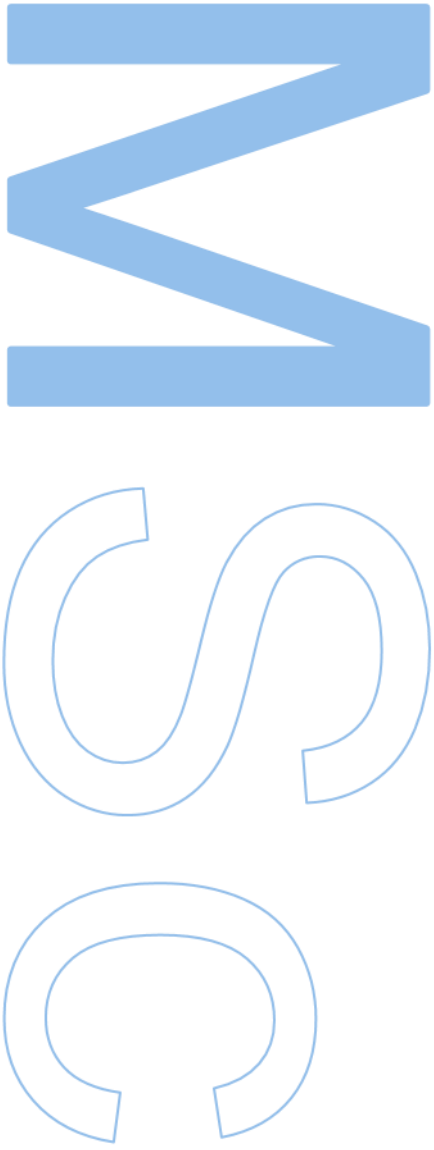
Informed Retail Product Similarity

Tatiana Araújo
Master's in Computer Science
Department of Computer Science
2023

Supervisor
Inês Dutra, Professor Auxiliar, Faculty of Sciences
University of Porto



Deloitte.

M

S

S

Abstract

Making informed purchasing decisions in the age of e-commerce amidst a vast diversity of products is challenging. From a customer's point of view, it is important to compare prices, quantities or features such as environmental-friendly products. On the other hand, retailers would like to find the best replacement for an out-of-stock product which is of interest to a customer. To address these issues, we use a known methodology based on the TF-IDF technique to represent a product, and apply K-means clustering to group similar products. Our main challenge is the poor quality of the available textual data associated with the products. Despite this challenge, results show that our approach successfully groups the products that are actually similar, providing valuable insights into their distribution. It is also important to note that we only hold information about the products themselves and have no data about consumers or purchase history. This dissertation was conducted within the corporate setting of Deloitte.

Resumo

Na era do e-commerce, tomar decisões de compra informadas com uma vasta diversidade de produtos é desafiador. Do ponto de vista do cliente, é importante comparar preços, quantidades ou características tais como saber se os produtos são ecológicos ou não. Por outro lado, para os retalhistas seria útil encontrarem o melhor substituto para um produto fora de stock que seja interessante para o cliente. Para enfrentar estes problemas, usamos uma metodologia conhecida com base na técnica TF-IDF para representar um produto e aplicamos a clusterização K-means para agrupar produtos similares. O principal desafio é a baixa qualidade dos dados textuais disponíveis associados aos produtos. Apesar disto, os resultados mostram que nossa abordagem agrupa com sucesso os produtos que são realmente semelhantes, fornecendo insights valiosos sobre a sua distribuição. É ainda importante referir que possuímos apenas informações sobre os próprios produtos e não temos dados sobre os consumidores ou histórico de compras. Esta dissertação foi realizada no contexto empresarial da Deloitte.

Acknowledgements

I am deeply grateful to Professor Inês Dutra, for the constant dedication, persistence, guidance and sharing of knowledge in the development of the project.

I would like to extend my deepest gratitude to my supervisors at Deloitte, Paulo Mauricio and Vasco Pena, for their unwavering trust in me and my abilities throughout this challenging project. Their support and belief in my capabilities have been invaluable, and I am truly honored to have had the opportunity to work under their guidance.

To my colleagues at Deloitte, particularly Bárbara Correia, for the remarkable level of support they have provided, which has played a significant role in my personal and professional growth within the company.

I want to convey my deepest gratitude to my parents for the exceptional love and care they bestowed upon me, and for their constant encouragement and unwavering support in all my choices. The education they provided and their unwavering belief in me have been instrumental in shaping my path, and I am forever indebted to them for their boundless love and dedication.

Some special words of gratitude go to my dear friends and the rest of my family who have always been a major source of support when things would get a bit discouraging. Thanks guys for always being there for me.

I am grateful to my teachers at DCC for sharing their knowledge during my master's program. Their guidance and expertise have greatly influenced my educational journey, and I appreciate their commitment to my growth and development.

Finally, I could not fail to mention the four beautiful stars I have in the sky, for the love.

Contents

List of Tables

List of Figures

Acronyms

DCC Departamento de Ciência de Computadores

FCUP Faculdade de Ciências da Universidade do Porto

NLP Natural Language Processing

BOW Bag-of-Words

TF-IDF Term Frequency-Inverse Document Frequency

EAN European Article Number

DBSCAN Density-Based Spatial Clustering of Applications with Noise

MinPts Minimum Number of Data Points

ML Machine Learning

AI Artificial Intelligence

GPT-2 Generative Pre-trained Transformer 2

AWS Amazon Web Services

RDS Relational Database

Chapter 1

Introduction

In this initial chapter, we begin by introducing the dissertation, covering its context, motivation, problem description, and goals. In Section 1.1, we delve into the competitive landscape of the retail industry, underscoring the need for retailers to adapt and stay relevant in the face of disruptive forces such as supermarkets and e-commerce. To thrive in this evolving environment, retailers are increasingly turning to advanced data analytics to glean insights and enhance customer relationships.

Moving on to Section 1.2, we explore the concept of product comparison platforms and the challenges they encounter. This section offers a succinct overview of the critical issues and obstacles in this domain. Furthermore, we outline the study's goals, which involve creating a solution to identify similar products based on their textual attributes.

Finally, in Section 1.3, we provide an overview of the document's structure. It encompasses a theoretical background, an examination of related work, details on the implementation framework, a discussion of results, and a concluding section.

1.1 Context and Motivation

The retail industry plays a vital role in the global economy, encompassing a wide range of businesses that directly sell goods and services to consumers. This sector is characterized by intense competition, requiring retailers to constantly evolve and adjust their strategies to stay relevant. Over the past few years, the industry has undergone significant transformations due to technological advancements and changing consumer preferences. Notably, the emergence of supermarkets and the rapid growth of e-commerce have disrupted the traditional brick-and-mortar retail models, compelling retailers to adapt in order to maintain their competitiveness [33].

Retailing has a long history, dating back to prehistoric times, but it was the introduction of money that paved the way for modern retail practices. While some European cities had retail chains in the 16th and 17th centuries, the true development of retailing is recognized to have

occurred in the late 19th and early 20th centuries. Initially, retailers offered a wide range of merchandise, but over time, specialized stores gained popularity. In the past fifty years, the retail industry has undergone significant transformations, shifting from traditional proximity-based retailing to target retailing, where consumers are willing to travel further for better options and prices. Today, the retail industry has become a crucial driver of economic growth in major economies worldwide. However, the landscape of retailing is constantly evolving, marked by increasing competition, margin pressures, and a rise in acquisition activity. Modern customers are more discerning and demanding, requiring retailers to provide a unique in-store experience to attract and retain their loyalty. As retail assortments and services become increasingly similar, retailers must explore innovative marketing strategies to differentiate themselves and capture the attention of customers [32].

A recent report from Deloitte [2], reveals that the top 250 retailers have experienced significant revenue growth in FY2021, with a year-on-year increase of 8.5%. This is a marked improvement from the previous year's 5.2% growth. Additionally, the retailers have achieved a net profit margin of 4.3%, up from 3.3% in FY2020.

One of the key factors that holds influence is the growing prevalence of supermarkets and expansive self-service stores. These establishments offer a wide array of products, including fresh products, packaged foods, household items, and clothing. In the retail industry, especially in developed countries, they have gained significant prominence and contribute substantially to overall sales. The rise and widespread presence of supermarkets have profound effects on various aspects of the retail sector, leading to changes in consumer habits, supply chain dynamics, and pricing approaches.

At the core of the retail industry lies the reliance on data-driven decisions. With new market dynamics, advanced data analytics has become indispensable for retailers. It plays a vital role in identifying growth prospects, enhancing customer relationships, and adapting to ever-evolving consumer behavior. By harnessing the power of robust data analytics, retailers can delve into the knowledge of consumer patterns, fine-tune pricing strategies, forecast future demand trends, and streamline their supply chain operations. The effective application of data analytics sets retailers apart from their competitors, granting them a strategic edge in the highly competitive retail landscape [12].

In the era of digital transformation, the concept of "big data" has gained tremendous importance. Big data refers to the vast amount of structured and unstructured data generated from diverse sources. Its core elements include volume, velocity, variety, and veracity. Leveraging the power of big data opens up new possibilities for businesses and organizations, including the retail sector. Within retail, big data offers in-depth insights across five crucial dimensions: customers, products, time, location, and channel. For instance, the ability to track individual-level customer data has proven invaluable in boosting customer acquisition and increasing transaction value. Retailers have made significant progress in collecting comprehensive data on customer behavior, demographic information, and in-store visits through loyalty programs and various tracking methods. Furthermore, the abundance of product information allows analysts

to explore brand premiums and product similarities more extensively. The dimension of time is also vital, as ongoing measurements of customer behavior and product assortment have become feasible. Location data has taken a central role in driving marketing decisions and improving hyper-targeted strategies, while data from various channels helps retailers understand and trace customer journeys across multiple touchpoints [22]. The application of big data in the retail industry enables the development of more focused strategies and the precise measurement of their effectiveness. This has led to a growing demand for a data revolution in retailing, aiming to create more targeted strategies and achieve greater precision in evaluating their impact [22].

1.2 Problem Description and Goals

In recent times, the retail market has experienced a notable transformation driven by technological advancements, leading to intense competition among sellers. To maintain a competitive edge, retailers must consistently upgrade their operations, revenues, market shares, and overall user experience.

A concept that has gained popularity among both retailers and customers in recent years is the use of product comparison platforms. These platforms gather information about retail products from various sources and present comparisons between similar products available in different stores, along with their respective prices. This enables consumers to easily compare prices and features across different brands and retailers, empowering them to make well-informed purchasing decisions. Presently, the market offers numerous product comparison platforms, including Google Shopping, PriceGrabber, and Shopzilla, each with its own set of unique features and capabilities.

Despite the advantages offered by these platforms, users often encounter challenges related to their functionality. Two common issues are their limited adaptability to changing market conditions and the continuous maintenance required for these platforms, given the ever-evolving retail landscape, making it difficult for retailers to keep pace. Furthermore, the data utilized by these platforms are prone to inaccuracies and inconsistencies, which can undermine the effectiveness of the entire operation

Another challenge users often face is the overwhelming array of products available in the market, making it essential to identify those serving similar purposes. By showing these products together, customers can explore and discover new items that they may be interested in.

For instance, when a customer is searching for a particular product, like a kilogram of Basmati rice, it can be beneficial to group the rice products by their specific types, such as Basmati, Long Grain, Jasmine, and Black Rice, instead of displaying all the various types of rice available in a supermarket haphazardly. This grouping enables customers to compare prices, quantities, and other features of the same type of product, allowing for a more efficient shopping experience. Furthermore, identifying similar products can benefit companies in several ways. First, it provides valuable market insights, helping companies gain a better understanding of

their customers' preferences and behaviors. Additionally, this information can be used to build a robust recommendation system. Suppose a kilogram of basmati rice is unavailable; in that case, we can suggest an alternative brand of basmati rice. This ensures customer satisfaction and avoids any financial loss for the company.

An additional example illustrating our ultimate goal is depicted in Figure 1.1. Our primary objective is to autonomously segregate similar products. Specifically, for markers, it is logical to divide them into distinct categories such as Color Markers, Board Markers, Surligneurs, and Permanent Markers, as each group serves a unique purpose.

Figure 1.1: Categorizing Markers: Visual Representation of Color Markers, Board Markers, Surligneurs, and Permanent Markers Segregated into Distinct Groups.

The introduction of this feature in a successful product comparison platform capable of establishing accurate correspondences between similar products sold by different retailers and providing reliable price comparisons would be a highly innovative addition to any retail market.

This achievement hinges on two key components: data extraction and product matching. Data extraction serves as the initial step in the process and plays a critical role in the platform's functionality. Any inconsistencies in the extracted data can lead to incorrect insights, which can be detrimental to users. The second step, product matching, forms the platform's core, allowing it to combine identical products sold by different retailers.

Product matching is a complex task that can be approached through two main methods: text matching and image matching. Text matching involves comparing textual descriptions of products, while image matching entails evaluating product images to determine their similarity. Image matching poses particular challenges due to the variability in product images and the need for sophisticated algorithms to ensure accurate matching.

This dissertation explores an ongoing project at Deloitte, a leading global professional services

firm renowned for its excellence in audit, consulting, tax, and advisory services. This project employs Data Engineering and Data Science to integrate all data related to retailers, products, and stores into a unified platform.

Essentially, this thesis elucidates the process of identifying similar products and how this can be helpful in the app's recommendation system, as well as in a section where users can compare similar products considering that we only possess information about the products themselves and lack data about consumers or purchase history. To achieve this, we propose a method to group products based on their textual attributes. It is important to note that we work with unstructured and noisy text data, such as product names and descriptions.

Our results show that product names and simple descriptions, even lack of detailed text, are useful to characterize a product. The method can correctly find relevant groups for the datasets used in this work. It also produces better quality groups than the ones distinguished solely using the International Article Numbering (also known as European Article Number – EAN).

1.3 Document Organization

This dissertation is composed of 8 chapters: Introduction, Background, State of the Art, Project Setup and Data Preparation, Product Representation and Clustering of Similar Products, Results, Applications, and Conclusion.

Chapter 1: Introduction: The introductory chapter offers a concise contextualization of the subject matter. It introduces the problem, outlines the primary objectives, previews the analysis of results, discusses the methodology employed, and provides an overview of the document's structure and organization.

Chapter 2: Background: This chapter presents an overview of the research areas associated with this thesis: Recommendation Systems, Natural Language Processing, Clustering Algorithms, Metrics to find the Optimal Number of Clustering and Principle Component Analysis.

Chapter 3: State of the Art: Chapter 3 presents a comprehensive literature review of related works that share similarities with our research objectives. In this chapter, we conduct a detailed comparative analysis of their methodologies, elucidating the strengths and weaknesses of each approach.

Chapter 4: Project Setup and Data Preparation: This chapter elucidates the contributions of the undertaken work. Initially, we delve into the current project status, gaining a comprehensive understanding of the database and its contained data. Following that, we immerse ourselves in the complexities of data preprocessing, clarifying the strategies and methodologies we employ.

Chapter 5: Product Representation and Clustering of Similar Products: In Chapter 5, we delve into an exploration of diverse approaches, systematically tested to discover the most optimal model. Additionally, we provide insights into the tools and hardware employed throughout our research.

Chapter 6: Results The results chapter begins by discussing the results of data preprocessing, proceeds to elucidate the findings of the models that were tested and then analyzes how the algorithm models work with unobserved data. It ultimately concludes by offering a thorough examination of the results within the context of 2 specific categories.

Chapter 7: Applications In this chapter, we delve into the practical applications of the model introduced and elucidated in the preceding chapters. Specifically, we explore two key use cases: Similar Product Identification and Recommendation Systems. To illustrate the real-world utility of these applications, we provide a concrete example by applying them to a specific product category.

Chapter 8: Conclusions: This chapter begins with a research summary and then presents the main findings, conclusions, limitations and future work.

Chapter 2

Background

This chapter describes the definitions and terminology used in this project. First, section 2.1 introduces the product substitutes definition. Second, Section 2.2 introduces the Machine Learning concept and describes the 2 types of learning: supervised and unsupervised. Third, Section 2.3 describes the goals, vantages, types and data required for a good recommendation system. After that, Section 2.4 explains some common techniques used to deal with text variables. Section 2.5 introduces clustering algorithms and explains in detail the most commonly used clustering algorithms. The metrics to get the best number of clusters are explained in Section 2.6. Finally, Section 2.7 gives information about Principal Component Analysis, one of the most common techniques used for data reduction.

2.1 Substitute Products

Substitute products are products or services that fulfill similar customer needs or offer comparable features, benefits, or functionalities. They are alternatives that customers can choose instead of a particular product or service. In the context of substitute products, there are generally two types: perfect substitutes and imperfect substitutes. These terms are used to describe different degrees of similarity and substitution between products.

In the context of business and marketing, identifying substitute products is crucial for understanding market competition and consumer behavior. When a substitute product is available, it poses a competitive threat to an existing product or service. Customers may opt for the substitute instead, resulting in decreased demand for the original offering.

2.1.1 Perfect Substitutes

Perfect substitutes refer to a specific type of substitute products that have identical uses or functionalities. In other words, they are products that can be used interchangeably with no discernible difference in terms of their benefits, features, or purposes. Producers and sellers

of perfect substitute goods engage in direct price competition since the products are seen as completely interchangeable by consumers. For example, a 1-liter bottle of Coca-Cola and a 6-pack of Coca-Cola cans are considered perfect substitutes. This means that both the 1-liter bottle and the 6-pack of cans serve the same purpose, which is to provide the customer with Coca-Cola beverages. These products are termed perfect substitutes because they are functionally identical, allowing customers to use them interchangeably to satisfy their craving for a Coke.

2.1.2 Imperfect Substitutes

Imperfect substitutes refer to similar products that fulfill the same needs but possess slight variations in their characteristics. These substitutes are considered indirect competitors, as sellers of these goods compete for the same consumers. Although imperfect substitutes may serve the same purpose, their differences in features, branding, or other attributes make them distinct choices for consumers.

In certain situations, when the preferred 1-liter package of milk is unavailable, customers may consider purchasing an alternative milk product from a different brand. This flexibility arises from the customer's willingness to consider alternatives when their preferred choice is not available.

On the other hand, let's consider the scenario where a customer seeks to buy a 1-liter bottle of Coca-Cola, but it is currently out of stock. In this case, the customer's loyalty to the Coca-Cola brand might prevent them from considering an equivalent product from a competing brand, such as Pepsi. Despite the unavailability of their preferred choice, the customer's strong allegiance to the Coca-Cola brand influences their decision not to evaluate or opt for Pepsi as a substitute.

2.2 Machine Learning

Machine Learning (ML) is a sub-field of Artificial Intelligence (AI), and it has many applications in various fields, such as finance, healthcare, and even entertainment. This chapter aims to provide an overview of ML, including its key concepts, techniques, and applications. ML can be broadly categorized into two approaches: supervised and unsupervised.

2.2.1 Supervised Learning

Supervised learning is a type of ML where a model is trained on labeled data, meaning that the data is tagged with the correct output. The model can then predict future data based on the relationships and patterns it has learned from the training data [15].

Several advantages to using supervised learning can produce more accurate results than unsupervised learning, especially when working with a large amount of labeled data [31]. Another

advantage of supervised learning is that it allows for more controlled experimentation. Since the model is trained on labeled data, it is possible to evaluate its performance on a test data set and compare the predicted output to the true labels [36]. This allows for a more precise measurement of the model's accuracy and can help identify areas where the model may be lacking.

The requirement for labeled data in supervised models can pose a significant challenge, primarily due to the difficulty in acquiring such labels. This limitation stems from the need for extensive training data to achieve higher levels of accuracy in supervised learning.

Supervised learning is a powerful tool that can allow accurate predictions and recommendations based on patterns and relationships identified in the data [15]. This can help to increase sales and improve the customer experience [45].

2.2.2 Unsupervised Learning

Unsupervised learning is a type of ML where a model is given a dataset and must find patterns and relationships within the data independently; unlike Supervised Learning, labeled outputs are not used [15].

There are several advantages to using unsupervised learning to develop an algorithm. One significant advantage is that it does not require labeled data [36], which can be time-consuming and challenging to obtain. This makes unsupervised learning valuable for analyzing large amounts of data that may not be labeled. Additionally, unsupervised learning can help discover hidden patterns and relationships in the data that may not be immediately apparent [31]. Another advantage of unsupervised learning is that it can reduce the dimensionality of data. This can be especially useful for data sets with many variables [31].

However, there are also some disadvantages to using unsupervised learning. One major disadvantage is that the model needs to be guided on what patterns to look for, making it more challenging to interpret the results [31]. It can also be more computationally expensive than supervised learning, requiring the model to search for patterns and relationships in the data without guidance [31]. Finally, unsupervised learning may not be suitable for tasks that require a clear output or prediction, such as classification or regression [31].

In conclusion, unsupervised learning uncovers insights from unlabeled data and has diverse applications in various domains. Further research is needed to maximize its potential in data-driven discovery.

2.3 Recommendation Systems

Recommendation systems are technologies that utilize data from users' interactions, preferences, and feedback to provide personalized recommendations. These systems analyze past user behavior, such as ratings, purchases, and browsing history, to infer their interests and preferences. The

recommendations are typically made for items, such as movies, books, products, or content, that the user might be interested in [14].

They take advantage of various sources of data to understand customer interests. This data can be explicit, such as numerical ratings provided by users, or implicit, such as the simple act of purchasing or browsing an item. By analyzing these interactions between users and items, recommendation systems aim to identify patterns and dependencies that can help make accurate predictions about users' future choices.

Recommendation systems can also incorporate other types of data, such as contextual information (e.g., temporal, location, social, or network data) and domain knowledge to enhance the recommendation process. Advanced models may use additional data types and techniques to improve the accuracy and relevance of recommendations.

The field of Recommendation systems encompasses various types of systems, including collaborative, content-based, and knowledge-based systems.

Before discussing the goals of recommendation systems, we introduce the various ways in which the recommendation problem may be formulated. The two primary models are as follows:

1. *Prediction version of the problem*: This approach involves predicting the rating value for a user-item combination. Training data is available, which indicates user preferences for items. Represented as an incomplete $m \times n$ matrix for m users and n items, the observed values in the matrix are used for training, while the missing values are predicted using the training model. This problem is known as the matrix completion problem since it deals with predicting the remaining values in the incomplete matrix.

2. *Ranking version of the problem*: In practice, it's not always necessary to predict user ratings for specific items to make recommendations. Instead, a merchant may want to recommend the top-k items for a particular user or determine the top-k users to target for a specific item. The focus is more commonly on determining the top-k items, although the methods for both cases are analogous. Only the determination of top-k items is discussed since it is the more common scenario. This problem is referred to as the top-k recommendation problem, representing the ranking formulation of the recommendation problem.

In the ranking version, the absolute values of the predicted ratings are not crucial. The first formulation is more general, as the solutions to the second case can be derived by solving the first formulation for various user-item combinations and then ranking the predictions. However, in many instances, it is easier and more natural to design methods specifically for solving the ranking version of the problem directly.

The primary goal of a recommendation system is to increase product sales and profits for merchants. By recommending relevant items to users, these systems aim to attract their attention and increase sales volume. While revenue generation is the main objective, there are other important operational and technical goals of recommendation systems:

1. Relevance: The most crucial operational goal is to recommend items that are relevant to the user. Users are more likely to consume items that interest them.

2. Novelty: Recommendation systems are useful when they recommend items that users have not seen before. Repetitive recommendations of popular items can reduce sales diversity, so introducing novel choices is important.

3. Serendipity: Serendipity refers to recommending items that are unexpected and surprising to the user. These recommendations go beyond novelty and can uncover latent interests or initiate new trends. Although serendipitous recommendations may sometimes be irrelevant, their long-term benefits outweigh the short-term disadvantages.

4. Increasing recommendation diversity: Recommendation systems typically provide a list of top k items. To ensure user satisfaction, it is important to recommend diverse items of different types. This prevents users from getting bored with repetitive recommendations of similar items.

Overall, recommendation systems aim to increase sales by recommending relevant, novel and diverse items while incorporating an element of surprise and user interest.

2.3.1 Collaborative Filtering

Collaborative Filtering (CF) recommendations are based on the premise that a user should like the items favored by a similar user. Usually, it does not assume that the current user is aware of preferences from similar users. Instead, the recommendation system is charged with finding similar users based on the preferences of the current user and then decide which are the most interesting items to be recommended [28].

The data used in CF, named user feedback, states the degree of preference (feedback) a user has provided towards a given item [20]. User feedback can be categorized into explicit and implicit feedback. Explicit feedback involves users knowingly expressing their preferences for items. This can be in the form of numerical ratings on a Likert scale, ranked lists of preferred items, or binary indicators of favorability. On the other hand, implicit feedback is derived from user behavior within a domain, such as click-through data and the duration of time spent on a web page. Implicit feedback, also known as positive-only feedback, only captures user interest and does not provide information about disinterest.

CF algorithms can be divided into two classes: memory-based and model-based [20] [57]. Memory-based algorithms apply heuristics on a rating matrix to compute recommendations, whereas model-based algorithms induce a model from a rating matrix and use this model to recommend items. Memory-based algorithms are mostly represented by Nearest Neighbor algorithms, while model-based algorithms are mostly based on Matrix Factorization.

