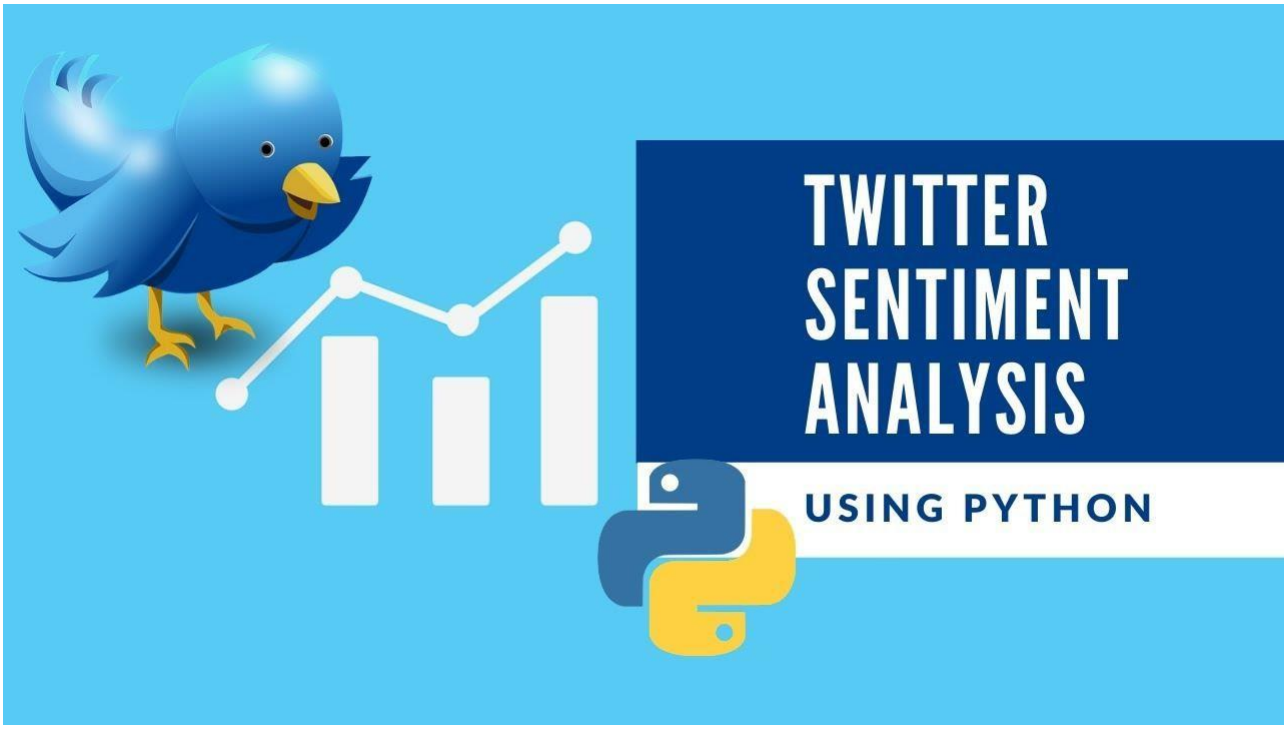


Table of contents:

- Sentiment Analysis Results
- Advanced Technique
- Business Insights
- Conclusion
- Acknowledgments

1. Executive Summary:

The Executive Summary provides a concise yet comprehensive overview of the entire project. It should include:

- This phase involves summarizing the project's journey, methodologies, and results, ensuring that no important detail is missed.
- It represents the closure of the analysis process and the beginning of the implementation of valuable insights in real-world business decisions.
- The project documentation serves as a lasting record, offering reference material for future work and presentations to stakeholders.

2. Introduction:

- The project focuses on sentiment analysis and its significance in understanding customer perceptions of competitor products.
- Understanding customer sentiments is vital for improving products and enhancing marketing strategies.
- This project utilizes various NLP methods to extract insights from customer feedback.
- It encompasses problem definition, data collection, preprocessing, advanced sentiment analysis, and visualization.
- The project aims to enhance sentiment analysis accuracy by exploring advanced techniques.
- The ultimate goal is to generate actionable business insights.

3. Problem Statement Revisited:

- The problem is to perform sentiment analysis on customer feedback to gain insights into competitor products."

4. Data Collection and Preprocessing:

- This section should provide detailed information on data collection and preprocessing, including:

Data Collection:

- The project begins with the essential step of data collection. A dataset containing customer reviews and sentiments about competitor products is identified for analysis.
- The provided dataset comes from a reputable source, such as Kaggle, and contains relevant data for sentiment analysis.

Data Preprocessing:

- Data preprocessing is a critical phase in the project, involving cleaning and transforming the textual data for analysis.
- Textual data is carefully cleaned to remove noise and irrelevant information, ensuring that the dataset is suitable for analysis.

5. Sentiment Analysis Techniques:

- The project utilizes various Natural Language Processing (NLP) methods, including Bag of Words (BoW), Word Embeddings, and Transformer models like BERT and RoBERTa.

- These techniques enable the analysis of customer sentiments and opinions expressed in the textual data.
- Specific NLP libraries and tools, such as NLTK, spaCy, and Hugging Face Transformers, are employed to implement these techniques effectively.

6. Feature Extraction:

- Feature extraction is a critical step in your project, where textual data is transformed into numerical representations suitable for sentiment analysis.
- This phase is responsible for converting raw text into structured features that can be analyzed effectively, facilitating the understanding of customer sentiments.
- The chosen techniques and algorithms for feature extraction play a pivotal role in revealing the underlying patterns and nuances within the data.

7. Sentiment Analysis Results:

- Sentiment analysis results form a core part of the project, presenting insights into customer feedback and opinions.
- These results showcase the distribution of sentiments, trends in customer responses, and key findings that shed light on the strengths and weaknesses of competitor products.
- The use of visualizations, such as charts and graphs, aids in presenting the sentiment analysis findings in a clear and accessible manner.
- The analysis results help in drawing actionable business insights, guiding marketing strategies, product improvements, and competition analysis.

8. Advanced Techniques:

- Advanced techniques represent a pivotal phase in your project, where fine-tuning pre-trained sentiment analysis models, such as BERT and RoBERTa, are investigated for improving prediction accuracy.
- These state-of-the-art models are leveraged to capture nuanced sentiments and contextual understanding within customer feedback.
- The application of advanced techniques signifies the project's commitment to enhancing the quality of sentiment predictions.

