SENTIMENT ANALYSIS FOR MARKETING

BATCH MEMBER

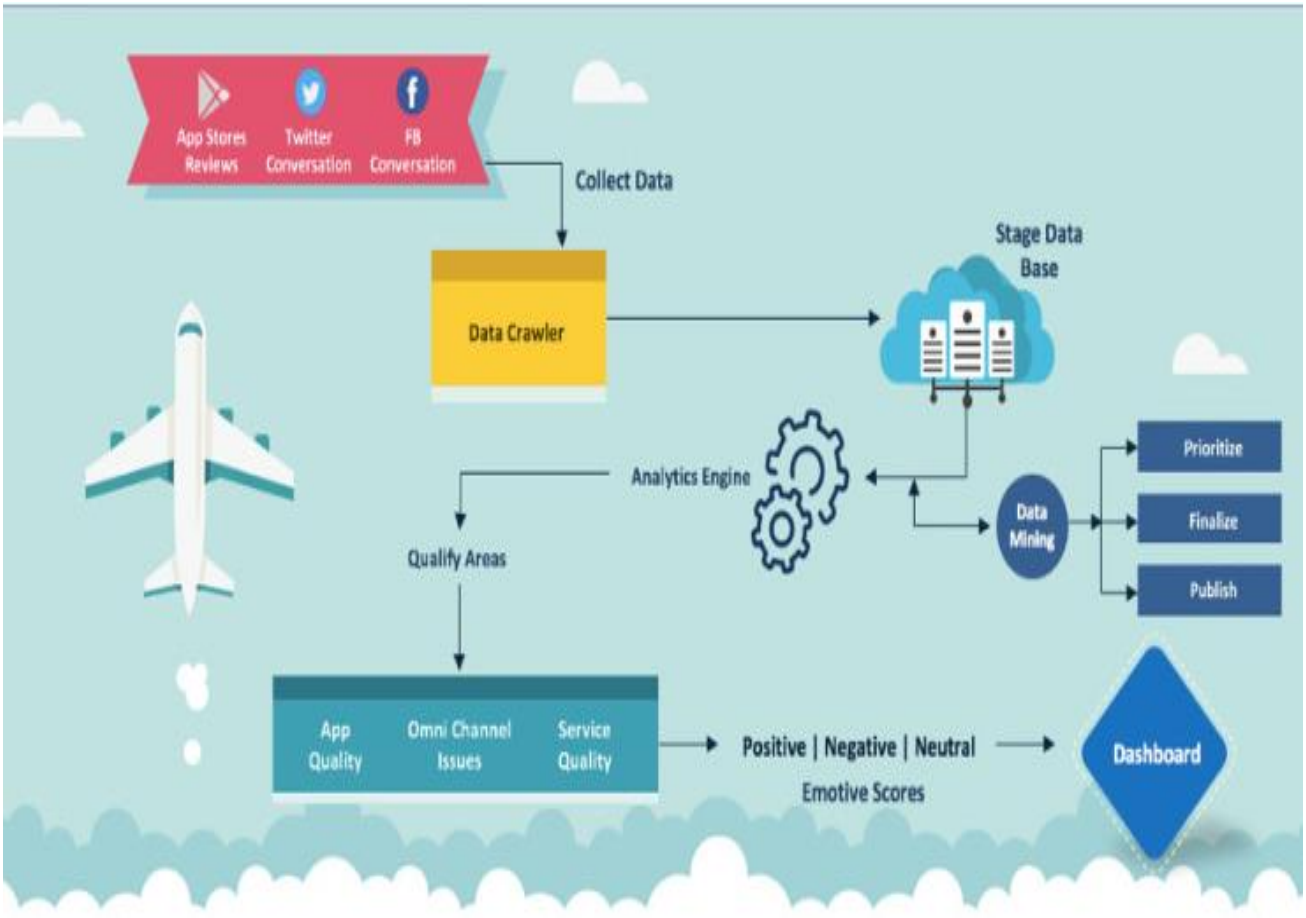
815621104028: PUVIYARASI EZHILARASAN

Phase-4 submission document

Project Title: Sentiment Analysis for Marketing

Phase 4: Development Part 2

Topic: Continue building the sentiment analysis solution by Employing NLP techniques and Generating insights.



