Source code:

```
        import pandas as pd
from textblob import TextBlob

# Sample DataFrame with text data
data = {'Text': ["I love this product!", "This is terrible.", "It's okay, nothing special.",
"Absolutely amazing!", "Worst experience ever."]}
df = pd.DataFrame(data)

# Function to analyze sentiment
def get_sentiment(text):
    analysis = TextBlob(text)
    return analysis.sentiment.polarity  # Returns sentiment score (-1 to 1)

# Apply sentiment analysis to DataFrame
df['Sentiment'] = df['Text'].apply(get_sentiment)

# Display results
print(df)

import pandas as pd
import matplotlib.pyplot as plt
from textblob import TextBlob

# Sample DataFrame with text data
data = {'Text': ["I love this!", "This is terrible.", "It's okay.", "Absolutely amazing!", "Worst
experience ever."]}
df = pd.DataFrame(data)

# Function to analyze sentiment
```

```
def get_sentiment(text):
    analysis = TextBlob(text)
    return analysis.sentiment.polarity  # Returns sentiment score (-1 to 1)

# Apply sentiment analysis
df['Sentiment'] = df['Text'].apply(get_sentiment)

# Plot histogram
plt.hist(df['Sentiment'], bins=5, edgecolor='black', alpha=0.7)
plt.xlabel('Sentiment Score')
plt.ylabel('Frequency')
plt.title('Sentiment Analysis Histogram')
plt.show()

import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from textblob import TextBlob

# Sample DataFrame with text data
data = {'Text': ["I love this!", "This is terrible.", "It's okay.", "Absolutely amazing!", "Worst
experience ever."]}
df = pd.DataFrame(data)

# Function to analyze sentiment
def get_sentiment(text):
    analysis = TextBlob(text)
    return analysis.sentiment.polarity  # Returns sentiment score (-1 to 1)

# Apply sentiment analysis
```

```
df['Sentiment'] = df['Text'].apply(get_sentiment)

# Create box plot
sns.boxplot(y=df['Sentiment'])
plt.ylabel('Sentiment Score')
plt.title('Sentiment Analysis Box Plot')
plt.show()

import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error

# Sample DataFrame with text and sentiment scores
data = {'Text': ["I love this!", "This is terrible.", "It's okay.", "Absolutely amazing!", "Worst
experience ever."],
        'Sentiment_Score': [0.9, -0.8, 0.2, 1.0, -1.0]}
df = pd.DataFrame(data)

# Convert text into numerical features
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(df['Text']).toarray()
y = df['Sentiment_Score']

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train Linear Regression model
```

```
model = LinearRegression()
model.fit(X_train, y_train)

# Predict sentiment scores
y_pred = model.predict(X_test)

# Evaluate model performance
mse = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error: {mse}")

import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from textblob import TextBlob

# Sample DataFrame with text data
data = {'Text': ["I love this!", "This is terrible.", "It's okay.", "Absolutely amazing!", "Worst experience ever."]}
df = pd.DataFrame(data)

# Function to analyze sentiment
def get_sentiment(text):
    analysis = TextBlob(text)
    return analysis.sentiment.polarity  # Returns sentiment score (-1 to 1)

# Apply sentiment analysis
df['Sentiment'] = df['Text'].apply(get_sentiment)

# Create KDE plot
sns.kdeplot(df['Sentiment'], shade=True, color="blue")
```

```
plt.xlabel('Sentiment Score')
plt.ylabel('Density')
plt.title('Sentiment Analysis KDE Plot')
plt.show()

import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from textblob import TextBlob

# Sample DataFrame with text data
data = {'Text': ["I love this!", "This is terrible.", "It's okay.", "Absolutely amazing!", "Worst experience ever."],
        'Engagement': [100, 50, 75, 200, 30]}  # Example engagement metric
df = pd.DataFrame(data)

# Function to analyze sentiment
def get_sentiment(text):
    analysis = TextBlob(text)
    return analysis.sentiment.polarity  # Returns sentiment score (-1 to 1)

# Apply sentiment analysis
df['Sentiment'] = df['Text'].apply(get_sentiment)

# Create scatter plot
sns.scatterplot(x=df['Engagement'], y=df['Sentiment'])
plt.xlabel('Engagement')
plt.ylabel('Sentiment Score')
plt.title('Sentiment Analysis Scatter Plot')
plt.show()
```

```
import pandas as pd
from textblob import TextBlob

# Sample DataFrame with text data
data = {'Text': ["I love this!", "This is terrible.", "It's okay.", "Absolutely amazing!", "Worst experience ever."]}
df = pd.DataFrame(data)

# Function to analyze sentiment
def get_sentiment(text):
    analysis = TextBlob(text)
    return analysis.sentiment.polarity  # Returns sentiment score (-1 to 1)

# Apply sentiment analysis
df['Sentiment'] = df['Text'].apply(get_sentiment)

# Calculate mean sentiment score
mean_sentiment = df['Sentiment'].mean()
print(f"Mean Sentiment Score: {mean_sentiment}")
```


```
import pandas as pd
from textblob import TextBlob

# Sample DataFrame with text data
data = {'Text': ["I love this!", "This is terrible.", "It's okay.", "Absolutely amazing!", "Worst experience ever."]}
df = pd.DataFrame(data)

# Function to analyze sentiment
```

```
def get_sentiment(text):
    analysis = TextBlob(text)
    return analysis.sentiment.polarity  # Returns sentiment score (-1 to 1)

# Apply sentiment analysis
df['Sentiment'] = df['Text'].apply(get_sentiment)

# Calculate median sentiment score
median_sentiment = df['Sentiment'].median()
print(f"Median Sentiment Score: {median_sentiment}")

import numpy as np
import pandas as pd
from textblob import TextBlob

# Sample DataFrame with text data
data = {'Text': ["I love this!", "This is terrible.", "It's okay.", "Absolutely amazing!", "Worst experience ever."]}
df = pd.DataFrame(data)

# Function to analyze sentiment
def get_sentiment(text):
    analysis = TextBlob(text)
    return analysis.sentiment.polarity  # Returns sentiment score (-1 to 1)

# Apply sentiment analysis
df['Sentiment'] = df['Text'].apply(get_sentiment)

# NumPy Aggregation
mean_sentiment = np.mean(df['Sentiment'])
```

```
median_sentiment = np.median(df['Sentiment'])
std_dev_sentiment = np.std(df['Sentiment'])

print(f"Mean Sentiment Score: {mean_sentiment}")
print(f"Median Sentiment Score: {median_sentiment}")
print(f"Standard Deviation: {std_dev_sentiment}")

import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from textblob import TextBlob

# Sample DataFrame with text data
data = {'Text': ["I love this!", "This is terrible.", "It's okay.", "Absolutely amazing!", "Worst experience ever."]}
df = pd.DataFrame(data)

# Function to analyze sentiment
def get_sentiment(text):
    analysis = TextBlob(text)
    polarity = analysis.sentiment.polarity
    return "Positive" if polarity > 0 else "Negative" if polarity < 0 else "Neutral"

# Apply sentiment analysis
df['Sentiment'] = df['Text'].apply(get_sentiment)

# Count sentiment categories
sentiment_counts = df['Sentiment'].value_counts()

# Create bar chart
```

```
sns.barplot(x=sentiment_counts.index, y=sentiment_counts.values, palette="coolwarm")
plt.xlabel('Sentiment')
plt.ylabel('Frequency')
plt.title('Sentiment Analysis Bar Chart')
plt.show()
```