9. Business Insights:

Summarize the valuable insights generated from the sentiment analysis results:

- Business insights are the heart of the project, offering actionable guidance to shape marketing strategies and product development.
- These insights are distilled from the sentiment analysis results and highlight areas of improvement, customer preferences, and competitive strengths and weaknesses.
- They are a valuable asset for making data-driven decisions in the ever-evolving business landscape.

10. Conclusion:

- The conclusion marks the culmination of the project, summarizing the key findings and their implications for marketing and product development.
- It highlights the significance of sentiment analysis in the modern business landscape, emphasizing the value of understanding customer feedback.

- The insights obtained from the sentiment analysis results contribute to data-driven decision-making, providing a roadmap for improvement and competition analysis.
- The project underlines the power of NLP techniques in transforming raw textual data into actionable insights.
- The journey from problem definition to insight generation is a testament to the project's ability to enhance marketing strategies and guide businesses toward success in a competitive market.

11. Acknowledgments:

- Acknowledge individuals or organizations that provided support, data, or resources during the project:

- Express gratitude for their contributions or assistance.

Program:

[1]:
```
import pandas as pd
import numpy as np
# %load_ext nb_black

# library to suppress warnings or deprecation notes
import warnings

warnings.filterwarnings("ignore")


# import Regex, string and unicodedata.
import re, string, unicodedata

import contractions

# import BeautifulSoup.
from bs4 import BeautifulSoup

# import Natural Language Tool-Kit.
import nltk

# download Stopwords.
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('wordnet')

# import stopwords.
from nltk.corpus import stopwords

# import Tokenizer.
from nltk.tokenize import word_tokenize, sent_tokenize

# library to split data
from sklearn.model_selection import train_test_split, StratifiedKFold

# libaries to help with data visualization
import matplotlib.pyplot as plt
```

```
import seaborn as sns
import missingno as msno

# import wordcloud
import wordcloud
from wordcloud import STOPWORDS
from wordcloud import WordCloud

# remove the limit for the number of displayed columns
pd.set_option("display.max_columns", None)

# set the limit for the number of displayed rows
pd.set_option("display.max_rows", 200)

# to get diferent metric scores
from sklearn.metrics import (
    recall_score,
    accuracy_score,
    confusion_matrix,classification_report,
    f1_score,
    precision_score,
    precision_recall_fscore_support
)

# import vectorizers
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer

# import rfc and cross_val_score
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import cross_val_score

# import word prepocessors
from nltk.tokenize import word_tokenize
from nltk.stem import LancasterStemmer, WordNetLemmatizer
```

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

[2]:
```
df = pd.read_csv('Tweets.csv')
df.head()
```

[2]:

	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
0	570306133677760513	neutral	1.0000	NaN	NaN	Virgin America	NaN	cairdin	NaN	0	@VirginAmerica What @dhepburn said.	NaN	2015-02-24 11:35:52 -0800	NaN	Eastern Time (US & Canada)
1	570301130888122368	positive	0.3486	NaN	0.0000	Virgin America	NaN	jnardino	NaN	0	@VirginAmerica plus you've added commercials t...	NaN	2015-02-24 11:15:59 -0800	NaN	Pacific Time (US & Canada)
2	570301083672813571	neutral	0.6837	NaN	NaN	Virgin America	NaN	yvonnalynn	NaN	0	@VirginAmerica I didn't today... Must mean I n...	NaN	2015-02-24 11:15:48 -0800	Lets Play	Central Time (US & Canada)
3	570301031407624196	negative	1.0000	Bad Flight	0.7033	Virgin America	NaN	jnardino	NaN	0	@VirginAmerica it's really aggressive to blast...	NaN	2015-02-24 11:15:36 -0800	NaN	Pacific Time (US & Canada)
4	570300817074462722	negative	1.0000	Can't Tell	1.0000	Virgin America	NaN	jnardino	NaN	0	@VirginAmerica and it's a really big bad thing...	NaN	2015-02-24 11:14:45 -0800	NaN	Pacific Time (US & Canada)

[3]:
```
texts = [[word.lower() for word in text.split()] for text in df]
df.head()
```

[3]:

	tweet_id	airline_sentiment	airline_sentiment_confidence	negativereason	negativereason_confidence	airline
0	570306133677760513	neutral	1.0000	NaN	NaN	Virgin America
1	570301130888122368	positive	0.3486	NaN	0.0000	Virgin America
2	570301083672813571	neutral	0.6837	NaN	NaN	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

```
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)
```

[4] :
```
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
```

[5] :
```
df.isnull().sum()
```

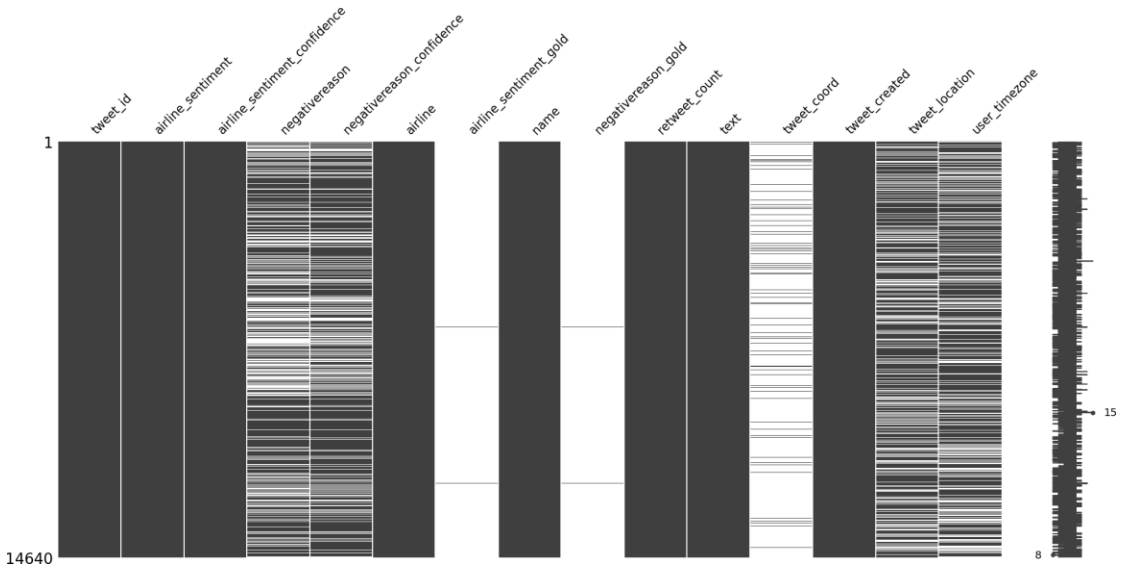
[5]:
```
tweet_id                          0
airline_sentiment                 0
airline_sentiment_confidence      0
negativereason                 5462
negativereason_confidence      4118
airline                           0
airline_sentiment_gold        14600
name                              0
negativereason_gold           14608
retweet_count                     0
text                              0
tweet_coord                   13621
tweet_created                     0
tweet_location                 4733
user_timezone                  4820
dtype: int64
```

