NOVA IMS Information Management School

MDSAA

Master Degree Program in

Data Science and Advanced Analytics

Applying Recommended systems to books regarding user's similarities and reader's ratings

Elena Nozal Moralejo

Dissertation

presented as partial requirement for obtaining the Master Degree Program in Data Science and Advanced Analytics

NOVA Information Management School

Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisbo

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

APPLYING RECOMMENDED SYSTEMS TO BOOKS REGARDING USER'S SIMILARITIES AND READER'S RATINGS

by

Elena Nozal Moralejo

Dissertation presented as partial requirement for obtaining the Master's degree in Advanced Analytics, with a Specialization in Data Science.

Supervisor: Prof. Roberto Henriques

January 2023

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration. I further declare that I have fully acknowledge the Rules of Conduct and Code of Honor from the NOVA Information Management School.

Elena Nozal Moralejo

Madrid, 22 of November 2023

ABSTRACT

Books are the key for knowledge, for entertainment, for widen the imagination. Are essential for individual growth and development, being described as the ladder of human progress. However, with the growth of the publishing business, the number of books offered to consumers are unmanageable. Becoming a problem for readers, which face the problem of how to choose or even discover a book they really will enjoy. The present work intends to detail the different recommendations techniques available today to identify the most accurate recommender system based on user similarities and previous ratings. Offering readers an efficient and quick way of discovering new books without being buried in options. The methodology adopted was the implementation and analysis of the most common and widely used recommender systems to conclude, based on the evaluation of the techniques, which is the most appropriate for this specific problem. Regarding Collaborative Filtering, the results stated that the best model was the Singular Value Decomposition with a Root Mean Square Error of 0.8. While, in Content-Based, the results stated that TF-IDF technique was better for extracting keywords and k-means was the ideal clustering algorithm for this specific problem. In conclusion, this Masters Project presents and compares the different algorithms applied to recommendation systems, finds the most suitable approach for the given problem, and offers a better understanding of the recommendations systems available nowadays.

KEYWORDS

Recommended Systems; Big Data; Machine Learning; Natural Language Processing; Data Analysis

Sustainable Development Goals (SGD):

INDEX

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

ML	Machine Learning
AI	Artificial Intelligence
NN	Neural Networks
CF	Collaborative Filtering
CB	Content-Based
RS	Recommendation Systems
SVM	Support Vector Machine
MAE	Mean Absolute Error
RMSE	Root Mean Square Error
MSD	Mean Squared Difference
ROC	Receiver Operating Characteristics
PRC	Precision Recall Curve
NLP	Natural Language Processing
IoT	Internet of Things
DSR	Design Science Research
SVD	Singular Value Decomposition
ALS	Alternating Least Squares
NMF	Non-Negative Matrix Factorization
SGD	Stochastic Gradient Descent

1. INTRODUCTION

For starters, the book industry is growing exponentially, becoming a problem not only for the sustainability of the business, but also for the consumers. The number of new books published are so overwhelming that it is becoming obvious that no reader is capable of buying, neither reading nor learning about the next novelty. This is due to the alarming publishing rate that bombards consumers with too many options each day.

This conclusion can be drawn by learning the following facts: only in Spain there are more than 750 publishing houses, that publishes more than 90 thousand books per year (which nearly 15 thousand are novelties), being the second country with the highest number of titles published behind Italy (Cedro, 2021). This means that, on average, 41 novelties are published per day only in Spain. Thats nearly 2 per hour. However, this phenomenon not only applies to Spain or Italy, with 95 thousand books per year, but it extends to the whole world, being 2.2 million books published only in 2022 (Orús, 2023).

However, most importantly is the average number of copies a book sell. Having Spain as an example, studies carried out by bookshops in 2022, state that 86% of the titles sell less than 50 copies per year. Not only that, but also, only a 0.1% sell more than 3000 copies (Efe, 2022).

These numbers state that it is impossible for consumers to follow the growth of the market as the sales are not aligned with the number of books published. This becomes more obvious when a book becomes, in average, obsolete in 6 months after being published. So, it is crystal clear that readers are bombarded with new titles each day. Meaning that the newest title overshadows the previous one and so on.

It has become overwhelming, not only for readers to decide what book they really want, but also for bookstores which 80% of sales are made with less than 20% of the titles.

These numbers are unmanageable for consumers. They cannot read all the novelties published, neither know every single title published. Therefore, many books are doomed to fail due to the overwhelming number of titles available. There are so many, that most of them won't be notice by consumers and, at the end of the day, a book that is not known will never be read neither recommended.

Furthermore, readers and writers have been talking about the phenomenon called Fear of Missing Out, FOMO, where a person feels anxiety due to the fear of missing out a social event or fashion trend. This applies to bestsellers which rapidly spread throughout the internet as the next big story as Harry Potter or Game of Thrones, but also its fame disappears as quickly as it came. Giving readers very little space to read and enjoy and be part of the phenomenon (TresB, 2023).

FOMO has become the epidemic of the XXI century. A new type of anxiety produced by the urge to be part of the group, to feel connected through the internet and participate actively in social media and its trends on real time. An epidemic that also affect readers and writers. Many suffer from anxiety as they know they cannot read every best seller on time to be part of the phenomena or the discussion. As for the writers, the fear of being left out has made them create books in less time, more stories to offer the readers so they do not become obsolete or forgotten. Which means even more titles are introduced into the market.

Therefore, this project looks to create a solution for readers that want to discover their next reading without being drowned by all the novelties the industry offers. One of the ways this can be done is by applying recommended systems. They have become well spread, being the most common those applied in videos, songs, and tv series and films. In the last years the interest in recommendation systems have increased significantly. They play a fundamental role in online platforms and companies such as Amazon, TikTok or YouTube. As they increase revenue and built connections to its costumers given them relevant and useful recommendations.

Moreover, applying a recommender system to the problematic described above will give customers a set of books than can be interesting to them, based on their past ratings and/or similarity to the books it has already read. It offers a new service that many users already enjoy in different platforms. Why not in books? We already know about Amazon which applies these algorithms in all its products, books included.

Therefore, the aim of this study is to analyse the different algorithms that are used in recommended system to find the best approach when applied to books. Which not only receives ratings from their buyers, but also can have similar characteristics between one and another. Personalized recommendations have positive impact in clients and revenues.

Consequently, this thesis project will provide an approach that offers readers and bookstores a new tool to learn about new books and, also, increase their competitiveness to the latter as it has been proven to be a good marketing tool.

1.1. OBJECTIVES OF THE STUDY

In this context, the problem described arises, discouraging readers from buying books from newcomers or less well-known publishing houses. Therefore, the following research question was formulated:

Can we apply an accurate and useful recommended system based on user's similarities and readers ratings to books? And *which approach is the most effective for this application?*

To answer the above Research Questions, it is defined the following Research Objectives for this project:

i. Study the different techniques used in recommended systems, such as Collaborative Filtering, Content-Based and NLP approaches;
ii. Develop a recommended system with the best approach in order to solve the problematic described above and apply it to a practical case;

The solution developed was done applying the algorithms and evaluation techniques described for recommended systems in the studies and literature relevant to the topic. Which included Collaborative Filtering, currently applied by Amazon and other, Content-based, an upgrowing technology with implicit data (description of items, types and more) rather than explicit data (ratings). Giving a greater perspective of how accurate the model is applied to this business and offers a greater insight of the different existing approaches.

These studies state that applying cosine similarity to implicit data can tend to recommend older books rather than newer ones, so in the study done in this project several similarity computations will be studied. Moreover, studies shown that hybrid approaches and use of enriched data gave better results than using only Collaborative Filtering or Content-based algorithms.

In any case, very few research used NLP to add implicit data to their models, even when using variables like the title of the books.

Therefore, the thesis will work in building a study trying to extend the previous research by applying the most common approaches, Collaborative Filtering, Content-based, SVM, and so on in one single study, while also using NLP techniques to improve results.

Regarding the methodology, the project followed the Cross Industry Standard Process for Data Mining, CRISP-DM, methodology. This method follows six phases, Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation and Deployment (Hotz, 2023).

Firstly, the Data Understanding. To explore and study the different algorithms applied to recommended systems it is needed a big quantity of data. Not only that, but it also needs to have both quantitative and qualitative data, to apply all the techniques. Therefore, the database used for this study is provided by the GoodReads website which collects data not only from millions of books, but also from the users which read and grade their last readings. From the different collections of datasets, it was selected those which contained the information for Young Adult books as they have a great range of ratings and users.

Secondly, the Data Preparation. Before any analysis or application of an algorithm, the data must be prepared. For this purpose, the data is selected, cleaned… It is checked for missing data, outliers, transformation of variables, and so on. To do so, the software used was Jupiter Notebook, and the selected language was Python. Note that for the qualitative data, content analysis was done.

Thirdly, modelling started by preparing data and then the different algorithms are selected to study the recommended systems. Furthermore, to apply them, the data is divided into training, validation, and test sets. Then the models are built and assessed.

Regarding the scope of this study is to identify the advantages and disadvantages of the algorithms implemented to find the most suitable for books recommendations which have unique characteristics and which likeness depend in tastes rather than quantitative reasons. It includes the implementation of the described algorithms and evaluation measures. And it excludes the creation of a specific algorithm for this application or a web page to offer users.