2.3.2 Content-based Filtering

Content-based Filtering (CBF) recommendations propose the use of item properties to drive recommendations. This rationale implies that if a user bought an item from a specific category in the past, the user will be probably interested in a new item from the same category in the future [28].

Most CBF methods take advantage of items the user found interesting in the past to serve as initial feedback. Next, similarity calculations are performed to find and recommend the most similar items [20]. Each item is described by several properties, which depend on the domain used. For instance, movies can be described by their actors and studio, while in music, artists and album properties can be used.

CBF typically uses a vector space model to represent items and their properties. In this representation, each row represents a different item and each column is a property of this item. With this formulation, the similarity between items is simply given by the similarity of their vectors (i.e. rows in the vector space model) [28].

2.4 Natural Language Processing

The core objective of Natural Language Processing (NLP) is to enable computers to understand and utilize language similar to humans, thus enhancing machine-human communication and permitting the extraction of valuable insights from text data [27]. The vast domain of NLP comprises various tasks and areas of focus.

Among these, textual processing stands out, centering on understanding and manipulating textual data. Several tasks fall under this umbrella, including but not limited to tokenizing, parsing, lemmatization, and part-of-speech tagging. Similarly, information extraction, another critical area within NLP, strives for the automated extraction of structured information from unstructured text data. This extraction process draws out entities, relationships, and other valuable insights that enrich the understanding and utility of the text.

Tackling meaning extraction from text is a significant hurdle in NLP. Structured data like numbers or dates contrast starkly with text, which teems with complexity, ambiguity, nuance, and cultural references, often perplexing for machines to decipher. In response to these challenges, NLP research has developed various techniques that employ statistical models, ML algorithms, and deep learning to analyze and interpret text [4].

In conclusion, NLP is a riveting field that continues to reshape interactions with computers. Despite the challenges ahead, the future of NLP promises many advancements.

In the next sections, we explain some NLP techniques.

2.4.1 Tokenization

In the context of artificial intelligence and natural language processing, tokenization refers to the process of breaking down a text or a sequence of characters into smaller units called tokens [47].

These tokens can be individual words, subwords, or even characters, depending on the specific requirements of the task or the language being processed. Tokenization is typically the first step in many natural language processing tasks, such as text classification, named entity recognition, and machine translation.

By breaking down the text into tokens, the AI system gains a better understanding of the underlying language and can perform various linguistic analyses and computations on the tokenized units.

2.4.2 Stemming

Stemming is a process by which word endings or other affixes are removed or modified in order that word forms that differ in non-relevant ways may be merged and treated as equivalent. A computer program that performs such a transformation is referred to as a stemmer or stemming algorithm. The output of a stemming algorithm is known as a stem [11].

Stemming algorithms can be rule-based or dictionary-based. Rule-based algorithms follow predefined rules to modify words, such as removing suffixes like "-ing". On the other hand, dictionary-based algorithms refer to a dictionary or lexicon to determine whether to apply a rule or not. For instance, a dictionary-based algorithm would not remove "-ing" if the word is already listed in the dictionary [47].

The impact of stemming varies across languages. In English, stemming has a relatively small effect, typically resulting in a slight increase in recall. However, in other languages like German, stemming becomes more crucial due to complex word constructions. In languages such as Finnish, Turkish, Inuit, and Yupik, which have recursive morphological rules, stemming is particularly important since these rules can generate words of unbounded length. In the case of the French language, stemming is also significant due to its complex morphology, marked by intricate inflectional and derivational processes. For example, verbs in French can have multiple conjugated forms based on tense, mood, and subject. Stemming would reduce these forms to the infinitive form [47].

2.4.3 Embeddings

Embeddings refer to a representation of words or other linguistic units as dense vectors in a continuous vector space. These vectors capture semantic and syntactic relationships between words, allowing for the modeling of various linguistic phenomena. [34]

Traditional approaches to natural language processing often relied on sparse representations of words, such as one-hot encoding, where each word is represented by a binary vector with a length equal to the vocabulary size, and only one element indicating the presence of that word. However, such sparse representations lack the ability to capture the meaning and relationships between words.

Embeddings, on the other hand, provide a continuous representation where words with similar meanings or contexts are mapped to vectors that are close together in the embedding space. These dense vectors encode semantic and syntactic information, allowing NLP models to exploit the relationships between words.

Overall, embeddings in the context of NLP refer to continuous vector representations of words that capture semantic and syntactic relationships, enabling more effective and nuanced language modeling and understanding.

In the following subsections, we will delve into a detailed exploration of various embedding algorithms.

2.4.3.1 Bag of Words

The bag-of-words (BOW) representation treats a document as an unordered "bag" of words, where the frequency or presence of each word is used as a feature. This approach discards the order in which words appear in the document and focuses solely on the presence or absence of words and their frequencies. The resulting representation is often a vector, where each dimension corresponds to a unique word in the vocabulary, and the value in each dimension represents the count or occurrence of that word in the document [34].

In the example in Figure 2.1, instead of representing the word order in all the phrases like *"I love this movie"* and *"I would recommend it"*, we simply note that the word *"I"* occurred 5 times in the entire excerpt, the word *"it"* 6 times, the words *"love"*, *"recommend"*, and *"movie"* once, and so on [34].

However, the bag-of-words approach has limitations, as it ignores the context and order of words, resulting in a loss of sequential and syntactic information. Nevertheless, it remains a valuable and widely used technique due to its simplicity and efficiency, especially when dealing with large-scale text data.

Overall, the bag-of-words approach refers to a simple representation of text documents as a collection of words, ignoring word order and focusing on word frequencies or presence, which is often used in various NLP and information retrieval tasks.

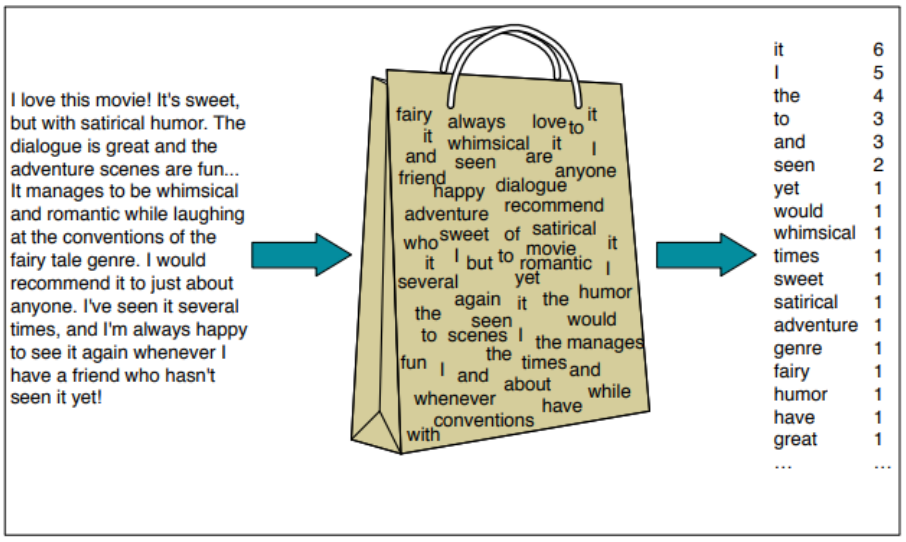


Figure 2.1: Intuition of the BOW Applied to a Movie Review: The position of the words is ignored (the BOW assumption) and we make use of the frequency of each word. (Source: [34])

2.4.3.2 Term Frequency-Inverse Document Frequency

Term Frequency-Inverse Document Frequency (TF-IDF) is a statistical measure used in information retrieval and text mining [6] to evaluate the importance of a term in a document within a collection of documents. It combines two measures to compute a weight for each term:

1. **Term Frequency (TF)**: This measures the frequency of a term within a document. It indicates how often a term occurs in a specific document. The assumption is that the more frequent a term is within a document, the more important it is to that document.
2. **Inverse Document Frequency (IDF)**: This measures the rarity of a term across a collection of documents. It calculates the logarithm of the total number of documents divided by the number of documents containing the term. The IDF score decreases as the term appears in more documents, thus assigning a higher weight to terms that are more specific or rare to a particular document.

By combining these two measures, TF-IDF assigns a weight to each term in a document that reflects its importance. Terms that appear frequently in a document but rarely in the entire collection will receive a higher weight, indicating their significance in distinguishing that document from others [34].

In practice, TF-IDF is calculated as the product of the term frequency (TF) and the inverse document frequency (IDF):

$$\text{TF-IDF}(t, d) = \text{TF}(t, d) \times \text{IDF}(t) \tag{2.1}$$

where TF(t, d) represents the term frequency of term t in document d, and IDF(t) represents the inverse document frequency of term t across the entire document collection.

Overall, TF-IDF is a method to assign weights to terms in a document collection based on their frequency within documents and rarity across the collection. It helps to identify important terms and distinguish documents from each other in information retrieval and text mining tasks.

2.4.3.3 Word2Vec

Word2Vec assumes that two words are similar (and have similar representations) if they have similar contexts [38]. In this case, the context refers to a predefined amount of neighboring words. One architecture proposed to learn these representations is the skip-gram, which predicts surrounding words given the current word. For such, each target word w_t, represented as the one-hot encoding for a vocabulary V, is connected to a hidden layer h. This hidden layer, where the distributed representations are, has a predefined size d. Each distributed representation is connected to the previous and next c context words (i.e. $w_{t-c}, w, ..., w_{t-1}, w_{t+1}, w, ..., w_{t+c}$) [28].

2.4.3.4 GPT-2 Embeddings

GPT-2 is a large transformer-based language model with 1.5 billion parameters, trained on a dataset of 8 million web pages. GPT-2 is trained with a simple objective: predict the next word, given all of the previous words within some text. The diversity of the dataset causes this simple goal to contain naturally occurring demonstrations of many tasks across diverse domains. GPT-2 is a direct scale-up of GPT, with more than 10X the parameters and trained on more than 10X the amount of data [3] [42].

2.5 Clustering Algorithms

Clustering is a fundamental technique in machine learning and data analysis that involves grouping similar data points together based on their inherent characteristics. It is an unsupervised learning approach that aims to identify patterns, structures, or relationships within a dataset without any predefined labels or categories.

The goal of clustering is to partition a dataset into clusters, where data points within the same cluster are more similar to each other compared to those in other clusters. These clusters can represent natural groupings or clusters that may exist in the data, providing valuable insights and facilitating further analysis [41].

2.5.1 Types of clustering

The relevant types of clustering for this research are Partition-based Clustering, Hierarchical Clustering and Density-based Clustering.

Partition-based clustering methods involve dividing the data into distinct clusters. Algorithms like K-means, K-medoids, and Fuzzy C-means are popular examples of this type. They aim to minimize an objective function that quantifies the dissimilarity or distance between data points and cluster centers. Partition-based clustering is efficient and suitable for large datasets [41].

Density-based clustering methods identify clusters based on the density of data points. They aim to discover regions of high density separated by regions of low density. Density-based algorithms, such as DBSCAN and OPTICS, consider the notion of density and connectivity to define and identify clusters. These methods can handle clusters of arbitrary shapes and are effective in detecting outliers [41].

Hierarchical clustering methods create a hierarchical structure of clusters. They build a dendrogram, which is a tree-like structure representing the relationships between clusters at different levels of granularity. Agglomerative clustering merges the closest clusters iteratively, while divisive clustering recursively splits clusters. Hierarchical clustering provides a visual representation of the clustering structure, facilitating interpretation [41].

2.5.2 K-means

K-means clustering is a widely employed unsupervised learning algorithm in the field of ML (Figure 2.2). Its main goal is to group data points into distinct clusters, revealing patterns and structures within high-dimensional datasets. Its effectiveness has been demonstrated in customer segmentation [35], image segmentation [29], and anomaly detection [39].

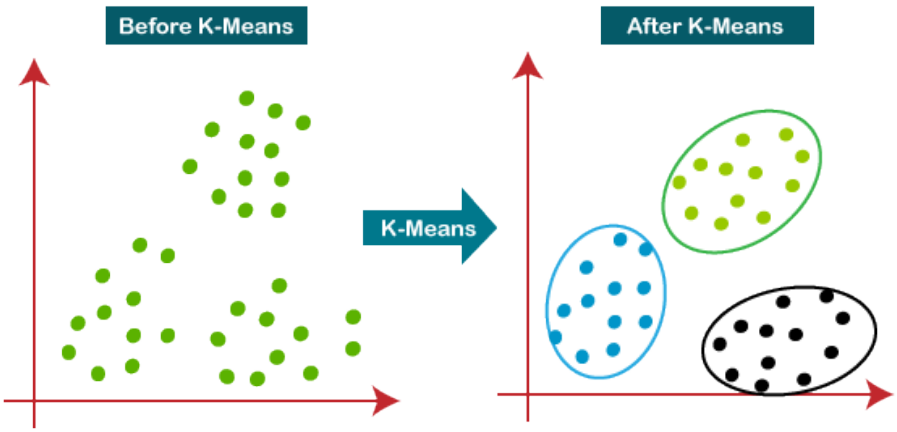


Figure 2.2: K-means Clustering Visual Representation [5]

The algorithm iteratively assigns data points to the nearest cluster centroid and updates the

centroids based on the mean of the assigned data points. Convergence is reached when data point assignments to clusters stabilize.

K-means clustering aims to partition data points into K clusters represented by centroids, minimizing the sum of squared distances between each data point and its assigned centroid [19].

Mathematically, the objective of K-means clustering can be expressed as follows:

$$\text{Minimize: } \sum_{i=1}^{K} \sum_{x \in C_i} |\tilde{x} - \mu_i|^2$$

Each data point x is assigned to a cluster in this equation based on its distance to the centroid μ_i. The distances determine the assignment, and C_i represents the data points assigned to the i-th cluster [19].

The optimization process involves two steps: assignment and update. In the assignment step, data points are assigned to the cluster with the closest centroid. In the update step, the centroids are recalculated based on the mean of the data points assigned to each cluster.

K-means clustering is known for its simplicity and efficiency, capable of handling large datasets with thousands to millions of data points. However, the curse of dimensionality can impact its performance when dealing with high-dimensional data. To mitigate this, dimensionality reduction techniques like principal component analysis can be combined with K-means clustering [30].

In conclusion, K-means clustering is an efficient and straightforward algorithm with practical applications in various domains. It is expected to remain relevant in the field of ML research and development.

2.5.3 DBSCAN

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) has emerged as a popular unsupervised learning algorithm in ML. Its primary objective is to group data points into clusters based on their density, offering significant advantages when dealing with arbitrary shapes and noise data. The algorithm has found successful applications across diverse domains, including anomaly detection, image segmentation, and social network analysis [10].

At the core of DBSCAN's functionality lie two essential parameters: ϵ, which defines the neighborhood radius around each data point, and MinPts, representing the minimum number of data points required to form a dense region. DBSCAN effectively partitions data points into clusters by considering both connectivity and density.

DBSCAN tackles the clustering problem by categorizing data points into three distinct types [40]:

- **Core points**: These data points have at least MinPts data points within a distance of ϵ,

indicating a dense region.

- **Border points**: Data points that do not meet the core point criteria but are within a distance of ϵ from a core point are classified as border points.
- **Noise points**: Any data points that are neither core nor border points are considered noise points.

To address the clustering challenge, DBSCAN employs an iterative algorithm. It begins with an arbitrary data point and progressively expands the neighborhood to incorporate all core and border points within the ϵ distance. This iterative process continues until all dense regions have been identified.

One of the notable strengths of DBSCAN is its ability to handle clusters with arbitrary shapes and densities. It excels in identifying non-linearly separable clusters and can effectively handle noisy data and outliers.

DBSCAN is also well-suited for analyzing high-dimensional data, although it is essential to account for the curse of dimensionality and its impact on performance. DBSCAN can be combined with dimensionality reduction techniques, such as Principal Component Analysis, to mitigate this issue, similar to the approach employed in the previous section with K-means.

In summary, DBSCAN is a versatile and efficient algorithm for clustering applications. It provides valuable insights into complex data structures and offers practical solutions to real-world problems across various domains. Its ability to handle data of arbitrary shapes and noise makes it a valuable tool in ML [40].

2.5.4 Hierarchical Clustering

Hierarchical Clustering groups (Agglomerative also called as Bottom-Up Approach) or divides (Divisive also called as Top-Down Approach) the clusters based on the distance metrics.

In agglomerative clustering, initially, each data point acts as a cluster, and then it groups the clusters one by one. This comes under one of the most sought-after clustering methods [50].

Divisive is the opposite of Agglomerative, it starts off with all the points into one cluster and divides them to create more clusters. These algorithms create a distance matrix of all the existing clusters and perform the linkage between the clusters depending on the criteria of the linkage. The clustering of the data points is represented by using a dendrogram. There are different types of linkages [50]:

- **Single Linkage:** In single linkage the distance between the two clusters is the shortest distance between points in those two clusters.
- **Complete Linkage:** In complete linkage, the distance between the two clusters is the farthest distance between points in those two clusters.

- **Average Linkage:** In average linkage the distance between the two clusters is the average distance of every point in the cluster with every point in another cluster.

2.5.5 Probabilistic Models: Gaussian and Bayesian Gaussian

A Gaussian Mixture Model (GMM) is a parametric probability density function represented as a weighted sum of Gaussian component densities. GMMs are commonly used as a parametric model of the probability distribution of continuous measurements or features in a biometric system, such as vocal-tract-related spectral features in a speaker recognition system. GMM parameters are estimated from training data using the iterative Expectation-Maximization (EM) algorithm or Maximum A Posteriori (MAP) estimation from a well-trained prior model [43].

A Gaussian mixture model is a weighted sum of M component Gaussian densities as given by the equation [43],

$$p(x|\lambda) = \sum_{i=1}^{M} w_i g(x|\mu_i, \Sigma_i)$$

where x is a D-dimensional continuous-valued data vector (i.e. measurement or features), w_i, $i = 1, \ldots, M$, are the mixture weights, and $g(x|\mu_i, \Sigma_i)$, $i = 1, \ldots, M$ are the component Gaussian densities, with $\sum_i$, the covariance matrix.

Bayesian Gaussian refers to the application of Bayesian statistics in the context of Gaussian (or normal) distributions. The term combines Bayesian principles with the properties and characteristics of the Gaussian distribution.

2.6 Optimal number of clusters

Selecting the ideal number of clusters is a critical aspect of clustering analysis, as it facilitates the efficient grouping of data points into meaningful clusters. Various methodologies exist to ascertain this optimal number of clusters, utilizing diverse mathematical metrics to evaluate the quality of the system. In the following sections, we explain each method.

2.6.1 Elbow Method

The Elbow method [55] is the oldest method to find the potential optimal cluster. The basic idea is to specify K=2 as the initial value and then keep increasing K by 1 to the maximal specified value. The optimal cluster number K is distinguished by the fact that before reaching K, the cost rapidly decreases to the called cost peak value, and after exceeding K, it continues to increase with the called cost peak value almost unchanged, as shown in Fig. 2.3 a) with an explicit elbow point.

However, a challenge with the Elbow method arises when the plotted curve is relatively smooth, as depicted in Fig. 2.3 b), making it difficult for experienced analysts to definitively identify the elbow point due to its ambiguity. [51].

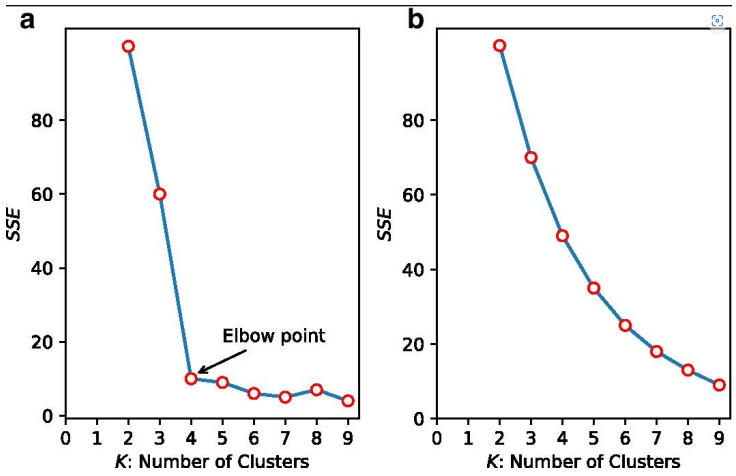


Figure 2.3: a) A Visual Curve with an Explicit Elbow Point. b) A Visual Curve being Fairly Smooth with an Ambiguous Elbow Point [51]

2.6.2 Silhouette Method

The Silhouette method [44] [46] is another well-known method with decent performance to estimate the potential optimal cluster number, which uses the average distance between one data point and others in the same cluster and the average distance among different clusters to score the clustering result. The metric of scoring of this method is named the silhouette coefficient (S), and S is defined as $\frac{b-a}{\max(a,b)}$, where a and b represent the mean intra-cluster distance and the mean nearest-cluster distance, respectively. The interval of the S values is $-1 \leq S \leq 1$. A value of S closer to 1 indicates that a sample is better clustered, and if it is closer to -1, the sample should be categorized into another cluster. This method is preferable for estimating the potential optimal cluster number. Meanwhile, the silhouette index can evaluate the best number of clusters in most cases for many distinct scenarios [16].

2.6.3 Calinski Harabasz Score

The Calinski-Harabasz index - also known as the Variance Ratio Criterion - can be used to evaluate the model, where a higher Calinski-Harabasz score relates to a model with better defined clusters [9].

For a set of data E of size n_E which has been clustered into k clusters, the Calinski-Harabasz score is defined as the ratio of the between clusters dispersion means and the within-cluster dispersion:

$$s = \frac{tr(B_k)}{tr(W_k)} \cdot \frac{n_E - k}{k - 1}$$

where $tr(B_k)$ is the trace of the between group dispersion matrix and $tr(W_k)$ is the trace of the within-cluster dispersion matrix defined by:

$$W_k = \sum_{q=1}^{k} \sum_{x \in C_q} (x - c_q)(x - c_q)^T$$

$$B_k = \sum_{q=1}^{k} n_q (c_q - c_E)(c_q - c_E)^T$$

with C_q the set of points in cluster q, c_q, the center of cluster q, c_E the center of E, and n_q the number of points in cluster q.

Advantages include assigning higher scores to clusters that are dense and well-separated, in line with standard cluster criteria. It also offers the advantage of quick computation. However, a notable disadvantage is its tendency to favor convex clusters with higher scores, potentially showing bias against other cluster types, such as those generated by DBSCAN, which are density-based [9].

2.6.4 Akaike Information Criterion

The Akaike information criterion (AIC Score) [7] [54] is one of the mathematical formulations that may be used for model selection and assessing parsimony in model structure. The equation of this criterion is defined as follows [21]:

$$AIC = -2ln(\sigma_n^2) + 2k$$

where n is the number of residuals, σ_n^2 maximum likelihood of the residuals' variance, and k is the sum of model parameters as $k = p + q + P + Q$ [21]. If the least square estimation is used and the errors are normally distributed, then the σ_n^2 equals the variance of the residuals from the fitted model and the resulting equation will be:

$$AIC = nln(\sigma_n^2) + 2k$$

where $2k$ is the penalty term added to prevent the formula from choosing a model with too many parameters. However, in the case of a data set containing a small number of sample sizes, Akaike's information criterion (AIC) method may tend to overestimate the parameters and select models with a greater number of parameters. The goal is to minimize AIC to obtain the best generalization [49].

2.6.5 Bayesian Information Criteria

When different Markov chains of different orders can possibly fit the data well, a comparison tool is required. Even if statistical tests exist in the case of Markov chains, a much more common

approach is now to rely on the Bayesian information criterion (BIC). This criterion is defined as [17] :

$$BIC = -2LL + klogn$$

where LL is the log-likelihood of the model, k is the number of independent parameters, and n is the sample size. By integrating a penalty term depending on the number of independent parameters, BIC tends to favor parsimonious models. The model achieving the lowest BIC value is chosen as the best model. Following Raftery's approach, we consider that a difference of BIC lower than 2 between two models is barely worth mentioning, a difference between 2 and 5 is positive, a difference between 5 and 10 is strong, and a difference larger than 10 is very strong. [8].

2.7 Principal Component Analysis

Principal Component Analysis (PCA) is a dimensionality reduction technique commonly used in data analysis and machine learning. The goal of PCA is to transform a high-dimensional dataset into a lower-dimensional space while retaining as much of the original information as possible. PCA achieves dimensionality reduction by identifying the principal components of the data. These components are linear combinations of the original features that capture the maximum variance in the data. The first principal component accounts for the largest amount of variability, followed by the second, third, and so on [1].

PCA is predominantly used as a dimensionality reduction technique in domains like facial recognition, computer vision and image compression. It is also used for finding patterns in data of high dimension in the fields of finance, data mining, bioinformatics, psychology, among others [1].

PCA has various applications, such as feature extraction, data visualization, and noise reduction. It helps to identify the most significant features, remove redundant information and simplify complex datasets, allowing for more efficient analysis and modeling.

Chapter 3

State of the Art

In order to present and develop a new contribution to the scientific literature, we must understand the current state of the art in the different disciplines that are related to this research. In this chapter, we present a description of the scientific work related to the main stages of this thesis. We begin by outlining our methodology for selecting pertinent papers for analysis in Section 3.1. Subsequently, we delve into a comprehensive discussion of the most relevant works, in section 3.2. We conclude this chapter with a summary in Section 3.3, providing an overview of the chapter's key points, along with an examination of the advantages and disadvantages associated with the research methods discussed.