[6] :
```
df.isnull().sum() / len(df) * 100
```

[6]:
```
tweet_id                       0.000000
airline_sentiment              0.000000
airline_sentiment_confidence   0.000000
negativereason                37.308743
negativereason_confidence     28.128415
airline                        0.000000
airline_sentiment_gold        99.726776
name                           0.000000
negativereason_gold           99.781421
retweet_count                  0.000000
text                           0.000000
tweet_coord                   93.039617
tweet_created                  0.000000
tweet_location                32.329235
user_timezone                 32.923497
dtype: float64
```

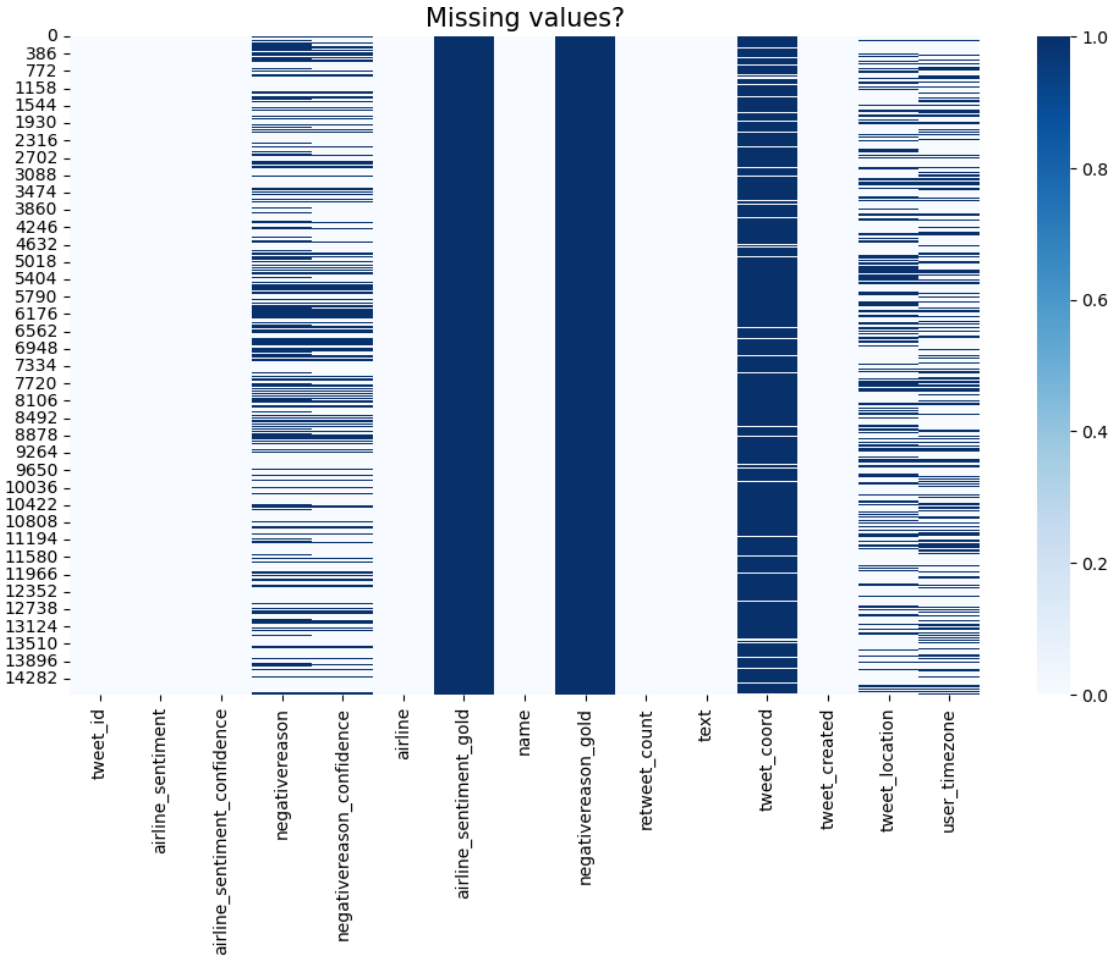
[7] :
```
msno.matrix(df)
```

[7] : <AxesSubplot:>



[8] :

```
plt.figure(figsize=(12,7))
sns.heatmap(df.isnull(), cmap = "Blues")                    #Visualization
  ↪of missing value using heatmap
plt.title("Missing values?", fontsize = 15)
plt.show()
```



[9] :
```
print("Percentage null or na values in df")
((df.isnull() | df.isna()).sum() * 100 / df.index.size).round(2)
```

Percentage null or na values in df

[9]:

tweet_id	0.00
airline_sentiment	0.00
airline_sentiment_confidence	0.00
negativereason	37.31
negativereason_confidence	28.13
airline	0.00
airline_sentiment_gold	99.73
name	0.00
negativereason_gold	99.78
retweet_count	0.00
text	0.00
tweet_coord	93.04

```
tweet_created                 0.00
tweet_location               32.33
user_timezone                32.92
dtype: float64
```

[10] :
```
del df["tweet_coord"]
del df["airline_sentiment_gold"]
del df["negativereason_gold"]
```