Sentiment Analysis for Marketing

Introduction:

- In the intricate domain of marketing, understanding customer sentiment is pivotal for businesses to adapt and thrive. A sentiment analysis solution holds the potential to discern the underlying emotions behind consumer responses, reviews, and feedback. This extensive guide will navigate through the development of a potent sentiment analysis solution by employing two vital areas: Natural Language Processing (NLP) techniques and the generation of actionable insights.

- Natural Language Processing (NLP) is a sub-field of artificial intelligence that focuses on the interaction between computers and humans through natural language. Leveraging NLP for sentiment analysis allows businesses to automatically parse, analyze, and interpret vast amounts of textual data, extracting sentiments and emotions expressed by consumers.

- Generating insights from the analyzed sentiments is the next step, which translates raw sentiment data into actionable strategies, guiding marketers to make informed decisions. These insights shed light on product reception, brand reputation, and areas of potential growth or concern.

Given data set:

	tweet_id	airline_se	airline_se	negativer	negativer	airline	airline_se	name	negativer	retweet_c	text	tweet_co	tweet_cre	tweet_loc	user_timezone
2	5.7E+17	neutral	1			Virgin America		cairdin		0	@VirginAmerica Wh		########		Eastern Time (US & Canada)
3	5.7E+17	positive	0.3486		0	Virgin America		jnardino		0	@VirginAmerica plu		########		Pacific Time (US & Canada)
4	5.7E+17	neutral	0.6837			Virgin America		yvonnalynn		0	@VirginAmerica I di		########	Lets Play	Central Time (US & Canada)
5	5.7E+17	negative	1	Bad Flight	0.7033	Virgin America		jnardino		0	@VirginAmerica it's		########		Pacific Time (US & Canada)
6	5.7E+17	negative	1	Can't Tell	1	Virgin America		jnardino		0	@VirginAmerica and		########		Pacific Time (US & Canada)
7	5.7E+17	negative	1	Can't Tell	0.6842	Virgin America		jnardino		0	@VirginA		########		Pacific Time (US & Canada)
8	5.7E+17	positive	0.6745		0	Virgin America		cjmcginnis		0	@VirginAmerica yes		########	San Franci	Pacific Time (US & Canada)
9	5.7E+17	neutral	0.634			Virgin America		pilot		0	@VirginAmerica Rea		########	Los Angel	Pacific Time (US & Canada)
10	5.7E+17	positive	0.6559			Virgin America		dhepburn		0	@virginamerica Wel		########	San Diego	Pacific Time (US & Canada)
11	5.7E+17	positive	1			Virgin America		YupitsTate		0	@VirginAmerica it w		########	Los Angel	Eastern Time (US & Canada)
12	5.7E+17	neutral	0.6769		0	Virgin America		idk_but_youtube		0	@VirginAmerica did		########	1/1 loner :	Eastern Time (US & Canada)
13	5.7E+17	positive	1			Virgin America		HyperCamiLax		0	@VirginAmerica I &l		########	NYC	America/New_York
14	5.7E+17	positive	1			Virgin America		HyperCamiLax		0	@VirginAmerica Thi		########	NYC	America/New_York
15	5.7E+17	positive	0.6451			Virgin America		mollanderson		0	@VirginAmerica @v		########		Eastern Time (US & Canada)
16	5.7E+17	positive	1			Virgin America		sjespers		0	@VirginAmerica Tha		########	San Franci	Pacific Time (US & Canada)
17	5.7E+17	negative	0.6842	Late Flight	0.3684	Virgin America		smartwatermelon		0	@VirginAmerica SFC		########	palo alto,	Pacific Time (US & Canada)
18	5.7E+17	positive	1			Virgin America		ItzBrianHunty		0	@VirginAmerica So		########	west covi	Pacific Time (US & Canada)
19	5.7E+17	negative	1	Bad Flight	1	Virgin America		heatherovieda		0	@VirginAmerica I fl		########	this place	Eastern Time (US & Canada)
20	5.7E+17	positive	1			Virgin America		thebrandiray		0	I âĪ¸ flying @VirginA		########	Somewhe	Atlantic Time (Canada)

Overview of the process:

Outlined below is a comprehensive walkthrough for constructing a sentiment analysis solution employing NLP techniques and generating insights:

1. **Data Collection:** Gather textual data from diverse sources like reviews, social media, feedback forms, and customer interactions.