3.1 Methodology

To understand the current state of the art, first, we search for papers in IEEEXplore and Scopus platforms with the queries shown in Table 3.1. These queries were built based on concepts and expressions related to the scope of this work.

All searches were performed on 30^{th} of January 2023.

We initially collected 1,040 articles using the specified queries. Subsequently, we conducted a title review, leading to the selection of 71 articles. Following a thorough analysis of abstracts, 51 articles were identified as relevant. Finally, upon a comprehensive reading of the entire text, 37 articles were deemed highly pertinent to our research objectives. The entire selection process is illustrated in Figure 3.1.

While we were reading these articles, we understood that there are different types of solutions to way out with the problem. Therefore, after analyzing the text of the 37 articles selected, we categorized the articles based on the type of solution they implemented, as illustrated in Table 3.2.

Although all of these types of solutions can be good ideas to face the problem of knowing which is the best product to recommend to a user, we have to take into consideration that our

Table 3.1: Number of Articles Retrieved for each Query

Query	**Engine**	**Number Papers**
("product similarity" OR "similarity of products" OR "similar products") AND NOT ("image-based" OR "gene" OR "bioinformatics" OR "3d" OR "twitter" OR "radio" OR "gas" OR "gases" OR "satellite" OR "medical" OR "network" OR "visual" OR "decision" OR "software" OR "robotic" OR "temperature" OR "fingerprint" OR "computer" OR "sustainability" OR "voice" OR "storage" OR "energy" OR "environmental" OR "electronic" OR "hotel" OR "forecast" OR "automotive" OR "medicine", OR “cloud”)	IEEEXplore	56
	Scopus	321
("product similarity" OR "similarity of products" OR "similar products") AND (“textual” OR “text analysis” OR “nlp”) AND NOT (“biomedical” OR “medicine” OR “medical” OR “gene” OR “sentimental” OR “coordinate matching” OR “cloud” OR “hotel” OR “second-hand” OR “video”)	IEEEXplore	15
	Scopus	44
(("product similarity" OR "similarity of products" OR "similar products") AND ("recommendation system") AND NOT ("fashion" OR "genetic" OR "image-based" OR "restaurant"))	IEEEXplore	8
	Scopus	78
("product similarity" OR "similarity of products" OR "similar products") AND ("machine learning") AND NOT ("reviews" OR "disease")	IEEEXplore	10
	Scopus	165
(("product similarity" OR "similarity of products" OR "similar products" OR "product recommendation" OR "recommendation of products") AND ("clustering"))	IEEEXplore	23
	Scopus	320

database can not have the information needed to use some solutions. For example, the database does not have information about the users' purchase history, which means that we cannot use solutions based on collaborative filtering (rows 1 and 3). In the same way, we do not have the user's opinion of a product, so we can also discard the papers that use users reviews (row 6).

Due to the observed in Table 3.2, some papers use techniques based on image analysis (row 4), nonetheless, this approach does not look the best one for recommending the most similar supermarket products. For instance, if we have a blue can of tuna and a blue can of sardines from the same brand, they are likely to be similar, but the contents of the cans are different. Additionally, using a solution based on image similarity is inefficient on a data set with a large

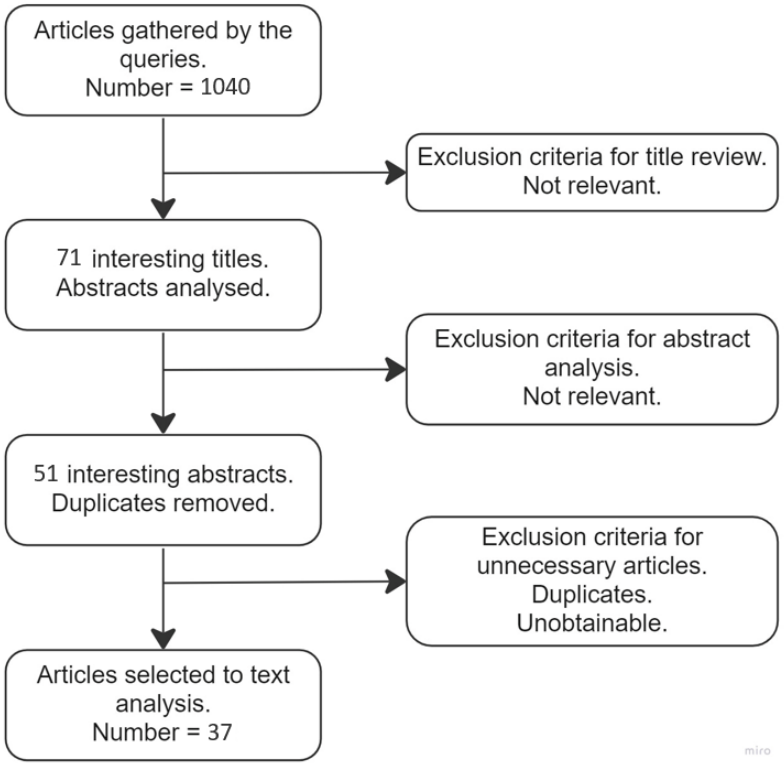


Figure 3.1: Procedure to Select the Best Articles

Row	Method	Number articles
1	Collaborative filtering	6
2	Text Analysis	9
3	Collaborative Filtering + Item Similarity	8
4	Image Analysis + Text Analysis	4
5	Ontology	2
6	Based on user reviews	4
7	Topic Modelling	2
8	Others	2

Table 3.2: Number of Articles Selected Divided by Method

set of images.

The methodologies based on ontology (row 5) will not work because the product description, product name and other fields with text are short, so we discarded them.

We have chosen to avoid using topic modeling (row 7) due to its high computational requirements and sensitivity to noise. Given our extensive dataset, which already contains inherent noise, we believe it is prudent to explore alternative approaches. Topic modeling algorithms demand substantial computational resources, making them impractical for large-scale applications. Additionally, these algorithms can be sensitive to data noise, leading to less accurate

or inconsistent topic assignments.

Finally, "Others" (row 8) should be understood as articles that have not used any of the techniques described above, but that aim to improve the recommendation of a product. There are only 2 articles on this situation. One of them takes into account eye gaze data which, according to this article [48] is very effective. Another article presents an architecture that aims to learn context-dependent vector representations of products from online session logs [18]. These 2 articles can be discarded because, just like the other articles mentioned above, there is no such information in the database.

To sum up, we choose only the ones that use text analysis (row 2) to recommend the most similar products to a user.

3.2 Analysis

In the last section, we explained the process of filtering the articles until we reached a reasonable number for detailed discussion. In this section, we will present the methodologies and conclusions of each of the 9 papers that utilized text analysis solutions.

All of the articles started by creating a data set and then performed a data cleaning. After that, they all group products by category before applying machine learning techniques.

Cherednichenko et al. [26] proposed a method that uses the description to cluster products of the same category. They used Term Frequency-Inverse Document Frequency (TF-IDF) and k-means to cluster products. They concluded that the k-means gives an acceptable result, because, recall and precision were 0.95. Additionally, they concluded that it only works well for data uniformly distributed by categories. Similarly, Mathivanan et al. [37] employed a similar technique to categorize large-scale e-commerce products from an online store website in Malaysia focusing on hair and face, oral and pet care products, but prior to applying k-means, they utilized PCA to reduce the dimensionality of the data. Their study concluded that this approach successfully clustered the data into distinct groups and that PCA significantly improved algorithm performance.

Cardoso et al. [23] describe in detail the architecture and methodology implemented at ASOS, one of the world's largest fashion e-commerce retailers. They use a set of images, a text description as well as product type and brand and build a multi-modal multi-task architecture to predict attributes of the fashion products such as "shirt type".

Shrivastava and Sisodia [52] initiated their approach by vectorizing the text associated with a product. Specifically, they focused on vectorizing the product titles, emphasizing that a sufficiently descriptive title is essential for accurate representation. If a title contains only a few words, it may not provide an adequate description of the product, potentially leading to inaccurate recommendations. Consequently, such titles were deemed unsuitable for vectorization. The vectorization process involved utilizing Bag of Words (BOW) and TF-IDF techniques. After

obtaining the vectors, Euclidean similarity was applied to identify the top k recommendations, based on the products with the highest similarity scores. The authors concluded that both BOW and TF-IDF proved to be highly practical approaches for text vectorization, leveraging word frequency and Inverse Document Frequency to effectively capture the essence of a text description or a corpus.

Chandra et al. [25] applied CountVectorizer on the bag of words column on preprocessed data to compute the cosine similarity between all the movies for generating the recommendation. After creating the cosine similarity matrix, they have created a simple recommendation function that takes the title of a movie entered by a user and recommends the top ten most similar movies to it.

Tantyo and Mauritsius [56] used the Approximate Nearest Neighbor model to find the equivalent products. The input of the model is a vector with a length of 128 dimensions that is the concatenation of price (after normalizing), category name (after One Hot Encoding) and output of word2Vec with the description of a product as input. The output is the id of the product recommended by the model for each product. They used 5 categories of the data set to test and evaluate the model and they concluded that Approximate Nearest Neighbor plays an important role in getting recommendations faster than KNN by only reducing a small level of performance.

Siswanto, Tjong and Saputra [53] tried 2 different approaches to do a vector representation of E-commerce products. The first step of each approach was to train the model with a set of product names using word2Vec, then, for each product name, they combined all of the word vectors using two different methods: 1) unweighted average and 2) weighted average with noise correction. Two different tasks were used for evaluation: calculating product similarity and integrating the embedding as a feature for predicting product category.

Goswami and Tilottama [24] conducted a case study to investigate the comparative effectiveness of three methods for identifying product similarity: text-based, semantic text-based, and visual similarity-based. The authors evaluate the four algorithms using the ranking measure of correctness metrics, i.e. Mean Reciprocal Rank (MRR). They concluded that the best approach is visual similarity based on product similarity. Additionally, the authors suggest that no single algorithm is suitable for all recommender systems and algorithms based on the type of user preferences, ensemble or hybrid are more promising.

Adak and Uçar [13] present a novel approach to enhancing book recommendations on Amazon e-commerce sites through the integration of decision tree-based fuzzy logic. In this study, unsupervised learning is applied by using the data of the books also bought-viewed, and the books are clustered into six different clusters. The books in the clusters are examined, and the collections are listed from Cluster 1 to Cluster 6 according to the closeness of the book types. Thus, another parameter is created to be used in the fuzzy model. A decision tree model is created with the data set obtained after data cleaning and clustering. This decision tree model is used for the rules of the fuzzy model to be created later. For the fuzzy model, the Jfuzzylogic

library is used in the Java environment.

3.3 Summary

Most of the works use some kind of vectorization of product description and a distance metric to measure product similarity. The most common approaches for vectorization are word2Vec and TF-IDF. TF-IDF is advantageous for its simplicity and interpretability, providing a straightforward way to quantify term importance in a document, but it may struggle to capture semantic relationships between words. On the other hand, word2Vec excels in capturing semantic meanings and contextual relationships, yet it requires substantial data and computational resources for effective training and might not always be interpretable.

The works presented highlight that product similarity is a widely used methodology in e-commerce, and two common distance metrics for measuring this similarity are Euclidean Distance and Cosine Similarity. Cosine similarity is advantageous for measuring similarity between vectors in high-dimensional spaces as it considers the angle between vectors, making it robust to differences in vector magnitudes. On the other hand, Euclidean distance is straightforward and intuitive, but it can be sensitive to the scale and dimensionality of the data, potentially leading to misleading similarity measurements.

Summarizing, works in the literature show the relevance of embedding algorithms, clustering algorithms, and product similarity approaches in effectively addressing the challenge of identifying similar products. In the next chapters, we explore some of these methods to characterize and measure product similarity in a set of products available from Deloitte. Some of the methods used successfully in these papers will be used in our work.

Chapter 4

Project Setup and Data Preparation

This chapter provides a comprehensive overview of the design process and key development decisions made to prepare the data for analysis.

Section 4.1 offers insights into the task and provides context regarding the data. In Section 4.2, we delve into the details of the database. Section 4.3 focuses on the selection of pertinent columns from the original database. Section 4.4 presents valuable insights derived from the dataset. Lastly, in Section 4.5, we describe the data preparation process. In Section 4.6 we summarize this chapter.

4.1 Project Description

The Retail Match project, being developed at Deloitte, is a comprehensive data engineering and data science framework designed to ingest, store, and analyze thousands of retail products from multiple retailers, as shown in Figure 4.1. This project results from the collaboration between a team of data engineers, data scientists, and retail experts who deeply understand the retail market and its challenges.

The data engineering process is developed on-demand in a serverless architecture using the Cloud service Amazon Web Services (AWS), which provides an efficient and scalable solution for handling large amounts of data.

The ingestion process is carefully orchestrated on a specific schedule for each retailer to ensure that all information is up-to-date. The process involves data extraction from the retailer's websites, which is delicate due to the nuances of each retailer's product data structure. Therefore, the team of data engineers has developed a sophisticated algorithm that can detect and adapt to different data structures and formats.

The ingested data is stored in multiple JSON files in an S3 bucket in AWS. The data is then parsed and structured into a relational database (RDS) for easy retrieval and analysis.

Figure 4.1: Retail Match Project Platform

The RDS is designed to store information related to stores, brands, categories, and products in multiple related tables, which provides a highly structured and organized way to store and access the data.

The matching process involves cross-referencing information from retailers using the European Article Number (EAN) to identify identical products. If a product is already in the database, the new product information is added to the respective row with updated information, such as price and product code (an internal retailer code), if available. This process is designed to ensure that the data is up-to-date and accurate, which is essential.

The Retail Match project is a sophisticated data engineering and data science framework designed to handle vast amounts of retail product data from multiple retailers. Through AWS and serverless architecture, the project is efficient and scalable, with a carefully orchestrated data ingestion process and a sophisticated data quality control process to ensure data accuracy and consistency.

Overall, the project demonstrates the importance of data engineering and data science in the retail industry, providing retailers with a comprehensive solution to gain insights and stay ahead of the competition.

4.2 Database

The data for this research is collected from a constantly updated database that reflects the changes made by supermarkets, such as adding new products, discontinuing others and changing product prices. The database contains information in French about products in 4 different Belgium supermarkets, as well as information about their respective stores. It is a data warehousing database with five tables. The schema of the database can be visualized in Figure 4.2.

A data warehouse is a specialized data storage system used by organizations to centralize and manage large volumes of data from various sources, facilitating efficient analysis and reporting. It is structured with two key types of tables: fact tables and dimension tables. Fact tables contain quantitative data and serve as the core of the warehouse, recording measures and metrics like sales revenue or quantities sold. Dimension tables, on the other hand, provide context and description to the data, containing attributes like customer names, product categories, or time periods. Together, fact and dimension tables enable complex queries and multidimensional analysis, empowering businesses to gain valuable insights from their data.

The store table contains information about the supermarket stores, such as their address, zip code, coordinates and city, but this information is not relevant to our purpose of finding similar products.

The most important table is the product_ref, which has all data about the products, including name, description and ingredients list. It only contains information about the product's characteristics and does not include the retailer who is selling the product or its price. This table has two offspring tables: Product Brand and Product Category, which contain information about a product's brand and category, respectively. The product_match table relates the Stores and Product Ref tables. Therefore, we can find data about what products are sold in each supermarket.

In the next section, we will initiate the dataset creation phase. This phase plays a crucial role in the development of a valuable dataset, which is a fundamental step in this project. It ensures that the data we work with is pertinent and well-suited for the task at hand. By gaining a comprehensive understanding of the data, we increase the likelihood of the algorithm delivering the desired outcomes.

4.3 Dataset creation

In order to understand which data can be useful for the dataset we have to go deeper into the database analysis and understand exactly which database fields are relevant. We will explain in detail what is the information in each table field, as well as, discuss and decide what relevant columns we can use in order to create the dataset.

As was explained above, we can ignore all store table data because it is useless to know what are the characteristics of a store in order to identify a similar product.

Regarding brand_table, it contains 5 columns and consists of 4,265 observations. There are no missing cells in the dataset, indicating that all the data points are complete. Additionally, there are no duplicate rows, suggesting that each observation is unique. In spite of that, we will just use the name column, because the other ones (in Figure 4.2) are not relevant to our purpose.

Concerning the categories table we will start by explaining how the categorization works. In the example of Figure 4.3, it is possible to see that the body care category is in the first level of a

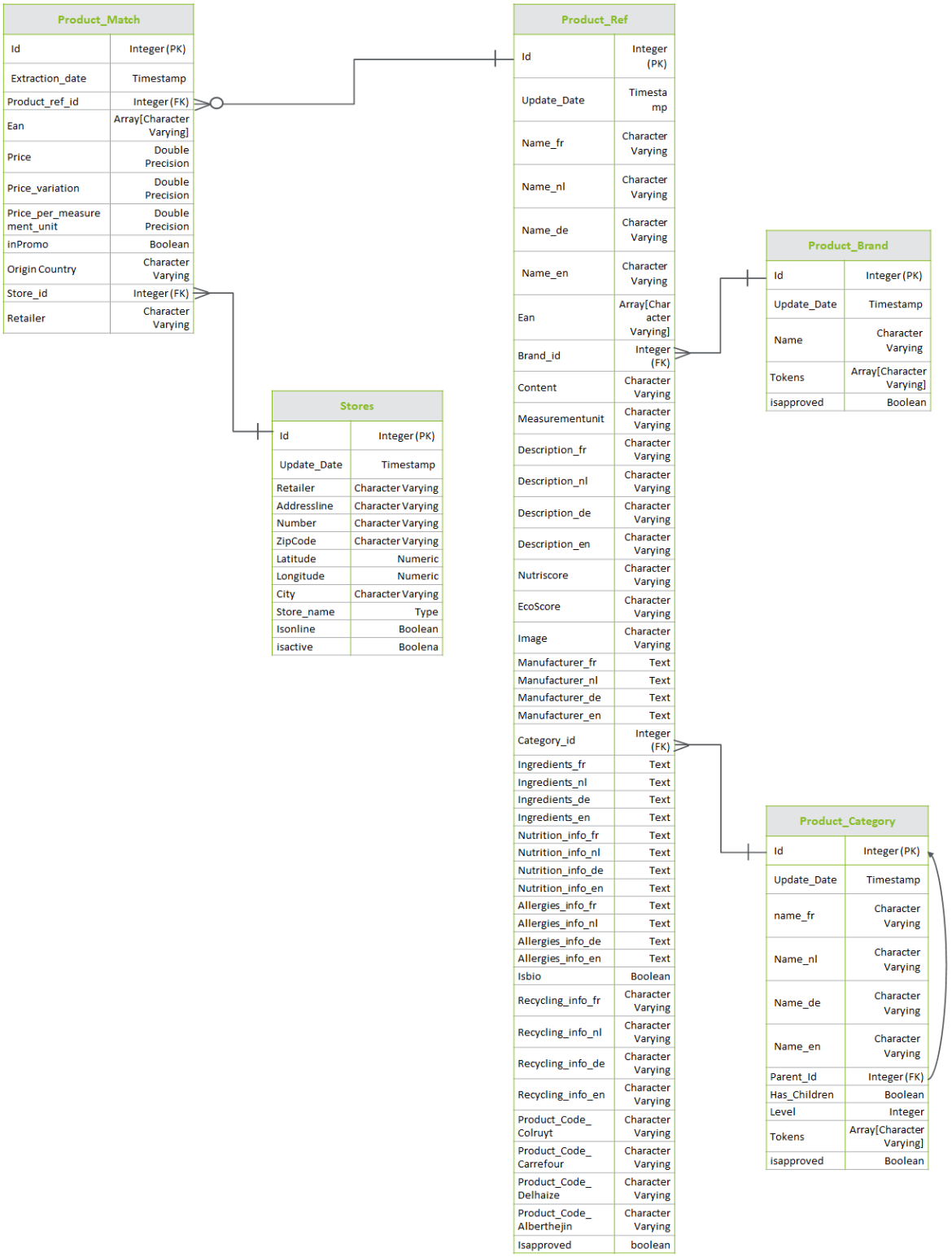


Figure 4.2: Database Schema

tree, and this is the most general category. This category has 3 children called *Make-up*, *Dental Care* and *Skincare*. This means that we have 3 different types of body care products. Each child in the body care category has its own children, and so on, until the end of the tree.

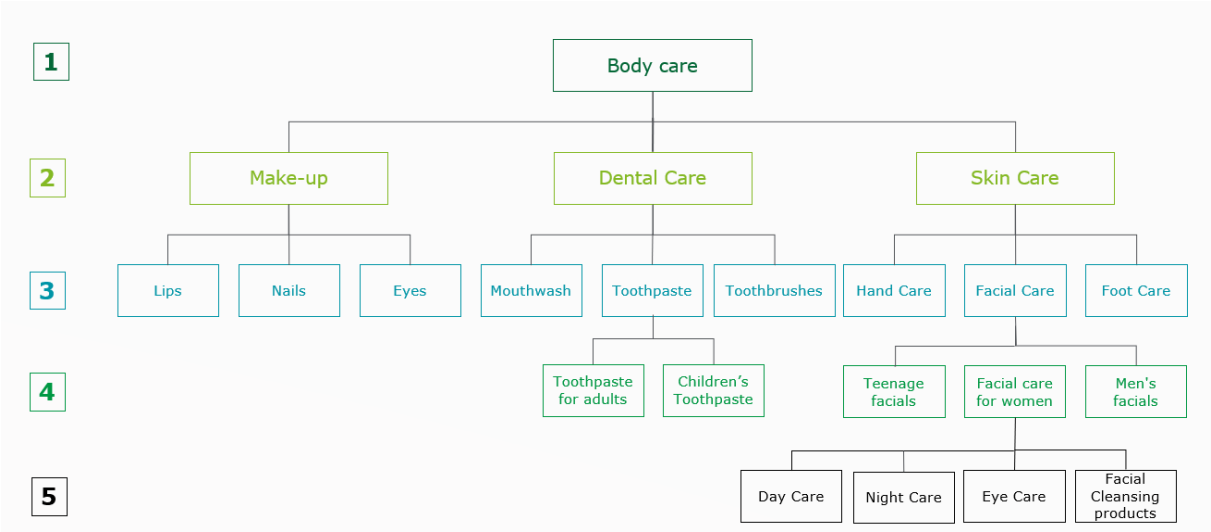


Figure 4.3: Categories Tree Example

To summarize how categorization works, we will set out the important rules that were taken into account about categories when the database was built (by other team members):

- Category level 1 is always the top-level category of a product;
- At the most each product has 5 category levels;
- Each category can have 0 or more children, which means that a product does not need to have all category levels defined;
- 2 first levels of categories are always defined;

In Figure 4.4, it is evident that the highest number of categories exists at level 3 and there are fewer categories at level 5.

The categories table contains information about categories, including the category name (in French, English, German and Dutch), level, a boolean value indicating the presence of child categories, and the id of the parent category in parent_id column. The data in the table is free of duplicates, ensuring unique entries for each category.

The categories table plays a crucial role as it captures the relationships between the products. Consequently, we have created a dataset that comprises all the pertinent information from this table.

The product_ref contains a total of 54 columns, capturing various details of the products. This table contains 59,262 data points, which means that this table offers a significant sample size for robust analysis.

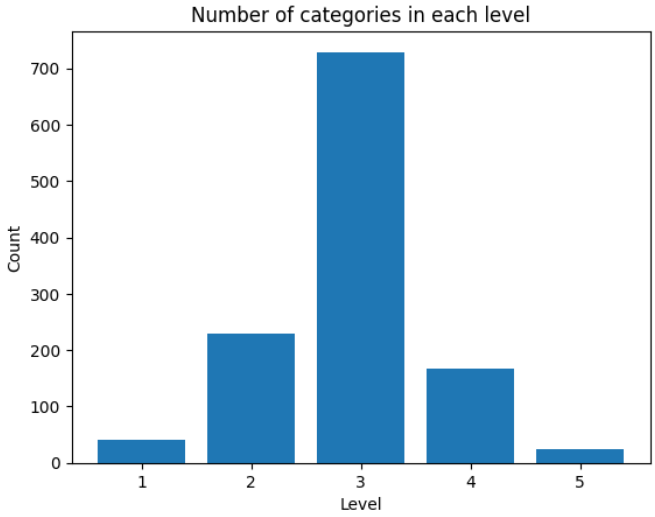


Figure 4.4: Number of Categories in each Level

However, it is worth noting that there are missing values within the dataset, accounting for 1,855,083 missing cells. This represents approximately 58.0% of the total cells in this table.

On a positive note, there are no duplicate rows within the dataset, indicating that each observation is unique. This eliminates any potential biases that could arise from duplicate entries.

We will proceed with analyzing each column of the product_ref table in order to gain a comprehensive understanding of the columns that may prove valuable for our intended purpose.

The product_ref table contains textual data in multiple languages, including French, Dutch, English, and German. However, only the French and Dutch columns are populated with information. Since the French columns have more data we will solely utilize the French textual columns and disregard the non-French columns in order to prevent redundancy.

In Table 4.1 it is possible to see the names and the respective descriptions of 2 products.