[11] :
```
df.head()
```

[11] :

	tweet_id	airline_sentiment	airline_sentiment_confidence	negativereason	negativereason_confidence	airline	name	retweet_count	text	tweet_created	tweet_location	user_timezone
0	570306133677760513	neutral	1.0000	NaN	NaN	Virgin America	cairdin	0	@VirginAmerica What @dhepburn said.	2015-02-24 11:35:52 -0800	NaN	Eastern Time (US & Canada)
1	570301130888122368	positive	0.3486	NaN	0.0000	Virgin America	jnardino	0	@VirginAmerica plus you've added commercials t...	2015-02-24 11:15:59 -0800	NaN	Pacific Time (US & Canada)
2	570301083672813571	neutral	0.6837	NaN	NaN	Virgin America	yvonnalynn	0	@VirginAmerica I didn't today... Must mean I n...	2015-02-24 11:15:48 -0800	Lets Play	Central Time (US & Canada)
3	570301031407624196	negative	1.0000	Bad Flight	0.7033	Virgin America	jnardino	0	@VirginAmerica it's really aggressive to blast...	2015-02-24 11:15:36 -0800	NaN	Pacific Time (US & Canada)
4	570300817074462722	negative	1.0000	Can't Tell	1.0000	Virgin America	jnardino	0	@VirginAmerica and it's a really big bad thing...	2015-02-24 11:14:45 -0800	NaN	Pacific Time (US & Canada)

[12] :
```
freq = df.groupby("negativereason").size()
```

[13] :
```
# Checking duplicates
df.duplicated().sum()
```

[13]: 39

[14] :
```
df.drop_duplicates(inplace = True)
df.duplicated().sum()
```

[14]: 0

[15] :
```
df.sample(n = 10)
```

[15]:

	tweet_id	airline_sentiment	airline_sentiment_confidence	negativereason	negativereason_confidence	airline	name	retweet_count	text
10589	569156425626329089	neutral	1.0000	NaN	NaN	US Airways	observepeople	0	@usairways Does anyone know the hold times for...
6182	568149878095753216	neutral	0.6545	NaN	0.0000	Southwest	Brian_Fox	0	@SouthwestAir I would but you need to follow m...
11336	568196165780578304	negative	1.0000	Can't Tell	0.3579	US Airways	thefisch26	0	@USAirways Secondary screenings, a piece of th...
623	570245555064074240	negative	1.0000	Flight Booking Problems	0.6740	United	fatwmnonthemtn	0	
1186	569902065247322112	negative	1.0000	Late Flight	1.0000	United	LukeXuanLiu	1	
2425	569213883371683840	positive	0.6679	NaN	NaN	United	PierreSchmit	0	
13299	569893723091238912	negative	1.0000	longlines	0.3512	American	elisakathleen	0	
7693	569343003476819969	neutral	0.6641	NaN	0.0000	Delta	dgruber1700	0	
5148	569308552671707136	negative	1.0000	Lost Luggage	1.0000	Southwest	scoobydoo9749	0	
11135	568486436355346432	negative	1.0000	Bad Flight	1.0000	US Airways	kristenlc	0	

```
623     @united What's going on with your website? I'm…
1186    @united and most frustratingly, all this delay…
2425    @united gave me a smile today, with a Zero Awa…
13299   @AmericanAir the most stressful morning and st…
7693                                        @JetBlue flite454
5148    @SouthwestAir 9 hrs in Baltimore, still not go…
11135   @USAirways we bought our tickets months ago. H…

                      tweet_created       tweet_location   \
10589  2015-02-21 07:27:20 -0800                  NaN
6182   2015-02-18 12:47:41 -0800   NH, United States
11336  2015-02-18 15:51:37 -0800      Washington, DC
623    2015-02-24 07:35:09 -0800          Summit, NJ
1186   2015-02-23 08:50:15 -0800                  NaN
2425   2015-02-21 11:15:39 -0800  Rixensart, Belgium
13299  2015-02-23 08:17:06 -0800          Boston, MA
7693   2015-02-21 19:48:44 -0800                  NaN
5148   2015-02-21 17:31:50 -0800     Tallahassee, FL
11135  2015-02-19 11:05:03 -0800                  NaN


                        user_timezone
10589 Eastern Time (US & Canada)
6182  Eastern Time (US & Canada)
11336 Central Time (US & Canada)
623   Central Time (US & Canada)
1186        Atlantic Time (Canada)
2425                      Brussels
13299                          NaN
7693                           NaN
5148             America/Chicago
11135 Eastern Time (US & Canada)
```

[16] :
```
df.describe().T
```

[16] :
```
                                count          mean           std   \
tweet_id                      14601.0  5.692156e+17  7.782706e+14
airline_sentiment_confidence  14601.0  8.999022e-01  1.629654e-01
negativereason_confidence     10501.0  6.375749e-01  3.303735e-01
retweet_count                 14601.0  8.280255e-02  7.467231e-01

                                     min           25%           50% \
tweet_id                    5.675883e+17  5.685581e+17  5.694720e+17
airline_sentiment_confidence 3.350000e-01 6.923000e-01 1.000000e+00
negativereason_confidence   0.000000e+00  3.605000e-01  6.705000e-01
retweet_count               0.000000e+00  0.000000e+00  0.000000e+00

                                     75%           max
```

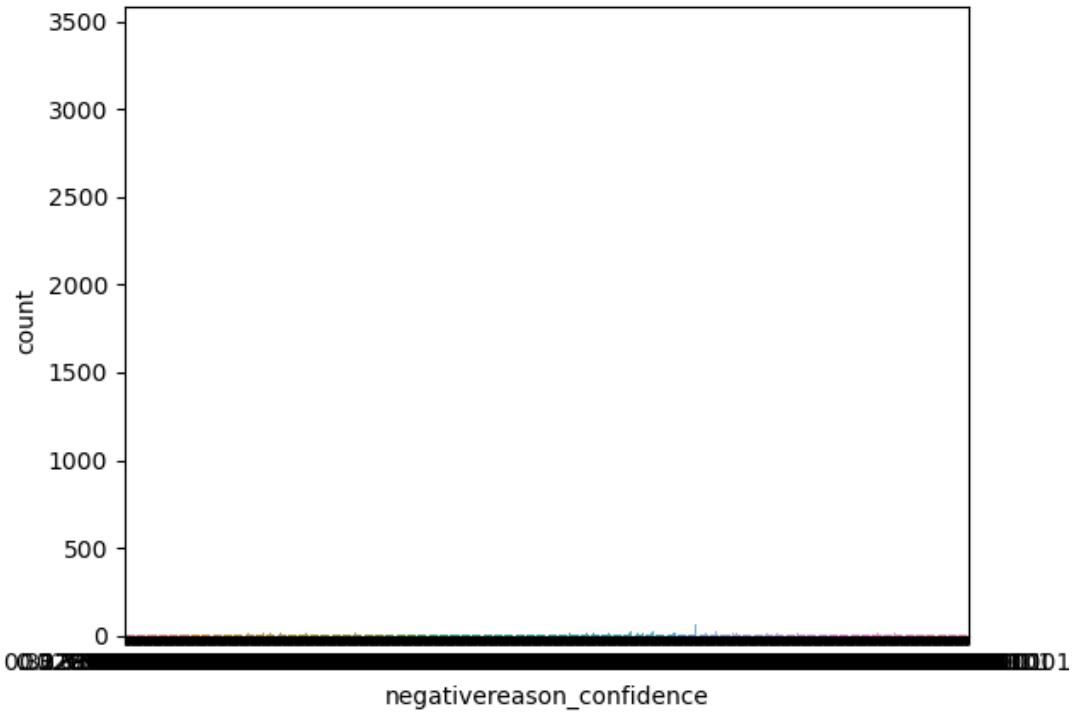
tweet_id	5.698884e+17	5.703106e+17
airline_sentiment_confidence	1.000000e+00	1.000000e+00
negativereason_confidence	1.000000e+00	1.000000e+00
retweet_count	0.000000e+00	4.400000e+01