2. **Data Preprocessing:** Implement text cleaning processes to remove noise, such as punctuation, special characters, and numbers. Normalize the text through stemming and lemmatization.

3. **Employ NLP Techniques:** Use tokenization to split text into words or sentences. Implement algorithms to classify the sentiments into categories like positive, negative, and neutral.

4. **Feature Engineering:** Develop features that can boost the model's prediction accuracy, such as the frequency of sentiment words, sentence structures, or context embeddings.

5. **Model Training:** Feed the processed data to machine learning algorithms, allowing them to learn and understand the sentiment patterns.

6. **Generating Insights:** Interpret the sentiment analysis results to derive actionable marketing strategies. This could involve understanding product reception, customer pain points, or market trends.

7. **Model Evaluation:** Use metrics such as accuracy, precision, recall, and F1-score to gauge the performance of the sentiment analysis model on unseen data.

8. **Deployment:** After satisfactory evaluation, the model can be integrated into marketing tools or platforms to provide real-time sentiment analysis and insights.

PROCEDURE:

Employing NLP Techniques:

1. **Tokenization:** Break down the text into smaller pieces, like words or sentences.
2. **POS Tagging:** Identify the part of speech for each token.
3. **Named Entity Recognition:** Detect and classify entities like product names, brands, or user names.
4. **Sentiment Lexicons:** Use pre-existing lexicons that have sentiment scores for words, helping in sentiment determination.
5. **Deep Learning:** Implement techniques like LSTM or BERT for sentiment classification, especially when context matters.

Employing NLP Techniques:

```
In[1]:

text = "I love this product! It's amazing."

sentiment_score = afinn.score(text)


if sentiment_score > 0:

   sentiment = "positive"

elif sentiment_score < 0:

   sentiment = "negative"

else:

   sentiment = "neutral"


print(f"Sentiment: {sentiment}, Score: {sentiment_score}")
```

In[2]:

```
# Identify and print the most positive words
print('Most Positive Words')
for word, index in word_index_map.items():
    weight = model.coef_[0][index]
    if weight > threshold:
        print(word, weight)
```

```
Most Positive Words
great 5.516378614880334
virginamerica 3.4165631737297506
thank 8.172492647617368
southwestair 2.728627527382746
jetblue 3.1586422137139065
thanks 8.083441401654769
good 2.805464965619352
love 4.449114200749592
best 3.8620140153411207
appreciate 2.336612511736386
awesome 4.091284298701974
nice 2.16154339981104
thx 2.4222423243948117
amazing 3.6943805117897175
excellent 2.6209683927563843
worries 2.7557781608971568
wonderful 2.240905852132964
kudos 2.87036770762045
```

In[3]:

```
# Vectorize text data using TF-IDF
vectorizer = TfidfVectorizer(max_features=2000)
x_train = vectorizer.fit_transform(df_train['text'])
x_test = vectorizer.transform(df_test['text'])
y_train = df_train['target']
y_test = df_test['target']
```

In[4]:

```
# Vectorize text data for the binary sentiment classification
x_train = vectorizer.fit_transform(df_b_train['text'])
```

```
x_test = vectorizer.transform(df_b_test['text'])
y_train = df_b_train['target']
y_test = df_b_test['target']
```

In[5]:

```
# Splitting the data into training and testing sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(df['text'], df['airline_senti
ment'], test_size=0.2, random_state=42)

# Feature Extraction
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer(max_features=2500, min_df=7, max_df=0.8)
X_train = vectorizer.fit_transform(X_train).toarray()
X_test = vectorizer.transform(X_test).toarray()

# Model Training
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators=1000, random_state=0)
classifier.fit(X_train, y_train)
```

Out[5]:

RandomForestClassifier

```
RandomForestClassifier(n_estimators=1000, random_state=0)
```

In[6]:

