```
"""
This cell imports all required libraries for:
- Data loading and manipulation
- Text preprocessing
- TF-IDF feature extraction
- Visualization
"""

import pandas as pd               # For data handling
import re                         # For regular expressions
import nltk                       # For NLP preprocessing
import matplotlib.pyplot as plt   # For plotting graphs

from sklearn.feature_extraction.text import TfidfVectorizer  # TF-IDF
from nltk.corpus import stopwords                             # Stopwords
from nltk.tokenize import word_tokenize                       # Tokenization
from wordcloud import WordCloud                               # Word cloud
```

```
"""
Download NLTK resources needed for:
- Tokenization
- Stopword removal
"""

nltk.download('punkt')
nltk.download('stopwords')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
True
```

```
"""
Load the Twitter US Airline Sentiment dataset from CSV file
and display the first few records.
"""

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

# Display first 5 rows
df.head()
```

	tweet_id	airline_sentiment	airline_sentiment_confidence	negativereason	nega
0	570306133677760513	neutral	1.0000	NaN	
1	570301130888122368	positive	0.3486	NaN	

```
"""
Select only the required columns:
- text: tweet content
- airline_sentiment: sentiment label
"""

df = df[['text', 'airline_sentiment']]

# Check for missing values
df.isnull().sum()
```

Next steps: Generate code with df | New interactive sheet

	0
text	0
airline_sentiment	0

dtype: int64

```
"""
This function preprocesses tweet text by:
1. Converting text to lowercase
2. Removing URLs
3. Removing mentions (@user)
4. Removing hashtags
5. Removing special characters
6. Tokenizing text
7. Removing stopwords
"""

# Load English stopwords
stop_words = set(stopwords.words('english'))

def clean_tweet(text):
    """
    Clean and preprocess a tweet.

    Parameters:
    text (str): Original tweet text

    Returns:
    str: Cleaned tweet text
    """

    # Convert text to lowercase
    text = text.lower()

    # Remove URLs
    text = re.sub(r'http\S+|www\S+', '', text)
```

```
    # Remove mentions
    text = re.sub(r'@\w+', '', text)

    # Remove hashtags
    text = re.sub(r'#\w+', '', text)

    # Remove special characters and numbers
    text = re.sub(r'[^a-z\s]', '', text)

    # Tokenize text into words
    tokens = word_tokenize(text)

    # Remove stopwords
    tokens = [word for word in tokens if word not in stop_words]

    # Join tokens back into a sentence
    return ' '.join(tokens)
```

```
"""
Apply the clean_tweet() function to all tweets
and store the cleaned text in a new column.
"""

df['clean_text'] = df['text'].apply(clean_tweet)

# Display cleaned tweets
df.head()
```

Next steps: Generate code with df New interactive sheet

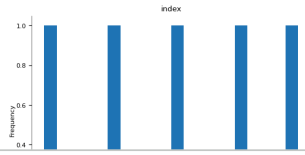
1 to 5 of 5 entries Filter

index	text	airline_sentiment	clean_text
0	@VirginAmerica What @dhepburn said.	neutral	said
1	@VirginAmerica plus you've added commercials to the experience... tacky.	positive	plus youve added commercials experience tacky
2	@VirginAmerica I didn't today... Must mean I need to take another trip!	neutral	didnt today must mean need take another trip
3	@VirginAmerica it's really aggressive to blast obnoxious "entertainment" in your guests' faces & they have little recourse	negative	really aggressive blast obnoxious entertainment guests faces amp little recourse
4	@VirginAmerica and it's a really big bad thing about it	negative	really big bad thing

Show 25 per page

Like what you see? Visit the data table notebook to learn more about interactive tables.

