DataAppendix

Adam Chow, Tori Stoner, Aniyah McWilliams

2024-09-27

Quantitative Variables

Row Number

Definition

Each Row consists of one individual Yelp review

Creation

Provided by Original Kaggle Dataset

Missing Value

N/A

Rank (Rank)

Definition

The ranking number of the restaurant in the list of top-recommended restaurants.

Creation

Provided by Original Kaggle Dataset

Missing Values

N/A

Star Rating (StarRating)

Definition

The average star rating of the restaurant. This is based on the typical 1-5 Star rating scale.

Creation

Provided by Original Kaggle Dataset

Missing Value

N/A

Summary Statistics

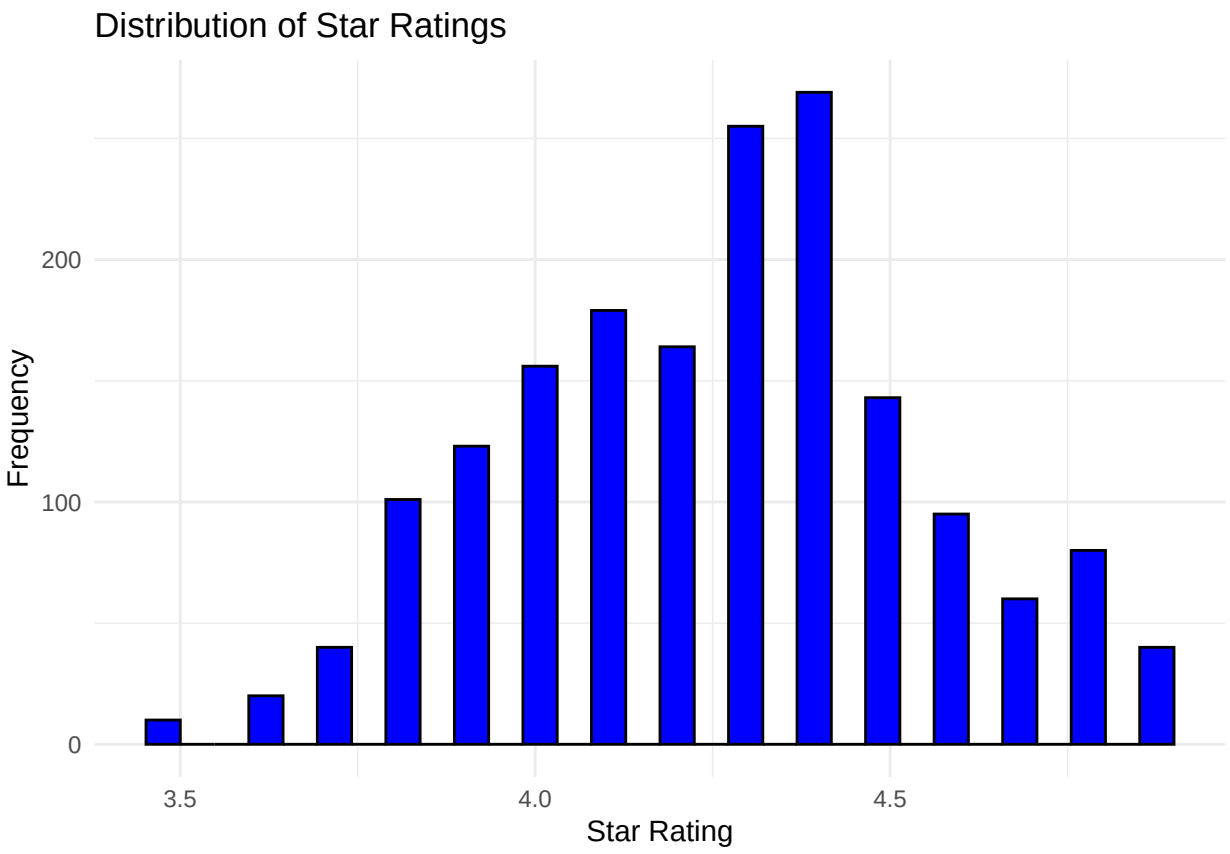
```
# Five-number summary
summary(top240$StarRating)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.500   4.000   4.300   4.258   4.400   4.900
```