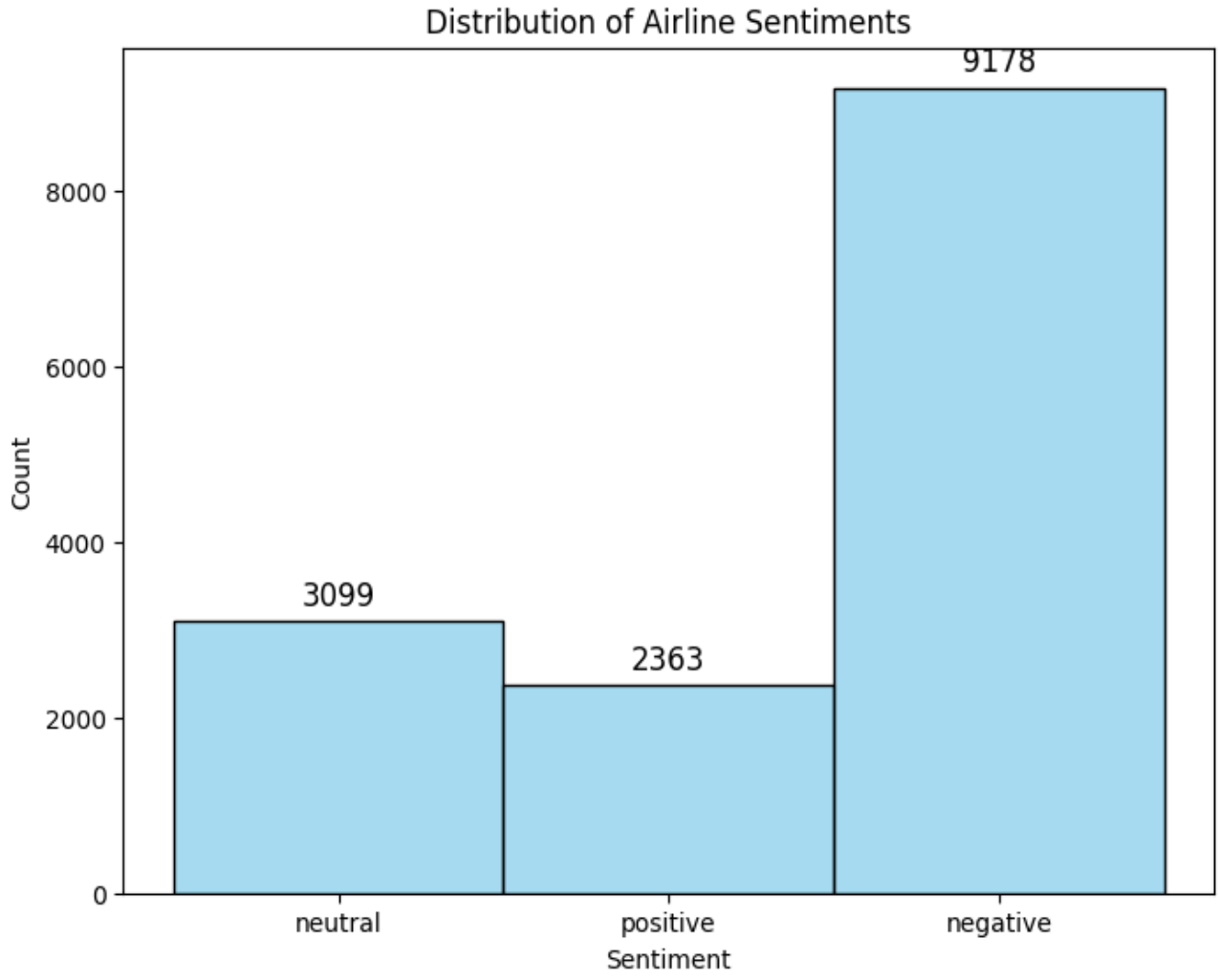
```
# Count the occurrences of each sentiment category
sentiment_counts = df['airline_sentiment'].value_counts()

# Visualize the distribution using a histogram with counts on bars
plt.figure(figsize=(8, 6))
ax = sns.histplot(df['airline_sentiment'], bins=3, color='skyblue', discrete=True
)
plt.xlabel('Sentiment')
plt.ylabel('Count')
plt.title('Distribution of Airline Sentiments')

# Add counts on top of the bars
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_heigh
t()), ha='center', va='center', fontsize=12, xytext=(0, 10), textcoords='offset p
oints')

plt.xticks()
```

```
plt.show()
```



Generating Insights:

1.Sentiment Distribution: Analyze the overall sentiment spread, understanding if most feedback is positive, negative, or neutral.

