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

# Load the dataset
data = pd.read_csv("linkedin-reviews.csv")

# Display the first few rows of the dataset
print(data.head())
```

```
                                              Review  Rating
0  Does absolutely nothing for a LinkedIn beginne...       1
1                           Force close(galaxy tab)        1
2  Slow and it tries to upload your contacts with...       1
3  Add ability to customize the profile and move ...       4
4  Good app, but it's a pain that it's not possib...       4
```

```
data.columns
```

```
Index(['Review', 'Rating'], dtype='object')
```

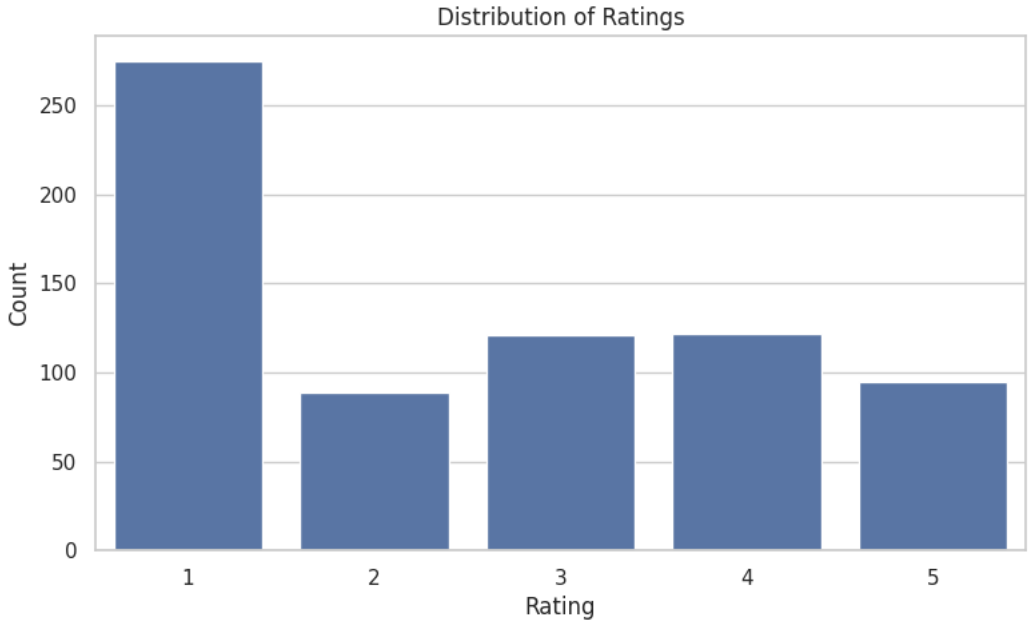
```
data.shape
```

```
(702, 2)
```

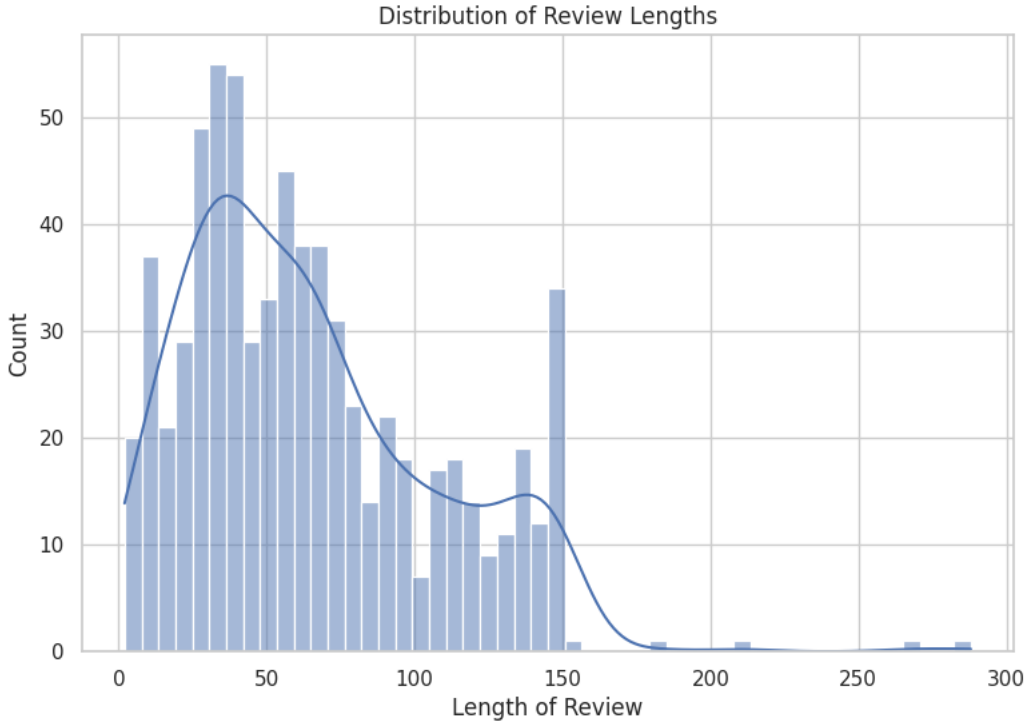
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 702 entries, 0 to 701
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   Review  702 non-null    object
 1   Rating  702 non-null    int64 
dtypes: int64(1), object(1)
memory usage: 11.1+ KB
```

```
# Plotting the distribution of ratings
sns.set(style="whitegrid")
plt.figure(figsize=(9, 5))
sns.countplot(data=data, x='Rating')
plt.title('Distribution of Ratings')
plt.xlabel('Rating')
plt.ylabel('Count')
plt.show()
```