	Name	Description
Product 1	Bloc lait noisettes	Goûtez à la puissance du chocolat lait Côte d Or allié au croquant de délicieuses noisettes entières. Un chocolat généreux pour des sensations gourmandes et intenses.
Product 2	chat Adult boeuf-poulet-lég	FRISKIES offre à votre chat les repas qu il aime et dont il a besoin, pour qu il continue à vous surprendre et vous rendre heureux chaque jour.

Table 4.1: Product Name and Description Example

In Figures 4.5 and 4.6 it is possible to see the word cloud to all names and all descriptions of the database before any preprocessing, respectively. An analysis of these word clouds indicates that the most frequently occurring words are often related to measurement units such as *g*, *kg*,

and *ml*. Meanwhile, in the descriptions, common stop words and prepositions are quite prevalent. To conduct meaningful data analysis, it will be essential to remove these types of words in the later stages of processing. This step will help to focus on the more significant and informative content within the text, facilitating more insightful findings.

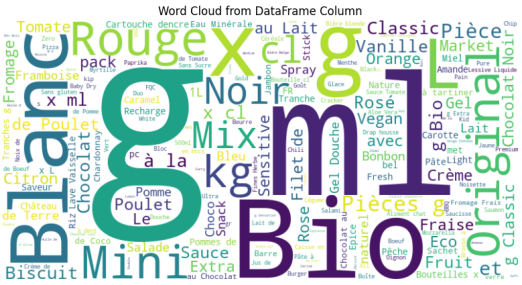


Figure 4.5: Word Cloud for the Name of the Products (before preprocessing)

Figure 4.6: Word Cloud for the Description of the Products (before preprocessing)

The product_ref table also includes a universal identifier for all products known as EAN (European Article Number). The EAN is a standardized barcode system utilized to identify a product, its manufacturer, and the country of origin. EAN barcodes typically consist of 13 digits and are widely employed by retailers and manufacturers globally to facilitate inventory tracking, streamline supply chain management, and expedite the checkout process at the point of sale. It consists of the first one to three digits representing the country of origin, followed by a unique number for the manufacturing company (4, 5, or 6 digits), a variation code for the specific product (3, 4, or 5 digits), and finally, a single digit checksum.

In the EAN column, we have 2607 missing values and no duplicate values. We think that this can be useful because it is a universal number with information about the product.

In this table, we also have information about the end-level category for all products stored in a column called category_id. However, the information related to the category_id for each category level is incorrect and therefore it is unreliable. We have determined that the end-level category information is useful for our purposes, while the information pertaining to the other category levels can be discarded because can lead us to wrong conclusions.

Regarding the quantities of the products, in the product_ref table, we have three columns that provide this information: quantity, content, and measurement_unit. However, upon analysis, we found that the quantity column contains only values of 1 or null, which does not provide

any new information about the products. As it is possible to see in Figure 4.7, we observed that there are only three distinct values in this column, which we believe could still be relevant. Additionally, the content column contains information such as "1.4kg," "1400g," or "20x10g," which directly relates to the product and is likely to be valuable.

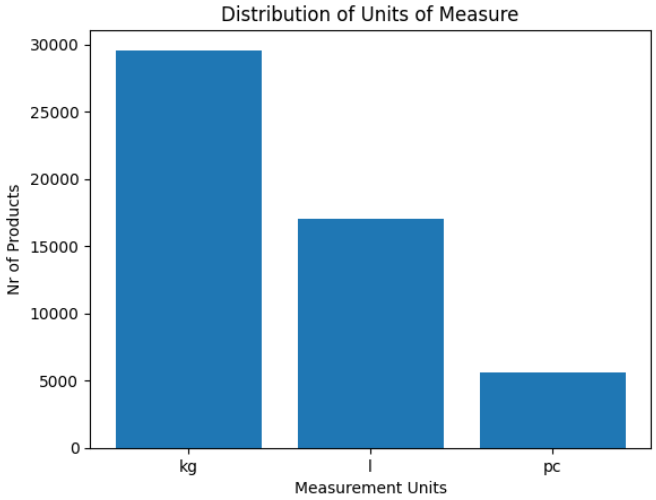


Figure 4.7: Measurements Values Distribution

Regarding nutrition score information it is a rating system used to evaluate the nutritional value of food products. It is typically based on a combination of different factors such as the amount of calories, fats, sugars, fiber, vitamins, and minerals present in a food item. Nutrition scores can be calculated using different methods and algorithms, and they are often displayed on food labels.

The purpose of nutrition scores is to help consumers make informed choices about the foods they eat by providing an easy-to-understand, standardized measure of their nutritional quality. By comparing the nutrition scores of different products, consumers can quickly identify the healthier options and make choices that align with their dietary goals and preferences.

In some European countries, nutrition score rates foods on a scale of A to E, with A being the healthiest and E being the least healthy.

In our dataset, only around 34% of the products have a nutrition score label defined. From the bar plot in the image 4.8 it is possible to see that the most frequent nutrition score in the dataset is the score A with around 6000 products and the least is E with around 3000 products.

In a similar manner, the ecological score label uses a rating or scoring system to evaluate the quality of the product based on certain criteria. The nutrition score evaluates the nutritional value of food products, while the ecological score evaluates the environmental impact of products, based on various factors such as the carbon footprint of a product, the use of natural resources, the impact on biodiversity, and the level of packaging waste generated by the product. It provides consumers with a clear and easy-to-understand rating of the environmental impact of the product, enabling them to make informed choices about the products they buy and their impact on the

planet.

Despite being a powerful rating system, only a small number of products choose to display the ecological score to their customers, and the same is true in our dataset as only 5% of products carry this indication. From the bar chart in the image 4.9 we can see that the distribution scores between A and D are similar, however, there are less than 300 products with the lowest ecological score.

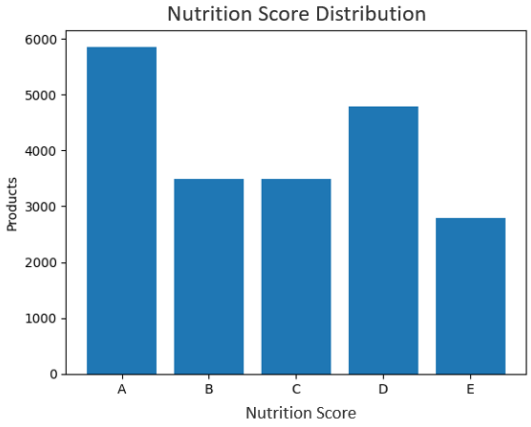


Figure 4.8: Nutrition Score Distribution

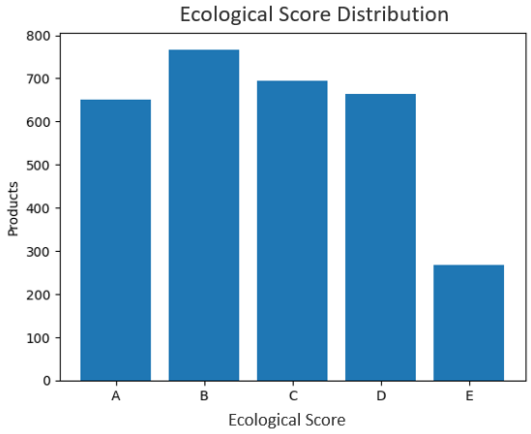


Figure 4.9: Ecological Score Distribution

Since the number of products with an ecological score is relatively small, we have decided to exclude this column from our analysis. However, we have chosen to retain the nutrition score column for further consideration.

We have approximately 40% of non-null values in the nutrition_info column. This column provides information, in a formatted way, such as calories, quantity of sugars, and fats per 100g of product. An example of the format is shown below.

{'100g': [{'label': 'Nutrition', 'description': 'Valeurs nutritionnelles moyennes pour 100g', 'values': {'Energie kj': '2235 kj', 'Energie kcal': '535 kcal', 'Total graisses': '30 g', 'Graisses saturées': '15 g', 'Total glucides': '58 g', 'Sucres': '57 g', 'Fibres alimentaires': '3.1 g', 'Protéines': '6.5 g', 'Sel': '0.17 g'}}], '9g': [{'label': 'Nutrition', 'description': 'Valeurs nutritionnelles moyennes pour 9g', 'values': 'Energie kj': '210 kj', 'Energie kcal': '50 kcal', 'Total graisses': '2.8 g 4%', 'Graisses saturées': '1.4 g 7%', 'Total glucides': '5.5 g 2%', 'Sucres': '5.3 g 6%', 'Fibres alimentaires': '0.3 g', 'Protéines': '0.6 g 1%', 'Sel': '0.02 g 1%'}]}

The information in this column can be useful, although it may not apply to non-food products. Similarly to the nutrition info, the ingredients list can provide valuable information for our analysis, even though approximately 40% of the values in this column are null. The information in this column is organized in descending order of predominance by weight, but it is not presented in a standardized format like the nutrition info. An example of an ingredient list is shown below.

"Ingrédients: Sucre, pâte de cacao, NOISETTES (7,5 %), LAIT écrémé en poudre, beurre de cacao, graisses végétales (palmiste, palme), AMANDES (6 %), lactosérum en poudre (de LAIT), BEURRE concentré, émulsifiants (lécithine de SOJA, lécithine de tournesol), arômes. PEUT CONTENIR BLÉ."

This table also includes information about whether a product is organic or not (in isBio Column), which we consider to be a valuable feature as it can influence a customer's decision to purchase. However, it's important to note that for non-food products, this feature always has a False value. The percentage of True values in the dataset for this column is approximately 6%, around 3500 products.

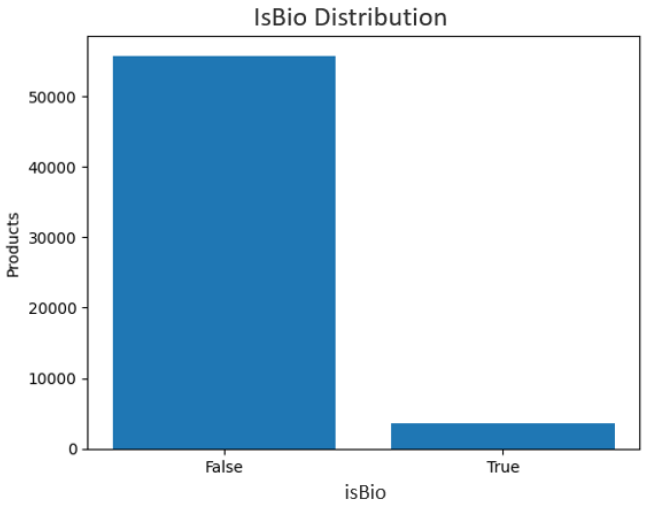


Figure 4.10: isBio Distribution

As previously explained, each product in the reference table contains brand information, which is essential for our analysis. Additionally, the data points in the reference table include an image and a product code specific to each retailer.

Information about the manufacturer can be rejected, also knowing who makes various products does not mean that the products made are similar. *Allergies_info* data can be ignored because all values are null. Lastly, we can discard *Recycling_info_fr* too because 2 similar products can be wrapped up with different materials and consequently have different recycling information.

Finally, all tables explained above have a column named isapproved. This field indicates the product approval status, because when a product comes up in a database needs to be approved, as well as, be in an approved category and in an approved brand.

Last but not least, Product_Match relates Product_ref and Stores tables and it contains information about what products are sold in each store, as well as, the data about the sale (like price, or if a product is in promotion or not).

On the other hand, the information about the price can be useful. Although it does not help

in identifying similar products, because they can have different prices, this feature can be used, for instance, when we want to recommend the cheapest or the most expensive similar product. In the same way, price_per_measurement_unit and origin country can be useful for a similar purpose.

To summarize, we have selected the important columns from the database tables and created a dataset comprising 59,262 observations and 24 features by executing an SQL query.

In the next section, we will delve deeper into the data and perform an exploration to gain a better understanding of its characteristics. This exploration may involve various analyses, such as descriptive statistics and data visualization. By exploring the data more thoroughly, we aim to uncover patterns, trends, and insights that can guide further analysis or decision-making processes.

4.4 Data Exploration

After the dataset is built we are able to analyse the data.

Concerning the categories column, it should be noted that the dataset contains nearly 630 distinct categories divided by 5 levels. 13 categories have only 1 product and 37 categories have less than 5 products. The five most frequently occurring categories, which include cookies, beers, chocolate, a category comprising French products, and a category labeled *others* each have an approximate count of 700 products.

In Figure 4.11, we observe the distribution of products across different categories at an end-level. Analyzing the plot, we can see varying product counts within each category. Some categories have approximately 700 products, while others have fewer than 50.

In Figure 4.12, the distribution of products by category level is illustrated. It's evident that a significant majority of the products belong to level 3 categories. This pattern arises from how the database columns are populated, aligning with the retailer's category tree. Products are rarely categorized at levels such as 4 and 5.

Subsequently, we analyzed the brand's column and we observed that there are 3,832 unique brands.

The 3 most common brands are the private labels from supermarkets where the products were taken. With almost 4,000 products is the retailer 1 private label and then is the retailer 2 private label with around 2,700 products. These 2 brands offer a wide range of products in various categories, including groceries, fresh produce, household and personal care items, electronics, clothing, and accessories. They also provide a range of household and personal care items, such as cleaning supplies, personal hygiene products, and baby care products. Additionally, they offer electronics and appliances, home goods, clothing, and accessories. Retailer 3 offers more than 1,600 products and it is another retailer 1 private label. This brand provides a variety of

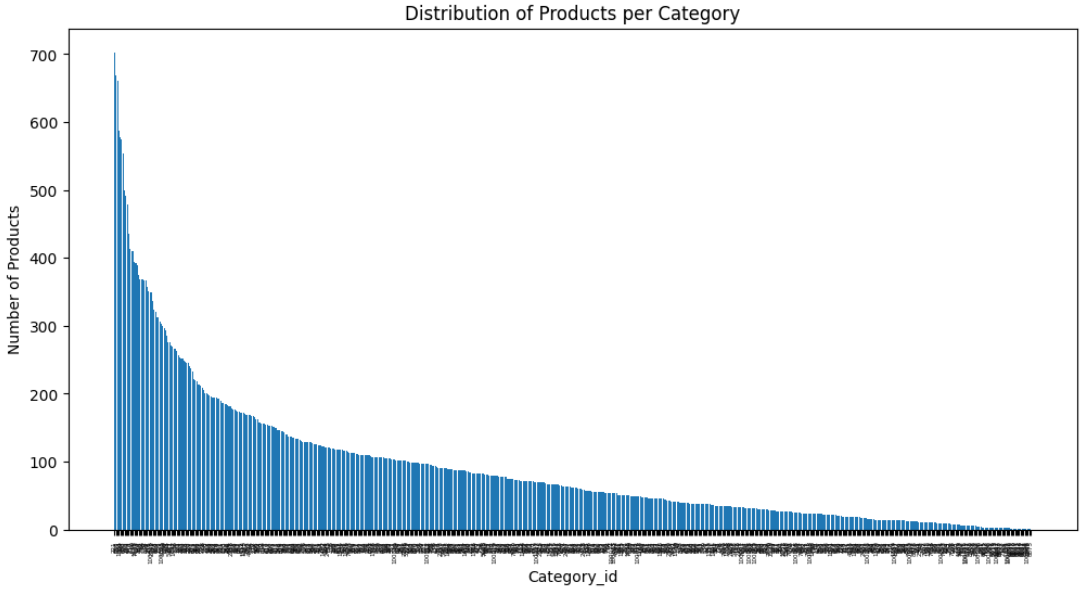


Figure 4.11: Product Distribution by Category

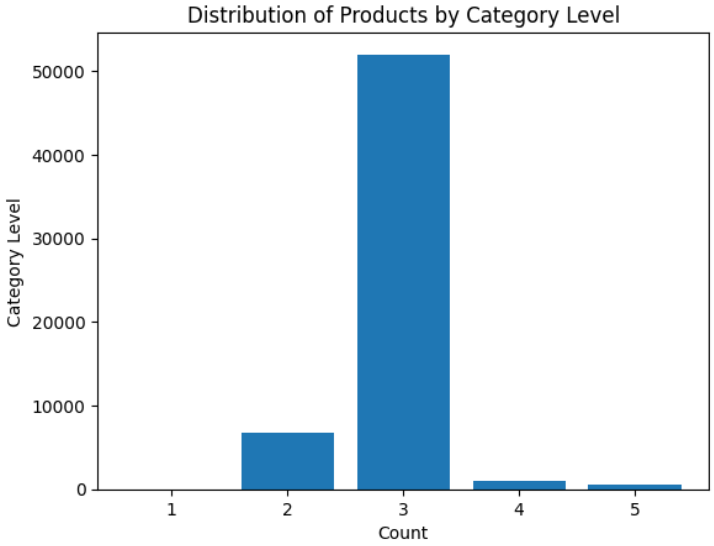


Figure 4.12: Number of Products per End-level category

personal hygiene and beauty products, including items such as shampoo, conditioner, soaps and moisturizing creams, among others.

In general, each brand has on average 14 products, but the median of the products per brand is 2.0 as it is possible to observe in the box plot of Figure 4.13.

Regarding the other correlations in the matrix, they generally show weak or negligible correlations between the variables. This suggests that the variables are not strongly associated with each other in a linear manner.

In the next section, we explain how we prepare the data. It involves cleaning, transforming,

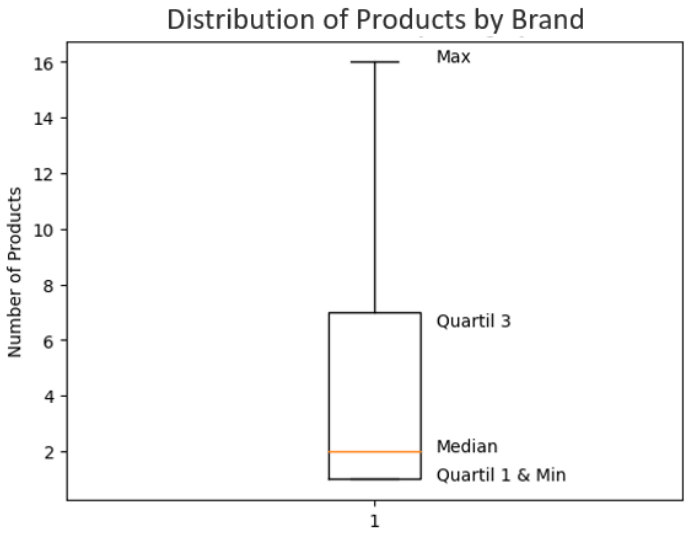


Figure 4.13: Distribution of Products by Brand

and restructuring data to ensure it is accurate, complete, and well-organized for analysis.

4.5 Data Preparation

Once the appropriate columns have been selected for the dataset, the data preparation process may begin. This step can lead to an improvement in data quality, time and resource savings, enhanced data analysis, and mitigation of errors and biases. By undertaking data preparation, the quality and accuracy of the data can be improved, leading to more reliable and meaningful insights. Furthermore, identifying and correcting errors and biases can help ensure that the insights drawn from the data are unbiased and reliable.

Overall, data preparation is an essential step in the data analysis process to ensure accurate, reliable, and well-organized data. Figure 4.14 illustrates these main steps.

The first step in the preprocessing stage involves removing unapproved products from the dataset as their presence could compromise the integrity of the analysis and invalidate the results. After that, we remove the columns containing this information as they are no longer required for the analysis.

Once the unapproved products have been removed, the next step is to perform text data cleaning on the textual variables. This typically involves removing any irrelevant elements such as integers, floats, punctuation marks, and stop words, as well as converting the text to lowercase to standardize the format.

Additionally, to avoid redundancy in the data, we also remove the brand name from the product name and description as other columns already account for brand information. By removing the brand name from the textual variables, we can simplify the data and make it easier

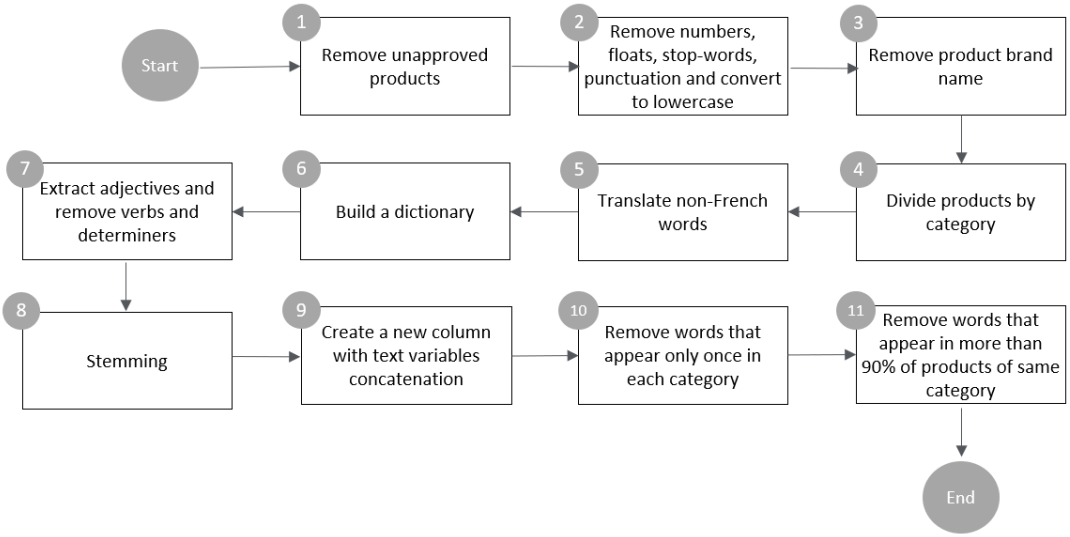


Figure 4.14: Preprocessing Workflow Schema

to analyze and extract insights.

As was explained before, in the dataset, each product is assigned to a category based on its type, and the categories are organized in a category tree structure where the top level represents the most general category and the lowest level represents the most specific category. Each category can have zero or more subcategories, which means a product does not necessarily require all category levels defined. However, all products have at least the first and second category levels defined. In the column "category_id" of each product, only the id of the end-level category is present.

Then, we create 5 new columns in the dataset called *category_level_1_id* , *category_level_2_id*, *category_level_3_id*, *category_level_4_id* and *category_level_5_id*. As the name says, in the respective column will be the id of each category in each level.

After that, we will check the category level of the category_id by consulting the categories dataframe and write it in the respective column in the products dataframe. Subsequently, we will check the parent_id of this category_id and write it in the column to save the category_id level - 1. We will continue this process until we reach the first level category.

Just as a reminder, in Figure 4.12, we saw that the vast majority of products do not belong to category levels 4 and 5. Consequently, the columns *category_level_4_id* and *category_level_5_id* will contain a substantial number of NULL values.

In order to obtain more meaningful results, products are separated according to their categories. Dividing the products by end-level category allows the clustering model to identify patterns and similarities within each category. In order to do that we first identify all category levels for all products in the dataset and choose all category ids without children. After that, we join products by the selected ids.

Although most textual data is in French, some product names and descriptions include English expressions and words. Leaving the text like it is could result in inaccurate or flawed analysis. For instance, if we want to group all apple ciders together, we might come across products labeled as "apple cider" or "pomme cidre" (French for apple cider - note also the distinct spelling), making it challenging to identify relationships or similarities between products. To mitigate this issue and ensure consistency in our data, in step 5, we implemented a translation process where any non-French language expression in the product names and descriptions is translated to French language equivalents using the `deep_translator` Python library.

Products belonging to the same category exhibit resemblances in their product names and descriptions, although not necessarily with the same words. For example, we came across products named "spray toilette" and "spray WC" that are both meant for use in the bathroom, but their names are slightly different. To solve this problem, we created a dictionary-building algorithm to ensure consistency in the language used across products within each category. In order to do that, for each category, we constructed a dictionary of synonyms for each word (step 6). To standardize the language used across products within the same category, we iterate through each word in the product name and description and:

- If the word is already a key in the dictionary, we do not make any changes.

- If the word appears in any of the values of the dictionary, we replace it with the corresponding key.

- Lastly, if the word is neither a key nor a value, in the category's dictionary, we add it as a key with a list of possible synonyms as the value.

We utilize the `Wordnet` library to find synonyms for each new word. This library allows to expand the dictionary's values and ensures that all possible synonyms are considered in the standardization process.

When describing a product, it is important to consider that not all words carry the same weight in conveying its essence. While certain words like adjectives and proper names carry significant importance in providing a clear and accurate description, others like verbs and adverbs might not contribute as much to the overall understanding of the product. For instance, in the description of a perfume, words such as 'floral" or "woody" would be crucial to understanding the scent's characteristics, whereas "sprayed" or "lightly" would have less impact. Therefore, when analyzing product descriptions, it is important to carefully choose and prioritize the words used in order to effectively convey the most relevant information about the product. To achieve this, we use the *Spacy* library that enables to determine the morphological tag of each word in the text. This provides information regarding the word's grammatical category, such as whether it is a noun, verb, adjective, etc. We then add a new column to the dataset containing all the adjectives identified in name and description. Furthermore, we clean the text variables by removing verbs, auxiliary verbs, adverbs, and other unnecessary types of words from the name and product description.

After standardizing the names and descriptions of each product, we use a French stemmer to reduce words to their base or root form, in step 8.

We then create a concatenated column, called *text_info*, consisting of the product name twice, followed by the description and adjectives. The purpose of duplicating the product name is to emphasize its importance in text analysis because we observed that the name often provides a more distinct and informative description of the product, and by including the name twice, we increase its weight in the analysis. By including the adjectives in the concatenated column, we can assign more weight to these descriptive words.

To further clean our data, we remove words that appear only once in each category and words that appear in over 90% of the products. This is done to reduce the impact of infrequent and very common words on the clustering algorithm.

Among all dataset columns, *text_info* stands out as the most informative for identifying similar products. With the completion of the text preprocessing phase, we are now fully prepared to utilize this column in the upcoming chapter, focusing on applying models to effectively group similar products together.

4.6 Summary

In this Chapter, we described the project and explored the available data. Following that, we explained the preprocessing phase, which consisted of removing punctuation and unnecessary words such as stopwords and brand names from all products in the dataset, both from their names and descriptions. Afterward, we categorized the products by end-level category and translated the text variables to French. We applied a methodology to ensure consistency in words with similar meanings, identified and removed verbs and determiners, and then applied stemming. Finally, we concatenated the name, description, and adjectives into a single column, called *text_info*, removing words that appear only once or those that appear in more than 90% of the products within the same category.

After successfully interpreting and preparing the data, the next Chapter focuses on the modeling phase. The primary objective of this phase is to identify and implement a machine-learning model capable of creating clusters within each category of the dataset based on the *text_info* column (column with the concatenated text).

Chapter 5

Product Representation and Clustering of Similar Products

Up to this point, we have comprehensively analyzed the data and implemented our methodology to clean the text effectively. With this preparation, we are now ready to develop a clustering methodology for categorizing the products. As was explained in the last chapter we will utilize the *text_info* column to find an efficient way to join similar products together. To achieve this, we will apply some of the methods reported in the literature.

We begin by presenting the hardware and software platform used for the experiments in Section 5.1 and Section 5.2, respectively. Following that, in Section 5.3, we describe the baseline model. Subsequently, in Section 5.4, we detail the modeling tests conducted, divided into two parts, each focusing on 20 specific categories. The first part focuses on determining the optimal combination of embeddings and the best method to determine the optimal number of clusters. The second part involves finding the most suitable clustering algorithm. After the modeling tests, we describe the process of saving the clustering data in the database. Subsequently, we explore the application of the methodology to assign unseen products to the appropriate clusters. Additionally, we provide detailed insights into implementing the chosen methodology across all categories within the dataset.

5.1 Hardware

The hardware components used in this research played a crucial role in ensuring efficient execution and processing capabilities. The specifications of the computer we used were as follows:

1. **System Type**: The research was conducted on a Windows 64-bit operating system with x64-based processor architecture. This system type provided compatibility and support for various software tools and code libraries.

2. **Processor**: The research utilized the 11th Gen Intel(R) Core(TM) i7-1185G7 processor,

operating at a frequency of 3.00GHz with a base frequency of 1.80 GHz. This high-performance processor provided substantial computing power for complex calculations and data processing.

3. **Installed RAM**: 32.0 GB RAM was installed, with 31.7 GB available—the ample memory capacity allowed for seamless handling of large datasets and computational tasks.

We also need to execute the code on AWS; for that purpose, we utilize an Amazon EC2 instance of type c5.xlarge. It is important to note that the hardware used in this research did not include a dedicated GPU.

5.2 Tools

The utilization of a variety of software tools was pivotal in facilitating data extraction, analysis, manipulation and modeling. The subsequent set of software tools were employed:

- **Visual Studio Code**: Visual Studio Code was the integrated development environment (IDE) for writing and editing Python code.
- **SQL**: SQL (Structured Query Language) was employed to query and retrieve data from the database.
- **Jupyter Notebooks**: Jupyter Notebooks were the primary environment for implementing algorithms, allowing for interactive code development and convenient result presentation.
- **GraphQL**: GraphQL is a query language and runtime for APIs that allows clients to request precisely the data they need, promoting efficiency and flexibility in data retrieval.
- **Python**: Python was the programming language of choice for its suitability in ML and data projects. Python's versatility, rich ecosystem, and extensive libraries make it an excellent choice as a programming language. The availability of numerous libraries in Python allows developers to leverage existing solutions and tools, saving time and effort in building new functionalities from scratch.
- **Amazon Web Service (AWS)**: Amazon Web Services (AWS) is a highly scalable cloud computing platform offered by Amazon. It provides a wide array of computing services, including computing power, storage options, and databases, allowing businesses and individuals to run applications and store data without the need to invest in physical hardware. AWS offers a pay-as-you-go pricing model, enabling cost efficiency by scaling resources based on demand. It supports various programming languages and frameworks, making it flexible and accessible for developers. Additionally, AWS provides a vast ecosystem of tools and services for artificial intelligence, machine learning, the Internet of Things (IoT), and more, empowering users to innovate and deploy advanced technologies in a seamless and secure manner.

- **AWS Sagemaker**: AWS SageMaker is a fully managed machine learning service offered by AWS that simplifies the process of building, training, and deploying machine learning models at scale. It provides a comprehensive set of tools and infrastructure to streamline the entire machine learning workflow, from data preprocessing to model deployment, making it easier for developers and data scientists to create and deploy machine learning models in production.

Some of the Python libraries used for the implementation of this solution are:

- **Psycopg2**: The psycopg2 library facilitated the establishment of a connection to the PgSql database and enabled efficient retrieval and storage of data. Its robust features and compatibility with PostgreSQL ensured seamless database interactions.
- **Pandas**: Pandas library was crucial in data manipulation and analysis. Its powerful data structures, such as data frames, provided flexible and efficient methods for cleaning, preprocessing, and transforming the research data.
- **Spacy**: Spacy is a popular and efficient natural language processing library for Python, known for its robust pre-trained models and easy-to-use API, making it a valuable tool for a wide range of text analysis tasks.
- **deep_translator**: Deep_translator is a Python library that facilitates an easy translation of text between multiple languages using various translation services and APIs. It simplifies the process of incorporating language translation capabilities into applications and workflows, making it a convenient tool for multilingual text processing tasks.
- **Numpy**: The Numpy library offered essential numerical computing capabilities. Its array processing functionalities and optimized mathematical operations enhanced the efficiency of calculations and transformations on large datasets.
- **Matplotlib**: The Matplotlib library proved instrumental in visualizing the research findings. It enabled the creation of a wide range of visual representations, including plots, charts, and graphs, facilitating data interpretation and presentation.
- **Seaborn**: The Seaborn library complemented Matplotlib by providing advanced data visualization capabilities and statistical plotting. Its integration with pandas allowed for generating visually appealing and informative visualizations to gain deeper insights from the data.
- **OpenCV(cv2)**: Cv2, short for OpenCV, is a widely-used computer vision library in Python that provides a comprehensive set of tools and functions for image and video processing. It is favored for its ability to handle tasks like image manipulation, object detection, and feature extraction, making it a cornerstone for computer vision applications and research.

- **Regular Expression (RE)**: This library, commonly known as REGEX, was employed for advanced text manipulation and extraction. It enabled identifying and extracting specific patterns or information from textual data, contributing to the preprocessing and analysis of text-based datasets.
- **Nltk (Natural Language Toolkit)**: The Nltk library proved invaluable in treating and preprocessing textual data. Its suite of tools and functions for tokenization, stemming, and other text preprocessing tasks enhanced the quality and relevance of the text-based research data.
- **Sklearn (Scikit-learn)**: The Sklearn library provided many ML algorithms, preprocessing techniques, and evaluation metrics.
- **Scipy - Cluster.Hierarchy Module**: It is a module within the SciPy library for Python that offers functions for hierarchical clustering and linkage analysis, allowing users to organize data points into a hierarchical structure based on their similarity or distance metrics. It's a valuable tool for tasks like cluster analysis and dendrogram visualization in various scientific and data analysis applications.
- **kneed**: The kneed library in Python is a useful tool for automatically detecting the "knee" or elbow point in a data curve, which helps users determine the optimal number of clusters or components in unsupervised machine learning tasks like clustering or dimensionality reduction. It simplifies the process of selecting the appropriate number of groups in data analysis by identifying the point where additional clusters or components no longer significantly improve the model's performance.

The software tools employed facilitate the processes of data extraction, analysis, and storage. A diverse array of code libraries enhanced the research by offering specialized features for data manipulation, analysis, and machine learning tasks. Together, these hardware and software resources served as the cornerstone for effectively achieving the research objectives.

5.3 Baseline Model

EAN stands for "European Article Number". It is a standardized barcode system used worldwide for identifying retail products, including consumer goods, books, and other items available for sale. EAN barcodes consist of a series of numbers and are typically displayed as a barcode on product packaging. They help automate and streamline the inventory and checkout processes in retail stores, making it easier to track products and manage sales. EANs are crucial for efficient supply chain management and accurate pricing at the point of sale.

EAN is constituted by a unique 13-digit numeric code that includes a country code, manufacturer identifier, and product identifier, all used to uniquely identify products for retail and inventory purposes.

We consider the baseline model to be an algorithm that groups products based on the product identifier part of the EAN, because this number can be more similar when the products are more similar.

5.4 Modeling Tests

After preparing the data, we are ready to commence the modeling phase. Our primary objective during this phase is to assign a unique cluster code to each product based on the end-level category and the *text_info* column.

In Chapter 3, we extensively examined the primary approaches for addressing the challenge of identifying similar products: product similarity and product clustering. Following careful consideration, we concluded that clustering is the optimal methodology to tackle our problem, primarily due to the substantial volume of data we need to manage.

By adopting clustering, we can efficiently handle the large dataset by storing centroid information, thereby significantly reducing the data storage burden. This is in contrast to the alternative approach of product similarity, where storing information on all products would be necessary, resulting in a considerably heavier storage requirement. Clustering, therefore, emerges as the more efficient and practical choice for our objectives.

Given our conclusion that the *text_info* column uniquely describes the data, we recognize the importance of utilizing embeddings. As we've chosen a clustering approach, our subsequent steps entail identifying the optimal number of clusters for each category and selecting the most suitable clustering algorithm.

The process of determining the optimal model for forming clusters was primarily inspected, explored, and tested locally using Jupyter Notebooks, with reliance on the hardware specified in Section 5.1. Once we identified a suitable model for cluster creation, we moved to a more powerful cloud infrastructure to generate clusters for all end-level categories: AWS SageMaker.

We chose to leverage AWS SageMaker due to the limited computational capabilities of the local hardware, which were insufficient for processing the large volume of data required for our algorithm. The extensive scalability and robust cloud infrastructure provided by AWS SageMaker allowed us to effortlessly handle the computational demands of our clustering process.

Now that we have determined our approach to the problem and defined how we will utilize the hardware, we can commence testing to ascertain the best model. The initial stage involves transforming textual data into a numerical format suitable for machine learning algorithms. Various vectorization techniques are commonly employed for this purpose, including Term Frequency-Inverse Document Frequency (TF-IDF), Bag-of-Words (BOW), Word2Vec, and the most recent one, chat GPT-2 embeddings. These techniques help in converting text into a format that can be easily processed by machine learning algorithms.

The next step is to determine the appropriate number of clusters to use for a category. In order to do that we tested 5 metrics: Elbow Method, Silhouette Method, Calinski Harabasz Score, AIC Score, and BIC Score.

Then, we can apply a clustering algorithm. The result is a set of clusters where the data points within each cluster are more similar to each other than to those in other clusters. In order to do that we test K-means, DBSCAN, Hierarchical Clustering, Gaussian, Bayesian Gaussian and a combination between DBSCAN and K-Means. We test this combination because DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is particularly effective in outlier detection. We take advantage of this DBSCAN strength to detect and remove outliers from our dataset. After removing the outliers using DBSCAN, we then utilize the K-means algorithm to perform clustering on the remaining data points.

To summarize what has been explained so far, basically, to create clusters, we need to go through three stages: choosing the best method for embeddings, selecting the most suitable metric to determine the optimal number of clusters, and finally choosing the best clustering algorithm. The scheme in Figure 5.1 outlines the entire modeling process.

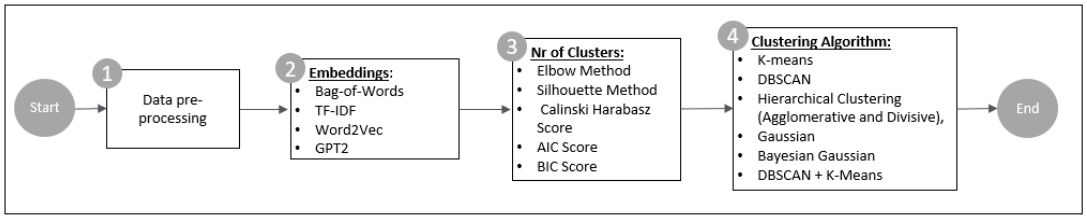


Figure 5.1: Modelling Schema

To determine the optimal combination of embeddings, the algorithm for selecting the best number of clusters, and the most suitable clustering algorithm, we consider all available options. However, given the computational expense and time constraints, it's prudent to streamline the approach.

To efficiently address our problem, we have structured a two-phase process. In the first phase, we focus on determining the optimal combination of embeddings and the algorithm for selecting the best number of clusters. In the subsequent phase, we use the insights from the first phase to choose the best clustering algorithm, ensuring an informed decision based on the previously determined optimal combination. Furthermore, instead of considering all categories in our dataset, we will concentrate on 20 specific categories during each phase of the process because there are a lot of categories.

In the first phase, our focus is on determining the best combination of embedding algorithms and the choice of the best K algorithm for our task. We evaluate the embedding algorithms and K algorithms mentioned above and compare their outputs with the real K value (the correct value). Our objective is to find an approach that produces results that are close to the correct value of K. By doing this, we can identify the combination that performs optimally and is most aligned with the actual data characteristics.

To determine the optimal K, it is necessary to establish a search range. The lower limit we have set is 2, while the upper limit depends on the number of products within a category. If a category contains fewer than 15 products, the maximum value is capped at 4. However, if the result of dividing the number of products by 4 is less than 20, then the maximum is set to that calculated value; otherwise, it remains at 20. We believe that 20 serves as a reasonable maximum value because, based on our preliminary assessment of the products within a category, it does not appear to make sense for a category to have more than 20 clusters. Additionally, we require an appropriate clustering algorithm for our process. As discussed in Chapter 3 and outlined in paper [37], employing k-means along with elbow and silhouette methods is a recommended approach to determine the optimal number of clusters (K).

After several tests and taking into account all the methods explained before, we conclude that TF-IDF + elbow is the best method to determine the optimal K.

However, there are instances where the elbow method may not provide a clear elbow point. In such cases, the silhouette method can be used to assess the quality of clustering, since it is the second best method. The silhouette method involves calculating the silhouette score for each point in the dataset, which is a measure of how well the point fits into its assigned cluster compared to other clusters. The average silhouette score across all data points is then calculated for each number of clusters, and the number of clusters with the highest average score is the optimal choice. If neither the elbow method nor silhouette scores provide a valid value for determining the optimal number of clusters, we abstain from clustering for this category due to the infrequent occurrence of this situation and none of the other methods tested seem to be reliable.

Overall, we have selected TF-IDF as our embedding method. For determining the optimal number of clusters, we have found that the Elbow method is the most effective. However, if the Elbow method does not provide a valid value, we employ the silhouette method to determine the appropriate number of clusters.

Once we determine the best combination in the first phase, we progress to the second phase. During this stage, we integrate the chosen approach from the first phase with all the previously mentioned clustering algorithms. These algorithms are then applied to the output of TF-IDF, and we assess the outcomes for each. This allows us to assess the performance of each clustering algorithm when paired with the optimal combination of embeddings and K-choice algorithms. It is important to note that, given the absence of a definitive ground truth for product division, we approximated what we believed to be the most appropriate division of products.

Upon completing the second phase of our analysis, we have determined that the K-means clustering algorithm emerges as the most effective choice.

Consequently, the optimal modeling approach is the combination of TF-IDF followed by the elbow method (or silhouette method, if the elbow method does not provide output) and finalized by the application of K-means clustering.

Additionally, we explored the integration of PCA prior to implementing K-means; however, our findings revealed that this approach did not yield as favorable outcomes, likely due to the non-linearity and sparsity of the data.

To store the cluster code information in the database, we proceed by adding a new column to the *product_ref* table, which we name *Cluster_Id*. This column will hold the cluster identification for each product. Additionally, we create a new table named *Product_Cluster*, consisting of three columns: *id*, *code*, and *name_fr* as described in Figure 5.2.

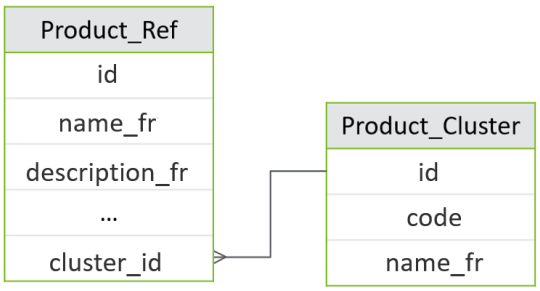


Figure 5.2: Database with Cluster Fields

The cluster name is dynamically generated by extracting the most common words in each cluster. We first identify the two most frequent words in the cluster. If these two words are present in more than 50% of the products, they are selected as the name of the cluster, otherwise, we choose the three most common words from the product text to serve as the cluster name. While this naming convention may not be perfect, it provides adequate insight into the cluster's content. Although not of crucial importance for our current purpose, we believe it could prove useful in the future.

Meanwhile, the code of the cluster is composed by concatenating the end-level category id with the cluster number (a number ranging from 1 to the total number of clusters.). This approach ensures the uniqueness of each cluster code.

Up to this point, we have outlined a methodology for clustering similar products and discussed how to store the data in the database. However, since the database is continuously updated, it is essential to understand how the algorithm performs when predicting the cluster of an unseen product.

To achieve this understanding, we initiate the process by partitioning products from 20 categories into an 80% training set and a 20% testing set. It is worth noting that we apply this approach selectively to specific categories due to the potential computational time-consuming nature of performing this operation across all categories.

To determine whether an unseen product belongs to the correct cluster, we have two different approaches. The first method involves saving the centroids after the training phase and, during testing, simply identifying the closest centroid to determine the associated cluster. Alternatively, we can persist the model post-training and employ the 'predict' method to get the optimal

cluster. Both options are equally effective, but the second method becomes impractical when dealing with a large number of categories. Hence, we opt to proceed with the first approach. It is important to note that before implementing either of these methods, it is necessary to apply the text preprocessing steps outlined above to the testing set.

Due to the unsatisfactory results of this test, we have decided not to implement this idea for clustering new products that are added to the database. Moreover, it's worth noting that this technique has a notable limitation: it solely assigns new products to a new cluster without the capability to create a new cluster when required.

After running the algorithm using AWS Sagemaker, for all end-level categories, we validate and manually correct the clusters.

Comparing our approach with those discussed in Chapter 3, we identified similarities with the methods employed by Cherednichenko et al. [26] and Mathivanan et al. [37]. This approaches also employed a clustering approach to address the problem, with the distinction that the latter did not utilize PCA. We conducted experiments comparing these two approaches and found that utilizing PCA is not optimal for our specific use case. Shrivastava and Sisodia [52], Chandra et al. [25], Siswanto, Tjong, and Saputra [53], as well as Goswami and Tilottama [24], explored different similarity measures to evaluate product similarities. However, given the substantial volume of data in our scenario, we determined that this is not the most suitable approach. Tantyo and Mauritsius [56] utilized word2Vec for product characterization and KNN for identifying similar products. We also experimented with word2Vec, but based on our dataset, we concluded that TF-IDF is a superior method for characterizing the products. Finally, Adak and Uçar [13] utilized a fuzzy approach, however, we did not conduct experiments to evaluate its efficacy for our specific case, primarily because this method is not widely adopted or tested in our domain.

5.5 Summary

This chapter provides a comprehensive overview of the hardware and software tools employed in the study, along with an in-depth explanation of the modeling process. The hardware specifications, including a high-performance processor and ample RAM capacity, are highlighted. Various software tools and Python libraries are detailed for data extraction, analysis, and machine learning tasks.

The modeling process involves transforming textual data into numerical formats with TF-IDF, determining optimal cluster numbers through methods like the elbow and silhouette methods, and applying clustering algorithms, with K-means emerging as the most effective choice. We indicate how the cluster information is stored in the database. Finally, we investigate how this algorithm performs when predicting the cluster of an unseen product and compare our approach with those mentioned in Chapter 3.

Overall, this chapter serves as a foundational framework for the research's data analysis and

modeling efforts.

Chapter 6

Results

In this section, we present the findings of our study, organized into six subsections. The first subsection details the results of the data preparation. Subsequently, in Section 6.2, we elaborate on the results obtained from the methodology chosen for determining the optimal embedding method and identifying the best K value. Following that, in Section 6.3, we discuss the selection of the most effective clustering algorithm. In Section 6.4, we present the application of our method to two specific categories. Finally, in Section 6.6, we conduct a comparative analysis of our method against the baseline approach.

6.1 Data Preparation Results

After removing the unapproved products, the dataset now has 58,413 products. All products have a name, but about 20% of the dataset has no information in the description column.

Figure 6.5 displays the different stages of the word cloud of the 'Puddings' category, illustrating the preprocessing methods outlined in Section 4.5. Figure 6.1 presents the initial word cloud for 'Puddings' before any preprocessing steps.

In Figure 6.2, we can observe the refined word cloud after the elimination of numbers, stop words, punctuation, product brands, and conversion to lowercase. Notably, Figure 6.1 contained words like *la*, *le*, *un*, *au*, *à*, *x*, *g*, and other irrelevant terms, all of which have been effectively removed.

After we translated non-french words and removing synonyms, verbs and determinants the word cloud result is present in Figure 6.3. Initially, it may appear that words like 'pudding' and 'dessert' are not in French; however, they indeed are.

Lastly, Figure 6.4 displays the final word cloud with stemmed words after completing all preprocessing.

Figure 6.1: Before any Preprocessing

Figure 6.2: After removing numbers, stop words, punctuation, product brands, and converting to lowercase

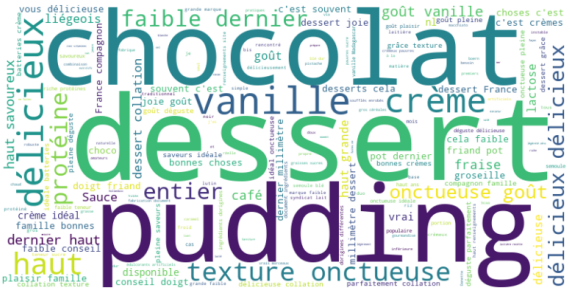


Figure 6.3: After Translating Non-French Words and Removing Synonyms, Verbs and Determinants

Figure 6.4: After Completing Preprocessing

Figure 6.5: Word Cloud of the Pudding Category in Different Preprocessing Stages

6.2 Exploring the best combination between Embeddings + Algorithm to find the best K

After completing the preprocessing phase, our objective is to initially determine the optimal combination of embedding algorithms and find the most effective method for identifying the optimal number of clusters. Subsequently, we aim to identify the best clustering algorithm.

To address the first step, we attempt to understand which combination of algorithms yields the best K. For that, we create the table in Figure 6.6. In this table, we can see the recommended number of clusters for each combination of embeddings and algorithms to find the best K, as well as the optimal K for each category.

The entries shaded in green represent values that are closest to the optimal K (the ideal number of clusters) and may include multiple values when there are multiple equally suitable options with the same proximity to the optimal outcome. Imagine that the optimal K is 5 and combination A gives the value 6, combination B gives the value 7 and combination C gives the value 4. Algorithms A and C are selected as the best ones, as the distance is 1 unit apart from the optimal. 'X' means that it was not possible to obtain a valid K.

Category		Elbow	Silhouette	BIC	AIC	Calinsky	Correct K
512	BOW	3	5	2	2	2	7
	TF-IDF	8	8	2	2	3	
	W2Vec	3	20	2	2	2	
	GPT2	10	11	2	2	2	
586	BOW	4	6	2	2	2	4
	TF-IDF	X	4	2	2	2	
	W2Vec	5	15	2	2	2	
	GPT2	11	14	2	2	2	
582	BOW	5	20	2	2	2	6
	TF-IDF	X	X	2	2	2	
	W2Vec	4	4	2	2	2	
	GPT2	11	9	2	2	2	
590	BOW	15	19	2	2	2	2
	TF-IDF	X	3	2	2	2	
	W2Vec	15	15	2	2	2	
	GPT2	9	8	2	2	2	
68	BOW	4	6	2	2	2	3
	TF-IDF	3	9	2	2	2	
	W2Vec	4	4	2	2	2	
	GPT2	5	4	2	2	2	
405	BOW	4	7	2	2	2	3
	TF-IDF	X	X	2	2	2	
	W2Vec	3	5	2	2	2	
	GPT2	5	5	2	2	2	
428	BOW	11	5	2	2	2	9
	TF-IDF	10	6	2	2	2	
	W2Vec	7	7	2	2	2	
	GPT2	7	8	2	2	2	
435	BOW	11	4	2	2	2	5
	TF-IDF	4	6	2	2	2	
	W2Vec	2	2	2	2	2	
	GPT2	7	2	2	2	2	
441	BOW	8	3	2	2	2	3
	TF-IDF	5	4	5	2	2	
	W2Vec	2	5	2	2	2	
	GPT2	2	5	2	2	2	
450	BOW	6	7	2	2	2	5
	TF-IDF	5	5	2	2	2	
	W2Vec	8	5	2	2	2	
	GPT2	5	8	2	2	2	

Category		Elbow	Silhouette	BIC	AIC	Calinsky	Correct K
456	BOW	10	4	2	2	2	6
	TF-IDF	8	3	2	2	2	
	W2Vec	3	5	2	2	2	
	GPT2	11	15	2	2	2	
511	BOW	4	5	2	2	2	5
	TF-IDF	5	X	9	2	2	
	W2Vec	6	20	2	2	2	
	GPT2	7	7	2	2	2	
513	BOW	X	4	2	2	2	5
	TF-IDF	5	X	2	2	2	
	W2Vec	4	2	2	2	2	
	GPT2	2	2	2	2	2	
522	BOW	5	5	2	2	2	7
	TF-IDF	4	6	4	2	2	
	W2Vec	17	9	2	2	2	
	GPT2	3	9	2	2	2	
524	BOW	10	6	2	2	2	3
	TF-IDF	10	15	2	2	2	
	W2Vec	9	11	2	2	2	
	GPT2	7	8	2	2	2	
527	BOW	5	3	2	2	2	4
	TF-IDF	4	4	2	2	2	
	W2Vec	5	3	2	2	2	
	GPT2	3	6	2	2	2	
551	BOW	12	4	2	2	2	3
	TF-IDF	5	13	8	2	2	
	W2Vec	20	8	2	2	2	
	GPT2	8	20	2	2	2	
659	BOW	4	7	2	2	2	6
	TF-IDF	6	4	2	2	2	
	W2Vec	3	2	2	2	2	
	GPT2	15	15	2	2	2	
259	BOW	9	6	2	2	2	4
	TF-IDF	5	5	2	2	2	
	W2Vec	3	6	2	2	2	
	GPT2	6	10	2	2	2	
279	BOW	9	8	2	2	2	4
	TF-IDF	3	7	5	2	2	
	W2Vec	15	18	2	2	2	
	GPT2	15	17	2	2	2	

Figure 6.6: Optimal Value of K for each Algorithmic Combination.

For example, using BOW and the Elbow Method for category 512, the indicated optimal K is 3. However, the correct K should be 7, so it is significantly far from the correct value. On the other hand, using the Elbow or Silhouette method with TF-IDF gives a value of 8, which is very close to the correct value. Therefore, we designate these two combinations as the most suitable options, highlighting them in green shading for clarity.

The initial observation upon inspecting the table in Figure 6.6 is that the optimal K value, as

determined by any embedding algorithm combined with Calinsky and AIC, consistently remains at 2. Moreover, it is important to highlight that nearly all combinations utilizing BIC also converge to an optimal K value of 2.

It becomes evident that this is erroneous, as the minimum K value is consistently chosen. Therefore, we should consider combinations that exclude BIC, Calinsky and AIC algorithms.

Table 6.1 presents the frequency of occurrences for each combination of Elbow and Silhouette being identified as the best choice.

	Elbow	**Silhouette**
BOW	3	1
TF-IDF	11	9
W2Vec	3	1
GPT2	1	0

Table 6.1: Frequency of Each Combination Outperforming Others across the 20 Study Categories

Among the embedding methods, TF-IDF outperforms the others for both algorithms, with 11 times for Elbow and 9 times for Silhouette. Bag of Words and Word2Vec perform similarly, each being the best combination of 3 categories with Elbow and once for Silhouette. GPT-2, on the other hand, is the least effective embedding method, being the best only for 1 category for Elbow and not performing the best for Silhouette in any instance. Overall, this analysis suggests that TF-IDF + Elbow Method is the most robust combination among the options considered, followed by TF-IDF + Silhouette Score.

Given its consistent performance in approximating the optimal value of k (whenever it yields an output), the TF-IDF + Elbow method emerges as our primary choice for determining the ideal k for the categories. In cases where the elbow method does not yield an output, we resort to the next most reliable option – the silhouette method. In Figure 6.7 it is possible to see the distribution of the number of clusters per category (considering all end-level categories) using that methodology.

6.3 Exploring the best clustering algorithm

Once we have established the optimal number of clusters using the chosen method (primarily the elbow method, with the silhouette method as a backup), our focus shifts to identifying the most suitable clustering algorithm.

Due to the substantial time and resource demands associated with comprehensively describing the outcomes of clustering algorithms across numerous categories, we have chosen to focus our discussion solely on one category. Please consider that the conclusions presented in this section have been drawn from our analysis across various categories.

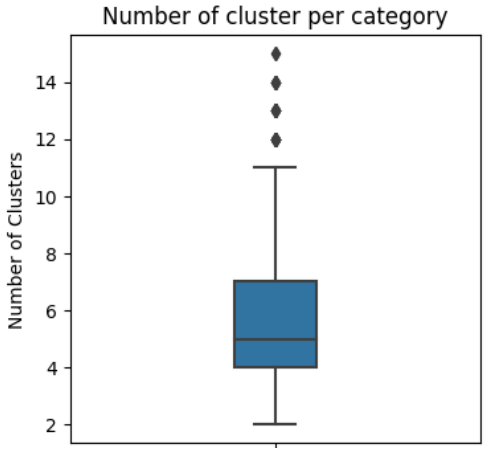


Figure 6.7: Boxplot with Number of Clusters per Category

We chose the category of 'Glues' to discuss what is the best clustering algorithm. This category is composed by 21 Products.

The elbow method, as illustrated in Figure 6.8, suggests that the optimal number of clusters is 4. However, upon examining the product distribution depicted in Figure 6.9, it becomes evident that we can categorize the products into 3 primary groups: liquid glues, adhesive tape, and glue sticks. For a more comprehensive analysis, we could further subdivide the glues into 4 distinct categories: super glues, liquid glues, adhesive tape, and glue sticks. This visualization also provides insights into the distribution of these products. Notably, this graphic is generated by applying PCA to reduce the data to 2 dimensions. Moving forward, our objective is to determine the most effective product division and evaluate how various algorithms categorize these products.

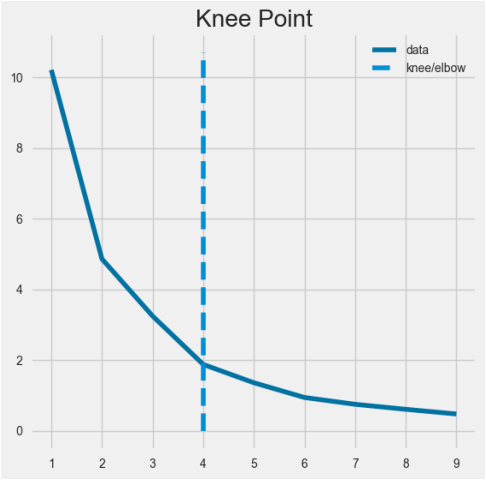


Figure 6.8: Best number of Clusters to Glues Category using Elbow Method

We begin by testing the DBSCAN algorithm and arrive at the conclusion that it is not a suitable clustering method for addressing the problem. This is primarily due to its tendency to partition the products into only two groups: one comprising outliers and the other containing the

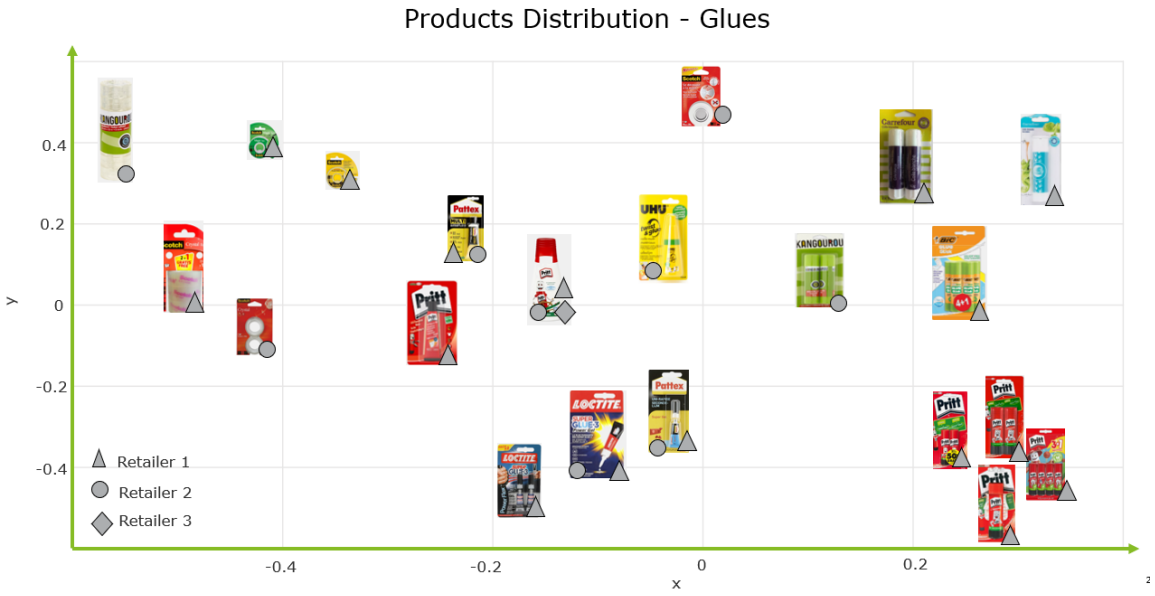


Figure 6.9: Distribution of Products in the Category of Glues

remaining products. In the category under consideration for determining the optimal clustering algorithm, DBSCAN clusters all products together, except for the product at position (0, 0.4), which is appropriately identified as an outlier. However, it is worth noting that when we explore other categories, we observe that DBSCAN's effectiveness in selecting outliers is not consistently high.

In the last chapter, we explain that if we combine DBSCAN and K-means we can first remove outliers and then cluster the remaining products. However, as the DBSCAN has a high probability of removing unwanted products we choose to discard both approaches with DBSCAN.

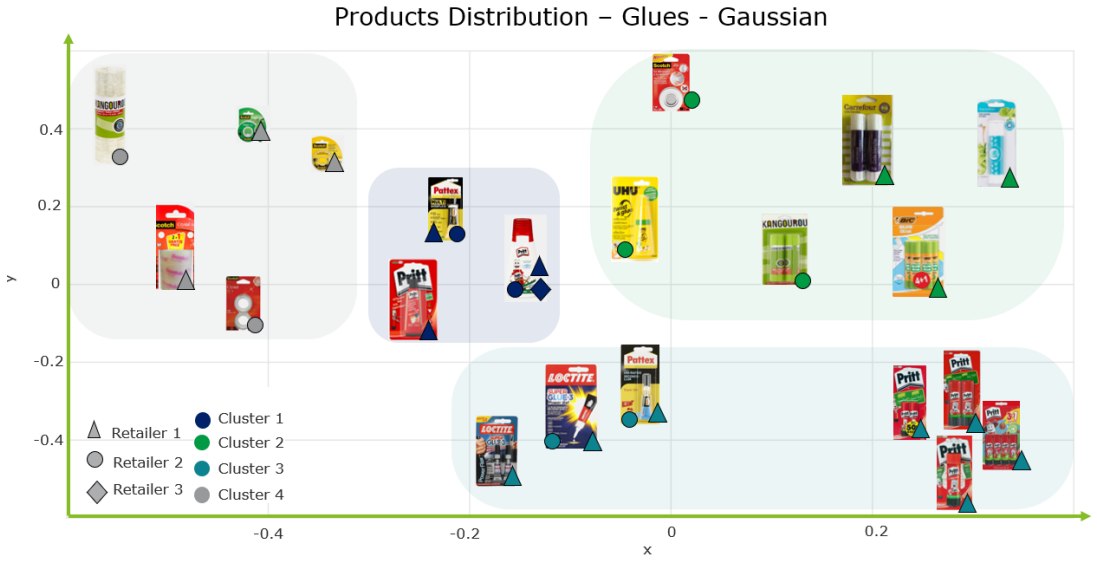


Figure 6.10: Clustering 'Glues' using Gaussian Method

In Figure 6.10 and Figure 6.11, the outputs of the Gaussian and Bayesian Gaussian models

are respectively illustrated. It is clear that neither of these distributions effectively divides the products based on their functionality.

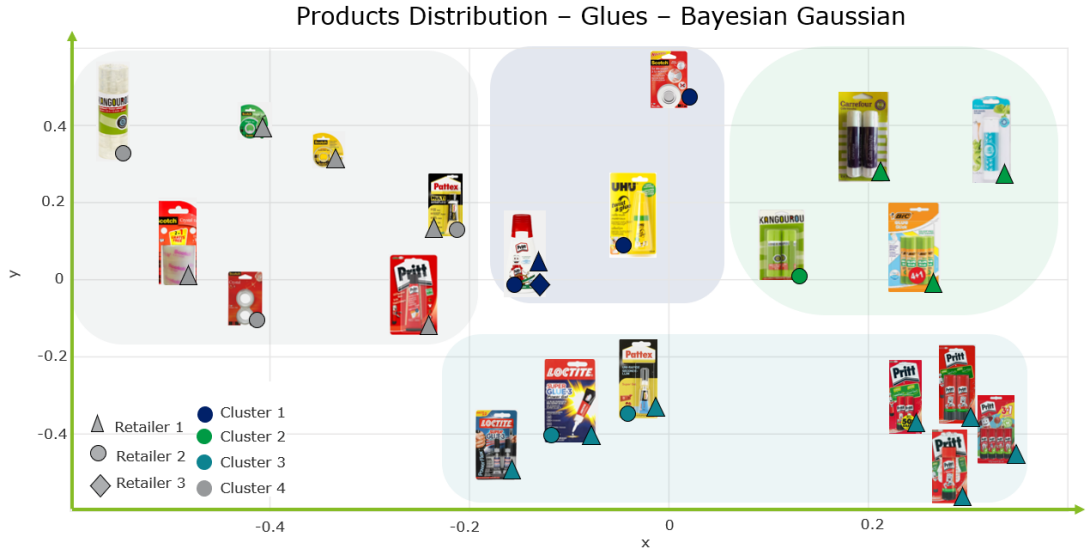


Figure 6.11: Clustering 'Glues' using Bayesian Gaussian Method

The best results are obtained using Hierarchical Clustering and k-means. In Figure 6.12 and in Figure 6.13, the outputs of the k-means and hierarchical clustering are respectively illustrated.

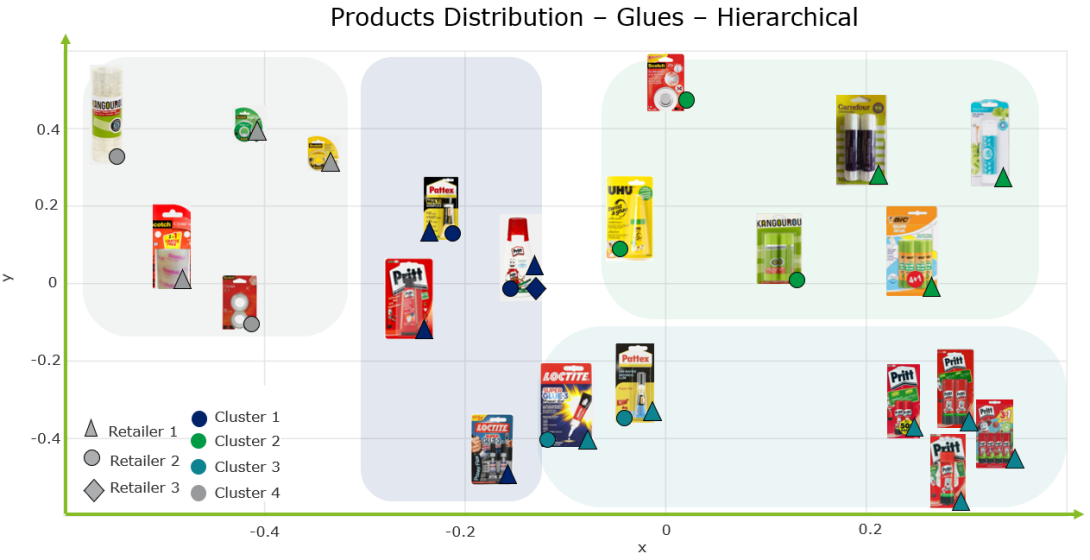


Figure 6.12: Clustering 'Glues' using Gaussian Hierarchical Clustering

As observed, none of the algorithms achieve a perfect division of the groups; however, K-means stands out as the method that comes closest to a satisfactory division, taking into account that, previously we see, that the optimal K is 4. In the K-means plot, we can discriminate the algorithm's attempt to distinguish adhesive tape from liquid glues while creating two distinct clusters for glue sticks. This is because the name and description of the products in Cluster 3 are highly alike, as they belong to the same brand.

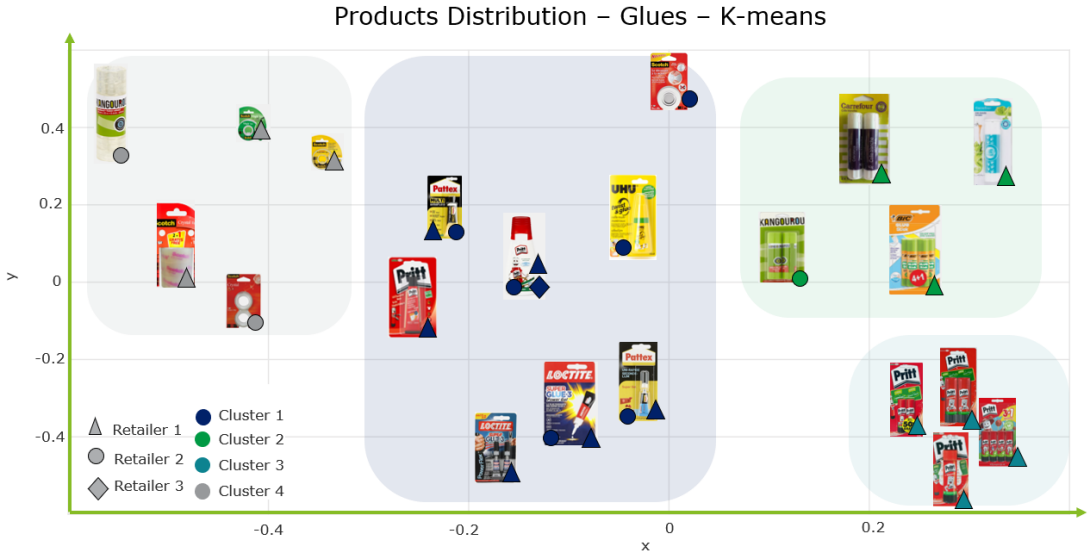


Figure 6.13: Clustering 'Glues' using K-means

As mentioned in the preprocessing section, we removed the brand information from the product names and descriptions. However, upon examining tables of Figure 6.14, it becomes evident that in Cluster 3, the textual descriptions of the products are similar. Consequently, the algorithm infers that clusters 2 and 3 should not be merged.

Cluster 1		Cluster 2		Cluster 3		Cluster 4	
	**Name:** Multi colle multi-usages **Description:** *Lavable		**Name:** Bâton de colle 2x10g Blanc **Description:** 2 Bâtons de colle 10g - Lavable		**Name**: Original 3+1 Bâtons de colle 4x22 g My Heroes **Description**: Colle fiable, simple à utiliser et donnant de beaux résultats pour le papier, le carton et les photos.		**Name:** Ruban adh. transparent **Description:** *8 x 18 mm x 20 m\n*Transparent
	**Name:** Tube de colle multi-usages 20g **Description:** La Pritt Colle Multi-Usages tube (20g) est une colle liquide dans un tube pratique. Ce produit est idéal pour coller papier, carton et photos. Caractéristiques de produit Pour une utilisation à lécole et à la maison.Applications-Papier- CartonPour la sécurité de vos enfantsCe Stylo à colle est sans...		**Name:** Bâton de colle 40g Blanc **Description:** Bâton de colle - Blanc. Grâce à son séchage lent, il permet aux enfants de maternelle et primaire de bien positionner leur su...		**Name**: Original Bâtons de colle Forte 2x22g 2ième à 1/2 prix **Description**: Colle fiable, simple à utiliser et donnant de beaux résultats pour le papier, le carton et les photos. Ideale pour les enfants.		**Name:** Magic Ruban Adhésif Invisible avec dévidoir 19 mm x 15 m **Description:** Scotch® Magic™ Ruban Adhésif - Dévidoir Rechargeable - 19 mm x 15 m
	**Name:** Super Glue Precision **Description:** Pour surfaces larges | Colle instantanée Precision		**Name:** ECOlutions Bâtons de Colle Blanche 8 g 4+1 **Description:** Bâton de colle Fabriqué à partir de 100% de matériaux recyclés. Sans solvant...		**Name**: Original bâton de colle 43g **Description**: La manière la plus propre, (...) Colle fiable, simple à utiliser et donnant de beaux résultats pour le papier, le carton et les photos. Sans solvant et lavable à 20°C.		**Name:** Crystal ruban adh. 12mmx10m **Description:** *10 m *Transparent
...	...	...	...	...	...	...	...

* Name and Description before pre-processing

Figure 6.14: Name and description of some products in the Glue category

6.4 Results for a specific category

In this section, we will delve into the outcomes achieved by implementing our algorithm within the context of two distinct categories: markers and correctors. These categories were selected

because they are easy to distinguish visually, allowing us to illustrate the algorithm's output with more clarity.

6.4.1 Markers Category

The markers subset consists of 48 products, with an average of 3.46 words in the product name and an average of 33.6 words in the description for each product. We have combined all the textual features of each product, including the product name repeated twice, adjectives, and the description, into a single column. In this column, the average number of words used to describe each product is 37 words.

The elbow method provides a clear indication of the optimal number of clusters for the markers category. Therefore, we confidently conclude that the optimal number of clusters is 3 (as it can be seen in Figure 6.15).

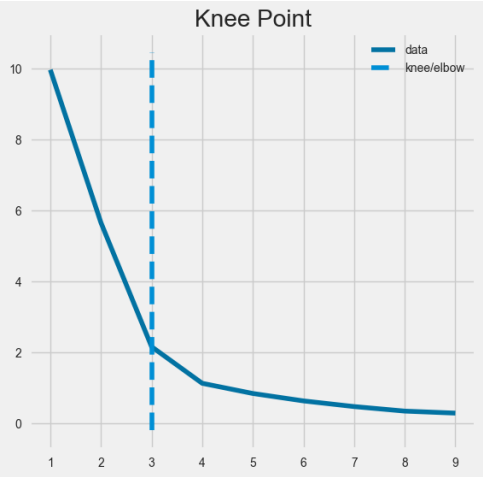


Figure 6.15: Best number of Clusters to Markers Category using Elbow method

The plot shown in Figure 6.16 provides a visual representation of the product distribution and the corresponding retailer(s), as well as the clusters they belong to after applying the k-means clustering algorithm. This plot allows for a clear understanding of the similarities between products.

The final step in the clustering process is to determine the name of each cluster. We have listed the most commonly occurring words in each cluster below:

Cluster 0: feutres, couleur, pointe, pen, moyenne, colorparade, feutre, velleda

Cluster 1: marqueur, permanent, bleu, noir, marqueurs, blanc, feutre, marker

Cluster 2: boss, surligneurs, pastel, fluo, swing, cool, surlign, mini. desksetIn Figures 6.17, 6.18 and 6.19, it is possible to see the textual columns (before preprocessing) used to group products, as well as, the images of the products and the cluster's names.

Figure 6.16: Clustering 'Markers' using K-means

Cluster 0 Nr of Products: 12 Cluster Name: feutres couleur

Number	Name	Description
9	Velleda feutres	
15	12 Marqueurs de couleur	Le stylo-feutre à colorier qui nettoie Adva...
16	18+6 Feutres pointe moyenne	Les feutres de coloriage BIC Kids Kid Co...
17	12 feutres de coloriage pointe fine, 3+	A base deau, lencre ultra lavable des feutres Carrefour se nettoie sans difficulté sur la...
18	Feutres de couleur Trio A-Z 16+8 gratuit	Ecrire et dessiner en couleursLa ergonomique et triangulaire du STABILO Trio A-Z min...
20	18 Feutres à pointe moyenne 3+	Feutres ultra lavables. Pointe moyenne. Pointe sécurisée.
36	20 feutres Colorparade Pen 68 Rose/Lilas	Disponible en 47 couleurs, STABILO Pen 68 est un feutre pointe moyenne pour le dessin...
37	20 feutres Colorparade Pen 68 Bleu/Turquoise	Disponible en 47 couleurs, STABILO Pen 68 est un feutre pointe moyenne pour le dessin...
47	Kids 12 Stylos Feutre Couleur XL	Les feutres Kid Couleur XL de BIC Kids ont une encre ultra-lavable à base deau, facile ...
48	5 Feutres pour tableaux blancs	Maped 741815.
50	ColorPeps Feutre effaçable x1	Pochette carton de 12 feutres jumbo ultra...
51	Pochette 10 Surligneurs Pen 68 Neon	Pochette carton de 12 feutres jumbo ultra lavables pointe large pour un grand pouvoir...

Figure 6.17: Cluster 0 of Markers Category

Cluster 1 Nr of Products: 14 Cluster Name: marqueur permanent

Number	Name	Description
1	marqueur 90N noir	ARTLINE 90 \n NOIR - 2 PCS
2	marqueurs permanents noirs	KANGOUROU MARQUEURS PERMANENTS...
3	marqueurs pr tabl. blanc	
10	marqueur permanent fin	
11	1 Stylo-feutre - Noir	Marqueur permanent- Pointe ultra fine...
21	Marqueur permanent 1,5 mm - Bleu	Largeur décriture 1.5 mm, Rouge
26	770 Freezer Bag Marker - Bleu	Marqueur à encre permanente sans xylène...
33	Marqueur 2-5 mm Rouge 90N	Marqueur permanent à pointe biseautée...
35	Marqueur indélébile 0,8 mm argenté 999 XF	Marqueur permanent à encre peinture métallique couvrante sans xylène...
38	Marqueur permanent 1,2mm (990 XF) - Doré	""Encre permanentePour usage sur papier, verre, plastique, etc.Sans xylène""
40	Garden Marker Feutre 0,8 mm Noir	ARTLINE Marqueur à encre permanente sans xylène...
42	Marqueur permanent 0,7mm - Bleu	Marqueur permanent avec clip et pointe ogive extra fine en feutre...
43	Marqueur permanent 0,7 mm (671102) - Rouge	Marqueur rechargeable à encre permanente. Séchage rapide. Ne contient ni xylène...
49	Marqueur 2-5 mm Bleu 90N	Marqueur permanent à pointe biseautée...

Figure 6.18: Cluster 1 of Markers Category

Cluster 2 Nr of Products: 22 Cluster Name: boss surligneur pastel

Number	Name	Description
4	Boss marqueurs jaune fluo	
5	Maqueurs | Tableu blanc | 4 couleurs | 1,2 mm	- Pointe fine: largeur de trait 1.4mm - Encre pour essuyage à sec...
6	Swing Cool surlign.fr.pastel	
7	Swing Cool surlign.ch.pastel	
8	surligneur fluo Classic	
12	Boss Surligneur Orange 2-5	Tout le monde connaît le BOSSORIGINAL...
14	4 Mini surligneurs Fluo	Lot de 4 mini surligneurs fluos- Pointe ...
19	Boss Surligneur recharg Jaune	Tout le monde connaît le BOSSORIGINAL...
22	Boss Surligneur recharg.-Vert	Tout le monde connaît le BOSSORIGINAL...
23	6 Surligneurs Swing Cool	Son apparence <<givré>>, ses teintes...
24	Boss Pastel Surligneur - Vert	Avec cette collection Pastel composée de 10...
27	4 Surligneurs Boss Original	Tout le monde connaît le BOSSORIGINAL...
28	Boss Mini Pop 5 surligneurs	Tendance, mini, sexy, ... STABILO BOSS...
29	6 Surligneurs Boss Original Pastel	Pochette 6 STABILO BOSS ORIGINAL Pastel STABILO ORIGINAL
30	6 Marqueur swing cool pastel	Surligneur de poche avec agrafe 6 nouvelles...
32	Boss Pastel Surligneur - Turquoise	Avec cette collection Pastel composée de 10 couleurs, STABILO BOSS ORIGINAL...
39	15 Surligneurs Boss Original +	Deskset 15 STABILO BOSS ORIGINAL - ...
41	Boss Surligneur Bleu 2-5 mm	Tout le monde connaît le BOSSORIGINAL...
44	15 Surligneurs Boss Original Pastel + deskset	14 couleurs pastel - Pointe biseautée pour 2 largeurs décriture 2 MM et 5 MM
45	6 Surligneur Boss Original Fluo	Le surligneur peut être laissé sans capuchon jusquà 4 heures sans sécher, il est...
46	Filet 5 surligneurs STABILO Neon	STABILO NEON matière soupleprocédè anti-dessèchementlargeur de tracé...
52	6 Surligneur Boss Original Pastel	Avec cette collection Pastel composée de 10 couleurs, STABILO BOSS ORIGINAL...

Figure 6.19: Cluster 2 of Markers Category

6.4.2 Correctors Category

The corrector subdataset consists of 26 products, with an average of 5.26 words in the product name and 30.9 words in the description for each product. After we have done the preprocessing and have joined the name twice, each product in the correctors category is described, on average, by 28 words. Based on the analysis using the elbow method, the most suitable number of clusters is 4 as shown in Figure 6.20.

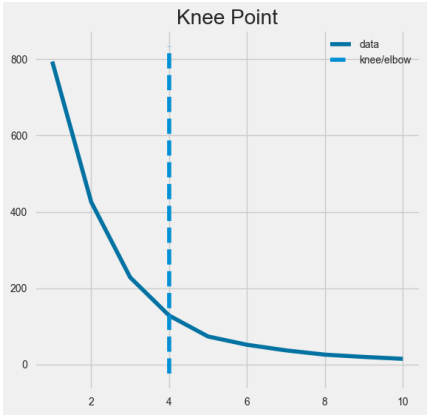


Figure 6.20: Best number of Clusters to Correctors Category using Elbow method

Similarly to markers products distribution, the plot in Figure 6.21 provides a visual representation of the correctors distribution and the corresponding retailer(s), as well as the clusters they belong to after applying the k-means clustering algorithm.

Figure 6.21 offers a visual distribution of various products and their corresponding retailers. The plot also reveals the clusters assigned to the products using the k-means clustering algorithm, enhancing understanding of the similarities among the products.

Figure 6.21: Clustering 'Correctors' using K-means

Cluster 0 Nr of Products: 7 Cluster Name: effaceurs dencre

Number	Name	Description
1	Effaceurs d'encre	
8	4 Effaceurs dencre medium	bleu
10	10 Effaçeurs dencre M	Effaceurs 10 Pcs
14	Effaceurs dencre F Super Sheriff (921759)	Le Pelikan Super Sheriff avec quatre nouveaux motifs et un effet métallique qui att...
15	Effaceurs Correcteur Encre Bleue 4 pièces	
17	6 Effaceurs dencre Super Pirat	2 types de pointes différents pour chaque type dutilisation: F et B. B (Large) = pour supp...
24	Effaceurs Correcteur Encre Bleue 8+4	Effaceur à Pointe Moyenne, Réécriture à lencre...

Cluster 1 Nr of Products: 7 Cluster Name: correcteur rubans

Number	Name	Description
5	Ruban correcteur 10m Soft Grip	Ruban correcteur 10m Soft Grip
7	Rubans correcteur 3x5m	Souris correctrices, 3 pcs
9	3 Rubans correcteur 3x5 m Couleur aléatoire	Le ruban correcteur permet une correction instanée pour une réécriture immédiate. Sa ...
11	2+1 Rubans correcteur Micro Tape 8Mx5mm Aléatoire	Principaux bénéfices: Ruban 5 mm x 8 m Ruban papier pour un meilleur contrôle C...
18	2+1 Rubans correcteur Easy Correct 12Mx4,2mm	Application précise, réécriture immédiate avec le ruban sec Tipp-Ex. Ruban correcte...
27		
20	Compact Roller Flex 4,2 mm	Application régulière et correction propre grâce à lapplicateur flexible mini roller.
25	2 + 1 Rubans correcteur Fancy Roller Turquoise...	""Roller Blanco® Fancy-Design à la mode-Petit et pratique-Ne contient pas de solvant...

Figure 6.22: Cluster 0 and 1

Cluster 2 Nr of Products: 1 Cluster Name: mouse mini pocket

Number	Name	Description
2	Mini Pocket Mouse Fun	
3	Mini Pock.Mouse	
4	Mini Pocket Mouse roller	
6	2+1 Rubans correcteur Mini Pocket Mouse 6Mx5mm	Ruban correcteur au design ultra compact avec un film à base de papier pour une bon...
12	2+1 Rubans correcteur Mini Pock Mouse	Ruban 5 mm x 5 mRuban papier pour un meilleur contrôleCorrection instantanée, pr...
16	Ruban correcteur 10m Pocket Mouse	Correcteur en de souris pour une correction instantanéePrincipaux bénéfices:- Ruban...
19	3 Rubans correcteur Mini Pocket Mouse	Principaux bénéfices: Ruban 5 mm x 5 m. Ruban papier pour un meilleur contrôle. Co...

(2) (3) (4) (6) (12) (16) (19)

Cluster 3 Nr of Products: 5 Cluster Name: correcteur stylo

Number	Name	Description
5	Ruban correcteur 10m Soft Grip	Ruban correcteur 10m Soft Grip
13	3 Correcteurs Stylo Séchage rapide	3 Stylo correcteurs 8ml
21	Stylo correcteur Shake N Squeeze	La formule à séchage rapide et le contrôle du débit de liquide facilitent la réécriture. Bon...
22	Stylo correcteur Pocket Pen 8ml 1+1 gratuit	Stylo pour correction précise de points, et de petites ou grandes surfaces. Grâce à sa form...
23	ECO Roller Flex 4,2 mm	Application latérale ergonomique Eco dans un boitier à 100% en plastique recyclé
26	2+1 Rubans correcteur Eco Flex 10Mx4,2mm Blanc	LLe roller de correction Eco Flex est fabriqué à partir de plastique 100% recyclé et perme...

(13) (21) (22) (23) (26)

Figure 6.23: Cluster 2 and 3

The final step is to determine the name of each cluster. We have listed the most commonly occurring words in each cluster below:

Cluster 0: effaceurs, dencre, encre, super, correcteur, bleue, medium, effaçeurs

Cluster 1: correcteur, rubans, aléatoire, roller, couleur, micro, tape, easy

Cluster 2: mouse, mini, pocket, correcteur, rubans, pock, fun, roller, ruban

Cluster 3: correcteur, stylo, flex, ruban, soft, grip, correcteurs, séchage rapide

In Figure 6.22 and in Figure 6.23, it is possible to see the textual columns used to group products (name and description), as well as, the images of the products and the cluster's names.

6.5 Discussion

In the preceding sections, we elucidated the clustering algorithm designed to group similar products based on textual similarities. In this section, our focus shifts to the discussion of the results, with special attention given to the markers and correctors categories. Furthermore, we will also compare the benefits of our approach to the baseline, and explore the limitations of the algorithm.

6.5.1 Analysis of the markers category clusters

Regarding the markers category, in cluster 0, the algorithm attempts to cluster coloring markers, while in cluster 1, it tries to cluster regular markers, and in cluster 2, it clusters highlighting markers.

Regarding cluster 0, two products, 9 and 48, do not belong to the correct cluster since they are not coloring markers. Yet, if we consider the names of these markers, we can see that they both have the word "feutres", which is the most common word in cluster 0, and no words related to the ones in cluster 1 (the correct cluster to these products). Therefore, it is not surprising that these products are not in the right cluster. Additionally, the products distribution plot indicates that they are on the border of the cluster, near to cluster 1.

In cluster 2, there is one product, product 5, that belongs to the wrong cluster. If we look at the plot, this product is also on the boundary, near to cluster 1.

A closer look at the product distribution plot reveals that some products in cluster 2, specifically products 12, 19, 22, 27, and 41, are slightly distant from the other markers in the cluster. Upon examining their description column, we find that it is the same for all of these products. Thus, the algorithm identifies more similarities between these products.

In cluster 1, there is a proximity among the black markers, as well as the blue markers. However, the product located in the bottom right corner, which is Product 10, is also black, but

it is farther away from the other black markers. This could be attributed to the fact that neither the product name nor its description explicitly mentions the color black.

6.5.2 Analysis of the correctors category clusters

The algorithm divides the items into four distinct clusters, which appear to be the appropriate number of clusters. In cluster 0, it groups ink erasers together. Cluster 1 contains normal-sized corrector tapes, while cluster 2 is dedicated to mini-sized corrector tapes. Lastly, in cluster 3, we can find pen correctors.

After analyzing the products in cluster 0, it can be concluded that all ink erasers are included in this cluster. However, in cluster 1, while the majority of products consist of normal-sized tape correctors, Product 11 and Product 18 are mini-sized correctors. This suggests that they should belong to cluster 2. Nevertheless, upon examining the names and descriptions of these products, an observation can be made. Product 18 lacks any reference to its size, making it impossible to determine that it is most similar to products in cluster 2.

As for Product 11, although the term “micro” indicates that it should be assigned to cluster 2, it is worth noting that cluster 2 products predominantly utilize the expression “Mini Pocket Mouse” in their descriptions. Consequently, the algorithm fails to recognize the similarity between the words “micro” and “Mini Pocket Mouse”. This underscores a failure in synonym processing, as these two words should be considered synonymous. This is because in the WordNet library, the term ’Mini’ does not have synonyms.

In cluster 3, we find a collection of pen correctors; however, it is worth noting that two normal-sized tape correctors (products 23 and 26) are also present within this cluster. Analyzing product 23, we observe that neither the name nor the description provides any indication of the type of corrector it is. As a result, the algorithm faces difficulty in determining the appropriate cluster for this product, as its classification relies only on textual features. Regarding product 26, its name clearly suggests that it should not belong to cluster 3. However, the algorithm’s decision to assign this product to that cluster is likely influenced by their shared attributes of ’eco’ and ’plastic.’

6.6 Baseline Vs Our Approach

As previously discussed, our baseline model involves clustering products based on their EAN (European Article Number) codes. We have employed K-means for this purpose. If we return our attention to the glues category, depicted in Figure 6.24, it is evident that the products are not evenly distributed, which suggests that we should not anticipate robust clustering results under these conditions.

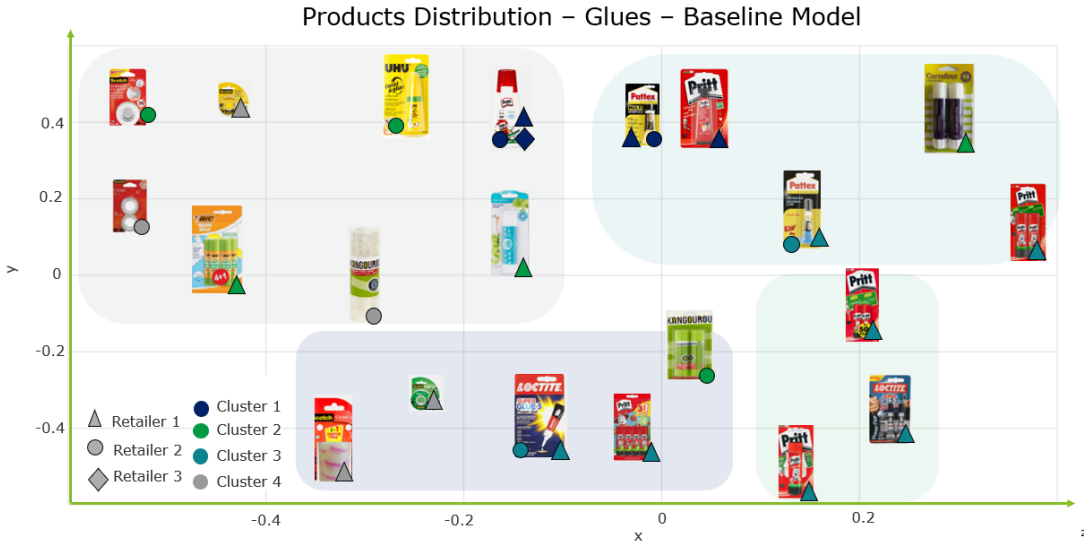


Figure 6.24: Clustering 'Glues' using Baseline Model

Comparing the outcomes of the baseline model with those of our approach, it becomes apparent that although our approach is not flawless, it significantly outperforms the baseline model.

6.7 Summary

In summary, this section starts by understanding the results that our preprocessing produced. Subsequently, we recognized the significant contributions of TF-IDF and the Elbow Method in assisting us in determining the optimal number of product clusters. Following this, we established that K-means emerged as the most effective approach for product clustering.

Lastly, we conducted an in-depth analysis of the results within two distinct categories. Despite encountering some problems, it is evident that valuable insights can be extracted from our approach.

Chapter 7

Applications

Up to this point, we have introduced a clustering algorithm specifically designed to categorize products according to their textual attributes. This methodology has enabled us to categorize products into distinct groups. As an example, consider the product division within the 'Colas' category, which is visually depicted in Figure 7.1, demonstrating the clear separation of colas into three distinct clusters.

In this chapter, we will delve into two practical applications of this algorithm, each of which holds significant potential for enhancing the user experience and the overall functionality of the system. We start with the example of Coca-Colas to illustrate the 2 applications mentioned previously.

In the first section, we present the first application that involves displaying groups of similar products, allowing users to effortlessly navigate through similar items. In the second section, we explain how the clustering algorithm can be used to implement a recommendation system in the app for users.

7.1 Similar Products Identification

One of the primary utilities of our clustering algorithm is its ability to organize products into meaningful groups based on their textual attributes. These clusters can serve as a valuable resource for both the company and its customers. By offering a visually coherent representation of similar products within each cluster, it becomes easier for users to compare prices, specifications, and other relevant information. This functionality empowers consumers to make more informed purchasing decisions, while also helping businesses highlight product similarities, facilitating strategic pricing adjustments, and streamlining inventory management.

For example, to identify similar products to product 4 (see Figure 7.1), simply look within its corresponding cluster. In Figure 7.2, it is possible to compare the prices of these similar products in each supermarket on the same day. The retailer is identified at the bottom of Figure 7.1 by

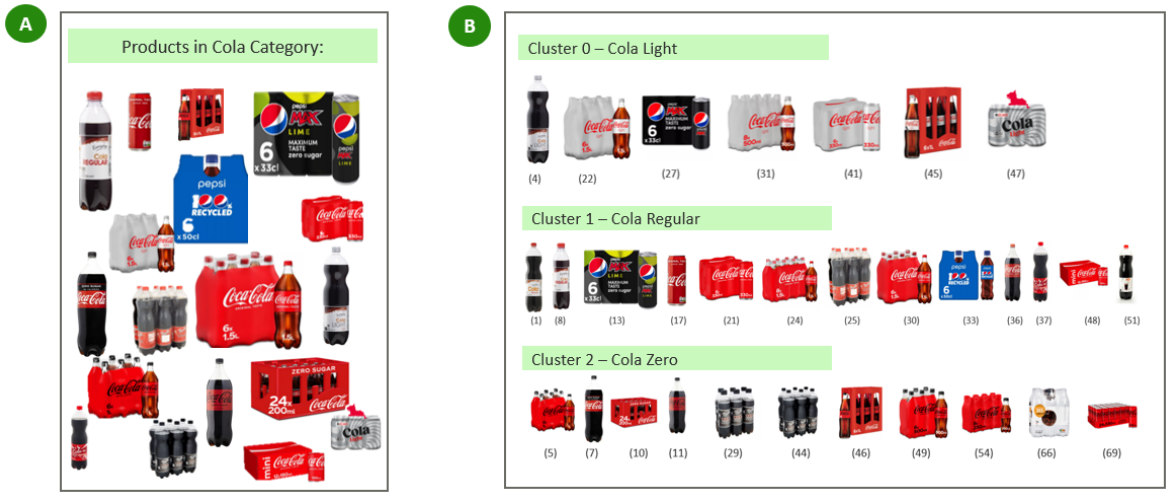


Figure 7.1: (A) Products in Cola Category (B) Products divided into 3 clusters: Light, Regular and Zero

the numbers 1, 2, 3, and 4, as well as the price and the price per measurement unit.

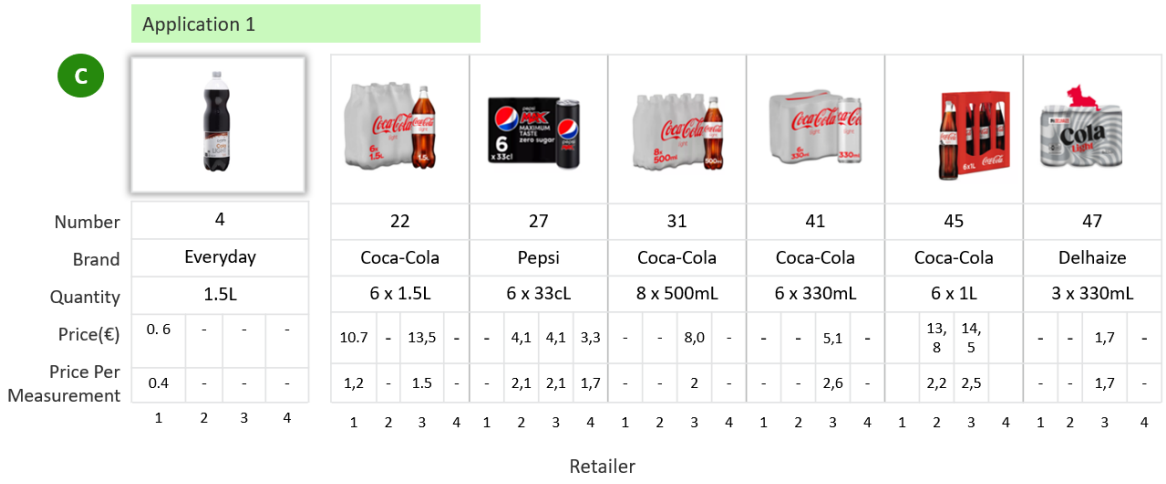


Figure 7.2: Similar Products of Product 4

7.2 Recommendation System

In addition, our clustering algorithm opens up the possibility of implementing a highly effective recommendation system within the application. By grouping products into clusters, we can leverage these associations to offer product recommendations to users. When a customer expresses interest in a product from a particular cluster, the algorithm can suggest other items from the same cluster. This personalized approach enhances user engagement, encourages cross-selling, and contributes to customer satisfaction by delivering relevant and appealing product suggestions.

In order to do that we will suggest perfect and imperfect substitutes based on the rules

presented in Figure 7.3 and in Figure 7.5, respectively.

Perfect Substitutes

The perfect substitutes refer to a pair of products with uses identical to one another. Producers and sellers of perfect substitute goods directly compete, that is, they are known to be in direct price competition because the content of a product is exactly the same.

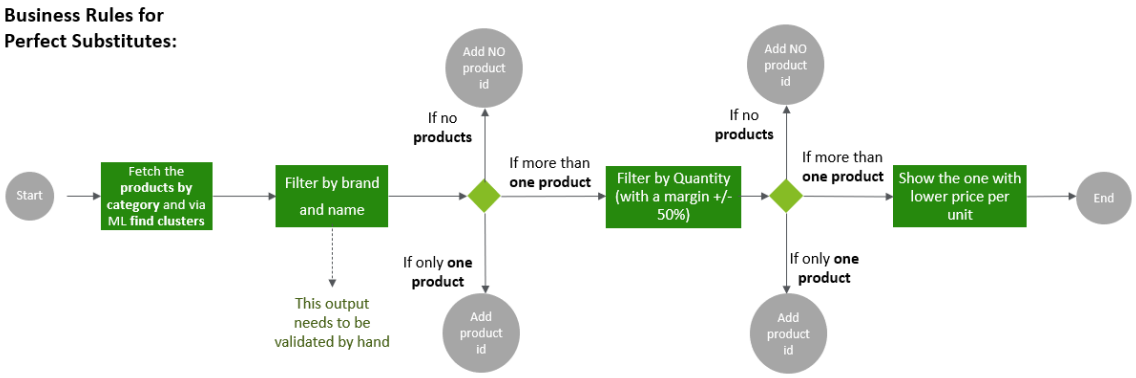


Figure 7.3: Business Rules to Recommend a Perfect Substitute Product

In order to get the most similar perfect substitute for a product, we follow these steps (Figure 7.3):

1. Identify the product's cluster and gather other products from the same cluster.
2. Select products with the same brand.
3. Compare the contents of the selected products manually, discarding any that do not match the desired content. Note: If ingredient lists are unavailable, we cannot rely on that information for comparison. (At this point we have all the perfect substitutes)
4. If multiple products remain, filter them based on quantity. Ensure the remaining products have quantities similar to the original product (within a range of +/- 50%).
5. If only one product remains, that becomes the substitute. Otherwise, choose the product with the lowest price per unit and consider it as a substitute.

Consider, for the same category (Figure 7.1), the case of product 17 and its perfect substitutes, namely products 21, 24, 30, and 48, as depicted in Figure 7.4. These products share the same cluster, brand, and content attributes. As we progress through step 4, it becomes evident that there are no remaining products to recommend, as none of them exhibit a similar quantity to that of product 17.

Now, let's shift our focus to the recommendation of perfect substitutes for product 24. These substitutes include products 17, 21, 30, and 48. However, after completing step 4, only product 30 remains as a viable recommendation due to its approximate equivalence in quantity.

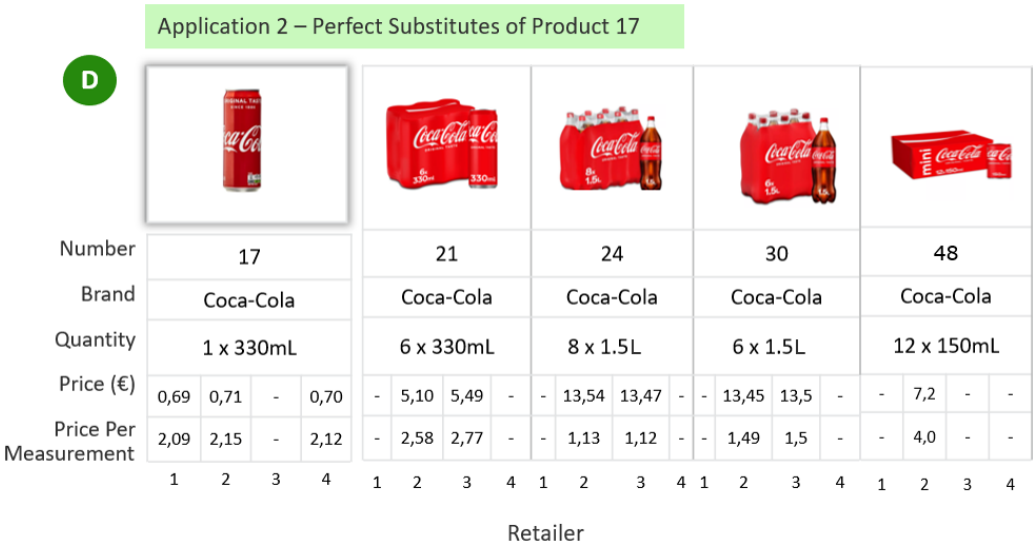


Figure 7.4: Perfect Substitutes of Product 17

Imperfect Substitutes

On the other hand, an imperfect product can serve as a viable alternative to another product. These products are similar in satisfying the same needs, but they possess slight variations in characteristics.

To find an imperfect substitute to recommend for a particular product, we can follow these steps (Figure 7.5):

1. Identify the product's cluster and gather other products from the same cluster.
2. Filter the selected products based on quantity, ensuring they have quantities similar to the original product within a range of +/- 50%. If less than 5 products, return these products.
3. Group the products by retailer.
4. Choose the products with the lowest price, the first product in the list with the best nutrition score, the first product in the list with the best eco score, the first private label product, and the first non-private label product.
5. If the resulting list has fewer than 5 products, continue selecting the first product in the list (that has not been selected yet) until the list contains 5 products.

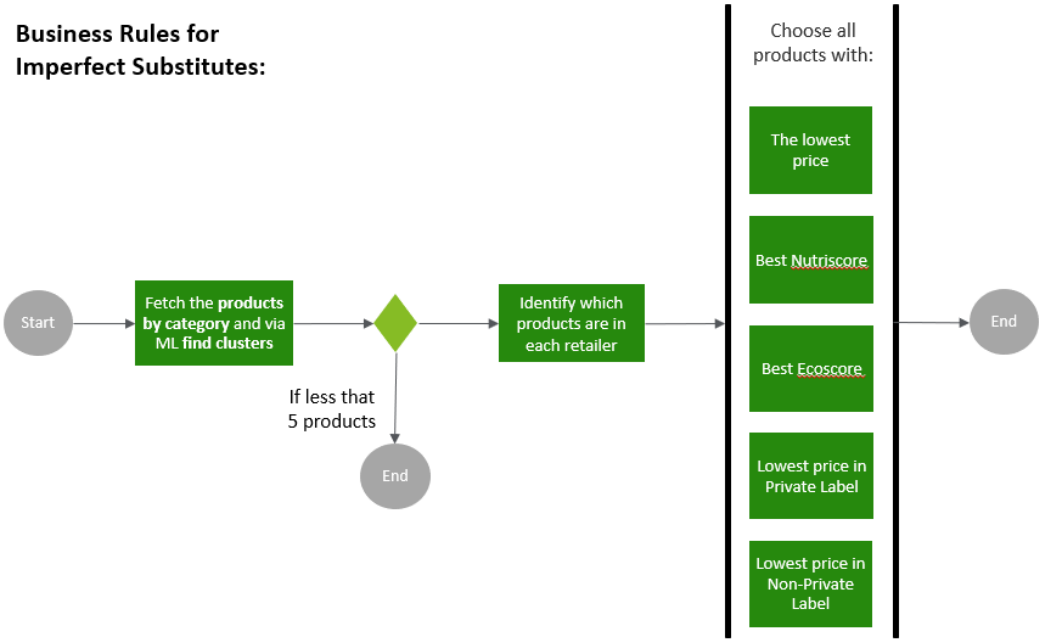


Figure 7.5: Business Rules to Recommend an Imperfect Substitute Product

Let's return once more to the example of Coca-Colas shown in Figure 7.1, for instance, the scenario involving product 5 and its imperfect substitutes, specifically products 29, 44, and 66, as illustrated in Figure 7.6. Given the similarity in quantity among all these products, and considering that we have only three imperfect substitutes (which are fewer than five), it results in the recommendation of all products.

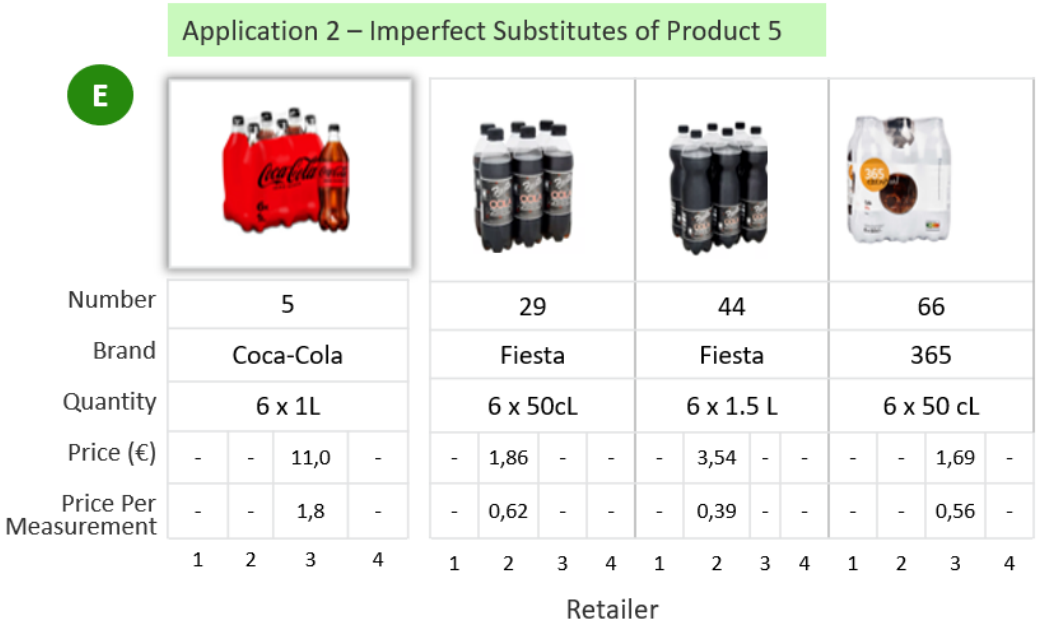


Figure 7.6: Imperfect Substitute of Product 5

After selecting the perfect and imperfect products, we employ GraphQL query to generate a

JSON file, enabling us to efficiently transmit recommendation information to the application's front end. GraphQL, a powerful query language and API runtime, empowers clients to request customized data, enhancing data retrieval efficiency and flexibility.

The GraphQL query retrieves a product's substitute ids, along with associated information such as nutrition score, organic label (isbio), and the list of retailers where these substitute products are available.

An example of the JSON File output of graphQL is:

```
{
   substituteProducts: {
        nodes[
            {
                matchtype: PERFECT
                id: 10
                exists_in: [1,2]
                nutriscore: A
                isBio: True
            },
            {
                matchtype: IMPERFECT
                id: 12321
                exists_in: [1,2,3]
                nutriscore: B
                isBio: FALSE
            },
            {
                matchtype: PERFECT
                id: 122
                exists_in: [3]
                nutriscore: B
                isBio: FALSE
            },
        ]

   }
}
```

7.3 Summary

In this comprehensive chapter, we embark on a journey to explore the versatile applications of clustering algorithms. We dive deep into the two distinct but equally impactful use cases: Similar Products Identification and Recommendation Systems.

When it comes to the identification of similar products, we have a comprehensive understanding of how to recognize and compare them effectively. In the realm of recommendation systems, we excel in distinguishing between perfect and imperfect substitutes and selecting products that closely align with the desired item, enabling us to provide good recommendations.

Chapter 8

Conclusions

This dissertation describes a solution to two problems: First, it addresses the issue of user difficulty in finding related products, offering a solution by enabling the display of similar products together, simplifying navigation and enhancing the user experience. Second, it tackles the problem of product recommendations, solving it by implementing a recommendation system powered by the clustering algorithm, thus helping users discover items aligned with their preferences and interests. In summary, the clustering algorithm resolves the challenges of product discoverability and recommendation in the app, leading to improved user satisfaction and engagement.

The clustering approach relied on textual similarity as its primary guiding principle. Throughout the study, data cleaning and pre-processing emerged as crucial initial steps to ensure the reliability of the results. The utilization of TF-IDF (Term Frequency-Inverse Document Frequency) for text vectorization proved to be an exceptionally practical choice. It showcased its effectiveness in capturing the significance of words within a corpus, even when dealing with noisy and limited textual data. Ultimately, the application of the k-means clustering algorithm successfully organized products into coherent and logical groups, yielding sets of similar products that were both insightful and useful.

However, it is important to acknowledge the limitations encountered during this research endeavor. One of the most notable shortcomings was the scarcity of high-quality data. A more meticulously curated and structured dataset would have undoubtedly strengthened the depth and accuracy of the analysis. Additionally, a significant challenge faced by the researchers was their limited proficiency in the French language, which potentially restricted the scope and depth of the study when dealing with French-language product descriptions.

Looking forward, there are promising avenues for advancing this research. One such path involves exploring methods to detect outliers within each category, thereby enhancing the overall cohesion of the clusters. This effort would contribute to refining the cluster representations.

Another avenue for improvement lies in formulating an efficient approach for clustering new products, considering the existing clusters, and potentially creating new ones as needed. This effort would ensure that the system remains adaptable and can accommodate the continuous

influx of new products, maintaining its effectiveness in product categorization.

Furthermore, the analysis of image data could be a valuable but challenging addition to this research. While textual analysis is effective in understanding product descriptions, incorporating image analysis could provide a more holistic view of product similarity and enhance the recommendation system. However, image analysis poses challenges such as image quality, variability in product appearance, and the need for robust computer vision algorithms. Ensuring accurate and efficient image processing and feature extraction would require significant technical expertise and resources. Nevertheless, if successfully integrated, image analysis could significantly improve the accuracy and depth of product clustering, offering users a more visually guided and personalized experience when navigating through similar items.

Having access to the clients' purchase register data is immensely valuable because it serves as the foundation for implementing advanced recommendation techniques like Collaborative Filtering, which is endorsed in research papers referenced in Chapter 3. By leveraging this data, Collaborative Filtering can identify meaningful patterns and relationships among clients and products. It enables the system to make highly personalized recommendations, drawing insights from clients who have similar purchase histories. This technique greatly enhances the user experience by suggesting products based on past interactions, ultimately leading to increased user satisfaction and engagement. Additionally, it can boost sales and customer retention by guiding clients toward items that align with their preferences.

It would be beneficial to know the sub-brand names from the textual data because the algorithm may group products based on it. This could potentially lead to inaccurate clustering and affect the effectiveness of the approach. By removing sub-brand names, the clustering approach would be more likely to group products based on their actual characteristics.

Additionally, when there are only a few products in a category that are distinct from the rest, it can be challenging for the algorithm to identify them as similar and create a separate cluster for them.

Because names and descriptions for the products are annotated manually, there is a large variety of distinct annotations for products that should belong to the same group. Besides, some products do not have all characteristics annotated like their size or color, for example. This makes it harder for modeling and retailers should devote resources to the curation and proper annotation of this data in order to take better advantage of the machine learning methods.

Bibliography

[1] Principal component analysis tutorial. Accessed on September, 2023.

[2] Retail across the world — deloitte.com. https://www.deloitte.com/global/en/Industries/consumer/analysis/retail-across-the-world.html.

[3] OpenAI GPT2 — huggingface.co. https://huggingface.co/docs/transformers/model_doc/gpt2. [Accessed 10-Jul-2023].

[4] Natural language processing: state of the art, current trends and challenges - Multimedia Tools and Applications — doi.org. https://doi.org/10.1007/s11042-022-13428-4. [Accessed 08-Jul-2023].

[5] KMeans Clustering for Customer Data — kaggle.com. https://www.kaggle.com/code/heeraldedhia/kmeans-clustering-for-customer-data?scriptVersionId=45166976&cellId=1. [Accessed 09-Jul-2023].

[6] TF-IDF: AI Terms Explained | Blog — playerzero.ai. https://www.playerzero.ai/advanced/ai-terms-explained/tf-idf-ai-terms-explained. [Accessed 03-09-2023].

[7] Akaike Information Criterion - an overview | ScienceDirect Topics — sciencedirect.com. https://www.sciencedirect.com/topics/earth-and-planetary-sciences/akaike-information-criterion, . [Accessed 03-09-2023].

[8] Bayesian Information Criterion - an overview | ScienceDirect Topics — sciencedirect.com. https://www.sciencedirect.com/topics/pharmacology-toxicology-and-pharmaceutical-science/bayesian-information-criterion, . [Accessed 03-09-2023].

[9] 2.3. Clustering — scikit-learn.org. https://scikit-learn.org/stable/modules/clustering.html?source=post_page-----db027e9a871c----------------------------#clustering, . [Accessed 30-08-2023].

[10] 2.3. Clustering — scikit-learn.org. https://scikit-learn.org/stable/modules/clustering.html?source=post_page-----db027e9a871c----------------------------#dbscan, . [Accessed 02-09-2023].

[11] Stemming — link.springer.com. https://link.springer.com/referenceworkentry/10.1007/978-1-4899-7993-3_942-2. [Accessed 08-Jul-2023].

[12] Analytics in retail going to market with a smarter approach — deloitte.com. https://www2.deloitte.com/content/dam/Deloitte/ch/Documents/consumer-business/ch-cb-en-Deloitte-Analytics-in-retail-0514.pdf, 2013.

[13] M. Fatih Adak and Metehan Uçar. A book recommendation system using decision tree-based fuzzy logic for e-commerce sites. In *2021 3rd International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, pages 1–5, 2021. doi:10.1109/HORA52670.2021.9461319.

[14] Charu C. Aggarwal. *The Recommender Systems*. Springer, 2016.

[15] Mohamed Alloghani, Dhiya Al-Jumeily Obe, Jamila Mustafina, Abir Hussain, and Ahmed Aljaaf. *A Systematic Review on Supervised and Unsupervised Machine Learning Algorithms for Data Science*, pages 3–21. 01 2020. ISBN: 978-3-030-22474-5. doi:10.1007/978-3-030-22475-2_1.

[16] Olatz Arbelaitz, Ibai Gurrutxaga, Javier Muguerza, Jesús M. Pérez, and Iñigo Perona. An extensive comparative study of cluster validity indices. *Pattern Recognition*, 46(1):243–256, 2013. ISSN: 0031-3203. doi:https://doi.org/10.1016/j.patcog.2012.07.021.

[17] A. Berchtold. Sequence analysis and transition models. In Michael D. Breed and Janice Moore, editors, *Encyclopedia of Animal Behavior*, pages 139–145. Academic Press, Oxford, 2010. ISBN: 978-0-08-045337-8. doi:https://doi.org/10.1016/B978-0-08-045337-8.00233-3.

[18] Federico Bianchi, Bingqing Yu, and Jacopo Tagliabue. BERT goes shopping: Comparing distributional models for product representations. In *Proceedings of the 4th Workshop on e-Commerce and NLP*, pages 1–12, Online, August 2021. Association for Computational Linguistics. doi:10.18653/v1/2021.ecnlp-1.1.

[19] Johannes Blömer, Christiane Lammersen, Melanie Schmidt, and Christian Sohler. Theoretical analysis of the k-means algorithm – a survey. In Lasse Kliemann and Peter Sanders, editors, *Algorithm Engineering: Selected Results and Surveys*, pages 81–116. Springer International Publishing, Cham, 2016. ISBN: 978-3-319-49487-6. doi:10.1007/978-3-319-49487-6_3.

[20] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez. Recommender systems survey. *Knowledge-Based Systems*, 46:109–132, 2013. ISSN: 0950-7051. doi:https://doi.org/10.1016/j.knosys.2013.03.012.

[21] Hossein Bonakdari and Mohammad Zeynoddin. Chapter 5 - goodness-of-fit precision criteria. In Hossein Bonakdari and Mohammad Zeynoddin, editors, *Stochastic Modeling*, pages 187–264. Elsevier, 2022. ISBN: 978-0-323-91748-3. doi:https://doi.org/10.1016/B978-0-323-91748-3.00003-3.

[22] Eric T. Bradlow, Manish Gangwar, Praveen Kopalle, and Sudhir Voleti. The role of big data and predictive analytics in retailing. *Journal of Retailing*, 93(1):79–95, 2017. ISSN: 0022-4359. The Future of Retailing. doi:https://doi.org/10.1016/j.jretai.2016.12.004.

[23] Ângelo Cardoso, Fabio Daolio, and Saúl Vargas. Product characterisation towards personalisation: Learning attributes from unstructured data to recommend fashion products. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, page 80–89, New York, NY, USA, 2018. Association for Computing Machinery. ISBN: 9781450355520. doi:10.1145/3219819.3219888.

[24] Abhijnyan Chandra, Arif Ahmed, Sandeep Kumar, Prateek Chand, Malaya Dutta Borah, and Zakir Hussain. Content-based recommender system for similar products in e-commerce. In Ripon Patgiri, Sivaji Bandyopadhyay, Malaya Dutta Borah, and Valentina Emilia Balas, editors, *Edge Analytics*, pages 617–628, Singapore, 2022. Springer Singapore. ISBN: 978-981-19-0019-8.

[25] Abhijnyan Chandra, Arif Ahmed, Sandeep Kumar, Prateek Chand, Malaya Dutta Borah, and Zakir Hussain. Content-based recommender system for similar products in e-commerce. *Lecture Notes in Electrical Engineering*, 869:617 – 628, 2022. Cited by: 1. doi:10.1007/978-981-19-0019-8_46.

[26] Olga Cherednichenko, Maryna Vovk, Olga Kanishcheva, and Mikhail Godlevskyi. Studying items similarity for dependable buying on electronic marketplaces. In Vasyl Lytvyn, Natalia Sharonova, Thierry Hamon, Victoria Vysotska, Natalia Grabar, Agnieszka Kowalska-Styczen, and Izabela Jonek-Kowalska, editors, *Proceedings of the 2nd International Conference on Computational Linguistics and Intelligent Systems, COLINS 2018, Volume I: Main Conference, Lviv, Ukraine, June 25-27, 2018*, volume 2136 of *CEUR Workshop Proceedings*, pages 78–89. CEUR-WS.org, 2018.

[27] K. R. Chowdhary. Natural language processing. In *Fundamentals of Artificial Intelligence*, pages 603–649. Springer India, New Delhi, 2020. ISBN: 978-81-322-3972-7. doi:10.1007/978-81-322-3972-7_19.

[28] Tiago Daniel Sá Cunha. *Recommending Recommender Systems: Tackling the Colaborative Filtering Algorithm Selection Problem*. PhD thesis, Department of Informatics Engineering, Faculty of Engineering, University of Porto, 2019.

[29] Nameirakpam Dhanachandra, Khumanthem Manglem, and Yambem Jina Chanu. Image segmentation using k -means clustering algorithm and subtractive clustering algorithm. *Procedia Computer Science*, 54:764–771, 2015. ISSN: 1877-0509. Eleventh International Conference on Communication Networks, ICCN 2015, August 21-23, 2015, Bangalore, India Eleventh International Conference on Data Mining and Warehousing, ICDMW 2015, August 21-23, 2015, Bangalore, India Eleventh International Conference on Image and Signal Processing, ICISP 2015, August 21-23, 2015, Bangalore, India. doi:https://doi.org/10.1016/j.procs.2015.06.090.

[30] Chris Ding and Xiaofeng He. K-means clustering via principal component analysis. In *Proceedings of the Twenty-First International Conference on Machine Learning*, ICML '04, page 29, New York, NY, USA, 2004. Association for Computing Machinery. ISBN: 1581138385. doi:10.1145/1015330.1015408.

[31] Salvador García, Julián Luengo, and Francisco Herrera. *Data Preprocessing in Data Mining*, volume 72. 01 2015. ISBN: 978-3-319-10246-7. doi:10.1007/978-3-319-10247-4.

[32] Kujtim Hameli. A literature review of retailing sector and business retailing types. *ILIRIA International Review*, 8, 07 2018. doi:10.21113/iir.v8i1.386.

[33] Alexander Hübner, Johannes Wollenburg, and Andreas Holzapfel. Retail logistics in the transition from multi-channel to omni-channel. *International Journal of Physical Distribution & Logistics Management*, 46:562–583, 07 2016. doi:10.1108/IJPDLM-08-2015-0179.

[34] Dan Jurafsky and James H. Martin. *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition*. Pearson Prentice Hall, Upper Saddle River, N.J., 2009. ISBN: 9780131873216 0131873210.

[35] Tushar Kansal, Suraj Bahuguna, Vishal Singh, and Tanupriya Choudhury. Customer segmentation using k-means clustering. pages 135–139, 12 2018. doi:10.1109/CTEMS.2018.8769171.

[36] Batta Mahesh. Machine learning algorithms -a review, 01 2019. doi:10.21275/ART20203995.

[37] Norsyela Muhammad Noor Mathivanan, Nor Azura Md. Ghani, and Roziah Mohd Janor. Analysis of k-means clustering algorithm: A case study using large scale e-commerce products. In *2019 IEEE Conference on Big Data and Analytics (ICBDA)*, pages 1–4, 2019. doi:10.1109/ICBDA47563.2019.8987140.

[38] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013.

[39] Gerhard Münz, Sa Li, and Georg Carle. Traffic anomaly detection using k-means clustering. 2007.

[40] Kamil Mysiak. Explaining DBSCAN Clustering — towardsdatascience.com. https://towardsdatascience.com/explaining-dbscan-clustering-18eaf5c83b31. [Accessed 02-09-2023].

[41] Sunit Prasad. Types of algorithms with different machine learning algorithm examples, Jun 2022.

[42] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Papers with Code - Language Models are Unsupervised Multitask Learners — paperswithcode.com. https://paperswithcode.com/paper/language-models-are-unsupervised-multitask, 2019. [Accessed 10-Jul-2023].

[43] Douglas Reynolds. *Gaussian Mixture Models*, pages 659–663. Springer US, Boston, MA, 2009. ISBN: 978-0-387-73003-5. doi:10.1007/978-0-387-73003-5_196.

[44] Mayra Z. Rodriguez, Cesar H. Comin, Dalcimar Casanova, Odemir M. Bruno, Diego R. Amancio, and Luciano da F. Costa. Clustering algorithms: A comparative approach. *PLOS ONE*, 14(1):e0210236, 2019.

[45] Luís Rosado, João Gonçalves, João Costa, David Ribeiro, and Filipe Soares. Supervised learning for out-of-stock detection in panoramas of retail shelves. pages 406–411, 10 2016. doi:10.1109/IST.2016.7738260.

[46] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20(1):53–65, 1987.

[47] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 3 edition, 2010.

[48] A. Shahriar, Mahedee Moon, Hasan Mahmud, and Md Kamrul Hasan. Online product recommendation system by using eye gaze data. pages 1–7, 01 2020. doi:10.1145/3377049.3377108.

[49] Priyanka Sharma, Deepesh Machiwal, and Madan Jha. *Overview, Current Status, and Future Prospect of Stochastic Time Series Modeling in Subsurface Hydrology*, pages 133–151. 05 2019. ISBN: 978-0-12-815413-7. doi:10.1016/B978-0-12-815413-7.00010-9.

[50] Rohit Sharma. What is clustering and different types of clustering methods.

[51] C. Shi, B. Wei, and S. et al Wei. A quantitative discriminant method of elbow point for the optimal number of clusters in clustering algorithm. 2020. doi:10.1186/s13638-021-01910-w.

[52] Rahul Shrivastava and Dilip Singh Sisodia. Product recommendations using textual similarity based learning models. In *2019 International Conference on Computer Communication and Informatics (ICCCI)*, pages 1–7, 2019. doi:10.1109/ICCCI.2019.8821893.

[53] Abe Vallerian Siswanto, Lilian Tjong, and Yordan Saputra. Simple vector representations of e-commerce products. In *2018 International Conference on Asian Language Processing (IALP)*, pages 368–372, 2018. doi:10.1109/IALP.2018.8629245.

[54] John Smith and Jane Doe. *Stochastic Modeling: A Thorough Guide to Evaluate, Pre-Process, Model and Compare Time Series with MATLAB Software*. Acme Publishing, New York, NY, 2022. ISBN: 978-1234567890.

[55] M A Syakur, B K Khotimah, E M S Rochman, and B D Satoto. Integration k-means clustering method and elbow method for identification of the best customer profile cluster. *IOP Conference Series: Materials Science and Engineering*, 336(1):012017, apr 2018. doi:10.1088/1757-899X/336/1/012017.

[56] Handy Tantyo and Tuga Mauritsius. Implementation of recommendation system in e-commerce using approximate nearest neighbor. *Journal of Theoretical and Applied Information Technology*, 100(12):3885 – 3893, 2022. Cited by: 0.

[57] Xiwang Yang, Yang Guo, Yong Liu, and Harald Steck. A survey of collaborative filtering based social recommender systems. *Computer Communications*, 41:1–10, 2014. ISSN: 0140-3664. doi:https://doi.org/10.1016/j.comcom.2013.06.009.