2.Temporal Analysis: Check sentiment trends over time to identify any changes or anomalies.

3.Product-specific Insights: Delve deeper into sentiments about specific products or services, aiding in product improvement or feature addition.

4.Competitor Analysis: By analyzing sentiments about competitors, derive strategies to gain a competitive edge.

5.Target Audience Sentiments: Understanding the sentiments of different consumer demographics can guide targeted marketing strategies.

In[7]:

```
# Function to preprocess the text
def preprocess_text(text):
    # Remove punctuations and numbers
    text = re.sub('[^a-zA-Z]', ' ', text)

    # Single character removal
    text = re.sub(r'\s+[a-zA-Z]\s+', ' ', text)

    # Removing multiple spaces
    text = re.sub(r'\s+', ' ', text)

    # Converting to Lowercase
    text = text.lower()

    # Lemmatization
    #text = text.split()
    #lemmatizer = WordNetLemmatizer()
    #text = [lemmatizer.lemmatize(word) for word in text if not word in set(stopw
ords.words('english'))]
    #text = ' '.join(text)

    return text

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

# Display the first 5 rows of the dataframe after preprocessing
df.head()
```

Out[7]:

	airline_sentiment	text
0	neutral	virginamerica what dhepburn said
1	positive	virginamerica plus you ve added commercials t...
2	neutral	virginamerica didn today must mean need to ta...
3	negative	virginamerica it really aggressive to blast o...
4	negative	virginamerica and it a really big bad thing a...

In[8]:

```
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

def evaluate_model(y_test, y_pred):
    print('Classification Report:')
    print(classification_report(y_test, y_pred))
    print('Confusion Matrix:')
    print(confusion_matrix(y_test, y_pred))
    print('Accuracy Score:')
    print(accuracy_score(y_test, y_pred))

y_pred = classifier.predict(X_test)
evaluate_model(y_test, y_pred)
```

```
Classification Report:
              precision    recall  f1-score   support

    negative       0.79      0.95      0.86      1889
     neutral       0.65      0.41      0.50       580
    positive       0.80      0.50      0.62       459

    accuracy                           0.77      2928
   macro avg       0.75      0.62      0.66      2928
weighted avg       0.76      0.77      0.75      2928
```

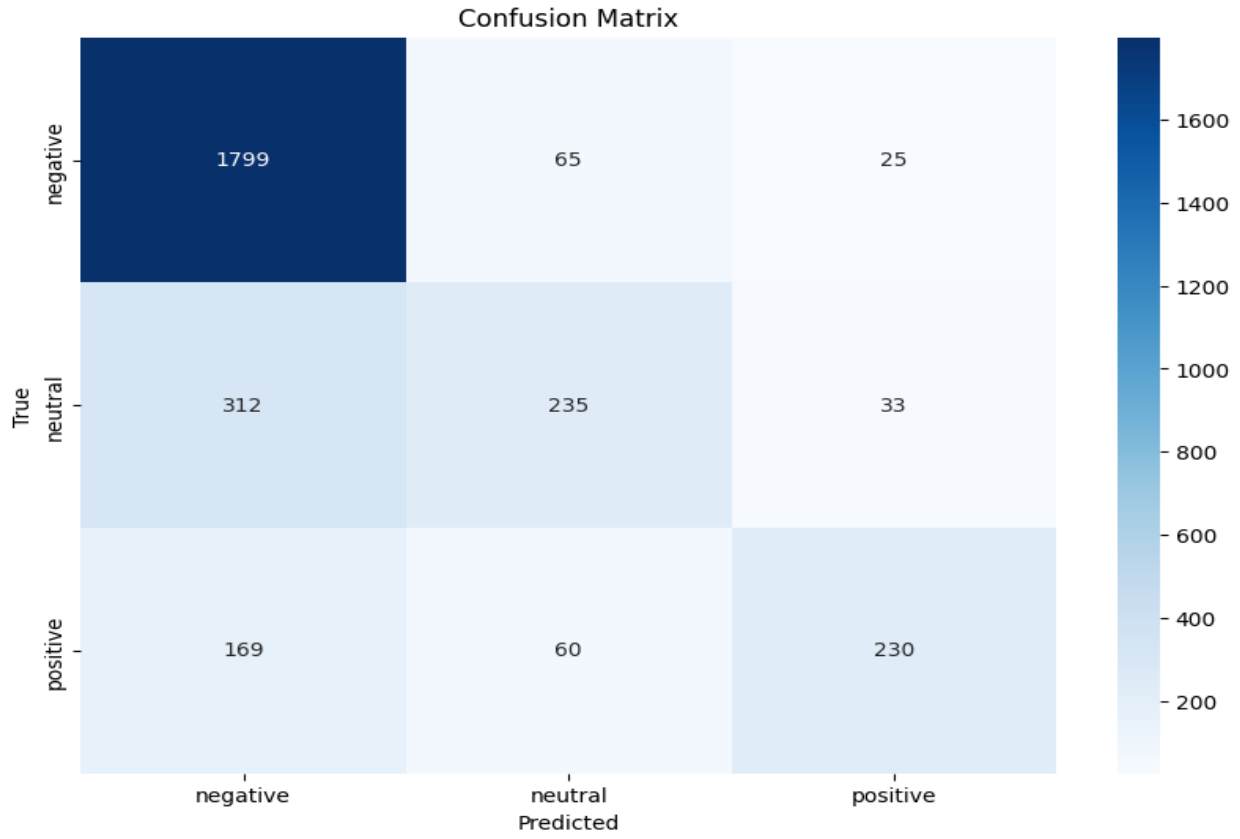
```
Confusion Matrix:
[[1799   65   25]
 [ 312  235   33]
 [ 169   60  230]]
Accuracy Score:
0.773224043715847
```