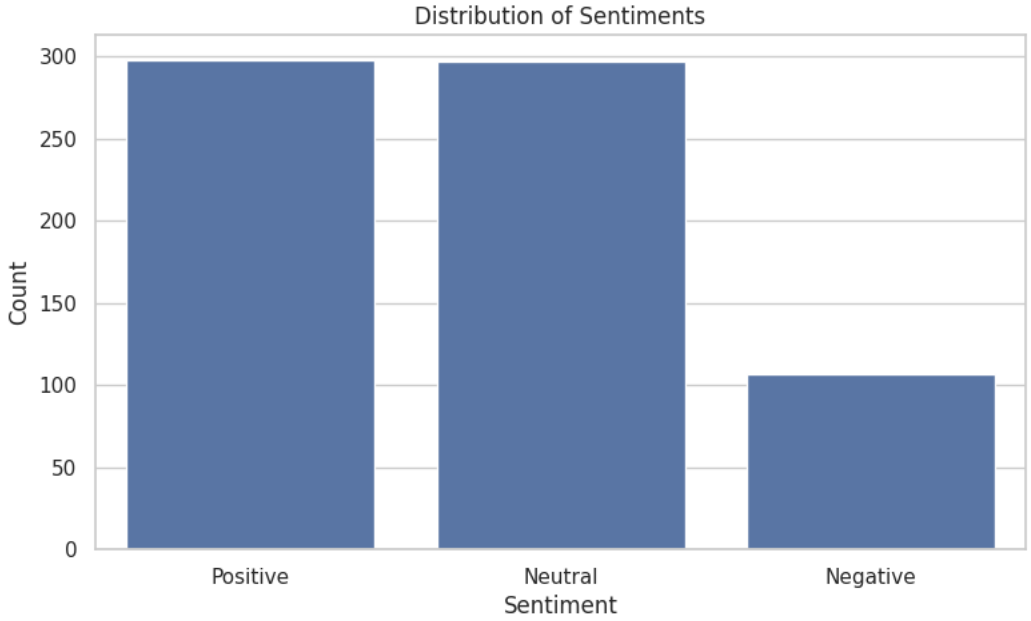
```
# Calculating the length of each review
data['Review Length'] = data['Review'].apply(len)

# Plotting the distribution of review lengths
plt.figure(figsize=(9, 6))
sns.histplot(data['Review Length'], bins=50, kde=True)
plt.title('Distribution of Review Lengths')
plt.xlabel('Length of Review')
plt.ylabel('Count')
plt.show()
```



