Data Loading

In [2]:
```
import pandas as pd

data = pd.read_csv("/content/zomato_reviews.csv")
```

In [3]:
```
print(data)
```

```
      Unnamed: 0  rating                                             review
0              0       5                                               nice
1              1       5  best biryani , so supportive staff of outlet ,...
2              2       4     delivery boy was very decent and supportive. 👌 👍
3              3       1  worst biryani i have tasted in my life, half o...
4              4       5  all food is good and tasty . will order again ...
...          ...     ...                                                ...
5474        5474       5                                           complain
5475        5475       5  it took 1 hour to assign valvet and thn prepar...
5476        5476       5  took for an hour to prepare 3 khawsa, which in...
5477        5477       1  very very late, littrally did time pass and it...
5478        5478       1  Taste was stale and they give only 5 pieces in...

[5479 rows x 3 columns]
```

In [4]:
```
print(type(data))
```

```
<class 'pandas.core.frame.DataFrame'>
```

Data Cleaning & Preprocessing

Convert text to lowercase

In [5]:
```
print(data.head())
```

```
   Unnamed: 0  rating                                             review
0           0       5                                               nice
1           1       5  best biryani , so supportive staff of outlet ,...
2           2       4     delivery boy was very decent and supportive. 👌 👍
3           3       1  worst biryani i have tasted in my life, half o...
4           4       5  all food is good and tasty . will order again ...
```

In [6]:
```
data["review"] = data["review"].str.lower()
```

In [7]:
```
print(data.head())
```

```
   Unnamed: 0  rating                                             review
0           0       5                                               nice
1           1       5  best biryani , so supportive staff of outlet ,...
2           2       4     delivery boy was very decent and supportive. 👌 👍
3           3       1  worst biryani i have tasted in my life, half o...
4           4       5  all food is good and tasty . will order again ...
```

Remove URLs

In [8]:
```
print(data.dtypes)
```

```
Unnamed: 0     int64
rating         int64
review        object
dtype: object
```

In [9]:
```
print(type(data["review"]))
```

```
<class 'pandas.core.series.Series'>
```

In [10]:
```
non_strings_mask = data["review"].apply(lambda x: not isinstance(x, str))
non_string_elements = data[non_strings_mask]["review"]
print(non_string_elements)
```

```
3689    NaN
Name: review, dtype: object
```

Thus element 3689 is not a string. thus dropping the record

In [11]:
```
data = data.drop(3689)
```

In [12]:
```
print(data.iloc[3689])
```

```
Unnamed: 0                3690
rating                       3
review        sadi hui brownie
Name: 3690, dtype: object
```

In [13]:
```
import re

def remove_urls(text):
  return re.sub(r"http\S+", "", text)
```

In [14]:
```
def handle_review(text):
  """Handles review text, removing URLs if present."""
  if isinstance(text, str):
    # Apply URL removal only to strings
    return remove_urls(text)
  else:
    # Handle non-string elements (e.g., return original value)
    return text

data["review"] = data["review"].apply(handle_review)
```

In [15]:
```
print(data.head())
```

```
   Unnamed: 0  rating                                             review
0           0       5                                               nice
1           1       5  best biryani , so supportive staff of outlet ,...
2           2       4     delivery boy was very decent and supportive. 👌 👍
3           3       1  worst biryani i have tasted in my life, half o...
4           4       5  all food is good and tasty . will order again ...
```

In [16]:
```
print(len(data))
```

```
5478
```

Remove anything except the English language and space.

In [17]:
```
import re

def remove_non_english_and_spaces(text):
  """Removes non-English characters and keeps spaces."""
  english_letters_and_space_pattern = r"[a-zA-Z ]+"
  clean_text = re.sub(r"[^\s\w]", "", text)  # Alternative approach
  return clean_text

data["review"] = data["review"].apply(remove_non_english_and_spaces)
```

In [18]:
```
print(data["review"].head())
```

```
0                                                 nice
1    best biryani  so supportive staff of outlet  p...
2          delivery boy was very decent and supportive
3    worst biryani i have tasted in my life half of...
4    all food is good and tasty  will order again a...
Name: review, dtype: object
```

Remove Stop words

In [19]:
```
!pip install nltk
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (3.8.
1)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from
nltk) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (fro
m nltk) (1.3.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-pack
ages (from nltk) (2023.12.25)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from
nltk) (4.66.2)
```

