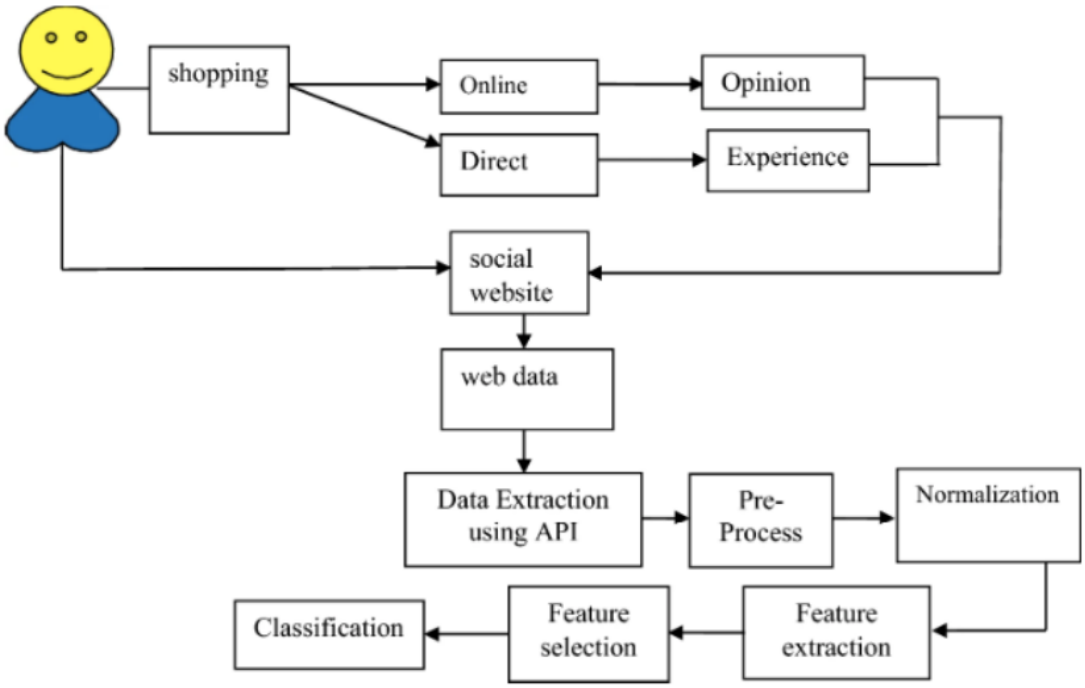
SVM in Action: Unmasking Sentiments in Customer Reviews through Web Crawling

Overview

Sentiment analysis, also known as opinion mining, involves evaluating emotions expressed in comments, feedback, or critiques using automated methods (Umarani, 2021). Recently, the internet, especially through social media and e-commerce websites, has become a major source of information, often presented as reviews that allow people to easily convey and share their feelings (Sultana et al., 2019). These daily interactions represent viewers' sentiments and opinions about specific services or items. The wealth of information is valuable for businesses, aiding in market research, product enhancement, managing brand reputation, and improving customer experience (8 Applications of Sentiment Analysis, 2020).



Process of Sentiment Analysis on Reviews for a Service/Product using ML (Yılmaz, 2023)

Automation, facilitated by machine learning (ML) techniques, accelerates the analysis of text patterns. Yılmaz (2023) highlights that supervised learning is a widely used ML technique for sentiment analysis. In this approach, data is labelled and used to train the algorithm, allowing it to classify new, unlabelled data based on the pre-labelled data. ML can grasp the intricacies and variations of human language through advanced sentiment analysis types such as aspect-based, entity-based, or intent-based sentiment analysis.

For this study, the chosen supervised ML model is the Support Vector Machine (SVM). SVM is trained to classify text within the sentiment polarity model, providing a method to understand and categorize sentiments in a reliable manner. The SVM (Putri et.al, 2023) method has great performance in determining the area under the curve (AUC) and can perform classifications to increase accuracy (Naz et al, 2019). Another reason for choosing SVM as a ML model is its ability to work with high dimensional data and apprehend complex relationships between the features.

Web Crawling for Customer Reviews

Domain Selection and Technological Optimisation in Advanced Web Crawling

The purpose of this study was to examine product reviews gathered from two online review platforms: Product Reviews (https://www.productreview.com.au/) and Trustpilot (https://au.trustpilot.com/). These websites serve as user-generated communities where individuals can share their reviews, aiding others in making informed purchasing decisions and promoting businesses, particularly startups (Top 28 Product Review Websites for Online Marketers | LiveChat Partners Blog, n.d.). For Trustpilot, the study retrieved 586 rows and 4 columns of data, while for the Product Review website, 431 rows and 4 columns of variables were obtained.

To collect data, a combination of web scraping and web crawling methods was employed. Web scraping was utilized to extract text data from web documents. Web crawling involved an iterative process of fetching web links for pages, originating from a list of URL links. The web crawling aimed for a comprehensive collection of customer reviews by indexing web pages. HTTP requests were made using BeautifulSoup, with time delays implemented to prevent reaching web scraping limits. Selenium was utilized for automated browser interaction. The web crawling was set to traverse 10 pages for this study.

```
# The number of pages for webcrawling
num_pages = 10

with requests.Session() as session:
    session.headers.update(headers)
    driver = webdriver.Chrome()  # Chrome driver in path

    for page_num in range(1, num_pages + 1):
        url = f"{base_url}{page_num}"

        # Selenium to load dynamic content
        driver.get(url)

        try:
            # Increase timeout for the first page and subsequent pages
            timeout = 30 if page_num == 1 else 10
            element_present =
EC.presence_of_element_located((By.CSS_SELECTOR, 'p.zBhAUA._E2YAm'))
            WebDriverWait(driver, timeout).until(element_present)
        except TimeoutException:
            print(f"Timed out waiting for page {page_num} to load.")
            continue  # Skip this page and move on to the next one

        # Introduce a delay after opening each page
        time.sleep(4)
```

```
        soup = BeautifulSoup(driver.page_source, 'html.parser')
```

Considerations regarding the copyright of data on product review sites ("Intellectual Property in Websites: Ownership and Protection," 2021) arise from the creation, publication, and utilization of content and data contributed by customers for rating and reviewing sites. These considerations encompass ownership, content protection respecting author rights, reviews, and user rights, as well as the potential for data infringement or misuse. Factors influencing copyright considerations involve the nature of the content and data, specifically opinion-based in this study. Additionally, they encompass the terms and conditions set by review sites, emphasizing transparent policies and agreements to adhere to relevant laws and regulations.

Limitations faced by product review sites revolve around challenges impacting the quality and quantity of retrieved data. These challenges include non-uniform data structures, posing difficulties for web crawlers in collecting readable data due to diverse formats such as HTML, CSS, JAVA, or XML. The constant update of new data to ensure current information and downloading all pages can lead to increased internet traffic (Palani, 2023). Metadata associated with reviews, comments, and content on review sites includes information like locations, ratings, and reviewers' review counts by users. The nature of this data varies depending on the type of reviews, whether for a service or a product.

Data Wrangling and Corpus Analysis for Robust Data Summarisation

Preparing text data for sentiment analysis in product reviews involves crucial preprocessing steps, including cleaning, normalization, and feature extraction. These steps are vital for eliminating inconsistencies, correcting errors, and discarding unnecessary information, resulting in a standardized and consistent format. This facilitates numerical feature extraction, representing the text data suitable for machine learning (ML) models.

In this study, specific cleaning steps were applied to the Product Reviews data, focusing on the 'Date of Experience' and 'Rating' variables. The goal was to achieve consistent formatting, ensuring the first character in the rating column represented the rating value. Additionally, integers preceding the word 'posts' were extracted, and column names were standardized to match those in the Trustpilot review data. This process aimed to create uniformity and facilitate the merging of the data into a processed_data.csv file.

```
### Cleaning data for product reviews
#%%
import pandas as pd

# Read data from CSV file
df = pd.read_csv("productreview_reviews_all_info1.csv")

# Change rows where "Date of Experience" is not in the expected format
df = df[df["Date of Experience"].str.match(r'\d{4}-\d{2}-
\d{2}T\d{2}:\d{2}:\d{2}.\d{3}Z')]

# Format the Date of Experience column
df["Date of Experience"] = pd.to_datetime(df["Date of
Experience"]).dt.strftime("%d/%m/%Y")
```

```
# Keeping first character in rating column
df["Rating"] = df["Rating"].str[0]

# Extract integers before the word "posts" in the Additional Info column
df["Additional Info"] = df["Additional Info"].str.extract('(\d+) posts')

# Rename columns
df.rename(columns={"Review Content": "Reviews", "Additional Info": "Reviews
Count"}, inplace=True)

# Save the modified DataFrame back to the same CSV file
df.to_csv("productreview_reviews_all_info1.csv", index=False)

# Display the modified DataFrame
print(df)
```

In the original data obtained from Trustpilot, the columns underwent renaming, and the values for both the rating and the reviewers' count were extracted from the character strings. The reviews count value represents the number of reviews a particular reviewer has conducted. This process aimed to enhance clarity and usability in the dataset, ensuring that relevant information could be readily accessed and analysed for further investigation.

```
### Cleaning data for Trustpilot reviews
#%%
import pandas as pd

# Assuming df is your DataFrame
df = pd.read_csv("trustpilot_reviews_all_info1.csv")

# Rename columns
df = df.rename(columns={"Review Content": "Review", "Reviewer Reviews
Count": "Reviews Count"})

# Extract the characters after "Rated" and before "out" in the "Rating"
column
df['Rating'] = df['Rating'].str.extract(r'Rated (.*?)(?= out)',
expand=False).str.strip()

# Remove 'reviews' or 'review' from each row in the 'Reviews Count' column
df['Reviews Count'] = df['Reviews Count'].apply(lambda x:
x.replace('reviews', '').replace('review', '').strip())

# If you want to save the changes to the original CSV file
df.to_csv("trustpilot_reviews_all_info1.csv", index=False)
```

In the text preprocessing phase, the techniques applied were in line with the recommendations of Perwej et al. (2022) and Vaghela (2016).

- One specific technique involved removing non-ASCII characters, which encompass accented letters, emojis, and special symbols. This step aimed to ensure that the text data was cleaned of any characters that might introduce noise or inconsistencies, promoting a standardized and more manageable dataset for subsequent analysis.

```
  # Remove non-ASCII characters
    text = re.sub(r'[^\x00-\x7F]+', '', text)
```

- By converting all words to lowercase, the analysis becomes case-insensitive, reducing the complexity and facilitating a more accurate understanding of the text content.

```
# Lowercasing
  text = text.lower()
```

- Removing punctuation and special characters helps in creating a cleaner and more focused text dataset, allowing subsequent analysis to focus on the essential linguistic elements.

```
# Removing Punctuation and Special Characters
  tokens = [re.sub(r'[^\w\s]', '', token) for token in tokens]
```

- By tokenizing the text, the analysis gains granularity, allowing for a more detailed examination of the language structure. Each token represents a distinct unit of information, facilitating subsequent natural language processing (NLP) tasks such as sentiment analysis or feature extraction. This step contributes to a more refined and structured representation of the text data.

```
# Tokenization
  tokens = word_tokenize(text)
```

- By removing stop words, the text data is refined, and the focus shifts to more meaningful words that are likely to convey important information for sentiment analysis or other natural language processing tasks. This step helps reduce noise and enhances the efficiency of subsequent analyses.

```
# Removing Stop Words
  stop_words = set(stopwords.words('english'))
  tokens = [token for token in tokens if token not in stop_words]
```

- A further method employed to derive the base form of words involved the use of Lemmatization, aimed at enhancing the performance of Natural Language Processing (NLP).

```
# Lemmatization
  lemmatizer = WordNetLemmatizer()
  tokens = [lemmatizer.lemmatize(token) for token in tokens]
```

- Performing Part of Speech (POS) tagging from NLTK library involving part of speech on a list of token and then using the result to perform Named Entity Recognition. The named entities were used to create word cloud to enable extract information and provide content summarisation to understand the reviewers for entities provided in the text data.

```
# Part-of-speech tagging
  tagged = nltk.pos_tag(tokens)
  print(f"POS Tags: {tagged[:10]}")
# Named Entity Recognition (NER)
```

```
    entities = nltk.chunk.ne_chunk(tagged)
    print("\nNamed Entities:")
    entities.pprint()
```

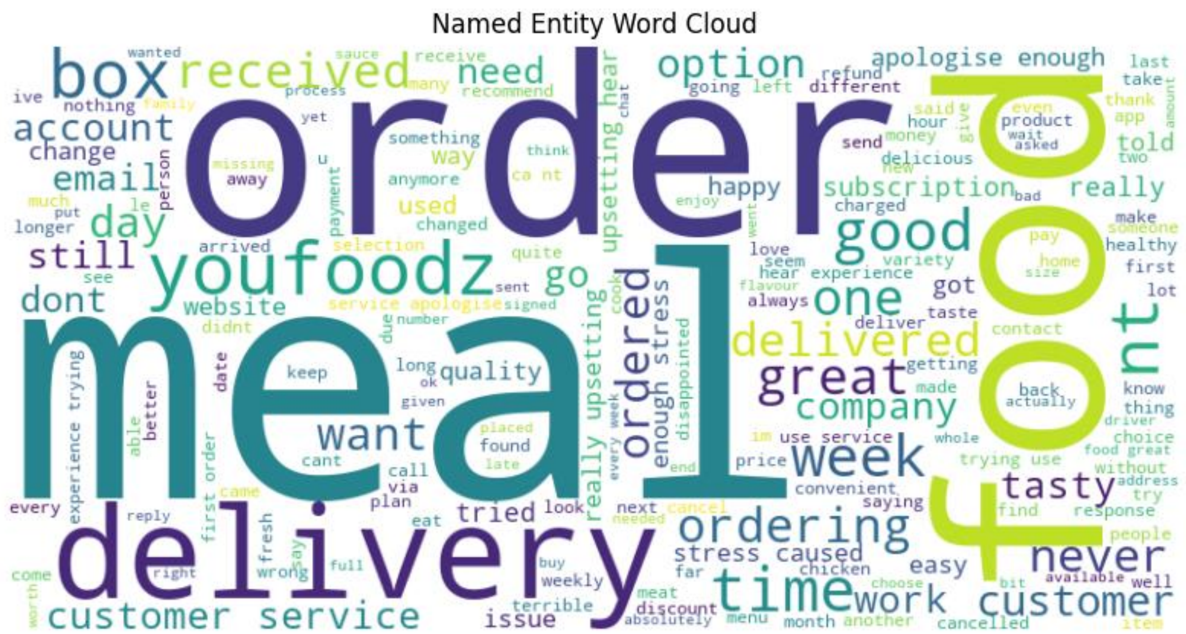


Figure 1: Named entity Summarisation using word cloud

- Additionally, in the process of reconstructing the text document, the tokens were combined into a single string through joining.

```
    # Join tokens back into a single string
    preprocessed_text = ' '.join(tokens)
```

The normalization feature of word embedding was implemented using transformers from the Hugging Face transformer library, a community-owned application builder in the field of machine learning. Word embedding involves converting words into high-dimensional vectors, capturing relationships with other words and aiding sentiment analysis models by preserving the meaning of words in a text. The specific transformer utilized for text classification in this study was the RoBERTa (Bidirectional Encoder Representations from Transformers) model. RoBERTa is an advanced transformer-based language model, an improved version of BERT, designed to process input words and generate contextualized representations of words within a sentence (GeeksforGeeks, 2023). It utilizes dynamic word embeddings to optimize training and hyperparameters, enhancing its language understanding through a large pre-trained dataset.

```
# Load the RoBERTa model and tokenizer
MODEL = "cardiffnlp/twitter-roberta-base-sentiment"
tokenizer = AutoTokenizer.from_pretrained(MODEL)
model = AutoModelForSequenceClassification.from_pretrained(MODEL)
```

An empty dictionary (‘res’) was created to store the results of the sentiment analysis.

```
# Initialize an empty dictionary to store results
```

```
res = {}
```

The sentiment was classified as ‘neg’ for negative and ‘pos’ for positive scores via loop through the data frame with the output shown below.

```
                                                 roberta_pos  \
0     food delivered promised still chilled got home...     0.980452
1     lot different meal choose eat balanced diet  m...     0.333405
2     meal tasty easy prepare  expect customer long ...     0.716518
3     food taste good  unlike pre packed brand  deli...     0.939826
4     ordered  website claimed would store payment m...     0.025792
...                                                 ...          ...
1009  sorry found menu little le snazzy  remember al...     0.520778
1010  thought signing flexible delivery food needed ...     0.101341
1011  truly sorry see experience youfoodz satisfactory      0.037064
1012  best pre cooked meal  husband love also  would...     0.964150
1013  good news finally finished yfood im pensioner ...     0.395334

      roberta_neg Sentiment
0        0.002852       pos
1        0.014368       neg
2        0.006205       pos
3        0.005510       pos
4        0.696194       neg
...           ...       ...
1009     0.047972       pos
1010     0.410569       neg
1011     0.698332       neg
1012     0.002700       pos
1013     0.208618       neg

[1014 rows x 4 columns]
```

The columns include: 'Index', 'roberta_neg', 'roberta_pos', 'processed_text', 'Date of Experience', 'Reviews Count', 'Rating', and 'Sentiment'. The statistics show ‘review count’ has a mean of 3837 with a minimum of 1 and maximum of 188. The size of corpus is considered as small length document with 1014 rows of entries and 8 columns.

```
             Index  roberta_neg  roberta_pos  Reviews Count       Rating
count  1014.000000  1014.000000  1014.000000     828.000000  1014.000000
mean    506.500000     0.348624     0.388702       3.836957     2.785996
std     292.860888     0.350876     0.395619      12.004181     1.678669
min       0.000000     0.001572     0.002472       1.000000     1.000000
25%     253.250000     0.011107     0.028699       1.000000     1.000000
50%     506.500000     0.250432     0.156936       2.000000     3.000000
75%     759.750000     0.685188     0.863559       3.000000     5.000000
max    1013.000000     0.977692     0.990329     188.000000     5.000000
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1014 entries, 0 to 1013
Data columns (total 8 columns):
```

The distribution of sentiment was calculated as 617 negative reviews and 397 positive reviews.

```
# Check the distribution of sentiment
sentiment_distribution = df['Sentiment'].value_counts()
print(sentiment_distribution)
```

The Figure 2 below for distribution of reviews over time highlights reviews have increased august 2022 and has become a frequent method og conveying opinions and reviews on the web. The highest review count was seen around month of August 2022 and August 2023. Furthermore, the reviewers count is seen between 0-25 reviews as the most common in the dataset with a frequency of 800 counts. Moreover the distribution of ratings show rating of 1 being the most common followed by 5 and 4. This portrays the negative reviews for the Youfoods product is common.

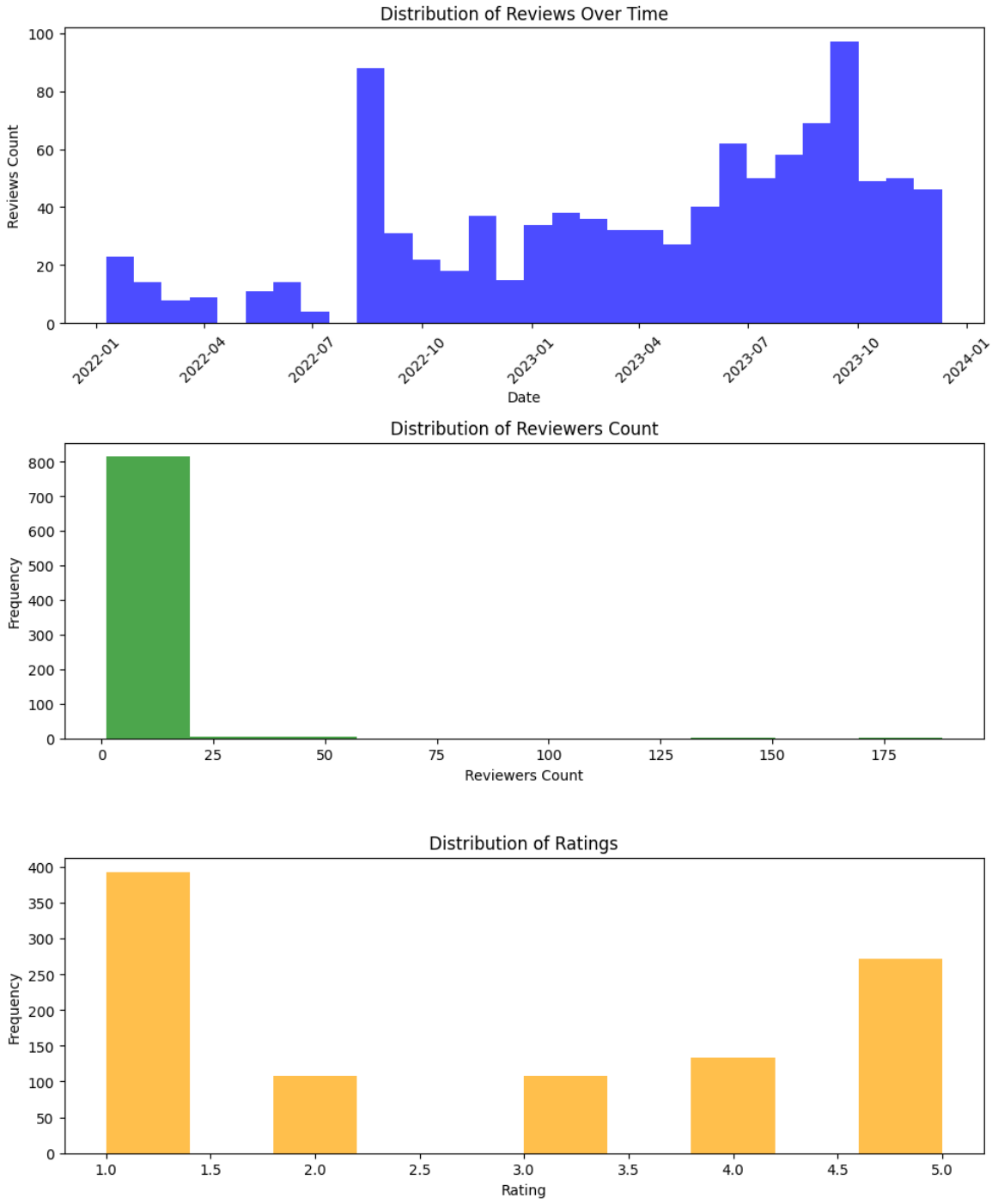


Figure 2: Distribution of Reviews over time, Reviewers count and ratings

The Figure 3 shows the most common used word in the reviews is meal followed by order which is likely as it's a food review for the service provided by the Youfoods company. The two least frequent words are like and delivered.

```
Top 10 Most Frequent Words:
[('meal', 892), ('order', 519), ('food', 416), ('delivery', 382),
('youfoodz', 290), ('nt', 281), ('box', 249), ('week', 244), ('service',
241), ('time', 211)]
```

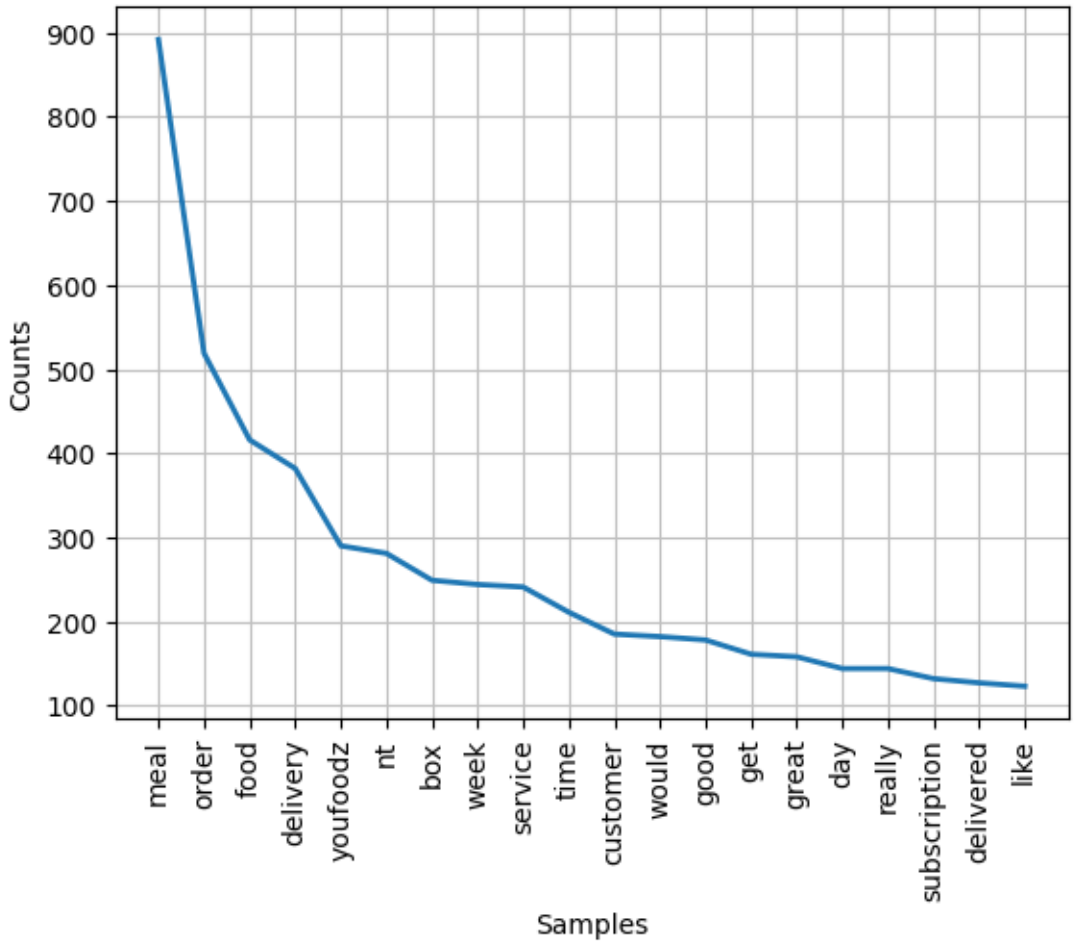


Figure 3: Text with most frequent used words.

Support Vector Machine Model

Machine Learning Structure

Several studies highlight the effectiveness of Support Vector Machine (SVM) as the best machine learning model due to its ability to establish hyperplanes that separate two classes, negative and positive (Putri et al., 2023; Zhang Z et al., 2011). The hybrid approach of combining transformers for text classification with SVM for machine learning has proven to yield accurate results. For text classification, a feature engineering technique involves using TF-IDF (Term Frequency-Inverse Document Frequency) to represent text data as numerical

vectors, enabling machine learning models to comprehend the information. Label encoding was performed by converting sentiment labels(positive/negative) into the numerical format. The dataset is divided into training and test data, where the training data is used to train the model, and the test data evaluates its performance on unseen data.

```
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.svm import SVC
from sklearn.metrics import roc_curve, auc, precision_recall_curve,
classification_report, accuracy_score, f1_score, precision_score,
recall_score
import matplotlib.pyplot as plt

# Extracting features and labels
X = results_df['processed_text'].values  # Features (processed_text)
y = results_df['Sentiment'].values  # Labels (Sentiment)

# Splitting the 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)

# Text vectorization using TF-IDF
vectorizer = TfidfVectorizer(max_features=1000)
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)
```

This study focuses on optimizing SVM performance through grid search techniques. Cross-validation is employed to further enhance accuracy, and evaluation metrics such as precision, recall, F1 score, and accuracy are calculated to assess the proposed techniques' performance.

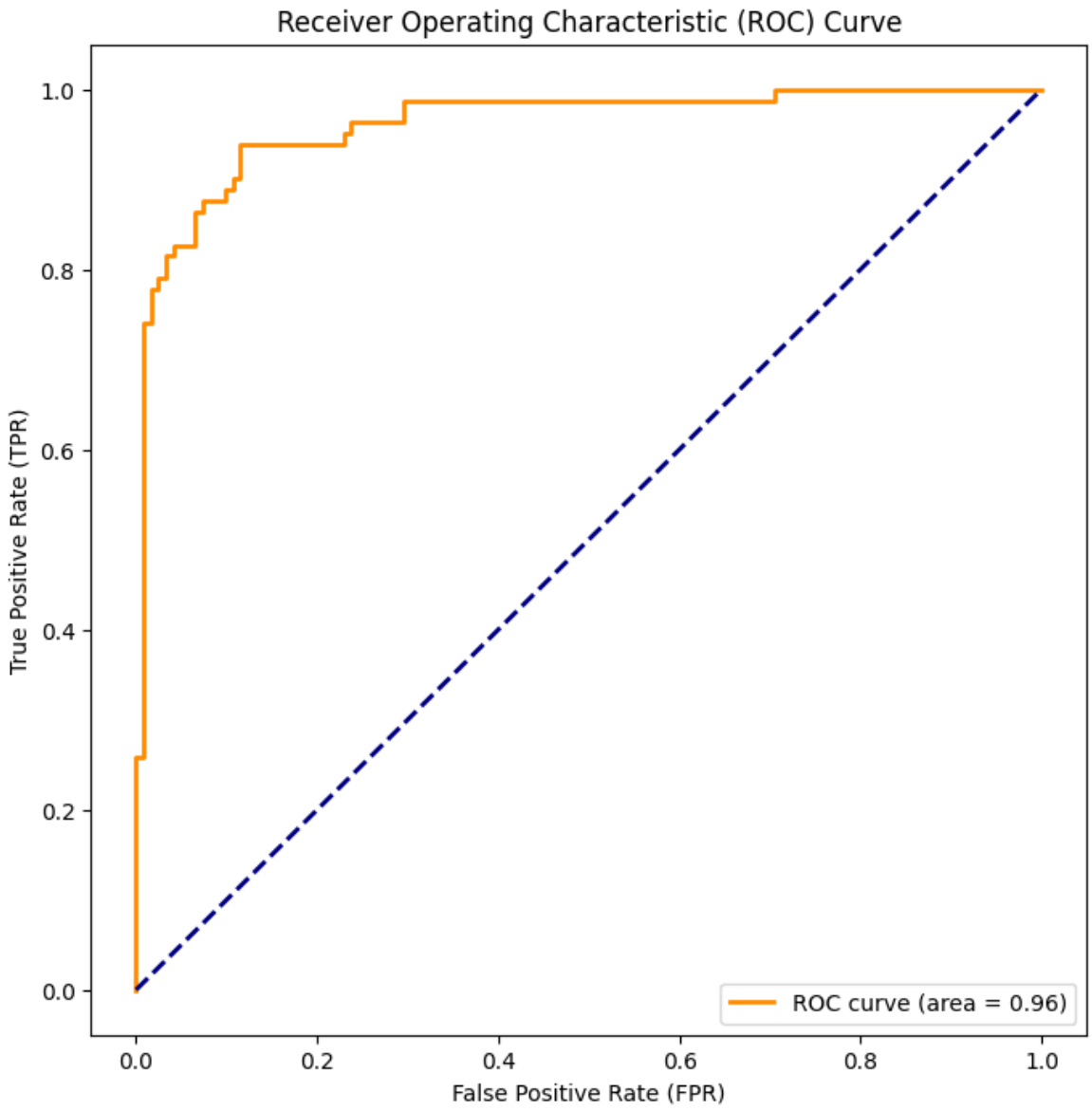
In the training phase, the model learns to differentiate between positive and negative sentiments based on text-sentiment patterns in the training data. To improve model performance further, hyperparameter tuning is conducted, adjusting the kernel type for decision boundary complexity and optimizing the regularization parameter (c) to achieve an optimal decision boundary and correctly classify training points. Model testing is crucial to generalize new text samples and accurately classify sentiments, identifying issues like overfitting and underfitting.

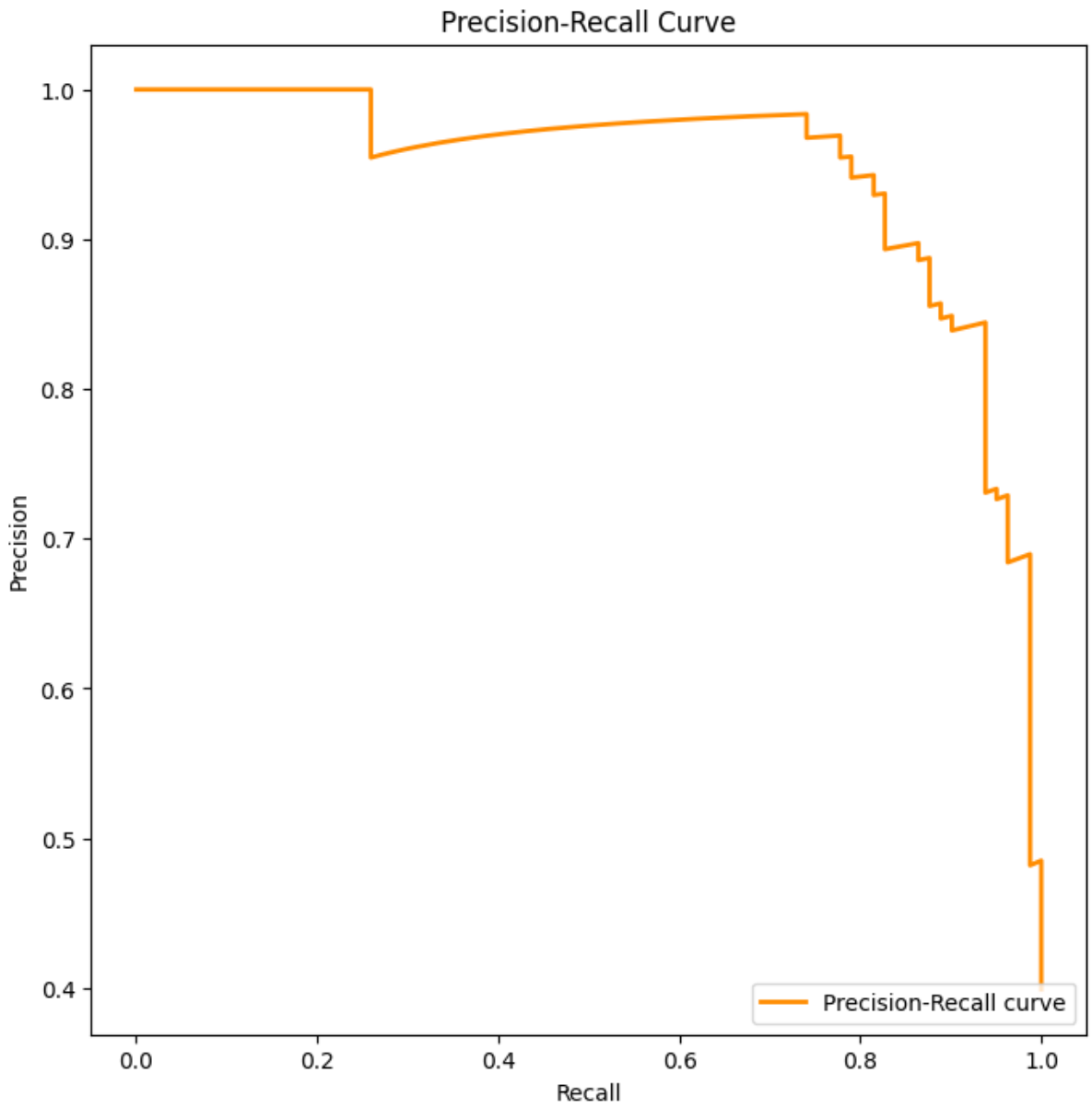
```
# Define the hyperparameter grid
param_grid = {
    'C': [0.1, 1, 10, 100],
    'kernel': ['linear', 'rbf', 'poly'],
    'gamma': ['scale', 'auto', 0.1, 0.01],
}

# Perform Grid Search
grid_search = GridSearchCV(svm_model, param_grid, cv=5, scoring='accuracy')
grid_search.fit(X_train_tfidf, y_train)

# Print the best hyperparameters
print("Best Hyperparameters:", grid_search.best_params_)

# Convert 'neg' and 'pos' labels to 0 and 1 for predicted labels
y_pred_binary = (y_pred == 'pos').astype(int)
```

The sentiment analysis assesses the classifiers' performance using precision, recall, F1 score, and accuracy metrics (Umarani et al., 2021). Various extensions, such as soft margin classifiers, are available to enhance efficiency and adaptability, classifying most data while ignoring outliers and noisy data (Ahmad et al., 2017). The best defined hyperparameter using paragrid highlighted C as 1, gamma to scale and kernel as linear. The results show class 0 for negative class and 1 for positive class. According to the results below in summary the model shows good performance based on the recall, precision, F1 score with a high accuracy. In instances precision can be prioritised over recall. For example, when the prediction for positive sentiments is required over capturing as many positive sentiments like in recall. In this study high F1 score depicts balance between the precision and recall.

```
Best Hyperparameters: {'C': 1, 'gamma': 'scale', 'kernel': 'linear'}
Classification Report:
              precision    recall  f1-score   support

           0       0.90      0.93      0.92       122
           1       0.90      0.85      0.87        81

    accuracy                           0.90       203
   macro avg       0.90      0.89      0.90       203
weighted avg       0.90      0.90      0.90       203

Accuracy: 0.9014778325123153
F1 Score: 0.8734177215189873
Precision: 0.8961038961038961
Recall: 0.8518518518518519
```

The visualisation was conducted using ROC-AUC and precision recall AUC with values of 0.9620 and 0.9478 respectively. The ROC-AUC suggests a good model performance in terms of specificity and sensitivity across the values. The high precision-recall AUC highlights high precision while getting good recall which is beneficial in cases where positives are not common.


Receiver Operating Characteristic (ROC) Curve
True Positive Rate (TPR)
False Positive Rate (FPR)
0.0
0.2
0.4
0.6
0.8
1.0
ROC curve (area = 0.96)



Evaluation

Sentiment analysis for machine learning models faces various data limitations and sampling biases, including imbalanced classes where there is a bias towards either positive or negative sentiment labels (Sultana el.al, 2019). This imbalance can be mitigated using class weights. Additionally, the scarcity of data for uncommon words in sentiment analysis poses a challenge, leading to incorrect expressions for sentiments and context-related issues. Sampling biases may also arise from reviewers themselves, where the data may not be representative of opinions. Therefore, careful consideration of data collection and biases is crucial.

Furthermore, noisy data containing errors or incorrect information can adversely impact model performance. Robust text preprocessing and cleaning methods are necessary to overcome this issue. Finally, data augmentation techniques can be employed to artificially increase the data size when training data is insufficient. These considerations are essential

for a comprehensive evaluation of the machine learning performance of SVM in sentiment analysis, highlighting the significance of addressing data limitations and biases to develop a reliable sentiment analysis tool. The sentiment analysis itself assesses classifier performance using metrics such as precision, recall, F1 score, and accuracy (Umarani et al., 2021).

The study conducted was through Python programming environment, Pycharm and analysis was conducted on CPU and RAM storage using libraries such as Pandas, scikit-learn and nltk using efficient data preprocessing, sentiment analysis model training and evaluation. The parallelism was not required and used such as Hadoop or Spark due to the moderate size of dataset and the capacity to endure using local machine. The chosen method of SVM showed scalability and handling the dataset efficiently. The parallelism would be required for larger datasets thus pertaining high performance and scalability.

References:

- Intellectual property in websites: ownership and protection. (2021, October 19). Pinsent Masons. https://www.pinsentmasons.com/out-law/guides/intellectual-property-in-websites-ownership-and-protection
- Trustpilot Legal - Copyright Dispute Policy. (n.d.). Trustpilot. https://legal.trustpilot.com/for-everyone/copyright
- Palani. (2023, November 14). Exploring web crawling challenges in depth| Quantzig. Quantzig. https://www.quantzig.com/blog/web-crawler-challenges-crawling/
- Top 28 product review websites for online marketers | LiveChat Partners blog. (n.d.). LiveChat Partners Blog. https://partners.livechat.com/blog/best-product-reviews-websites/
- Umarani, V., Julian, A., & Deepa, J. (2021). Sentiment Analysis using various Machine Learning and Deep Learning Techniques. *Journal of Nigerian Society of Physical Sciences*, *3*(4), 385–394. https://doi.org/10.46481/jnsps.2021.308
- Sultana, N., Kumar, P., Patra, M. R., & Alam, S. S. (2019). SENTIMENT ANALYSIS FOR PRODUCT REVIEW. ResearchGate. https://doi.org/10.21917/ijsc.2019.0266
- Yılmaz, B. (2023, December 4). Machine Learning in Sentiment Analysis: 4 use cases. AIMultiple. https://research.aimultiple.com/sentiment-analysis-machine-learning/
- 8 Applications of sentiment analysis. (2020, April 9). MonkeyLearn Blog. https://monkeylearn.com/blog/sentiment-analysis-applications/
- View of Sentiment Analysis using various Machine Learning and Deep Learning Techniques. (n.d.). https://journal.nsps.org.ng/index.php/jnsps/article/view/308/106
- ProductReview.com.au. (2023). Terms of use | ProductReview.com.au. https://www.productreview.com.au/i/terms-of-use
- Kaur, P. (2022). Sentiment analysis using web scraping for live news data with machine learning algorithms. Materials Today: Proceedings, 65, 3333–3341. https://doi.org/10.1016/j.matpr.2022.05.409
- Perwej, Y., Divya, K., Rastogi, N., & Yadav, P. K. (2022). Sentimental analysis on web scraping using machine learning method. ResearchGate. https://doi.org/10.12733/JICS.2022/V12I08.535569.67004

- Naz S Sharan A and Malik N, 2019 Sentiment Classification on Twitter Data Using Support Vector Machine Proc. - 2018 IEEE/WIC/ACM Int. Conf. Web Intell. WI 2018 p. 676–679.
- Zhang Z Ye Q Zhang Z and Li Y, 2011 Sentiment classification of Internet restaurant reviews written in Cantonese Expert Syst. Appl. 38, 6 p. 7674–7682.
- Ahmad, M., Aftab, S., Muhammad, S. S., & Awan, S. (2017). Machine Learning Techniques for Sentiment Analysis: A review. ResearchGate. https://www.researchgate.net/publication/317284281_Machine_Learning_Techniques_for_Sentiment_Analysis_A_Review
- GeeksforGeeks. (2023, January 10). Overview of ROBERTa model. https://www.geeksforgeeks.org/overview-of-roberta-model/
- Putri, D. A., Kristiyanti, D. A., Indrayuni, E., Nurhadi, A., & Muthia, D. A. (2023). Mining Twitter data on Covid-19 for sentiment analysis using SVM algorithm. AIP Conference Proceedings, 2714(1). https://doi.org/10.1063/5.0128833
- Vaghela, V. (2016). A
- nalysis of various sentiment classification techniques. https://www.semanticscholar.org/paper/Analysis-of-Various-Sentiment-Classification-Vaghela-Jadav/dd4c63adb2d67a4e652f6b0389c2f654ce8612b0