Πρόγραμμα Μεταπτυχιακών Σπουδών

στην Αναλυτική των Επιχειρήσεων και Επιστήμη των Δεδομένων

Τμήμα Οργάνωσης και Διοίκησης Επιχειρήσεων

<u>Διπλωματική Εργασία</u>

Topic Modeling in Chat Transcripts from an E-Commerce Website

Αναγνώριση Θεμάτων σε Συνομιλίες από Ιστοσελίδας Ηλεκτρονικού Εμπορίου

Ευθυμία Καραμπάση του Σπύρου

Υποβλήθηκε ως απαιτούμενο για την απόκτηση του μεταπτυχιακού διπλώματος στην Αναλυτική των Επιχειρήσεων και Επιστήμη των Δεδομένων

Φεβρουάριος 2024

Abstract

This thesis investigates the application of topic modeling, specifically through the use of Latent Dirichlet Allocation (LDA), to analyze customer service chat transcripts from an e-commerce platform. The aim is to identify latent topics within the chat data that can offer insights into customer behaviors, preferences, and frequently encountered issues.

The methodology includes a comprehensive data preprocessing phase, including the anonymization of personal information, tokenization of text, and removal of stopwords and irrelevant characters to prepare for LDA model training. The LDA model is calibrated across several iterations to identify the most coherent topic distribution, with the optimal number of topics determined by evaluating model coherence scores. Two different models are trained to provide a different number of identified topics. The final selection is based on the balance of detail and coherence in the topics.

The analysis revealed distinct topics that include various aspects of customer interaction, such as payment issues, shipping information, technical support, and product inquiries. These topics not only reflect the diverse range of customer service inquiries but also highlight specific areas for potential improvement in product offerings, delivery, and website functionality.

The findings of this study underscore the utility of topic modeling in discovering valuable insights from textual data, suggesting its applicability in enhancing the strategic decision-making process in the field of e-commerce. This contribution enriches the body of natural language processing applications in business intelligence, proposing an approach to leveraging unstructured data for strategic advantage.

Table of Contents

Tables and Figures

Tables

Figures

Table of Charts

Chapter 1. Introduction

With digital transformation shaping the competitive landscape of businesses, it has become crucial to understand the diverse ways customers interact. This thesis delves into the area of chat data analysis, a field that can enhance our understanding of customer needs and preferences. By analyzing unstructured chat transcripts through advanced topic modeling techniques, such as Latent Dirichlet Allocation (LDA), we aim to uncover the latent thematic structures that exist within. The investigation highlights the importance of integrating chat data analysis within traditional analytic methods to develop customer-centric strategies. As we navigate through the complexities of digital communication, the insights derived from this study offer significant contributions to both academic research and practical applications in the dynamic world of businesses.

Topic modeling, particularly through the application of Latent Dirichlet Allocation (LDA), is the main analytical technique in this study. LDA is widely used in natural language processing and text mining to automatically find hidden themes or topics within a collection of documents. It allows researchers to understand the content of complex datasets that would be challenging to analyze manually. By incorporating LDA into this study, enables us to connect the raw, unfiltered dialogues of customers to quantifiable patterns and topics, helping to understand the hidden feedback from customers' interactions.

Chapter 2. Literature Review

2.1 Topic Modeling

2.1.1 Definition

Topic modeling is a statistical method for finding latent thematic structures within a collection of documents. This allowing the discovery of abstract topics or themes present in the text (Blei, Ng, & Jordan, 2003). Through the analysis of the distribution of words across documents and the underlying topics, this technique organizes textual data into clusters representing coherent themes or subjects (Blei, 2012). It operates on the assumption that each document is a mixture of different topics, and each word's presence is attributed to the document's underlying topics (Blei, 2012; Blei, Ng, & Jordan, 2003).

Topic modeling as a method of unsupervised classification of documents (A Beginner's Guide to Latent Dirichlet Allocation(LDA), 2019) refers to a set of techniques designed to automatically structure, comprehend, query, and condense extensive digital repositories.

Its utility lies in:

1. Revealing hidden themes within a dataset or document collection.
2. Categorizing documents based on the identified themes or topics.
3. Use this categorization to facilitate organization, summarization, and search functionalities within the documents.

2.2.2 Topic Modeling using Python

Python's rich ecosystem offers an array of powerful libraries to apply topic modeling.

- **Gensim** is a prominent library in Python (Rehurek & Sojka, 2010) that provides efficient tools for applying algorithms like Latent Dirichlet Allocation (LDA) and Latent Semantic Analysis (LSA) to enable the identification of latent topics within text.
- **Scikit-learn** is a widely used machine learning library (Pedregosa et al., 2011) that integrates robust implementations of Non-Negative Matrix Factorization (NMF) and other models for topic extraction.
- Additionally, the **NLTK** toolkit (NLTK Project, 2023) within Python facilitates preprocessing steps such as tokenization, stemming, and stop word removal, essential for cleaning and preparing text data before performing topic modeling analysis.

Python's accessibility and the comprehensive functionality offered by these libraries enables practitioners to explore, analyze, and extract insights from diverse textual datasets using effective topic modeling methodologies.

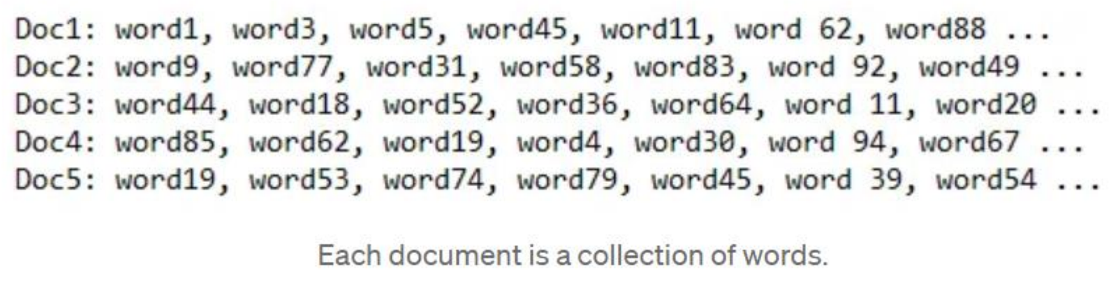
2.2 Latent Dirichlet Allocation (LDA)

2.2.1 Definition

LDA is a probabilistic model introduced by David Blei, Andrew Ng, and Michael Jordan in their seminal paper "Latent Dirichlet Allocation" (2003). In the context of topic modeling, LDA assumes that each document is a mixture of various topics, and each word in the document is attributable to one of the document's topics. It employs a generative process where documents are created based on a set of latent topics, and words are chosen from these topics according to certain probability distributions. LDA utilizes a Dirichlet distribution to model the topic distribution for documents and the word distribution for topics. The model aims to discover the latent topics that best explain the observed document-word relationships. LDA has become a widely used technique for tasks such as document classification, summarization, and information retrieval.

2.2.2 Foundational Principles of LDA

LDA serves as a probabilistic model for discovering latent topics within a corpus of documents (Blei, Ng, & Jordan, 2003). This model assumes that each document is a mixture of a fixed number of topics, and each word's presence in a document is attributed to these latent topics. The generative process of LDA involves the estimation of topic-word and document-topic distributions, providing a coherent representation of the underlying thematic structures within textual data (Blei, Ng, & Jordan, 2003).



```
Doc1: word1, word3, word5, word45, word11, word 62, word88 ...
Doc2: word9, word77, word31, word58, word83, word 92, word49 ...
Doc3: word44, word18, word52, word36, word64, word 11, word20 ...
Doc4: word85, word62, word19, word4, word30, word 94, word67 ...
Doc5: word19, word53, word74, word79, word45, word 39, word54 ...
```

Each document is a collection of words.

Figure 1: Example of document collection

In this example there are 5 documents with their words ordered by frequency of occurrence (A Beginner's Guide to Latent Dirichlet Allocation (LDA), 2019).

Table 1: The probability a word to belong to a topic

	Word1	Word2	Word3	Word4
Topic1	0.01	0.23	0.19	0.03
Topic2	0.21	0.07	0.48	0.02
Topic3	0.53	0.01	0.17	0.04

In the table above each cell contains the probability the word in the corresponding column to belong to the corresponding topic (row).

If the table is sorted the words' probability scores by row descending, the top x words (e.g.: x=10) will represent the topic.

Alternatively, a threshold on the score can be used and every word that has a probability score above the threshold will be a representative of the topic.

2.2.3 Applications and Advancements

LDA models have been applied in various fields according to Jelodar, H. et al. (2019), like linguistic science, political science, medical and biomedical, geographical, software engineering, social network, or crime prediction.

- In linguistic science, LDA is used for visualizing topics, discovering linguistic metaphors, and identifying documents in different languages.
- In political science research, LDA has been used as part of a broader approach that involves dataset selection, policy orientation classifiers, and machine learning techniques like SVM classifiers for analyzing Twitter users' political orientations.
- In medical, Corr-LDA topic model, was applied to mouse model expression datasets to study the issue of human breast cancer.
- In geographical field, an LDA model was proposed, combining multiscale image representation and probabilistic topic modeling, for clustering Very High-Resolution (VHR) satellite images.

- Advanced LDA models with genetic algorithms (GA) are applied in understanding malicious app behavior in the field of software engineering.
- LDA has been applied in a probabilistic topic model to discover emotions in Twitter conversations, allowing for the identification of different emotional aspects in tweets.
- Statistical topic modeling based on LDA applied to identify discussion topics in a city in the United States and these identified topics are then used for standard crime prediction.

2.2.4 Significance and Contributions

Understanding the nuances and complexity of LDA helps in the effective analysis and interpretation of textual data, offering profound insights into the latent structures present in diverse document collections. Moreover, its foundational influence on the evolution of other topic modeling techniques emphasizes its significance as a pivotal algorithm in this domain.

Some of these techniques will be described in detail below.

2.3 Other Methods for Topic Modeling

2.3.1 Hierarchical Dirichlet Process (HDP)

2.3.1.A Definition

The Hierarchical Dirichlet Process (HDP) is a Bayesian non-parametric model that extends the Dirichlet Process to facilitate the discovery of an unbounded number of latent groups or clusters within a dataset. It introduces a hierarchical structure enabling the model to handle an unknown or potentially infinite number of underlying components, allowing for the automatic determination of the appropriate number of clusters from the data. The HDP assigns higher probabilities to clusters with more data points providing a principled approach for modeling complex datasets without requiring the fixed specification of the number of clusters beforehand (Teh et al., 2006).

