LUISS

Department of Business and Management

Course of Statistics for Marketing

The effects of recommendation systems on sales forecasting and customer engagement in the fashion world

Prof. Salate Santone Francesco

RELATORE

Prof. Paolanti Marina

CORRELATORE

Terruli Vincenzo - Matr. 747771

CANDIDATO

Anno Accademico 2022/2023

Alla mia famiglia

Contents

Chapter 1

Overview and Problem Definition

1.1 Problem definition

In the current digital age, the fashion industry has been profoundly affected by the expansion of e-commerce and the adoption of online sales platforms. Online buying and selling of products and services can is called e-commerce. It has transformed the buying process from traditional personal shopping to online shopping, enabling customers to have an enjoyable experience by providing services that help customers in the entire selling process[1]. In addition, numerous emerging technologies, such as chatbots, are being used to provide a better experience for customers shopping online on e-commerce platforms. E-commerce facilitates various forms of business transactions between companies and customers on the Internet, reducing costs and increasing the efficiency of business processes and the timing of business transactions.

The concept of *Customer Journey* represents one of the central pillars of contemporary marketing. This concept encompasses a wide range of elements, including omnichannel, customer experience, multichannel marketing and even extends to other disciplines such as consumer sociology, social psychology and business management, especially when a significant change in brand strategy is needed. In short, the customer journey refers to the path a consumer takes from the awareness of a need to the purchase of a product or service. However, it is important to consider this journey as the result of an evolving relationship between the customer and the brand, a story of two-way connection between the two parties[2]. This concept fits perfectly with that of omnichannel, which aims to engage the consumer across all available channels, touch points and purchase stages. Touch points, which can be physical or virtual, act as stages in this journey, and the transition from one channel to another plays a key role in the customer relationship. For companies, understanding the stages of this journey in detail is crucial for strategic and operational planning, enabling them to optimise customer satisfaction through rational budget utilisation[3]. In the past, the customer journey process was described in a

relatively simple and linear way, through the five steps of awareness, familiarity, consideration, purchase and loyalty. However, today this process has become extremely more complex and articulated, especially with the spread of the Internet, devices and multi-channel sales strategies. Tools and contact points constantly overlap and intersect, with examples such as online shopping followed by in-store pick-up. This integration of physical and digital touch points creates a unified and complex whole with an enormous amount of information[4]. A critical aspect in this process is the *messy middle* which is the confusing purchasing stages that occur between the first stimulus and the actual purchase[5]. The pandemic of COVID-19 has accelerated the transition to online shopping and online search worldwide, making it even more crucial for brands to understand how consumers make decisions in an online environment with many options and information available. According to Statista, the value of the global online market will reach 6 trillion by 2026 [6]. The fashion industry plays an important role, as the global fashion e-commerce market is expected to reach a value of more than €820 billion by the end of 2023, and by 2026 it could reach just over €1.2 trillion. Today, 21 percent of the entire fashion industry's revenue comes from online transactions[7].
Consumers today have access to a wide range of products, services and information. On the other, they face choice overload, where the multiplicity of options can lead to confusion and indecision[8]. Finding a balance between the mental effort required to make a decision and the reward obtained therefore becomes crucial. Cognitive biases play a significant role in purchasing decisions by influencing motivations when choosing a brand or product. This leads to an omnichannel approach that must be designed in a holistic and integrated manner, rather than as a sporadic process. Multi-disciplinarity is becoming increasingly crucial in marketing as the industry faces complex and large-scale challenges, including consumers' behavioural and cognitive decisions.

Technological breakthroughs have facilitated the transition from market-centered to user-centered commerce. The massive growth in the amount of digital content available online and the number of Internet visitors has created a potential information overload problem that prevents timely access to items of interest on the Internet. For businesses, the problem arises when the number of choices is overabundant and it is necessary to prioritize and provide relevant information to alleviate the information overload problem, which affects businesses as well consumers.
This change has created a significant challenge for online fashion sites: how can they offer a personalized experience to shoppers in a virtual environment, similar to that offered in physical stores? The answer to this question lies in implementing technologies that can replace existing traditional sales methods in stores. Technologies such as visual try-on allow customers to try on the products they wish to purchase by simulating the action that takes place in the fitting room. On the other hand, companies need technologies that simulate the work performed by

personal shoppers such as chat both and especially recommendation systems.

Recommender systems have become a key pillar of online marketing and e-commerce. Such systems play a key role in helping consumers discover new products, increase their engagement on the platform and, as a result, improve customer conversion and retention. In the context of the fashion industry, the evolution of e-commerce has given rise to new challenges and opportunities to offer a highly personalized and engaging shopping experience.[9] Recommender systems have emerged as a key tool to address these challenges, enabling online fashion sites to offer tailored product recommendations that reflect consumers' unique tastes and preferences. These systems play a key role in directing visitors to relevant products, helping to increase the discovery of new styles and trends and increasing the effectiveness of marketing and sales strategies. The peculiarity of recommendation systems in the fashion industry lies in the highly subjective nature of consumer preferences in terms of styles, colours, brands and materials. Addressing this complexity requires solutions that capture the essence of each individual's personal aesthetic, as well as considering the context in which a garment might be worn. To this end, traditional collaborative and content algorithm-based approaches can be enhanced by more advanced methodologies that also consider the semantic and visual context of products.
However, the design and implementation of recommendation systems for online fashion sites present unique challenges. The subjective nature of fashion preferences, along with the wide range of products and styles available, requires advanced approaches to effectively capture and utilize information about user behaviour. This is where the Alternating Least Squares (ALS) algorithm comes in, a machine learning methodology that has shown promise for solving recommendation problems.

1.2 Objectives

The main objective of this thesis is to develop and evaluate an innovative recommendation system for online fashion sites using the Alternating Least Squares (ALS) algorithm. Specifically, the research points to realize a customized implementation of the Alternating Least Squares algorithm adapted to the specific characteristics of the online fashion industry. This will require adapting utility matrices and user feedback data to the nature of fashion preferences, taking into account factors such as styles, brands, and product categories. The thesis aims to develop a recommendation system that generates highly relevant and personalized product suggestions, considering customers' implicit information (e.g., browsing and purchasing behaviour). To evaluate the effectiveness of the implemented recommendation system, several evaluation metrics, such as accuracy, coverage, and diversity of recommendations, are analysed.
Based on the results, the study seek to formulate practical recommendations for the implementation and continuous improvement of the recommendation system in the fashion context. These recommendations could include strategies to address specific challenges in the fashion industry and to adapt to changes in consumer preferences over time.

Through the achievement of these goals, this thesis want to contribute to knowledge and innovation in the application of recommendation algorithms in the online fashion industry, offering effective solutions to improve user experience and business performance. This research aims to make a significant contribution to the intersection of Marketing Analytics and the online fashion industry by providing an in-depth and innovative analysis of the application of recommendation algorithms in the fashion context. The growing adoption of e-commerce and the increased importance of personalised shopping experiences have highlighted the need to develop advanced solutions that meet the needs of an increasingly demanding and diverse audience.
The value of this research lies in the integration of skills and knowledge from different disciplines, including machine learning, marketing and data analysis. The online fashion industry presents unique challenges, such as interpreting subjective consumer preferences and understanding the evolution of trends and tendencies. Furthermore, the research not only makes a technical contribution through the design and implementation of the recommendation system, but also has practical relevance for the fashion industry and the e-commerce sector. The results obtained can be used by companies to optimise marketing strategies, improve sales conversion and build long-term customer relationships.
The ability to deliver highly personalised product recommendations not only increases user engagement on the site but can also lead to greater customer satisfaction.

Chapter 2

Recommendation System and ALS

2.1 Recommendation System

The recommendation system is defined as a decision-making strategy for users in intricate information systems. A recommender system calculates and provides relevant content to the user based on knowledge of the user, content, and interactions between the user and the item.[1] From the perspective of e-commerce, it is defined as a tool that helps users search for knowledge records that are related to users' interests and preferences. The recommendation system has been also defined as a means to support and improve the process of using the recommendations of others to make choices when personal knowledge or experience of alternatives is scarce.[2]

A recommendation system is an enhanced technology that has revolutionised the way companies interact with consumers in today's digital landscape. In a world where the abundance of information and options often makes it difficult for users to find relevant content and products, recommender systems emerge as powerful tools to simplify this complexity. In essence, a recommender system is a set of algorithms and processes that analyse user data and content to provide personalised and relevant suggestions. These suggestions can cover a wide range of items, from products in an online shop to songs to listen to, from films to watch to articles to read. The main objective of a recommendation system is to facilitate users' decision-making process, enabling them to discover new content and products in line with their preferences and interests.

The evolution of data analysis and artificial intelligence has greatly improved the effectiveness of recommendation systems. These systems use historical data and complex algorithms to

identify hidden patterns in user behaviour and generate personalised recommendations. This not only improves the user experience but can also have a positive impact on companies' business strategies, increasing user engagement, conversions and retention.To achieve the desired results, companies should consider many factors before implementing a recommendation system, including:[1]

- The ***Domain*** where you operate as it is important to understand the type of content you want to recommend which may vary from the type of sector you operate in. The domain is important to understand what you want to do with the recommendation but above all it indicates how serious an error in the recommendation is, so an error in the field of medicine can affect you much more than in music for example.
- The ***Purpose*** of the system. It first needs to be understood whether the recommendation is intended for both the end user and the service provider. For end-users, the use of recommendations within a system aims to simplify the discovery of relevant content or products, tailored to their preferences and needs at a given time. Without any selection mechanism, it would be impossible to deal with an enormous amount of options. From the provider's point of view, the objective is basically to encourage interaction and continued use, ensuring that users find value in the content or services offered.
- The ***Context*** is the environment in which the user receives a recommendation. The context could be the device the client is using to view the recommendation or the current location of the receiver, the time of day and the activity the user is performing. Context can also be very personal and include factors such as the climate or even the user's mood. In particular, it is crucial to understand whether the user has time to reason about the recommended service or product or will make an impulse choice.

In their studies Khusro et al. [3] define recommender system as an utility function used to measure the usefulness of a particular item, to specific user. Mathematically it can be shown as:

$$F : U * S \rightarrow R$$

where S is the set of all those that could be recommended to users and U is the set of all users available in the system to recommend. F is the utility function used to measure the usefulness of a particular item u e U, to specific user s e S. R shows the total recommended set. For every user, the system recommends such an item s that maximize F for the user, which is generally denoted by rating provided by user. [3]

2.1.1 Level of Personalization

Based on the method used to derive and make a recommendation, we can have different levels of personalisation.

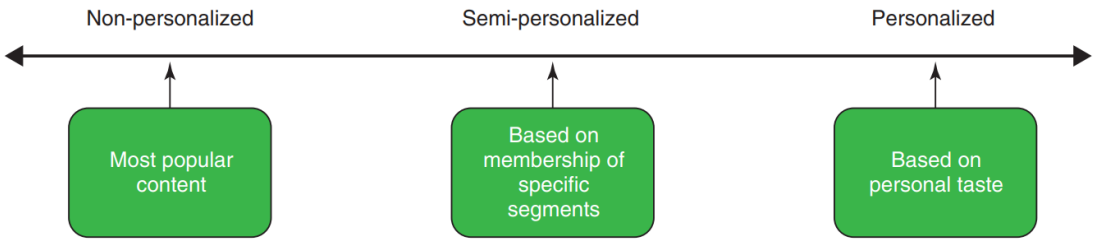

Figure 2.1: Level of personalization

- ***Non-personalised recommendations*** are defined as those that do not use any customer data and are the same for all users accessing the system. Examples of Non-personalised recommendations are the list of best-selling items, as well as items sorted by date.
- ***Semi-personalised recommendations*** refers to when the recommendation is made for user segments, so we find different recommendations according to the user cluster.
- Finally, ***Personalised recommendations*** are those that use the user's interaction data with the system and generate highly personalised recommendations for each user.

Recommendations can often be generated and derived from being human. In many areas it is preferable to use recommendations made by experts and specialists, especially for those areas where accurate product knowledge requires study and experience. One example is the world of wine where many websites offer advice and opinions provided by experts and sommeliers.
In general, recommendation systems do not relies only on one tiers of personalization but we can observe that the majority of systems amalgamate the first two tiers of personalization to engender personalized recommendations. Often, within the confines of a single website, several levels of personalisation are used simultaneously.
Netflix is an example of such fusion; the phenomenon can be observed on the Netflix homepage, where recommended films are curated according to the user's historical viewing patterns within the platform, as well as broader categorisation including genres and frequency.
Another large platform such as Amazon uses multiple levels of customisation combined. The section showing 'best-selling items' is accompanied by a segment showing 'customers who bought this product also bought this list of items', here we have both the non-personalised and the semi-personalised tier.
The level of personalization a website offers depends on the volume of user traffic it attracts.

As the number of users visiting and engaging with a specific site increases, the algorithm gains the capability to work with a larger dataset, resulting in the delivery of more sophisticated personalized recommendations.

2.1.2 Input and types of Feedback

Recommender systems require a structured and functional user profile in order to derive well accurate recommendations. The collection of relevant user information such as characteristics, habits and behaviour are essential for any recommender system as its performance results from its ability to represent the user's current interests.
There are different types of data related to the user's interaction with the system:[2]

- ***Explicit feedback*** are those ratings provided by users through the system interface. Usually recommendation system works much better when the quantity of rating is high. These types of feedback require more effort from the user in terms of time and most customers have no interest in rating an item they have already purchased. Furthermore, it has been noted that users are more likely to leave a review if their experience with the product or service was negative. Despite this, recommendation systems based on explicit feedback are more truthful and trustworthy as the final output is more transparent and does not require the extraction of information from shares.

- ***Implicit Feedback*** kicks in when the system does not have data about the user's experience with the product or service. This feedback is the result of the user's interaction with the system and is derived from purchase history, time spent web browsing, page clicks as well as social interactions. In contrast to explicit feedback, this does not require any effort on the part of the user but is consequently less accurate but more objective.

- ***Hybrid feedback*** is the result of combining implicit and explicit feedback. Recommender systems can have both types of feedback and combine them in order to counteract the weaknesses of both and maximise performance.

Explicit feedback refers to direct ratings or reviews provided by users, such as stars given to a product or written reviews. These explicit feedbacks tend to offer an accurate measure of user preferences, allowing recommendation systems to provide highly targeted recommendations. However, their abundance may be limited since not all users leave reviews or assign ratings to every item. On the other hand, implicit feedback is based on user behaviour and actions, such as views, clicks, purchases or time spent on a page. This feedback is often more abundant as it is generated passively by users during their interaction with a platform. However, they may be less precise in expressing user preferences than explicit feedback. For instance, a user might view a page without actually being interested in the content.

Both categories of feedback can be context-sensitive, considering the time at which they are generated. This allows recommendation systems to adapt to the changing needs of users. Another distinction between the two types of feedback concerns the expressiveness of user preferences. In implicit feedback, we can often only identify positive user preferences, i.e. what users seem to like. On the other hand, in explicit feedback, we can get a more complete view of users' preferences, including things they might not like. This makes explicit feedback more appropriate for addressing positive/negative challenges in recommendations. Finally, the way feedback is measured can vary. In implicit feedback, measurement is often relative, as what matters is the relationship between objects and not absolute values. In explicit feedback, measurement is more often absolute, with numerical ratings or written reviews[4]. Item ratings are usually represented by a rating scale indicating the user's liking of the article.

The user profile is built and updated with personal information that can be biographical data or browsing history, such as articles visited, rated or purchased. At the same time, each article has its own characteristics and specifications that are updated implicitly or explicitly when users make an interaction. Despite the diverse and numerous possibilities of acquiring information, and with the huge amount of articles and information, it is possible that some articles are not rated at all.

Consequently, a recommender system must use proper filtering methods so that these items are rated and evaluated implicitly or explicitly and recommended accordingly.[5]

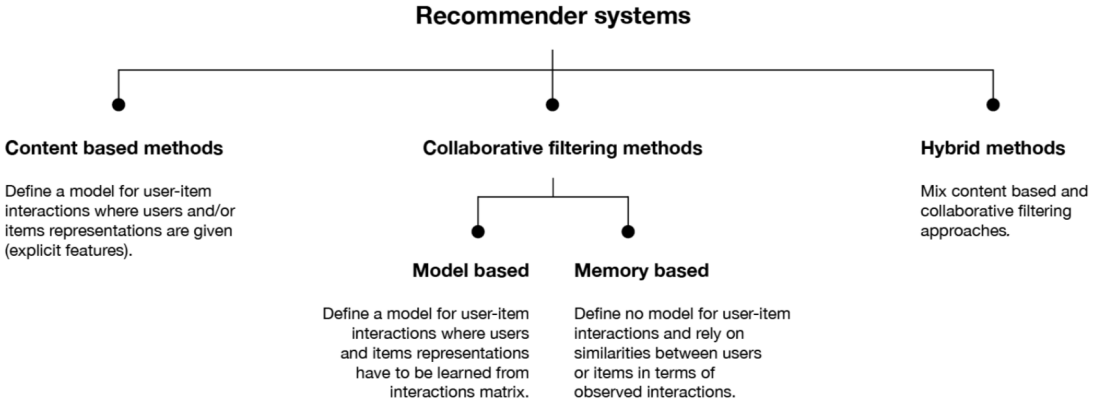

Figure 2.2: Types of Recommender system

2.1.3 Collaborative filtering

Collaborative recommendation systems are based on the principle of using the preferences of similar users to suggest products or articles that might be of interest to the current user. These

systems require the analysis of preferences and ratings provided by other users with similar tastes in the past, but who already have experience with products or items unknown to the current user. They collect users' opinions and then make recommendations based on the similarity of the ratings given by people. Users with similar opinions become co-workers in this process. Collaborative recommendation systems mainly rely on the details of several common products or articles to define users and influence the outcome.
These techniques work mainly on large amounts of data available online. Typical collaborative techniques include methods based on models, items and users. These methods, together with the analysis of participants' preferences, enable systems to generate effective and relevant recommendations for users, thus helping to simplify choice and improve the overall experience.[5]

User-based Collaborative Filtering is a recommendation approach that is based on user preferences and behaviour. The method uses the weighted sum of the average ratings of users who have rated this product or item in the past and the average user rating, these weights are defined as the similarities of these users with the target item or product. In this method, other users with similar tastes to the current user are identified.
Recommendations are generated using the preferences of these similar users. For example, if User A has positively rated a series of products that were also liked by User B, the system can suggest products to A that B has liked in the past. This method is based on the assumption that users with similar preferences will tend to like similar products. User-based Collaborative Filtering uses Pearson correlation algorithm to measure similarities of two variables or attributes and identifies similar users by finding similarity measures.

Item-based Collaborative Filtering focuses on calculating the similarity of items or products themselves instead of users. In this approach, recommendations are generated by identifying products that are similar to those that the user has already proven to like.
If a user has rated a particular product positively, the system will search for other products that have been rated positively by other users who have liked the same item. This method exploits the relationships between the items themselves to suggest related products.
Different methods are available to calculate the similarity of similar products or items. These are: cosine based similarity where two similar items are considered as two vectors and similarities between these products or items are calculated using cosine angle value of the position between these two variables or vectors; correlation-based similarity in which similarity is finding with Pearson correlation between two items; adjusted-cosine similarity finding similarities using simple cosine.

Model-based Collaborative Filtering relies on mathematical models or machine learning algorithms to create representations of user and item preference data. These models can

capture complex relationships between users and products, using techniques such as matrix decomposition and clustering. Once the model is trained, it can make recommendation predictions for new users and products based on the information learnt during the training phase. These methods are used to provide quick recommendations to a cluster or group of products or items using the pre-filled recommendation model. In this way, the recommendation is similar to recommendation techniques based on nearest neighbours. This approach can better handle situations with missing data or noise, as it is based on statistical models that can generalise the underlying relationships.
Examples of these techniques include Dimensionality Reduction technique such as Singular Value Decomposition (SVD), Matrix Completion Technique and Clustering.

In a nutshell, User-based Collaborative Filtering is based on user preferences, Item-based Collaborative Filtering focuses on the items themselves, while Model-based Collaborative Filtering uses mathematical models or machine learning algorithms to make recommendations. Each approach has its advantages and limitations, and the choice depends on the nature of the data and the specific objective of the recommendation system.

2.1.4 Challenge of Collaborative filtering

Collaborative Filtering is effective in areas where items have little associated content and where the content is difficult for a computer system to analyse, such as opinions or reviews. This technique can provide unexpected recommendations, suggesting items that are relevant to the user even when the content is not present in the user's profile.
However, despite its advantages, the use of Collaborative Filtering can present significant challenges that limit its applicability and accuracy.[2]

Cold Start is one of the main challenges, which occurs when new users or new articles are entered into the system. Since the collaborative filter is based on the gathering of historical preference data, it can be problematic to make accurate recommendations for articles or users who have few or no ratings in the dataset and whose tastes are not known to the system. This may lead to irrelevant results or even no recommendations at all.

Data Scarcity is observed when most of the possible interactions between users and items have not actually been recorded. This makes it difficult to calculate accurate relationships between users or items based only on available preferences, when only a few of the total items available in a database are evaluated by users. This always leads to a sparse user-item matrix, the inability to identify successful neighbours. Furthermore, data sparsity always leads to coverage problems which refers to the percentage of system elements for which recommendations can be made. Scarcity of data can lead to inaccurate results or poorly tailored recommendations.

Scalability is a Collaborative Filtering challenge, especially when the dataset grows considerably. Computing similarities between a large number of users or items may require considerable computational resources, slowing down the recommendation generation process and increasing the complexity of the system. This problem is intrinsic to recommendation algorithms, as the computation usually grows linearly with the increase in the number of users and elements. A recommendation technique that works efficiently with limited datasets may fail to maintain a satisfactory level of performance as the datasets become more voluminous.
Therefore, it becomes essential to employ recommendation approaches that can scale efficiently as the number of data items in the database increases. To solve the scalability challenge and improve the speed of recommendation generation, dimensionality reduction techniques are often employed, such as the Singular Value Decomposition (SVD) method, which demonstrates the ability to generate reliable and efficient recommendations even in large contexts.

Preference conformity is another known problem, where a user can be influenced by the ratings or preferences of other users without really considering their own personal tastes. This can lead to a kind of bandwagon effect, where users end up following the most popular preferences without expressing their own personal choices. This problem is most visible in systems where there are many items and some of them are popular. Furthermore, users may have a temporal variation of preferences, where users' preferences may change over time. Recommendations based on past preferences may no longer be relevant due to changing user tastes or needs.

Synonymy is a challenge for many recommendation systems. Especially in the case of large systems with thousands of products in the catalogue, items can have very similar names that are not recognisable by the system. Approaches like automatic term expansion, thesaurus construction and Singular Value Decomposition (SVD) can solve the problem of synonymy.

2.1.5 Content-based filtering

Content-based filtering operates by utilizing the data that users provide, whether in an explicit or implicit manner. This data contributes to the creation of user-related information, which in turn fuels the recommendations made to the user. As users contribute more ratings, inputs, or responses to recommendations, the effectiveness and accuracy of the recommender system improve. Content-based recommender systems focus on suggesting products or items based on the user's past likes and dislikes.
The techniques employed in content-based filtering rely on a comparison between the detailed attributes of products and the user's profile. The content of each product or item is defined by a set of terms or attribute details, often the characteristics present in a document. User profiles are generated, representing these similar attributes and derived from analyzing the

attributes of products that the user has rated. Developing a content-based system presents several challenges. Attributes can be provided either manually or automatically. When using automatic methods, a chosen technique extracts attributes from various items, which are then employed to represent user profiles and items for further processing. A learning algorithm is used for training purposes, learning user information based on observed products or items and generating recommendations based on the user's profile.

