Notebook Gemini

```
import pandas as pd
import re
import nltk
from sklearn.feature_extraction.text import TfidfVectorizer

# Load the dataset
df = pd.read_csv('/content/Tweets.csv.zip')

print("Libraries imported and dataset loaded successfully.")
print(df.head())
```

```
Libraries imported and dataset loaded successfully.
             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
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)
```

```
print("First 5 rows of the DataFrame:")
print(df.head())

print("\nDataFrame Info (column names, data types, non-null values):")
df.info()

print("\nMissing values per column:")
print(df.isnull().sum())
```

```
First 5 rows of the DataFrame:
             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
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)

DataFrame Info (column names, data types, non-null values):
<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 
```

```
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

# Download necessary NLTK data (only needs to be run once)
nltk.download('stopwords', quiet=True)
nltk.download('punkt', quiet=True)

print("NLTK stopwords and punkt tokenizer are ready.")
```

```
NLTK stopwords and punkt tokenizer are ready.
```

```
import nltk

# Download the 'punkt_tab' resource which was indicated as missing
nltk.download('punkt_tab', quiet=True)

print("NLTK 'punkt_tab' resource downloaded successfully.")
```

```
NLTK 'punkt_tab' resource downloaded successfully.
```

```
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

# Initialize stopwords set for efficient lookup
stop_words = set(stopwords.words('english'))

def preprocess_text(text):
    # a. Remove URLs
    text = re.sub(r'http\S+|www\S+|https\S+', '', text, flags=re.MULTILINE)
    # b. Remove user mentions
    text = re.sub(r'@\w+', '', text)
    # c. Remove hashtags
    text = re.sub(r'#\w+', '', text)
    # d. Remove punctuation
    text = re.sub(r'[\W_]+', ' ', text) # Replaced with space to avoid merging words
    # e. Remove numbers
    text = re.sub(r'\d+', '', text)
    # f. Convert to lowercase
    text = text.lower()
    # g. Tokenize the text
    tokens = word_tokenize(text)
    # h. Remove English stopwords
    filtered_tokens = [word for word in tokens if word.isalnum() and word not in stop_words]

    return filtered_tokens

# Apply the preprocessing function to the 'text' column
df['processed_text'] = df['text'].apply(preprocess_text)

print("Tweet text preprocessing complete. Displaying original and processed text.")
print(df[['text', 'processed_text']].head())
```

```
Tweet text preprocessing complete. Displaying original and processed text.
                                                 text  \
0                @VirginAmerica What @dhepburn said.
1  @VirginAmerica plus you've added commercials t...
2  @VirginAmerica I didn't today... Must mean I n...
3  @VirginAmerica it's really aggressive to blast...
4  @VirginAmerica and it's a really big bad thing...

                                      processed_text
0                                             [said]
1      [plus, added, commercials, experience, tacky]
2     [today, must, mean, need, take, another, trip]
3  [really, aggressive, blast, obnoxious, enterta...
4                           [really, big, bad, thing]
```

```
from sklearn.feature_extraction.text import TfidfVectorizer

# Join the lists of tokens back into strings
df['processed_text_str'] = df['processed_text'].apply(lambda x: ' '.join(x))

# Initialize TfidfVectorizer
tfidf_vectorizer = TfidfVectorizer(tokenizer=lambda x: x.split(), lowercase=False)

# Fit and transform the processed text
tfidf_features = tfidf_vectorizer.fit_transform(df['processed_text_str'])

# Get feature names
feature_names = tfidf_vectorizer.get_feature_names_out()

print("TF-IDF Vectorization complete.")
print(f"Shape of TF-IDF features: {tfidf_features.shape}")
print(f"Total number of features: {len(feature_names)}")
print("First 10 feature names:")
print(feature_names[:10])
```

```
/usr/local/lib/python3.12/dist-packages/sklearn/feature_extraction/text.py:517: UserWarning: The parameter 'token_pattern' will
  warnings.warn(
TF-IDF Vectorization complete.
Shape of TF-IDF features: (14640, 9722)
Total number of features: 9722
First 10 feature names:
['aa' 'aaaand' 'aadavantage' 'aadv' 'aadvantage' 'aal' 'aaron' 'aaso' 'ab'
 'aback']
```

```
from sklearn.feature_extraction.text import TfidfVectorizer

# Join the lists of tokens back into strings
df['processed_text_str'] = df['processed_text'].apply(lambda x: ' '.join(x))

# Initialize TfidfVectorizer
tfidf_vectorizer = TfidfVectorizer(tokenizer=lambda x: x.split(), lowercase=False)

# Fit and transform the processed text
tfidf_features = tfidf_vectorizer.fit_transform(df['processed_text_str'])

# Get feature names
feature_names = tfidf_vectorizer.get_feature_names_out()

print("TF-IDF Vectorization complete.")
print(f"Shape of TF-IDF features: {tfidf_features.shape}")
print(f"Total number of features: {len(feature_names)}")
print("First 10 feature names:")
print(feature_names[:10])
```

```
/usr/local/lib/python3.12/dist-packages/sklearn/feature_extraction/text.py:517: UserWarning: The parameter 'token_pattern' will
  warnings.warn(
TF-IDF Vectorization complete.
Shape of TF-IDF features: (14640, 9722)
Total number of features: 9722
First 10 feature names:
['aa' 'aaaand' 'aadavantage' 'aadv' 'aadvantage' 'aal' 'aaron' 'aaso' 'ab'
 'aback']
```

```
import pandas as pd

# 1. Filter the DataFrame for negative sentiment tweets
negative_tweets_df = df[df['airline_sentiment'] == 'negative']

print(f"Number of negative sentiment tweets: {len(negative_tweets_df)}")

# 2. Extract the processed text strings for negative tweets
negative_text_data = negative_tweets_df['processed_text_str']

# 3. Transform the negative sentiment text using the already fitted TF-IDF vectorizer
tfidf_negative_features = tfidf_vectorizer.transform(negative_text_data)

# 4. Display the shape of the resulting TF-IDF matrix for negative tweets
print("\nTF-IDF vectorization for negative sentiment tweets complete.")
print(f"Shape of TF-IDF features for negative tweets: {tfidf_negative_features.shape}")

# Optionally, display some top terms for negative sentiment
# This part is illustrative and can be expanded if deeper analysis is needed
# For simplicity, we'll just show the shape for now as per the task requirements.
```

```
Number of negative sentiment tweets: 9178

TF-IDF vectorization for negative sentiment tweets complete.
Shape of TF-IDF features for negative tweets: (9178, 9722)
```

```
Start coding or generate with AI.
```

```
print("Sample of processed_text_str column:")
print(df['processed_text_str'].head())
```

```
Sample of processed_text_str column:
0                                                said
1              plus added commercials experience tacky
2               today must mean need take another trip
3    really aggressive blast obnoxious entertainmen...
4                                   really big bad thing
Name: processed_text_str, dtype: object
```

```
print("\nShape of TF-IDF features for the entire corpus:")