2.3.1.B Principles of HDP

HDP functions by assuming a potentially infinite number of clusters or topics, allowing for the automatic determination of the appropriate number of clusters within a dataset (Teh et al., 2006). It performs a hierarchical structure to model the distribution of these latent topics, facilitating the sharing of statistical strength across related clusters. Clusters with more statistical strength or influence are those that have accumulated more data points or evidence from the dataset and, therefore, have a higher probability of attracting additional data points, reflecting the “rich-get-richer” phenomenon.

2.3.2.C Applications and Advantages

HDP finds applications in various fields such as natural language processing, document modeling, and image analysis (Teh et al., 2006). It is particularly useful when dealing with datasets where the number of clusters or topics is unknown or changes over time, enabling flexible modeling without requiring a fixed number of components.

The advantages of HDP are:

- Flexibility in Model Complexity: HDP, as mentioned before, has the ability to adapt to the data by inferring the number of clusters or topics without specifying a fixed number which is advantageous in cases with unknown structures (Teh et al., 2006).
- Sharing of Statistical Strength: The hierarchical nature of HDP allows related clusters to share statistical information. This results to an enhanced modeling for complex relationships.

2.3.2.D Limitations and Future Discussions

HDP can be computationally intensive, especially when dealing with large-scale datasets as noted by Teh et al. (2006). Additionally, while offering flexibility, interpreting an unbounded number of latent components might pose challenges in deriving meaningful insights. This means that interpretability is another concern when using such a method.

2.3.2 Latent Semantic Analysis (LSA)

2.3.2.A Definition

Latent Semantic Analysis (LSA), proposed by Deerwester et al. in 1990, is a seminal technique used in natural language processing and information retrieval. Unlike traditional bag-of-words approaches, LSA uses singular value decomposition (SVD) to analyze relationships between terms and documents, aiming to capture the latent structure in a corpus (Deerwester et al., 1990).

2.3.2.B Principles of LSA

LSA creates a matrix where rows represent terms and columns represent documents, containing their frequency. By performing Singular Value Decomposition (SVD) on this matrix reduces its dimensions, LSA captures the underlying semantic relationships between words and documents. This process enables LSA to identify hidden patterns and extract essential semantic information, aiding in tasks like information retrieval, document classification, and automatic summarization (Deerwester et al., 1990).

2.3.2.C Applications and Advantages

LSA has found extensive applications in various fields owing to its capability to capture semantic relationships within textual data. One of these applications is in information retrieval systems, where LSA enhances search accuracy by considering the semantic similarity between query terms and documents (Deerwester et al., 1990) which is related to Search Engine Optimization (SEO) practices. LSA's capability to understand the context and semantics of words helps search engines comprehend the content's underlying meaning and delivering more accurate and relevant results to user queries by considering not only exact word matches but also the semantic relationships and contextual relevance between words and documents.

LSA’s advantage lies in its ability to derive meaningful representations of textual content despite variations in language usage. It limits issues related to exact word matching by finding semantic associations and contextually aligns documents and words, resulting in higher accuracy and relevance of information retrieval systems (Dumais, 2004).

2.3.2.D Limitations and Future Directions

Despite its advantages, LSA has limitations in handling polysemy and synonymy, where words have multiple meanings or different words convey similar meanings. Additional considerations are the scalability and the need for a large training corpus. Recent developments have explored hybrid models combining LSA with other techniques to address these limitations (Landauer & Dumais, 1997).

2.3.3 Non-Negative Matrix Factorization (NMF)

2.3.3.A Definition

Non-negative Matrix Factorization (NMF) is a dimensionality reduction technique that has gained prominence in topic modeling. Unlike Latent Dirichlet Allocation (LDA) and other methods, NMF operates on non-negative data matrices (matrices that contain only non-negative elements or entries), which makes it particularly suitable for tasks where non-negative constraints are vital, such as text and image processing (Lee & Seung, 1999).

2.3.3.B Principles of NMF

Non-negative Matrix Factorization (NMF) deconstructs a non-negative matrix into two lower-rank matrices (A & B), representing the original data's (X) approximate parts. These matrices, when multiplied (product AB), approximate the original matrix (X), facilitating the extraction of latent features or topics. In text analysis, the usage of this unsupervised learning technique aims to represent documents as combinations of distinct topics, with words associated with those topics (Lee & Seung, 1999).

2.3.3.C Applications and Advantages

Text mining and image processing are two of the various domains where NMF demonstrates its effectiveness (What is Non-Negative Matrix Factorization (NMF)?, 2022).

The advantages of NMF are:

- Interpretability, as represents data as a combination of meaningful components or features

- Non-negative constraints, which ensure that extracted features or topics are additive and interpretable, often aligning better with real word scenarios
- Dimensionality reduction, which helps in efficient representation, visualization, and analysis of complex datasets
- Versatility, as it is applicable to various domains and handles different types of data for data analysis and pattern extraction

(Lee & Seung, 1999; Cichocki et al., 2009).

2.3.3.D Limitations and Future Directions

As mentioned by Gun J. et al., 2021, there are several limitations associated with NMF method:

- **Accuracy**: There is a need for improvement in the accuracy and convergence rate of the various NMF methods.
- **Efficiency with Large-Scale Data**: Current NMF methods face challenges in efficiently processing large-scale data. This limitation suggests a need for more optimized algorithms that can handle big datasets without significant performance degradation.
- **Broader Application in Real-World Scenarios**: Although NMF has already been applied successfully in many areas, there is potential for its application in a wider range of real-world scenarios.

2.3.4 Biterm Topic Model (BTM)

2.3.4.A Definition

The Biterm Topic Model (BTM) is a probabilistic topic modeling technique that focuses on capturing co-occurrences of pairs of words, termed "biterms," within a corpus, independently from specific topics. This not only keeps track of how words are related but also lets the model capture different topics within a single document making BTM a powerful and adaptable tool for uncovering hidden topics in short text (Xiaohui Yan, et al., 2013).

2.3.4.B Principles of BTM

Biterm Topic Model (BTM) is based on the idea that biterms, representing pairs of words that appear together in short texts, carrying valuable semantic information (Zhao et al., 2013). It uses a generative process to model these biterms across documents, assigning them to latent topics probabilistically.

2.3.4.C Applications and Advantages

BTM is a suitable method to analyze short texts in social media posts, tweets, or product reviews where the frequency of word occurrences is low (Yan et al., 2013).

Effective extraction of meaningful topics from short texts, especially in cases of sparse data (*datasets where a large portion of the information or variables have zero or near-zero values*) is a notable advantage of BTM (Zhao et al., 2013). It overcomes sparsity issues prevalent in traditional topic modeling techniques when dealing with short texts or limited word occurrences.

2.3.4.D Limitations and Future Directions

Despite its effectiveness, BTM faces areas for improvement, including sensitivity to noise or irrelevant word pairs, and limitations in the need to enhance semantic understanding beyond word co-occurrences - when there isn't enough context available, it becomes more challenging to disambiguate ambiguous words and understand their intended meanings accurately within a short piece of text. (Zhao et al., 2013). These aspects represent active research directions aimed at refining BTM's performance and expand its capabilities to better understand the fundamental meanings and connections among words present in texts. There is also room for improvement in more real-word scenarios like content recommendation, event tracking, etc. (Yan et al., 2013).

2.3.5 Dynamic Topic Models (DTM)

2.3.5.A Definition

Dynamic Topic Models (DTM) are probabilistic models designed to capture topic evolution within a corpus over time. Unlike static models, DTM considers temporal dynamics, allowing topics to change across different time slices (Blei & Lafferty, 2006).

2.3.5.B Principles of DTM

DTM operates under the assumption that topics evolve over time, connecting them across time intervals and recognizing varying topic distributions in documents across different periods (Blei & Lafferty, 2006). This perspective enables the model to track the dynamic nature of topics, allowing researchers to identify the differences between the evolution and interconnectedness of themes over time slices within a given dataset.

2.3.5.C Applications and Advantages

DTM finds utility in analyzing time-stamped document collections like news articles, social media posts, or historical records where content changes over time, enabling tracking of changing topics (Blei & Lafferty, 2006).

Advantages of DTM are:

- Temporal insights, as it provides insights into how topics shift and emerge over time, offering a dynamic view of evolving themes (Blei & Lafferty, 2006).
- Dynamic adaptation as it provides a more accurate representation of evolving themes than static models.

2.3.5.D Limitations and Future Discussions

Computational complexity while scaling to larger datasets or numerous time slices is the main limitation for this method (Blei & Lafferty, 2006). While, enhanced methods for interpreting dynamic topics and visualizing temporal changes are are areas of improvement (Blei & Lafferty, 2006).

Chapter 3. Methodology

3.1 Data Collection

The data utilized in this study are chat transcripts from the chat tool of an electronics e-commerce website and contains textual interactions between customer support agents and website visitors. The data includes conversations throughout 2022 given the chance to collect customer feedback about their online experience and identify possible issues with the website or the products/services.

Privacy and Ethical Considerations:

To uphold privacy standards, all personally identifiable information (PII) has been anonymized. Additionally, ethical considerations have been paramount in this research, ensuring compliance with data protection regulations and guidelines.

3.2 Dataset

The initial dataset consists of 4854 rows and 466 columns. Each row symbolizes a unique chat session, identified by an 'id'. A significant portion of the dataset is devoted to the chat messages themselves, structured in a serialized format. For each message, there is information for the 'sender_t' (*type*, value ‘v’ for visitors and value ‘a’ for agents), 'sender_n' (*name*, which is available only for agents), and 'msg' (message content). This structure captures the flow of conversation. As expected, there are different message counts per chat so there are many null values in the messages columns especially in the last ones.