In [22]:
```
import nltk
from nltk.corpus import stopwords
```

In [23]:
```
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

Out[23]:
```
True
```

In [24]:
```
def remove_stopwords_and_ampersand(text):
    """Removes stop words and "&amp;" from text using NLTK."""
    stop_words = stopwords.words('english')
    filtered_words = [word for word in text.split() if word not in stop_words and word
    return " ".join(filtered_words)

data["review"] = data["review"].apply(remove_stopwords_and_ampersand)
```

In [25]:
```
print(data["review"].head())
```

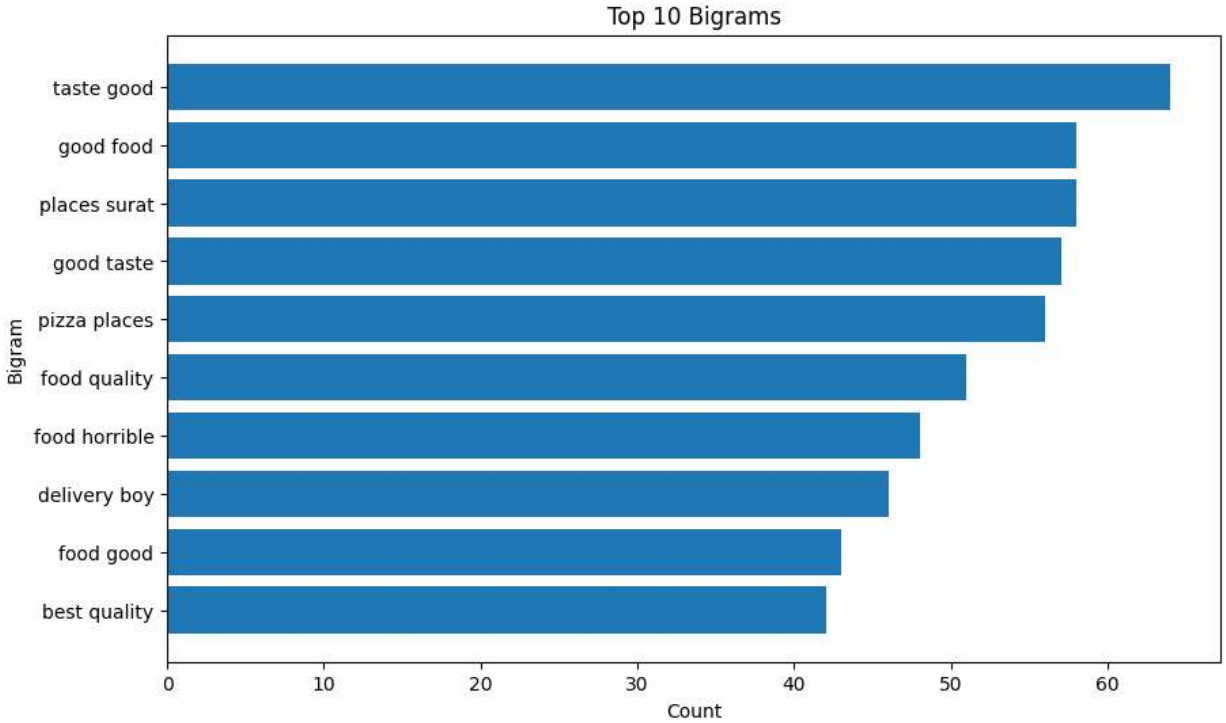
```
0                                                  nice
1    best biryani supportive staff outlet personali...
2                        delivery boy decent supportive
3       worst biryani tasted life half biryani dustbin
4    food good tasty order lots explore bawarchis menu
Name: review, dtype: object
```

Visualize top 20 most common words

In [26]:
```
# Import libraries
import pandas as pd
from nltk.corpus import stopwords
from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt

# Combine all reviews into a single string (optional if you want to analyze all review
all_reviews_text = " ".join(data["review"].tolist())

# Create a WordCloud object
wordcloud = WordCloud(background_color="white", stopwords=STOPWORDS, max_words=20).ger

# Create a new figure size
plt.figure(figsize=(8, 8))

# Display the word cloud
plt.imshow(wordcloud)
plt.axis("off")
plt.title("Top 20 Most Frequent Words", fontsize=15)
plt.show()
```

Visualize top 10 bigrams

In [27]:
```
from sklearn.feature_extraction.text import CountVectorizer
import matplotlib.pyplot as plt

