

airline\_tweets["date\_created"]

df = airline\_tweets.groupby(['date\_created','airline'])

df = df.airline\_sentiment.value\_counts()

df.unstack()

features = airline\_tweets.iloc[:, 10].values

labels = airline\_tweets.iloc[:, 1].values

features

labels

processed\_features = []

for sentence in range(0, len(features)):

# Remove all the special characters

processed\_feature = re.sub(r'\W', ' ', str(features[sentence]))

# remove all single characters

processed\_feature= re.sub(r'\s+[a-zA-Z]\s+', ' ', processed\_feature)

# Remove single characters from the start

processed\_feature = re.sub(r'\^[a-zA-Z]\s+', ' ', processed\_feature)

# Substituting multiple spaces with single space

processed\_feature = re.sub(r'\s+', ' ', processed\_feature, flags=re.I)

# Removing prefixed 'b'

processed\_feature = re.sub(r'^b\s+', '', processed\_feature)

# Converting to Lowercase

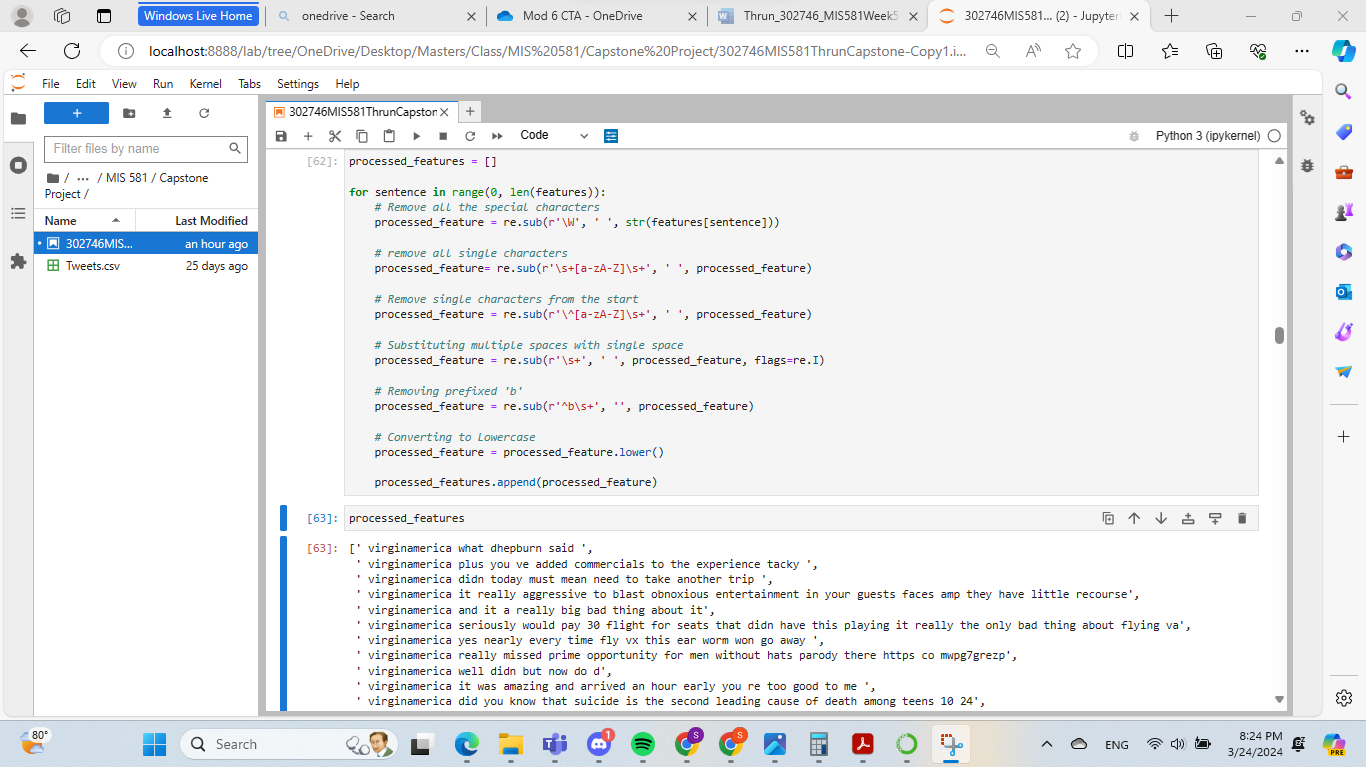
processed\_feature = processed\_feature.lower()

processed\_features.append(processed\_feature)

processed\_features

**Figure 9**

*Data Cleaning Commands for Cleaning Entire Data Set*



*Note: Performed in Jupyter Lab.*

Not pictured in Figure 9 above is how the user data was masked with this dataset. To further protect the username data associated with each twitter post, numbers were randomly assigned from 1 to 14640 to replace the individual usernames in excel. This was done through creating a list up to 14640 in excel. After this, the RAND() function was used in an adjacent column up to 14640 cells. From this, the data was sorted by the values assigned with the RAND() function. This eliminated the usernames from the dataset and gave back further anonymity by randomizing the numbers further with the RAND() function.

To have algorithms work with text-based data, the text needed to be converted to numbers. The natural language toolkit was used to do this and the TF-IDF approach was adopted by Sanjana to do so (2020). This can be interpreted as term frequency (TF) and Inverse Document frequency (IDF). The results for this are displayed in Figure 10 below and is calculated as follows: TF = (Frequency of word in document)/(Total words in the document) IDF = Log((Total number of docs)/(Number of docs containing the word)).

nltk.download('stopwords')

from nltk.corpus import stopwords

from sklearn.feature\_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer (max\_features=2500, min\_df=7, max\_df=0.8, stop\_words=stopwords.words('english'))

processed\_features = vectorizer.fit\_transform(processed\_features).toarray()

processed\_features

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

X\_train, X\_test, y\_train, y\_test = train\_test\_split(processed\_features, labels, test\_size=0.2, random\_state=0)

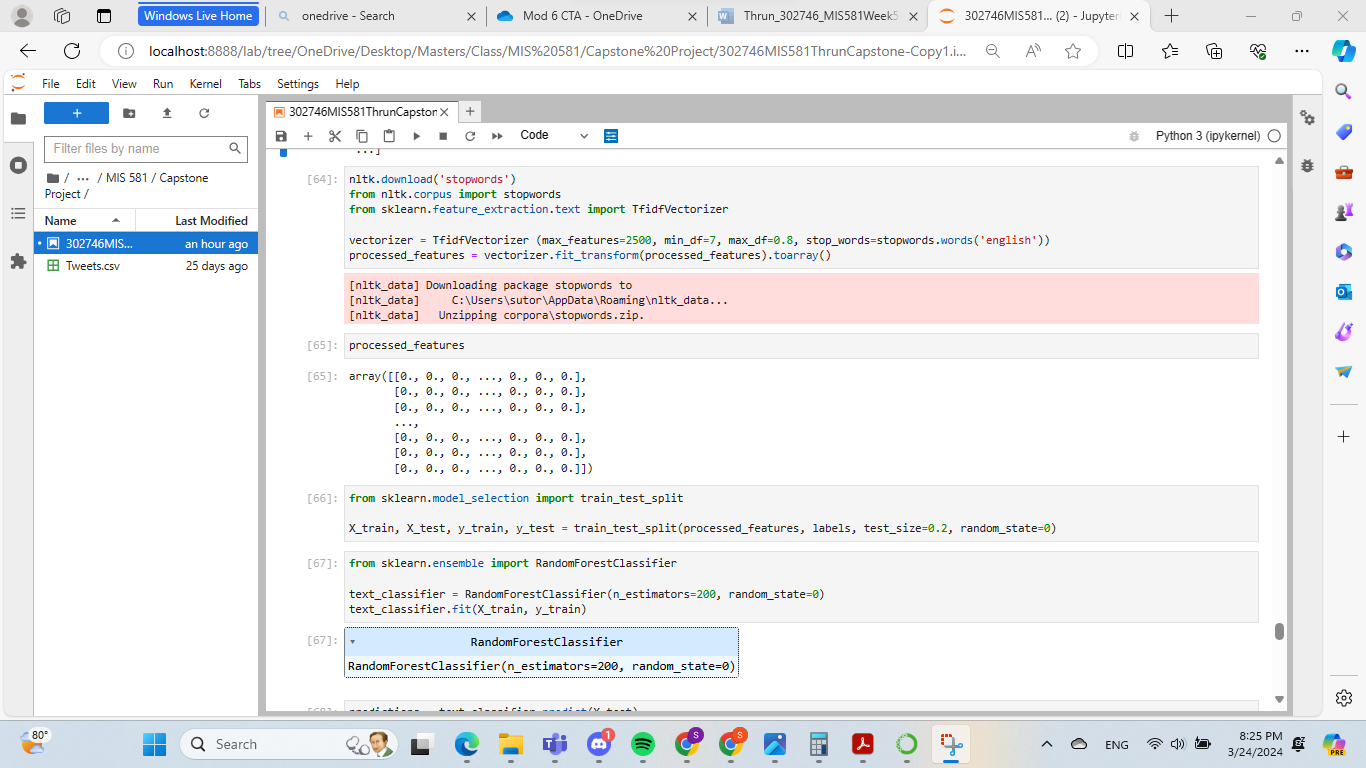
from sklearn.ensemble import RandomForestClassifier

text\_classifier = RandomForestClassifier(n\_estimators=200, random\_state=0)

text\_classifier.fit(X\_train, y\_train)

**Figure 10**

*Converting Text to Numeric Representation*



*Note: Performed in Jupyter Lab.*

Below in Figures 11-14, several types of machine learning results were attempted with various degrees of success. Coding is also displayed before each Figure. Figure 14 will conclude this document.

predictions = text\_classifier.predict(X\_test)

#Baseline RandomForest Confusion Matrix

from sklearn.metrics import classification\_report, confusion\_matrix, accuracy\_score

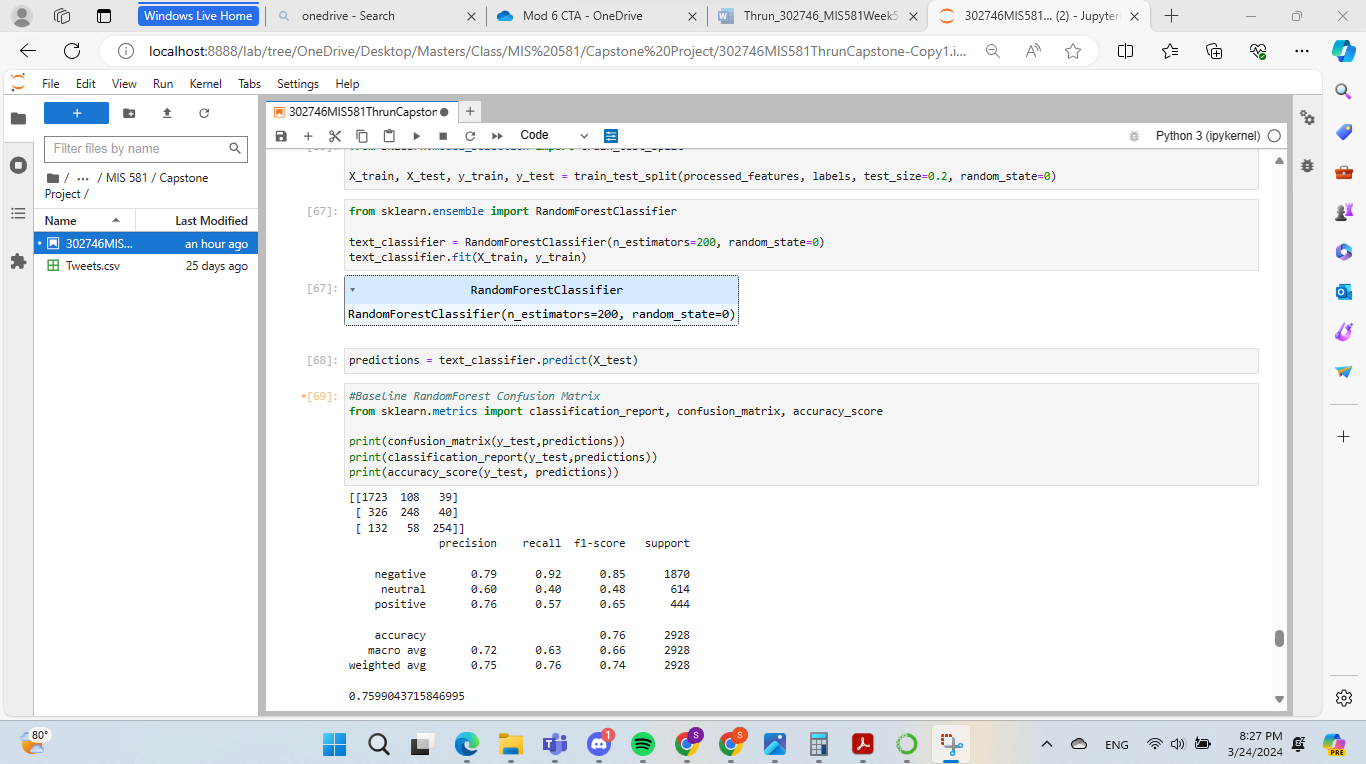
print(confusion\_matrix(y\_test,predictions))

print(classification\_report(y\_test,predictions))

print(accuracy\_score(y\_test, predictions))

**Figure 11**

*Random Forest Confusion Matrix Results*



*Note: Performed in Jupyter Lab.*

from sklearn.neighbors import KNeighborsClassifier

text\_classifier2 = KNeighborsClassifier(n\_neighbors = 5)#no of neighbors is hyper parameter

text\_classifier2.fit(X\_train, y\_train)

predictions2 = text\_classifier2.predict(X\_test)

#KNearestNeighbors Confusion Matrix

from sklearn.metrics import classification\_report, confusion\_matrix, accuracy\_score

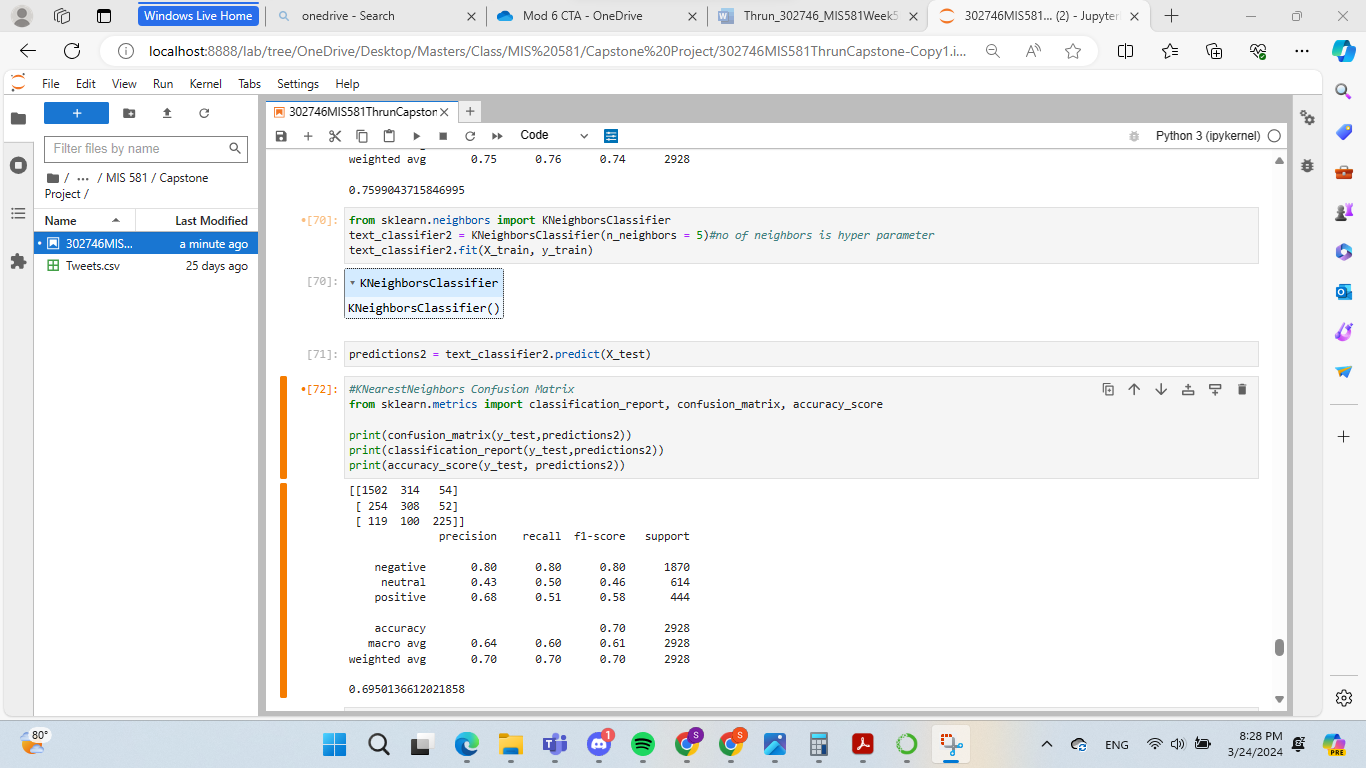
print(confusion\_matrix(y\_test,predictions2))

print(classification\_report(y\_test,predictions2))

print(accuracy\_score(y\_test, predictions2))

**Figure 12**

*K Nearest Neighbors Confusion Matrix Results*



*Note: Performed in Jupyter Lab.*

from sklearn.linear\_model import LogisticRegression

model =LogisticRegression()

model.fit(X\_train, y\_train)

predictions3 = model.predict(X\_test)

#Logistic Regression Confusion Matrix

from sklearn.metrics import classification\_report, confusion\_matrix, accuracy\_score

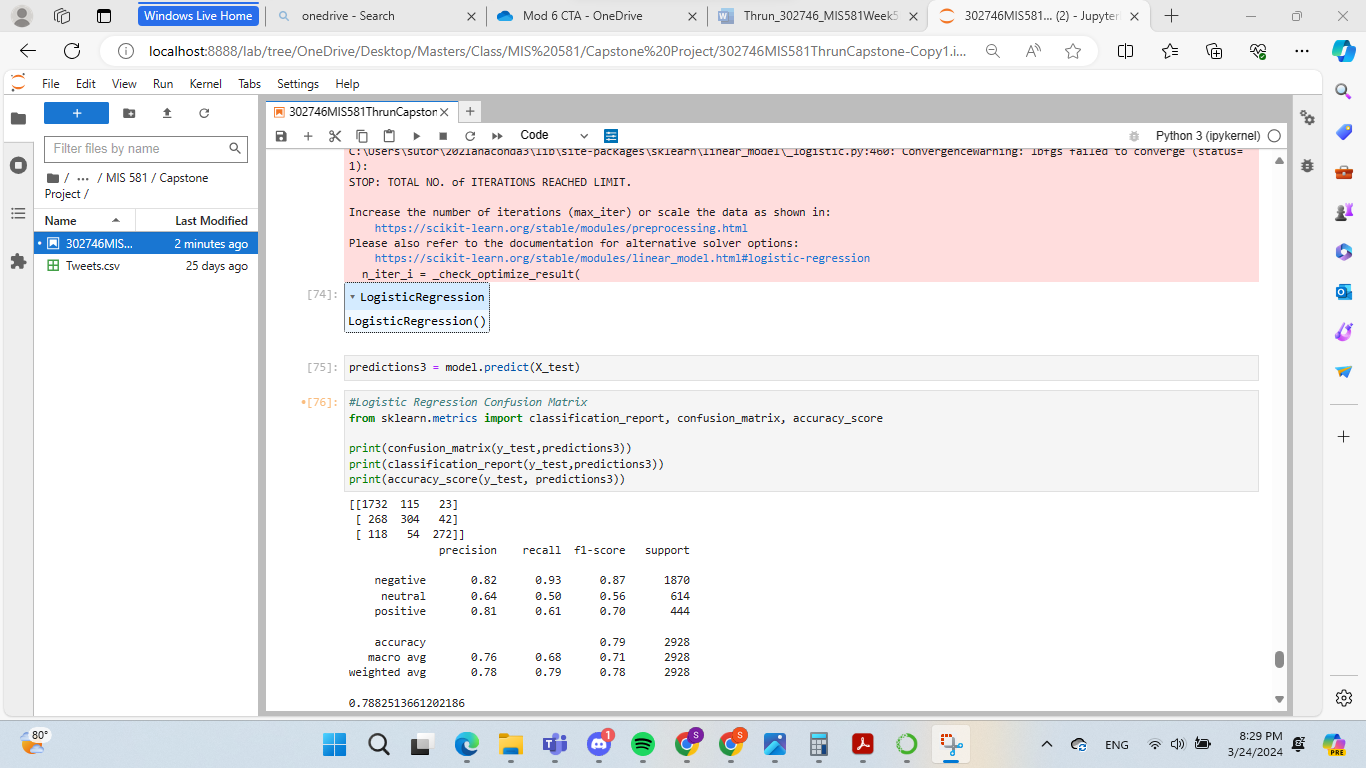
print(confusion\_matrix(y\_test,predictions3))

print(classification\_report(y\_test,predictions3))

print(accuracy\_score(y\_test, predictions3))

**Figure 13**

*Logistic Regression Confusion Matrix Results*



*Note: Performed in Jupyter Lab.*

from sklearn.tree import DecisionTreeClassifier

model3= DecisionTreeClassifier(criterion="gini")

#here we are facing the problem of overfitting

#train the model

model3.fit(X\_train, y\_train)

predictions4 = model3.predict(X\_test)

#Decision Tree Algorithm Confusion Matrix

from sklearn.metrics import classification\_report, confusion\_matrix, accuracy\_score

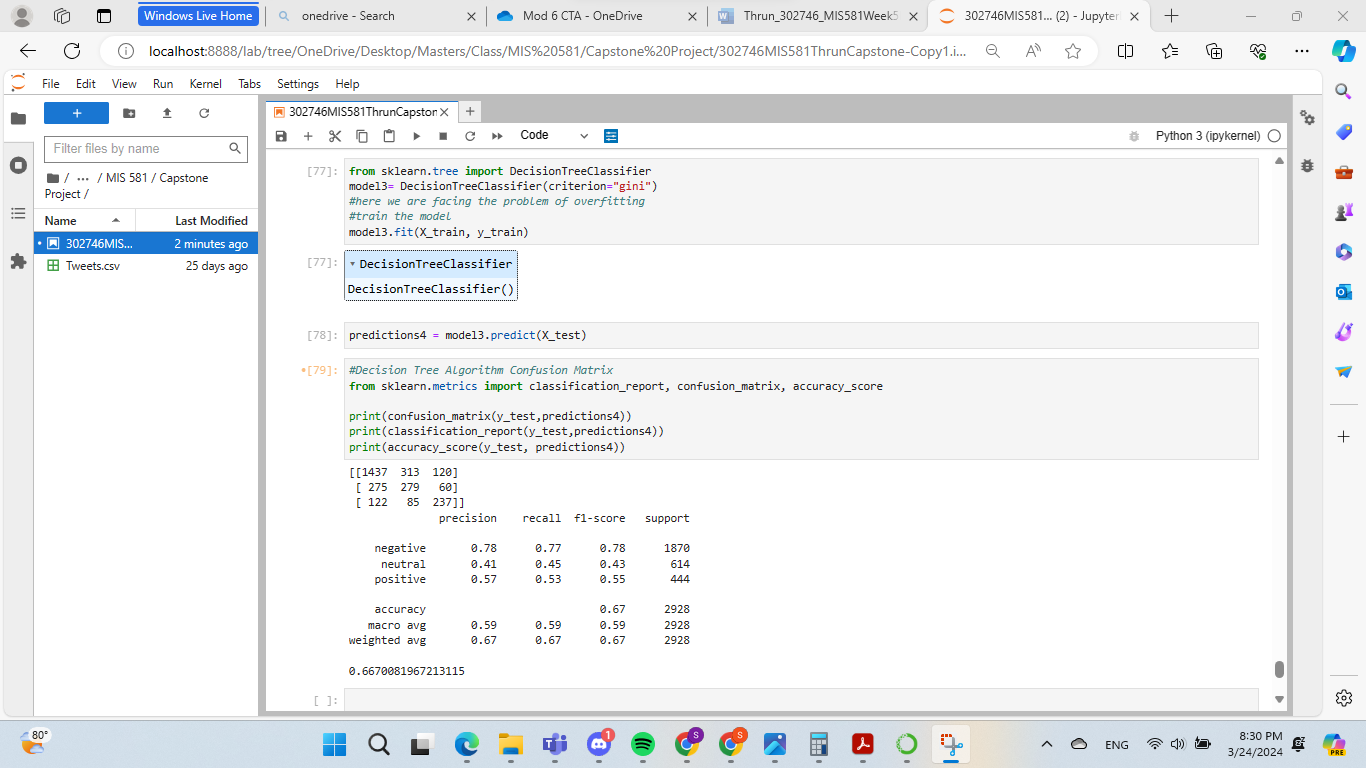
print(confusion\_matrix(y\_test,predictions4))

print(classification\_report(y\_test,predictions4))

print(accuracy\_score(y\_test, predictions4))

**Figure 14**

*Decision Tree Algorithm Confusion Matrix Results*



*Note: Performed in Jupyter Lab.*

**GitHub Link**

<https://github.com/ST5S918/MIS581SentimentAnalysisCapstone/tree/main>

**References**

Chai, W. (2022). Data Dictionary. *TechTarget.* Retrieved from: <https://www.techtarget.com/searchapparchitecture/definition/data-dictionary>

Cheng, Y., Liu, Y., Chen, T. & Yang, Q. (2020). Federated Learning for Privacy-Preserving AI: Engineering and algorithmic framework to ensure data privacy and user confidentiality. Communications of the ACM, 63(12), 33–36. <https://doi.org/10.1145/3387107>

D, N. S. B. K., P V G D, P. R., & Venkata Rao, K. (2023). Emotion recognition in election day tweets using optimised kernel extreme learning machine classifier. Journal of Experimental & Theoretical Artificial Intelligence, 35(2), 289–307. <https://doi.org/10.1080/0952813X.2021.1960633>

Figure Eight. (2020). Twitter US Airline Sentiment. Retrieved from: <https://www.kaggle.com/datasets/crowdflower/twitter-airline-sentiment>

Hossain, I., Puppala, S., Alam, M. J. and Talukder, S. (2024). Monitoring Dynamics of Emotional Sentiment in Social Network Commentaries. In Proceedings of the 2023 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM '23). Association for Computing Machinery, New York, NY, USA, 51–55. <https://doi-org.csuglobal.idm.oclc.org/10.1145/3625007.3627730>

Nagarajan, S. M., & Gandhi, U. D. (2019). Classifying streaming of Twitter data based on sentiment analysis using hybridization. Neural Computing & Applications, 31(5), 1425–1433. <https://doi.org/10.1007/s00521-018-3476-3>

Nannini, L., Balayn, A., and Smith, A. L. (2023). Explainability in AI Policies: A Critical Review of Communications, Reports, Regulations, and Standards in the EU, US, and UK. In Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency (FAccT '23). Association for Computing Machinery, New York, NY, USA, 1198–1212. https://doi-org.csuglobal.idm.oclc.org/10.1145/3593013.3594074

Polonsky, M. J., & Waller, D. S. (2019). Designing and managing a research project: A business student's guide (4th ed.). SAGE Publications. ISBN: 9781544316468

Sanjana, V. (2020). Sentimental Analysis using nlp- for begineers. Retrieved from: <https://www.kaggle.com/code/sanjanavoona1043/sentimental-analysis-using-nlp-for-begineers>

Taderhoost, H & Madanchian, M. (2023). Artificial Intelligence and Sentiment Analysis: A Review in Competitive Research. *Computers*. <https://doi.org/10.3390/computers12020037>

Webb, H. et al. (2017). The Ethical Challenges of Publishing Twitter Data for Research Dissemination. WebSci '17: Proceedings of the 2017 ACM on Web Science Conference. <https://doi.org/10.1145/3091478.3091489>

WhiteHouse. (2024). Executive Order on Preventing Access to Americans’ Bulk Sensitive Personal Data and United States Government-Related Data by Countries of Concern. Retrieved from: <https://www.whitehouse.gov/briefing-room/presidential-actions/2024/02/28/executive-order-on-preventing-access-to-americans-bulk-sensitive-personal-data-and-united-states-government-related-data-by-countries-of-concern/>