[17] :
```
df.nunique()
```

[17]:
tweet_id	14485
airline_sentiment	3
airline_sentiment_confidence	1023
negativereason	10
negativereason_confidence	1410
airline	6
name	7701
retweet_count	18
text	14427
tweet_created	14247
tweet_location	3081
user_timezone	85

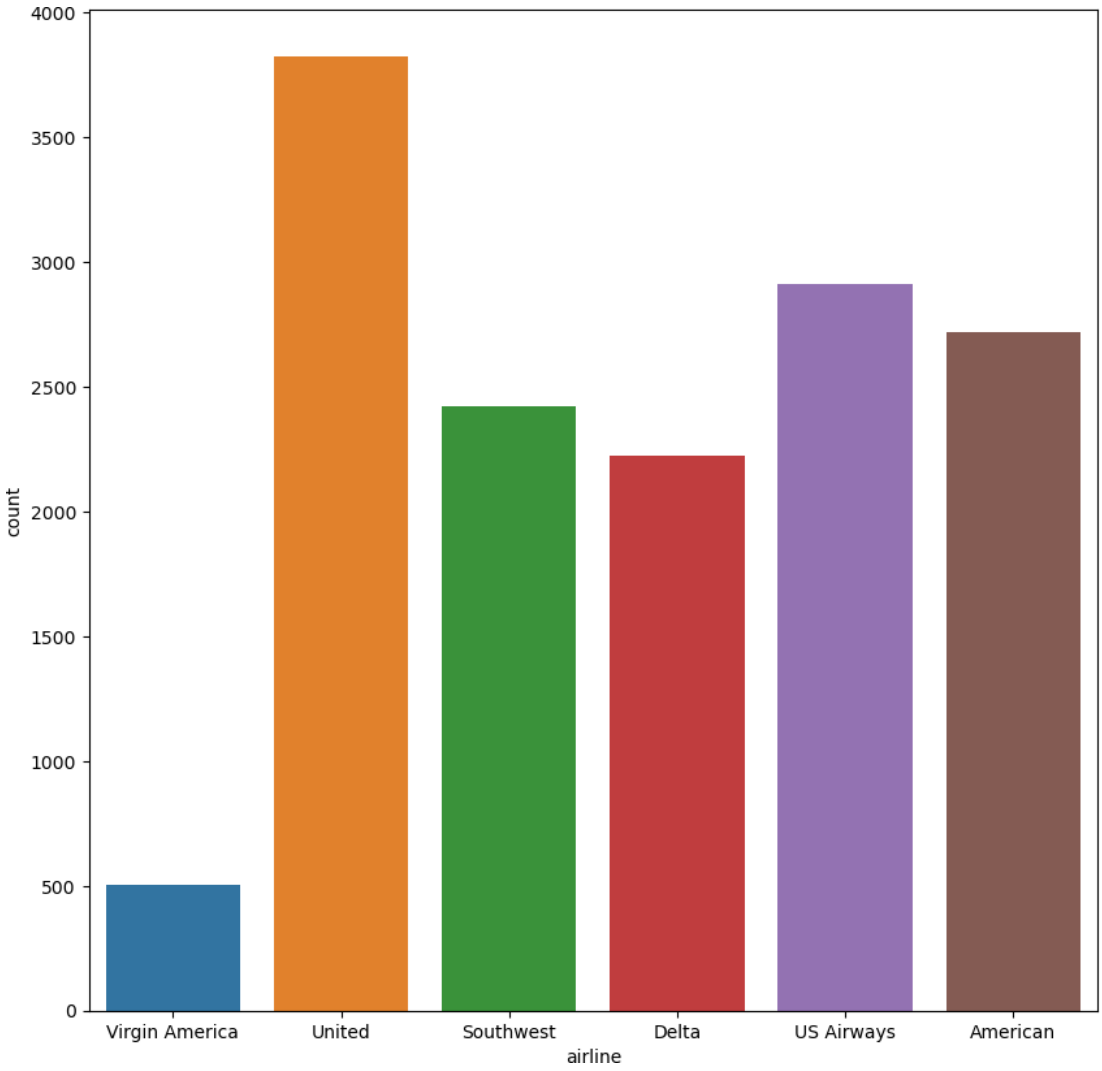
dtype: int64

[18] :
```
ax = sns.countplot(x = "negativereason_confidence", data = df)
```



[19] :

```
plt.figure(figsize = (10, 10))
ax = sns.countplot(x = "airline", data = df)
```



[20] :

```
import plotly.graph_objects as go
crosstab_sentiments=pd.crosstab(df.airline, df.negativereason)
companies=list(crosstab_sentiments.index)

fig = go.Figure(data=[
    go.Bar(name=col_name, x=companies, y=list(crosstab_sentiments[col_name]))
for col_name in list(crosstab_sentiments.columns)])#
Change the bar mode
fig.update_layout(barmode='stack',
                  title='Sentiment distribution per company',
                  yaxis=dict(title='Sentiment distribution'));
```

```
                  xaxis=dict(title="Companies"))
fig.show()
```