# Create a CountVectorizer object
vectorizer = CountVectorizer(ngram_range=(2, 2))

# Fit and transform the text data
X = vectorizer.fit_transform(data["review"])

# Get the feature names (bigrams)
feature_names = vectorizer.get_feature_names_out()

# Sum up the counts of each bigram
bigram_counts = X.sum(axis=0).A1

# Create a dictionary of bigrams and their counts
bigram_dict = dict(zip(feature_names, bigram_counts))

# Sort the dictionary by counts in descending order
sorted_bigrams = sorted(bigram_dict.items(), key=lambda x: x[1], reverse=True)

# Extract top 10 bigrams and their counts
top_10_bigrams = sorted_bigrams[:10]
bigrams, counts = zip(*top_10_bigrams)

# Plot the top 10 bigrams
plt.figure(figsize=(10, 6))
plt.barh(bigrams, counts)
plt.xlabel('Count')
plt.ylabel('Bigram')
plt.title('Top 10 Bigrams')
plt.gca().invert_yaxis()
plt.show()
```



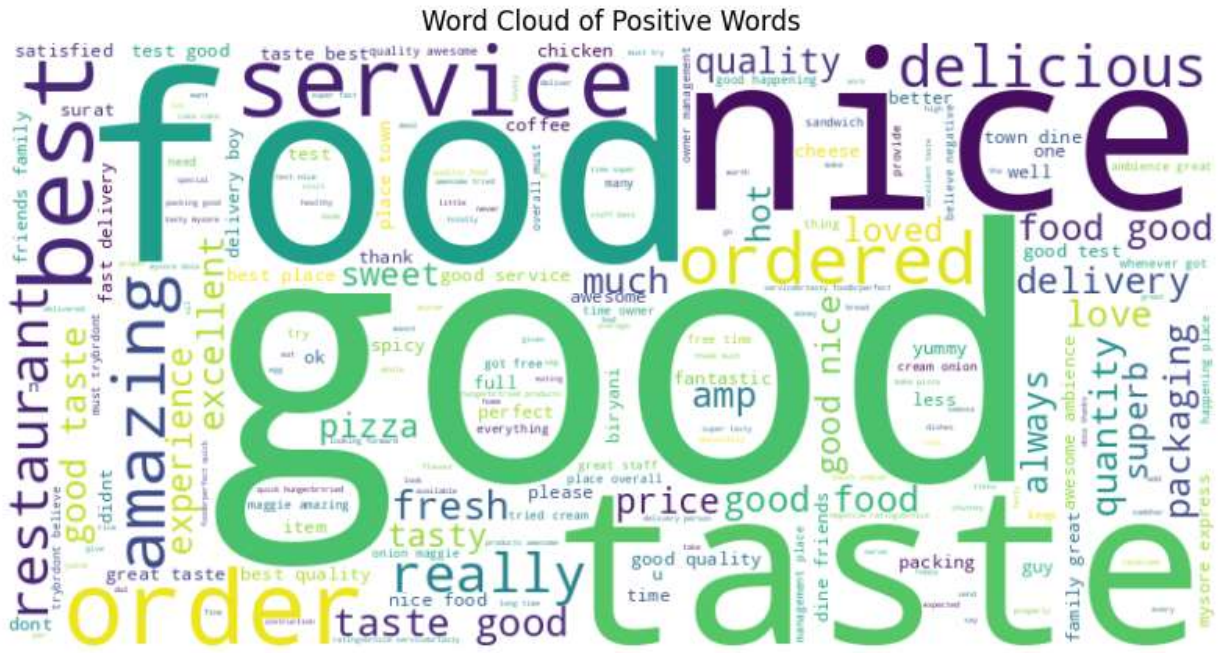
Perform the sentiment analysis using textblob

In [28]:
```
from textblob import TextBlob

# Assuming data["review"] is a pandas Series
sentiments = data["review"].apply(lambda x: TextBlob(x).sentiment)

# Extract polarity and subjectivity scores
polarities = [sentiment.polarity for sentiment in sentiments]
subjectivities = [sentiment.subjectivity for sentiment in sentiments]

# Add polarity and subjectivity scores to the DataFrame
data["polarity"] = polarities
data["subjectivity"] = subjectivities