In[9]:

```
import matplotlib.pyplot as plt
import seaborn as sns

def plot_confusion_matrix(y_test, y_pred):
    cm = confusion_matrix(y_test, y_pred)
    df_cm = pd.DataFrame(cm, index = [i for i in ['negative', 'neutral', 'positiv
e']],
                  columns = [i for i in ['negative', 'neutral', 'positive']])
    plt.figure(figsize = (10,7))
    sns.heatmap(df_cm, annot=True, fmt='d', cmap='Blues')
    plt.title('Confusion Matrix')
    plt.xlabel('Predicted')
    plt.ylabel('True')
    plt.show()

plot_confusion_matrix(y_test, y_pred)
```



In[10]:

```
# Train a logistic regression model
model = LogisticRegression(max_iter=500)
model.fit(x_train, y_train)
```

Out[10]:

```
LogisticRegression
LogisticRegression(max_iter=500)
```

Model Evaluation:

1.Accuracy: Measure the fraction of predictions the model got right.

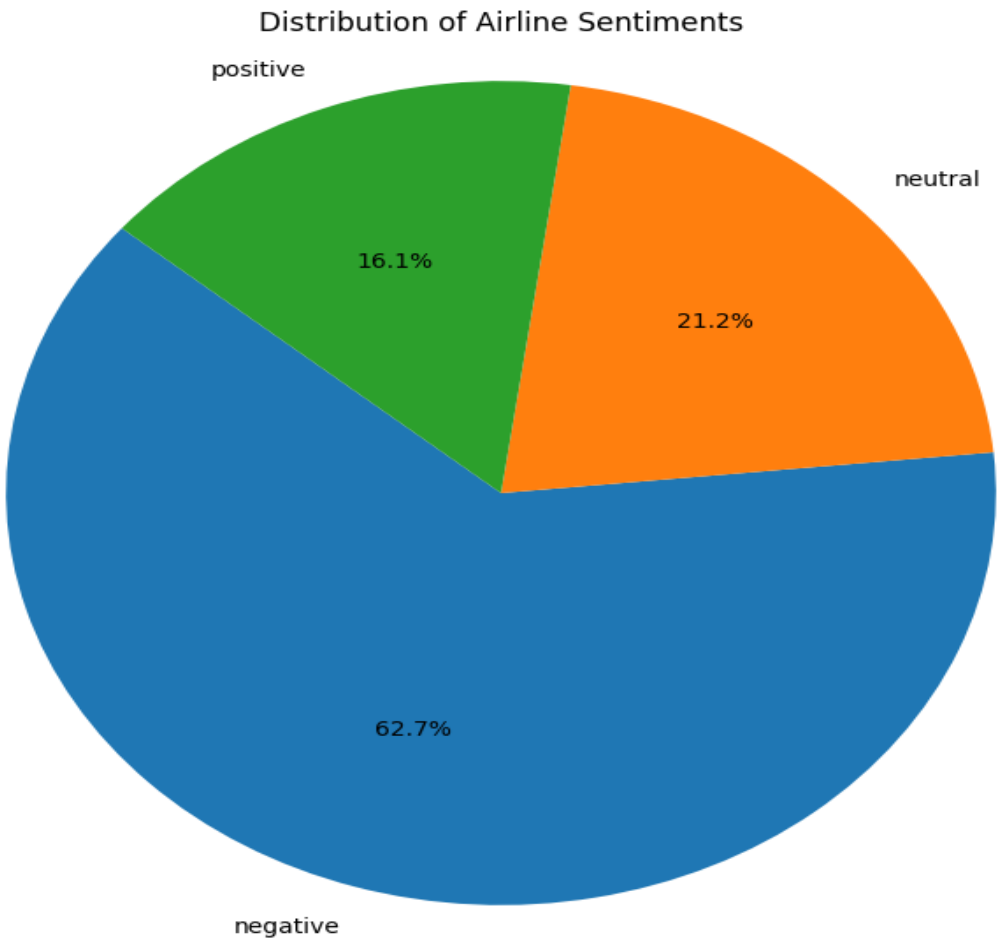
2.Precision & Recall: Gauge the model's ability to minimize false positives and false negatives.

3.F1-score: A balance between precision and recall.

4.Confusion Matrix: A summary of prediction results on a classification problem, highlighting false positives, false negatives, true positives, and true negatives.

In[11]:

```
# Visualize the distribution of airline sentiments using a pie chart
sentiment_counts = df['airline_sentiment'].value_counts()
plt.figure(figsize=(8, 8))
plt.pie(sentiment_counts, labels=sentiment_counts.index, autopct='%1.1f%%', start
angle=140)
plt.title('Distribution of Airline Sentiments')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()
```