Distributions




```
"""
Filter the dataset to include only tweets
with negative sentiment.
"""

# Select negative sentiment tweets
negative_df = df[df['airline_sentiment'] == 'negative']

# Extract cleaned text
negative_text = negative_df['clean_text']

# Display number of negative tweets
len(negative_text)
```

```
9178
```

Values

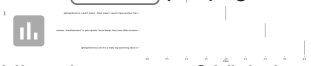
```
"""
Convert negative sentiment tweets into
TF-IDF feature vectors.
"""

# Initialize TF-IDF Vectorizer
tfidf_vectorizer = TfidfVectorizer(max_features=1000)

# Fit and transform the cleaned negative tweets
tfidf_matrix = tfidf_vectorizer.fit_transform(negative_text)

# Convert matrix to DataFrame
tfidf_df = pd.DataFrame(
    tfidf_matrix.toarray(),
    columns=tfidf_vectorizer.get_feature_names_out()
)

# Display TF-IDF matrix
tfidf_df.head()
```

1 to 5 of 5 entries Filter

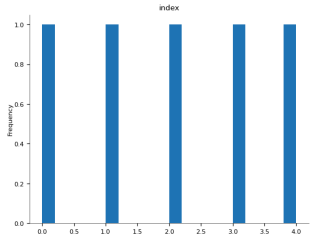
index	aa	able	absolute	absolutely	acceptable	access	accommodate	account	actual	actually	add	a
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

Show 25 per page

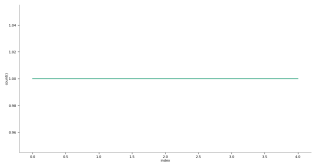
Like what you see? Visit the data table notebook to learn more about interactive tables.

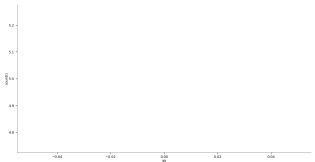


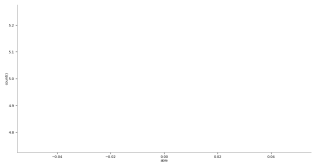
Distributions

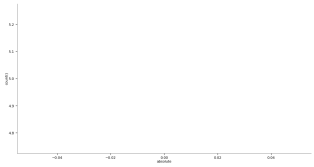


Time series





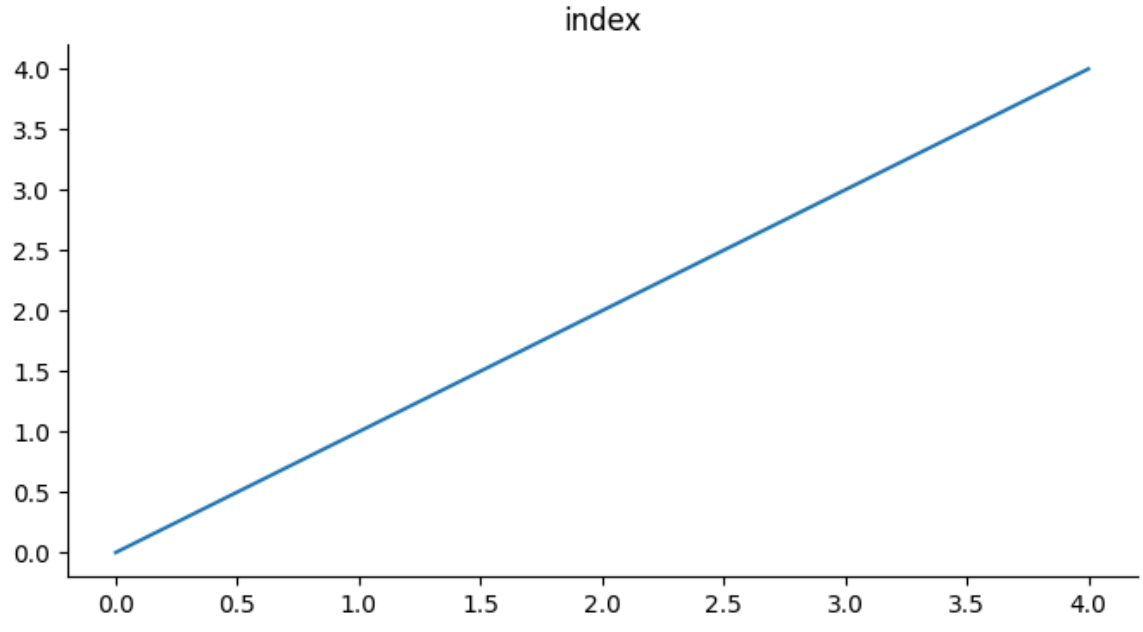




Values

```
<string>:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assi
Warning: Total number of columns (1000) exceeds max_columns (20) limiting to first (20) colu

<string>:5: FutureWarning:

ıt assigning `hue` is deprecated and will be removed in v0.14.0. Assi
```

```
from matplotlib import pyplot as plt
_df_18['index'].plot(kind='line', figsize=(8, 4), title='index')
plt.gca().spines[['top', 'right']].set_visible(False)
```