# Display the DataFrame with sentiment analysis results
print(data.head())
```

```
   Unnamed: 0  rating                                             review  \
0           0       5                                               nice
1           1       5  best biryani supportive staff outlet personali...
2           2       4                        delivery boy decent supportive
3           3       1     worst biryani tasted life half biryani dustbin
4           4       5  food good tasty order lots explore bawarchis menu

   polarity  subjectivity
0  0.600000      1.000000
1  0.616667      0.616667
2  0.333333      0.833333
3 -0.583333      0.583333
4  0.700000      0.600000
```

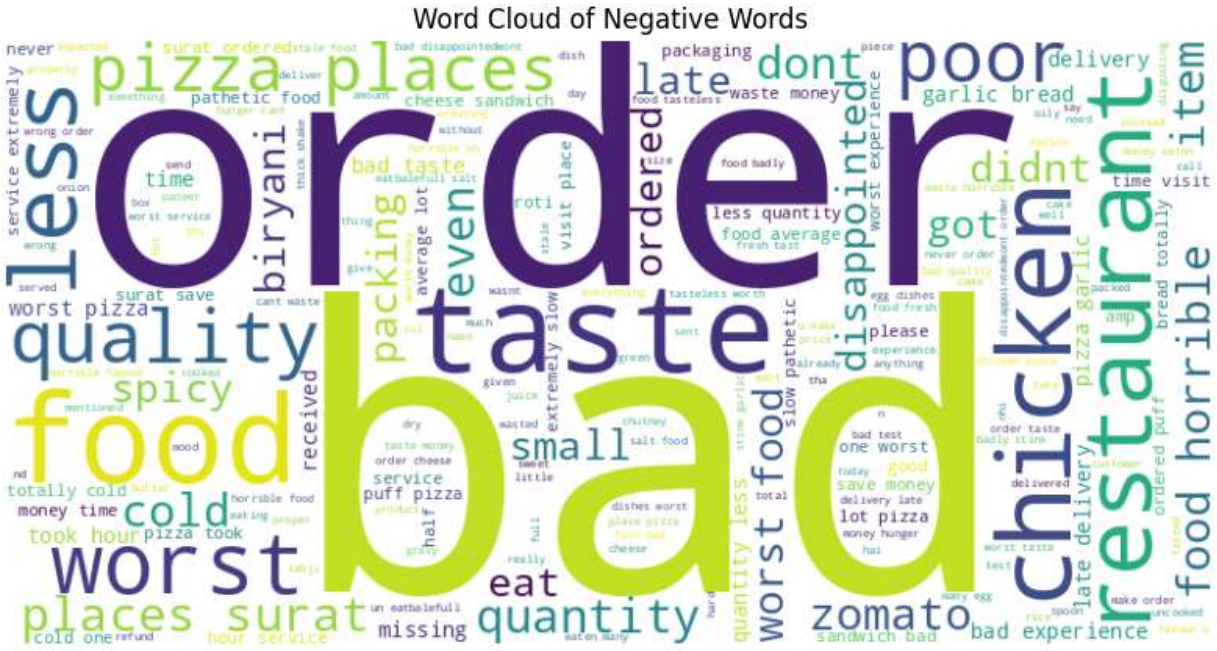
Display the word cloud of positive words

In [29]:
```
from wordcloud import WordCloud
import matplotlib.pyplot as plt

# Filter reviews with positive polarity
positive_reviews = data[data['polarity'] > 0]['review']

# Join positive reviews into a single string
positive_text = ' '.join(positive_reviews)

# Generate word cloud
wordcloud = WordCloud(width=800, height=400, background_color='white').generate(positi

# Display the word cloud
plt.figure(figsize=(10, 6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.title('Word Cloud of Positive Words')
plt.axis('off')
plt.show()
```



Display the word cloud of negative words

In [30]:
```
from wordcloud import WordCloud
import matplotlib.pyplot as plt

# Filter reviews with negative polarity
negative_reviews = data[data['polarity'] < 0]['review']

# Join negative reviews into a single string
negative_text = ' '.join(negative_reviews)

# Generate word cloud
wordcloud = WordCloud(width=800, height=400, background_color='white').generate(negati

# Display the word cloud
plt.figure(figsize=(10, 6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.title('Word Cloud of Negative Words')
plt.axis('off')
plt.show()
```



Display the word cloud of neutral words

In [31]:
```
from wordcloud import WordCloud
import matplotlib.pyplot as plt

# Filter reviews with neutral polarity (close to zero)
neutral_reviews = data[data['polarity'].between(-0.1, 0.1)]['review']

# Join neutral reviews into a single string
neutral_text = ' '.join(neutral_reviews)

# Generate word cloud
wordcloud = WordCloud(width=800, height=400, background_color='white').generate(neutra

# Display the word cloud
plt.figure(figsize=(10, 6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.title('Word Cloud of Neutral Words')
plt.axis('off')
plt.show()
```

Word Cloud of Neutral Words