[21] :
```
crosstab_neg_reasons = pd.crosstab(df["airline"], df["negativereason"])
companies = list(crosstab_neg_reasons.index)

fig = go.Figure(data = [
    go.Bar(name = col_name, x = companies, y =␣
 ↪list(crosstab_neg_reasons[col_name]))
for col_name in list(crosstab_neg_reasons.columns)])

fig.update_layout(barmode = "stack",
                  title = "Negative Reasons Distribution per Company",
                  yaxis = dict(title = "Negative reasons Distribution"),
                  xaxis = dict(title = "Companies"))
fig.show()
```

[22] :
```
labels = list(crosstab_neg_reasons.columns)
values = [crosstab_neg_reasons[col_name].sum() for col_name in labels]

# Use `hole` to create a donut-like pie chart
fig = go.Figure(data=[go.Pie(labels=labels, values=values, hole=.3)])
fig.update_layout(title='Overall distribution for negative reasons')
fig.show()
```

[23] :
```
df.drop(df.loc[df["airline_sentiment"] == "neutral"].index, inplace = True)
```

[24] :
```
data = df[["airline_sentiment", "text"]]
data.head()
```

[24] :

	airline_sentiment	text
1	positive	@VirginAmerica plus you've added commercials t...
3	negative	@VirginAmerica it's really aggressive to blast...
4	negative	@VirginAmerica and it's a really big bad thing...
5	negative	@VirginAmerica seriously would pay $30 a fligh...
6	positive	@VirginAmerica yes, nearly every time I fly VX...

[25]:
```
X = df["text"]
y = df["airline_sentiment"]
X
```

[25] :
```
1     @VirginAmerica plus you've added commercials t...
3     @VirginAmerica it's really aggressive to blast...
4     @VirginAmerica and it's a really big bad thing...
5     @VirginAmerica seriously would pay $30 a fligh...
6     @VirginAmerica yes, nearly every time I fly VX...
...
```

```
14633    @AmericanAir my flight was Cancelled Flightled...
14634          @AmericanAir right on cue with the delays
14635    @AmericanAir thank you we got on a different f...
14636    @AmericanAir leaving over 20 minutes Late Flig...
14638    @AmericanAir you have my money, you change my ...
Name: text, Length: 11510, dtype: object
```

[26] :
```
y
```

[26] :
```
1          positive
3          negative
4          negative
5          negative
6          positive
             ...
14633      negative
14634      negative
14635      positive
14636      negative
14638      negative
Name: airline_sentiment, Length: 11510, dtype: object
```

[27] :
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,␣
↪random_state = 42)
print(X_train.shape, X_test.shape, y_train.shape, y_test.shape)
```

```
(9208,) (2302,) (9208,) (2302,)
```

[28] :
```
tfidf = TfidfVectorizer(stop_words="english")
```

[29] :
```
tfidf.fit(y_train)
```

[29] :
```
TfidfVectorizer(stop_words='english')
```

[30] :
```
print(tfidf.get_feature_names_out())
```

```
['negative' 'positive']
```

[31] :
```
print(tfidf.vocabulary_)
```

```
{'negative': 0, 'positive': 1}
```

[32] :
```
print(df)
```

```
                tweet_id airline_sentiment  airline_sentiment_confidence  \
1     570301130888122368          positive                        0.3486
3     570301031407624196          negative                        1.0000
4     570300817074462722          negative                        1.0000
```

```
5       570300767074181121         negative                       1.0000
6       570300616901320704         positive                       0.6745
...                    ...              ...                          ...
14633  569587705937600512         negative                       1.0000
14634  569587691626622976         negative                       0.6684
14635  569587686496825344         positive                       0.3487
14636  569587371693355008         negative                       1.0000
14638  569587188687634433         negative                       1.0000

                negativereason   negativereason_confidence         airline  \
1                          NaN                      0.0000  Virgin America
3                  Bad Flight                       0.7033  Virgin America
4                  Can't Tell                       1.0000  Virgin America
5                  Can't Tell                       0.6842  Virgin America
6                          NaN                      0.0000  Virgin America
...                        ...                         ...             ...
14633        Cancelled Flight                       1.0000        American
14634              Late Flight                      0.6684        American
14635                      NaN                      0.0000        American
14636  Customer Service Issue                       1.0000        American
14638  Customer Service Issue                       0.6659        American

                  name  retweet_count  \
1              jnardino              0
3              jnardino              0
4              jnardino              0
5              jnardino              0
6            cjmcginnis              0
...                 ...            ...
14633  RussellsWriting               0
14634   GolfWithWoody                0
14635  KristenReenders               0
14636          itsropes              0
14638       SraJackson               0