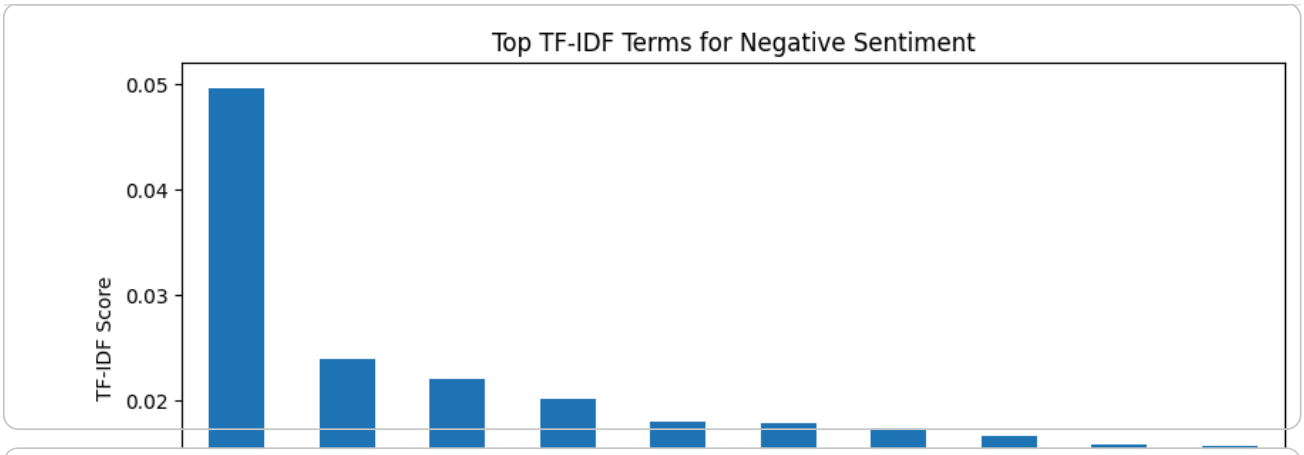


```
"""
Identify the most important words by
calculating average TF-IDF score for each term.
"""

# Calculate mean TF-IDF score for each term
tfidf_scores = tfidf_df.mean()

# Sort scores in descending order
tfidf_scores = tfidf_scores.sort_values(ascending=False)

# Select top 10 important terms
top_terms = tfidf_scores.head(10)

top_terms
```

	0
flight	0.049523
get	0.023843
cancelled	0.021949
service	0.020090
hours	0.017980
hold	0.017870
customer	0.017225
help	0.016579
time	0.015838
im	0.015664

dtype: float64

```
"""
Visualize the top TF-IDF terms for
negative sentiment using a bar chart.
"""

plt.figure(figsize=(10, 5))
top_terms.plot(kind='bar')
plt.title('Top TF-IDF Terms for Negative Sentiment')
plt.xlabel('Terms')
plt.ylabel('TF-IDF Score')
plt.xticks(rotation=45)
plt.show()
```




```
"""
Generate a word cloud directly from
cleaned negative sentiment tweets.
"""

def generate_text_wordcloud(text_series):
    """
    Generate and display a word cloud from text data.

    Parameters:
    text_series (pd.Series): Cleaned tweet text

    Returns:
    None
    """

    # Combine all tweets into one large text
    combined_text = " ".join(text_series)

    # Create WordCloud object
    wordcloud = WordCloud(
        width=800,
        height=400,
        background_color='white'
    ).generate(combined_text)

    # Display the word cloud
    plt.figure(figsize=(12, 6))
    plt.imshow(wordcloud, interpolation='bilinear')
    plt.axis('off')
    plt.title("Word Cloud of Negative Sentiment Tweets")
    plt.show()

# Call the function
generate_text_wordcloud(negative_text)
```