Furthermore, regarding its limitations, the amount of data that can be processed is limited due to the capacity and processor power of the computer used. Furthermore, there’s no cloud storage or data lake implementation, so there is no real time data to feed the model to create an App for this application. Also, note that the enrichment of data is complicated as most repositories about books are private, valuable information to this study was not available, such as the number of books sold for each title. Therefore, the study is subjected to the data recollected and the tools available.

Finally, the significance of this project is subjected to the information extracted from the comparison between the different techniques and algorithms applied nowadays into recommendations system.

Offering a more overall view and insights of the models: Content Based, Collaborative Filtering and NLP approaches. Having a better understanding of their individual advantages and disadvantages when applied to a different application. Which benefits researchers to have a study joining the different approaches having into consideration past ratings and books characteristics and language preprocessing.

In the second chapter, literature review, where it will be studied the actual state of the techniques that will be used in the thesis. In first place, an introduction to Machine Learning and Recommendations Systems, followed by its algorithms and evaluation methods. Later, it will be commented the different use cases and applications of the recommendation systems. Furthermore, it will be talked about the ethics and challenges of the techniques. Finally, the most relevant case studies will be summarized to further understand the state of the recommendations systems and its applications in the book industry.

In the third chapter, methodology, it will be stated the research design and methods carried out in the study. It will appear the different approaches, tools and data sources that were used to conduct the study and collect the data. Next, it described the preprocessing techniques carried out, followed by the modelling and the evaluation.

In the fourth chapter, results and discussion, we will carefully examine the results of the different approaches, focusing on those methods that gave better results. It will be explained in detail the different adjustments of the hyperparameters to improve the results. Followed by a comparison of the best approaches and its results.

In the fifth chapter, we present a summary of the study and the obtained results. It will be studied if the results were the adequate and if the objectives have been achieved.

In the sixth chapter, limitations and recommendations for future works, it is stated the limitations encountered in the study and, also, it is written the future works than can be done in order to expand the work done in this project.

In the seventh chapter, references, there are exposed all the information sources where the information written in the thesis have been extracted and that are indicated throughout the document.

2. LITERATURE REVIEW

Learning is the ability to acquire new, or change existing, skills, values, behaviours, knowledge, or preferences. There are different theories of how human learns, from Behaviourism to Experientialism. However, while humans naturally learn through experience, machines rely on data, (Deng et al., 2020).

Machine Learning (ML), is a subfield of Artificial Intelligence (AI) and computer science that can be defined as the ability of a machine to replicate how humans learn through the use of data that is processed by algorithms as input to predict output values (Tucci, 2023)

ML models fall into three main categories:

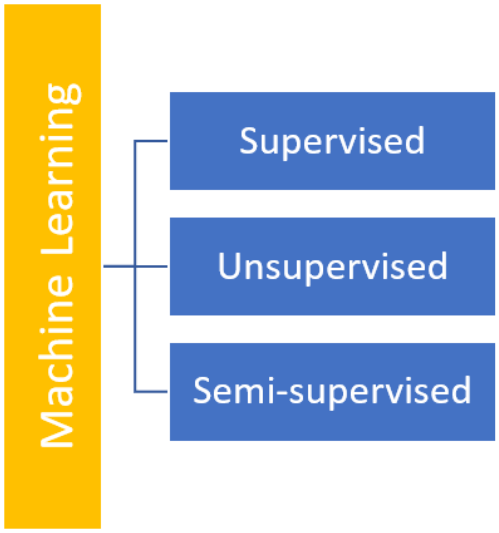

Figure 1. Classification of Machine Learning Models

Supervised learning uses labelled datasets to train algorithms to predict or classify data accurately. It is useful to solve real-world problems such as classify spam emails, detect fraud, predict scores, and more. Methods that are used in supervised ML include Neural Networks (NN), Naïve Bayes, Linear and Logistic Regression, Random Forest, and Support Vector Machine (SVM).

While unsupervised learning uses unlabelled datasets which it analyses and cluster with the use of algorithms. It can discover patterns in the data without human intervention. It is ideal for exploratory data analysis, customer segmentation, and pattern and image recognition. Algorithms used in unsupervised ML include NN, k-means and probabilistic clustering.

An in-between supervised and unsupervised learning is the semi-supervised learning which uses a smaller labelled data set while learning to help classification and feature extraction from a larger, unlabelled data set. Offering a way to tackle problems with not enough labelled data and apply a supervised learning algorithm.

Machine Learning is a key technology that embraces the intelligent power to harness knowledge from data. Becoming a dynamic area of research and opportunities to solve real-life problems (Alzubi et al., 2018). Furthermore, IBM, a global technological innovator company, one of the top 10 company's leaders in AI and ML in 2023 (Medina, 2023), says that one of the most common use cases is the

Recommendation Engines, which uses past consumption behavioural data to help discover trends that can later be used to develop more effective cross-selling strategies (IBM, 2023).

2.1. RECOMMENDED SYSTEMS

A recommendation system is a branch of ML whose main objective is to generate meaningful recommendations to users for products that might interest them. Another definition of recommended systems says that "any system that produces individualized recommendations as output or has the effect of guiding the user in a personalized way to interesting or useful objects in a large space of possible options" (Burke, 2002a). These recommendations can be based on users' interest and previous purchases or preferences (Jayaram et al., 2022). To make these recommendations, algorithms can apply models based on similarities, either by proximity of an item to another, or by similarity of a user with another based-on ratings.

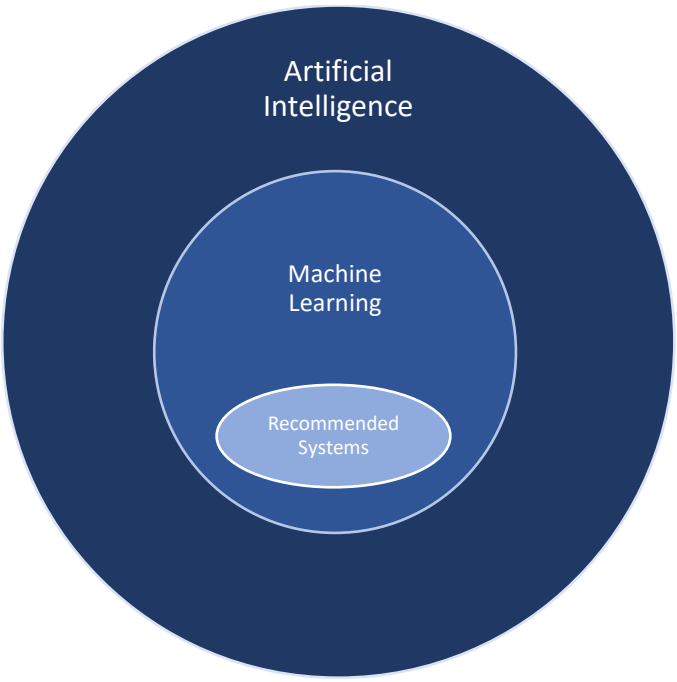

Figure 2. Recommended systems being part of Machine Learning.

Recommendation systems are a potential solution to the information overload problem. Offering users, a more simple and faster way to obtain useful information.

It has two basic principles:

- Generate *personalized* recommendations. Its goal is to create a meaningful experience for one user, not to generate a cluster of similarities.
- To help the user choose among few options.

These principles state the difference between recommended systems and search engines. When issued a query, a search engine will output the same set of results regardless of who made the request. To generate personalized outputs, recommended systems may maintain data of user's activity. This data can be long or short term (Burke et al., 2011).

One of the issues around recommended systems is that the application domain has a strong influence over the methods that can be successfully applied. This means that depending on the item, the user's taste can fluctuate through time with different rates.

In the case of music, taste can vary slowly, whether a user's interest in celebrities news can change drastically. This affects the reliability of preferences recorded in the past may vary depending on the application domain. For books, its consumption and recommendations are available for long period of times—often years. On the other hand, the technological domain can become rapidly obsolete, the same to domains subjected to timelines such as news, cultural events, and so on.

Therefore, recommended systems must understand the knowledge underneath them (Burke et al., 2011).