                                                    text  \
1      @VirginAmerica plus you've added commercials t...
3      @VirginAmerica it's really aggressive to blast...
4      @VirginAmerica and it's a really big bad thing...
5      @VirginAmerica seriously would pay $30 a fligh...
6      @VirginAmerica yes, nearly every time I fly VX...
...                                                  ...
14633  @AmericanAir my flight was Cancelled Flightled...
14634        @AmericanAir right on cue with the delays
14635  @AmericanAir thank you we got on a different f...
14636  @AmericanAir leaving over 20 minutes Late Flig...
14638  @AmericanAir you have my money, you change my ...
```

	tweet_created	tweet_location	user_timezone
1	2015-02-24 11:15:59 -0800	NaN	Pacific 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)
5	2015-02-24 11:14:33 -0800	NaN	Pacific Time (US & Canada)
6	2015-02-24 11:13:57 -0800	San Francisco CA	Pacific Time (US & Canada)
...	...	...	...
14633	2015-02-22 12:01:06 -0800	Los Angeles	Arizona
14634	2015-02-22 12:01:02 -0800	NaN	Quito
14635	2015-02-22 12:01:01 -0800	NaN	NaN
14636	2015-02-22 11:59:46 -0800	Texas	NaN
14638	2015-02-22 11:59:02 -0800	New Jersey	Eastern Time (US & Canada)

[11510 rows x 12 columns]

[33] :
```
data[data["airline_sentiment"]    ==    "negative"]["text"]
```

[33]:
```
3          @VirginAmerica it's really aggressive to blast...
4          @VirginAmerica and it's a really big bad thing...
5          @VirginAmerica seriously would pay $30 a fligh...
15             @VirginAmerica SFO-PDX schedule is still MIA.
17         @VirginAmerica I flew from NYC to SFO last we...
                                  ...
14631      @AmericanAir thx for nothing on getting us out...
14633      @AmericanAir my flight was Cancelled Flightled...
14634              @AmericanAir right on cue with the delays
14636      @AmericanAir leaving over 20 minutes Late Flig...
14638      @AmericanAir you have my money, you change my ...
Name: text, Length: 9157, dtype: object
```

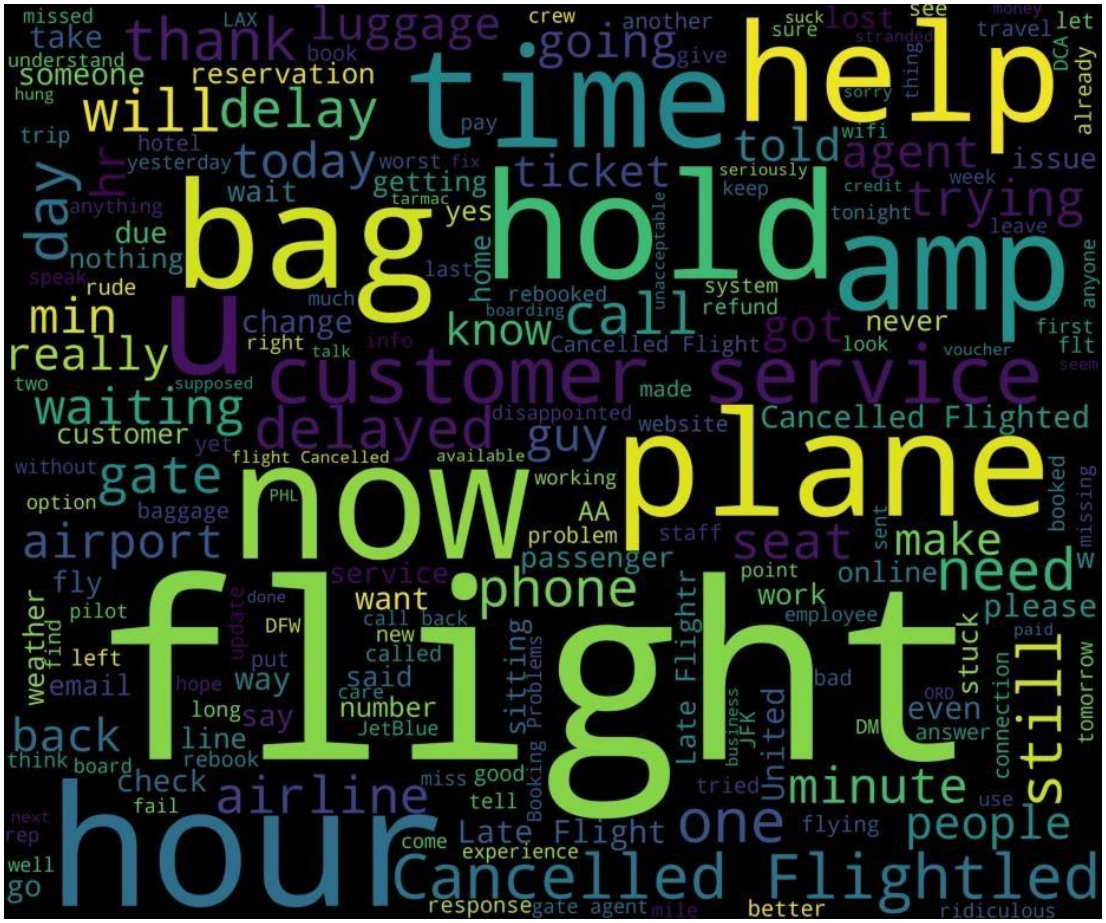
[34] :
```
count_vect = CountVectorizer(stop_words="english")
neg_matrix = count_vect.
  ↪fit_transform(data[data["airline_sentiment"]=="negative"]["text"])
freqs = zip(count_vect.get_feature_names_out(), neg_matrix.sum(axis=0).
  ↪tolist()[0])
# Sort from largest to smallest
print(sorted(freqs, key=lambda x: -x[1])[:100])
```

```
[('flight', 2937), ('united', 2899), ('usairways', 2375), ('americanair', 2089),
('southwestair', 1214), ('jetblue', 1051), ('cancelled', 921), ('service', 746),
('hours', 646), ('just', 622), ('help', 618), ('hold', 611), ('customer', 609),
('time', 596), ('plane', 530), ('delayed', 505), ('amp', 503), ('hour', 452),
('flightled', 445), ('http', 436), ('flights', 419), ('bag', 415), ('gate',
410), ('ve', 398), ('don', 388), ('late', 377), ('need', 373), ('phone', 367),
('waiting', 341), ('thanks', 315), ('got', 298), ('airline', 294), ('like',
291), ('trying', 288), ('delay', 272), ('wait', 272), ('today', 269),
('minutes', 266), ('day', 251), ('going', 249), ('bags', 245), ('luggage', 245),
('told', 245), ('airport', 244), ('people', 242), ('worst', 241), ('fly', 237),
```

('really', 236), ('did', 227), ('guys', 224), ('weather', 224), ('lost', 221), ('agent', 218), ('hrs', 217), ('way', 212), ('make', 211), ('change', 210), ('seat', 208), ('flighted', 205), ('want', 205), ('check', 204), ('know', 201), ('days', 200), ('home', 194), ('virginamerica', 191), ('baggage', 190), ('getting', 181), ('sitting', 179), ('ticket', 176), ('tomorrow', 176), ('let', 174), ('min', 171), ('customers', 169), ('flying', 168), ('line', 164), ('email', 163), ('online', 163), ('experience', 162), ('didn', 161), ('stuck', 160), ('work', 159), ('bad', 157), ('number', 156), ('won', 156), ('said', 155), ('seats', 154), ('30', 153), ('10', 150), ('problems', 150), ('times', 150), ('crew', 149), ('flightr', 148), ('doesn', 146), ('good', 145), ('ll', 144), ('aa', 143), ('travel', 142), ('yes', 142), ('response', 139), ('miss', 137)]

[35] :
```
new_df = data[data["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(figsize = (12, 12))
plt.imshow(wordcloud)
plt.axis("off")
plt.show()
```

[36] :
```
new_df = data[data["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(figsize = (12, 12))
plt.imshow(wordcloud)
plt.axis("off")
plt.show()
```



[37] :
```
data.drop(data.loc[data["airline_sentiment"] == "neutral"].index, inplace =␣
↪True)
```

[38] :
```
from sklearn.preprocessing import LabelEncoderle
= LabelEncoder()