Table 2. Dataset’s variables

Field Name	**Explanation**	**Type**
id	Unique identifier for each chat record	string
type	Type of chat record, indicating the nature of the interaction (“chat” = a conversation between a site visitor and an agent *or* “ticket” = a summary of interactions following a customer question; it may be assigned to another agent or department for follow-up)	string
pageId	Property ID	string

visitor_name	Visitor’s name; if the visitor name is unknown, the software automatically assigns a random identifier, starting with a “V”	string
visitor_id	Unique identifier for the visitor	string
visitor_email	Visitor’s email; these values are null	string
location_countryCode	Country code of the visitor's location (e.g.: GB, GR, etc.)	string
location_city	City of the visitor’s location	string
messageCount	Number of messages exchanged during the chat	integer
chatDuration	Duration of the chat rounded to seconds	integer
rating	Rating given to the chat session from the visitors (takes the values -1, 0, 1 which follow the logic of negative, neutral, positive)	integer
createdOn	Timestamp indicating when the chat was initiated	date
messages_0_sender_t	Sender type of the first message (take the values ‘v’ for visitor or ‘a’ for agent)	string
messages_0_type	Type of the first message (e.g., text, image)	string

messages_0_time	Timestamp of the first message	date
messages_0_msg	Content of the first message	string
messages_1_sender_t	Sender type of the second message (take the values ‘v’ for visitor or ‘a’ for agent)	string
messages_1_type	Type of the second message (e.g., text, image)	string
messages_1_time	Timestamp of the second message	date
messages_1_msg	Content of the second message	string
…(and so on)	…	…
domain	Domain or website associated with the chat transcript	string
humanId	Sender identifier; available only for type = ‘ticket’ which is an offline message	string
source	Source of the ticket (e.g.: chat); available only for type = ‘ticket’ which is an offline message	string
subject	Subject of the ticket produced by the sender; available only for type = ‘ticket’ which is an offline message	string
requester_email	Email of the requester (if applicable)	string
requester_name	Name of the requester (if applicable)	string

3.3 Dataset Manipulation

3.3.1 Consolidated transcripts

For the scope of topic modeling only the textual data of chat transcripts needed. A new field was created, named ‘full_transcripts’ in the existing dataframe. In the process of generating the 'full_transcripts' field, the objective was to combine the content of individual messages per chat, into a consolidated and coherent transcript. This operation involves parsing through each message within the conversations, extracting the textual content, and systematically appending it to the 'full_transcripts' field. The outcome is a holistic transcript that contains the entire conversation, integrating both the messages from visitors and agents. Afterwards, data anonymization techniques were applied to the ‘full_transcripts’ removing personally identifiable information (PII) to ensure data privacy. The results of this process were stored to a new column named ‘anonymized_data’, which is the final dataset used for topic modeling.

3.3.2 Data anonymization guide

1. **Mask digits**
   Digits are replaced with a generic identifier, '<NUMBER>' contributing to the anonymization of potentially sensitive information like phone numbers.

2. **Mask postal codes**
   Postal codes are replaced with a generic identifier, '<POSTALCODE>', ensuring that location-related information is hidden to product user privacy.
3. **Mask Email Addresses**
   Email addresses are replaced within the text with a standard identifier, '<EMAIL_ADDRESS>' contributing user privacy by concealing specific email details.
4. **Mask dates**
   Replacing various date expressions within the text with standardized identifier '<DATE>' ensures that temporal details are obscured, preserving user privacy during data analysis and sharing. Formats such as 'dd/mm/yyyy', 'dd-mm-yyyy', 'dd.mm.yyyy' and full names and abbreviations of months, along with optional years are replaced with '<DATE>'.

5. **Mask names**

   The process of anonymizing names and surnames within a given conversation is crucial for preserving the privacy of individuals involved. Identified names are replaced with a generic identifier, '<NAME>', ensuring that personal information is hidden in the quest for privacy protection. This step aligns with ethical considerations and privacy standards.

6. **Mask brand identifiers**

   Apart from personal identifiable information (PII), masking the brand identifiers is crucial for data governance purposes. Brand identifiers are replaced with the generic identifier '<BRAND>'.

7. **Mask URLs**

   Masking URLs with '<URL>' protects user and company's privacy by preventing the exposure of specific web addresses and related information, which might contain sensitive or confidential data.

3.4 Data Pre-processing

The *'basic_preprocess_text'* function is designed to perform text preprocessing on the input text.

```
# Define the preprocessing function
def basic_preprocess_text(text):
    """
    This function performs basic text preprocessing on the input text.
    Parameters:
    - text (str): The input text to preprocess.
    Returns:
    - str: The preprocessed text.
    """
    # Convert to lowercase
    text = text.lower()
    # Remove special characters and numbers
    text = re.sub(r'[^a-z\s]', '', text)
    # Tokenize using basic split
    tokens = text.split()
    # Define the list of custom stopwords
    custom_stopwords = set(['hi', 'hello','hey', 'thanks', 'sorry', 'yes', 'ok',
                      'would', 'youre', 'dont', 'im', 'thank', 'pls', 'please',
                      'cheers','also', 'still', 'much', 'good', 'morning',
                      'center', 'centre', 'name', 'uk'])

    # Get the English stopwords along with the custom stopwords
    all_stopwords = set(stopwords.words('english')).union(custom_stopwords)

    # Remove stopwords
    tokens = [word for word in tokens if word not in all_stopwords]

    # Join the remaining tokens into a string
    return ' '.join(tokens)

# Apply the preprocessing function to the dataset and save the results to
#a new column 'preprocessed_text'
chats['processed_text'] = chats['anonymized_data'].apply(basic_preprocess_text)
```

The *preprocessing steps* include:

3.4.1 Lowercasing

```
    # Convert to lowercase
    text = text.lower()
```

Lowercasing (or case-folding) is a text preprocessing technique in natural language processing (NLP), that involves converting all alphabetic characters in a given text to lowercase. The goal of this process is standardizing the text by ensuring uniformity in the representation of words, irrespective of their original casing to reduce the impact of inconsistent casing when building and applying machine learning models. Manning et al. (2009) provide insights into information retrieval, highlighting the importance of lowercasing in mitigating challenges associated with inconsistent casing. Additionally, Jurafsky and Martin (2023) emphasize the role of lowercasing in enhancing generalization in text processing tasks. It is essential, though, to consider specific cases where the casing information might be crucial, depending on the context of the analysis.

3.4.2 Special Characters and Numbers Removal

Eliminating special characters and numbers from the text leads to retaining only alphabetic characters.

```
# Remove special characters and numbers
 text = re.sub(r'[^a-z\s]', '', text)
```

Regular expressions definition:

^: It matches any character that is not in the specified set in brackets.

a-z: Specifies all lowercase English characters from 'a' to 'z'. It matches any single lowercase letter.

\s: This represents whitespace characters, including spaces, tabs, and newline characters.

Combined, [^a-z\s] matches any character that is not a lowercase letter or whitespace.

re.sub(r'[^a-z\s]', '', text): This regular expression is used with the re.sub() function to replace any character in the input text that is not a lowercase letter or whitespace with an empty string, effectively removing all non-alphabetic characters.

3.4.3 Tokenization

```
# Tokenize using basic split
 tokens = text.split()
```

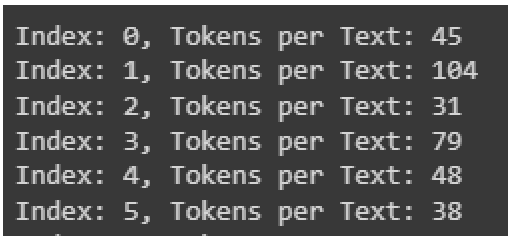
Tokenization is a crucial preprocessing technique in natural language processing (NLP) where the text is broken down into smaller units, known as tokens. These tokens can be words, subwords, or even characters, depending on the granularity chosen for analysis. The primary objective of tokenization is to convert raw text into a format suitable for subsequent linguistic analysis or machine learning tasks. The process facilitates the extraction of meaningful information from the text, and it is essential for tasks like sentiment analysis, language modeling, and information retrieval. Common methods of tokenization include whitespace-based tokenization, which separates words based on spaces, and more advanced techniques such as word segmentation for languages without explicit word boundaries. (Manning, C. D., 2009)

The number of tokens per chat and the overall tokens used in the model, are calculated below:

```
# Define the preprocessing function
def print_metrics(df):
    overall_tokens = 0 # Initialize a counter for the total number of tokens in the dataset
  # Iterate over each row in the DataFrame
    for index, row in df.iterrows():
      # Split the preprocessed text in the 'processed_text' column into tokens (words)
       tokens = row['processed_text'].split()
      # Count the number of tokens in this text
       num_tokens = len(tokens)
      # Print the index of the current row and the number of tokens it contains
       print(f"Index: {index}, Tokens per Text: {num_tokens}")
      # Add the number of tokens in the current row to the overall token count
       overall_tokens += num_tokens
```

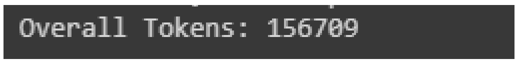
```
# After iterating through all rows, print the total number of tokens found

print(f"Overall Tokens: {overall_tokens}")
```

The results are as below:



```
Index: 0, Tokens per Text: 45
Index: 1, Tokens per Text: 104
Index: 2, Tokens per Text: 31
Index: 3, Tokens per Text: 79
Index: 4, Tokens per Text: 48
Index: 5, Tokens per Text: 38
```

Figure 2: Number of Tokens per Chat Transcript



```
Overall Tokens: 156709
```

Figure 3: Overall Number of Tokens Added to the Model

3.4.4 Stopword Removal

Removing common English stopwords (such as 'the', 'and', 'is', etc.) as well as a set of custom stopwords ('hi', 'hello', 'thanks', etc.) to filter out non-informative words.

```
# Define the list of custom stopwords
    custom_stopwords = set(['hi', 'hello','hey', 'thanks', 'sorry', 'yes', 'ok',
                   'would', 'youre', 'dont', 'im', 'thank', 'pls', 'please',
                   'cheers','also', 'still', 'much', 'good', 'morning',
                   'center', 'centre', 'name', 'uk'])

    # Get the English stopwords along with the custom stopwords
    all_stopwords = set(stopwords.words('english')).union(custom_stopwords)

    # Remove stopwords
    tokens = [word for word in tokens if word not in all_stopwords]
```

Stopwords are an important aspect of text preprocessing in natural language processing (NLP). These are words that are commonly used in a document but are often considered irrelevant in the context of text analysis due to their high frequency and low informativeness. Examples of stopwords include common articles, prepositions, and conjunctions, like “the”, “to”, and “and”. The removal of stopwords during the preprocessing phase helps improve the efficiency of text analysis and focuses on more meaningful words. As mentioned in various studies in the field of NLP, such as the works of Manning et al. (2008) and Bird et al. (2009), which emphasize the significance of stopwords removal in enhancing the accuracy and effectiveness of natural language processing tasks.