In practice, a content-based recommender system suggests items that share features similar to those the user has previously liked. This approach involves creating a user profile based on feedback and ratings, comparing this profile with item features, and then recommending matched items. One illustrative instance is LIBRA, which suggests books based on descriptions extracted from Amazon.com webpages. It learns user profiles and recommends books using user ratings alongside the features extracted from these webpages.
In content-based approaches, diverse techniques are employed to match the features of new items with user profiles and determine whether a particular item aligns with the user's interests. User profiles are updated either implicitly or explicitly, while utilities are assigned to items based on previous utility assignments to observed items. Item profiles are represented by the features and descriptions of the items using Boolean values or integer values reflecting term frequency, or term frequency-inverse document frequency. Items are categorized as relevant or non-relevant through a comparison of item representations with user interest and preference representations, leveraging keyword matching, nearest neighbor approaches, cosine similarity, and typical classification methodologies.[3]

2.1.6 Pros and Cons of Content-based Filtering

This method is particularly effective in addressing the cold start problem, where new users or items lack sufficient historical data for recommendations. The system can make suggestions based solely on the content attributes, enabling personalized recommendations even for new users. Furthermore, content-based filtering can offer diverse recommendations, accommodating individual tastes that might not match the majority. However, this method has limitations. They struggle to capture complex behaviors or inter-dependencies. For instance, a user might favor items on Data Mining only when they include practical applications along with theoretical content, rather than just theory alone. This kind of nuanced information cannot be created by content-based recommender systems.

Content-based approaches heavily depends on the quality of content attributes and the system's ability to accurately capture user preferences from these attributes. It can create a "filter bubble" where users are repeatedly recommended similar items, potentially limiting their ex-

posure to new interests. Additionally, content-based filtering struggles to capture complex user preferences that go beyond straightforward attributes. For instance, if the user's interests are dynamic or influenced by contextual factors, content-based filtering might not adapt well. In essence, content-based filtering excels at providing personalized suggestions based on item characteristics, but it can fall short in addressing user novelty and capturing intricate, evolving preferences.

2.1.7 Hybrid Filtering

Hybrid filtering techniques represent a strategic fusion of different recommendation approaches, strategically engineered to enhance system optimization and circumvent the limitations inherent in pure recommendation systems. The core concept underlying hybrid techniques is that a harmonious blend of various algorithms yields recommendations that are both more accurate and effective compared to those derived from a single algorithm. This combination leverages the strengths of one algorithm to counteract the weaknesses of another, ensuring a more robust recommendation system. By employing multiple recommendation techniques, it becomes possible to mitigate the inherent shortcomings of each technique within a unified model. The combination of approaches can be executed in various ways, such as independently implementing algorithms and subsequently combining their results, introducing elements of content-based filtering into collaborative approaches, incorporating aspects of collaborative filtering into content-based methods, or crafting a holistic recommendation system that seamlessly integrates both content-based and collaborative approaches. Hybrid filtering thus serves as a powerful strategy to optimize recommendation systems, delivering more precise and adaptable suggestions to users, and enhancing the overall user experience.
Hybrid recommendation systems encompass various categories, each designed to leverage multiple recommendation techniques for enhanced performance:

- ***Weighted Hybrid*** approach stands out for its ability to blend results from different recommenders cohesively, resulting in a recommendation list or prediction. By employing a linear formula to integrate scores from each technique, weighted hybridization ensures that the strengths of all recommendation systems are optimally utilized. This approach is particularly advantageous because it provides a clear and direct means of combining various algorithms. It accounts for the unique capabilities of each technique, assigning different weights based on their relative performance. This adaptability allows the system to produce more accurate and personalized recommendations by dynamically adjusting the influence of each recommender based on its effectiveness.

- ***Switching Hybrid*** is adept at mitigating the limitations inherent in individual recommendation methods. For example, it can address the "new user problem" common in

content-based recommenders by seamlessly transitioning to a collaborative recommendation system when required. This transition is governed by a heuristic criterion reflecting the recommender's competency in producing reliable ratings. One key advantage of this approach is its sensitivity to the strengths and weaknesses of each constituent recommender. By employing this switching mechanism, the system can make recommendations that adapt to changing user profiles and preferences. However, it's important to note that this strategy may introduce increased complexity to the recommendation process due to the necessity of managing the switching criterion, which adds parameters to the recommendation system.

- ***Cascade Hybridization*** technique introduces an iterative refinement process to construct a preference order among different items. This method begins with one recommendation technique generating a preliminary, coarse list of recommendations. Subsequently, another recommendation technique fine-tunes these initial recommendations. The cascade hybridization approach is highly efficient and noise-tolerant, thanks to its iterative nature. By progressing from a coarse to a finer level of recommendation refinement, it ensures that recommendations are not only accurate but also progressively improved. This iterative refinement can enhance the precision and relevance of recommendations, particularly in situations where the initial recommendations may be imprecise or incomplete.

- ***Mixed Hybrid*** combines different recommendation techniques simultaneously for each item, rather than presenting a single recommendation per item. As a result, each item may have multiple recommendations associated with it from various recommendation techniques. This approach offers versatility and diversification in recommendation generation. It acknowledges that different techniques excel in various contexts and with different user preferences. By presenting users with multiple recommendations per item, the system allows for a more comprehensive and adaptable user experience. This approach is especially beneficial when user preferences are diverse or when items have multifaceted characteristics that are best captured by different recommendation methods.

- ***Feature-Combination*** entails taking the features generated by one recommendation technique and feeding them into another recommendation technique. For instance, the ratings of similar users, which are a feature of collaborative filtering, can be incorporated into a case-based reasoning recommendation technique as one of the features to determine the similarity between items. This technique offers several advantages. It doesn't solely rely on collaborative data, diversifying the information used for recommendations. By integrating features from different techniques, it leverages the unique strengths of each approach, potentially leading to more robust and nuanced recommendations. Additionally, it enhances recommendation accuracy by considering additional dimensions beyond

ratings.

- ***Feature-Augmentation*** goes a step further by utilizing ratings and other information generated by the initial recommender while also requiring additional functionality from the recommender systems. In feature-augmentation hybrids, a small number of new features are introduced to complement the primary recommender's output. These added features can include item attributes, contextual information, or user-specific data. The benefit of this approach is that it enriches the recommendation process by incorporating supplementary information, enabling the system to make more informed and context-aware suggestions. Feature-augmentation methods are particularly effective when the primary recommender's output is limited in terms of information richness.

- ***Meta-Level Models*** generate one recommendation that serves as input for another. These meta-level models are inherently richer in information compared to a single rating. Meta-level hybrids tackle the sparsity problem often encountered in collaborative filtering techniques. They do this by using the comprehensive model learned by the first technique as input for the second technique. This method enhances recommendation quality by considering not only ratings but also patterns, correlations, and latent structures present in the data. It is especially advantageous in scenarios where explicit ratings data is scarce or incomplete, as it leverages the holistic understanding of user preferences provided by the initial technique to improve recommendation accuracy.

In conclusion, hybrid recommendation systems strategically combine multiple recommendation techniques to optimize the recommendation process. Each hybrid category offers a unique way to harness the strengths of diverse methods and adapt to the challenges presented by users, items, and contextual factors. By embracing these hybridization strategies, recommendation systems can deliver more accurate, personalized, and context-aware recommendations to users.

2.1.8 Evaluation Metrics

Recommender systems have as primary objectives user satisfaction and profit maximization. The first goal is to enhance user satisfaction by providing recommendations tailored to their preferences. Simultaneously, these systems aim to maximize revenue by encouraging users to engage with the recommended content, potentially resulting in purchases or other revenue-generating actions. However, the emphasis here is on ensuring customer happiness to achieve profitability. Another objective is profit maximization with customer satisfaction as a byproduct. In this scenario, customer satisfaction is viewed as a means to an end – a way to drive revenue. The recommendation system focuses on revenue generation by promoting items or content that are likely to result in transactions or interactions that directly benefit the platform.

The effectiveness of a recommendation system in understanding user preferences can be assessed through various metrics. One common approach involves evaluating the system's ability to predict whether a user will like an item they are already familiar with and have rated. This prediction accuracy is often measured by how closely the system's rating predictions align with the user's actual ratings. Additionally, decision-support metrics categorize predictions into groups based on user reactions. For instance, if the system predicts a rating above a certain threshold, it suggests that the user would likely be interested in the item, while lower predictions indicate less interest.

A critical concern in recommendation systems is the overemphasis on popular items, which can lead to what is commonly referred to as a "filter bubble." Popular items are frequently recommended, making them even more popular, while lesser-known but potentially excellent items are overlooked. This narrowing of recommendations can limit user discovery of novel content. Moreover, it contradicts the fundamental purpose of recommendation systems – helping users navigate extensive catalogues beyond what is visible on a single page. To combat this, diversity and coverage metrics are employed. Diversity measures aim to ensure that recommendations span a wide range of content, thereby reducing the filter bubble effect. Coverage encompasses content coverage, ensuring that the entire catalogue is recommended, and user coverage, ensuring that recommendations reach all registered users. User coverage is defined as follows:

$$Coverage_{user} = \frac{\sum u \epsilon U^P u}{|U|}$$

where P is 1 if the number of recommendations the system returns for user u is >0, otherwise it is 0. |U| represents all users.

Using the result of the method execution, you can also calculate the catalogue coverage, which is defined by:

$$Coverage_{catalogue} = \frac{|all\ item\ in\ recs|}{|I|}$$

where |I| is the number of items in the catalogue

Serendipity is another crucial aspect in recommendation systems. It embodies the notion of pleasantly surprising users with unexpected but highly appealing recommendations, thereby enhancing the user experience. Achieving serendipity is challenging, as it involves delivering recommendations that deviate from a user's known preferences. Ensuring serendipity requires balancing personalized recommendations with the introduction of diverse, potentially unexplored content. While quantifying serendipity is complex, it remains an important objective in recommendation systems, emphasizing the need to avoid overly constraining the system's out-

puts, as constraints can diminish the potential for serendipitous discoveries in the user's journey.

The assessment of recommendation algorithms hinges on the application of various metrics, often centered on accuracy and coverage. The choice of metrics is contingent upon the specific filtering technique employed. "Accuracy" signifies the proportion of correct recommendations relative to the total number of possible recommendations, while "coverage" denotes the extent to which the system can provide recommendations across the entire spectrum of items within the search space.
A fundamental objective of recommendation systems is to introduce users to a selection of new items that resonate with their interests and needs. A prevalent method for assessing the algorithm's proficiency is to evaluate its capability to predict user ratings accurately. This assessment often involves withholding a portion of high-rated items as a benchmark to evaluate the recommendation system's predictive accuracy. Such testing procedures facilitate the gauging of how effectively the system understands user preferences and can generate recommendations aligned with those preferences.
Accuracy metrics for evaluating recommendation filtering systems are categorized into two primary groups: statistical accuracy metrics and decision support accuracy metrics. The selection of a suitable metric depends on the characteristics of the dataset under consideration and the nature of tasks performed by the recommendation system. These metrics are instrumental in quantifying the system's ability to deliver recommendations that align with user preferences, thereby enhancing user satisfaction.

Statistical Accuracy Metrics serve as essential tools to assess the accuracy of recommendation algorithms e by comparing the predicted ratings directly with the actual user rating.

Mean Average Error (MAE) quantifies the average magnitude of the errors between predicted and actual ratings or preferences given by users. It measures the absolute difference between the predicted and true values, making it less sensitive to extreme outliers

$$MAE = \frac{1}{N} \sum_{u,i} |p_{u,i} - r_{u,i}|$$

where $p_{u,i}$ is the predicted rating for user u on item i, $r_{u,i}$ is the actual rating and N is the total number of ratings on the item set.
Lower MAE values indicate better predictive accuracy, with a smaller error signifying a closer alignment between the recommender system's predictions and the user's actual preferences.

Root Mean Squared Error (RMSE) is an extension of MSE and is widely used for recommendation system evaluation. It involves taking the square root of the average squared

differences between predicted and actual values.

$$RMSE = \sqrt{\frac{1}{N}\sum_{u,i}(p_{u,i} - r_{u,i})^2}$$

RMSE offers several advantages, such as being in the same unit as the original ratings, which facilitates interpretation. Like MSE, lower RMSE values signify superior prediction accuracy.

In evaluating recommendation systems, these metrics share a common thread—they aggregate the collective error, whether squared or not. When assessing the performance of such systems, the presence of users who have contributed numerous ratings eases the task of predicting ratings. Conversely, users with limited ratings pose a greater challenge. The Root Mean Squared Error (RMSE) assigns substantial penalties to significant errors, magnifying the impact of a single substantial discrepancy in a prediction, while multiple smaller errors have a comparatively reduced influence. In contrast, the Mean Absolute Error (MAE) mitigates the influence of large errors or outliers, thereby preventing them from exerting undue pressure on the overall error calculation. The choice between RMSE and MAE hinges on the specific objective: if the priority is to ensure that none of the users receive poor recommendations, RMSE is the preferred metric. However, if the understanding is that achieving universal user satisfaction may be unattainable, MAE may suffice as a more balanced evaluation metric. This choice is influenced by the broader goals of the recommendation system and the level of importance placed on user satisfaction.

Decision Support Accuracy Metrics These evaluation metrics play a crucial role in aiding users to discern items of exceptional quality from the available pool of items.
The concept of decision support revolves around scrutinizing each recommendation individually to ascertain its correctness. When examining a recommendation system, we can analyze each recommended item in relation to the relevance for the user, leading to four possible outcomes for a given item. Specifically, an item may be recommended or not, and the user may consider it relevant or not.
The resulting outcomes can be categorized as follows:

- ***True Positive (TP)***: This designation is assigned when the system recommends an item that is relevant for consumer.
- ***False Positive (FP)***: This label is applied when the system recommends an item, but the user do not consider it relevant.
- ***False Negative (FN)***: when the system does not include an item in its recommendations, but it is still relevant for user.

- ***True Negative (TN)***: This category encompasses situations where the item was neither recommended by the system nor relevant for the user.

This outcomes are used to calculate two important metrics that are ***Precision*** that is the fraction of recommended items that is actually relevant to the user and ***Recall*** that can be defined as the fraction of relevant items that are also part of the set of recommended items.

$$Precision = \frac{TP}{TP + FP}$$
$$Recall = \frac{TP}{TP + FN}$$

Frequently, a recommendation system is structured to ensure that users are presented with at least one viable option for their subsequent purchase or viewing selection. Maintaining the presence of at least one pertinent item in a Top-N recommendation is of paramount significance. The objectives aligned with business imperatives are typically intertwined with the collective set of predictions, rather than being evaluated in isolation. Hence, it becomes imperative to delve into the analysis of two metrics that specifically address Top-N recommendations: Precision at K and Recall at K.

$$P@k = \frac{tot\ relevant\ items\ TopK\ position}{K}$$

Precision at K calculates the precision over the top k items, similar to normal precision, but it takes in consideration the order of items. This formula allows to be more flexible to vary the K and adjust the precision score.

$$R@k = \frac{tot\ relevant\ items\ TopK\ position}{tot\ relevant\ items}$$

Recall at k calculates the fraction of relevant items that are also part of the top k recommended items, it is important in case where there are few relevant items and the system have to include them.[6]

Average Precision (AP), which is a score that reflects the precision at each position in the recommendation list, considering both relevant and irrelevant items.

$$AP = \frac{\sum_1^k P@K}{k}$$

Where k is the number of items in the recommended list. Simply it sums all the precision for each item recommended and than divided by k. It quantifies how well the algorithm ranks the most relevant items higher. AP works per recommendation, while *Mean Average Precision at K* works for recommender system.

Mean Average Precision at K (MAP@k) is a widely used evaluation metric for recommendation systems. It assesses the quality of ranked recommendations by considering both their

relevance and their position in the recommendation list. MAP@K is a comprehensive metric for evaluating recommendation quality that consider both relevance and ranking position. It provides valuable insights into how well a recommendation system performs in delivering personalized and relevant content to users.

$$MAP@k = \frac{\sum_1^N AP@K}{N}$$

It take the average of the AP scores across all users to obtain the MAP@K score. This metric provides a single number that summarizes the overall quality of recommendations across the entire user base. MAP@K considers the entire ranked list of recommendations, accounting for the possibility of multiple relevant items. It rewards diverse and personalized recommendations. It focuses on user satisfaction by measuring the quality of recommendations based on the user's perspective. A higher MAP@K score indicates that users are more likely to find relevant items at the top of their recommendation lists. Higher MAP@K scores signify better recommendations, making it a valuable tool for fine-tuning and comparing recommendation algorithms.

Discounted Cumulative Gain (DCG) is a measure of ranking quality for a given query. The concept of DCG may seem intricate to articulate, yet its essence is readily comprehensible. It revolves around the identification of each relevant item and subsequently applying penalties based on its position within the list.

$$DCG = \sum_{i=1}^{K} \frac{2^{rel[i]} - 1}{\log_2([i] + 2)}$$

To compute DCG, each item is assigned a relevancy score, which can be represented as the predicted rating or profit in the context of a recommender system. In this context, the term "relevance" is interchangeably referred to as "gain." Alternatively, it can be conceived as the "discounted cumulative relevance," where the relevance scores are weighted according to their position in the list and then cumulatively aggregated to yield the Discounted Cumulative Gain.

Normalized Discounted Cumulative Gain (nDCG) represents a prevalent metric within the realm of information retrieval, particularly in evaluating the effectiveness of recommendation systems. This metric is harnessed to compute an aggregate score for a sequentially arranged collection of items. The 'discounted' facet of this metric introduces penalties for items of elevated relevance that find themselves positioned lower in the ranking. To derive nDCG, one computes the DCG up to a specific rank denoted as 'p' and subsequently normalizes it by dividing it by the maximum attainable DCG achievable through list sorting. This sorting operation entails arranging the list according to the relevance scores, wherein the most pertinent items occupy the uppermost positions, thereby yielding the highest attainable DCG value and

serving as the benchmark for normalization.

$$nDCG = \frac{DCG}{IDCG}$$

IDCG is the ideal or the highest possible DCG till the rank k

The ***Receiver Operating Characteristic (ROC) Curve*** is a graphical representation widely employed in the evaluation of recommendation systems. It serves as a valuable tool for assessing the system's ability to discriminate between relevant and irrelevant recommendations.

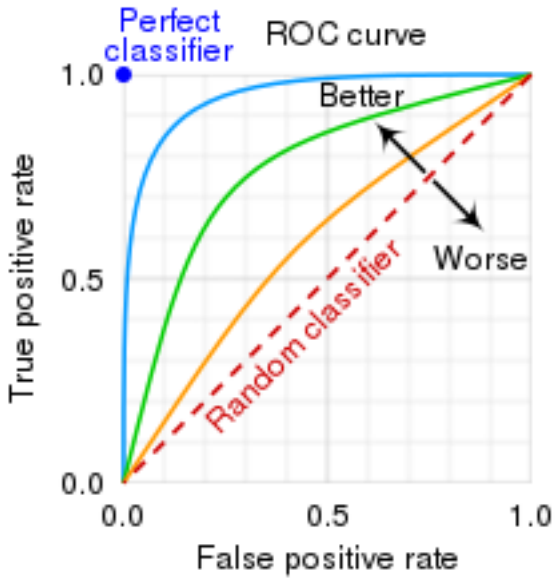

Figure 2.3: ROC Curve

The ROC curve visually depicts the trade-off between the true positive rate (sensitivity) and the false positive rate (1-specificity) across various decision thresholds. In the context of recommendation systems, it enables a nuanced examination of the system's performance in distinguishing items that users find valuable from those they do not. The ROC curve's utility lies in its capacity to reveal the system's discriminative power and to assist in optimizing the recommendation algorithm's parameters, ultimately enhancing the precision of the recommendations provided to users.

2.1.9 Marketing Application

In today's world, recommendation systems play a key role in the ecosystem of many companies and online platforms. Thanks to sophisticated algorithms and the analysis of user data, these systems contribute significantly to improving the user experience, increasing customer satisfaction and even boosting revenue. In this context, leading companies such as Amazon, Netflix

and Spotify actively use recommendation systems to suggest products, films or music tracks in line with users' tastes and preferences. These industry giants are driven by the need to provide a personalised and highly relevant service, underlining how the application of advanced recommendation techniques has become a key pillar of business success in the modern world. The following analysis delves into the strategies and technologies behind these systems, examining noteworthy case studies.

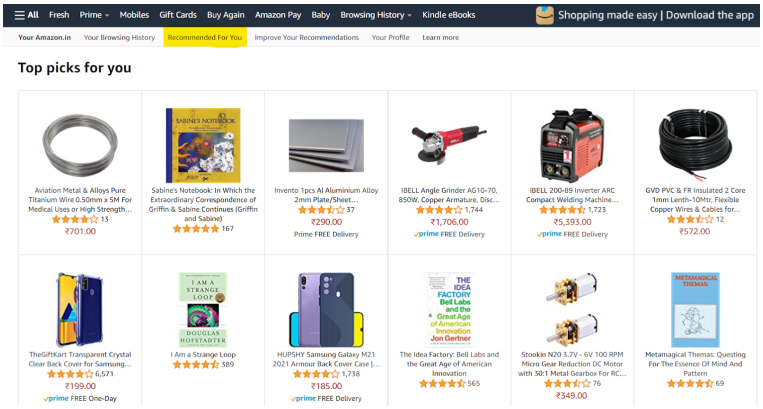

Figure 2.4: Amazon "Top picks for you" feature

Amazon.com serves as a prominent illustration of an e-commerce platform employing scalable item-to-item collaborative filtering techniques to offer personalized product recommendations to diverse users. As reported by McKinsey, an impressive 35% of Amazon's sales can be attributed to the efficacy of its recommendation systems. Notably, the computational algorithm scales seamlessly regardless of the size of the user and item database.[7] Amazon.com employs an explicit information gathering approach, encompassing sections such as 'your browsing history,' 'rate these items,' 'improve your recommendations,' and 'your profile.' The system anticipates user preferences based on their item ratings, subsequently scrutinizing their browsing patterns to discern relevant items for recommendation. Additionally, Amazon.com popularized the feature 'people who bought this item also bought these items.

Netflix, another data-driven entity, harnesses recommendation systems to enhance customer satisfaction. As underscored by the aforementioned McKinsey study, a substantial 75% of Netflix's viewership results from the influence of recommendation algorithms.[7] In fact, Netflix's commitment to delivering top-tier user experiences culminated in the Netflix Prize, a data science competition offering a lucrative $1,000,000 reward for the development of the most precise movie recommendation algorithm. Netflix's movie recommendation system epitomizes a hybrid recommender system, leveraging multiple strategies. It generates recommendations by examining the browsing behavior of akin users and evaluating viewing histories. Additionally, it

suggests movies that share similar attributes or characteristics with those the user has previously appreciated.

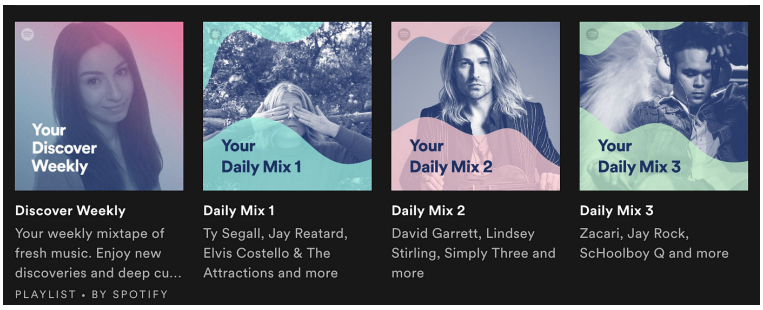

Figure 2.5: Spotify "Discover Weekly" feature

Spotify, an audio streaming platform, continually generates tailored playlists for its subscribers known as "Discover Weekly." These playlists comprise 30 songs curated based on users' unique musical preferences. Spotify's acquisition of Echo Nest, a music intelligence and data analytics startup, enabled the creation of a music recommendation engine incorporating three distinct recommendation models[8]:

- Collaborative Filtering: This model filters songs by juxtaposing users' historical listening patterns with those of other users.
- Natural Language Processing: Spotify's recommendation system scours the internet for comprehensive information about specific artists and songs. Each entity is then assigned a dynamic list of top terms that evolves daily, weighted by relevance. The algorithm subsequently discerns musical similarities based on these terms.
- Audio File Analysis: The recommendation algorithm evaluates the distinctive characteristics of each individual audio file, encompassing aspects such as tempo, loudness, key, and time signature, culminating in tailored music suggestions.

Ringo, a user-based collaborative filtering (CF) system, specializes in recommending music albums and artists. Upon a user's initial interaction with the system, they are presented with a list of 125 artists to rate, reflecting their preferences. This list encompasses two distinct sections: the first section comprises frequently rated artists, fostering a sense of similarity between users' profiles by allowing them to rate artists that others have similarly evaluated. The second section involves a random selection of items from the entire user-item matrix, ensuring that all artists and albums are eventually rated throughout the initial rating phases.

GroupLens adopts a collaborative filtering approach based on a client/server architecture to provide recommendations for Usenet news, a high-volume discussion list service on the internet.

Overcoming the challenges posed by the transient nature of Netnews and the inherent sparsity of rating matrices, this system employs user and Netnews clustering, considering existing news groups. Implicit ratings are derived by analyzing the time users spend engaging with Netnews content.[2]

Hybrid systems, which amalgamate multiple recommendation techniques, define the architectural landscape of numerous projects. For instance, NewsDude combines the K-Nearest Neighbors (KNN) classifier and the Naïve Bayes algorithm to recommend news articles to users. Pipper exemplifies a feature combination technique that incorporates collaborative filter ratings into a content-based system as a feature for movie recommendations. Moreover, CiteSeer, an automatic citation indexing system, leverages a medley of heuristics and machine learning algorithms to process academic documents, transforming it into one of the largest and most widely utilized repositories for research papers on the internet.

2.2 Alternating Least Square Matrix

The advent of AI and Big Data analysis has revolutionised the behaviour of many companies, causing the influence of data-driven companies to grow exponentially in recent times. The increasing integration of artificial intelligence (AI) and machine learning (ML) has played a key role in the development of systems that benefit not only the company itself, but also its users. In this landscape, recommendation systems stand out as a powerful tool for digital enterprises[9]. The recommendation problem can be divided into two main categories: explicit and implicit feedback settings. In the explicit feedback scenario, users provide ratings for items and the system can shape user preferences based on these ratings. In contrast, in the more challenging but prevalent implicit feedback scenario, the system must infer user preferences from indirect data, such as the presence or absence of events, e.g. a user's viewing of an article. To deal with implicit feedback, one approach involves minimising a classification objective function, instead of the traditional mean square error used in prediction. However, directly minimising a ranking objective function can be computationally intensive. To strike a balance between accuracy and computational efficiency, a common strategy is to sample the objective function.[10]

A widely adopted technique for this purpose is ***Alternating Least Squares (ALS)***, which efficiently approximates rankings and user preferences for items, especially in large-scale recommendation systems.
Alternating Least Squares (ALS) is an iterative optimization technique commonly used in collaborative filtering recommendation systems. Its primary purpose is to factorize a large user-item interaction matrix into two lower-dimensional matrices, one representing user features and the other representing item features. ALS is particularly effective in handling implicit feedback

data, where explicit ratings may be absent, and the system must infer user preferences based on user-item interactions, such as clicks, views, or purchase history.

ALS begins by initializing random values for the user and item feature matrices. These matrices represent latent factors that capture underlying patterns in user-item interactions. ALS alternates optimization between two main steps:

- Fix User Features, Optimize Item Features: In this step, ALS holds the user feature matrix constant and optimizes the item feature matrix. It does this by solving a least-squares problem that minimizes the error between the observed user-item interactions (implicit feedback) and the predicted interactions based on the current user features and updated item features.
- Fix Item Features, Optimize User Features: ALS then switches roles and keeps the item feature matrix fixed while optimizing the user feature matrix. It solves another least-squares problem to minimize the error between observed interactions and predicted interactions, but this time it uses the updated item features and the current user features.

ALS iteratively repeats these two steps until convergence. The convergence criterion is usually based on a certain number of iterations or when the changes in the user and item features become sufficiently small. Once ALS converges, the final user and item feature matrices are obtained.
These matrices can then be used to make recommendations by calculating the predicted user-item interactions based on the dot product of user and item feature vectors. Higher predicted values indicate a higher likelihood of user interest in the corresponding items.

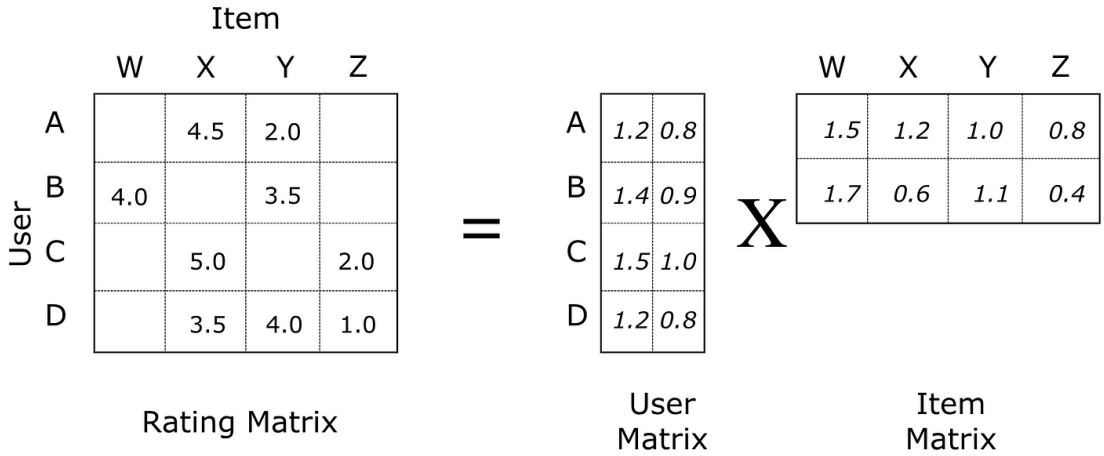

Figure 2.6: Example of Matrix Factorization

Mathematically, the interaction data between user and item could be represented as an p x q

matrix S where p refers to users and q refers to items. The $(u, i)^{th}$ entry is s_{ui} in the matrix S which implies that $(u, i)^{th}$ ratings for i^{th} item by u user. The matrix Q is a sparse matrix, since the elements do not receive ratings from the majority of users. Consequently, matrix Q has the highest number of missing values. In order to solve the sparse matrix issue the solution is to use matrix factorization.
Thus, we have two k-dimensional vectors that are called factors[11].

- x_u is k dimensional vectors summarizing's every user u.
- y_i is k dimensional vectors summarizing's every item i.

The matrix factorization process involves decomposing a user-item interaction matrix into two smaller matrices, in such a way that their product approximates the original matrix as closely as possible. This decomposition aims to capture the latent features of both users and items, enabling the model to make personalized predictions. However, it's essential to recognize that this optimization problem is inherently non-convex. This non-convexity arises because the objective function, which measures the error between the original matrix and the product of the decomposed matrices, often contains multiple local minima. This means that traditional convex optimization methods, which aim to find the global minimum of a function, may not be directly applicable.
Despite the complexity introduced by non-convexity, optimization remains crucial in this context because it determines how the model adjusts its decomposed matrices to progressively improve predictions [12]. Through successive iterations, the algorithm consistently seeks to reduce the error, thus learning the hidden relationships between users and items. In other words, optimization is at the heart of the matrix factorization learning process and represents a fundamental element for the success of recommendation systems based on this approach.

Let,

$$s_{ui} \approx x_u^T Y_i$$

$$x_u = x_1, x_2, ..., x_n \epsilon R^k$$

$$y_i = y_1, y_2, ..., y_n \epsilon R^k$$

We can formulate an optimization problem to find:

$$argmin \sum_{r_u i} (s_{ui} - x_u^T y_i)^2 + \lambda (\sum_u ||x_u||^2 + \sum_u ||y_u||^2)$$

The over-fitting problem is solved by the regularization factor λ. The value of λ can be tuned to solve over-fitting whereas default value is 1.[13]
The optimize value of x_u and y_i can be found by repeating the mentioned procedure until convergence is achieved.

To provide a clearer and more real-life view of how to apply an ALS algorithm to a recommendation system, a practical example is needed. Let us take the case of a popular online film streaming platform, with the goal of improving the recommendation system to provide more personalised recommendations to users, with the intention of implementing ALS, a matrix factorisation technique. First, you gather historical data about user interactions with your platform. This dataset contains information about which movies each user has watched, how they've rated them, and possibly additional information like genre preferences. Then, you organize this data into a user-item interaction matrix, where rows represent users, columns represent movies, and each cell contains a user's rating or interaction with a movie. However, this matrix is usually sparse because not all users have rated all movies.

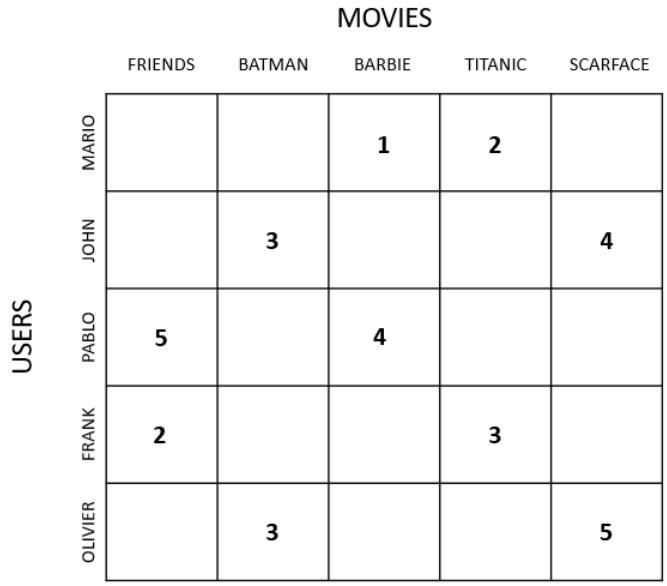

Figure 2.7: Example of User-Item Matrix

Now, you apply ALS to factorize this matrix into two lower-dimensional matrices: one representing users and the other representing movies. ALS aims to find latent factors (hidden characteristics) that describe both users and movies. For example, latent factors might represent features like "action," "romance," "comedy," or "science fiction" for movies and "preferences" or "tastes" for users. The goal of model is to identify latent factors that explain observed ratings.

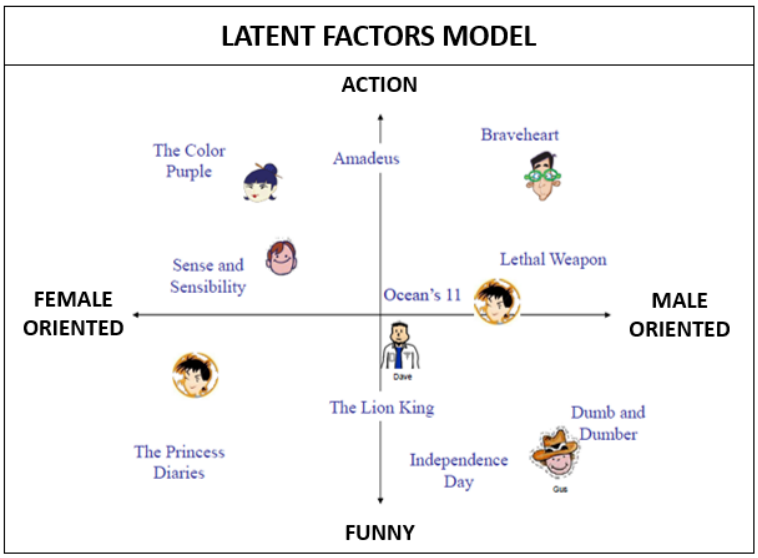

Figure 2.8: Example of Latent Factors Model

ALS iteratively alternates between updating the user matrix and the item matrix. In each iteration, it minimizes the squared error between the original user-item interaction matrix and the product of these two lower-dimensional matrices. This process proceed until convergence. Once it converges, you now have two matrices that capture latent factors. These matrices can be multiplied to reconstruct the original user-item interaction matrix. The power of ALS is that it fills in the gaps in your sparse matrix, providing predictions for user ratings on movies they haven't yet interacted with. With the filled-in matrix, you can recommend movies to users based on their predicted preferences. For example, if a user has shown a preference for action and adventure movies in the latent factors, you can recommend movies with similar latent factor representations.
As users interact with the platform and provide feedback on recommendations (e.g., by rating or watching suggested movies), the ALS model continuously learns and adapts, making future recommendations even more personalized. In this way, ALS helps your movie streaming platform provide tailored movie suggestions to users, improving their experience and increasing user engagement. Plus, it's a scalable and efficient approach, making it a valuable tool for real-world recommendation systems in various domains.

ALS is especially beneficial for large-scale recommendation systems because it can efficiently handle vast amounts of data. By decomposing the user-item interaction matrix into lower-dimensional matrices, ALS reduces the dimensionality of the problem, making it computationally tractable while capturing the essential patterns in user-item preferences. This makes ALS a valuable tool for personalized recommendation in various domains, including e-commerce, content streaming, and more.

LinkedIn utilizes Alternating Least Squares (ALS) in its "People You May Know" feature to provide users with personalized recommendations for connecting with others on the platform. ALS is an essential component of LinkedIn's recommendation system, allowing the platform to suggest connections that are likely to be relevant and interesting to users. The model considers various factors, such as user behavior, connections, and shared interests, to generate these recommendations. It examines patterns in user interactions, identifying users who may have similar professional backgrounds, industry interests, or mutual connections. In practice, a user has several connections who work in the same industry or have attended the same universities, LinkedIn's ALS-based system can identify similar profiles and suggest them as potential connections. This enhances the user experience by making it easier to find and connect with professionals who share common interests or networks, ultimately helping users expand their professional circles.

The music streaming service Pandora uses ALS to create personalized radio stations for users based on their musical tastes and preferences. Pandora, the popular music streaming platform, utilizes an ALS-based recommendation system to enhance the music listening experience for its users and plays a crucial role in Pandora's ability to suggest personalized playlists and songs tailored to each user's musical preferences. Pandora's model works by analyzing the historical listening behavior of users. It examines patterns such as the genres, artists, and songs users have interacted with in the past. By identifying these patterns, the ALS algorithm can make predictions about what users might enjoy listening to in the future. For instance, if a user frequently listens to classic rock and folk music, Pandora's ALS-based system can identify similar music profiles and recommend tracks and artists within those genres. Additionally, it can introduce users to new songs and artists they may not have discovered otherwise, expanding their musical horizons.
Pandora's use of ALS demonstrates the effectiveness of collaborative filtering techniques in the music streaming industry. It helps users discover music that aligns with their tastes while providing a platform for emerging artists to gain exposure[14].

In conclusion, the Alternating Least Squares (ALS) algorithm stands as a formidable tool in the field of recommender systems. Its ability to navigate the intricate landscape of user-item interactions and extract latent patterns has revolutionized how businesses deliver personalized recommendations. ALS has found applications in diverse domains, from e-commerce to content streaming, reshaping the way consumers engage with products and services. The non-convex optimization challenge it poses, while computationally demanding, has been met with innovative solutions and powerful computing resources, further enhancing its utility. ALS showcases the

fusion of mathematical rigor and real-world applicability, demonstrating how complex problems can be tackled with elegant algorithms. As the world of data continues to evolve, ALS remains a prominent choice for recommendation engines, continually adapting to the ever-expanding dimensions of big data. With its capacity to harness collaborative filtering, ALS keeps the user experience at the forefront, delivering tailored suggestions that reflect individual preferences. It has become an indispensable asset for businesses striving to meet the demands of today's data-driven, customer-centric markets.

Chapter 3

Methodology

3.1 Data Collection

This study was conducted with the support of data provided by the H&M Group, which launched a competition on Kaggle, a famous portal for data scientists, to develop a recommendation system based on past transaction data. H&M also provided metadata for products and anonymized customers.
The set of datasets provided by the company includes:

- ***Products*** - The dataset contains all information about the product. There are 105.542 different products. Characteristics reported include: category (including macro-categories, sub-categorie and catalogue classification), presence of graphics or illustrations, colour, gender and sector.

	article_id	product_code	prod_name	product_type_no	product_type_name	product_group_name	graphical_appearance_no	graphical_appearance_name	colour_group_name	...	department_name	index_code	index_name	index_group_no	index_group_name	section_no	section_name	garment_group_no
0	0108775015	108775	Strap top	253	Vest top	Garment Upper body	1010016	Solid	Black	...	Jersey Basic	A	Ladieswear	1	Ladieswear	16	Womens Everyday Basics	1002
1	0108775044	108775	Strap top	253	Vest top	Garment Upper body	1010016	Solid	White	...	Jersey Basic	A	Ladieswear	1	Ladieswear	16	Womens Everyday Basics	1002
2	0108775051	108775	Strap top (1)	253	Vest top	Garment Upper body	1010017	Stripe	Off White	...	Jersey Basic	A	Ladieswear	1	Ladieswear	16	Womens Everyday Basics	1002

- ***Clients*** - Customer data includes master data regarding users. There are 1.371.980 unique clients. The dataset cointains multiple variable such as age, postal code, newsletter frequency, member status and activness.

	customer_id	FN	Active	club_member_status	fashion_news_frequency	age
0	00000dbacae5abe5e23885899a1fa44253a17956c6d1c3...	NaN	NaN	ACTIVE	NONE	49.0
1	0000423b00ade91418cceaf3b26c6af3dd342b51fd051e...	NaN	NaN	ACTIVE	NONE	25.0
2	000058a12d5b43e67d225668fa1f8d618c13dc232df0ca...	NaN	NaN	ACTIVE	NONE	24.0

- ***Transactions*** - This is the reference dataset where we find all the details of the transactions recorded over two years. This dataset counts 31.788.324 transactions and contains variable such as expenditure per product, date of purchase and channel used.

	t_dat	customer_id	article_id	price	sales_channel_id
0	2018-09-20	000058a12d5b43e67d225668fa1f8d618c13dc232df0ca...	0663713001	0.050831	2
1	2018-09-20	000058a12d5b43e67d225668fa1f8d618c13dc232df0ca...	0541518023	0.030492	2
2	2018-09-20	00007d2de826758b65a93dd24ce629ed66842531df6699...	0505221004	0.015237	2

The objective of the project is to build a recommendation system that predicts 12 items in order of relevance to each user.

There are no user evaluations in our dataset which implies that the recommendation system must rely on a different type of feedback, often referred to as 'implicit feedback'. This feedback may come from various sources, such as in our case past transactions, or product views, browsing preferences, or any other action that suggests an interest on the part of the user in a particular item. In the following chapter, the data present to exploit this type of implicit feedback to develop an accurate and relevant recommendation system is examined in detail. In this context, the chapter on methodology serves as a fundamental guide to understanding the entire process of developing a recommendation system based on implicit data.

3.2 Explorative Data Analysis

In the study phase of the dataset, several characteristics of customer behaviour in the decision-making process can be observed.

Looking at the available database, the user base counts 1.371.980 customer IDs.

The age distribution of a population can often present interesting and significant characteristics. Data is composed of two well-defined demographic groups.

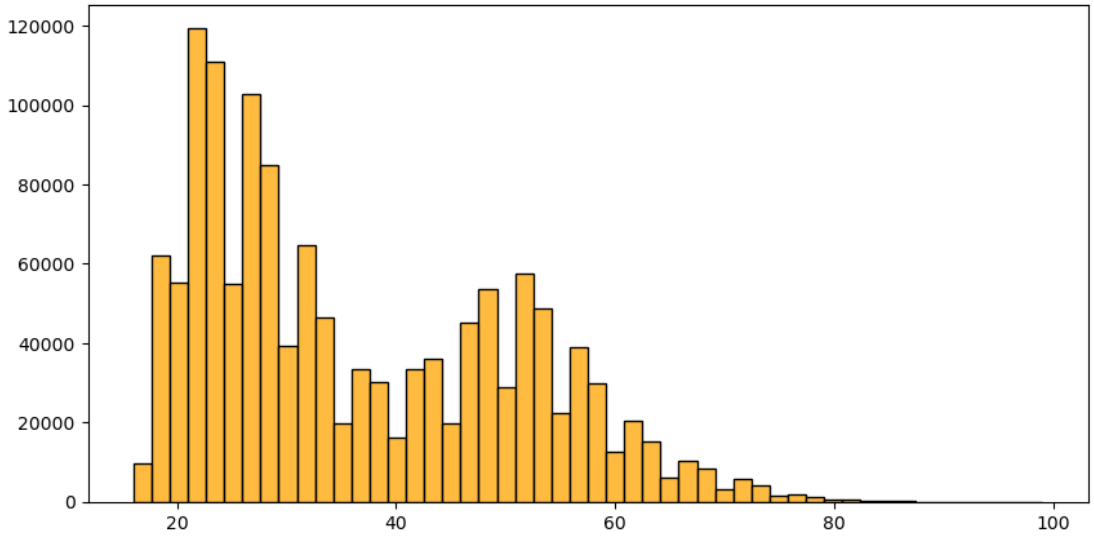

Figure 3.1: Distribution of Customers Age

The bigger cluster is between 25 and 30, suggests the presence of a significant proportion of young or working-age individuals within the population. The second peak, on the other hand, around the age of 50, suggests a concentration of middle-aged or older individuals.
To better understand the behaviour of the different clusters, I divided users in 3 different groups per age group that could represent the macro differences between the social groups in terms of decision-making processes and spending power.

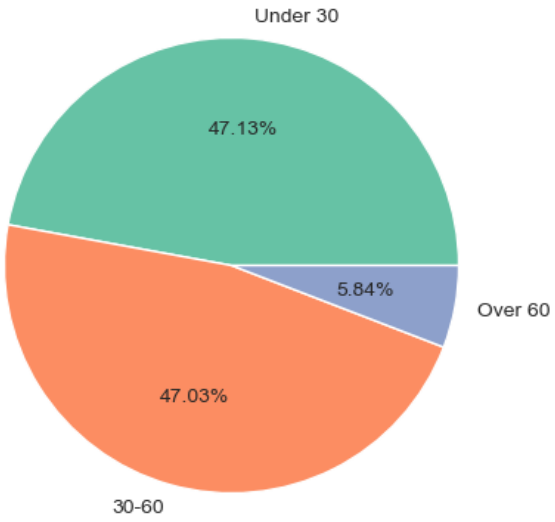

Figure 3.2: Distribution of Customers Age-Range

With these settings we can verify that 47.1% of total customer are Under-30, 47% belong to the segment 30-60 and 5.8% are Over-60.
The dataset presents other important costumer characteristics for company and the study.
Two details about customers concern their active participation in the company. First, we find their status as members in the company club.

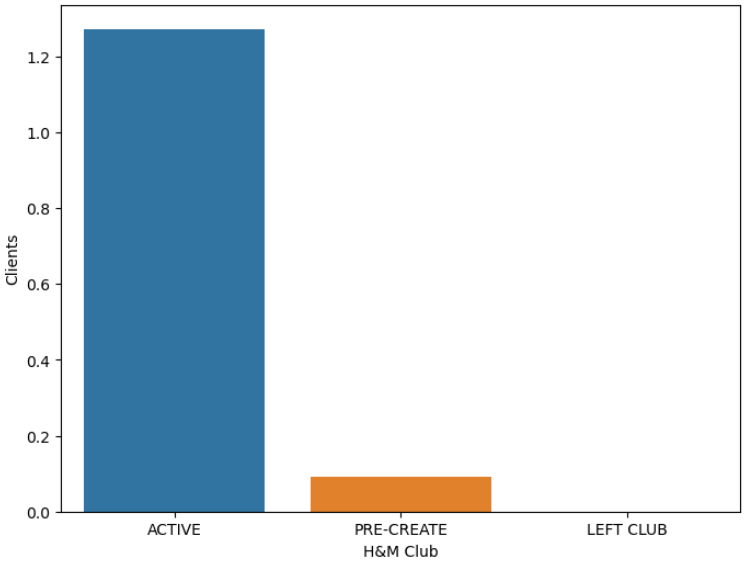

Figure 3.3: Customers H&M Club Status

Here is observed that 97% of the users are active within the company. Active means that they make a purchase at least once a year. This does not express much analytically but helps to give a dimension of the data we are manipulating.

Another important attribute that demonstrates the frequency of interaction between users and corporate systems is the frequency of newsletters. The newsletter is an online 'bulletin' that companies use as a corporate communication method to contact their subscribers and inform them about their services and products.

This type of company-customer relationship creates a close relationship between the two that grows over time. The main objectives of newsletters are: to strengthen the corporate brand by helping to enhance reputation and promote corporate image; to increase sales capacity through targeted marketing communication; to build customer loyalty by strengthening ties with customers; and finally to increase site visits through the use of links.

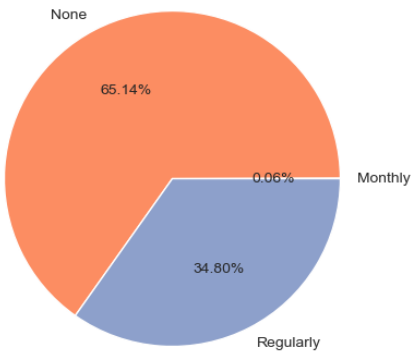

Figure 3.4: Customers Newsletter Frequency Distribution

The pie chart shows that 65% of customers do not receive any newsletters, while the remaining portion receive regular communications.
The last detail per customer is the residence expressed as a postcode. Unfortunately, this data has been encoded for company's privacy, so no insight can be gained.
The number of items available in the dataset is 105.542 and them can be divided by category and characteristics.
The available data are very detailed regarding the differences between the products. Starting with the macro-categories, certainly the sector is the one of the most important.

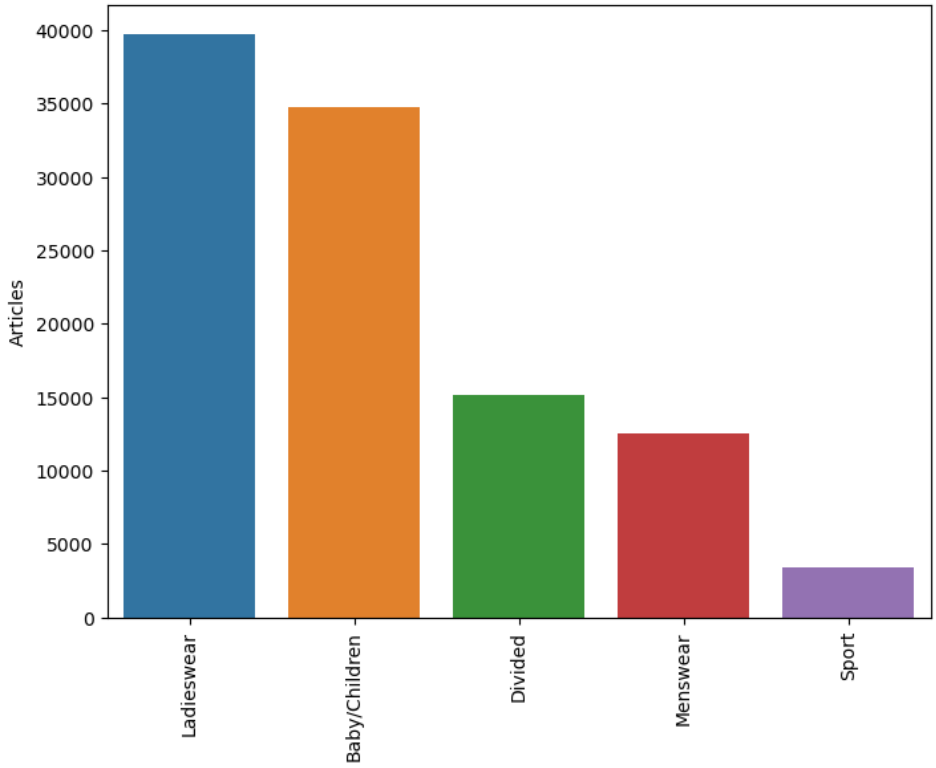

Figure 3.5: Number of Items by Sector

Ladieswear results to be the most populated category sector followed by Children. It is interesting to note the clear difference in items between men and women. It seems that the company focuses a lot on women and this is also shown by the large availability of children's items, which are usually bought by mothers or relatives.
Lastly, there is also the sport sector which counts a few thousand of items.
Going into greater detail for each product, another macro category to highlight is the product category in the catalogue.

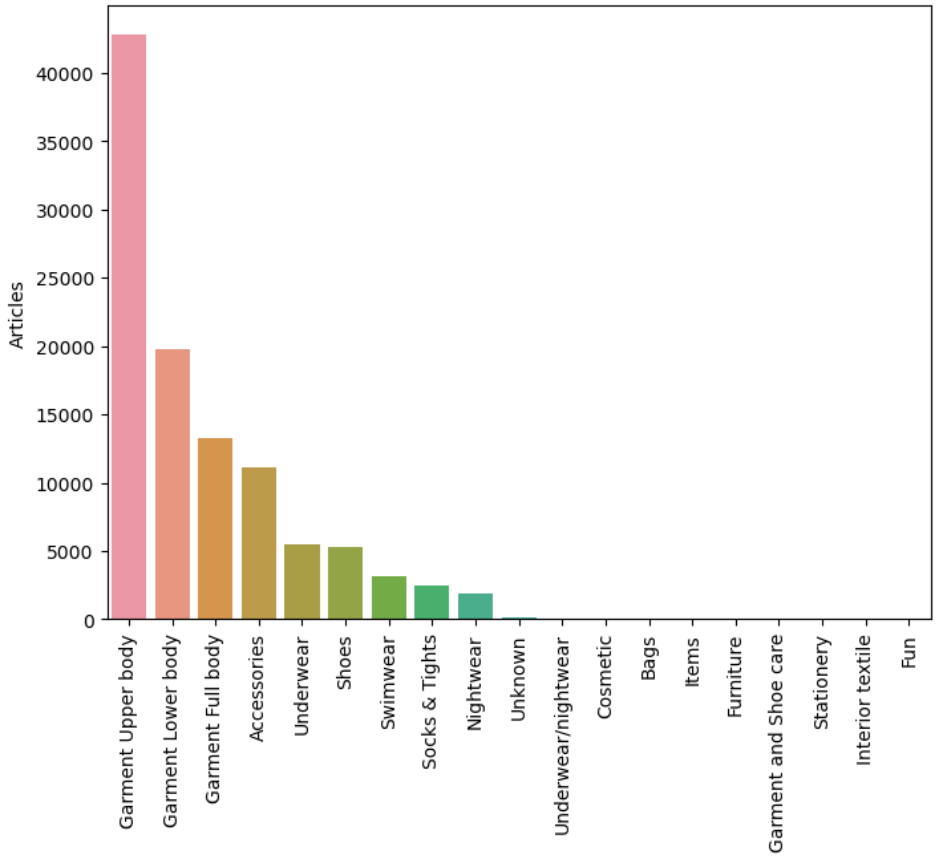

Figure 3.6: Number of Items by Macro-Category (Catalogue)

The main insight that emerges from the graph is clear, in fact the products are mainly categorised in the macro category 'Upper Body' and 'Lower body' followed by 'Full body' this is because a company operating in the world of fast fashion has casual wear as its core products. Nevertheless, we can identify no less than 19 different categories in the database catalogue.
The available database also makes it possible to study the graphic and visual attributes of products. There are 50 different colours of which the most commonly used are Black, Blue and White. In addition, the data provides details on any graphics on the products. This variable indicates the presence of particular patterns, symbols or prints.
Exploratory data analysis in the context of recommendation systems for fashion sites requires an in-depth understanding of product attributes, including 'colour' and 'prints'. These attributes are of crucial importance as they directly influence consumers' fashion preferences and choices. The colour of a garment or accessory can have a significant impact on its attractiveness to an individual. The choice of colour is often influenced by personality, fashion trends, social context and even seasonal preferences. For example, some users may be attracted to bright and bold shades, while others may prefer neutral and sober colours. Understanding colour is essential for the recommendation system as it allows it to suggest products that align with each user's colour preferences.

Prints represent another crucial aspect in the fashion world. Different prints, such as flowers, stripes, animal prints and many others, can convey different styles and tastes. Some users may like floral and romantic patterns, while others may prefer geometric and minimalist designs. The ability to understand users' print preferences allows the recommendation system to suggest products that suit their personal style. Furthermore, the combination of attributes such as colour and prints can lead to highly customised recommendations. For example, a user might want a blue dress with a floral print, and the system should be able to identify products that match this specific description. Ultimately, a thorough understanding of product attributes, including colour and print, is crucial to creating an effective recommendation system on fashion sites. These attributes enable the system to offer relevant recommendations that reflect individual user preferences, thus enhancing the online shopping experience and customer satisfaction.

To conclude the product analysis regarding attributes, the last insight concerns the sub-category.

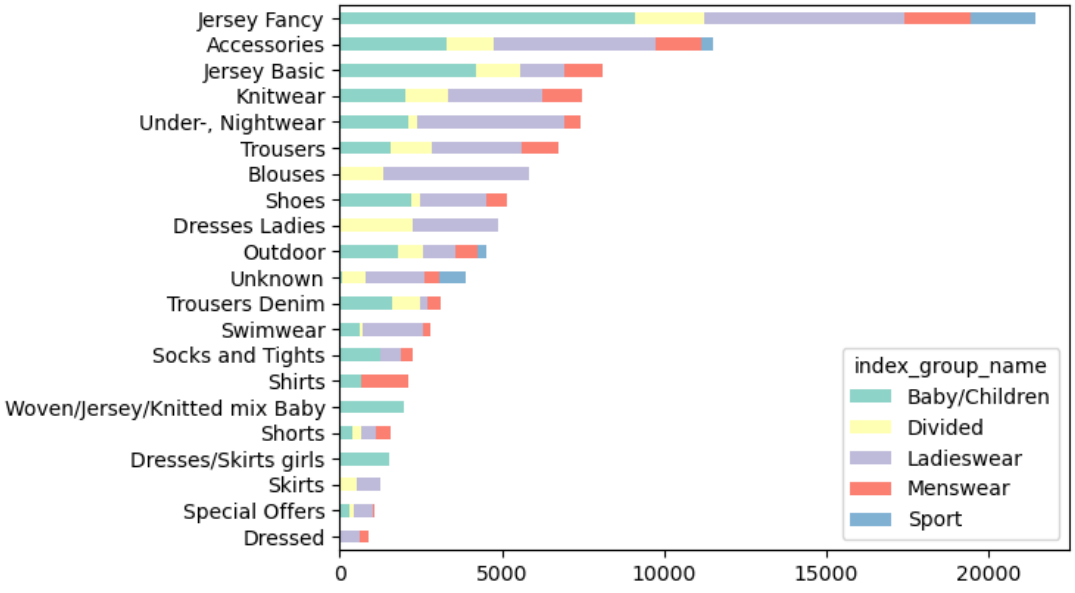

Figure 3.7: Number of Items by Sub-Category with Sector details

The bar graph shows the top twenty subcategories by product number. Jerseys emerge as the main product in the catalogue among other things in the fast fashion sector it is also a very easy item to sell for reasons of economy and everyday use.
In the above plot, a stacked bar graph is shown where each subcategory is divided by its sector. No interesting insights emerge from this graph as the distribution of sectors is more or less uniform across all subcategories.

The last dataset available is that of only the transactions of each customer. This dataset alone is not useful for deriving interesting studies, which is why I decided to merge all the datasets provided and have a complete view of the data. The transaction file contains the price of the products, the sales channel as well as the date.
The data collected starts from 20^{th} September 2018 until 20^{nd} September 2020, and it counts 31.788.324 transactions registered in two consecutive years.

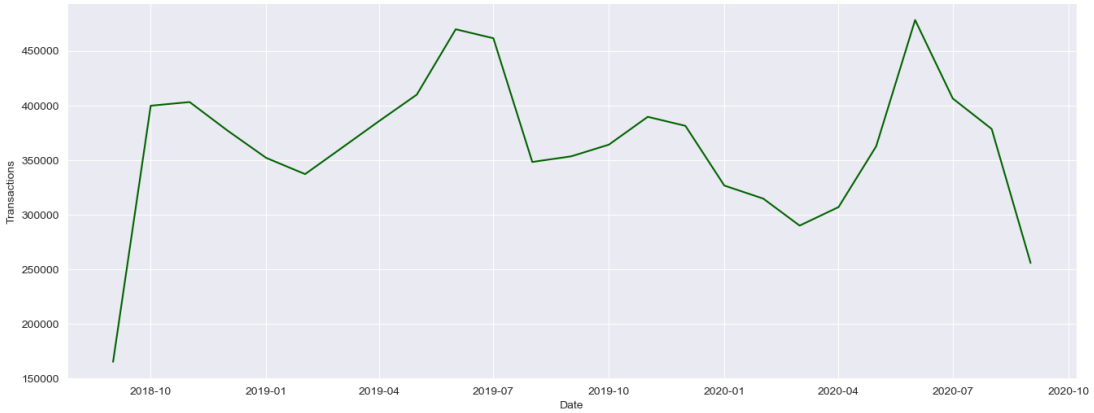

Figure 3.8: Number of transactions by month

The sales trend fluctuates with many ups and downs over time and the two biggest peaks are observed in the summer months, in particular, in June.
The data base is also influenced by the presence of Covid-19 in fact since February/March 2020 the epidemic has affected the whole world and halved or zeroed the sales of many companies.

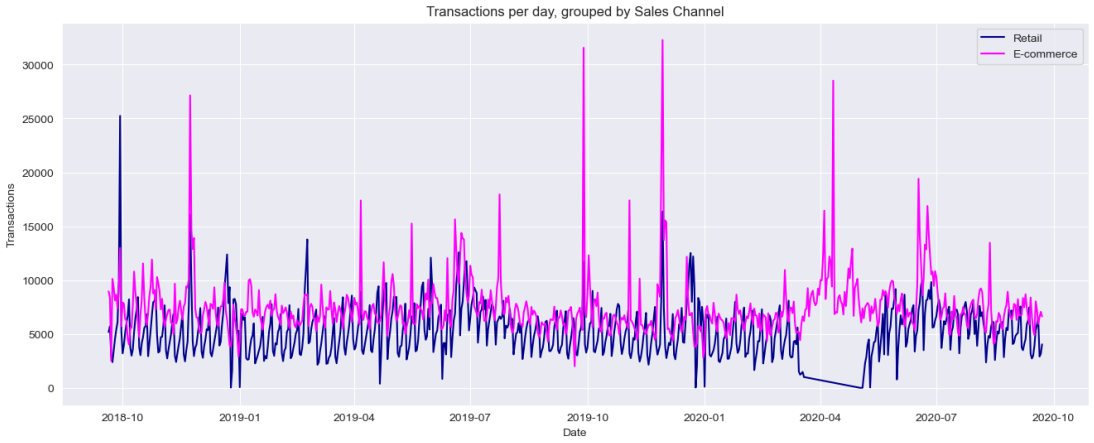

Figure 3.9: Number of Daily transactions by Sales Channel

In the line graph, transactions were divided by sales channel. The pink line represents transactions made online while the blue line represents transactions made in physical shops. It can observed that between March and May purchases in physical shops are very close to 0. On the other hand, it can be observed that online transactions definitely increased during the same period. During the pandemic, many fast-fashion companies experienced significant benefits from having a reliable e-commerce site. These advantages made clear the importance of a robust online presence for the fashion industry. One of the main benefits has been the continuity of sales, despite restrictions on physical shops and shopper travel, ensuring an open and operational sales channel during lockdown and closures of physical shops. E-commerce offered the possibility to quickly adapt to changing market conditions. Companies could easily adjust their offerings, removing or adding products according to consumer needs and preferences in real time. This flexibility was crucial during the pandemic, when customer demands could change rapidly. This experience has shown that e-commerce has become an essential component for the success of modern fashion companies.

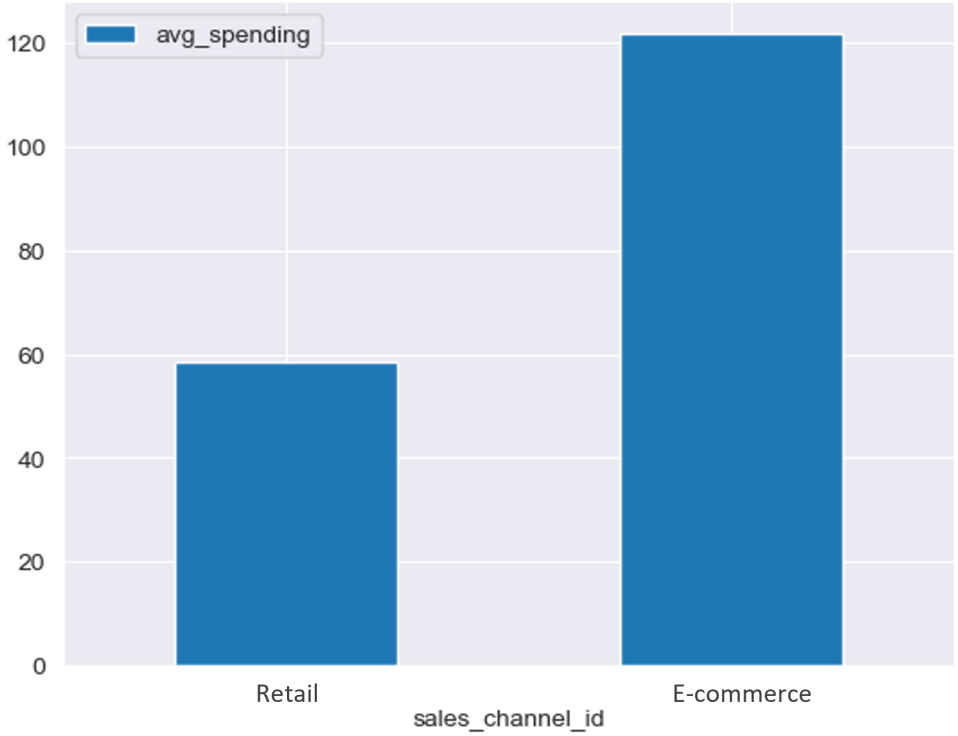

Figure 3.10: Customers average spending by Sales Channel

Continuing to analyse the two sales channels, it can be observed that the average spending for each user in E-commerce is twice as much as the average spend in retail. This disparity can be attributed to several key factors that highlight consumer dynamics in these two different sales channels. First, the convenience and accessibility offered by e-commerce can positively

influence customer spending. Secondly, the presence of promotions, discounts and special offers on the e-commerce site may be a factor that encourages customers to spend more. E-commerce companies often use dynamic pricing strategies and targeted promotions to attract and retain customers, which can lead to more consistent transactions. In addition, the online shopping experience may be more engaging for some customers, with features such as product reviews, personalised suggestions and an efficient checkout process helping to facilitate purchase. Finally, e-commerce offers an environment in which it is easier to monitor purchase history and customer preferences. This data can be used to generate personalised recommendations, suggesting relevant products that customers might be interested in. This personalization can positively influence online customer spending.

The merged dataset, which contains all the details of customers together with their transactions, makes it possible to analyse the customer base in greater depth.

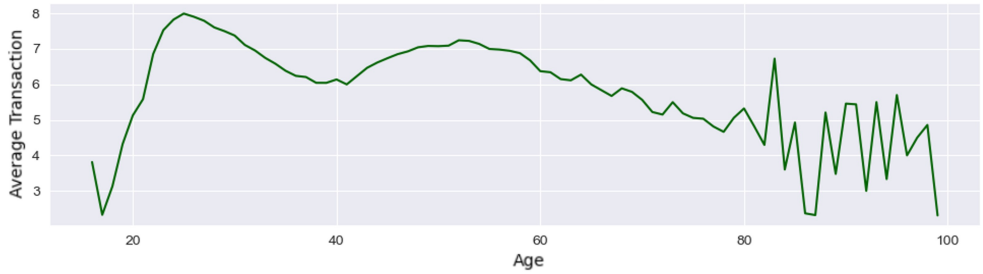

Figure 3.11: Average Transactions by age

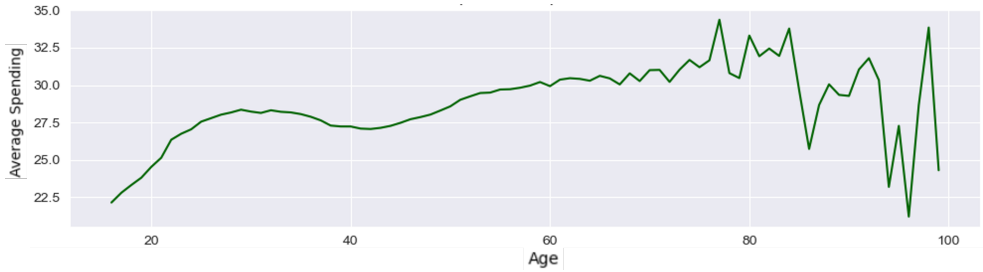

Figure 3.12: Average Spending by age

The graphs above show the change in the number of transactions and average expenditure in relation to age. By observing the graphs, it can be seen that they have completely opposite trends. The analysis of the average number of transactions shows a trend where the largest number of transactions belongs to the part of the population under 30 years of age and the more one moves towards an older age, the fewer transactions on average are recorded. In contrast, the average expenditure is higher for individuals with an older age, this trend is explained by the greater spending power an adult individual may have compared to a teenager, moreover,

it must be considered that an adult often also shops for children and family members and this has a great impact on the analysis of the average expenditure.

Another curious analysis would be to see if there are differences in customer purchasing behaviour between weekdays and weekends. Analyzing customer behavior in relation to the days of the week is a crucial aspect for many businesses and institutions. Distinguishing between weekdays, known as "weekday," and weekends provides valuable insights into user habits, preferences, and trends. During weekdays, a significant portion of the population is engaged in work and daily activities, which can greatly influence purchasing decisions and online interactions. At the same time, weekends offer a unique opportunity for individuals to dedicate more time to leisure, shopping, and entertainment experiences.

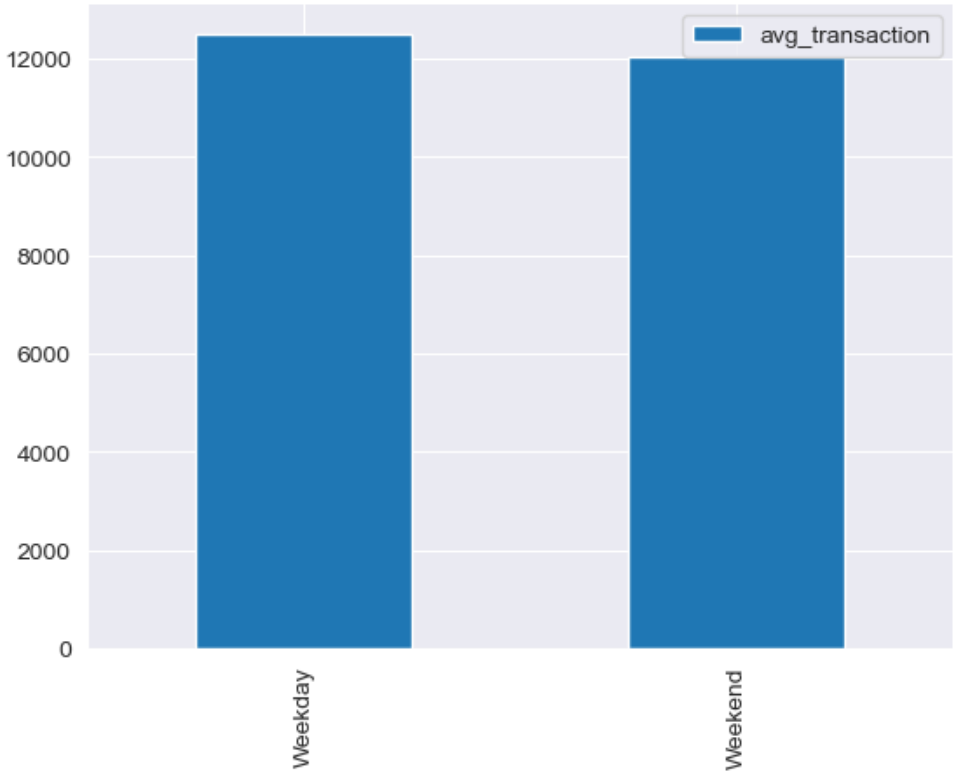

Figure 3.13: Average Transaction by Day Type

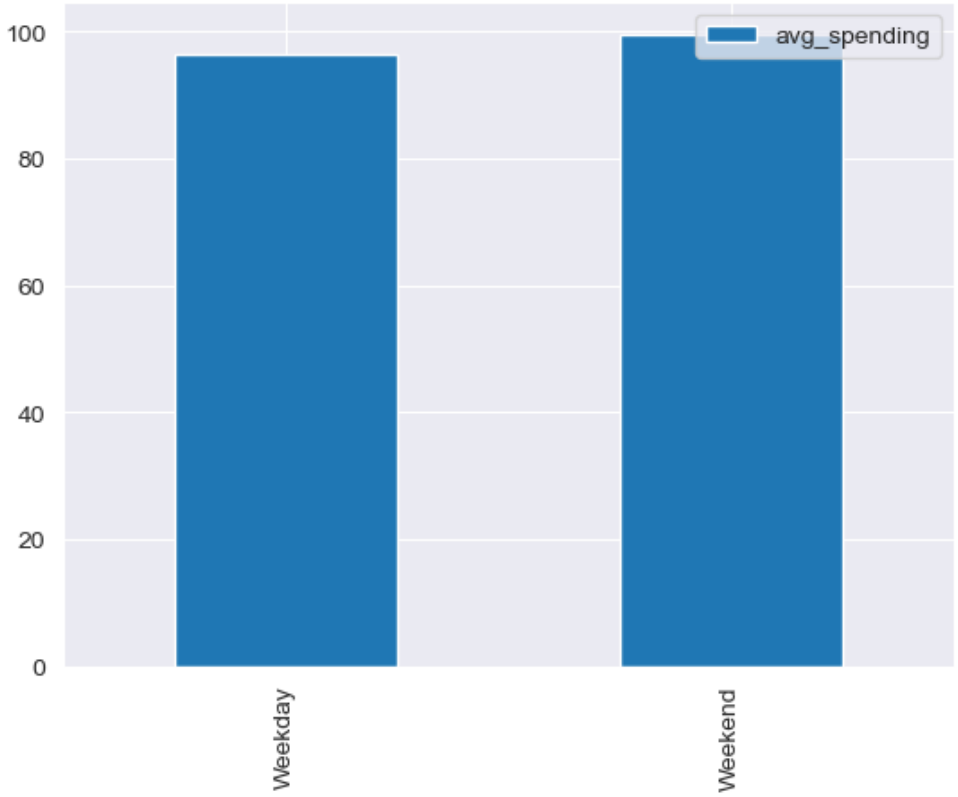

Figure 3.14: Average Spending by Day Type

It can be seen from the bar plot that the average transactions between the periods of the week do not differ much. Surprisingly, the company receives more purchases during the weekdays. This is probably because the ecommerce platform allows users to make purchases at any time and does not restrict the customer to making purchases only on free days, which in most cases is during the weekend. With regard to the average spend, although the difference in this case also remains minimal, the trend is the opposite.

For the study, the analysis of customer behaviour under different circumstances is crucial to develop an effective recommendation system. Likewise, it is interesting to shift the focus to the products and see the dynamics of the customer transaction data for each item.

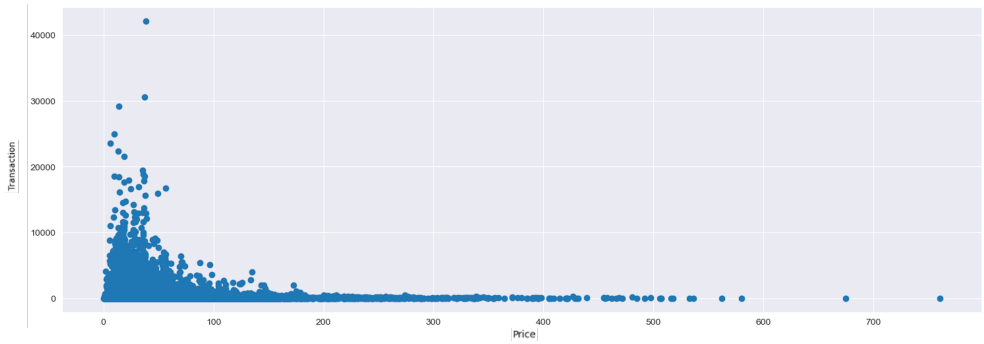

Figure 3.15: Scatter plot: Transaction vs Price

The graph presents the price of the items on the x-axis, while the y-axis shows the number of transactions associated with each item. This scatter plot reveals an interesting pattern that highlights some significant trends within the dataset. The observation of a high point density in the lower left of the graph, where item prices are lowest, suggests that there are a number of affordable products that have a fair number of associated transactions. This could indicate that customers are inclined to make more frequent purchases of affordable items, thus creating a significant market segment for affordable products. On the other hand, as we move up the x-axis, we observe a decrease in the number of transactions associated with higher priced items. This suggests that as the price of items increases, the overall volume of transactions tends to decrease. This phenomenon is consistent with economic theory, in which more expensive items may require more considered and less frequent purchasing decisions by customers.

3.3 ALS Implementation

The recommendation system was developed using an implicit model through the use of the *implicit* library. The choice to use an implicit recommendation system based on Alternating Least Squares (ALS) was driven by the nature of the available data. Available data does not contain explicit ratings but rather transaction-based information. This makes the dataset well-suited for an implicit recommendation system since implicit feedback is often more abundant and easier to collect compared to explicit feedback.

The ALS method is particularly suitable for addressing this situation as it allows modeling implicit interactions between users and items. The ALS approach decomposes the user-item interaction matrix into two smaller submatrices, one for users and one for items, representing vector embeddings of users and items in the same space. This enables the efficient calculation of recommendations based on similarity between users and items within the context of implicit interactions. In essence, the items recommended for a given user are the items whose embedding vectors are closest to the embedding vector of that user. ALS leverages this approach to generate personalized recommendations by identifying items that are most akin to a user's

preferences based on their embeddings.
Furthermore, the ALS method is known for its computational efficiency, which is a significant advantage when working with large datasets, such as those from e-commerce transactions. Its ability to handle large amounts of data efficiently is crucial for providing real-time personalized recommendations in a business context.

As model preparation, I assigned a unique integer number to each user ID and article ID, respectively, and then used custom dictionaries to map the original user and article IDs to the corresponding numeric IDs. This is used to create a numeric representation of the user and article data.
The sparse user-item interaction matrix based on the transaction data was constructed with an array where each entry is set to 1. This array represents the implicit user-item interactions. The value of 1 signifies that a user interacted with an item by transaction. As result, a sparse matrix created using the *COO* format. It's a binary matrix where rows correspond to users, columns correspond to items, and the values represent user-item interactions. A Coordinate List (COO) matrix is a sparse matrix representation that efficiently stores and manipulates data in a structured manner. In a COO matrix, the focus is on non-zero entries, and any 0 entries are disregarded to conserve storage space. This approach is particularly advantageous when dealing with matrices that have a large number of zero elements, as it allows for significant savings in memory. The COO format is well-suited for scenarios where incremental matrix construction is a requirement. It excels in situations where new rows and columns need to be seamlessly added to the matrix. Sparse matrices are used to efficiently handle large datasets and conserve memory. This user-item interaction matrix, will be a key input for training and evaluating the ALS recommendation model. It captures the implicit feedback from user interactions with items, forming the foundation for generating personalized recommendations.
Before training the model, it is important to set specific parameters. The factors parameter specifies the number of latent factors to use for modeling user and item interactions. In this case, it's set to 10, meaning the model will create 10-dimensional embeddings for users and items. The iterations parameter defines how many times the model iteratively updates its embeddings. In this code, it's set to 2, indicating two iterations.
The ALS model can be trained using the provided *coo_train matrix*. During training, the model learns latent factors for users and items that best represent the observed user-item interactions. ALS alternates between optimizing user embeddings and item embeddings to minimize the reconstruction error on the observed interactions. Now the model contains the learned embeddings, which can be used to make recommendations for users based on their interactions with items in the dataset.

After training the model, it must be validated. To validate it, the dataset was split into training and validation set. The validation set was calculated by taking only the transactions from the last seven days in the entire dataset, then creates various sparse matrices for both sets.

Model validation was performed by training ALS model with specified factors (embeddings dimension), iterations, and regularization strength. It uses Mean Average Precision at K (MAP@K) as the evaluation metric, where K is set to 12 because the objective is that to predict the best 12 items to users. In order to achieve optimal results based on best MAP@K score, I decided to perform a grid search over different hyperparameters which are *factors, iterations and regularization* to find the best combination that maximizes the MAP@12 score. The ALS model iterates through different values of these hyperparameters and selects the combination with the highest MAP@12 score.

Once the optimal parameters have been obtained to train the final model, I converted the COO matrix into a Compressed Sparse Row (CSR) matrix format, which is a compact and efficient way to represent sparse data in a structured manner. In CSR format, the matrix is divided into three arrays: one for the non-zero values, another for the column indices corresponding to these values, and a third for storing the starting index of each row in the first two arrays. This representation allows for quick access to non-zero elements and reduces memory overhead, as zero entries are not explicitly stored. CSR matrices are particularly well-suited for scenarios where matrix-vector multiplications are frequently performed, as they provide a good balance between storage efficiency and computational speed. Additionally, CSR matrices are highly adaptable and can easily accommodate new data or changes to the matrix structure, making them a practical choice for a wide range of applications dealing with sparse datasets. Once trained, the model is ready to produce 12 recommendations for each user using parameters that maximise the MAP@K score.

Chapter 4

Results

Although a recommendation system can indeed offer valuable suggestions to help users navigate through a huge amount of information, it becomes imperative, both for effective implementation and continuous improvement, to obtain information on the algorithm's efficiency in providing recommendations that are truly in line with user tastes and preferences. Therefore, a critical aspect of evaluating recommendation algorithms is understanding their propensity for errors and the extent to which these occur. This evaluation process is akin to scrutinising the algorithm's decision-making capacity. It is a question of verifying whether the algorithm makes erroneous recommendations and, more importantly, of quantifying the extent of these errors. The evaluation methodology is based on comparing the recommendations generated by the algorithm, typically in a test environment where the user's actual purchases are withheld, with purchases that were previously unknown to the system. By undertaking such an evaluation, we gain a nuanced understanding of the algorithm's accuracy, its potential pitfalls, and its ability to cater to users' evolving preferences. In essence, this examination allows us to measure the actual performance of the recommender system, ultimately guiding us in refining its functionality and maximising satisfaction.

In this chapter, we further analyse the performance of our recommendation algorithm. Through a set of evaluation metrics, we examine the effectiveness of our model in generating personalised recommendations. These metrics provide a comprehensive view of how well our recommendation system identifies and suggests relevant products or content to users. These metrics not only serve as benchmarks to assess the accuracy of our system, but also clarify its discriminative ability and overall effectiveness. It is crucial to emphasise the importance of these metrics in our evaluation process, as they serve as a litmus test, allowing us to measure the degree to which our algorithm is able to meet users' needs and improve their experience. Through this analysis, therefore, we try to understand the extent to which our algorithm is able to meet users' expectations and help improve their experience.

The process of choosing an appropriate test set for evaluating recommendation systems is a critical step, as it directly impacts the validity of the assessment. In this study, meticulous attention was given to constructing a test set that would effectively gauge the performance of the recommendation algorithm developed. The first criterion for test set inclusion was customer activity, aiming to focus on users who had exhibited substantial engagement with the platform. Consequently, only those clients who had purchased a minimum of three distinct items were retained for further analysis. This choice ensures that the selected user base is characterized by a certain degree of activity and is more likely to benefit from personalized recommendations. The second criterion revolved around item selection, wherein the goal was to ensure a representative yet manageable set of items for evaluation. To this end, the 400 most frequently purchased items were chosen, considering the frequency of item transactions in descending order. By selecting these items, the aim was to create a test set that encompasses a wide spectrum of user preferences and product diversity while avoiding an unwieldy dataset. Moreover, focusing on the most commonly purchased items helps in the assessment of the recommendation system's ability to effectively suggest well-established and popular products, as these hold substantial commercial importance. Finally, to maintain the temporal integrity of the test set, only the most recent transaction of each customer was retained, representing the user's latest interaction with the platform. This decision was rooted in the idea that users' preferences and needs may evolve over time, making their most recent transaction a pertinent point of reference for evaluating the recommendation algorithm's accuracy and adaptability.
In summary, the test set used in this study consists of 86,994 clients who meet the engagement criteria, a selection of the top 400 items, and only the latest transaction for each user. These criteria were thoughtfully chosen to create a comprehensive yet manageable evaluation dataset, ensuring that the recommendation system's performance is assessed under conditions that closely mirror real-world usage patterns.

Mean Average Precision at K *($MAP@K$)* is a crucial metric in the context of recommender systems as it provides a measure of the quality of recommendations by considering both their accuracy and their positioning in the results. The metric consider the average performance over a set of users or queries, and also assesses how well the system positioned relevant versus irrelevant items in the recommendation results. It is a measure of the accuracy of the first K recommended items. This specifies the number of recommended items to be considered when calculating Average Precision. In the study, $K = 12$ was used, which means that the metric evaluates the precision of the first 12 recommendations for each user. $MAP@K$ is a useful metric because it considers the positioning of recommended results. This means that it not only rewards correct recommendations, but also rewards the fact that these recommendations are placed higher up, where they are more visible and probably more influential for the

user. Therefore, an increase in $MAP@K$ suggests a significant improvement in the quality of recommendations.
A $MAP@12$ value of 0.1075 suggests that, on average, your recommendation algorithm is able to place relevant objects among the first 12 results about 10.75% of the time. This is a promising result, as it means that the algorithm is doing a good job in providing relevant and relevant recommendations to your users. However, it is important to consider the specific context of your recommendation system and the type of data you are working with. In some applications, a $MAP@12$ value might be considered excellent, while in others it might be improvable.

Evaluating recommendation systems is a very complex undertaking, encompassing a number of metrics that collectively shed light on the effectiveness of the system in providing relevant and engaging recommendations. Among these key metrics, the ***Normalised discounted cumulative gain (NDCG)*** emerges as a key indicator of recommendation quality and user satisfaction.
NDCG is a metric that evaluates the ranking quality of recommended items. It is particularly well-suited to scenarios where recommendations are presented in a ranked order, which is often the case in real-world applications. The fundamental idea behind *NDCG* is to assign higher rewards to items that are not only relevant but also appear higher in the recommendation list. In essence, it places a premium on the ranking's top positions, recognizing that users are more likely to engage with and appreciate items ranked at the beginning of the list. The formula for *NDCG* combines two essential components: Discounted Cumulative Gain (DCG) and Ideal Discounted Cumulative Gain (IDCG). DCG quantifies the quality of a ranking by summing the relevance scores of recommended items while applying a logarithmic discount to reflect the decreasing impact of items further down the list. IDCG represents the theoretical highest possible DCG for a given user. *NDCG* is then computed by dividing the actual DCG by the IDCG, resulting in a value between 0 and 1, where higher values signify superior ranking quality.
In the context of this study, achieving an *NDCG* value of 0.13129 underscores the recommendation system's ability to effectively prioritize items that align with users' preferences. It reflects the system's proficiency in not only identifying relevant items but also in presenting them in a compelling order. Consequently, a higher *NDCG* score indicates that the system excels in enhancing the user experience by delivering recommendations that are not only relevant but also strategically ranked to capture user attention and engagement.
In conclusion, *NDCG* serves as a pivotal metric in the evaluation of recommendation systems, offering nuanced insights into the quality of item rankings. The achieved *NDCG* value of 0.13129 reflects the recommendation algorithm's prowess in presenting items in a manner that maximizes their appeal and relevance to users, ultimately contributing to a more satisfying and effective recommendation experience.

In the evaluation landscape of recommendation systems, the Area Under the ROC Curve (AUC) metric stands as a pivotal measure of predictive accuracy and model performance. AUC is particularly relevant when assessing the system's ability to distinguish between relevant and non-relevant items in the context of binary classification, a fundamental task in recommendation engines.

The Receiver Operating Characteristic (ROC) curve, from which AUC derives, plots the true positive rate (TPR) against the false positive rate (FPR) as the classification threshold varies. TPR represents the proportion of relevant items correctly identified as such, while FPR signifies the fraction of non-relevant items incorrectly categorized as relevant. AUC then quantifies the area under this curve, resulting in a value between 0 and 1. An AUC score of 0.6046, as observed in this study, indicates a notable level of discrimination.

An AUC of 0.6046 signifies that the recommendation system under examination possesses a satisfactory capacity to distinguish between items preferred by users and those that are not. A higher AUC score indicates that the model is effective in ranking relevant items higher than non-relevant ones, contributing to the overall precision of recommendations. Importantly, an AUC exceeding 0.5 suggests that the model performs better than random guessing—a crucial benchmark for any recommendation system. Furthermore, AUC is robust to class imbalance, a common occurrence in recommendation scenarios where users interact with only a small fraction of available items. It reflects the model's ability to make reliable recommendations even when there's a disproportionate number of non-relevant items. This ensures that users are presented with suggestions that align with their interests, enhancing their overall experience.

In conclusion, an AUC value of 0.6046 signifies that the recommendation algorithm in this study demonstrates commendable discriminatory power, effectively distinguishing between relevant and non-relevant items. This metric plays a fundamental role in assessing the model's ability to provide meaningful recommendations, thereby contributing to user satisfaction and the system's overall success.

The effectiveness of a recommendation system within the business management is inherently tied to its impact on sales performance. In a business context, the true measure of a recommendation system's success lies in its ability to influence consumer behavior, particularly in driving sales. The pivotal question revolves around how many customers, after being exposed to the system's tailored suggestions, proceed to make purchases. A comprehensive analysis of the system's predictions necessitates a tangible examination of how many customers have made purchases from the 12 recommended items. To delve into this assessment, a test set containing the latest transactions of each customer becomes an indispensable tool. Therefore, the test set serves as the critical bridge between predictive algorithms and real-world business outcomes.

By isolating the last transactions of each customer, the test set provides a comprehensive view of the system's impact on actual purchasing decisions. It acts as the benchmark against which the accuracy and relevance of recommendations can be evaluated. The ultimate goal here is to gauge the extent to which the recommendation system aligns with customers' preferences and drives conversions. This strategic approach ensures that businesses can assess not only the algorithm's predictive accuracy but also its tangible contributions to the bottom line. In essence, it goes beyond algorithmic evaluations to translate recommendation system performance into concrete business metrics, reflecting its true value in driving sales and enhancing the overall customer experience.

client_id	article_id	prediction_0	prediction_1	prediction_2	prediction_3	prediction_4	prediction_5	prediction_6	prediction_7	prediction_8	prediction_9	prediction_10	prediction_11
0000000002	0723529001	351484002	723529001	699080001	759871002	699081001	599580055	599580038	599580052	609719001	759871001	673677002	776237020
0000000014	0516859008	706016001	706016002	573085028	706016003	610776002	706016015	610776001	706016006	573937001	573085004	562245046	554450001
0000000022	0749699001	685816002	749699002	685813001	685816001	749699001	685814001	841383002	716672001	685814003	673677002	678942001	841383003
0000000031	0685814001	685816002	685816001	685813001	685814001	685814003	673677002	806388001	806388002	806388003	573937001	720125001	711053003
0000000036	0685814001	772773001	778064001	673677002	778064003	554772002	772773002	685814001	561445005	685813001	685816002	728162001	685816001
0000000039	0572998001	803757001	573716012	783346001	751471001	796210001	655784001	484398001	740519002	720125001	399201002	751664001	752814003
0000000048	0751471001	759871002	759871001	751471001	733749001	783346001	759871003	759871025	759871004	624486001	408875001	573716012	752814003
0000000053	0706016002	706016002	706016001	706016003	706016006	706016015	573085004	706016019	539723005	573085028	573085043	706016007	706016004
0000000086	0739590002	160442007	160442010	507909001	160442043	673396002	507909003	568601006	470789001	228257001	464297007	507910001	148033001
0000000139	0768912001	717464001	179123001	768912001	355569001	189634001	688432001	711053003	570003002	189616006	179208001	108775015	767834001

The figure presented illustrates a subset of the test set. Specifically, it showcases critical columns for our analysis. The *article_id*column, highlighted in green, provides insight into the products that were indeed acquired by each respective client in their most recent transaction. Conversely, the *prediction_0 ... prediction_11* columns exhibit the system's recommendations for each of these clients. Notably, any product codes highlighted in green within the prediction columns signify items that were successfully predicted and subsequently purchased by the client. An intriguing aspect arises when we consider clients for whom the recommendations diverged from their actual purchases. These particular clients are highlighted in red. As observed, a compelling outcome emerges from this examination: in 70% of the instances (7 out of 10), the predicted items aligned with the items that were ultimately purchased by the clients. Furthermore, it's essential to note that the prediction columns are meticulously organized by their degree of relevance. While examining the tabular representation, it becomes evident that, in this illustrative sample, the most pertinent recommendations consistently occupy the uppermost positions. This alignment between the predictive capabilities of the system and the actual purchase behavior of the clients underscores the effectiveness and precision of the recommendation model. In sum, this visual representation and analysis of the test set results offer a valuable glimpse into the performance of our recommendation system. The observations made here not only emphasize the congruence between predictions and actual purchases but also highlight the model's proficiency in ranking recommendations by relevance, thereby enhancing its practical utility for end-users.

Those results comes from an observation of a test set sample, moving to the whole test set,

the analysis of clients who purchased at least one items from the recommended ones can be defined as the percentage of true positive predictions which in statistics is called true positive rate (TPR), also known as sensitivity or recall. The TPR is basically a measure of , thus in the case of the model in question products that the model successfully recommends to users. The value of TPR, in this case, stands at 0.6246, signifying that the recommendation system excels in identifying items that align with users' preferences. Specifically, it denotes that approximately 62.46% of relevant items, items that users would find appealing, are correctly suggested by the system. This is a pivotal metric as it directly speaks to the system's ability to enhance user satisfaction by presenting them with items they are more likely to engage with or purchase. A high TPR value indicates that the recommendation algorithm is adept at minimizing missed opportunities, ensuring that a significant portion of relevant items is included in the user's personalized recommendations. This is crucial for user engagement and can contribute to increased user loyalty and higher conversion rates. TPR value of 0.6246 reflects the recommendation system's proficiency in identifying items that genuinely resonate with users' preferences. This metric's significance lies in its direct impact on user satisfaction, user engagement, and ultimately, the success of the recommendation system. It demonstrates the system's capability to effectively navigate the vast landscape of available items to pinpoint those that truly matter to users, enhancing their overall experience.

In conclusion, the evaluation of this recommendation system has provided valuable insights into its performance and its potential impact. The results indicate that the system excels in offering personalized recommendations to users, significantly enhancing their browsing and shopping experiences. It is evident that the recommendations generated by the model resonate with user preferences, as seen in the high rate of successful recommendations. Moreover, the rigorous selection criteria for the test set, focusing on clients who have made multiple purchases and considering their most recent interactions, underscores the commitment to assessing the system's performance in real-world scenarios.

These results are not only a testament to the effectiveness of the recommendation algorithm but also a reflection of dedication to continuously improving user experiences and driving business outcomes. The positive outcomes observed in this evaluation lay the foundation for future enhancements and optimizations.

Chapter 5

Managerial Implications and Conclusion

The managerial and practical implications of this research are pivotal in understanding the real-world value of our recommendation system and delineating possible directions for the implementation and enhancement of business strategies. Throughout the data analysis and the evaluation of our system's performance, several key considerations have come to the forefront, which can be applied to enhance the effectiveness and efficiency of business activities. One of the most conspicuous outcomes is the effectiveness of this system in providing personalized recommendations. This implies a significant enhancement in the customer experience, resulting in increased satisfaction and loyalty. Business managers can employ this insight to design and implement strategies aimed at further enhancing the customer experience, such as personalized loyalty programs or targeted communications. Data analysis has also revealed the efficiency of recommendations in influencing user purchase decisions. This implies that marketing resources can be allocated more judiciously, focusing efforts on products or services with a higher likelihood of user adoption. Managers can use this information to optimize marketing budgets and maximize return on investment (ROI). The dynamic nature of data analysis and this recommendation system enables rapid adaptation to market trends and shifts in user behavior. This flexibility is crucial for remaining competitive in an ever-evolving environment. Managers can employ this adaptability to stay at the forefront of their industry and respond promptly to customer needs. The positive results obtained also indicate the possibility of developing new business opportunities based on recommendations. For instance, managers can explore the possibility of offering consulting services based on user preferences or forming strategic partnerships with recommended product manufacturers. These new opportunities can contribute to diversifying business revenue streams and increasing profitability.

The primary objective of a recommendation system extends far beyond merely suggesting items

that align with a customer's historical preferences. It goes deep into the domain of influencing customer choices and behaviour so as to improve not only user satisfaction, but also company growth and competitiveness. This paradigm shift in the role of recommendation systems is rooted in the understanding that they possess the power to shape consumer decisions, foster brand loyalty, and drive revenue generation in multiple ways. At its core, a recommendation system leverages data-driven insights to create a personalized shopping experience. However, this personalization goes beyond catering to known preferences; it serves as a persuasive tool. By intelligently recommending items that align with a user's interests, the system can nudge customers towards exploring new products or services they might not have considered otherwise. This persuasive element capitalizes on the cognitive biases and behavioral psychology principles that underpin human decision-making. Recommendation systems are adept at cross-selling and upselling, strategies that directly impact the average transaction value. By suggesting complementary or higher-end products, they can substantially increase the revenue generated from each customer. For instance, a recommendation system in an e-commerce platform might propose accessories or upgrades alongside a selected product, thereby influencing the customer to make a more substantial purchase. Furthermore, these systems have the capability to align recommendations with seasonal trends, special occasions, or even cultural events. By adapting to these factors, they can motivate customers to make timely purchases that they might otherwise have overlooked. This adaptive approach ensures that businesses stay relevant and responsive to market dynamic.

Recommendation systems are not static; they are in a state of continuous learning. They analyze user behavior and adapt recommendations accordingly. This adaptability can be harnessed to reinforce certain buying patterns, introduce customers to new product categories, or foster brand loyalty by suggesting items that resonate with a user's evolving tastes. Beyond their influence on individual customers, recommendation systems contribute to a company's competitive advantage. Businesses that can effectively guide and influence their customers' choices through tailored recommendations gain a significant edge in the market. They build stronger customer relationships, foster trust, and engender a sense of brand affinity that can be challenging for competitors to replicate.

The potential impact of recommendation systems within marketing strategies is significant and multidimensional. These tools not only offer significant value for improving the customer experience, but also represent a valuable resource for corporate strategies. A company can harness the power of recommendation systems in several ways to achieve key marketing objectives. One of the main strategies is the personalisation of communications. Using the information gathered through recommendation systems, a company can create targeted and highly personalised marketing campaigns. For example, it can send newsletters or promotional messages

to customers with specific recommendations based on their past tastes and behaviour. This not only increases the relevance of communications, but also the likelihood of conversion, as customers are more likely to engage with products or services that match their interests. In addition, recommendation systems can be used to optimise inventory and stock management. If a company has a surplus of a certain product, it can identify customers who are most inclined to buy it and offer them special incentives to complete the purchase. This helps free up warehouse space and maximise sales. Finally, the data generated by recommendation systems can be used to analyse customer purchasing trends and to develop long-term marketing strategies. Information on preferred products, buying seasons and customer segments can guide strategic decisions such as stock planning, new product design and the development of long-term marketing campaigns. Among others, one of the key applications is the customisation of product pages, where details and information are presented according to individual customer tastes. But the innovation does not stop there. Recommender systems can exploit localisation to offer location-based suggestions, create personalised user experiences, promote user-generated content and even influence purchasing decisions through psychological persuasion strategies. The opportunities are manifold: from customising the user interface to supporting targeted subscriptions, from promoting complementary products to highly targeted marketing campaigns. Furthermore, recommendation systems can foster social interaction and the sharing of recommendations between users, increasing the reach of marketing strategies. Overall, these systems stand as versatile and indispensable tools in the modern marketing arsenal, not only offering suggestions, but also actively shaping customer choices and opening new avenues for business growth. Their ability to understand customer behaviour and exploit cognitive biases transforms them from mere assistants into influencers, offering a cutting-edge vision for corporate marketing strategies.

In conclusion, the findings of this research underscore the intrinsic value of an effective and personalized recommendation system within the world of marketing and e-commerce. The managerial and practical implications stemming from these results provide a robust foundation for optimizing business strategies, enhancing the customer experience, resource optimization, and opening up new avenues for growth. In light of these findings, the implementation of an intelligent recommendation system emerges as a fundamental element for business success in today's competitive landscape. Furthermore, the true power of a recommendation system lies in its capacity to be an active agent in shaping customer choices. While it certainly aims to facilitate convenient and relevant product discovery, it goes a step further by wielding the psychology of persuasion. By understanding customer behavior and leveraging cognitive biases, recommendation systems become not just assistants but influencers. This dual role, where technology seamlessly blends with behavioral science, not only elevates user experiences but also

fuels business growth, revenue enhancement, and market competitiveness. In essence, recommendation systems emerge as pivotal instruments in the strategic arsenal of modern businesses, transcending mere suggestions to become influential partners in customer decision-making.

Python Code

Import Libraries

```
#Librares for data manipulation
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from datetime import datetime

#Libraries for implementing ALS model
import os; os.environ['OPENBLAS_NUM_THREADS']='1'
import implicit
from scipy.sparse import coo_matrix
from implicit.evaluation import mean_average_precision_at_k
```

Load Datasets

```
products = pd.read_csv('C:/Users/vincenzo.terruli/Desktop/LUISS/tesi/articles.csv',
    dtype={'article_id': str})
clients = pd.read_csv('C:/Users/vincenzo.terruli/Desktop/LUISS/tesi/customers.csv')
trans = pd.read_csv('C:/Users/vincenzo.terruli/Desktop/LUISS/tesi/transactions_train.
    csv', dtype={'article_id': str})
```

EDA - Products

```
products.head(5)

products.info()

products.product_code.nunique()

products.prod_name.nunique()

#47224 unique identifier for every product. These variables have the same meaning but
     some product have name that differ for some cues.

products.product_type_no.nunique()

products.product_type_name.nunique()

products.product_group_name.nunique()

temp = products.groupby(["product_group_name"])["article_id"].nunique()
df = pd.DataFrame({'Product Group': temp.index,'Articles': temp.values})
df = df.sort_values(['Articles'], ascending=False)
plt.figure(figsize = (8,6))
plt.title('Number of Articles per each Product Group')
sns.set_color_codes("pastel")
```

```
s = sns.barplot(x = 'Product Group', y="Articles", data=df)
s.set_xticklabels(s.get_xticklabels(),rotation=90)
plt.show()

#There exist 19 Macrocategories. TOP 4 Macrocategories are Upper body; Lower body;
    Full body; Accessories

df=products.groupby('product_type_name').article_id.nunique()
df.sort_values(ascending=False).head(10)

#132 categories. Top 4 categories are: Trousers; Dress; Sweater; T-shirt.

products.department_no.nunique()

df=products.groupby('department_name').article_id.nunique()
df.sort_values(ascending=False).head(15)

#299 subcategories

products.graphical_appearance_no.nunique()

df=products.groupby('graphical_appearance_name').article_id.nunique()
df.sort_values(ascending=False)

#30 different graphics. Most of the products have SOLID graphic

products.colour_group_code.nunique()

df=products.groupby('colour_group_name').article_id.nunique()
df.sort_values(ascending=False).head(15)

#50 different colour. Black, Dark Blue and White are the principal. "perceived_colour
    " variables are others colour info

products.index_code.nunique()

temp = products.groupby(["index_name"])["article_id"].nunique()
df = pd.DataFrame({'SubSector Group': temp.index,'Articles': temp.values})
df = df.sort_values(['Articles'], ascending=False)
plt.figure(figsize = (8,6))
plt.title('Number of Articles per each Sector Group')
sns.set_color_codes("pastel")
s = sns.barplot(x = 'SubSector Group', y="Articles", data=df)
s.set_xticklabels(s.get_xticklabels(),rotation=90)
plt.show()

products.index_group_no.nunique()
```

```

temp = products.groupby(["index_group_name"])["article_id"].nunique()
df = pd.DataFrame({'Sector Group': temp.index,'Articles': temp.values})
df = df.sort_values(['Articles'], ascending=False)
plt.figure(figsize = (8,6))
plt.title('Number of Articles per each Sector Group')
sns.set_color_codes("pastel")
s = sns.barplot(x = 'Sector Group', y="Articles", data=df)
s.set_xticklabels(s.get_xticklabels(),rotation=90)
plt.show()

#There are 5 Sector group. Ladieswear and Baby are the principal.

counts = products.groupby(['garment_group_name', 'index_group_name']).size().unstack(
    fill_value=0)

total_counts = counts.sum(axis=1)

sorted_counts = counts.loc[total_counts.sort_values(ascending=True).index]

plt.figure(figsize=(18, 18))
sorted_counts.plot(kind='barh', stacked=True, color=sns.color_palette('Set3'))
plt.xlabel('Index Group')
plt.ylabel('Count')
plt.title('Relationship between index_group_name and garment_group_name')
plt.show()

products.groupby(['product_group_name', 'product_type_name']).count()['article_id']
```

EDA - Client

```
clients.info()

clients.head(5)

df=clients.groupby('club_member_status').customer_id.nunique()
df.sort_values(ascending=False).head(15)

temp = clients.groupby(["club_member_status"])["customer_id"].nunique()
df = pd.DataFrame({'H&M Club': temp.index,'Clients': temp.values})
df = df.sort_values(['Clients'], ascending=False)
plt.figure(figsize = (8,6))
plt.title('Number of Clients per each Member Status')
sns.set_color_codes("pastel")
s = sns.barplot(x = 'H&M Club', y="Clients", data=df)
plt.show()

#Most of Clients are active. Some of them are being active.
```

```
18
19 f, ax = plt.subplots(figsize=(10,5))
20 ax = sns.histplot(data=clients, x='age', bins=50, color='orange')
21 ax.set_xlabel('Distribution of the customers age')
22 plt.show()
23
24 bins = [0, 30, 60, 120]
25 labels = ['Under 30', '30-60', 'Over 60']
26 clients['age_range'] = pd.cut(clients.age, bins, labels = labels,include_lowest =
       True)
27
28 df=clients.groupby('age_range').customer_id.nunique()
29 df.sort_values(ascending=False).head(15)
30
31 sns.set_style("darkgrid")
32 f, ax = plt.subplots(figsize=(10, 5))
33 colors = sns.color_palette('Set2')
34 id_news = clients[['customer_id', 'age_range']].groupby('age_range')['customer_id'].
       count()
35
36 percentages = id_news / id_news.sum() * 100
37
38 ax.pie(id_news, labels=id_news.index, colors=colors, autopct='%1.2f%%')
39 ax.set_facecolor('lightgrey')
40 ax.set_xlabel('Age Range Distribution')
41 plt.show()
42
43 clients.groupby('postal_code', as_index=False).count().sort_values('customer_id',
       ascending=False).head(10)
44
45 #One postal code includes many clients. Probably we can use it as N\A value
46
47 clients.loc[~clients['fashion_news_frequency'].isin(['Regularly', 'Monthly']), '
       fashion_news_frequency'] = 'None'
48 df=clients.groupby('fashion_news_frequency').customer_id.nunique()
49 df.sort_values(ascending=False).head(15)
50
51 sns.set_style("darkgrid")
52 f, ax = plt.subplots(figsize=(10, 5))
53 colors = sns.color_palette('Set2')
54 id_news = clients[['customer_id', 'fashion_news_frequency']].groupby('
       fashion_news_frequency')['customer_id'].count()
55
56 percentages = id_news / id_news.sum() * 100
57
58 ax.pie(id_news, labels=id_news.index, colors=colors, autopct='%1.2f%%')
59 ax.set_facecolor('lightgrey')
```

```
ax.set_xlabel('Distribution of fashion news frequency')

plt.show()

#In order to reduce the transaction file dimension i will add a new column in the
    Clients dataset in order to have a ID of maximum 10 character

clients['client_id'] = [str(i).zfill(10) for i in range(1, len(clients) + 1)]
```

Data Manipulation and Featuring Engineering

```
# Let's create a unique dataset which contains all variables

products_for_merge = products[['article_id', 'prod_name', 'product_group_name','
    product_type_name','garment_group_name', 'index_group_name']]
clients_for_merge = clients[['customer_id', 'client_id', 'club_member_status','
    fashion_news_frequency', 'age']]
trans_2 = pd.merge(pd.merge(trans, clients_for_merge, on='customer_id', how='inner'),
    products_for_merge,on='article_id', how='inner')

trans_2.drop('customer_id', axis=1, inplace=True)
columns = trans_2.columns.tolist()
columns = ['client_id'] + [col for col in columns if col != 'client_id']
trans_2 = trans_2[columns]

# Here "transaction_key" is the variable that register the receipts for each clients

for x in trans_2:
    client_id = trans_2["client_id"]
    date = trans_2["t_dat"]
    receipt_key = client_id + "_" + date
    trans_2["transaction key"] = receipt_key

# The items's price has been adjusted to be analyzed in a real-world context

trans_2['price']=trans_2['price']*1000
```

EDA - Transaction

```
trans_2.info()

trans_2.head(10)

trans_2['t_dat'] = pd.to_datetime(trans_2['t_dat'])
print("Period of analysis")
print("First Transaction:", trans_2['t_dat'].min())
print("Last Transaction:", trans_2['t_dat'].max())

print("Price range")
```

```
print("Min Price:", trans_2['price'].min())
print("Max Price:", trans_2['price'].max()

df_grouped = trans_2.groupby('age').agg({'client_id': pd.Series.nunique,'transaction
    key': pd.Series.nunique, 'price': 'mean'})
df_grouped['avg_trans'] = df_grouped['transaction key']/df_grouped['client_id']

plt.figure(figsize=(10, 6))
plt.subplot(2, 1, 1)
plt.plot(df_grouped.index, df_grouped['avg_trans'], color="Darkgreen")
plt.xlabel('Età')
plt.ylabel('Numero medio di transazioni')
plt.title('Numero medio di transazioni per età')

plt.subplot(2, 1, 2)
plt.plot(df_grouped.index, df_grouped['price'], color="Darkgreen")
plt.xlabel('Età')
plt.ylabel('Spesa media')
plt.title('Spesa media per età')

plt.tight_layout()
plt.show()

trans_2['year_month'] = trans_2['t_dat'].dt.to_period('M')
trans_2['year_month'] = trans_2['year_month'].dt.strftime('%Y-%m')

#Transaction per day

df = trans_2.groupby(["t_dat"])["transaction key"].nunique().reset_index()
df['t_dat'] = pd.to_datetime(df['t_dat'])
df.columns = ["Date", "Transactions"]
fig, ax = plt.subplots(1, 1, figsize=(16,6))
plt.plot(df["Date"], df["Transactions"], color="Darkgreen")
plt.xlabel("Date")
plt.ylabel("Transactions")
plt.show()

df = trans_2.groupby(["year_month"])["transaction key"].nunique().reset_index()
df['year_month'] = pd.to_datetime(df['year_month'])
df.columns = ["Date", "Transactions"]
fig, ax = plt.subplots(1, 1, figsize=(16,6))
plt.plot(df["Date"], df["Transactions"], color="Darkgreen")
plt.xlabel("Date")
plt.ylabel("Transactions")
plt.show()

df = trans_2.groupby(["t_dat", "sales_channel_id"])["transaction key"].nunique().
```

```
    reset_index()
df['t_dat'] = pd.to_datetime(df['t_dat'])
df.columns = ["Date", "Sales Channel Id", "Transactions"]
fig, ax = plt.subplots(1, 1, figsize=(16,6))
g1 = ax.plot(df.loc[df["Sales Channel Id"]==1, "Date"], df.loc[df["Sales Channel Id"
    ]==1, "Transactions"], label="Retail", color="Darkblue")
g2 = ax.plot(df.loc[df["Sales Channel Id"]==2, "Date"], df.loc[df["Sales Channel Id"
    ]==2, "Transactions"], label="E-commerce", color="Magenta")
plt.xlabel("Date")
plt.ylabel("Transactions")
ax.legend()
plt.title("Transactions per day, grouped by Sales Channel")
plt.show()

df = trans_2.groupby(["year_month", "sales_channel_id"])["transaction key"].nunique()
    .reset_index()
df['year_month'] = pd.to_datetime(df['year_month'])
df.columns = ["Date", "Sales Channel Id", "Transactions"]
fig, ax = plt.subplots(1, 1, figsize=(16,6))
g1 = ax.plot(df.loc[df["Sales Channel Id"]==1, "Date"], df.loc[df["Sales Channel Id"
    ]==1, "Transactions"], label="Retail", color="Darkblue")
g2 = ax.plot(df.loc[df["Sales Channel Id"]==2, "Date"], df.loc[df["Sales Channel Id"
    ]==2, "Transactions"], label="E-commerce", color="Magenta")
plt.xlabel("Date")
plt.ylabel("Transactions")
ax.legend()
plt.title("Transactions per day, grouped by Sales Channel")
plt.show()

#Online vs Retail Trends

df_grouped = trans_2.groupby('sales_channel_id').agg({'transaction key': pd.Series.
    nunique, 'price': 'sum'})
df_grouped['avg_spending'] = df_grouped['price']/df_grouped['transaction key']
df_grouped.plot(y="avg_spending",kind='bar')

df_grouped = trans_2.groupby('club_member_status').agg({'transaction key': pd.Series.
    nunique, 'price': 'sum'})
df_grouped['avg_spending'] = df_grouped['price']/df_grouped['transaction key']
df_grouped.plot(y="avg_spending",kind='bar')

df_grouped = trans_2.groupby('fashion_news_frequency').agg({'transaction key': pd.
    Series.nunique, 'price': 'sum'})
df_grouped['avg_spending'] = df_grouped['price']/df_grouped['transaction key']
df_grouped.plot(y="avg_spending",kind='bar')

#Weekend vs Weekday
```

```
95
96  trans_2['t_dat'] = pd.to_datetime(trans_2['t_dat'])
97  trans_2['day_type'] = np.where(trans_2['t_dat'].dt.dayofweek < 5, 'Weekday', 'Weekend
        ')
98  df_grouped = trans_2.groupby('day_type').agg({'transaction key': pd.Series.nunique, '
        t_dat': pd.Series.nunique, 'price': 'sum'})
99  df_grouped['avg_spending'] = df_grouped['price']/df_grouped['transaction key']
100 df_grouped['avg_transaction'] = df_grouped['transaction key']/df_grouped['t_dat']
101 df_grouped.plot(y="avg_transaction",kind='bar')
102 df_grouped.plot(y="avg_spending",kind='bar')
103
104 Transactions analysis
105
106 #Which are the TOP 10 articles in terms of sold quantity?
107
108 df_grouped = trans_2.groupby('prod_name').agg({'transaction key': pd.Series.nunique,
        'price': 'sum'})
109 top_products = df_grouped.sort_values(by='transaction key', ascending=False).head(10)
110 top_products = top_products.sort_values(by='transaction key', ascending=True)
111 plt.figure(figsize=(10, 6))
112 top_products.plot(y="transaction key",kind='barh')
113 plt.xlabel('Numero di Transazioni')
114 plt.ylabel('Nome Prodotto')
115 plt.title('Top 10 Prodotti per Numero di Transazioni')
116 plt.tight_layout()
117 plt.show()
118
119 df_grouped = trans_2.groupby('product_type_name').agg({'transaction key': pd.Series.
        nunique, 'price': 'sum'})
120 top_products = df_grouped.sort_values(by='transaction key', ascending=False).head(10)
121 top_products = top_products.sort_values(by='transaction key', ascending=True)
122 plt.figure(figsize=(10, 6))
123 top_products.plot(y="transaction key",kind='barh')
124 plt.xlabel('Numero di Transazioni')
125 plt.ylabel('Categoria Prodotto')
126 plt.title('Top 10 Prodotti per Numero di Transazioni')
127 plt.tight_layout()
128 plt.show()
129
130 #Which are the TOP 10 articles that generated most earnings for the company?
131
132 df_grouped = trans_2.groupby('prod_name').agg({'transaction key': pd.Series.nunique,
        'price': 'sum'})
133 top_products = df_grouped.sort_values(by='price', ascending=False).head(10)
134 top_products = top_products.sort_values(by='price', ascending=True)
135 plt.figure(figsize=(10, 6))
136 top_products.plot(y="price",kind='barh')
```

```
plt.xlabel('Turnover Generato')
plt.ylabel('Prodotto')
plt.title('Top 10 Prodotti per Turnover Generato')
plt.tight_layout()
plt.show()

df_grouped = trans_2.groupby('product_type_name').agg({'transaction key': pd.Series.
    nunique, 'price': 'sum'})
top_products = df_grouped.sort_values(by='price', ascending=False).head(10)
top_products = top_products.sort_values(by='price', ascending=True)
plt.figure(figsize=(10, 6))
top_products.plot(y="price",kind='barh')
plt.xlabel('Turnover Generato')
plt.ylabel('Categoria Prodotto')
plt.title('Top 10 Prodotti per Turnover Generato')
plt.tight_layout()
plt.show()

df_grouped = trans_2.groupby('article_id').agg({'transaction key': pd.Series.nunique,
     'price': 'sum'})
df_grouped['avg_price'] = df_grouped['price']/df_grouped['transaction key']

plt.figure(figsize=(18, 6))
plt.scatter(df_grouped['avg_price'], df_grouped['transaction key'])
plt.xlabel('Prezzo dell\'articolo')
plt.ylabel('Numero di transazioni')
plt.title('Scatterplot: Prezzo dell\'articolo vs Numero di transazioni')
plt.show()

#Price outliers

sns.set_style("darkgrid")
f, ax = plt.subplots(figsize=(10,5))
ax = sns.boxplot(data=trans_2, x='price', color='orange')
ax.set_xlabel('Price outliers')
plt.show()

sns.set_style("darkgrid")
f, ax = plt.subplots(figsize=(12,8))
ax = sns.boxplot(data=trans_2, x='price', y='product_group_name')
ax.set_xlabel('Price outliers', fontsize=22)
ax.set_ylabel('Index names', fontsize=22)
ax.xaxis.set_tick_params(labelsize=22)
ax.yaxis.set_tick_params(labelsize=22)
plt.show()

sns.set_style("darkgrid")
```

```
f, ax = plt.subplots(figsize=(12,10))
_ = trans_2[trans_2['product_group_name'] == 'Accessories']
ax = sns.boxplot(data=_, x='price', y='product_type_name')
ax.set_xlabel('Price outliers', fontsize=22)
ax.set_ylabel('Index names', fontsize=22)
ax.xaxis.set_tick_params(labelsize=22)
ax.yaxis.set_tick_params(labelsize=22)
del _
plt.show()
```

Data selection

```
#Decrease testset size

#Select only top 400 most frequently bought articles. Get only customers with at least 3 transactions.

TOP_N = 400
most_frequent_articles = trans_2["article_id"].value_counts().reset_index()
most_frequent_articles.columns = ["article_id", "count"]
most_frequent_articles =  most_frequent_articles.sort_values(by='count', ascending=False).head(TOP_N)
most_frequent_articles =  most_frequent_articles["article_id"]

trans_2 = trans_2[trans_2["article_id"].isin(most_frequent_articles)].reset_index(drop=True)

df_grouped = trans_2.groupby(["client_id","article_id"])["t_dat"].count().reset_index()
df_grouped.columns = ["client_id","article_id", "purchase_count"]
df_grouped = df_grouped[df_grouped['purchase_count'] >= 3]
df_grouped =  df_grouped["client_id"]

trans_final = trans_2[trans_2["client_id"].isin(df_grouped)].reset_index(drop=True)

trans_final.info()
```

Model Implementation

```
ALL_USERS = clients['client_id'].unique().tolist()
ALL_ITEMS = products['article_id'].unique().tolist()

user_ids = dict(list(enumerate(ALL_USERS)))
item_ids = dict(list(enumerate(ALL_ITEMS)))

user_map = {u: uidx for uidx, u in user_ids.items()}
item_map = {i: iidx for iidx, i in item_ids.items()}

trans_final['user_id'] = trans_final['client_id'].map(user_map)
```

```
trans_final['item_id'] = trans_final['article_id'].map(item_map)

row = trans_final['user_id'].values
col = trans_final['item_id'].values
data = np.ones(trans_final.shape[0])

coo_train = coo_matrix((data, (row, col)), shape=(len(ALL_USERS), len(ALL_ITEMS)))

coo_train
```

Model Training

```
%%time
model = implicit.als.AlternatingLeastSquares(factors=10, iterations=2)
model.fit(coo_train)
```

Validation Set and Hyperparameters tuning

```
def to_user_item_coo(df):
    """ Turn a dataframe with transactions into a COO sparse items x users matrix"""
    row = df['user_id'].values
    col = df['item_id'].values
    data = np.ones(df.shape[0])
    coo = coo_matrix((data, (row, col)), shape=(len(ALL_USERS), len(ALL_ITEMS)))
    return coo


def split_data(df, validation_days=14):
    """ Split a pandas dataframe into training, validation, and test data, using <<
    validation_days>>.
    """
    # Trova la data massima per ciascun client_id
    test_idx = df.groupby('client_id')['t_dat'].idxmax()
    test_set = df.loc[test_idx]

    # Rimuovi le righe del test set dal dataframe originale
    df = df.drop(test_idx)

    # Calcola la data di cut per la validazione
    validation_cut = test_set['t_dat'].max() - pd.Timedelta(validation_days)

    # Dividi il dataframe in training e validation
    df_train = df[df['t_dat'] < validation_cut]
    df_val = df[df['t_dat'] >= validation_cut]

    return df_train, df_val, test_set

def get_val_matrices(df, validation_days=14):
    """ Split into training and validation and create various matrices
```

```
31
32          Returns a dictionary with the following keys:
33              coo_train: training data in COO sparse format and as (users x items)
34              csr_train: training data in CSR sparse format and as (users x items)
35              csr_val:  validation data in CSR sparse format and as (users x items)
36
37      """
38      df_train, df_val, test_set = split_data(df, validation_days=validation_days)
39      coo_train = to_user_item_coo(df_train)
40      coo_val = to_user_item_coo(df_val)
41      coo_test = to_user_item_coo(test_set)
42
43      csr_train = coo_train.tocsr()
44      csr_val = coo_val.tocsr()
45      csr_test = coo_test.tocsr()
46
47      return {'coo_train': coo_train,
48              'csr_train': csr_train,
49              'csr_val': csr_val,
50              'csr_test': csr_test
51            }
52
53
54  def validate(matrices, factors=200, iterations=20, regularization=0.01, show_progress
        =True):
55      """ Train an ALS model with <<factors>> (embeddings dimension)
56      for <<iterations>> over matrices and validate with MAP@12
57      """
58      coo_train, csr_train, csr_val = matrices['coo_train'], matrices['csr_train'],
        matrices['csr_val']
59
60      model = implicit.als.AlternatingLeastSquares(factors=factors,
61                                                   iterations=iterations,
62                                                   regularization=regularization,
63                                                   random_state=42)
64      model.fit(coo_train, show_progress=show_progress)
65
66      map12 = mean_average_precision_at_k(model, csr_train, csr_val, K=12,
        show_progress=show_progress, num_threads=4)
67      print(f"Factors: {factors:>3} - Iterations: {iterations:>2} - Regularization: {
        regularization:4.3f} ==> MAP@12: {map12:6.5f}")
68      return map12
69
70  trans_final['t_dat'] = pd.to_datetime(trans_final['t_dat'])
71
72  matrices = get_val_matrices(trans_final)
73
```

```
%%time
best_map12 = 0
for factors in [40, 50, 60, 100, 200, 500]:
    for iterations in [3, 12, 14, 15, 20]:
        for regularization in [0.01]:
            map12 = validate(matrices, factors, iterations, regularization, show_progress=False)
            if map12 > best_map12:
                best_map12 = map12
                best_params = {'factors': factors, 'iterations': iterations, 'regularization': regularization}
                print(f"Best MAP@12 found. Updating: {best_params}")

coo_train = to_user_item_coo(trans_final)
csr_train = coo_train.tocsr()

best_params

def train(coo_train, factors=50, iterations=15, regularization=0.01, show_progress=True):
    model = implicit.als.AlternatingLeastSquares(factors=factors,
                                                 iterations=iterations,
                                                 regularization=regularization,
                                                 random_state=42)
    model.fit(coo_train, show_progress=show_progress)
    return model

model = train(coo_train, **best_params)
```

Recommendation prediction

```
def submit(model, csr_train, submission_name="submissions.csv"):
    preds = []
    batch_size = 2000
    to_generate = np.arange(len(ALL_USERS))
    for startidx in range(0, len(to_generate), batch_size):
        batch = to_generate[startidx : startidx + batch_size]
        ids, scores = model.recommend(batch, csr_train[batch], N=12, filter_already_liked_items=False)
        for i, userid in enumerate(batch):
            customer_id = user_ids[userid]
            user_items = ids[i]
            article_ids = [item_ids[item_id] for item_id in user_items]

            # Aggiungi ogni 'article_id' come una nuova colonna al posto della colonna 'prediction'
            preds.append([customer_id] + article_ids)

```

```
    # Crea un DataFrame con le 13 colonne
    col_names = ['client_id'] + [f'prediction_{i}' for i in range(12)]
    df_preds = pd.DataFrame(preds, columns=col_names)
    df_preds.to_csv(submission_name, index=False)

    display(df_preds.head())
    print(df_preds.shape)

    return df_preds

%%time
df_preds_final = submit(model, csr_train);
```

Evaluation

```
df_train, df_val, test_set = split_data(trans_final, validation_days=7)
clients_test =  test_set["client_id"]
df_preds_2 = df_preds_final[df_preds_final["client_id"].isin(clients_test)].
    reset_index(drop=True)
df_preds_2.head(15)

columns_to_keep = ['client_id', 'article_id']  # Lista delle colonne da mantenere
test_set = test_set[columns_to_keep]

merged_df = test_set.merge(df_preds_2, on='client_id', how='inner')

def has_article(article_id, predictions):
    return any(article_id == pred for pred in predictions)

prediction_cols = [f'prediction_{i}' for i in range(12)]
merged_df['true_prediction'] = merged_df.apply(lambda row: has_article(row['
    article_id'], row[prediction_cols]), axis=1)

count_true_false = merged_df['true_prediction'].value_counts()
print("Numero di TRUE:", count_true_false[True])
print

matrices = get_val_matrices(trans_final)

coo_train, csr_train, csr_val, csr_test = matrices['coo_train'],

matrices['csr_train'], matrices['csr_val'], matrices['csr_test']

#MAP@K
map_k12 = mean_average_precision_at_k(model, csr_train, csr_test, K=12)


#NDCG
```

```
NDCG = implicit.evaluation.ndcg_at_k(model, csr_train, csr_test, K=12)

#AUC
AUC = implicit.evaluation.AUC_at_k(model, csr_train, csr_test, K=12)

#TPR
true_positives = 0
false_positives = 0
false_negatives = 0
true_negatives = 0

for index, row in merged_df.iterrows():
    true_item = row['article_id']  # Item effettivamente acquistato dal cliente
    recommended_items = [row[f'prediction_{i}'] for i in range(12)]  # Lista degli 12
     item raccomandati

    # Verifica se l'item effettivamente acquistato è presente tra quelli raccomandati
    if true_item in recommended_items:
        true_positives += 1
    else:
        false_negatives += 1

total_recommendations = len(test_set) * 12
false_positives = total_recommendations - true_positives

true_negatives = total_recommendations - false_positives - true_positives

tpr = true_positives / (true_positives + false_negatives)
fpr = false_positives / (false_positives + true_negatives)

print(f"True Positive Rate (TPR): {tpr:.4f}")
print(f"False Positive Rate (FPR): {fpr:.4f}")
```

References

[1] Joseph A. Konstan and John Riedl. "Recommender systems: from algorithms to user experience". In: *User Modeling and User-Adapted Interaction* 22 (2012). ISSN: 1110-8665. DOI: https://doi.org/10.1007/s11257-011-9112-x. URL: https://www.sciencedirect.com/science/article/pii/S1110866515000341.

[2] Asbjørn Følstad and Knut Andreas Kvåle. "Customer journeys: a systematic literature review". In: *Journal of Service Theory and Practice* 28 (2018), pp. 196–227. URL: https://api.semanticscholar.org/CorpusID:55967336.

[3] Katherine N. Lemon and Peter C. Verhoef. "Understanding Customer Experience Throughout the Customer Journey". In: *Journal of Marketing* 80 (2016), pp. 69–96. URL: https://api.semanticscholar.org/CorpusID:168279405.

[4] Costanza Nosi. *Consumer Journey. Gestire strategicamente la customer experience per competere*. Franco Angeli, 2019.

[5] Scott Belsky. *The Messy Middle: Finding Your Way Through the Hardest and Most Crucial Part of Any Bold Venture*. Penguin, 2018.

[6] Stephanie Chevalier. *Retail e-commerce sales worldwide from 2014 to 2026*. 2022. URL: https://www.statista.com/statistics/379046/worldwide-retail-e-commerce-sales/.

[7] Statista Research Department. *Fashion e-commerce worldwide - statistics & facts*. 2023. URL: https://www.statista.com/topics/9288/fashion-e-commerce-worldwide/#topicOverview.

[8] Liat Hadar and Sanjay Sood. "When Knowledge Is Demotivating". In: *Psychological science* 25 (July 2014). DOI: 10.1177/0956797614539165.

[9] Ben Schafer, Joseph Konstan, and John Riedl. "Recommender Systems in E-Commerce". In: *1st ACM Conference on Electronic Commerce, Denver, Colorado, United States* (Oct. 1999). DOI: 10.1145/336992.337035.

[1] Kim Falk. *Practical Recommender Systems*. Manning Publications, 2019.

[2] F.O. Isinkaye, Y.O. Folajimi, and B.A. Ojokoh. "Recommendation systems: Principles, methods and evaluation". In: *Egyptian Informatics Journal* 16.3 (2015), pp. 261–273. ISSN: 1110-8665. DOI: https://doi.org/10.1016/j.eij.2015.06.005. URL: https://www.sciencedirect.com/science/article/pii/S1110866515000341.

[3] Shah Khusro, Zafar Ali, and Irfan Ullah. "Recommender Systems: Issues, Challenges, and Research Opportunities". In: Feb. 2016, pp. 1179–1189. ISBN: 978-981-10-0556-5. DOI: 10.1007/978-981-10-0557-2_112.

[4] Gawesh Jawaheer, M. Szomszor, and Patty Kostkova. "Comparison of implicit and explicit feedback from an online music recommendation service". In: *HetRec '10*. 2010. URL: https://api.semanticscholar.org/CorpusID:14111790.

[5] Farhin Mansur, Vibha Patel, and Mihir Patel. "A review on recommender systems". In: Mar. 2017, pp. 1–6. DOI: 10.1109/ICIIECS.2017.8276182.

[6] Mayukh Bhattacharyya. *Metrics of Recommender Systems: An Overview*. 2022. URL: https://towardsdatascience.com/metrics-of-recommender-systems-cde64042127a.

[7] Ian MacKenzie, Chris Meyer, and Steve Noble. *How retailers can keep up with consumers*. 2023. URL: https://www.mckinsey.com/industries/retail/our-insights/how-retailers-can-keep-up-with-consumers.

[8] Cem Dilmegani. *Recommendation Systems: Applications and Examples in 2023*. 2023. URL: https://research.aimultiple.com/recommendation-system/.

[9] Himanshu Kriplani. *Alternating Least Square for Implicit Dataset with code*. 2019. URL: https://towardsdatascience.com/alternating-least-square-for-implicit-dataset-with-code-8e7999277f4b.

[10] Gábor Takács and Domonkos Tikk. "Alternating Least Squares for Personalized Ranking". In: *Proceedings of the Sixth ACM Conference on Recommender Systems*. RecSys '12. Dublin, Ireland: Association for Computing Machinery, 2012, pp. 83–90. ISBN: 9781450312707. DOI: 10.1145/2365952.2365972. URL: https://doi.org/10.1145/2365952.2365972.

[11] Subasish Gosh et al. "Recommendation System for E-commerce Using Alternating Least Squares (ALS) on Apache Spark". In: *Intelligent Computing and Optimization*. Ed. by Pandian Vasant, Ivan Zelinka, and Gerhard-Wilhelm Weber. Cham: Springer International Publishing, 2021, pp. 880–893. ISBN: 978-3-030-68154-8.

[12] Charu C. Aggarwal. *Recommender Systems*. Springer, 2019.

[13] Xiangrui Meng et al. “MLlib: Machine Learning in Apache Spark”. In: *CoRR* abs/1505.06807 (2015). arXiv: 1505.06807. URL: http://arxiv.org/abs/1505.06807.

[14] Jennifer Golbeck. “Chapter 13 - Social Information Filtering”. In: *Analyzing the Social Web*. Ed. by Jennifer Golbeck. Boston: Morgan Kaufmann, 2013, pp. 191–201. ISBN: 978-0-12-405531-5. DOI: https://doi.org/10.1016/B978-0-12-405531-5.00013-4. URL: https://www.sciencedirect.com/science/article/pii/B9780124055315000134.

Summary

Chapter 1 - Introducion

In the modern era of e-commerce, the fashion industry has been profoundly impacted by the shift to online sales platforms. This transformation has revolutionized the buying process, offering customers an enjoyable and seamless experience through emerging technologies like chatbots. E-commerce has streamlined business transactions between companies and customers on the internet, reducing costs and improving efficiency. A key concept, the Customer Journey, plays a pivotal role in contemporary marketing. It involves the path consumers take from recognizing a need to making a purchase, emphasizing the dynamic relationship between customers and brands. Understanding this journey's stages is vital for businesses, enabling strategic planning to optimize customer satisfaction. The conventional linear customer journey has evolved into a complex, multi-channel process, with touchpoints and tools constantly overlapping. The COVID-19 pandemic accelerated the transition to online shopping, making it crucial for brands to understand consumer decision-making in a landscape filled with choices and information. The fashion industry plays a significant role, with e-commerce expected to reach over €820 billion by 2023. However, consumers face choice overload, emphasizing the need for businesses to balance cognitive effort and reward in decision-making. Recommender systems have become essential in online marketing and e-commerce, helping consumers discover new products, increasing engagement, and enhancing conversion and retention. These systems face the challenge of catering to highly subjective style preferences in the fashion industry.
The Alternating Least Squares (ALS) algorithm presents a promising solution, capable of addressing recommendation challenges in this complex landscape. ALS offers a practical yet rigorous approach, aligning with the data-driven, customer-centric market trends.

This thesis aims to create a tailored recommendation system for online fashion sites using the Alternating Least Squares (ALS) algorithm. It focuses on adapting ALS to fashion industry nuances, generating personalized product suggestions from browsing and purchase data. The evaluation considers accuracy, coverage, and diversity, leading to practical recommendations for system improvement. The research bridges Marketing Analytics and online fashion, addressing the need for personalized e-commerce experiences. It blends machine learning, marketing, and data analysis to overcome the challenge of understanding subjective fashion preferences. The outcomes hold practical value for businesses, enhancing marketing strategies and customer satisfaction through personalized recommendations.

Chapter 2 - Literature Review

Recommendation System

Recommender systems are advanced decision-making tools widely used in e-commerce and various domains. Their primary role is simplifying content discovery for users in an era of information overload. By analyzing user data and employing complex algorithms, these systems generate personalized recommendations, enhancing user experiences and benefiting businesses. However, implementing a recommender system isn't one-size-fits-all. Considerations include the domain (content type and error sensitivity), purpose (user or provider-focused), and context (device, location, time, and user state). In essence, recommender systems are indispensable in modern digital landscapes, blending AI with user behavior analysis to guide users to relevant content. Their design must align with specific industry needs, user preferences, and contextual factors to optimize user engagement, conversions, and retention.

Recommendation systems offer varying degrees of personalization, categorized into three main levels: non-personalized, semi-personalized, and highly personalized recommendations. Non-personalized recommendations, such as lists of best-selling items, are uniform for all users. Semi-personalized recommendations consider user segments, tailoring suggestions to distinct user clusters. Highly personalized recommendations, on the other hand, rely on users' interaction data to provide individually tailored suggestions. In certain domains, like wine, expert recommendations hold significant value due to the expertise required for accurate product knowledge. For example, wine websites often feature advice and opinions from sommeliers and experts. It's worth noting that many recommendation systems don't limit themselves to just one level of personalization. Instead, they often blend non-personalized and semi-personalized tiers to generate recommendations that balance general appeal with individual relevance. The extent of personalization offered by a website often corresponds to the volume of user traffic it attracts. Larger user bases empower algorithms to process more extensive datasets, resulting in more nuanced and highly personalized recommendations.

Recommender systems rely on structured user profiles to provide accurate recommendations. These profiles gather essential user information, including characteristics, habits, and behavior, critical for representing a user's current interests. User interaction data with the system falls into different categories: Explicit Feedback (users provide direct ratings or reviews through the system interface); Implicit Feedback (system lacks of explicit user experience data); Hybrid Feedback (combine both explicit and implicit feedback)
Explicit feedback, like star ratings and written reviews, offers a clear measure of user preferences, leading to targeted recommendations. However, it may be sparse. Implicit feedback, based on user behavior like clicks and views, is more abundant but less precise in expressing

preferences. Implicit feedback primarily identifies positive preferences, while explicit feedback provides a more comprehensive view, including dislikes. This makes explicit feedback suitable for addressing positive/negative challenges in recommendations. The way feedback is measured also differs. Implicit feedback focuses on relative relationships between objects, while explicit feedback uses absolute values like numerical ratings or written reviews. User profiles accumulate personal information like biographical data or browsing history, while item profiles include characteristics updated implicitly or explicitly through user interactions. However, some items may lack ratings, requiring recommender systems to employ filtering methods for proper evaluation and recommendations. In essence, recommender systems rely on various forms of user feedback, whether explicit or implicit, to craft personalized recommendations, each with its unique strengths and limitations.

Collaborative recommendation systems utilize similar users' preferences to suggest items of interest to the current user. They analyze preferences and ratings from users with comparable tastes, who have previously interacted with unknown items. The recommendations rely on user ratings' similarity, turning users with akin opinions into collaborators. These systems typically operate on vast online datasets and employ various techniques, including user-based, item-based, and model-based collaborative filtering.
User-based Collaborative Filtering focuses on user preferences and behavior. It calculates recommendations based on the weighted sum of similar users' average ratings for a particular item. Users with akin tastes are identified, and recommendations stem from their preferences. It assumes that users with similar tastes will favor similar items.
Item-based Collaborative Filtering: calculates item similarity. It recommends items similar to those the user has previously liked. It exploits relationships between items to suggest related products.
Model-based Collaborative Filtering utilizes mathematical models or machine learning algorithms to create user and item preference representations. These models capture intricate relationships and make predictions for new users and items based on the training data. Model-based filtering is especially effective in handling missing data and noise.
Each of these collaborative filtering approaches has its strengths and limitations, making the choice contingent on data nature and the recommendation system's specific objectives. User-based focuses on user preferences, item-based on the items themselves, while model-based leverages mathematical models for recommendations.

Collaborative Filtering is a recommendation technique effective in scenarios where items have limited associated content or when analyzing content is challenging, such as user opinions. It can offer unexpected recommendations based on user preferences, even without explicit content

in their profiles. However, it faces several challenges: Cold Start: This challenge arises when new users or items enter the system with insufficient historical preference data. It's problematic to provide accurate recommendations for users or items with few or no ratings, leading to potentially irrelevant or no recommendations.
Data Scarcity: Many possible user-item interactions remain unrecorded, resulting in a sparse user-item matrix. It leads to coverage issues and potentially inaccurate recommendations.
Scalability: As the dataset grows, computing similarities between numerous users or items demands significant computational resources, slowing down the recommendation process.
Preference Conformity: Users may be influenced by the ratings of others, potentially leading to a bandwagon effect where users follow popular preferences rather than expressing their own choices.
Synonymy: In systems with a vast product catalog, similar items may have names that the system cannot distinguish. Techniques like automatic term expansion, thesaurus construction, and Singular Value Decomposition (SVD) can address synonymy challenges.
Despite these challenges, Collaborative Filtering remains a valuable recommendation approach, particularly in scenarios where content analysis is difficult, offering serendipitous recommendations based on user preferences.

Content-based filtering is a recommendation system that relies on user-provided data to make personalized suggestions. It uses attributes of products or items to match them with a user's profile, which is created based on the user's past preferences. The more data users contribute, the better the system becomes at making accurate recommendations. Attributes can be gathered manually or automatically, with automatic methods extracting attributes from various items. A learning algorithm is then used to train the system, learning user preferences and generating recommendations based on user profiles. This method suggests items that share features similar to what the user has liked before. One example is LIBRA, which recommends books based on descriptions from Amazon.com webpages. Content-based systems employ various techniques to match item features with user profiles, determining if an item aligns with the user's interests. User profiles can be updated implicitly or explicitly, and items are categorized as relevant or non-relevant using methods like keyword matching and cosine similarity. Overall, content-based filtering offers personalized recommendations based on user preferences and item attributes, with user data playing a key role in improving recommendation accuracy.

Content-based filtering is an effective method for addressing the cold start problem, making personalized recommendations even for new users or items by relying on content attributes. It offers diverse recommendations that cater to individual tastes. However, it has limitations in capturing complex user behaviors and nuanced preferences. It depends on content quality

and may create a "filter bubble," limiting exposure to new interests. It may struggle with dynamic or context-influenced user preferences. Content-based filtering excels in personalized suggestions based on item characteristics but may not fully address user novelty and intricate, evolving preferences.

Hybrid recommendation systems are designed to enhance the accuracy and effectiveness of recommendation systems by combining various recommendation approaches. This fusion of algorithms aims to leverage the strengths of one method to compensate for the weaknesses of another, resulting in a more robust recommendation system. These hybrid approaches can be executed in different ways, including combining results from multiple algorithms, seamlessly transitioning between recommendation methods, iteratively refining recommendations, presenting multiple recommendations for each item, and incorporating features and supplementary information from various techniques.
One notable approach is the weighted hybrid method, which blends results from different recommenders using a linear formula with adjustable weights. This adaptability optimizes the utilization of each technique's unique capabilities, producing more accurate and personalized recommendations.
The switching hybrid approach is sensitive to the strengths and weaknesses of individual recommenders, allowing for a smooth transition between methods based on their competency in producing reliable ratings.
Cascade hybridization refines recommendations iteratively, progressively improving precision and relevance, particularly when initial recommendations are imprecise or incomplete.
Mixed hybrid simultaneously combines different recommendation techniques for each item, offering users a more comprehensive and adaptable experience, especially in contexts with diverse user preferences or multifaceted item characteristics.
Feature-combination and feature-augmentation hybrids enrich recommendations by integrating features from different techniques, potentially leading to more robust and nuanced suggestions.
Meta-level models generate recommendations based on a holistic understanding of user preferences learned by the initial technique, addressing the sparsity problem often encountered in collaborative filtering.
Hybrid recommendation systems offer a powerful strategy to optimize recommendation processes, providing users with more accurate, personalized, and context-aware recommendations by strategically combining multiple recommendation techniques.

Recommender systems aim to achieve two main objectives: enhancing user satisfaction and maximizing profits. The primary goal is to provide personalized recommendations that make users happy, ultimately driving engagement and potential revenue. On the other hand, some

systems prioritize profit maximization, viewing customer satisfaction as a means to boost revenue. Measuring the effectiveness of a recommendation system involves assessing its ability to predict user preferences accurately. This often involves comparing the system's predictions to users' actual ratings. Decision-support metrics further categorize predictions based on user reactions, such as interest levels. One challenge is avoiding the "filter bubble" phenomenon, where popular items dominate recommendations, limiting user exploration of diverse content. Diversity and coverage metrics help combat this issue by ensuring recommendations span a wide range of items and reach all users. Serendipity, the pleasant surprise of unexpected but appealing recommendations, is another important aspect. Balancing personalized recommendations with the introduction of diverse content is key to achieving serendipity and enhancing the user experience. Assessment metrics, focusing on accuracy and coverage, vary depending on the recommendation technique used. Accuracy metrics evaluate the system's ability to predict user ratings accurately, often by withholding high-rated items for testing. These metrics help quantify the system's capacity to deliver user-aligned recommendations, thus improving user satisfaction.

The evaluation of recommendation systems relies on various metrics, grouped into statistical accuracy metrics and decision support accuracy metrics.

Statistical Accuracy Metrics focus on the system's ability to predict ratings accurately. Two common metrics are:

- Mean Average Error (MAE): It measures the average magnitude of errors between predicted and actual ratings, with lower values indicating better accuracy.
- Root Mean Squared Error (RMSE): An extension of MAE, RMSE is widely used. It involves the square root of the average squared differences between predicted and actual values.

The choice between MAE and RMSE depends on the system's goals; RMSE penalizes large errors more, while MAE balances user satisfaction.

Decision Support Accuracy Metrics aid users in determining the quality of recommendations individually. They involve analyzing each recommended item's relevance to the user, leading to four possible outcomes: True Positive, False Positive, False Negative, and True Negative. These outcomes are used to calculate Precision and Recall metrics, which assess the correctness and relevance of recommendations. For Top-N recommendations, two crucial metrics are:

- Precision at K: Ensures at least one relevant item is present in the recommendations.
- Recall at K: Measures the fraction of relevant items among the top K recommendations.

Average Precision (AP) scores recommendations based on precision at each position in the list, and Mean Average Precision at K (MAP@K) assesses the quality of ranked recommendations

comprehensively.

Mean Average Precision at K (MAP@k) is a widely used evaluation metric for recommendation systems. It assesses the quality of ranked recommendations by considering both their relevance and their position in the recommendation list. MAP@K is a comprehensive metric for evaluating recommendation quality that takes into account both relevance and ranking position. It provides valuable insights into how well a recommendation system performs in delivering personalized and relevant content to users. MAP@K considers the entire ranked list of recommendations, accounting for the possibility of multiple relevant items. It rewards diverse and personalized recommendations. Higher MAP@K scores signify better recommendations, making it a valuable tool for fine-tuning and comparing recommendation algorithms. Discounted Cumulative Gain (DCG) measures ranking quality by considering penalties based on the position of relevant items.

Normalized Discounted Cumulative Gain (nDCG) is a widely used metric in information retrieval and recommendation systems. It penalizes relevant items positioned lower in the ranking. Receiver Operating Characteristic (ROC) Curve visually depicts a recommendation system's ability to distinguish between relevant and irrelevant recommendations. It helps optimize recommendation algorithms by assessing their discriminative power and enhancing precision.

Recommendation systems are now integral to the success of many companies and online platforms, such as Amazon, Netflix, and Spotify. These systems, powered by advanced algorithms and user data analysis, significantly enhance user experiences, increase customer satisfaction, and boost revenue. Amazon's item-to-item collaborative filtering system, which examines user preferences and browsing patterns, contributes to an impressive 35% of its sales. Netflix relies on recommendation algorithms for a substantial 75% of its viewership, using a hybrid recommender system that evaluates user behavior and suggests similar movies. Spotify, the audio streaming platform, creates tailored playlists like "Discover Weekly" based on users' music preferences, utilizing three recommendation models: Collaborative Filtering, Natural Language Processing, and Audio File Analysis, thanks to its acquisition of Echo Nest. Other recommendation systems like Ringo specialize in music album and artist recommendations, while GroupLens focuses on Usenet news recommendations using clustering and implicit ratings. Hybrid systems that combine multiple recommendation techniques, such as NewsDude and Pipper, are prevalent. Overall, recommendation systems are vital tools for enhancing user experiences, influencing consumer choices, and driving business growth in the modern world.

Alternating Least Square Matrix

In the era of AI and Big Data, data-driven companies have gained immense influence. Recommendation systems, powered by AI and machine learning, have become crucial tools for digital

enterprises. These systems address the recommendation problem, which can involve explicit feedback where users rate items or implicit feedback where user preferences are inferred from indirect data like clicks or views.

To handle implicit feedback efficiently, Alternating Least Squares (ALS) is a widely adopted technique in collaborative filtering recommendation systems. ALS factorizes a user-item interaction matrix into lower-dimensional matrices representing user and item features. It iteratively optimizes these matrices, first fixing user features and optimizing item features, and then vice versa, until convergence. The resulting matrices are used to make recommendations based on predicted user-item interactions. Matrix factorization, while powerful, involves a non-convex optimization problem due to multiple local minima in the objective function. Traditional convex optimization methods are not directly applicable. Despite this complexity, optimization is essential as it guides the model to learn hidden user-item relationships over successive iterations, making it a fundamental element in the success of recommendation systems based on matrix factorization.

Alternating Least Squares (ALS) is a powerful algorithm used in large-scale recommendation systems. It efficiently handles extensive data by breaking down the user-item interaction matrix into smaller matrices, making it computationally manageable while capturing essential user-item patterns. This versatility finds applications in various domains, such as e-commerce and content streaming.

LinkedIn employs ALS in its "People You May Know" feature, enhancing user connections by suggesting relevant profiles based on user behavior, connections, and shared interests. For instance, it identifies users with similar professional backgrounds or mutual connections, making networking easier.

Pandora, a music streaming service, utilizes ALS to create personalized radio stations for users by analyzing their listening history. It predicts user preferences based on past interactions, suggesting music genres, artists, and songs aligned with their tastes and even introducing them to new music.

In conclusion, ALS is a formidable tool in recommender systems, revolutionizing personalized recommendations across various industries. Despite its computational challenges, ALS continues to adapt to big data's evolving landscape, delivering tailored suggestions and enhancing the user experience in data-driven markets.

Chapter 3 - Methodologies

This study utilized data from H&M Group through a Kaggle competition to create a recommendation system based on past transaction data. H&M provided datasets containing information about products, clients, and transactions.
The project's objective was to develop a recommendation system capable of predicting 12 items in order of relevance to each user. Notably, the dataset lacked explicit user evaluations, requiring the system to rely on "implicit feedback." This feedback could originate from various sources, including past transactions, product views, browsing behavior, or any actions indicating user interest in specific items. To achieve this goal, the study carefully examined the dataset and outlined a methodology for building an accurate and relevant recommendation system based on implicit data.

Explorative Data Analysis

During the dataset analysis phase, several key characteristics of customer behavior in their decision-making process were observed. The user base consisted of 1,371,980 customer IDs, with an age distribution revealing two distinct demographic groups. The larger cluster, aged between 25 and 30, indicated a significant presence of young or working-age individuals. The second peak, around age 50, suggested a concentration of middle-aged or older individuals. To gain a deeper understanding of these different user groups, they were divided into three categories based on age, representing macro-level differences in decision-making processes and spending power. This categorization revealed that 47.1% of customers were under 30, 47% fell within the 30-60 age range, and 5.8% were over 60.
The dataset also provided insights into other important customer characteristics for both the company and the study. Two noteworthy details pertained to customers' active engagement with the company. Firstly, their status as members of the company's club was examined, revealing that 97% of users were active members, defined as those making at least one purchase per year. While this information may not provide deep analytical insights, it helped contextualize the dataset.
Another significant attribute was the frequency of newsletters received by customers. Approximately 65% of customers did not receive newsletters, while the remaining portion received regular communications. These details indicated the level of interaction between users and the company's systems, shedding light on customer engagement with corporate communications.

The dataset comprises 105,542 items, which can be categorized and characterized in various ways. The available data offer detailed insights into product distinctions. Among the macro-categories, Ladieswear emerges as the most extensive category, followed by Children's items.

Notably, there's a significant contrast in the number of items available for men and women. This indicates a strong focus on women's fashion by the company, underscored by the ample availability of children's items, often purchased by mothers or relatives.
In the context of recommendation systems for fashion websites, conducting exploratory data analysis necessitates a comprehensive understanding of product attributes, specifically "color" and "prints." These attributes are pivotal as they directly shape consumers' fashion preferences and choices. Color, for instance, significantly influences the appeal of a garment or accessory to an individual. Color choices often reflect personality, fashion trends, social context, and seasonal preferences. The recommendation system's ability to comprehend color preferences is vital, enabling it to suggest products aligned with each user's color inclinations.
Prints represent another critical facet of the fashion world. Various prints, such as florals, stripes, animal patterns, and more, convey distinct styles and tastes. Users might have a penchant for floral and romantic patterns, while others lean towards geometric and minimalist designs. The system's capacity to discern users' print preferences empowers it to recommend products that align with their personal style. Moreover, when attributes like color and prints are combined, the system can generate highly customized recommendations. For instance, if a user desires a blue dress with a floral print, the system should identify products matching this precise description.

The final dataset exclusively comprises customer transactions, initially lacking the potential for meaningful analysis. Consequently, the decision was made to merge all available datasets, providing a comprehensive perspective of the data. This transaction dataset includes product prices, sales channels, and dates, spanning from 20 September 2018, to 20 September 2020, encompassing a total of 31,788,324 transactions over two consecutive years.
Examining the sales trend reveals significant fluctuations over time, with the most prominent peaks occurring during the summer months, notably in June. The data is notably impacted by the COVID-19 pandemic, which emerged in February/March 2020 and had global ramifications, drastically reducing or even nullifying sales for many companies. Notably, between March and May, purchases in physical stores nearly ceased, while online transactions experienced a notable increase.
This experience demonstrated that e-commerce has become an indispensable component of modern fashion companies' success. Comparing the two sales channels, it's evident that the average spending per user in e-commerce is double that of retail. Several factors contribute to this disparity, shedding light on consumer behavior in these distinct sales channels. First, the convenience and accessibility of e-commerce can positively influence customer spending. Furthermore, the presence of promotions, discounts, and special offers on e-commerce platforms can encourage increased spending. E-commerce companies often employ dynamic pricing

strategies and targeted promotions to attract and retain customers, leading to more consistent transactions.

Additionally, the online shopping experience can be more engaging for some customers. Features such as product reviews, personalized product suggestions, and efficient checkout processes contribute to easier and more enjoyable shopping, potentially facilitating more spending. Lastly, e-commerce offers an environment conducive to monitoring purchase history and customer preferences, enabling the generation of personalized recommendations. Such personalization can have a positive impact on online customer spending, as it directs customers to products they are likely to be interested in.

ALS Implementation

The recommendation system was developed using an implicit model based on Alternating Least Squares (ALS). This choice was driven by the nature of the available data, which primarily consists of transaction-based information rather than explicit ratings. ALS is well-suited for implicit recommendation systems as it allows modeling implicit interactions between users and items. The ALS method decomposes the user-item interaction matrix into smaller submatrices, representing user and item embeddings in the same space. This enables efficient recommendation calculations based on similarity between users and items within implicit interactions. ALS efficiently handles large datasets, making it suitable for providing real-time personalized recommendations in a business context.

As model preparation, User and article IDs are converted to numeric IDs, creating a numeric representation. A sparse user-item interaction matrix is constructed, where a value of 1 indicates user-item interactions, and this matrix is stored in COO format, an efficient representation for sparse data.

In order to train the model, specific parameters for the ALS model are set, such as the number of latent factors (set to 10) and the number of iterations (set to 2). The ALS model is trained using the COO matrix, learning latent factors for users and items to model interactions. For the model validation, the dataset is split into training and validation sets. Validation involves training ALS models with different hyperparameters (factors, iterations, and regularization) and using Mean Average Precision at K (MAP@K) as the evaluation metric (K is set to 12). A grid search helps find the optimal hyperparameter combination for the highest MAP@12 score. Now, the COO matrix is converted into a Compressed Sparse Row (CSR) matrix format, which is efficient for storing and manipulating sparse data, optimizing matrix-vector multiplications. The trained model is ready to provide 12 personalized recommendations for each user, maximizing the MAP@K score.

Chapter 4 - Results

Evaluation is a crucial aspect of recommendation systems as it assesses their ability to provide accurate and user-aligned suggestions. It involves scrutinizing the algorithm's decision-making accuracy and measuring the extent of errors in its recommendations. This evaluation process is essential for improving the system's functionality continually. To evaluate a recommendation algorithm, a test environment is created, withholding actual user purchases and comparing the algorithm's recommendations to previously unknown purchases. This comparison offers insights into accuracy, potential weaknesses, and adaptability to evolving user preferences. Various evaluation metrics are employed to gauge the algorithm's effectiveness in generating personalized recommendations. These metrics act as benchmarks, demonstrating how well the system identifies and suggests relevant content to users. They are a litmus test for meeting user needs and enhancing their experience. Selecting an appropriate test set is critical for a valid evaluation. The criteria for inclusion focus on customer activity, retaining engaged users with a history of at least three distinct purchases. The top 400 frequently purchased items are chosen to ensure diversity while avoiding data overload. This selection also evaluates the system's ability to recommend popular products, which is commercially significant. To maintain the test set's temporal integrity, only the most recent transaction for each user is retained. This aligns with the idea that user preferences evolve over time, making the latest interaction a relevant reference point for evaluating accuracy and adaptability. The test set is carefully constructed to ensure a comprehensive yet manageable evaluation dataset, closely mirroring real-world usage patterns.

The Mean Average Precision at K (MAP@K) metric is vital in assessing recommender systems. It evaluates recommendation quality by considering both accuracy and item positioning in results. This metric measures the precision of the first K recommendations, which in this study is set to 12. A higher MAP@K indicates better recommendations because it rewards not only correctness but also favorable item placement. The result of this study is a value of MAP@K equals to 0.1075 which implies that the algorithm places relevant items among the first 12 results about 10.75% of the time. This suggests good performance but should be context-specific; what's considered excellent can vary depending on the application and data type.

The Normalized Discounted Cumulative Gain (NDCG) is a crucial metric for evaluating recommendation systems, especially when recommendations are ranked. NDCG assesses both item relevance and their ranking, emphasizing that higher-ranked items are more engaging. It combines Discounted Cumulative Gain (DCG) and Ideal DCG (IDCG), with a higher NDCG indicating superior ranking quality. In this study, achieving an NDCG value of 0.13 signifies the system's ability to prioritize relevant items effectively and present them in an engaging order. A higher NDCG score reflects the system's success in enhancing user experiences by providing not just relevant but strategically ranked recommendations.

The Receiver Operating Characteristic (ROC) curve, from which the Area Under the Curve (AUC) is derived, measures the ability of the recommendation system to distinguish between relevant and non-relevant items as the classification threshold changes. A higher AUC score, such as the observed 0.6046, indicates a good ability to discriminate between user-preferred items and others. This means the system excels at ranking relevant items higher, improving recommendation precision. Importantly, an AUC exceeding 0.5 demonstrates the model's superiority over random guessing. AUC is also robust to class imbalance, common in recommendation scenarios. In summary, an AUC of 0.6046 reflects the system's strong ability to distinguish relevant items, contributing to user satisfaction and system success.

The effectiveness of a recommendation system in business management is measured by its impact on sales. The key question is how many customers, after receiving tailored recommendations, actually make purchases. A test set containing the latest transactions is crucial for this analysis, acting as a benchmark to evaluate recommendation accuracy and real-world impact. In this evaluation, 70% of predicted items aligned with actual purchases, showcasing the system's precision. The true positive rate (TPR) stands at 0.6246, indicating that about 62.46% of relevant items are correctly suggested. This metric is vital as it enhances user satisfaction and engagement by minimizing missed opportunities for relevant recommendations. In conclusion, this recommendation system excels in providing personalized and effective recommendations, enhancing user experiences, and potentially driving business outcomes. These positive results lay the foundation for future improvements.

Chapter 5 - Managerial Implications and Conclusions

The research conducted provides valuable insights into the managerial and practical implications of an effective and personalized recommendation system in the realm of business and marketing. Key takeaways from the study highlight the substantial impact of such a system on sales performance and the versatile applications that can enhance overall business operations.
Firstly, the study underscores the system's effectiveness in delivering personalized recommendations, leading to an enhanced customer experience, heightened satisfaction, and increased customer loyalty. Business managers can leverage this insight to design strategies like tailored loyalty programs or targeted communications to further enhance the customer journey.
Secondly, the research reveals that recommendations efficiently influence user purchase decisions. This implies that marketing resources can be allocated more efficiently, focusing efforts on products or services with a higher likelihood of user adoption. Managers can utilize this information to optimize marketing budgets and maximize ROI.
Moreover, the study highlights the system's adaptability to changing market trends and user behavior. This flexibility is vital for staying competitive in a rapidly evolving business landscape. Managers can harness this adaptability to remain industry leaders and swiftly respond to customer needs. Furthermore, the positive outcomes of the research indicate the potential for new business opportunities based on recommendations. Managers can explore avenues like offering consulting services based on user preferences or forming strategic partnerships with recommended product manufacturers. These new opportunities can diversify revenue streams and boost profitability. The core objective of a recommendation system extends beyond merely suggesting items aligned with a customer's historical preferences. It delves into influencing customer choices and behaviors to improve not only user satisfaction but also company growth and competitiveness. These systems leverage data-driven insights to create a personalized shopping experience that persuasively guides customers toward exploring new products or services. By intelligently recommending items that resonate with user interests, these systems can enhance the average transaction value, cross-selling, upselling, and alignment with market trends.
In conclusion, the research findings underscore the intrinsic value of an effective and personalized recommendation system in the domains of marketing and e-commerce. The managerial and practical implications arising from these results provide a robust foundation for optimizing business strategies, enhancing customer experiences, resource optimization, and exploring new avenues for growth. These systems not only offer suggestions but also actively shape customer choices, making them pivotal instruments in the modern business arsenal, transcending from mere assistants to influential partners in customer decision-making.