le.fit(data["airline_sentiment"])
data["airline_sentiment_encoded"] = le.transform(data["airline_sentiment"])
data.head()
```

[38] :

	airline_sentiment	text
1	positive	@VirginAmerica plus you've added commercials t...
3	negative	@VirginAmerica it's really aggressive to blast...
4	negative	@VirginAmerica and it's a really big bad thing...
5	negative	@VirginAmerica seriously would pay $30 a fligh...
6	positive	@VirginAmerica yes, nearly every time I fly VX...

	airline_sentiment_encoded

1	1
3	0
4	0
5	0
6	1

[39]:
```
def tweet_to_words(tweet):
    letters_only = re.sub("[^a-zA-Z]", " ", tweet)
    words = letters_only.lower().split()
    stops = set(stopwords.words("english"))
    meaningful_words = [w for w in words if not w in stops]
    return(" ".join( meaningful_words ))
```

[40]:
```
nltk.download("stopwords")
data["clean_tweet"] = data["text"].apply(lambda x: tweet_to_words(x))
```

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

[41]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 11510 entries, 1 to 14638
Data columns (total 4 columns):
 #   Column                     Non-Null Count  Dtype 
---  ------                     --------------  ----- 
 0   airline_sentiment          11510 non-null  object
 1   text                       11510 non-null  object
 2   airline_sentiment_encoded  11510 non-null  int32 
 3   clean_tweet                11510 non-null  object
dtypes: int32(1), object(3)
memory usage: 404.6+ KB
```

[42]:
```
X = data["clean_tweet"]
y = data["airline_sentiment"]
```

[43]:
```
print(X.shape, y.shape)
```

```
(11510,) (11510,)
```

[44]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state = 42)
print(X_train.shape, X_test.shape, y_train.shape, y_test.shape)
```

```
(8632,) (2878,) (8632,) (2878,)
```

[45]:
```
vect = CountVectorizer()
vect.fit(X_train)
```

[45] : CountVectorizer()

[46] :
```
X_train_dtm = vect.transform(X_train)
X_test_dtm = vect.transform(X_test)
```

[47] :
```
vect_tunned = CountVectorizer(stop_words = "english", ngram_range = (1, 2),
    min_df = 0.1, max_df = 0.7, max_features = 100)
vect_tunned
```

[47]: CountVectorizer(max_df=0.7, max_features=100, min_df=0.1, ngram_range=(1, 2),
stop_words='english')

[48]:
```
from sklearn.svm import SVC
model = SVC(kernel = "linear", random_state = 10)
model.fit(X_train_dtm, y_train)
pred = model.predict(X_test_dtm)
print("Accuracy Score: ", accuracy_score(y_test, pred) * 100)
```

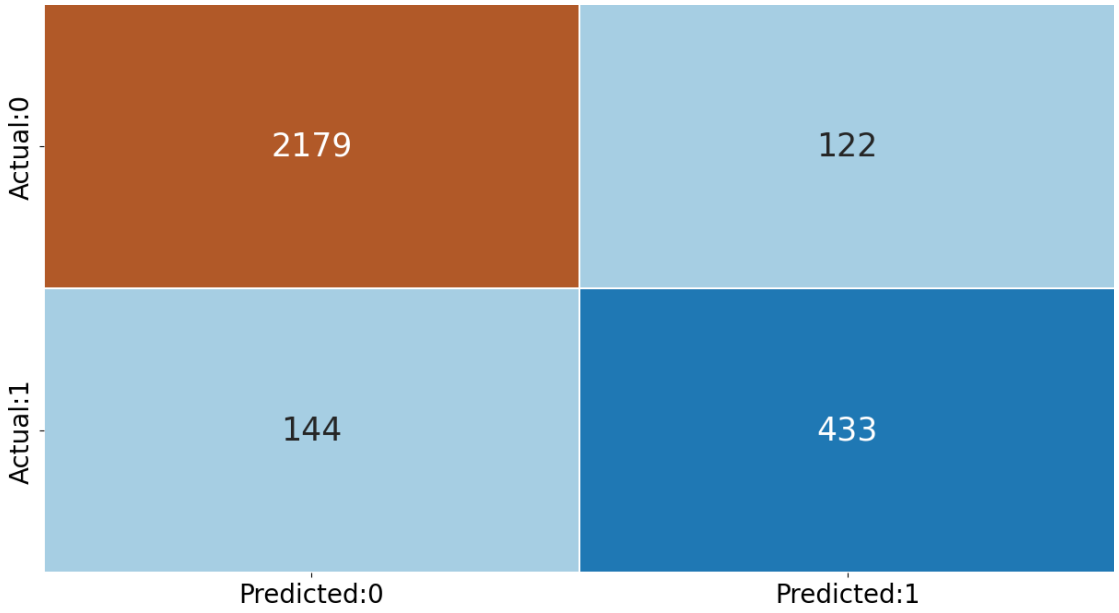
Accuracy Score: 90.7574704656011

[49]:
```
print("Confusion Matrix\n\n", confusion_matrix(y_test, pred))
```

Confusion Matrix

[[2179 122]
[144 433]]

[50]:
```
#defining the size of the canvas
plt.rcParams['figure.figsize'] = [15,8]
#confusion matrix to DataFrame
conf_matrix = pd.DataFrame(data = confusion_matrix(y_test, pred),columns =
    ['Predicted:0','Predicted:1',], index = ['Actual:0','Actual:1',])
#plotting the confusion matrix
sns.heatmap(conf_matrix, annot = True, fmt = 'd', cmap = 'Paired', cbar =
    False,linewidths = 0.1, annot_kws = {'size':25})
plt.xticks(fontsize = 20)
plt.yticks(fontsize = 20)
plt.show()
```



[51]:
```
print(classification_report(y_test, pred))
```

	precision	recall	f1-score	support
negative	0.94	0.95	0.94	2301
positive	0.78	0.75	0.77	577
accuracy			0.91	2878
macro avg	0.86	0.85	0.85	2878
weighted avg	0.91	0.91	0.91	2878

[ ]: