Contents

1 Introduction

In this exercise, we are presented with a new commercial tool that collects customer feedback from multiple channels, such as websites, mobile apps, and social media. The Head of Customer Experience recognizes the potential of this data and seeks to gain deeper insights into customer satisfaction levels. The primary objective is twofold: conducting a topic clustering analysis to uncover the main themes and topics discussed in the customer feedback, and developing a supervised sentiment model to classify the sentiment of the reviews.

By leveraging Natural Language Processing (NLP) techniques and machine learning algorithms, we aim to extract meaningful information from the customer feedback data. By identifying the main topics and sentiment patterns, the organization can make informed decisions to enhance customer satisfaction, improve products or services, and ultimately drive business growth.

1.1 Problem Overview

Our primary objective was to gain insights into customer satisfaction levels and understand the key topics discussed in the feedback. We approached this problem in two main tasks: topic clustering analysis and sentiment classification.

For the topic clustering analysis, we utilized Natural Language Processing (NLP) techniques to extract the main themes and topics from the customer feedback data. We employed unsupervised learning algorithms, such as Latent Dirichlet Allocation (LDA), to cluster the reviews based on their content. This allowed us to identify common topics and understand the main areas of focus for customers.

In the second task, we aimed to classify the sentiment of the reviews as positive, negative, or neutral. We adopted a supervised learning approach and experimented with various machine learning models. We trained these models using labelled data, where sentiments were manually assigned to a subset of the reviews. We then evaluated the performance of each model using evaluation metrics such as accuracy, precision, recall, and F1 score.

In summary, our analysis involved employing NLP techniques and machine learning models to extract insights from customer feedback data. Through topic clustering and sentiment classification, we aimed to provide valuable information to guide decision-making and drive improvements in customer satisfaction.

2 Data Preprocessing and Exploration

The dataset contains the following variables:

- datetime: This variable represents the date and time when the customer review was recorded. It provides a timestamp for each review, allowing for temporal analysis of customer feedback.
- review: This variable contains the actual text of the customer review. It includes the feedback provided by customers regarding their experiences with the online banking system, credit cards, car loans, or other relevant aspects.
- rating: The rating variable represents the numerical rating given by customers to their experiences. It indicates the level of satisfaction or dissatisfaction on a scale, where higher values typically correspond to more positive experiences.
- agent_response: This variable captures any response provided by the customer service agent or representative to the customer's review. It indicates if any action was taken by the organization to address the customer's concerns or provide assistance.
- review_id: Each review is assigned a unique review ID. This variable serves as an identifier for individual customer reviews and can be used for tracking and referencing specific feedback instances.

Upon examining the dataset, it is evident that a significant portion of the customer comments has a rating of 5 (Figure 28). This implies that a considerable number of customers express high levels of satisfaction.

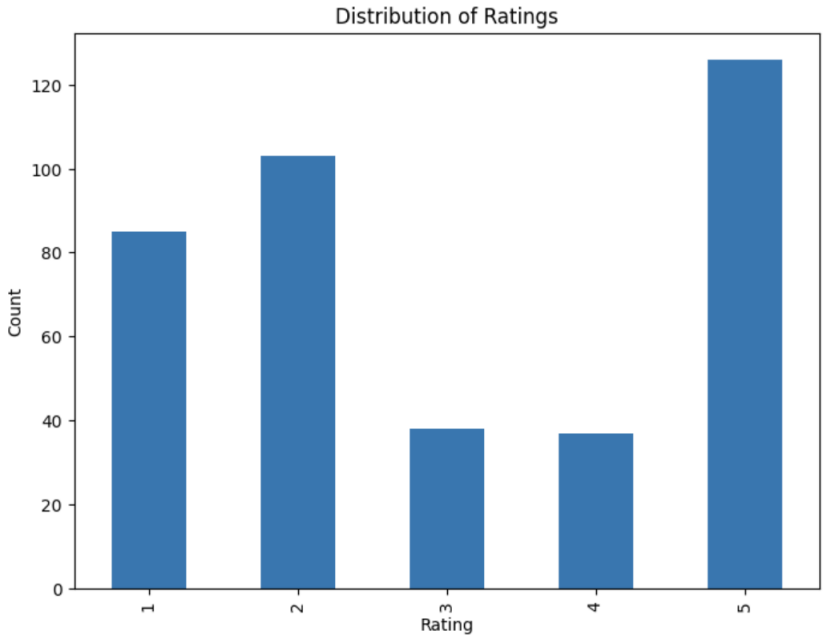

Figure 28: Bar plot of reviews ratings

During the analysis, we conducted an examination of the average ratings per month (Figure 29) and discovered that the month of January stood out with a significantly lower average rating compared to the other months. This finding suggests that there might have been certain factors or circumstances during January that resulted in a decrease in customer satisfaction.

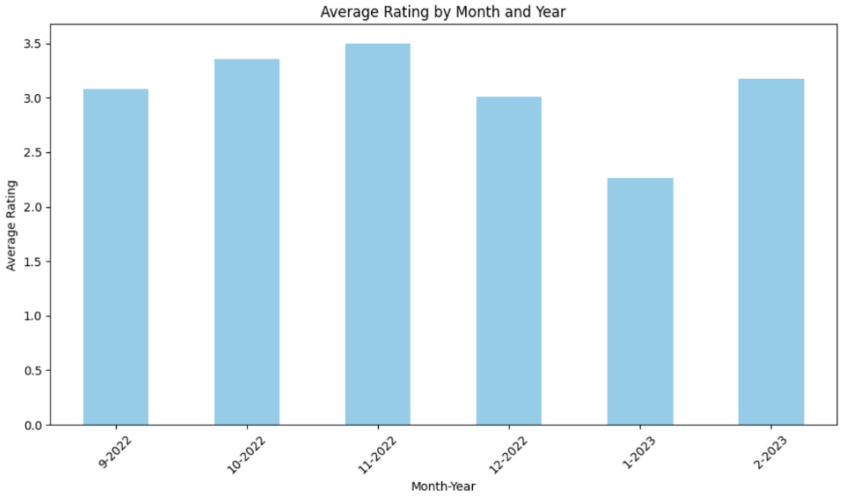

Figure 29: Average reviews rating by month

During the analysis, we also examined the total number of comments received for each month (Figure 30). It was observed that the number of comments experienced a significant increase in the month of February, indicating a potential correlation with a higher level of customer engagement with the company's electronic tools and platforms.

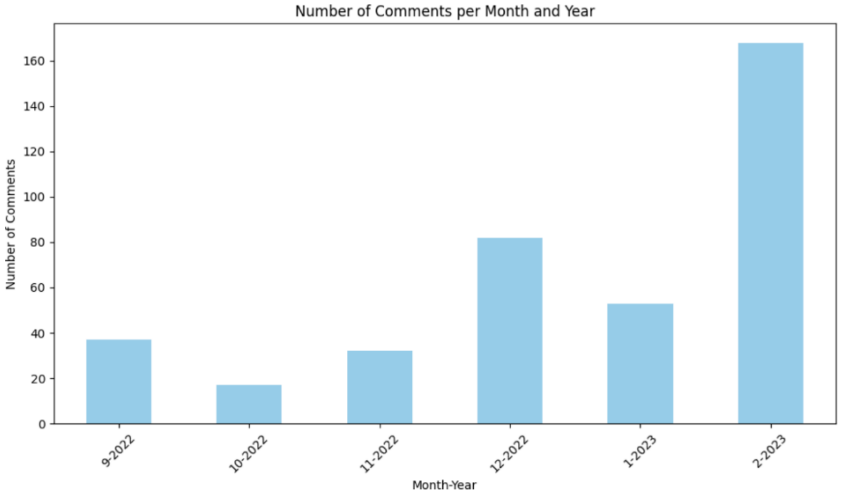

Figure 30: Number of reviews per month

During the analysis of the "review" column, it was observed that there are reviews that consist of a single emoji. These emoji-only reviews pose a unique challenge in terms of their interpretation and analysis. Unlike textual reviews that provide specific insights and feedback, emoji-only reviews convey emotions or reactions in a concise manner. This indicates the need for special handling or preprocessing techniques to appropriately deal with these reviews during sentiment analysis or topic clustering.

We thoroughly checked for missing values in the dataset. Fortunately, there were no missing values in the majority of the columns. However, we did observe missing values specifically in the agent_response column. This can be attributed to instances where agents did not provide a response to certain customer reviews. It is worth noting that the presence of missing values in the agent_response column does not significantly impact our analysis, as our focus is primarily on the customer feedback and sentiment analysis.

During the preprocessing stage of our analysis, we implemented a series of text cleaning techniques to prepare the data for further analysis. The preprocess_text() function was designed to perform various operations on the text data. First, it removed emojis from the text by replacing them with an empty string. Next, contractions were expanded to their full forms. Punctuation marks were removed, and the text was converted to lowercase. The text was then tokenized into individual words using word_tokenize from the NLTK library. Stop words, commonly occurring words with little semantic value, were removed from the tokenized words. Finally, the remaining words were lemmatized to their base form using the WordNetLemmatizer. The preprocessed text was then rejoined into a single string. This preprocessing step ensured that the text data was standardized and ready for further analysis.

In addition to the general preprocessing steps, we also identified specific words that appeared frequently in the text but did not contribute significantly to the analysis. These words, including "bank," "would," "really," "could," and "also," were considered as candidates for removal. To implement this, we created the remove_words() function. This function took the pre-processed text as input and split it into individual words. The specified words, as defined in the words_to_remove list, were then removed from the list of tokens. Finally, the remaining tokens were rejoined into a single string. By removing these specific words, we aimed to eliminate any potential noise or bias they might introduce to the analysis. This allowed us to focus on the most meaningful and relevant information contained within the customer reviews.

3 Topic Clustering Analysis: Identifying Key Themes in Customer Feedback

3.1 Introduction

In this task, our objective is to conduct a topic clustering analysis on the customer feedback data to uncover the main themes and topics discussed in the reviews. By applying advanced natural language processing techniques, we aim to gain deeper insights into the customer satisfaction levels and extract valuable information from the feedback.

3.2 Topic clustering analysis using Latent Dirichlet Allocation (LDA)

The process involves several steps to perform the topic clustering analysis using Latent Dirichlet Allocation (LDA):

A) Count Vectorization: We utilize the CountVectorizer from the scikit-learn library to transform the preprocessed text data into a numerical representation. This step converts the text into a matrix where each row represents a document and each column represents a unique word in the corpus. Count vectorization is a commonly used technique in natural language processing (NLP) that converts text data into a numerical representation that machine learning algorithms can understand. It is a foundational step in many text analysis tasks, including topic modelling, sentiment analysis, and document classification. Count vectorization involves the following steps:

- Tokenization: The text is divided into individual words or tokens, such as word or n-gram tokenization, to create meaningful units for analysis.
- Vocabulary Creation: A vocabulary is formed by collecting unique words or tokens from the text, where each word becomes a feature in the numerical representation.
- Counting: The count vectorization process tallies the occurrences of each word in the vocabulary for each document in the text data.
- Vectorization: The counts are converted into numerical vectors. Each document is represented as a vector with a length equal to the vocabulary size, and the count of each word in the document becomes the corresponding vector element's value.

B) Corpus Generation: The count matrix is then converted into a gensim corpus, which is a specific format required by the LDA model. This transformation allows us to feed the count matrix into the LDA model for topic modelling.

C) Dictionary Creation: We create a dictionary that maps word IDs to their respective words. This dictionary is used to interpret the results of the LDA model by associating topic IDs with their corresponding words.

D) LDA Model Training: The LDA model is trained using the gensim library. We specify the number of topics we want to identify and pass in the corpus and dictionary. The LDA model learns the underlying topic structure in the customer feedback data. Latent Dirichlet Allocation (LDA) is a generative probabilistic model that allows us to discover latent topics within a collection of documents.

E) Top Words Extraction: Finally, we extract the top words for each topic. We define the number of top words to display and retrieve the most relevant words associated with each topic. These words provide insights into the main themes and topics discussed in the customer feedback.

By following this process, we can effectively cluster the customer feedback data into topics and gain a deeper understanding of the key themes and topics that emerge from the reviews. These topics can help identify common concerns, sentiment patterns, and valuable insights to inform decision-making and improve the customer experience.

3.3 Topic clustering analysis results

After conducting multiple experiments and iterations, we settled on training the LDA model with 8 topics (Figure 31). The decision to choose this specific number was based on the trade-off between granularity and

interpretability. By experimenting with different numbers of topics, we found that 8 topics provided a reasonable balance between capturing distinct themes in the customer feedback data and ensuring meaningful interpretations of the topics.

```
Topic 1:
['service', 'recently', 'rate', 'loan', 'customer']

Topic 2:
['app', 'mobile', 'using', 'banking', 'service']

Topic 3:
['notification', 'make', 'transaction', 'push', 'finance']

Topic 4:
['notification', 'push', 'account', 'saving', 'banking']

Topic 5:
['app', 'account', 'notification', 'customer', 'issue']

Topic 6:
['apps', 'app', 'make', 'like', 'feature']

Topic 7:
['loan', 'installment', 'order', 'standing', 'failed']

Topic 8:
['inflation', 'app', 'support', 'agent', 'banking']
```

Figure 31: Topics extracted from the clustering analysis

In this part of the analysis, we focused on assigning topics to the customer feedback documents using the trained LDA model. After transforming the sparse count matrix into a dense matrix, we iterated through each document to determine its dominant topic.

To achieve this, we first extracted the word frequencies in the document, considering only the words that occurred more than zero times. We then obtained the topic distribution for the document by using the LDA model's get_document_topics function. The topic distribution was sorted in descending order based on the probability of each topic.

We extracted the dominant topic and its corresponding probability for each document in the corpus using the trained LDA model. We initialized two empty lists, 'dominant_topics' and 'topic_probabilities', to store the dominant topic and its probability for each document. We then iterated over each document in the corpus and retrieved the topic distribution using the get_document_topics method provided by the LDA model.

Next, we sorted the topic distribution in descending order based on the probability of each topic. The dominant topic and its corresponding probability were obtained from the first element of the sorted topic distribution. These values were appended to the respective lists.

A plot was created to helps us observe the distribution of probabilities across topics (Figure 32). In this case, all topics (except topic one) have probabilities higher than 0.7, indicating that our method is robust and consistent in assigning high probabilities to the dominant topics. This information is valuable in evaluating the effectiveness of the topic modelling approach used in analysing the customer feedback data.

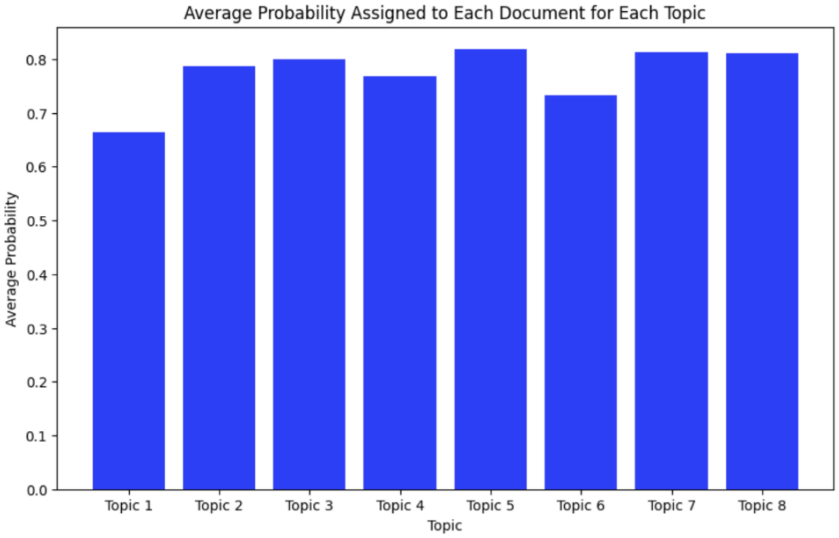

Figure 32: Bar plot of average probability for each topic

A bar plot was created showing the number of documents assigned to each dominant topic (Figure 33). The plot provides insights into the distribution of documents across topics and highlights the topic with the highest number of assigned documents. By analysing the plot, we can observe that the majority of the reviews belong to Topic 0, while Topic 2 has the lowest number of assigned documents. This information suggests that a larger proportion of the reviews are associated with Topic 0, while Topic 2 has fewer reviews. This insight helps in understanding the distribution of reviews among different topics and provides valuable context for further analysis and decision-making.

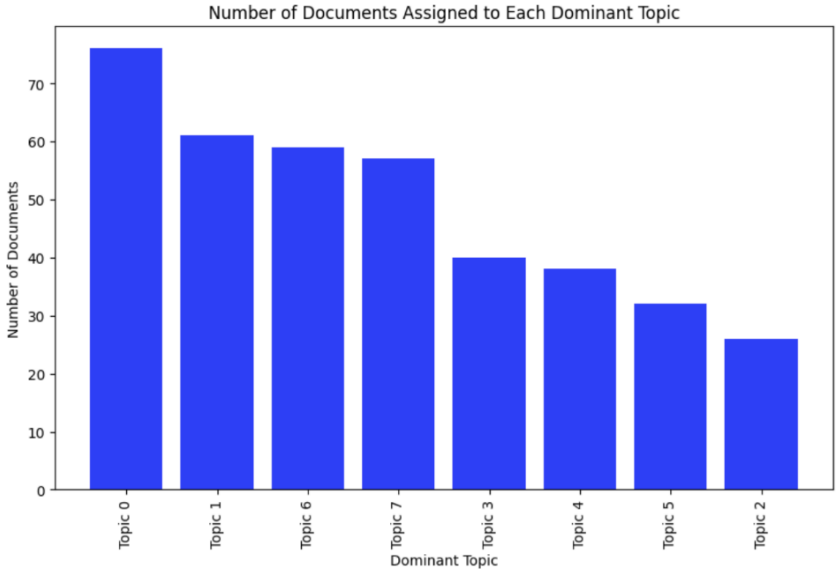

Figure 33: Number of reviews for each topic cluster

Next, we extract the word frequencies for each topic from the topic model. These frequencies indicate the importance of each word within a specific topic. We create a word frequency dictionary using these frequencies. Word cloud image was generated based on the word frequencies. The resulting word cloud visually represents the most significant words in each topic, with word size corresponding to frequency.

When examining the word cloud (Figure 34), we can observe that the most significant words in Topic 1 are related to "loan" and "rate". In Topic 2, the prominent word is "app", and Topic 8 is associated with the term "inflation". A very common word is the app. It seems that customers express comments and opinions about the new app etc.

Figure 34: Word cloud for each topic

By performing similar analyses, we can extend our approach to encompass all the reviews collectively and gain a broader understanding of what matters most to the customers. From the word cloud image, we can observe that the prominent keywords include "app," "rate," "inflation," "call center," and others. These findings suggest that customers often mention aspects related to the mobile application, interest rates, inflation rates, and their experiences with the call center. By exploring the word cloud (Figure 35), we can identify the most significant terms and themes that are frequently mentioned, providing valuable insights into the customer's perspective and highlighting areas of focus for further analysis and improvement.

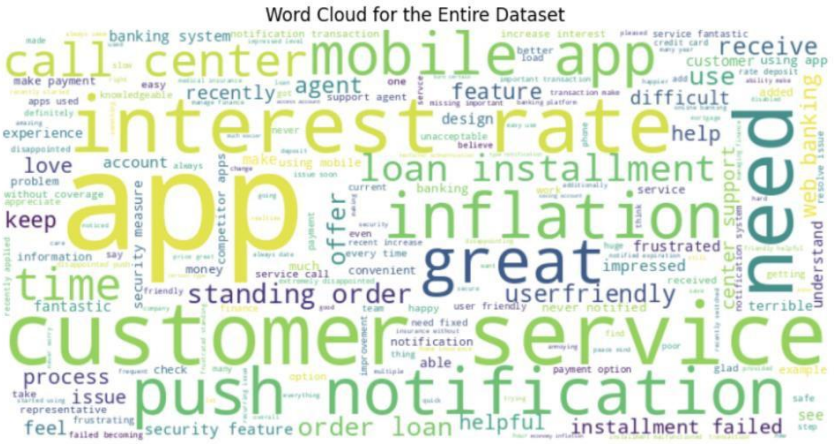

Figure 35: Word cloud for all reviews

Topic coherence is a metric used to evaluate the interpretability and quality of topics generated by a topic modelling algorithm. It measures the semantic coherence of the words within each topic and provides insights into how well the topics capture meaningful themes. In the given code snippet, the c_v coherence

measure is used to calculate the topic coherence. A higher coherence score indicates that the words within each topic are more semantically related and coherent.

Having a coherence score of 0.77 indicates a relatively good level of coherence in the generated topics. This suggests that the topics are capturing meaningful patterns and are semantically coherent. A higher coherence score generally implies that the topics are more interpretable and can provide valuable insights for analysis. However, it is important to note that the interpretability and coherence of topics can be subjective, and the optimal coherence score may vary depending on the specific application and dataset.

4 Exploring Customer Satisfaction through Sentiment Analysis: Unveiling Insights from Online Reviews

4.1 Introduction

Customer satisfaction is a crucial aspect of any business, and understanding customers' sentiments is vital for improving products, services, and overall customer experience. In today's digital age, online reviews have become a valuable source of feedback, providing a wealth of information about customers' opinions and experiences. This analysis aims to delve into the world of online reviews, utilizing sentiment analysis techniques to extract valuable insights. By examining the sentiments expressed in these reviews, we can gain a deeper understanding of customer satisfaction levels, identify areas of improvement, and make data-driven decisions to enhance overall customer satisfaction.

The ratings range from 1 to 5. It is evident from the table (Figure 36) that the majority of customers provided a rating of 5 (126 counts), indicating a high level of satisfaction. Conversely, a considerable number of customers assigned a rating of 2 (103 counts) or 1 (85 counts), suggesting areas where improvements may be needed. Relatively fewer customers provided ratings of 3 (38 counts) or 4 (37 counts), indicating a moderate level of satisfaction. By examining this distribution, businesses can gain insights into the overall satisfaction levels of their customers and focus on addressing any issues that may be causing lower ratings to improve customer experience.

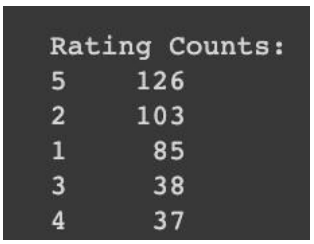

Figure 36: Rating counts

In order to categorize the ratings into distinct groups based on their values, a new column named 'rating_new' was created in the dataset. This column was derived from the existing 'rating' column. The decision was made to assign the labels 'bad', 'ok', and 'good' to the ratings based on specific criteria. Ratings with values 1 or 2 were classified as 'bad', indicating a negative or unsatisfactory experience. Ratings with a value of 3 were categorized as 'ok', representing a moderate or neutral experience. Finally, ratings with values 4 or 5 were labelled as 'good', indicating a positive or satisfactory experience. By creating this new column, we can easily analyse and compare customer feedback based on their overall sentiment, allowing for better understanding of customer satisfaction levels and potential areas for improvement.

4.2 Comparative Evaluation of Models

In this step we aim to assess the performance and effectiveness of different models in predicting the sentiment or rating of customer reviews.

In this analysis, I have utilized a simple classification approach to predict the sentiment of customer reviews based on their cleaned text. However, it is worth mentioning that for this specific problem, I would prefer to use an ordinal model instead of a standard classification model. An ordinal model takes into account the

inherent order and magnitude of the ratings (i.e., "bad," "ok," and "good") and can provide more meaningful insights into the sentiment analysis task. Unlike traditional classification models that assume equal intervals between categories, ordinal models consider the ordinal relationship among the target labels. They estimate the probability of each category relative to the others, taking into account the cumulative probabilities of the lower categories. This allows for a more nuanced analysis, providing insights into the degree or intensity of sentiment expressed in the customer reviews.

Unfortunately, I have not explored specific packages or libraries that offer ready-to-use ordinal models in this code snippet. Nevertheless, further investigation and research are required to identify suitable packages or implementations that support ordinal models. Therefore, my analysis was focused on identifying the best classification model.

The methodology involves several key steps, as outlined below:

A) Data Preparation: Initially, we extract the feature matrix and target variable from the dataset. The feature matrix, denoted as X, is obtained by selecting the 'cleaned_review' column, which contains the pre-processed text data. The target variable, denoted as y, is extracted from the 'rating_new' column, representing the categorized ratings.

B) Train-Test Split: To evaluate the models' performance, we divide the data into training and testing sets. The train_test_split function from the scikit-learn library is utilized for this purpose. We specify a test size of 0.2, indicating that 20% of the data will be reserved for testing, while the remaining 80% will be used for training.

C) Feature Extraction: In order to transform the textual data into a numerical representation suitable for machine learning models, we employ the CountVectorizer. This vectorizer converts the text into a matrix of token counts, representing the frequency of each word in the documents. The training and testing data are separately transformed using the vectorizer.

D) Modelling: The next step involves the selection and evaluation of various classification models. Here, we utilize the LazyClassifier library, which provides a convenient way to compare multiple classifiers simultaneously. We fit the models using the transformed training data and evaluate their performance on the testing data. The resulting F1 scores are calculated for each model.

E) Model Comparison Visualization: To facilitate a comprehensive comparison of the models' performance, we generate a line plot (Figure 37). The F1 scores of the different models are plotted on the y-axis, while the x-axis represents the model names.

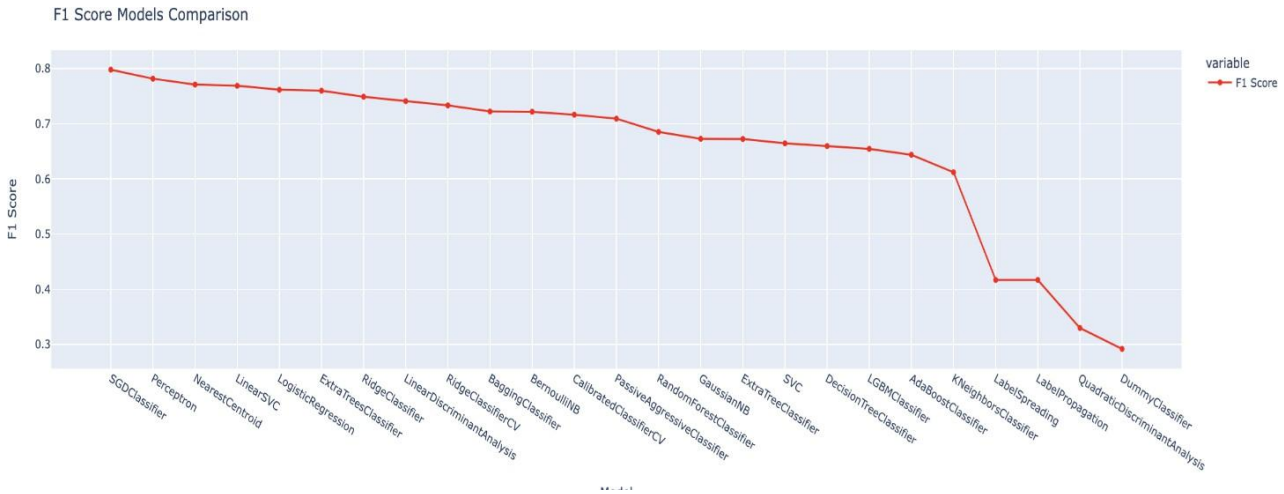

Figure 37: Comparison of classification models for sentiment analysis

By following this methodology and analysing the F1 scores of various logistic models, we can gain insights into their effectiveness in predicting the sentiment or rating of customer reviews. This comparative evaluation helps us identify the most suitable model for our specific task and make informed decisions regarding model selection for future analysis.

The SGDClassifier has the best F1 Score. Logistic regression has also very high F1 Score.

4.3 SGDClassifier

The SGDClassifier is a linear classification algorithm that uses stochastic gradient descent (SGD) to optimize model parameters. It is efficient and scalable, making it suitable for large-scale datasets. The algorithm updates the model's weights iteratively based on the gradient of the loss function, calculated on mini-batches of training examples. The SGDClassifier supports different loss functions for binary and multiclass classification, as well as regularization techniques to control model complexity. It is particularly useful for handling high-dimensional data and online learning scenarios. However, proper hyperparameter tuning is crucial for achieving optimal performance. Overall, the SGDClassifier is a popular choice for linear classification tasks due to its efficiency and scalability.

In this step we performed grid search cross-validation using the SGDClassifier model to find the best combination of hyperparameters. IA parameter grid was defined with different options for the loss function, regularization parameters, and maximum iterations. The GridSearchCV class is then used to iterate through all parameter combinations, evaluating each model's performance using the specified scoring metric. The best model is obtained, and its parameters are extracted. Finally, the best model is used to make predictions on the test set.

To evaluate the performance of a model, one commonly used tool is the confusion matrix. The confusion matrix provides a summary of the predicted and actual values of a classification model. It allows us to analyse the model's performance by examining the number of true positives, true negatives, false positives, and false negatives.

In our Confusion Matrix (Figure 38):

- The diagonal elements represent the correctly classified instances for each class, where 32 "bad" instances, 25 "good" instances, and 2 "ok" instances were accurately predicted.
- The off-diagonal elements indicate misclassifications, revealing areas where the model has made errors in predictions.
- For instance, the model misclassified 9 instances as "bad" that were actually "good," and 6 instances as "bad" that were "ok."

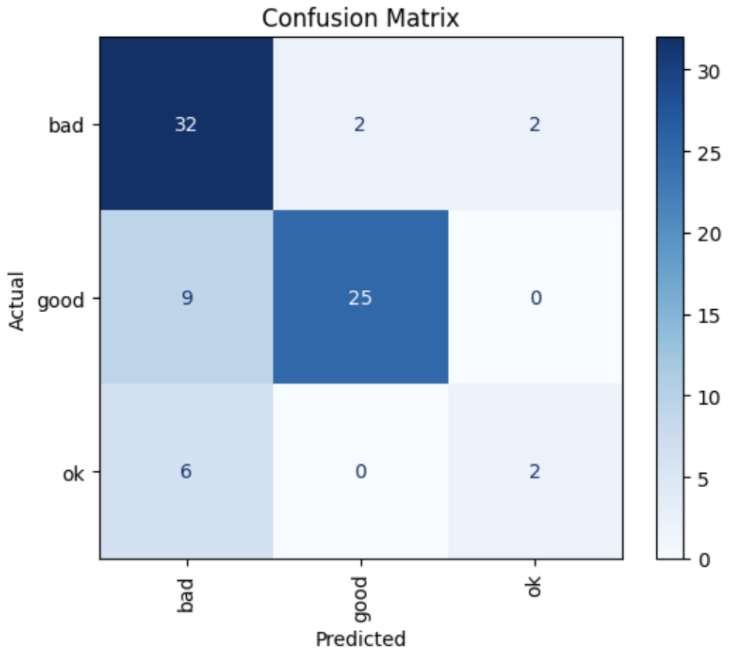

Figure 38: Confusion matrix for SGDClassifier sentiment analysis results

These observations can guide us in understanding the strengths and weaknesses of our model. We can see that the model performs relatively well in classifying "good" instances but faces challenges in distinguishing

between "bad" and "ok" instances. To improve the model's performance, we might consider obtaining more training data or experimenting with different features and algorithms. Regular evaluation and iteration are essential to ensure that our model continues to make accurate predictions for real-world scenarios.

4.4 Logistic Regression Model

Logistic Regression is a popular statistical model used for binary classification. It predicts the probability of an instance belonging to a particular class using the logistic or sigmoid function. The model assumes a linear relationship between the input features and the log-odds of the target class. Logistic Regression is known for its interpretability, as the coefficients provide insights into the feature importance. It can handle categorical and continuous features and can be extended to multi-class problems. However, it assumes a linear relationship, is sensitive to outliers and multicollinearity, and may not perform well in complex datasets. Overall, Logistic Regression is a simple and efficient model for binary classification tasks with interpretable results.

We experimented with the Logistic Regression model to assess its performance and analysed the resulting Confusion Matrix (Figure 39). Comparing this Confusion Matrix with the previous model (SGDClassifier), we observe very similar results. There is only as small improvement in the "good" class, with an increase in true positives (26).

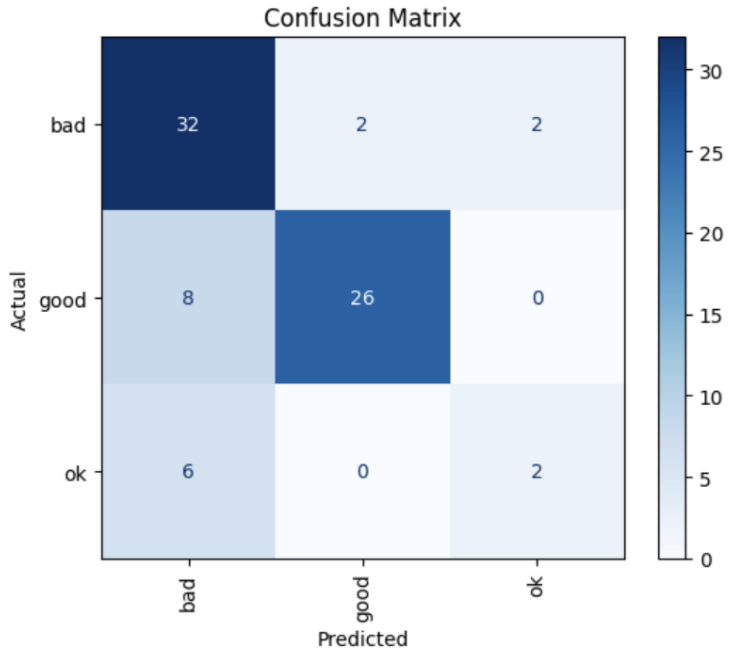

Figure 39: Confusion matrix for Logistic Regression sentiment analysis results

5 Summary and Future Steps

In the topic clustering analysis, we employed techniques such as text preprocessing, feature extraction, and clustering algorithms to identify key topics discussed in the customer reviews. By analysing the textual content, we were able to extract meaningful clusters that represented the main themes emerging from the feedback. This analysis provided valuable insights into the customers' concerns, preferences, and experiences.

For the sentiment classification task, we developed a supervised machine learning model to classify the sentiment of the reviews as positive, negative, or neutral. By training the model on labeled data, we leveraged sentiment analysis techniques to accurately classify the sentiment polarity of customer feedback. This allowed us to gain a comprehensive understanding of the overall sentiment distribution and customer satisfaction levels.

Looking forward, there are several future steps that can be taken to enhance the analysis of customer feedback.

- Firstly, incorporating more advanced topic modelling techniques.

- Furthermore regarding the clustering of topics it would be beneficial to find an automated way to evaluate the clustering results. In this way we would be able to test different number of clusters-topics etc more efficiently.
- The application of sentiment analysis and the classification problem could indeed differ if we had utilized ordinal models or if we had a larger sample size.
- Furthermore, exploring the integration of domain-specific knowledge or pre-trained language models, such as BERT or GPT, can enhance the sentiment classification task. These models have been trained on vast amounts of textual data and can capture complex language patterns and contextual information, leading to improved sentiment classification accuracy.
- Additionally, conducting a thorough analysis of the sentiment misclassifications and investigating potential causes can provide insights for model refinement. Understanding the specific challenges and limitations of the sentiment classification model can guide future improvements in model performance.

In conclusion, this exercise allowed us to delve into the analysis of customer feedback, uncover key topics, and classify sentiment polarity. By further exploring advanced topic modelling techniques, leveraging domain-specific knowledge, and expanding the data sources, we can enhance the accuracy, granularity, and applicability of the analysis. This deeper understanding of customer feedback can inform strategic decision-making, drive improvements in customer experience, and foster customer-centric approaches within the organization.