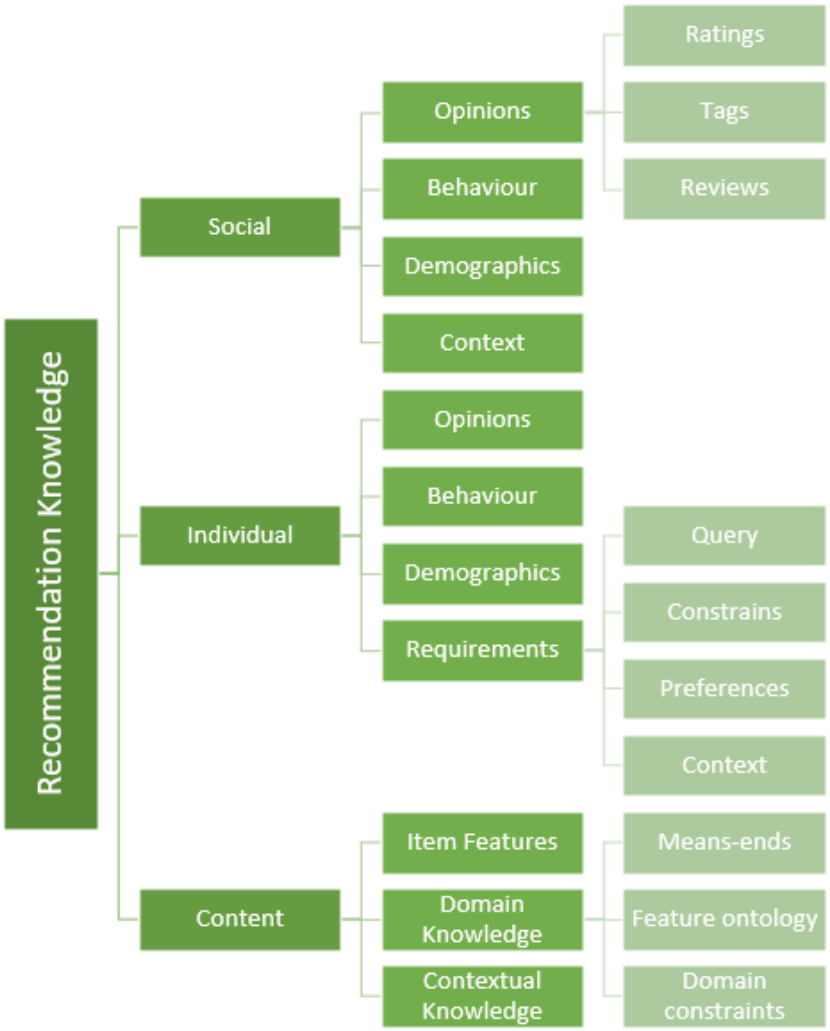

Figure 3. Taxonomy of Knowledge Sources in Recommendation (Felfernig & Burke, 2008).

Felfering and Burke, (Felfernig & Burke, 2008) defined three basic types of recommendation knowledge: (i) social knowledge of the user in general, (ii) individual knowledge about the user we want to make recommendations and (iii), content knowledge about the items to be recommended.

2.1.1. Algorithms

Recommendation algorithms are the core of the recommended systems. We can divide the recommendation technology into recommendations based on association rules, content-based, collaborative filtering, and hybrid recommendations.

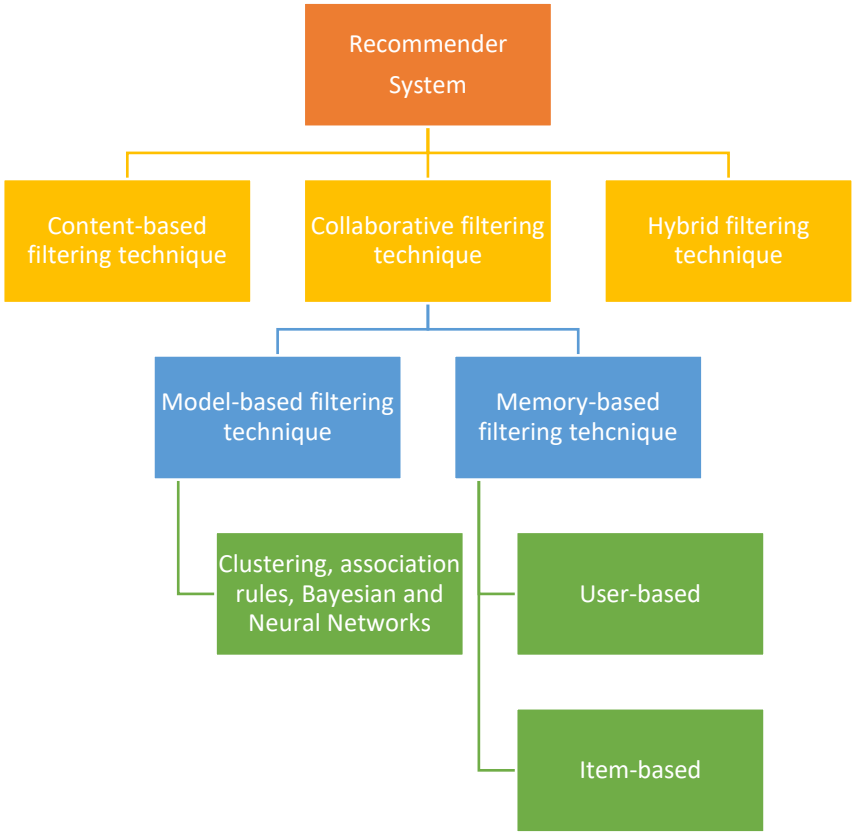

Figure 4. The main types of recommendation systems, (Zahrawi & Mohammad, 2021).

i. Recommendations based on association rules

Its goal is to obtain the internal connection between items from a large amount of data. In other words, to find strong rules to make recommendations based on them.

To generate association rules there are two steps:

- Search for the frequent item in the set —the item whose support is not less than the actual minimum value.
- Search for strong rules in the frequent item set —the associated item which support, and confidence degree are not less than the actual value.

The more common association rules algorithm is Apriori and FP-Growth algorithm (Xu et al., 2020).

ii. Content-based recommendations

To recommend similar items or products according to the items users liked previously.

The algorithms of content-based recommendations core are to extract the features of the items the system wants to recommend, and the user's interest features, while establishing the user's interest model.

In these algorithms is important to extract the most relevant features, the most commonly used method is the TF-IDF algorithm with word frequency statistics (Wu et al., 2008). Regarding the user's interest model, the algorithms used includes decision trees, NN and vector-based representation method.

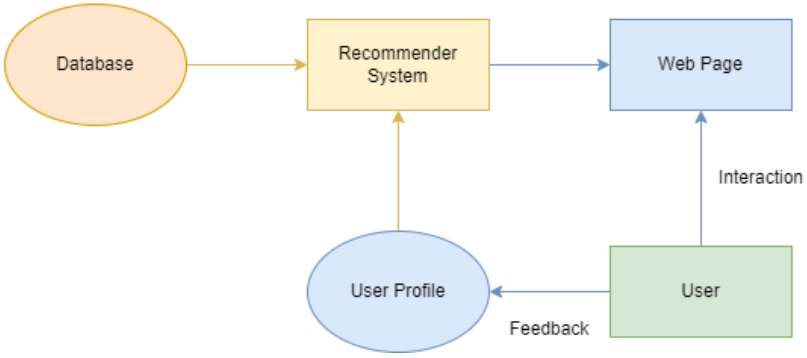

Figure 5. Diagram of Content-based recommendations (GeeksforGeeks, 2020)

iii. Collaborative filtering recommendation

To search for the interest relationship between the recommended items and the users through the previous behavior of the users, and then make recommendations based on it (Bobadilla et al., 2011).

This algorithm can be divided into:

- *User-based recommendation.* Is the one that looks for the level of preference for an item over the historical data of all users. The calculations search for groups of users with similar interests to the targeted user and makes recommendations to the target based on the group history information preference (Cheng et al., 2016).

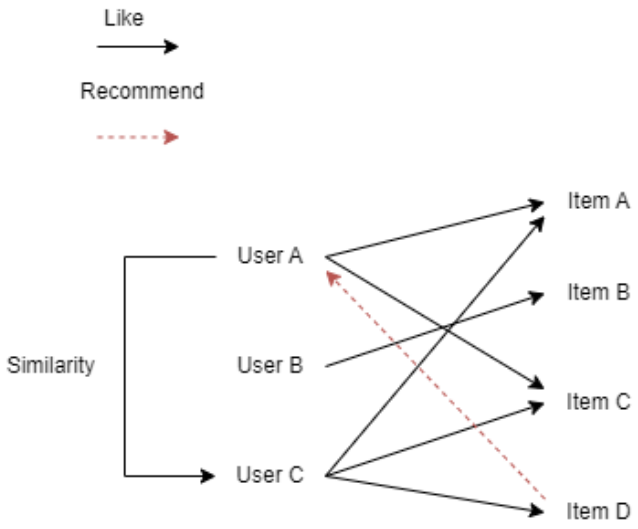

Figure 6. Diagram of user-based recommendation (Cheng et al., 2016).

- *Item-based recommendation.* The difference with user-based recommendations is that changes the similarity between users to the similarity between items. It searches for the nearest neighbors of the unrated item and predicts its score based on the rating value of the neighbor's. At last, it generates a list of top ranked items to the user (Karypis, 2001).

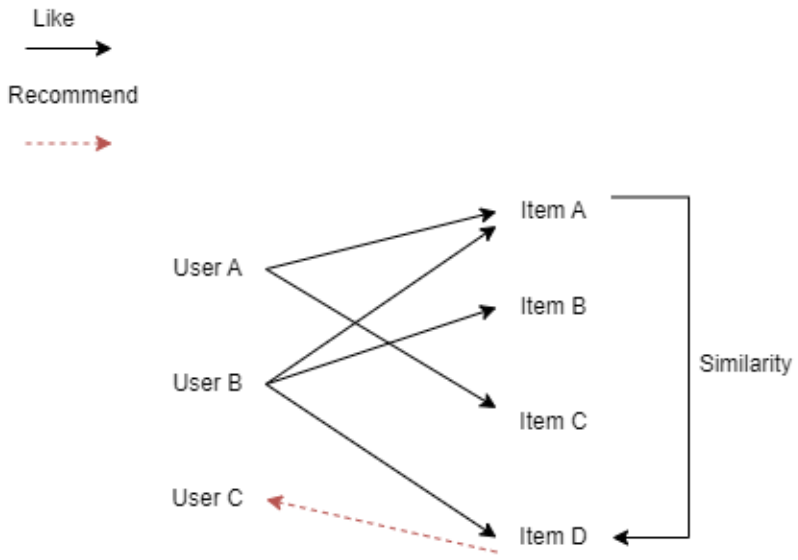

Figure 7. Diagram of item-based recommendation, (Sarwar et al., 2001)

iv. Hybrid recommendations

To combine different recommendation algorithms to improve the quality and performance of the recommendation systems by complementing the advantages of different algorithms to a specific application (Xu et al., 2020).

Robin Burke, Professor in Information systems and Decisions Sciences in California State University, wrote in the article: *Hybrid Recommender Systems: Survey and Experiments*, seven approaches to obtain a hybrid recommendation model (Burke, 2002b). These algorithms were:

- *Weighted hybridization*
- *Switching hybridization*
- *Cascade hybridization*
- *Mixed hybridization*
- *Feature-combination*
- *Feature-augmentation*
- *Meta-level*

Weighted hybridization considers as variables in a linear combination all the recommendation approaches and computes the prediction score giving a weight to each of them and summing up the results (Dwiastuti, 2017).

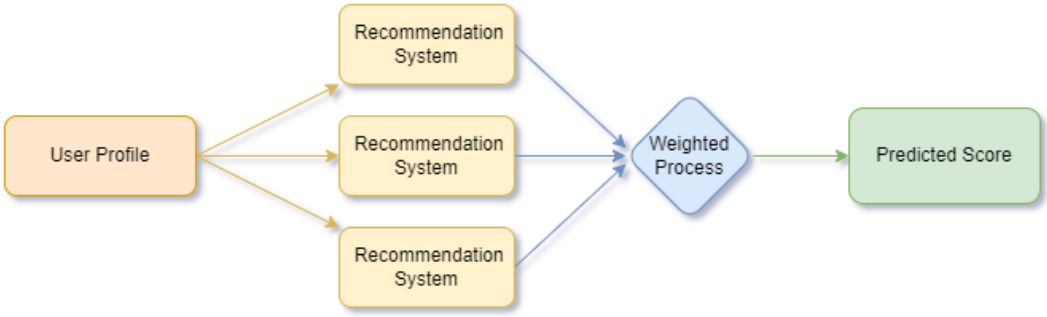

Figure 8. Diagram of Weigthed Hybrid Recommendation System Method, (Chiang, 2021)

While *switching hybridization* builds item-level sensitivity using a criterion to switch between techniques. When an approach does not make a difference in the system it switches to another one. This gives certain sensitivity to the weaknesses and strengths to the recommenders, but also it increases the complexity of the process by adding another parameter (the criterion to switch).

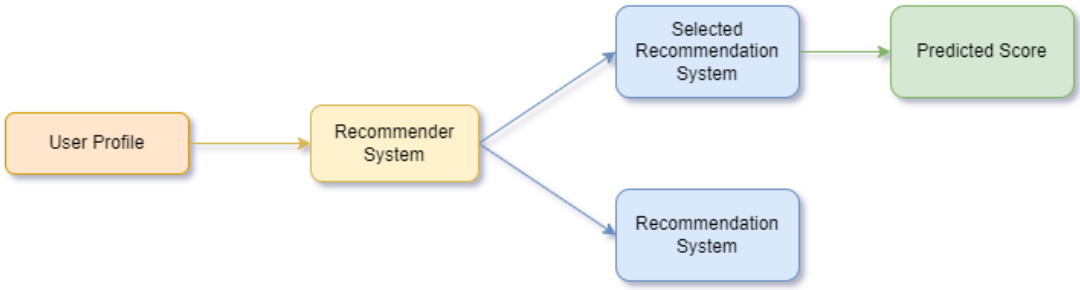

Figure 9. Diagram of Switching Hybridization Method, (Chiang, 2021)

On one hand, *cascade hybridization* follows a process where one technique is done first to generate a uneven ranking of items or users and then another technique is employed to refine the results. The benefit of this method is that the second technique, being a less refine method, is not used if the item is already well-defined by the first one.

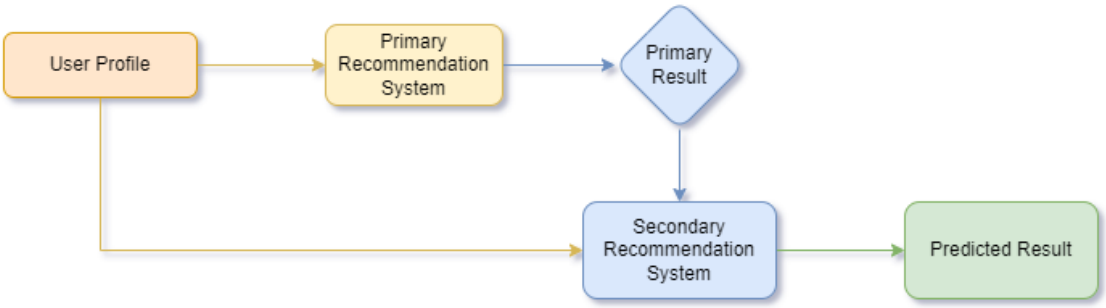

Figure 10. Diagram of Cascade Hybridization, (Chiang, 2021)

On the other hand, in *Mixed hybridization* system, one approach is used to measure some data and another to do the same in another data. At the end the preferences from both techniques are combined to display the final the result. Is the most used after the meta level recommender system

and the advantage is that the method avoids the 'new item', 'new user' startup problem. Moreover, it brings new items where other techniques would eliminate.

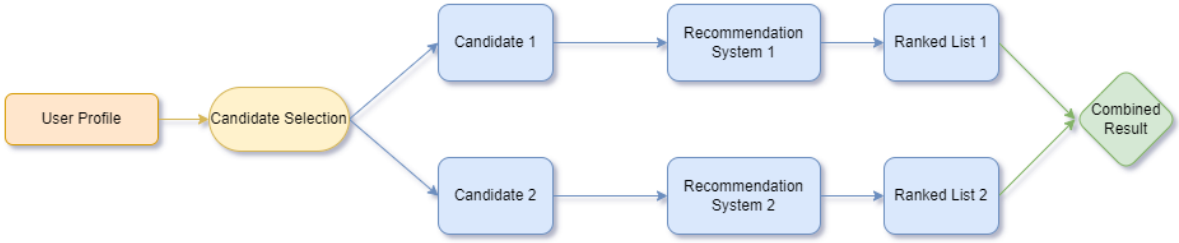

Figure 11. Diagram of Mixed Hybrid Recommender System, (Chiang, 2021)

Regarding *Feature-combination,* the method treats the collaborative information as additional features and the content-based approach is used over the augmented data set. The main benefit is that the method lets the recommender consider collaborative data without exclusively relying on it, reducing the sensitivity to the number of users that have rated the item.

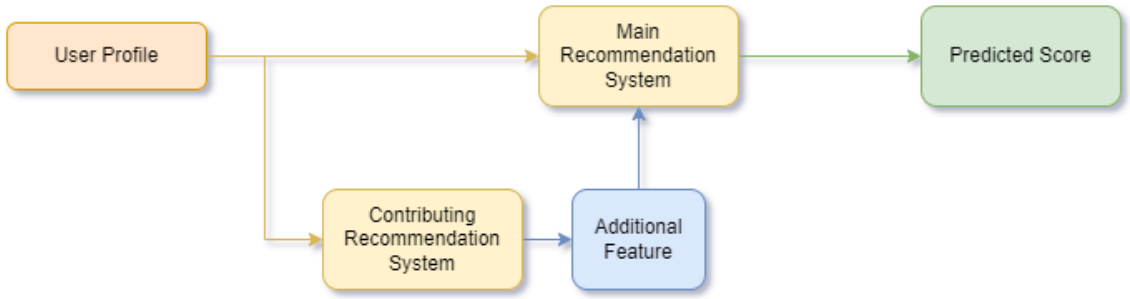

Figure 12. Diagram of Feature-Combination

In the case of *Feature-augmentation,* a method can be used to give a rating or to classify an item and use that information to the processing of another technique. Additional functionalities such as filters can be applied. The advantage is that it can combine two types of recommender systems son the output of the first becomes the input of the second (Bluepi, 2016).

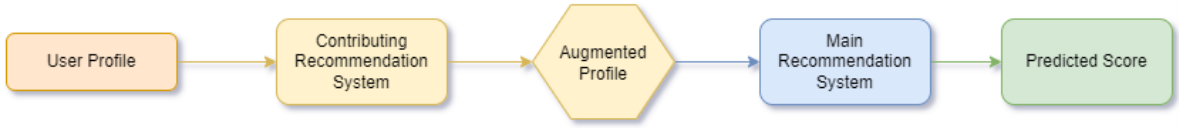

Figure 13. Diagram of Feature-Augmentation, (Chiang, 2021)

Finally, *Meta-level* is similar to the hybrid method: feature augmentation. The main difference is that a recommender system receives as the input dataset a learned model. Is one of the most widely used method for recommender systems (Chiang, 2021).

2.1.2. Evaluation

Recommender systems have conventionally been evaluated in the following ways, being used on or both approaches:

- *Prediction accuracy.* This measure evaluates how well the system's predicted ratings are compared to those that are known, but not used in the training test.
- *Precision of recommendation lists.* In this case, it evaluates a short list of recommendations generated by the model seeing how many of the items match known "liked" elements.

Both evaluation measures can fail in some key particulars, therefore new research has been done to generate other evaluation metrics to supplement these conventional ones.

2.1.2.1. Problems of traditional evaluation

One of the bias this metrics can have, is related to the long-tailed nature of ratings distribution in data sets. When a recommendation technique seeks for high accuracy, it will implicitly bias towards well known items. Those who have more ratings will opaque new items. This bias is problematic as an item that the user already knows is intrinsically less useful than a prediction on an unseen item. Therefore, recommended systems should balance between accuracy and diversity, and work with sensitive algorithms to item distributions.

Another problem of these traditional evaluations is that they are commonly static. The model is trained with fixed databases which ratings are divided into sets: training and test and used to evaluate the usefulness of an algorithm. However, is not considering the user's experience. A user's taste changes over time, new users, and, overall, new items incoming. Many application domains need dynamic recommendations. These changes should be taken into consideration when generating and in the evaluation of algorithms (Burke et al., 2011).

2.1.2.2. Evaluation metrics

A recommended algorithm can be evaluated using accuracy or coverage measurements. In order to decided which type of metric to use depend on the filtering technique used. In the case of accuracy, is the fraction of correct recommendations out of all possible recommendations, while coverage is the fraction of objects in the search space the system can give recommendations to.

To measure accuracy in recommendation filtering there are decision support and statistical accuracy metrics (Sarwar et al., 2001). Which metric is more suitable depends on the features of the dataset and the task the recommender system will perform.

In the case of *statistical accuracy metric,* it evaluates the accuracy by comparing the predicted ratings with the actual user rating. The most used metrics for statistical accuracy are:

- *The Mean Absolute Error (MAE).* Is the most used. It measures the deviation of recommendation from the user's value. The lower the MAE, the more accurate is the recommendation engine. In other words, the more accurate are the predictions of the user's ratings.
- *The Root Mean Square Error (RMSE).* The lower the RMSE, the better the accuracy of the system.
- *Correlation*

Regarding *decision support accuracy metrics:*

- *Reversal rate*

- *Weighted errors*
- *Receiver Operating Characteristics (ROC).* Ideal for comprehensive assessments of the performance of some algorithms.
- *Precision Recall Curve (PCR).*
- *Precision.* Fraction of recommended items that are relevant to the users.
- *Recall.* Fraction of relevant items that are also part of the recommended items.
- *F-measure.* Simplifies precision and recall in one metric. Makes it easy and straightforward to compare between algorithms and within datasets.

All of these metrics are useful to distinguish between high quality items from the available set of items. They differentiate between bad and good items.

Finally, *coverage* is the percentage of items and users that a system can give predictions. If there are no users or nearly no ratings on an item, the prediction can be impossible to make. The coverage calculation can be simplified by creating small neighborhoods for users or items (Isinkaye et al., 2015).

2.1.3. Use cases and Applications

Mainly, business companies use recommended systems to offer a more personalized and engaging user experience to their clients. As a result of their implementation, they increase their sales and retain more customers, helping companies obtain their business goals (Didur, 2021). There are plenty of examples of businesses that implement these algorithms, some of the use cases and applications are:

- YouTube uses collaborative filtering to personalize it network by generating a subset of videos from a large database. A user's activity history is analyzed by recommended systems as an input —videos already seen are labelled as highly relevant. The similarity between other users is used to generate a personalized list of suggested videos, search query tokens, and so on.

- Netflix also tracks the activity history of its users, which films and series are being watched to suggest which video to stream next. Even further, Netflix recommender systems chooses which poster to show when searching to meet better the preference of its users. It creates a personalized page, taking into consideration image selection, search, everything is taken into consideration…

- Instagram powers its recommendation section through the help of the Facebook AI team which created a domain-specific language recommender system called IGQL. Its recommendations are based according to similar accounts interactions and session's recommendations.

- TikTok features "For you", a fully customized content feed for each individual user. Followed accounts, liked and shared videos, comments, created content, all these actions and more determines the user's interest indicator and the recommender systems uses it to generate the feed. However, this social network does boost accounts with more followers.

- Amazon is well known for using recommended systems within its products, it uses the user's history activity to match relevant items, reviews, and searches to recommend meaningful

items to its clients. It boosts the most relevant items from the most sold ones to the ones with the best ratings.

Although the mention companies are considered social networks, there are plenty of other applications domains that implements recommended systems —learning materials, television programs, tourism, financial services, books, etc.

Despite promoting different products, with engines design for different purposes, all have the same outcome: longer, better, and deeper engagement which results in more sales —therefore, revenue.

2.1.4. Ethics and challenges of Recommender Systems

As shown, recommender systems can improve our lives, by filtering information and delivering meaningful recommendations scoping through all the available options.

However, there are ethics recommender systems should meet, avoiding social risks from individual privacy to social relationships. Recommender technology should be deployed carefully to diminish these risks (Konstan & Riedl, 2012).

Privacy

A key challenge of recommender systems is that using a physical location or a user's friends search history, may discourage the use of personalization technologies due to privacy risks. Moreover, with behavioural profiling, systems track users through cookies for long periods of time, identifying a user between consecutive sessions.

This presents a potential privacy risk due to this information falling into undesired central servers. Users can not want to provide their preference and behavioural data to potentially untrusted third parties. This can affect the user's perception of the goal of recommended systems. In the case of collaborative filtering, data from a user is enrich with information from others, that can lead to privacy risks.

Furthermore, user attitudes towards behavioural profiling states that most Internet users believe their privacy has been invaded through Internet information and only 28% would be comfortable with advertise recommendations that uses web browsing history (Aleecia McDonald & Lorrie Faith Cranor, 2012).

Data collection	User-provided information	Tracking user actions	Automatic context information
User model creation	Inference-based analysis	Inference-based analysis	Collaborative Analysis
Adaptation	Only to the user	To user's social relation	To the World Wide Web

Figure 14. Privacy management in recommended systems, (Toch et al., 2012)

In order to mitigate privacy risk in recommended systems, it is desirable to give the user the most control possible of the data the system is feed with.

Social effects

As recommended systems are becoming widespread, they are also being designed to fit social contexts. This can cause a challenge as it can dictate the norm for new items. When a user is presented with a high rated item it is subjected to also rating it with a high score, influenced by the bias and probably affecting other users' ratings too.

Another social risk is that humans tend to spend more time to others with whom they share religious, political, and cultural beliefs. Therefore, recommended systems will make easier for people to find others with similar interest, and it can create isolated sub communities (Van & Brynjolfsson, 1997). Other studies state that use of recommender systems can lead to a "winner-take-all", where a leading item will dominate market segments (Fleder & Hosanagar, 2009).

Therefore, recommended systems have to be diverse and understand it social impact to offer a positive impact to it users.

Robustness and manipulation resistance

One of the main challenges of recommend systems is to be robust and be able to withstand attacks such as dishonest ratings. Malicious users can try to alter the system behaviour by introducing misleading ratings distorting the information presented to genuine users. Therefore, understanding the nature and impact of these attacks will lead to more secure and robust recommendation systems (Sandvig et al., 2008).

Several studies have demonstrated that misleading information has a great impact in the models of recommendation systems. As said in the previous point, an item can opaque others due to having a boost of good ratings. To avoid this, robust and manipulation resistance algorithms are important. Furthermore, organizations or even users can have a vested interest in having their items boosted in recommended systems and may try to manipulate it to their own profit.

Therefore, many collaborative filtering is vulnerable to shilling[1] attacks. To minimize their effect, the recommended system should give more weight to honest and constant users and distinguish ratings added by attackers (Resnick, 2008).

Moreover, research have established than even blunt attacks are able to change recommendations and can be difficult to detect. Although, in recommender systems with a huge community its effect on popular items is limited because of the large countervailing data available (Lam & Riedl, 2004), algorithms should be able to minimize manipulation through shilling attacks.

[1] Attacks on recommender system behaviour is known as a "shilling" attack or "profile injection" attack.

2.2. RECOMMENDED SYSTEMS APPLIED TO BOOK RECOMMENDATIONS

As Maxim Gorki, five times nominee of the Nobel Prize in Literature, once said: “books are the ladders of human progress”. Without them we would lack of a place where imagination has no limits, knowledge is written down for future generations and history is available for everyone to discover and learn.

Books keep us informed, awakens our imagination, fuels our inspiration and makes ideas emerge. Allows us to connect and put ourselves in the shoes of other people and characters. Furthermore, as demonstrated by the study of Washington University in 2009 by Nicole K. Speer, “Readers mentally simulate each new situation they encounter in a narrative. The details of actions recorded in the text are integrated into their personal knowledge of past experiences”, (Speer et al., 2009). This means that not only books feed the imagination and promotes concentration skills, but also it improves our social skills, for example the empathy (Saiz, 2012).

It is questionless that books are fundamental for human progress, not only regarding industry advanced, society or knowledge, but also for human relationships, understanding one another, linguistics, and social skills (Manrique Sabogal, 2015). Therefore, we, as part of society, should support and encourage reading, not only in childhood, but for every vital stage of life as we can only benefit ourselves from it.

However, we should note that books are a valuable item, many describe it as a luxury, so it is clear that choosing a book to read is a matter of importance. As talked in the introduction of this thesis, the book industry is drowned in an abundance of books that affects its consumption and future stability. This problem needs to be addressed from the industry itself, but, for now, in this project it is proposed the implementation of recommendations systems to try alleviating the situation by improving the decision making of readers and, also, provide a service for bookstores and libraries that are affected by the growth of the publishing industry.

Regarding the studies made about recommended systems applied to books, they apply different approaches, but all agree that the current state of the industry makes it nearly impossible for the readers to find their next lecture without being drowned with options (Lambert, 2014).

Consequently, there is an intrinsic need for the consumer to find a useful and simple way to filter all the available options and find within their interest and previous liking their next purchase. Becoming a distinct service for publishing houses, libraries, and bookstores to count with a recommended system to offer to their clients.

To create, train and provide a recommendation system for books, firstly we need to understand how readers behave. The study carried out by Written Word Media, a UK marketplace for book promotion for self-published authors, concluded the following:

In the first place, the most important factors when deciding which book to read were, by order of importance, description, author, book cover, review score and number of reviews (Noblit, 2023). This implies that a good recommendation system should take into account the description and author name, meaning NLP techniques, and for book covers, image recognition (deep learning).

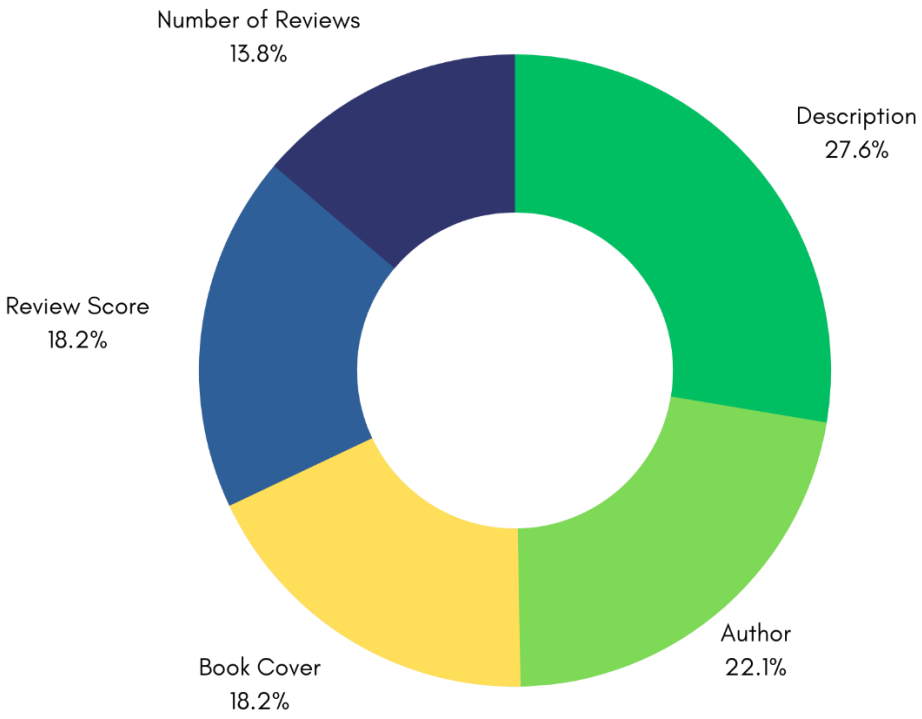

Figure 15. The five most important factors for selecting a book to purchase

Moreover, regarding reviews, 30% of readers say that sometimes they write a review, 26% rarely does and 21% state that they never write one. Only 23% of readers are active reviewers which means that number of reviews won't match the actual number of people that have read a book and that a low percentage leave one, even though it is an important factor when choosing a book to read.

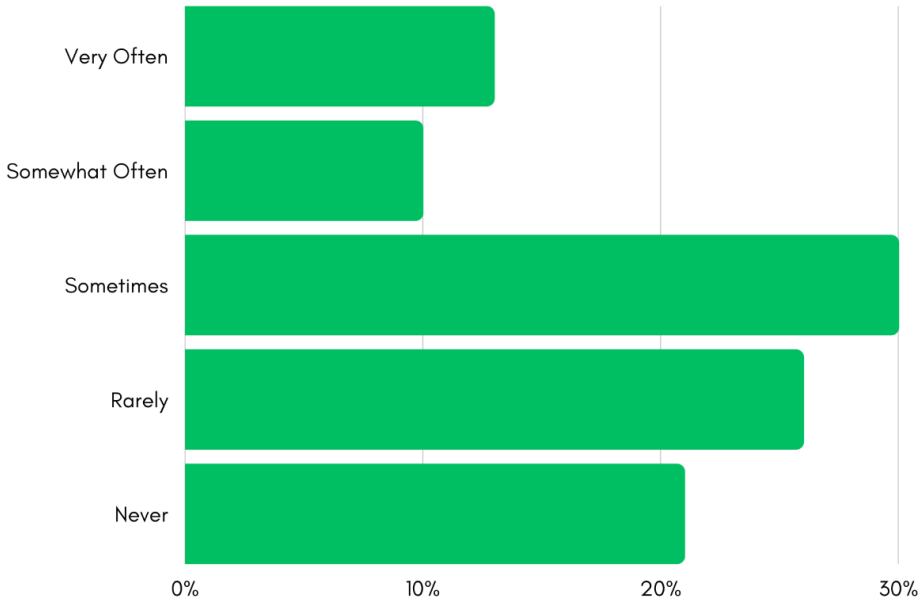

Figure 16. How often readers leave a review?

Furthermore, regarding how readers discover books or authors, most readers find very useful and determinant the recommendations of newsletters from book webpages (75%), being the second source for readers to find their next read where Amazon and GoodReads, being fourth friends and family recommendations and in fifth place, social media (Noblit, 2023).

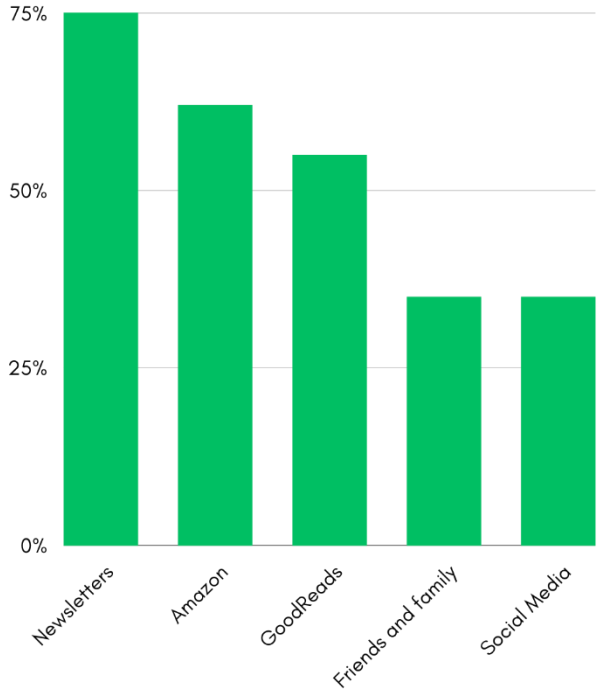

Figure 17. How do readers discover new books and authors?

Therefore, recommendations systems applied to books should be a relevant and useful tool for readers. It will help them choose their new read as they use those key factors on their algorithms. To have a better understanding of them, in the following subchapter it shows the most relevant studies regarding these techniques.

2.3. RELATED STUDIES

The aim of this dissertation is to develop a recommended system applied to the greatest platform of book reviews to offer a meaningful recommendation of books to the users. Therefore, it is important to learn and understand the previous studies carried out by researchers. This will allow us to further improve and correctly apply the techniques in order to develop correctly a recommended system to a books ratings dataset.

Some of the research done in this topic used collaborative filtering (Harada & Masuda in 2010), association rules (Tsuji et al, 2014), content based (Jayaram et al, 2022) and hybrid Techniques (Chandak et al, 2015 and Srivastava, 2022). They all used different techniques to create a recommended system suitable for recommendations systems.

For example, the effectiveness of using library loan records for recommended systems was studied by Tsuji et al. They wanted to know which approach or combination was more favorable to apply to a

recommendation system for books used by students in the T University Library. To do so they used data were (1) confidence and support with an association rule based on the loan records, (2) similarities between book titles, (3) matches/mismatches between the Nippon Decimal Classification, NDC, categories of a book, and (4) similarities between the outlines of the books in the *BOOK Database.* Furthermore, they count with 32 students from the university to rate the recommendations (Tsuji et al., 2014).

The results showed that the combination of (1), (2), (3) and (1), (2) were rated more favorably by the subjects than the others. The evaluation method was conducted by showing the recommendations of each approach to the subjects and they described their level of interest in each book using a five-point scale.

Their study concluded that the best approaches could work for different type of students, although they encountered a problem when using simple cosine measure for calculating the similarities between titles as newer books tend to have longer titles while older books tend to have shorter and simpler titles, so the recommendations tended to recommend older books (Tsuji et al., 2014).

Regarding similarity coefficients, (Kurmashov et al., 2016) used Pearson correlation based on Collaborative Filtering and applied in on online book recommendation system and evaluated through an online survey. While (Ayub et al., 2018) proposed Jaccard Similarity to find alike items and used with nearest neighbors CF.

Another study, *Introducing Hybrid Technique for Optimization of Book Recommender System,* where the authors studied the main drawbacks of the most used techniques: Collaborative Filtering and Content Based and combine them to overcome them. CF has a problem when there is not enough data for user-item ratings, whereas CB does not used ratings data and therefore overcomes the cold start problem. In the research they combined both methods and joined the recommendations generated by both techniques.

The authors concluded that due to sparsity in ratings, CF accuracy decreased when increased the number of users, the CB accuracy was slightly better than CF although it gave more relevant results thanks to its ability to predict out of the box. Finally, the hybrid technique outperformed and overcome the drawbacks of CF and CB while also increasing the recall. Furthermore, they overcome the cold start problem for new items and user by adding demographic attributes (Chandak et al., 2015).

To sum up, the literature survey suggests that recommended systems are used by a large scale of online website markets to increase their sales by offering items to customers that are relevant to them. Although these RS can suffer from diverse problems such as data sparsity, trust, cold start, privacy, and scalability. Therefore, there is a need for improved recommendation systems or implementation of hybrid techniques that can overcome these challenges.

This project aims to study the different techniques while using diverse similarity functions to obtain those models that perform better when applied to books.

3. METHODOLOGY

The approach carried out in this study has tried to answer the research questions that arose by the literature review and were proposed in the dissertation chapter Objectives of the study. Many articles applied different techniques to solve the problem of how to make it easier for consumers to find their next reading. Although they lack a comparison between them and focused on one or two techniques they wanted to study. Some of these articles talked about the disadvantages of Collaborative Filtering while others explained the drawbacks of using cosine similarity for recommendations. Therefore, the objectives stablished were:

i. Study the different techniques used in recommended systems, such as Collaborative Filtering, Content-Based and NLP approaches;
ii. Develop a recommended system with the best approach in order to solve the problematic described above and apply it to a practical case;

That way, this study will analyze the different techniques applied to recommended systems in one, while trying to identify the main problems of each and extract the best one for this application and database.

To address the research question and achieve the research objectives, the work has been framed in the Cross-Industry Standard Process for Data Mining, CRISP-DM, used to extract value from data for Data Mining projects (Martinez-Plumed et al., 2021) as mentioned in the **¡Error! No se encuentra el origen de la referencia.** subchapter.

Step	Phases	Description
1	Data Understanding	**Initial data collection, search for data quality problems identification:** Download of data from GoodReads. Study of data collection.
2	Data Preparation	**Data pre-processing:** Attribute selection, data transformation, cleaning dataset…
3	Modelling	**Modelling techniques selection and application, parameters calibration:** Create different models following the literature review concepts to design an accurate recommender system.
4	Evaluation	**Business objectives evaluation:** Validate the best model by comparing the accuracy and identify if it answers the problem. If necessary, redesign the model.
5	Deployment	**Result model deployment.**

Table 1. Phases and description of CRISP-DM methodology

For our project, based on the CRISP-DM methodology, these are the steps and phases followed:

- **Data Collection**: in this phase the information is downloaded from the database, data is obtained by web-scrapping the GoodReads web. Once this is done, data is processed in order to keep the files cleaned, organized and with the necessary columns.
- **Data preprocessing:** as said, the data has been written by users and extracted from the webpage, this means data can be wrong or incorrect, therefore we need to detect outliers or this misinformation, clean the data for NLP, drop unnecessary columns and most importantly, visualize the data to understand its distributions and values.
- **Modelling:** once we have the desired final database, we split the data into training, validation and test and apply the different models. In this case, we are going to use the collaborative filtering, but also content based, applying NLP.
- **Evaluation:** in this phase, the models are evaluated in order to choose between those who perform better than the others and adjust their parameters. Those models that seems to perform the best are further evaluated.
- **Deployment:** finally, in this phase, we have trained the best approaches and are then applied to a specific user. The results are then compared and discussed.

In summary, we can see the steps and phases done in Figure 18:

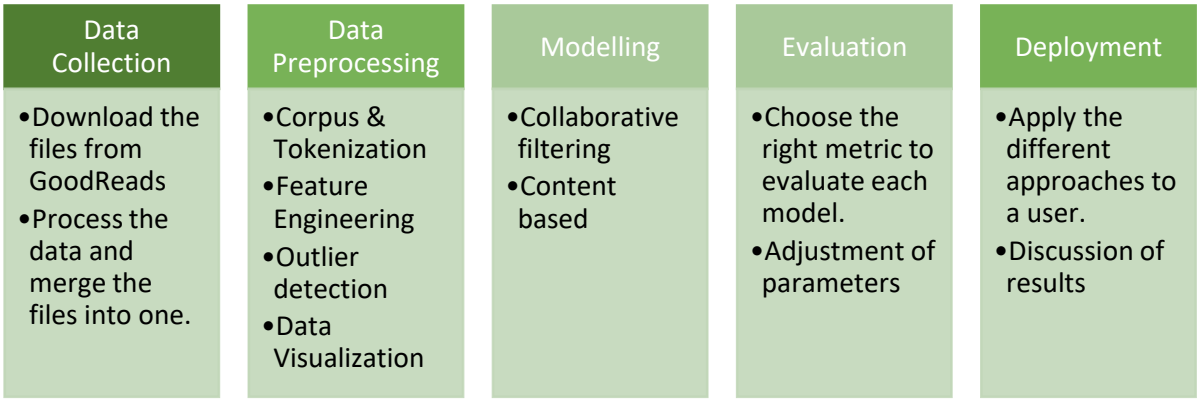

Figure 18. Phases implemented in the project based on CRISP-DM

3.1. DATA COLLECTION

The collection of initial data has been provided by Goodreads, the biggest website for book lovers. As of December 2020, it no longer issues developer keys for its API, but an online dataset can be obtained with more than 2 million books with updated data called Goodreads Book Graph Datasets (*Goodreads Datasets*, 2022).

The datasets used are called *goodreads_books_young_adult.json.gz* where we can find more than 100 thousand records with information of the books in the category of young adult. The dataset counts with 11 variables from the author identification number to details of the book such as the title, description, and number of pages.

The second dataset used is called *goodreads_reviews_young_adult.json.gz* with more than 2 million records. This collection has the information about the individual rating written by the user. This dataset has 10 variables.

Finally, for extracting the author's names, we use the database called *goodreads_book_authors.json.gz,* which counts with nearly 1 million records and has 5 variables, from its name, identification number and the average rating of its books.

The analysis of this database has been done for either a recommended system with collaborative filtering or a content-based algorithm (NLP approach). Therefore, there will be different analyses developed for each case. This is because those approaches need different type of information. In the case of the collaborative filtering, it will recommend items that similar users have read in the past, being necessary a punctuation analysis, while the content based will recommend items similar to items read in the past based on implicit data (finds the attributes or keywords most related to the product that a user likes) and therefore need a relationship analysis.

The information provided by the *goodreads_books_young_adult.json.gz* database looks like this:

#	Column	Non-Null Count	Dtype
0	author_id	117691 non-null	object
1	text_reviews_count	117691 non-null	object
2	country_code	117691 non-null	object
3	average_rating	117691 non-null	object
4	description	112725 non-null	object
5	publisher	95613 non-null	object
6	num_pages	90061 non-null	object
7	publication_year	99046 non-null	object
8	book_id	117691 non-null	object
9	ratings_count	117691 non-null	object
10	title_without_series	117691 non-null	object
11	Is_Series	117691 non-null	int32

Figure 19. Structure of books file

- ***Author_id:*** is the identifier of the author.
- ***Text_reviews_count:*** is the number of reviews that has text along the rating.
- ***Country_code:*** is the code of the country the book has been published.
- ***Average_rating:*** is the average rating of the book.
- ***Description:*** a brief description of the content of the book.
- ***Publisher:*** the name of the publisher.
- ***Num_pages:*** how many pages the book has.
- ***Publication_year:*** the year the book was published.
- ***Book_id:*** the identifier of the book.
- ***Ratings_count:*** how many ratings the book has received.
- ***Title_without_series:*** is the book title (although the name says that is has no series, it does).
- ***Is_series:*** is a Boolean data, 1 if the book belongs to a series, 0 if not.

This first file only gives us information about the books and its average rating. But we do not have information about the reviews or the users. For that reason, we downloaded also the *goodreads_reviews_young_adult.json.gz file*, with the following structure:

```
 #   Column        Dtype
---  ------        -----
 0   user_id       object
 1   book_id       object
 2   review_id     object
 3   rating        int64
 4   review_text   object
 5   date_added    object
 6   date_updated  object
 7   read_at       object
 8   started_at    object
 9   n_votes       int64
 10  n_comments    int64
```

Figure 20. Structure of Reviews file

- ***User_id:*** identifier of the user.
- ***Book_id:*** identifier of the book.
- ***Review_id:*** identifier of the review made.
- ***Rating:*** value from 0 to 5.
- ***Review_text:*** a text review of the book read.
- ***Date_added:*** date when the user added the book to its profile.
- ***Date_updated:*** date when the last change was done.
- ***Read_at:*** when the user read the book.
- ***Started_at:*** when the user started to read the book.
- ***N_votes:*** count of how many likes a review or rating has.
- ***N_comments:*** count of how many comments a review or rating has.