After data preprocessing the below code is used to concatenate the remaining tokens into a single string.

```
# Join the remaining tokens into a string
    return ' '.join(token)
```

‘token’ is a list of individual tokens or words. The *join* method is a string method in Python that concatenates each element of the provided input (in this case, the list of tokens) into a single string. The space between the single quotes *(' ')* is used as a separator, indicating that a space should be inserted between each pair of tokens when they are joined together. The separator could be any other character but comma *(‘,’)*, semicolon *(‘;’)*, or hyphen (*‘-’)* are the most frequently used ones. This step is common in natural language processing and text analysis workflows to convert tokenized text back into a readable and coherent format.

The function is then applied to the 'anonymized_data' column of the 'chats' dataset, and the results are stored in a new column named 'processed_text'. This preprocessing is crucial for enhancing the quality of text data by removing noise and irrelevant information, making it more suitable for analysis and modeling tasks.

```
chats['processed_text'] = chats['anonymized_data'].apply(basic_preprocess_text)
```

Chapter 4. Applying Topic Modeling in Chat Transcripts

4.1 Prepare data for Topic Modeling with Gensim

After data preprocessing, topic modeling techniques are applied in the data that is stored in ‘processed_text’ column.

The first step, before proceeding with topic modeling, is tokenization (*read more in 3.4.3*)

```
texts = chats['processed_text'].str.split().tolist()
```

Gensim's Dictionary class is used to create a mapping of normalized words to unique integer IDs. It scans through the tokenized texts and assigns a unique identifier to each distinct word. (corpora.dictionary – Construct word<->id mappings, Gensim documentation, online)

```
dictionary = corpora.Dictionary(texts)
```

The tokenized texts are converted into bag-of-words representations using the Gensim library. Each document in the 'texts' list is transformed into a sparse vector representation, where each element of the vector corresponds to a unique word's frequency in that document.

```
corpus = [dictionary.doc2bow(text) for text in texts]
```

4.1.1 Gensim Dictionary

Gensim is an essential Python package for processing and modeling textual data for machine learning applications. Gensim helps converting textual data to numerical representations by moving from strings to vectors. There are several ways to represent strings as vectors like bag-of-words, TF-IDF (term frequency-inverse document frequency), LSI (latent semantic indexing), and word2vec.

Gensim’s capability in handling extensive information retrieval tasks is noteworthy, implementing algorithms like TF-IDF, LSI and LDA. It is worth mentioning the platform's adeptness, scalability, and seamless incorporation with scientific libraries like NumPy and SciPy. Its design avoids the necessity of loading complete datasets into the system's memory, which is a significant advantage when working with large volumes of data. (Bhargav Srinivasa-Desikan, 2018)

4.1.2 Bag of Words

Bag-of-words (BoW) model is a simple yet fundamental method for representing text in numerical form. It represents a document as an unordered set of words and their respective frequencies. Each unique word in the document is treated as a feature, and the frequency of occurrence of each word is used to create a numerical vector, forming the basis of the BoW representation. (Bhargav Srinivasa-Desikan, 2018).

An example to better understand the concept of bag-of-words is:

Sentence1: 'I love pasta.'

Sentence2: 'I went to Italy and ate pasta every day.'

After basic preprocessing steps, like removing stopwords, the sentences remain as:

Sentence1: 'love pasta'

Sentence2: 'went Italy ate pasta every day'

Then a vocabulary must be created including the unique words found in the sentences:

vocabulary = ['love', 'pasta', 'ate', 'day', 'went', 'Italy', 'every']

So, the representation of each sentence will be a vector with 7 dimensions.

As mentioned above, the bag-of-words model uses the word's frequencies to create a vector for each sentence. The vectors will be:

Sentence1: [1,1,0,0,0,0,0]

Sentence2: [0,1,1,1,1,1,1]

4.1.3 Bag of Words with Python

As mentioned in 4.1, bag-of-words can be created using Gensim that provides the function doc2bow(). doc2bow(), can be applied to each tokenized sentence as follows:

```
bow = [dictionary.doc2bow() for token in tokens_list]
```

- dictionary is Gensim’s dictionary
- doc2bow is Gensim’s function to create the bag-of-words
- tokens_list is the list with the processed data
- token is every element in the tokens_list

4.2 Train LDA model

4.2.1 Coherence Scores

```
coherence_scores = []
topic_range = range(3, 15)

random_state = 41

# Create a list to store LDA models for different numbers of topics
lda_models = []

# Iterate over different numbers of topics
for num_topics in topic_range:
    # Train an LDA model using the specified number of topics
    lda_model = LdaModel(corpus, num_topics=num_topics, id2word=dictionary,
passes=15, random_state=random_state)

    # Append the trained model to the list
    lda_models.append(lda_model)

    # Calculate coherence score for the trained model
    coherence_model = CoherenceModel(model=lda_model, texts=texts,
dictionary=dictionary, coherence='c_v')
    coherence_score = coherence_model.get_coherence()

    # Append the coherence score to the list
    coherence_scores.append(coherence_score)
```

Coherence scores for different numbers of topics are calculated using Latent Dirichlet Allocation (LDA) models. The code iterates over a specified range of topic numbers (from 3 to 14) and trains an LDA model for each number of topics using the Gensim library.

Below is the code used with the detailed explanations:

Initialization of Variables: *coherence_scores* is an empty list to store the coherence scores, and *topic_range* defines the range of numbers of topics to explore.

```
coherence_scores = []
topic_range = range(3, 15)
```

Training of LDA Models: A loop iterates over the specified *topic_range*. For each *num_topics* (number of topics) in the range, an LDA model is trained using the LdaModel function from Gensim. The model is trained on the previously generated bag-of-words corpus (corpus), the dictionary (dictionary), and other parameters like the number of passes (15 passes) and random state (is set to 41).

The *passes* parameter represents the number of passes through the entire corpus during the training of the LDA model (models.ldamodel – Latent Dirichlet Allocation, Gensim documentation, online). Increasing the number of passes can improve the model's accuracy, but it also requires more computation. The optimal number of passes depends on the specific characteristics of the dataset.

The *random_state* parameter is used to seed the random number generator during the training of the LDA model. Setting a specific random seed ensures a stable output. (Scikit learn documentation, online). If you use the same random seed (value for *random_state*), you should get the same results when training the model multiple times, assuming other conditions (like the dataset and model parameters) remain constant. This is particularly important for research or analysis where reproducibility is crucial (models.ldamodel – Latent Dirichlet Allocation, Gensim documentation, online).

In the provided code, *random_state* is set to 41. This means that, when the code runs with the same dataset and parameters, the results remain the same in terms of the LDA model having a stable *random_state*.

```
random_state = 41

# Create a list to store LDA models for different numbers of topics
lda_models = []

# Iterate over different numbers of topics
for num_topics in topic_range:
    # Train an LDA model using the specified number of topics
    lda_model = LdaModel(corpus, num_topics=num_topics, id2word=dictionary,
passes=15, random_state=random_state)
```

Store Results: The trained LDA models are appended to the list lda_models.

```
# Append the trained model to the list
    lda_models.append(lda_model)
```

Calculation of Coherence Scores: For each trained LDA model, a coherence score is calculated using the CoherenceModel from Gensim. The coherence is computed using the *'c_v'* coherence measure, which evaluates the semantic coherence of topics based on word co-occurrence.

The C_V measure utilizes a sliding window method and applies a one-set segmentation approach to the most significant words identified within a topic. (Evaluate Topic Models: Latent Dirichlet Allocation (LDA), 2019, online) Its computation involves an indirect confirmation measure, utilizing normalized pointwise mutual information (NPMI), which quantifies the degree of association between words while accounting for their individual frequencies and the overall frequency of their co-occurrence in the corpus (Michael Röder et al.,2015) and cosine similarity, which quantifies the similarity between two vectors by measuring the cosine of the angle between them in a multidimensional space (Pal, S. et al., 2022). This methodology enables coherence assessment by evaluating the semantic coherence and interconnection among words within a topic.

```
# Calculate coherence score for the trained model
    coherence_model = CoherenceModel(model=lda_model, texts=texts,
dictionary=dictionary, coherence='c_v')
    coherence_score = coherence_model.get_coherence()
```

Store Results: The corresponding coherence scores are appended to the list *coherence_scores*.

```
# Append the coherence score to the list
    coherence_scores.append(coherence_score)
```

4.2.2 Plot Coherence Scores per Different Number of Topics

```
# Plot coherence scores against different numbers of topics
plt.plot(topic_range, coherence_scores)

# Set minor ticks on the x-axis
plt.minorticks_on()

# Add labels and title to the plot
plt.xlabel("Number of Topics")
plt.ylabel("Coherence Score")
plt.title("Coherence Score vs. Number of Topics")

# Add major grid lines
plt.grid(True, which='major', linestyle='-', linewidth='0.5')

# Add minor grid lines
```

```
plt.grid(True, which='minor', linestyle=':', linewidth='0.5')


# Display the plot
plt.show()
```

Plot Coherence Scores

The line below creates a line plot. *topic_range* represents the different numbers of topics, and *coherence_scores* represents the corresponding coherence scores. The plot shows how coherence scores change with varying numbers of topics.

```
# Plot coherence scores against different numbers of topics
plt.plot(topic_range, coherence_scores)
```

Set Minor Ticks on the X-axis

The line enables minor ticks on the x-axis for more detailed markings.

```
# Set minor ticks on the x-axis
plt.minorticks_on()
```

Labels and Title

These lines set labels for the x-axis, y-axis, and the title of the plot.

```
# Add labels and title to the plot
plt.xlabel("Number of Topics")
plt.ylabel("Coherence Score")
plt.title("Coherence Score vs. Number of Topics")
```

Add Grid Lines

These lines add both major and minor grid lines to the plot for better readability.

```
# Add major grid lines
plt.grid(True, which='major', linestyle='-', linewidth='0.5')

# Add minor grid lines
plt.grid(True, which='minor', linestyle=':', linewidth='0.5')
```

Display the Plot

```
# Display the plot
plt.show()
```

Results

The visualization represents the relationship between the number of topics and coherence scores using a line plot. The goal is to determine an optimal number of topics for a given dataset based on coherence scores.