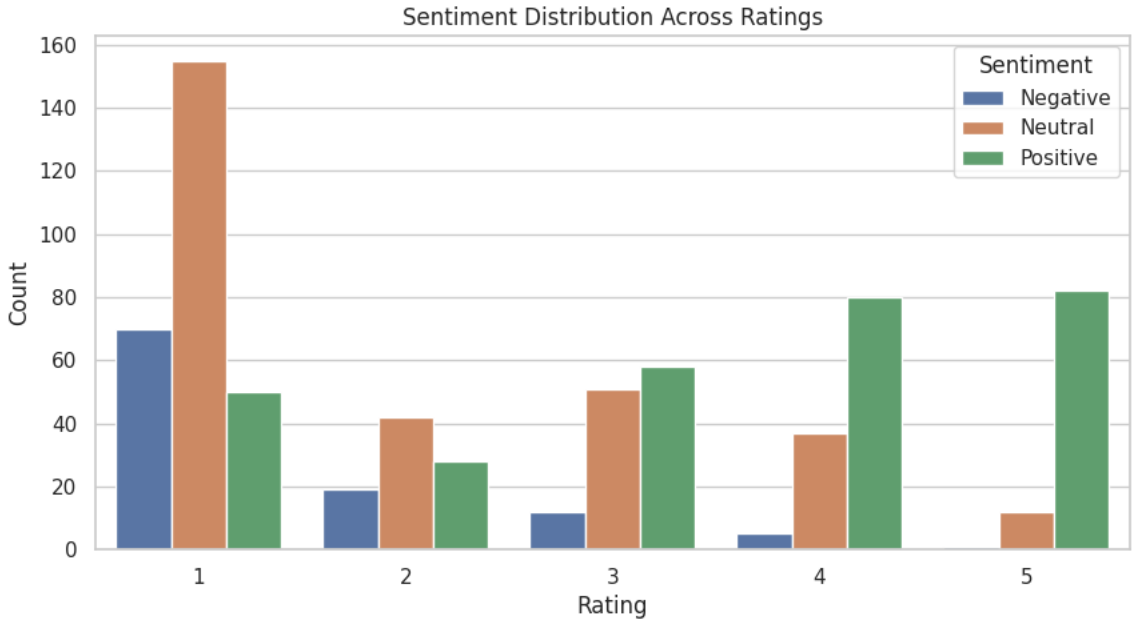

```
from textblob import TextBlob

def textblob_sentiment_analysis(review):
    # Analyzing the sentiment of the review
    sentiment = TextBlob(review).sentiment
    # Classifying based on polarity
    if sentiment.polarity > 0.1:
        return 'Positive'
    elif sentiment.polarity < -0.1:
        return 'Negative'
    else:
        return 'Neutral'

# Applying TextBlob sentiment analysis to the reviews
data['Sentiment'] = data['Review'].apply(textblob_sentiment_analysis)

# Displaying the first few rows with the sentiment
print(data.head())
```

```
                                              Review  Rating  Review Length  \
0  Does absolutely nothing for a LinkedIn beginne...       1             80
1                          Force close(galaxy tab)       1             23
2  Slow and it tries to upload your contacts with...       1             61
3  Add ability to customize the profile and move ...       4             90
4  Good app, but it's a pain that it's not possib...       4            133

  Sentiment
0  Negative
1   Neutral
2  Negative
3   Neutral
4  Positive
```

```
# Analyzing the distribution of sentiments
sentiment_distribution = data['Sentiment'].value_counts()

# Plotting the distribution of sentiments
plt.figure(figsize=(9, 5))
sns.barplot(x=sentiment_distribution.index, y=sentiment_distribution.values)
plt.title('Distribution of Sentiments')
plt.xlabel('Sentiment')
plt.ylabel('Count')
plt.show()
```




```
plt.figure(figsize=(10, 5))
sns.countplot(data=data, x='Rating', hue='Sentiment')
plt.title('Sentiment Distribution Across Ratings')
plt.xlabel('Rating')
plt.ylabel('Count')
plt.legend(title='Sentiment')
plt.show()
```




```
from wordcloud import WordCloud

# Function to generate word cloud for each sentiment
def generate_word_cloud(sentiment):
    text = ' '.join(review for review in data[data['Sentiment'] == sentiment]['Review'])
```

```
        wordcloud = WordCloud(width=800, height=400, background_color ='white').generate(text)
        plt.figure(figsize=(10, 5))
        plt.imshow(wordcloud, interpolation='bilinear')
        plt.title(f'Word Cloud for {sentiment} Reviews')
        plt.axis('off')
        plt.show()

    # Generating word clouds for each sentiment
    for sentiment in ['Positive', 'Negative', 'Neutral']:
        generate_word_cloud(sentiment)
```

Word Cloud for Positive Reviews