Histogram

```
# Create a simple histogram
ggplot(top240, aes(x = StarRating)) +
  geom_histogram(fill = "blue", color = "black") +
  labs(title = "Distribution of Star Ratings", x = "Star Rating", y = "Frequency") +
  theme_minimal()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Number of Reviews (NumberOfReviews)

Definition

The total number of reviews the restaurant has received.

Creation

Provided by Original Kaggle Dataset

Missing Values

N/A

Summary Statistics

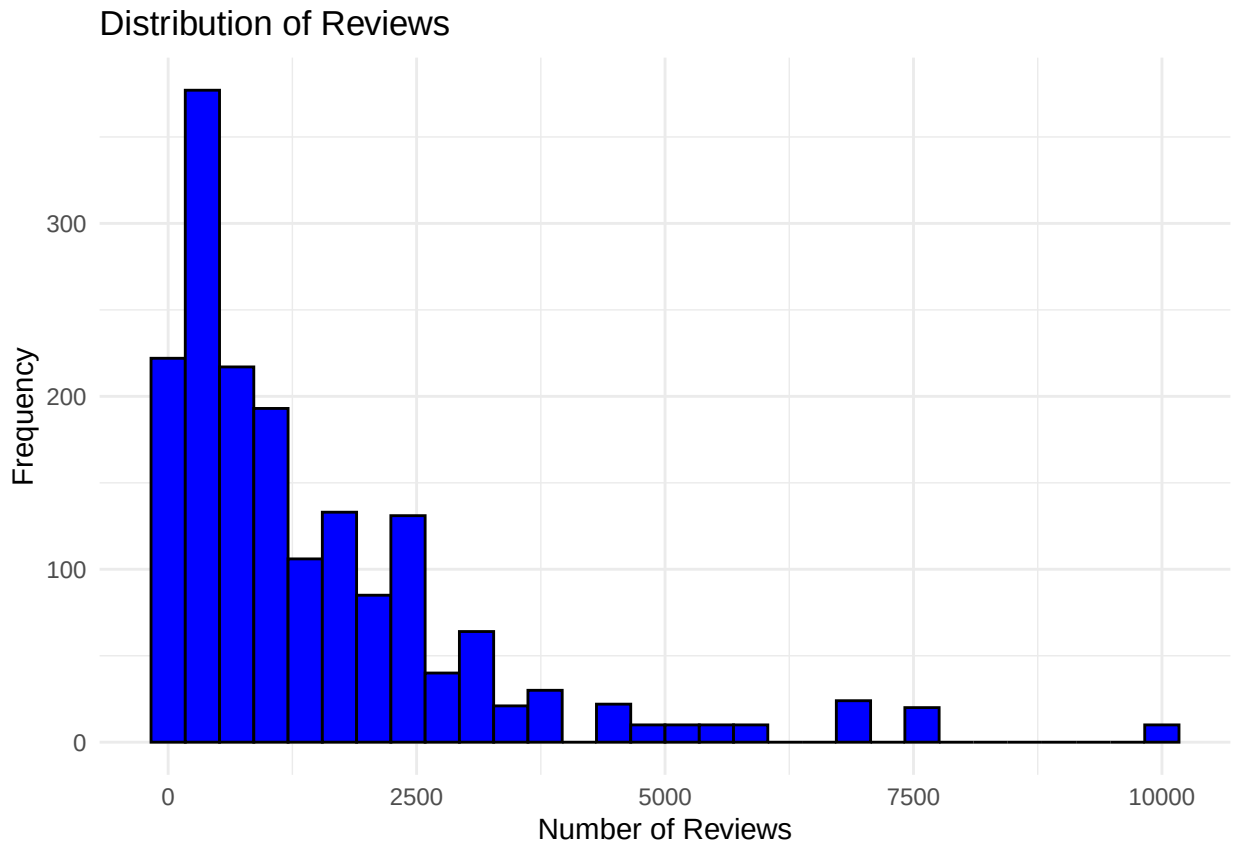
```
# Five-number summary
summary(top240$NumberOfReviews)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      21     351     936    1475    2044   10020
```

Histogram

```
# Create a simple histogram
ggplot(top240, aes(x = NumberOfReviews)) +
  geom_histogram(fill = "blue", color = "black") +
  labs(title = "Distribution of Reviews", x = "Number of Reviews", y = "Frequency") +
  theme_minimal()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Length of Review (LengthReview)

Definition

Creation

df['LengthReview'] = df['Comment'].apply(len)

Missing Values

Summary Statistics

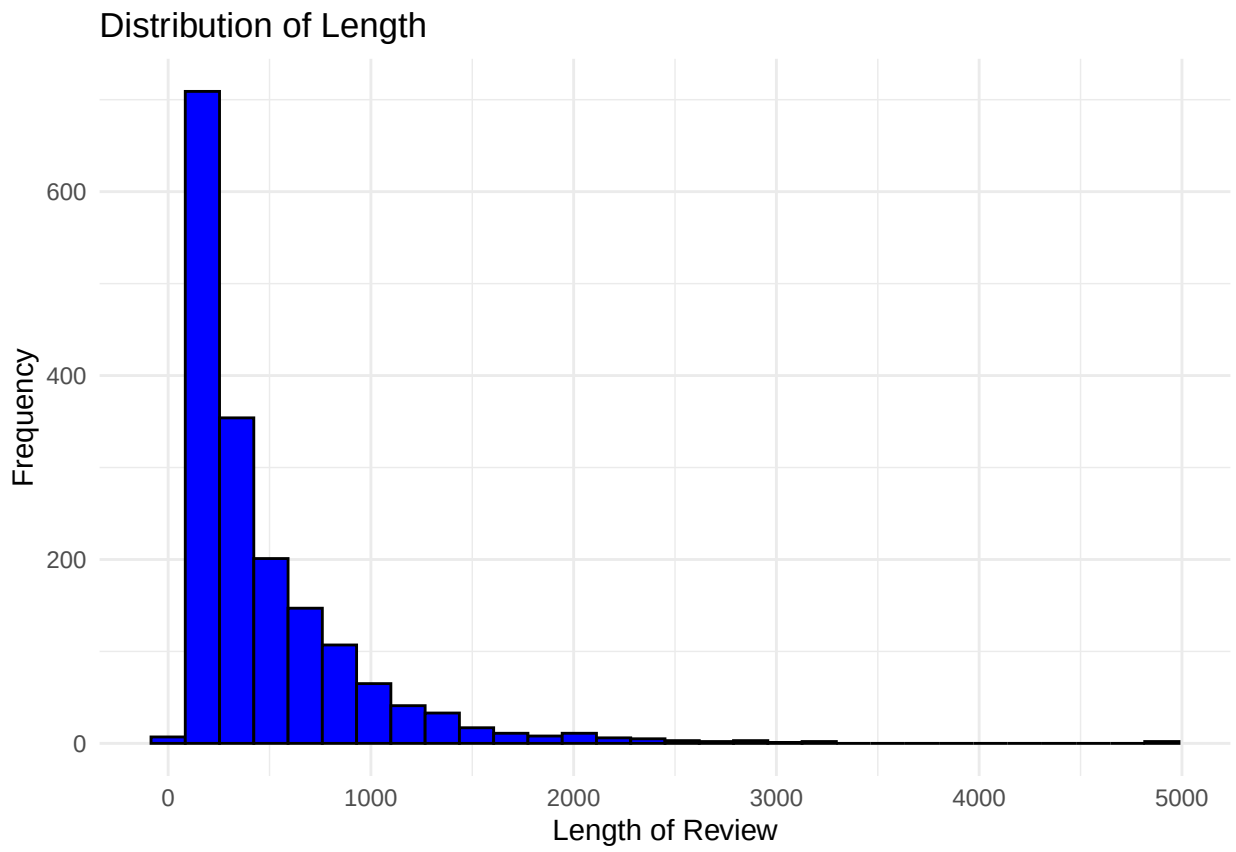
```
# Five-number summary
summary(top240$LengthReview)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    70.0   169.5   320.0   475.4   613.5  4971.0
```