Based on the visual output, the highest coherence scores are generated from the models with either 4 or 5 topics.

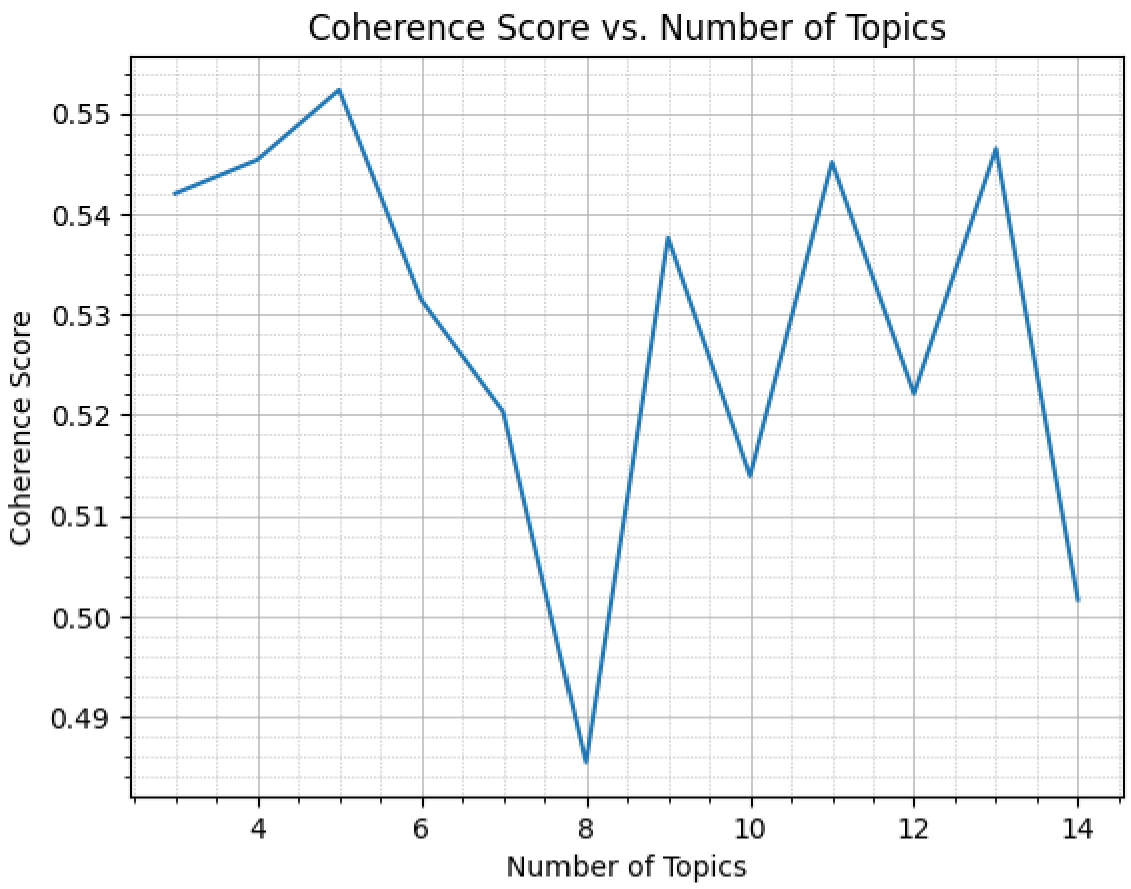


Chart 1: Coherence Scores per Number of Topics

4.3 Perform Topic Modeling - Apply LDA

```
# Choose the optimal number of topics
# 'lda_models' is a list containing LDA models for different 'num_topics'
for lda_model in lda_models:
    topics = lda_model.show_topics(num_topics=-1, num_words=10)  # num_words
determines the number of words to display for each topic
    # setting 'num_topics=-1' instructs the method to display topics for all the topics present
in the model

    for topic_id, topic in topics:
        print(f"Topic {topic_id}: {topic}")
```

Iterate Over LDA Models

This loop iterates over the list of LDA models (*lda_models*), which were trained with different numbers of topics.

```
# Choose the optimal number of topics
# 'lda_models' is a list containing LDA models for different
'num_topics'
for lda_model in lda_models:
```

Retrieve Topics for Each Model

For each LDA model, the line below retrieves the top words associated with each topic. The *num_words*=10 parameter specifies that it should display the top 10 words for each topic. The *num_topics*=-1 parameter indicates that it should show topics for all the topics present in the model.

```
topics = lda_model.show_topics(num_topics=-1, num_words=10)  # num_words
determines the number of words to display for each topic
    # setting 'num_topics=-1' instructs the method to display topics for
all the topics present in the model
```

Print Topic Information

This nested loop iterates through each topic's information obtained from the *show_topics* method. It prints the topic ID (*topic_id*) and the associated top words (*topic*).

```
for topic_id, topic in topics:
        print(f"Topic {topic_id}: {topic}")
```

Results

An example of the output when the *topic_id* equals *4*:

```
Topic 0: 0.114*"number" + 0.031*"brand" + 0.030*"warranty" + 0.025*"refurbished" +
0.023*"tv" + 0.012*"years" + 0.012*"year" + 0.009*"new" + 0.009*"come" + 0.006*"price"
```

Topic 1: 0.132*"number" + 0.063*"brand" + 0.023*"stock" + 0.010*"available" + 0.010*"price" + 0.009*"one" + 0.009*"help" + 0.008*"camera" + 0.008*"looking" + 0.008*"problem"
Topic 2: 0.065*"number" + 0.014*"order" + 0.014*"call" + 0.011*"payment" + 0.010*"card" + 0.010*"delivery" + 0.010*"get" + 0.009*"address" + 0.008*"contact" + 0.008*"purchase"
Topic 3: 0.082*"brand" + 0.064*"order" + 0.058*"number" + 0.036*"contact" + 0.030*"email" + 0.023*"store" + 0.019*"confirmation" + 0.019*"need" + 0.014*"able" + 0.012*"directly"

Similar results are printed for each number of topics in the range that was specified previously (3,15).

Mapping of topics

The dictionary below associates topics with specific categories/themes.

The outer keys '4' and '5' represent different numbers of topics.

The inner keys "0", "1", "2", "3", and "4" represent individual topics within each category.

The values associated with each inner key are strings describing the specific theme/category associated with that topic including business logic and context. For example, topics within the category '4' (which has 4 topics) are related to "Refurbished Products and Warranties," "Stock Availability and Customer Support," "Order Information & Payment Issues," and "Confirmation Email & Contact Information.

```
config = {
        '4': {
                "0": "Refurbished Products and Warranties",
                "1": "Stock Availability and Customer Support",
                "2": "Order Information & Payment Issues",
                "3": "Comfirmation Email & Contact Information"
              },
        '5': {
                "0": "Refurbished Products and Warranties",
                "1": "Stock Availability and Customer Support",
                "2": "Order Information & Payment Issues",
                "3": "Comfirmation Email & Contact Information",
                "4": "Pricing and Model Comparison"
              }
      }
```

This configuration allows for easy reference and categorization of topics based on their respective themes. It is a useful structure for organizing and interpreting the topics generated by a topic modeling algorithm in a more human-readable and meaningful way.

Assign Topic Function

assign_topic function takes a text input, preprocesses and tokenizes it, and then assigns a dominant topic to that text based on a pre-trained Latent Dirichlet Allocation (LDA) model.

The function is useful for assigning topics to new or unseen text data based on the patterns learned by the LDA model during training. A pre-trained LDA model (lda_model) and a dictionary (dictionary) are required to use this function.

```
# Preprocess and tokenize the text before applying the LDA model
def assign_topic(text):
    # Tokenize the text (adjust this tokenization process based on your data)
    tokens = text.split()  # Split the text into words by spaces; you may need more advanced tokenization for better results
```

```
    # Convert tokens to bag-of-words format using the dictionary
    bow = dictionary.doc2bow(tokens)
    # Use your trained LDA model to get the topic distribution for the text
    topic_distribution = lda_model[bow]

    # Find the topic with the highest probability in the distribution
    dominant_topic = max(topic_distribution, key=lambda item: item[1])

    # Return the topic ID (you can map this to your topic labels if needed)
    return dominant_topic[0]
```

In detail, the input text is tokenized by splitting it into words based on spaces.

```
tokens = text.split()
```

The tokenized text is then converted to a bag-of-words (BoW) representation using a previously created Gensim dictionary (dictionary).

```
bow = dictionary.doc2bow(tokens)
```

The bag-of-words representation of the text is used to predict the topic distribution using a pre-trained LDA model (lda_model). The result is a list of tuples, where each tuple consists of a topic ID and its corresponding probability.

```
topic_distribution = lda_model[bow]
```

The topic with the highest probability in distribution is identified as the dominant topic. The *dominant_topic* variable, holds a tuple with the format (*topic_id*, *probability*).

```
dominant_topic = max(topic_distribution, key=lambda item: item[1])
```

The function returns the topic ID of the dominant topic.

```
return dominant_topic[0]
```

Topics Function

topics is designed to train an LDA (Latent Dirichlet Allocation) model with a specified number of topics. The number of topics is passed to the function as the parameter "*number_topics*." The LDA model is trained using the Gensim library, where the corpus (bag-of-words representation) and the dictionary (mapping words to unique integer ids) are utilized. The function returns a dictionary of topic labels based on the pre-defined configuration, "*config*".

```
def topics(number_topics):
  # Train the LDA model with the optimal number of topics
  lda_model = LdaModel(corpus, num_topics=number_topics, id2word=dictionary,
passes=15, random_state=random_state)



  # Assign topic labels based on a dictionary created above "config"
  topic_labels = config[str(number_topics)]

  return topic_labels
```

In detail, an LDA model is trained using the Gensim library. The *corpus* is the bag-of-words representation, *number_topics* is the specified number of topics, *id2word* is the dictionary mapping words to unique integer ids, *passes* is the number of training passes, and *random_state* is used to seed the random number generator for reproducibility.

```
lda_model = LdaModel(corpus, num_topics=number_topics, id2word=dictionary,
passes=15, random_state=random_state)
```

Then, the topic labels are retrieved from the pre-defined configuration dictionary named "*config*." The key used to access the labels is the string representation of the input *number_topics*.

```
# Assign topic labels based on a dictionary created above "config"
  topic_labels = config[str(number_topics)]
```

The function returns the obtained topic models.

```
    return topic_labels
```

The final step is to perform topic modeling for different numbers of topics, assign topics to chat transcripts, and save the results in separate Excel files.

```
# According to the Coherence Scores, I will train 2 models (one having 4 topics and one
having 5 topics)
for num_topics in (4,5):
  # Train the LDA model with the optimal number of topics
  lda_model = LdaModel(corpus, num_topics=num_topics, id2word=dictionary, passes=15,
random_state=random_state)

  # Get the topics and their top words
  topics_list = lda_model.show_topics(num_topics=num_topics, num_words=10)

  # Apply the assign_topic function to the 'Transcript' column and create a new 'Topic'
column
  chats['topic'] = chats['processed_text'].apply(lambda text: assign_topic(text))

  # Now, the 'topic' column in the dataFrame will contain the assigned topic for each chat
```

```
transcript
  print(num_topics)
  topic_labels = topics(num_topics)
  new_labels = {int(key): value for key, value in topic_labels.items()}
  print(topic_labels)
  print("new labels", new_labels)
  print(chats['topic'])
  chats['topic_label'] = chats['topic'].map(new_labels)

  # Save the results of topic modeling to an Excel file - 1 file per number of topics
  file_name = f"{num_topics} topics.xlsx"
  chats.to_excel(file_name,index=False)
```

A loop runs for *num_topics* in the tuple (4, 5), indicating that it trains two LDA models, one with 4 topics and another with 5 topics.

```
for num_topics in (4,5):
```

Inside the loop, an LDA model (*lda_model*) is trained using the specified number of topics, 15 passes, and other already defined parameters.

```
lda_model = LdaModel(corpus, num_topics=num_topics, id2word=dictionary, passes=15,
random_state=random_state)
```

The topics and their top words are retrieved using *lda_model.show_topics*.

```
topics_list = lda_model.show_topics(num_topics=num_topics, num_words=10)
```

The *assign_topic* function is applied to the *'processed_text'* column of the *'chats'* DataFrame, creating a new column *'topic'* that contains the assigned topic for each chat transcript.

```
chats['topic'] = chats['processed_text'].apply(lambda text: assign_topic(text))
```

The *topics* function is then used to obtain the topic labels based on the specified number of topics.

```
topic_labels = topics(num_topics)
```

The *'topic'* column is mapped to the obtained topic labels using the *'new_labels'* mapping.

```
new_labels = {int(key): value for key, value in topic_labels.items()}
chats['topic_label'] = chats['topic'].map(new_labels)
```

Finally, the results are saved to an Excel file with a filename indicating the number of topics, such as "4 topics.xlsx" and "5 topics.xlsx". The Excel file includes columns for *'processed_text'*, *'topic'*, and *'topic_label'*.

Chapter 5. Data Analysis and Interpretation

5.1 Generated topics

The results provided by the model are:

4-topics

Topic 0: 0.114*"number" + 0.031*"brand" + 0.030*"**warranty**" + 0.025*"**refurbished**" + 0.023*"**tv**" + 0.012*"**years**" + 0.012*"year" + 0.009*"**new**" + 0.009*"come" + 0.006*"**price**"

Topic 1: 0.132*"number" + 0.063*"brand" + 0.023*"**stock**" + 0.010*"**available**" + 0.010*"**price**" + 0.009*"one" + 0.009*"**help**" + 0.008*"**camera**" + 0.008*"looking" + 0.008*"**problem**"

Topic 2: 0.065*"number" + 0.014*"**order**" + 0.014*"call" + 0.011*"**payment**" + 0.010*"**card**" + 0.010*"**delivery**" + 0.010*"get" + 0.009*"address" + 0.008*"**contact**" + 0.008*"**purchase**"

Topic 3: 0.082*"brand" + 0.064*"**order**" + 0.058*"number" + 0.036*"contact" + 0.030*"**email**" + 0.023*"store" + 0.019*"**confirmation**" + 0.019*"need" + 0.014*"able" + 0.012*"directly"

<u>5-topics</u>

Topic 0: 0.101*"number" + 0.043*"**warranty**" + 0.038*"**refurbished**" + 0.036*"brand" + 0.018*"**years**" + 0.018*"year" + 0.015*"**tv**" + 0.013*"**new**" + 0.012*"come" + 0.009*"**grade**"

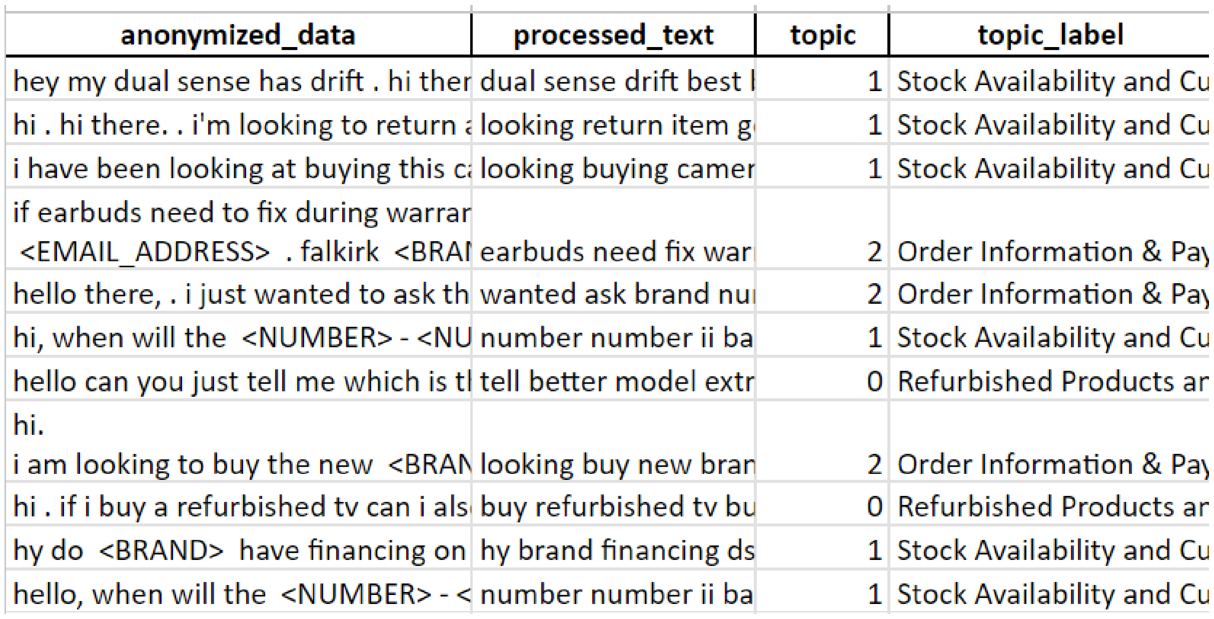
Topic 1: 0.151*"number" + 0.071*"brand" + 0.027*"**stock**" + 0.011*"**available**" + 0.010*"call" + 0.010*"**help**" + 0.009*"one" + 0.009*"**problem**" + 0.008*"looking" + 0.008*"harrow"

Topic 2: 0.063*"number" + 0.015*"**order**" + 0.012*"**payment**" + 0.012*"call" + 0.010*"contact" + 0.010*"**card**" + 0.010*"address" + 0.009*"get" + 0.009*"**purchase**" + 0.009*"**delivery**"

Topic 3: 0.082*"brand" + 0.068*"**order**" + 0.052*"number" + 0.037*"contact" + 0.033*"**email**" + 0.026*"store" + 0.022*"**confirmation**" + 0.020*"need" + 0.014*"able" + 0.012*"directly"

Topic 4: 0.085*"number" + 0.025*"**tv**" + 0.016*"**price**" + 0.015*"brand" + 0.009*"**sound**" + 0.008*"**model**" + 0.008*"**better**" + 0.008*"one" + 0.007*"**discount**" + 0.007*"need"

5.1.1 4-Topics



anonymized_data	processed_text	topic	topic_label
hey my dual sense has drift . hi ther	dual sense drift best l	1	Stock Availability and Cu
hi . hi there. . i'm looking to return a	looking return item g	1	Stock Availability and Cu
i have been looking at buying this ca	looking buying camer	1	Stock Availability and Cu
if earbuds need to fix during warrar <EMAIL_ADDRESS> . falkirk <BRAI	earbuds need fix war	2	Order Information & Pay
hello there, . i just wanted to ask th	wanted ask brand nu	2	Order Information & Pay
hi, when will the <NUMBER> - <NU	number number ii ba	1	Stock Availability and Cu
hello can you just tell me which is tl	tell better model extr	0	Refurbished Products ar
hi. i am looking to buy the new <BRAN	looking buy new bran	2	Order Information & Pay
hi . if i buy a refurbished tv can i als	buy refurbished tv bu	0	Refurbished Products ar
hy do <BRAND> have financing on	hy brand financing ds	1	Stock Availability and Cu
hello, when will the <NUMBER> - <	number number ii ba	1	Stock Availability and Cu

Figure 4: Snapshot of the results of the 4 generated topics

Example transcripts after categorization:

Topic 0: Refurbished Products and Warranties

- hello there, is the a **warranty** on **refurbished** oled tv's?
- hi yes they come with <NUMBER> year and this can be extended up to <NUMBER> years
- okay, i was looking at a <DATE> inch. when i try to purchase the price it change
- ok i can process this for you if you want to call me not sure why that is happening. <NUMBER>
- let me try once again
- ok thank you
- it worked now thank you
- excellent thank you

Topic 1: Stock Availability and Customer Support

- do you currently have **stock** of <BRAND> <NUMBER> noise cancelling wireless headphones <NUMBER> . <NUMBER>
- hi there , my name is <name> from harrow <BRAND> <BRAND> . would you like brand new or refurbished ?

- new
- if you call me on <name> <NUMBER> <NUMBER> i will be able to process the order for you
- i am at the checkout on your web store, just need to check **stock**
- yes there is **stock** available
- great thanks
- no problem.
- have a nice day

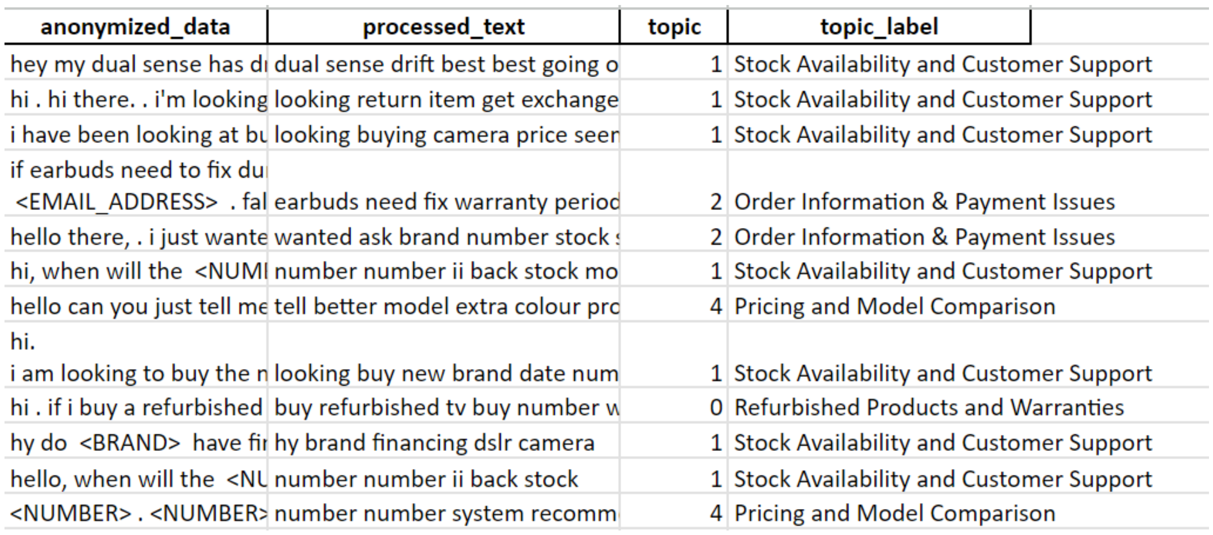
Topic 2: Order Information & Payment Issues

- hello i’m about to buy the <BRAND> <NUMBER> and i’d like to know if it will be delivered by thursday
- hi there
- hi
- where are you located?
- winchelsea
- if you give us a call on <NUMBER> <NUMBER> <NUMBER> we can proccess this for you and ensure delivery on time too
- actually my wife just made the purchase on line the order number is web-<NUMBER>
- okay, on your email confirmation it will state a <BRAND> <BRAND> i would contact them to ensure you can get it by thursday. just so your on the safer side :) . : <NUMBER> : .
- thanks very much. have a good day
- you too

Topic 3: Confirmation Email & Contact Information

- hi <name> , just to check, will i get a confirmation email for the order?
- confirmation email has been sent. please check your junk folder.
- found it, thanks <name>
- have a great day

5.1.2 5-Topics



anonymized_data	processed_text	topic	topic_label
hey my dual sense has di	dual sense drift best best going o	1	Stock Availability and Customer Support
hi . hi there. . i'm looking	looking return item get exchange	1	Stock Availability and Customer Support
i have been looking at bu	looking buying camera price seer	1	Stock Availability and Customer Support
if earbuds need to fix du <EMAIL_ADDRESS> . fal	earbuds need fix warranty period	2	Order Information & Payment Issues
hello there, . i just wante	wanted ask brand number stock s	2	Order Information & Payment Issues
hi, when will the <NUMI	number number ii back stock mo	1	Stock Availability and Customer Support
hello can you just tell me	tell better model extra colour prc	4	Pricing and Model Comparison
hi. i am looking to buy the n	looking buy new brand date num	1	Stock Availability and Customer Support
hi . if i buy a refurbished	buy refurbished tv buy number w	0	Refurbished Products and Warranties
hy do <BRAND> have fi	hy brand financing dslr camera	1	Stock Availability and Customer Support
hello, when will the <NU	number number ii back stock	1	Stock Availability and Customer Support
<NUMBER> . <NUMBER>	number number system recomm	4	Pricing and Model Comparison

Figure 5: Snapshot of the results of the 5 generated topics

Example transcripts after categorization:

Topic 0: Refurbished Products & Warranties

- hi there when ordering from here do i also get the <BRAND> two year **guaranteeing**
- hi there . what are you looking to order?
- <BRAND> <NUMBER> in black
- this comes with a <NUMBER> year **warranty**, with an option to extend to <NUMBER> , <NUMBER> or <NUMBER> years at an extra cost
- thank you
- you're welcome

Topic 1: Stock Availability & Customer Support

- hi, i am having issues processing my order
- good morning, how may l help?
- its saying its unable to calculate delivery options so not letting me process to payment
- what are you trying to purchase?
- <NUMBER> digital compact camera with <NUMBER> optical zoom
- l am checking stock one moment

- thank you
- unfortunately this is because the camera is not in stock.
- but it didnt say it was out of stock and its literally in my basket waiting for payment?
- it is very possible that it has just gone out of stock, and you missed it by a small margin
- is there any way of checking availability in store? or is it completely out of stock? thanks
- it is completely out of stock at <BRAND> and in all stores
- ok thanks
- you're welcome

Topic 2: Order Information & Payment Issues

- why will this not accept my american express <name>
- hi there
- hi. may l ask what you are reffering to?
- i am trying to buy a tv on your website it will not let me pay by american express
- ok bear with me while l check this for you. can you call us on <NUMBER>
- yes
- thank you

Topic 3: Confirmation Email & Contact Information

- delivery address
- hi there
- hi there i have pre ordered a <BRAND>   <NUMBER> iv but for delivery address i want to change it or change the date of delivery. is it possible as i will be going away for <NUMBER> weeks and i won’t be home to collect the order
- you would need to contact the supplying <BRAND>   <BRAND>
- how do i contact them pls? is there any number or email to contact them?
- it will be at the bottom of your confirmation email

- alright thank you, and do you know probably how long does it take for pre order to be delivered?
- did you order from this website?
- as i notice it’s been a few days and nothing is updated . yes this website as it’s out of stock on <BRAND> website . can you see which store it is?
- i can’t see which store . it will be at the bottom of your confirmation email
- <NUMBER> . <name> on themes . <NUMBER> high street <name> on themes <name>
- you can call them and they will give you an update
- ok thank you
- you're welcome

Topic 4: Pricing & Model Comparison

- hello can you just tell me which is the better model <name>
- hello .the <name> has an extra colour processing level so is the better of the two
- ok are they both <NUMBER> <NUMBER>
- yes they are
- thank you
- my pleasure

5.1.3 Selection Criteria of the Optimal Number of Topics

When selecting the optimal configuration for the topic modeling approach, it was crucial to balance detail and coherence in the topics identified within the dataset of chat transcripts.

Having said that, the model configured to identify **5 topics is chosen as the best choice**. This decision is supported by the improvement in coherence scores, which are indicative of the model’s ability to produce more meaningful and distinct topic categorization. The one additional topic, compared to the 4-topics model which has also high coherence score, allows for a finer segmentation of the dataset, finding patterns and themes that are less visible with only 4 topics.

5.2 Findings

Topic 0: Refurbished Products and Warranties

Insights

Warranty Coverage Concerns: Customers may be highly interested in understanding the warranty coverage for refurbished products, especially around the duration of the warranty.

Interest in Extended Warranties: A demand for extended warranty options for refurbished products may exist. Customers may investigate the possibility of purchasing additional coverage, indicating a desire for long-term security and protection for their investments.

Seeking Assurance on Product Quality: While customers are interested in the value offered by refurbished products, they may seek reassurance about the product's condition and reliability, which they associate closely with warranty terms.

Comparison with New Products: A comparison of the warranty and overall value proposition of refurbished products against new items is possible. This comparison influences customers' purchasing decision, highlighting the importance of clear communication about the benefits and warranties of refurbished items.

Strategic Recommendations

Clarify Warranty Terms: Enhance product listings and FAQ sections to clearly state the warranty terms for refurbished products. This should include duration, coverage details, or how these terms compare to new product warranties.

Promote Extended Warranty Options: Actively promote the availability of extended warranties for refurbished products. Make it easy for customers to purchase these add-ons by integrating the option into the online purchasing process and highlighting the benefits of extended coverage.

Highlight Quality Assurance Processes: To reassure customers about the quality and reliability of refurbished products, provide detailed information on the refurbishment process. This can include checks and repairs made, certification processes, and any replacement parts used.

New - Refurbished Models Comparison: Offer clear comparisons between refurbished and new products, focusing on warranty coverage, price differences, features and more. This can help customers make informed decisions by understanding the value of each category.

Customer Reviews and Testimonials: Encourage and showcase customer reviews and testimonials specifically for refurbished products. Positive experiences regarding product quality and warranty claims can significantly boost confidence among prospective buyers.

Topic 1: Stock Availability and Customer Support

Insights

Product Availability Concerns: Customers may inquire about the availability of specific products, including new releases and items currently out of stock. This indicates a strong customer interest in timely and accurate stock information.

Return and Exchange Inquiries: Reference to a problem may indicate that there is a number of queries related to returning or exchanging items, suggesting that a clear, straightforward return and exchange policy is crucial for customer satisfaction.

Interest in Upcoming Products: Customers may be keen on obtaining information about upcoming product releases and their availability, indicating a demand for preemptive communication strategies about new stock arrivals.

Request for Specific Product Features: Customers may need help finding specific features or specifications of the products, highlighting the importance of detailed product descriptions and easily accessible product information.

Strategic Recommendations

Improve Real-Time Stock Updates: Develop and implement a system for real-time stock updates on the website, enabling customers to see current availability and expected restock dates for out-of-stock items.

Streamline Return Processes: Ensure the return and exchange policy is clearly communicated. Consider simplifying the process to enhance customer trust and loyalty, such as by offering free returns or extended return periods.

Leverage Pre-Order Options for New Releases: For anticipated new product releases, offer pre-order options to capture customer interest early. Communicate availability dates and any special offers to generate excitement and drive sales.

Enhance Product Descriptions and FAQs: Expand product descriptions to include comprehensive specifications, features, and user guides. Update the FAQs section to address common inquiries, potentially reducing the volume of similar customer support requests.

Topic 2: Order Information & Payment Issues

Insights

Order and Delivery Clarifications: Customers may seek clarification on order statuses, delivery timelines, suggesting some confusion or lack of information during the purchasing process or delays in delivery.

Payment and Order Processing Issues: The intense use of words like ‘payment’, ‘card’, purchase’ highlight possible issues with payment processing, pointing to potential technical or procedural bottlenecks that could impact customer satisfaction.

Strategic Recommendations

Streamline Order and Delivery Information: Ensure that customers receive timely updates about their order status and expected delivery dates. This could be facilitated through automated emails, SMS updates, or a customer portal where they can track their order's progress.

Address Payment Processing Challenges: Review and optimize the payment processing system to prevent issues that could lead to order cancellation or customer frustration. Ensure there are multiple payment options available to cater to various customer preferences.

Customer Service Training: Train customer service teams to handle inquiries related to orders and payments. Empower them with the tools and information needed to provide accurate, helpful responses.

Request Feedback: Implement a system for collecting and analyzing customer feedback on order processing, payment, and post-purchase support. Use this feedback to identify areas for improvement and make necessary adjustments to processes and policies.

Topic 3: Confirmation Email & Contact Information

Insights

Concerns Over Order Confirmation: Customers may express concerns about receiving confirmation emails after placing orders. This indicates a need for immediate and clear communication to reassure customers their orders have been successfully processed.

Requests for Order Modifications and Cancellations: May there are several inquiries about changing or canceling orders before fulfillment. This suggests that flexibility in order management and clear instructions on how to modify orders are important to customers.

Importance of Accurate Contact Information: The theme of contact information suggests some customers may be unsure of how to get in touch with customer service or stores for support, indicating the necessity of making contact information readily available and visible.

Strategic Recommendations

Automate Order Confirmation Emails: Ensure that an automated, error-free system is in place to send immediate confirmation emails upon order placement, including order details, next steps, and customer service contact information.

Implement Easy Order Modification Processes: Provide clear guidelines and an easy-to-use online interface for customers to modify or cancel their orders within a certain timeframe after placement.

Clearly Mention Steps in the Checkout Process: Add a progress tracking line in the checkout process. Its benefits are the reduced abandonment, improved user engagement and error prevention.

Proactive Delivery Updates: Implement a proactive communication strategy for delivery updates, utilizing email, SMS, or account notifications to keep customers informed about the status of their orders.

Prominently Display Contact Information: Make sure that contact information for customer support is prominently displayed across all digital platforms, including the website, confirmation emails, and social media profiles.

Topic 4: Pricing and Model Comparison

Insights

Comparison Queries: Customers questions about the differences between models, indicating a need for clear, accessible comparative information that highlights the features, benefits, and pricing of various products.

Price Sensitivity: May be a notable concern with pricing and requests for price matching, suggesting that customers are price-sensitive and actively seek the best deals, comparing the retailer's prices with those of competitors.

Need for Detailed Product Information: Customers may be looking for detailed information to help them make informed decisions especially for TVs, including specifics about product features, technical specifications, and the value added by higher-priced models over lower-priced alternatives.

Strategic Recommendations

Enhance Product Comparison Tools: Develop and integrate advanced product comparison tools on the website that allow customers to easily compare different models side-by-side, focusing on key features, specifications, prices, and customer reviews.

Implement Price Match Warranties: Consider offering a price match warranty to reassure customers they are getting the best deal possible. Clearly communicate the terms and conditions of the price match policy.

Provide Comprehensive Product Details: Ensure product pages are detailed and informative, including high-quality images, technical specifications, feature comparisons, and FAQs. This could also include video tutorials or demos that highlight key features and benefits.

Educate Sales and Support Staff: Train sales and customer support staff to effectively communicate the differences and benefits of various models and pricing options, enabling them to offer personalized recommendations based on customer needs and budgets.

Promote Special Offers and Bundles: Introduce special offers, bundles, or promotions that provide additional value to customers, such as discounts on related products, complimentary services, or extended warranties at reduced rates.

Chapter 6. Conclusion

In this thesis, the primary objective was to categorize chats into distinct themes, with each theme providing valuable insights into customer behaviors, preferences, and needs. By analyzing chat transcripts from an electronics e-commerce platform, the study identified patterns that offer a deeper understanding of customer concerns and areas requiring improvement in online customer service.

The application of Latent Dirichlet Allocation (LDA) has not only validated the effectiveness of topic modeling in extracting meaningful information from unstructured data, like chats, but also highlighted its potential to inform strategic decision-making in e-commerce.

The findings underscore the criticality of transparent communication regarding product warranties, stock availability, and pricing strategies, which directly influence customer satisfaction and loyalty. Moreover, the analysis indicates the necessity for ecommerce platforms to streamline their order processing, payment systems, and customer support to enhance the overall user experience.

Future research directions could expand upon this work by exploring the integration of sentiment analysis to gauge customer emotions and attitudes. This additional layer of analysis could offer a more nuanced understanding of customer satisfaction and pinpoint areas for emotional engagement and personalized customer service strategies.

Furthermore, investigating the temporal dynamics of customer inquiries through dynamic topic modeling could reveal how customer concerns evolve over time, offering insights into seasonal trends or shifting consumer priorities.

In conclusion, this study underscores the potential of advanced data analytics techniques, like topic modeling, to be incorporated into the business analytics workflows to help businesses transform ecommerce customer service. By leveraging these hidden insights in unstructured and hard-to-analyze data, ecommerce platforms can tailor their strategies to better meet customer needs and foster positive experiences. Future research in this area is set to explore the complexities of customer interactions, paving the way for innovative solutions to enhance the digital commerce landscape.

Bibliography

1. David M. Blei, Andrew Y. Ng and Michael I. Jordan (2003), Latent Dirichlet Allocation, Journal of Machine Learning Research 3 (2003) p.p. 993-1022.
2. Blei, D. M. (2012), Probabilistic topic models, Communications of the ACM, 55(4), p.p. 77-84.
3. Rehurek, R., & Sojka, P. (2010), Software Framework for Topic Modelling with Large Corpora, Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks, p.p. 45-50.
4. Pedregosa, F., et al. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, p.p. 2825-2830.
5. Natural Language Toolkit [Online] Available from: https://www.nltk.org/ [Accessed: 03 January 2024].
6. A Beginner's Guide to Latent Dirichlet Allocation(LDA) [Online] Available from: https://towardsdatascience.com/latent-dirichlet-allocation-lda-9d1cd064ffa2
   [Accessed: 03 January 2024]
7. Deerwester, S., et al. (1990). Indexing by latent semantic analysis. Journal of the American Society for Information Science, 41(6), p.p. 391-407. Available from:
   https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=1944fda0d2aad86fdfac5f3c0d5c3b4055fa1336 [Accessed: 04 January 2024]

8. Dumais, S.T.. (2004). Latent Semantic Analysis. Annual Review of Information Science and Technology. 38. 188-230. 10.1002/aris.1440380105.
9. Landauer, T. K., & Dumais, S. T. (1997). A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. Psychological Review, 104(2), 211-240.
10. Lee, D. D., & Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. Nature, 401(6755), 788-791.
11. What is Non-Negative Matrix Factorization (NMF)? (2022). [Online] Available from: https://medium.com/codex/what-is-non-negative-matrix-factorization-nmf-32663fb4d65. [Accessed: 28 January 2024]
12. Jiangzhang Gan, Tong Liu, Li Li, Jilian Zhang (2021), Non-negative Matrix Factorization: A Survey, The Computer Journal, p.p. 1080–1092
13. Zhao, W., et al. (2013). "BTM: Topic Modeling over Short Texts." In Proceedings of the 22nd International Conference on World Wide Web.
14. Yan, X., et al. (2013). "A Biterm Topic Model for Short Texts." In Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics.
15. Blei, D. M., & Lafferty, J. D. (2006). "Dynamic Topic Models." In Proceedings of the 23rd International Conference on Machine Learning.
16. Teh, Y. W., Jordan, M. I., Beal, M. J., & Blei, D. M. (2006). "Hierarchical Dirichlet Processes." Journal of the American Statistical Association, 101(476), p.p. 1566-1581.
17. Jurafsky, D., & Martin, J. H. (2023). Speech and Language Processing (3rd ed.). p.p. 12-13, 30.
18. Manning, C. D., Raghavan, P., & Schütze, H. (2009). Introduction to Information Retrieval. Cambridge University Press. p.p. 56-69.
19. Bird, S., Klein, E., & Loper, E. (2009). Natural Language Processing with Python. O'Reilly Media.
20. corpora.dictionary, Gensim documentation [Online] Available from: https://radimrehurek.com/gensim/corpora/dictionary.html [Accessed: 18 January 2024]
21. Bhargav Srinivasa-Desikan (2018). Natural Language Processing and Computational Liguistics. Packt Publishing. p.p. 51-55

22. models.ldamodel – Latent Dirichlet Allocation, Gensim documentation [Online] Available from: https://radimrehurek.com/gensim/models/ldamodel.html [Accessed: 20 January 2024]
23. sklearn.model_selection.train_test_split, scikit-learn [Online] Available from: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html [Accessed: 20 January 2024]
24. Jelodar, H., Wang, Y., Yuan, C. et al. (2019). Latent Dirichlet allocation (LDA) and topic modeling: models, applications, a survey. Multimed Tools Appl 78, p.p. 15169–15211.
25. Xiaohui Yan, Jiafeng Guo, Yanyan Lan, et al. (2013). A Biterm Topic Model for Short Texts. Conference: Proceedings of the 22nd international conference on World Wide Web. p.p. 1445–1456.
26. Evaluate Topic Models: Latent Dirichlet Allocation (LDA) [Online] Available from: https://towardsdatascience.com/evaluate-topic-model-in-python-latent-dirichlet-allocation-lda-7d57484bb5d0 [Accessed: 27 January 2024]
27. Michael Röder, Andreas Both, and Alexander Hinneburg (2015). Exploring the Space of Topic Coherence Measures. In Proceedings of the Eighth ACM International Conference on Web Search and Data Mining (WSDM '15). Association for Computing Machinery, New York, NY, USA. p.p. 399–408.
28. Pal, S., Chang, M., Iriarte, M.F. (2022). Summary Generation Using Natural Language Processing Techniques and Cosine Similarity. p.p. 508-517. In: Abraham, A., Gandhi, N., Hanne, T., Hong, TP., Nogueira Rios, T., Ding, W. (eds) Intelligent Systems Design and Applications. ISDA 2021. Lecture Notes in Networks and Systems, vol 418. Springer, Cham.