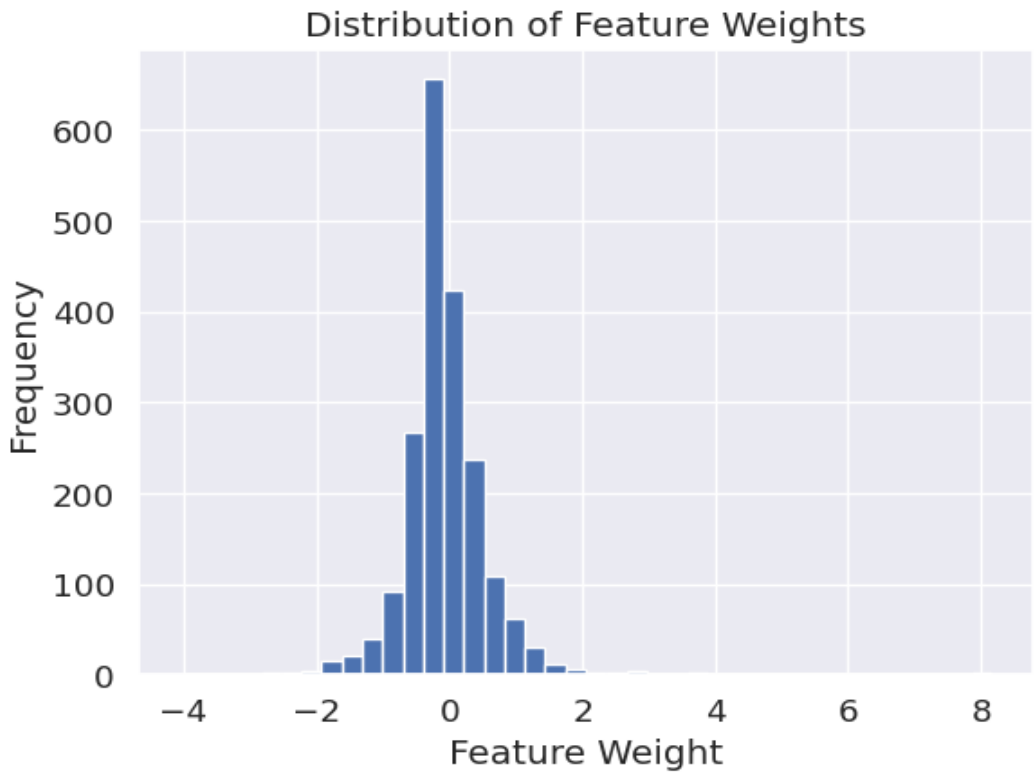


In[12]:

```
# Define a threshold for identifying most positive and most negative words
threshold = 2
```

In[13]:

```
# Plot a histogram of feature weights (coefficients)
plt.hist(model.coef_[0], bins=40)
plt.xlabel('Feature Weight')
plt.ylabel('Frequency')
plt.title('Distribution of Feature Weights')
plt.show()
```



In[14]:

```
from textblob import TextBlob


text = "I love this product! It's amazing."
blob = TextBlob(text)
sentiment = blob.sentiment


if sentiment.polarity > 0:
    sentiment_label = "positive"
elif sentiment.polarity < 0:
    sentiment_label = "negative"
```

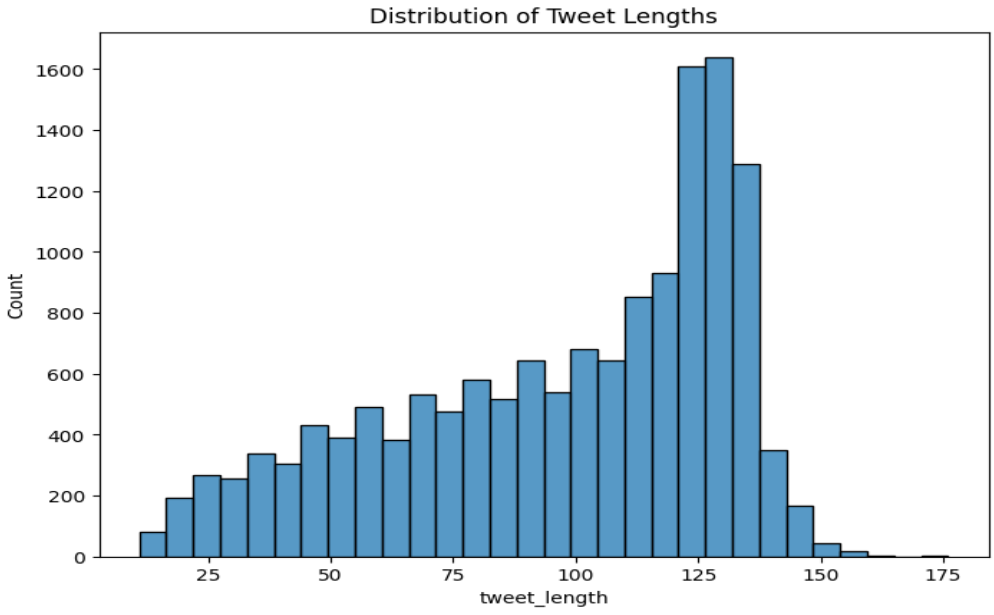
else:

sentiment_label = "neutral"

print(f"Sentiment: {sentiment_label}, Polarity: {sentiment.polarity}")

In[15]:

```
# Boxplot of tweet lengths
plt.figure(figsize=(8,6))
sns.boxplot(x='airline_sentiment', y='tweet_length', data=df)
plt.title('Distribution of Tweet Lengths by Sentiment')
plt.show()
```



In[16]:

```
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()
```

In[17]:

```
df.airline.value_counts()
```

Out[17]:

```
United            3822
US Airways        2913
American          2759
Southwest         2420
Delta             2222
Virgin America     504
Name: airline, dtype: int64
```

Conclusion:

- Developing a sentiment analysis solution for marketing is an ambitious endeavor that promises transformative results.

- By employing NLP techniques, we are equipped to navigate the vast seas of consumer data, extracting valuable sentiments.

- Translating these sentiments into actionable insights is the next logical step, bridging the gap between consumer voice and marketer response.