Histogram

```
# Create a simple histogram
ggplot(top240, aes(x = LengthReview)) +
  geom_histogram(fill = "blue", color = "black") +
  labs(title = "Distribution of Length", x = "Length of Review", y = "Frequency") +
  theme_minimal()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Positive Score Sentiment ('positive score')

Definition

Proportion of the score that is classified as positive by Vader.

Creation

import nltk nltk.download('vader_lexicon')

from nltk.sentiment.vader import SentimentIntensityAnalyzer analyzer = SentimentIntensityAnalyzer()

for comment in df['Comment']: scores = analyzer.polarity_scores(comment)

for index, row in df.iterrows(): scores = analyzer.polarity_scores(row['Comment']) df.loc[index, 'positive score'] = scores['pos'] df.loc[index, 'negative score'] = scores['neg'] df.loc[index, 'neutral score'] = scores['neu'] df.loc[index, 'compound score'] = scores['compound']

Missing Values

N/A

Summary Statistics

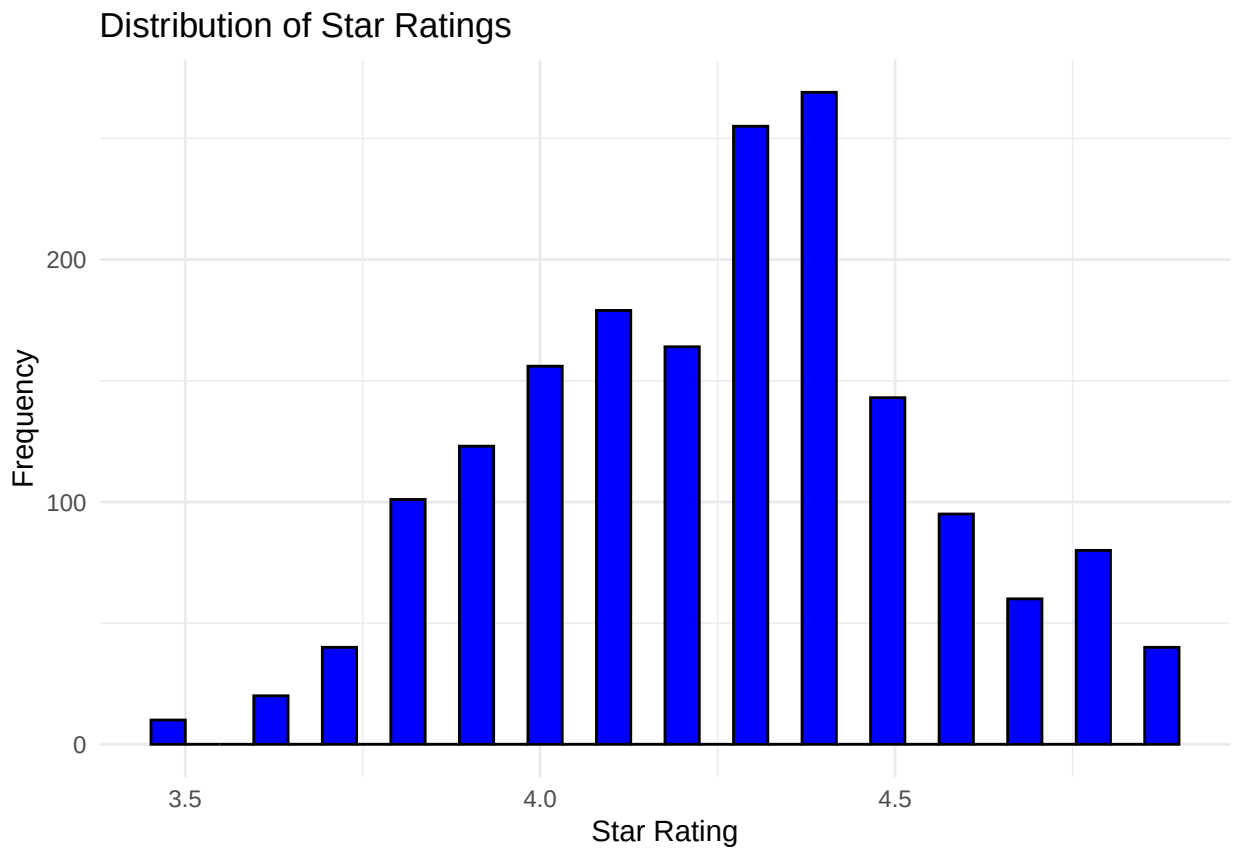
```
# Five-number summary for StarRating
summary(top240$StarRating)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.500   4.000   4.300   4.258   4.400   4.900
```

Histogram

```
# Create a simple histogram
ggplot(top240, aes(x = StarRating)) +
  geom_histogram(fill = "blue", color = "black") +
  labs(title = "Distribution of Star Ratings", x = "Star Rating", y = "Frequency") +
  theme_minimal()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Negative Score Sentiment ('negative score')

Definition

Proportion of the score that is classified as negative by Vader

Creation

import nltk nltk.download('vader_lexicon')

from nltk.sentiment.vader import SentimentIntensityAnalyzer analyzer = SentimentIntensityAnalyzer()

for comment in df['Comment']: scores = analyzer.polarity_scores(comment)

for index, row in df.iterrows(): scores = analyzer.polarity_scores(row['Comment']) df.loc[index, 'positive score'] = scores['pos'] df.loc[index, 'negative score'] = scores['neg'] df.loc[index, 'neutral score'] = scores['neu'] df.loc[index, 'compound score'] = scores['compound']