print(tfidf_features.shape)

print("\nShape of TF-IDF features for negative sentiment tweets:")
print(tfidf_negative_features.shape)
```

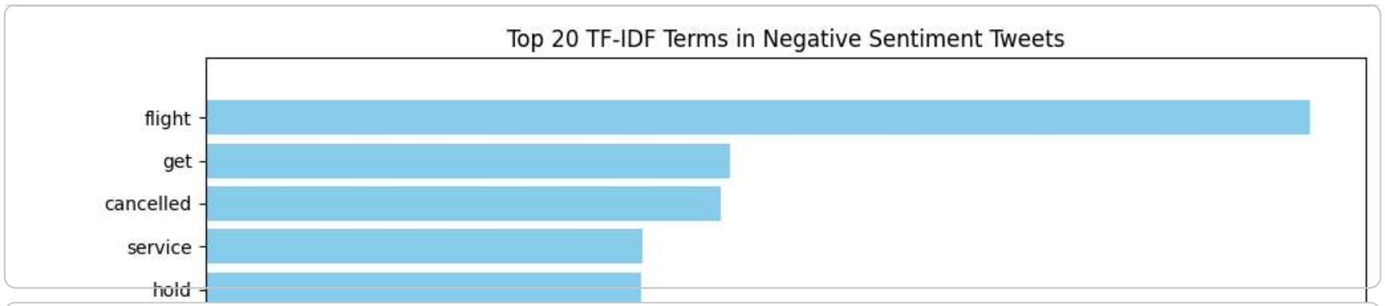
```

Shape of TF-IDF features for the entire corpus:
(14640, 9722)

Shape of TF-IDF features for negative sentiment tweets:
(9178, 9722)
```

```
import pandas as pd
import numpy as np

# 1. Sum the TF-IDF scores for each feature across all negative sentiment tweets
# Summing along axis 0 of the sparse matrix and converting to a dense array
sum_tfidf_scores = np.array(tfidf_negative_features.sum(axis=0))[0]

# 2. Create a Pandas Series mapping feature names to summed TF-IDF scores
top_terms_series = pd.Series(sum_tfidf_scores, index=feature_names)

# 3. Sort these terms by their summed TF-IDF scores in descending order
sorted_top_terms = top_terms_series.sort_values(ascending=False)
```

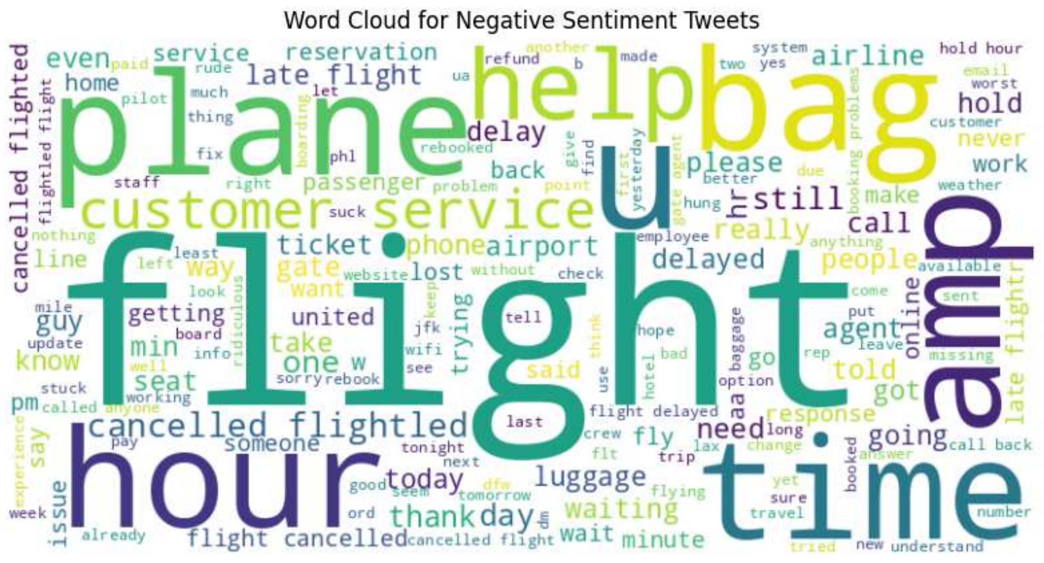
```
# 4. Display the top N terms (e.g., top 20) along with their scores
print("\nTop 20 TF-IDF terms for negative sentiment tweets:")
print(sorted_top_terms.head(20))
```

```

Top 20 TF-IDF terms for negative sentiment tweets:
flight       390.487967
get          185.293127
cancelled    182.218991
service      154.537677
hold         154.142449
hours        153.129098
customer     134.044460
help         130.878358
time         124.018919
delayed      121.466794
still        118.496188
hour         116.539998
plane        115.765994
call         111.372964
flightled    109.498591
us           103.616676
one          100.299923
bag           95.901635
amp           93.879998
gate          93.072301
dtype: float64
```

```
import matplotlib.pyplot as plt

# Select the top 20 terms for visualization
top_20_terms = sorted_top_terms.head(20)

# Create a horizontal bar chart
plt.figure(figsize=(10, 8))
plt.barh(top_20_terms.index[::-1], top_20_terms.values[::-1], color='skyblue')

# Add labels and title
plt.xlabel('Cumulative TF-IDF Score')
plt.ylabel('Terms')
plt.title('Top 20 TF-IDF Terms in Negative Sentiment Tweets')

# Adjust layout to prevent labels from overlapping
plt.tight_layout()

# Display the plot
plt.show()
```



```
from wordcloud import WordCloud
import matplotlib.pyplot as plt

# Filter the DataFrame for negative sentiment tweets
negative_tweets_text = df[df['airline_sentiment'] == 'negative']['processed_text_str']

# Concatenate all the text from negative sentiment tweets into a single string
combined_negative_text = ' '.join(negative_tweets_text)

# Instantiate a WordCloud object
wordcloud = WordCloud(width=800, height=400, background_color='white', min_font_size=10).generate(combined_negative_text)

# Display the generated image:
plt.figure(figsize=(10, 5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title('Word Cloud for Negative Sentiment Tweets')
plt.show()
```


Word Cloud for Negative Sentiment Tweets
flight
plane
help
bag
hour
time
amp
customer service
u