Missing Values

N/A

Summary Statistics

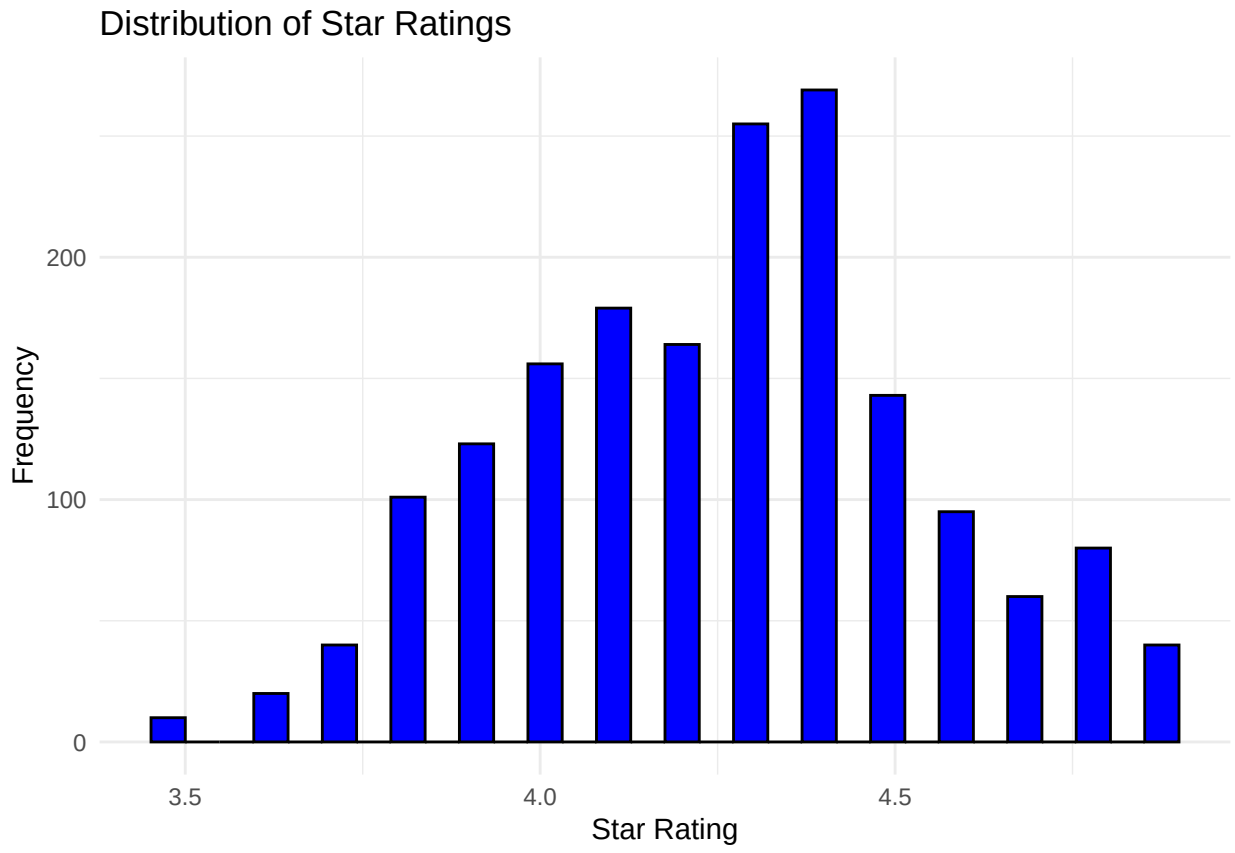
```
# Five-number summary for StarRating
summary(top240$StarRating)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.500   4.000   4.300   4.258   4.400   4.900
```

Histogram

```
# Create a simple histogram
ggplot(top240, aes(x = StarRating)) +
  geom_histogram(fill = "blue", color = "black") +
  labs(title = "Distribution of Star Ratings", x = "Star Rating", y = "Frequency") +
  theme_minimal()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Neutral Score Sentiment ('neutral score')

Definition

Proportion of the score that is classified as neutral by Vader

Creation

import nltk nltk.download(‘vader_lexicon’)

from nltk.sentiment.vader import SentimentIntensityAnalyzer analyzer = SentimentIntensityAnalyzer()

for comment in df[‘Comment’]: scores = analyzer.polarity_scores(comment)

for index, row in df.iterrows(): scores = analyzer.polarity_scores(row[‘Comment’]) df.loc[index, ‘positive score’] = scores[‘pos’] df.loc[index, ‘negative score’] = scores[‘neg’] df.loc[index, ‘neutral score’] = scores[‘neu’] df.loc[index, ‘compound score’] = scores[‘compound’]

Missing Values

N/A

Summary Statistics

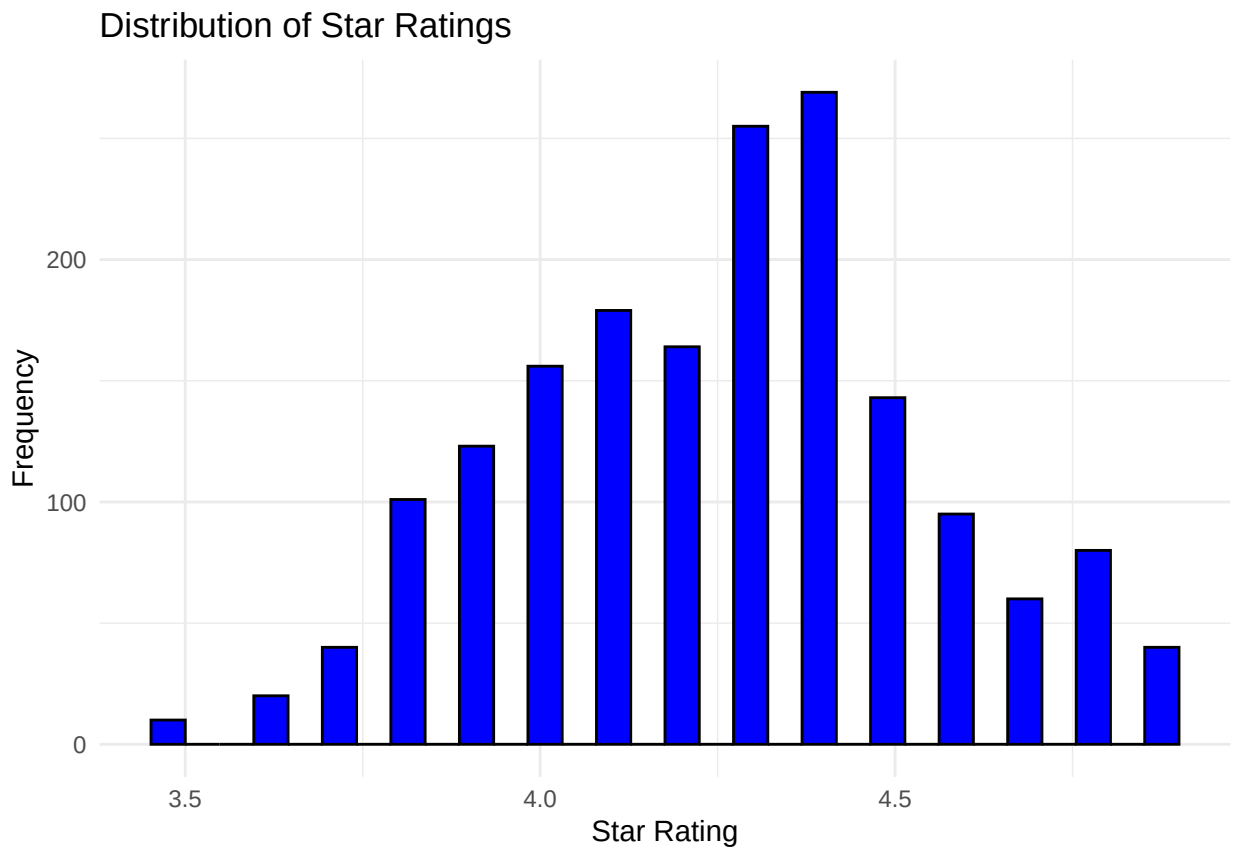
```
# Five-number summary for StarRating
summary(top240$StarRating)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.500   4.000   4.300   4.258   4.400   4.900
```

Histogram

```
# Create a simple histogram
ggplot(top240, aes(x = StarRating)) +
  geom_histogram(fill = "blue", color = "black") +
  labs(title = "Distribution of Star Ratings", x = "Star Rating", y = "Frequency") +
  theme_minimal()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Compound Score Sentiment ('compound score')

Definition

Overall Compound score calculated by Vader

Creation

import nltk nltk.download('vader_lexicon')

from nltk.sentiment.vader import SentimentIntensityAnalyzer analyzer = SentimentIntensityAnalyzer()

for comment in df['Comment']: scores = analyzer.polarity_scores(comment)

for index, row in df.iterrows(): scores = analyzer.polarity_scores(row['Comment']) df.loc[index, 'positive score'] = scores['pos'] df.loc[index, 'negative score'] = scores['neg'] df.loc[index, 'neutral score'] = scores['neu'] df.loc[index, 'compound score'] = scores['compound']

Missing Values

N/A

Summary Statistics

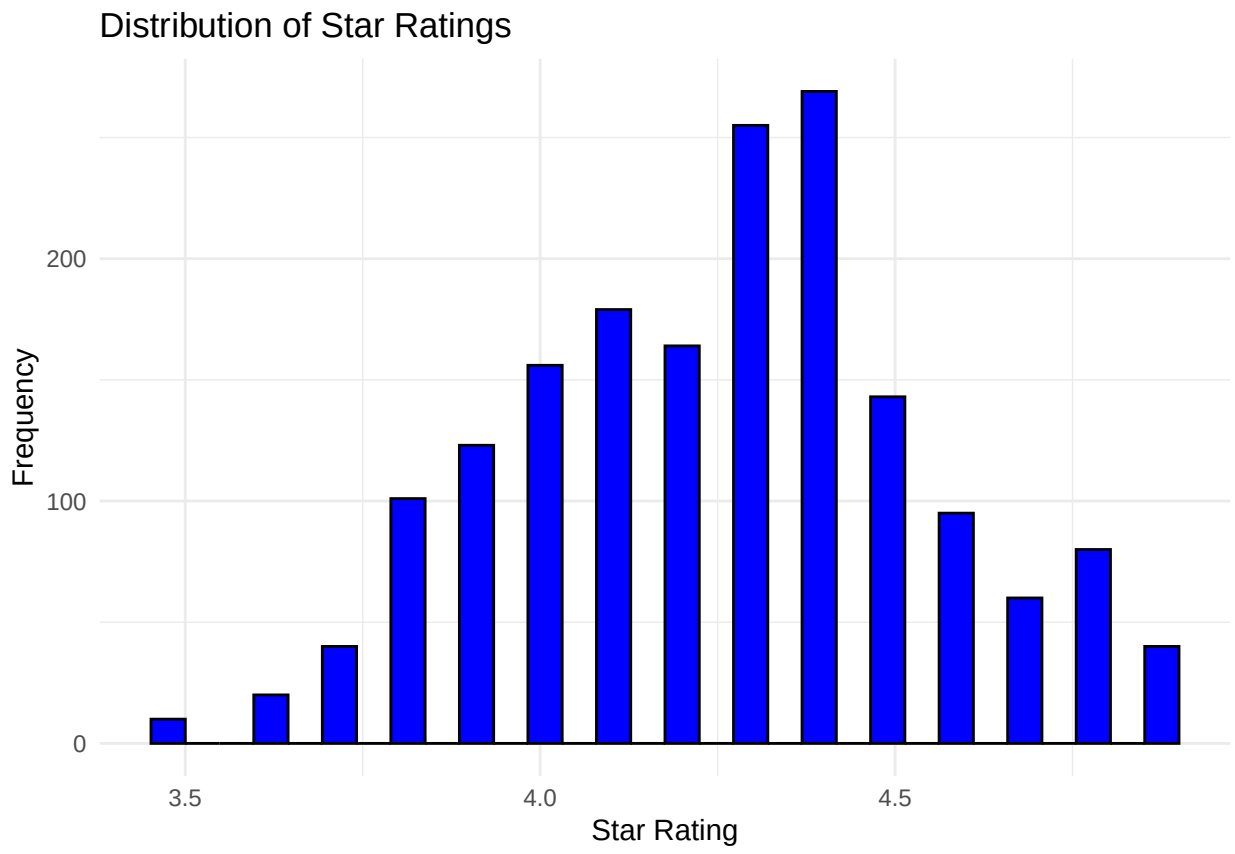
```
# Five-number summary for StarRating
summary(top240$StarRating)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.500   4.000   4.300   4.258   4.400   4.900
```

Histogram

```
# Create a simple histogram
ggplot(top240, aes(x = StarRating)) +
  geom_histogram(fill = "blue", color = "black") +
  labs(title = "Distribution of Star Ratings", x = "Star Rating", y = "Frequency") +
  theme_minimal()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Categorical Variables

Comment Date (CommentDate)

Definition

The date when the comment was posted.

Creation

Provided by Original Kaggle Dataset

Missing Values

N/A

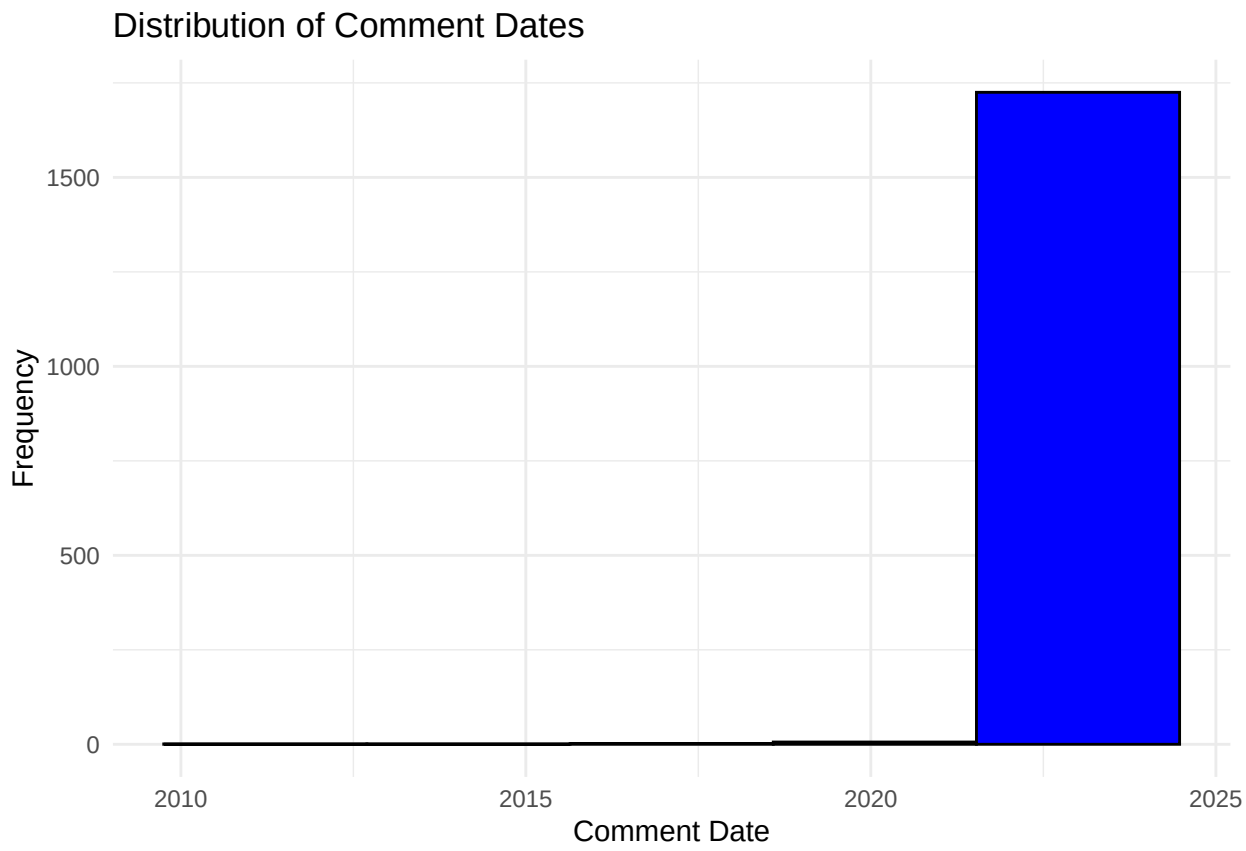
Frequency Table

```
summary(top240$CommentDate)
```

```
##         Min.      1st Qu.       Median         Mean      3rd Qu.         Max.
## "2011-12-07" "2023-08-19" "2023-09-03" "2023-08-06" "2023-09-10" "2023-09-17"
```

Bar Chart

```
# Create a simple histogram
ggplot(top240, aes(x = CommentDate)) +
  geom_histogram(bins = 5, fill = "blue", color = "black") +
  labs(title = "Distribution of Comment Dates", x = "Comment Date", y = "Frequency") +
  theme_minimal()
```



Date (Date)

Definition

The date when the data was scraped.

Creation

Provided by Original Kaggle Dataset

Missing Values

N/A

Frequency Table

All Dates are 2023-09-17.

Restaurant Name (RestaurantName)

Definition

The name of the restaurant.

Creation

Provided by Original Kaggle Dataset

Missing Values

N/A

Frequency Table

There are roughly 10 comments per restaurant (235).

Comment (Comment)

Definition

Customer comments about the restaurant.

Creation

Provided by Original Kaggle Dataset

Missing Values

N/A

Frequency Table

All Unique

Address (Address)

Definition

The physical address of the restaurant.

Creation

Provided by Original Kaggle Dataset

Missing Values

There are some missing values. For these values, we decided to make all of these NA values "Unkown".

Frequency Table

Same number of unique addresses as Restaurant names.

Style (Style)

Definition

The style(s) or type(s) of cuisine the restaurant offers.

Creation

Provided by Original Kaggle Dataset

Missing Values

N/A

Frequency Table

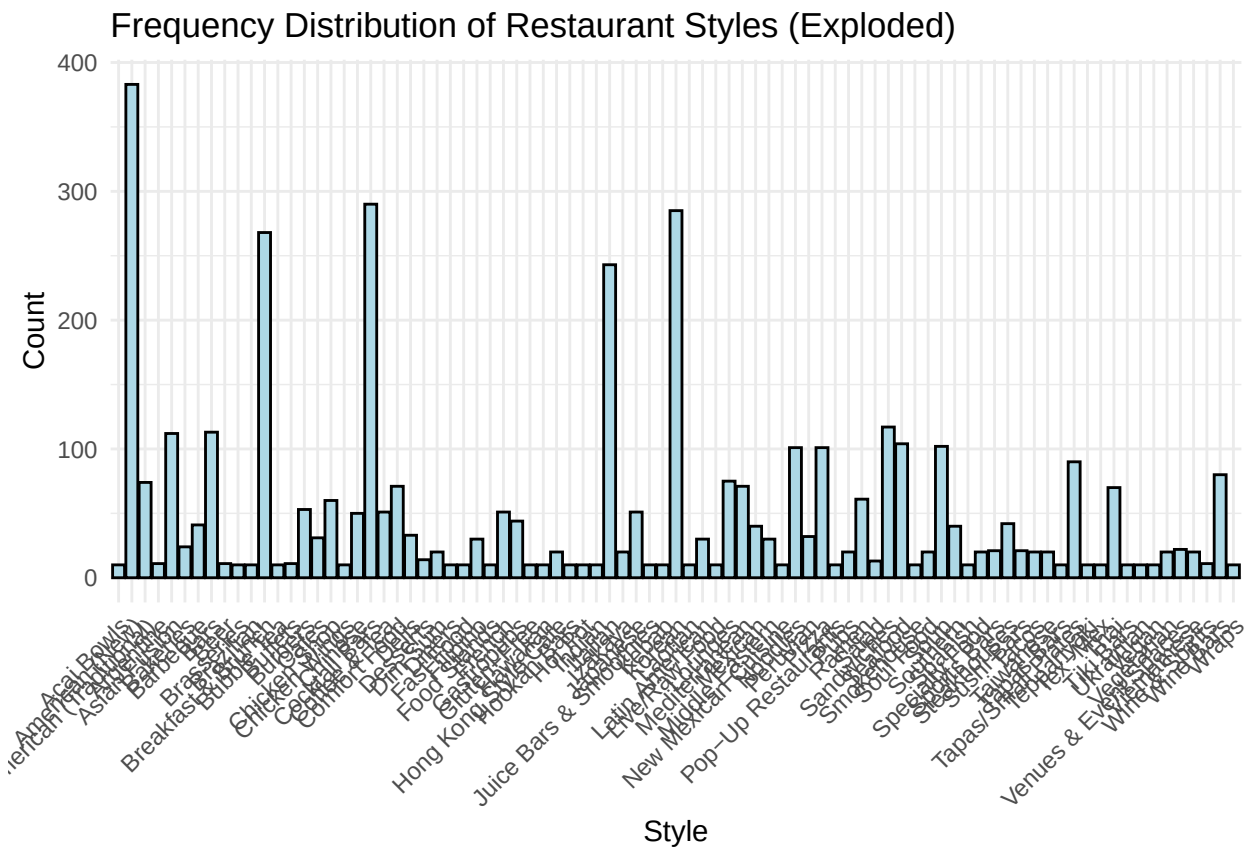
```
# Explode the Style column based on ", "
top240_exploded <- top240 %>%
  separate_rows(Style, sep = ", ")

table(top240_exploded$Style)
```

```
##
##             Acai Bowls         American (New) American (Traditional)
##                     10                   383                      74
##              Argentine           Asian Fusion                Bakeries
##                     11                   112                      24
##               Barbeque                  Bars                    Beer
##                     41                   113                      11
##             Brasseries              Brazilian     Breakfast & Brunch
##                     10                    10                     268
##             Bubble Tea               Buffets                 Burgers
##                     10                    11                      53
##                  Cafes           Chicken Shop          Chicken Wings
##                     31                    60                      10
##                Chinese          Cocktail Bars           Coffee & Tea
##                     50                   290                      51
##           Comfort Food                 Delis                Desserts
##                     71                    33                      14
##                Dim Sum                Diners              Fast Food
##                     20                    10                      10
##               Filipino            Food Stands                 French
##                     30                    10                      51
##             Gastropubs            Gluten-Free                Hawaiian
##                     44                    10                      10
##   Hong Kong Style Cafe            Hookah Bars                 Hot Pot
##                     20                    10                      10
##                 Indian                Italian                Izakaya
##                     10                   243                      20
##               Japanese Juice Bars & Smoothies                   Kebab
```

```
##                   51                   10                   10
##               Korean              Laotian       Latin American
##                  285                   10                   30
##        Live/Raw Food              Lounges        Mediterranean
##                   10                   75                   71
##              Mexican       Middle Eastern   New Mexican Cuisine
##                   40                   30                   10
##              Noodles              Peruvian                Pizza
##                  101                   32                  101
##    Pop-Up Restaurants                 Pubs                Ramen
##                   10                   20                   61
##               Salad             Sandwiches              Seafood
##                   13                  117                  104
##           Smokehouse             Soul Food                 Soup
##                   10                   20                  102
##             Southern              Spanish       Specialty Food
##                   40                   10                   20
##          Sports Bars            Steakhouses           Sushi Bars
##                   21                   42                   21
##               Tacos             Taiwanese           Tapas Bars
##                   20                   20                   10
##    Tapas/Small Plates            Teppanyaki              Tex-Mex
##                   90                   10                   10
##                 Thai             Tiki Bars            Ukrainian
##                   70                   10                   10
##               Vegan            Vegetarian Venues & Event Spaces
##                   10                   20                   22
##           Vietnamese       Wine & Spirits             Wine Bars
##                   20                   11                   80
##               Wraps
##                   10
```

Bar Chart

```r
# Bar chart for exploded Style column
ggplot(top240_exploded, aes(x = Style)) +
  geom_bar(fill = "lightblue", color = "black") +
  labs(title = "Frequency Distribution of Restaurant Styles (Exploded)", x = "Style", y = "Count") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) # Rotate x-axis labels for readability
```



Price (Price)

Definition

The price range of the restaurant, usually represented in terms of dollar signs.

Creation

Provided by Original Kaggle Dataset

Missing Values

There are missing values in this column and we decided to handle them by replacing the NAs with the mode of the column.

Frequency Table

```
table(top240$Price)
```

```
##
##   $$  $$$ $$$$
## 1256  382   97
```

Bar Chart

```
# Bar chart
ggplot(top240, aes(x = Price)) +
  geom_bar(fill = "lightblue", color = "black") +
  labs(title = "Frequency Distribution of Price", x = "Price", y = "Count") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

Predicted Sentiment (predicted_sentiment)

Definition

Character value of the classification of the sentiment

Creation

VaderModel = SentimentIntensityAnalyzer() def extract_score(text): score = VaderModel.polarity_scores(text) compound = score[‘compound’]

```
sentiment = 'neutral'
if(compound >= 0.05):
    sentiment = "positive" #Fixed indentation
```

```
elif(compound <= -0.05):
    sentiment = "negative" #Fixed indentation

return sentiment
```

df[“predicted_sentiment”] = df[“Comment”].apply(extract_score)

Missing Values

N/A

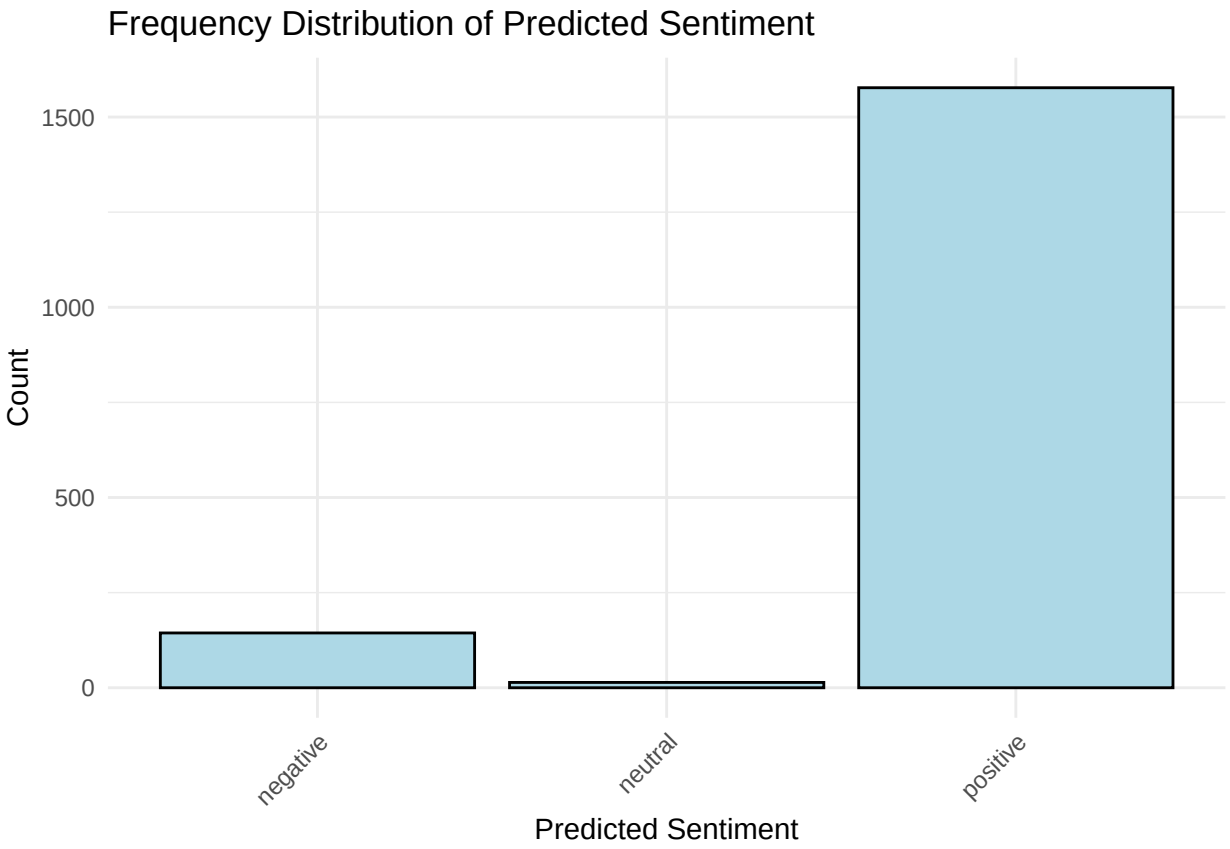
Frequency Table

```
table(top240$predicted_sentiment)
```

```
##
## negative  neutral positive
##      144       14     1577
```

Bar Chart

```
# Bar chart
ggplot(top240, aes(x = predicted_sentiment)) +
  geom_bar(fill = "lightblue", color = "black") +
  labs(title = "Frequency Distribution of Predicted Sentiment", x = "Predicted Sentiment", y = "Count") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) # Rotate x-axis labels for readability
```


Frequency Distribution of Predicted Sentiment
Count
1500
1000
500
0
negative
neutral
positive
Predicted Sentiment