Some of these columns has too many missing values or are not relevant for the analysis. Therefore, we deleted ***date_added*** and ***date_updated*** and created a new column named ***Days_Read*** that is the number of days a user took to read the book. To do it the columns ***Read_at*** and ***Started_at*** had to be transformed to datetime type and then subtracted. Once done, we dropped those two columns. We also changed the name of ***rating*** to ***review_rating*** so we understand that this column refers to the user rating and not the book average rating. In the end, we have a data frame with 8 columns.

The only data we are missing is the name of the author. So, we downloaded a third file called *goodreads_book_authors.json.gz,* with the following structure*:*

	average_rating	author_id	text_reviews_count	name	ratings_count
0	3.98	604031	7	Ronald J. Fields	49
1	4.08	626222	28716	Anita Diamant	546796
2	3.92	10333	5075	Barbara Hambly	122118
3	3.68	9212	36262	Jennifer Weiner	888522
4	3.82	149918	96	Nigel Pennick	1740

Figure 21. Structure of Authors file

- ***Average_rating:*** is the average rating of the author.
- ***Author_id:*** is the identification number of the author.
- ***Text_reviews_count:*** is the number of reviews the author has received.

- ***Name:*** is the name of the author.
- ***Ratings_count:*** is the number of reviews the author has received.

The only columns we are interested in are ***author_id*** (to merge with books dataset) and ***name***, the rest of the columns are dropped.

3.2. DATA PREPROCESSING

Before doing a merge of the files, we checked the following:

- Regarding the books file, we check if there are duplicated books (same ***book_id*** and same ***author_id***) and we saw that there were. We dropped 31 records and kept 117691.
- Regarding the books file, we drop all those records that have missing values. For doing this we first replace empty strings for NaN, keeping 81082 records.
- Regarding the reviews file, we check if there are any user that have rated a book more than once. There were none.
- Regarding the reviews file, we filter the database to keep only those users that have rated at least 200 books, reducing the database from 2 million to 300 thousand records.
- Regarding the author dataset we drop the duplicated records.

Once a preliminary preprocessing process has been completed, we can then proceed to merge the three files. First the book and author files by ***author_id*** with an inner join, resulting in a data frame with 78482 records. Before joining with the review file, we changed the ***name*** column by ***author_name*** and then dropped the ***author_id*** column. Finally, we join the previous resulting data frame with the review file by ***book_id***, having as a result a database with 277780 entries.

```
 #   Column               Non-Null Count   Dtype
---  ------               --------------   -----
 0   text_reviews_count   277780 non-null  int64
 1   country_code         277780 non-null  object
 2   average_rating       277780 non-null  float64
 3   description          276100 non-null  object
 4   publisher            271709 non-null  object
 5   num_pages            277780 non-null  int64
 6   publication_year     277780 non-null  int64
 7   book_id              277780 non-null  int64
 8   ratings_count        277780 non-null  int64
 9   title_without_series 277780 non-null  object
 10  Is_Series            277780 non-null  bool
 11  BookSeriesInfo       156290 non-null  object
 12  title                277780 non-null  object
 13  author_name          277780 non-null  object
 14  user_id              277780 non-null  object
 15  review_id            277780 non-null  object
 16  review_rating        277780 non-null  int64
 17  review_text          277664 non-null  object
 18  review_n_votes       277780 non-null  int64
 19  review_n_comments    277780 non-null  int64
 20  Days_read            277780 non-null  float64
dtypes: bool(1), float64(2), int64(8), object(10)
memory usage: 44.8+ MB
```

Figure 22. Structure of the final database

3.2.1. Corpus & Tokenization

Regarding the columns of type string, we need to clean the corpus by:

- Joining the name and surname of the author by '_' to later apply techniques such as word embeddings.
- We do the same for publisher column, while also removing quotations marks, unnecessary punctuation…
- For keywords and word embeddings it is also necessary to make text lowercase, remove text in square brackets, remove punctuation and unnecessary links and words with numbers. We clean the text of the description and review text columns.
- Finally, we also remove from the title unnecessary punctuation or quotation marks.

3.2.2. Feature Engineering

Once we have our desired database created, we need to apply further feature engineering regarding the analysis for a recommended system based on NLP.

It was necessary to split the column called ***title_without_series*** into ***BookSeriesInfo*** that, by looking for a certain pattern, extracted the information regarded to the series of the book. After, we created a new column called **title** with only the book's name and drop the used column. Then a new column was created by joining ***BookSeriesInfo***, ***author_name, publisher*** and ***country code*** named ***bow***.

In the other hand, regarding the collaborative filtering, some modifications had to be done to the ***book_id*** and ***user_id*** columns.

- The ***book_id*** column did not follow a sequential order, meaning that, when creating the matrix for the collaborative filtering, there would be created unnecessary rows. To fix this problem, the column was factorized, giving a unique identifier a new sequential number from 0 to 1049.
- Regarding ***user_id***, this column was anonymized and therefore contained an encoded string value. The values were also transformed to have a sequential, numerical identifier. Values went from 0 to 29712.

Finally, we check with a correlation matrix if there are any columns that are redundant or irrelevant for our objective, Figure 23. The correlation matrix indicates which percentage of information is also explained in another column.

For example, the columns ***ratings_count*** and ***text_reviews_count*** correlate in a 92%, while the columns ***review_n_votes*** and ***review_n_comments*** also correlate moderately in a 64%.

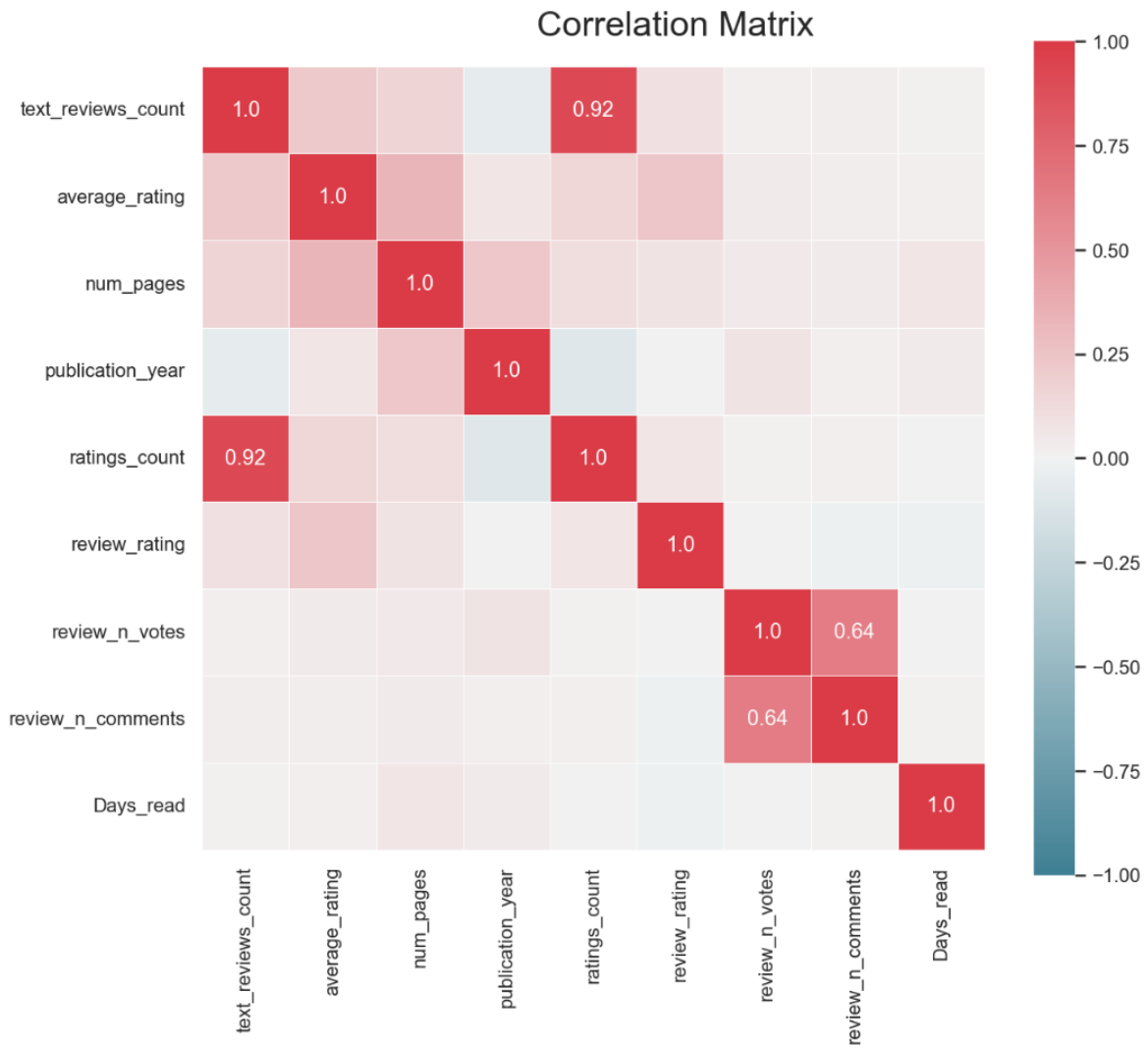

Figure 23. Correlation Matrix

The correlation matrix on its own cannot tell us if a column should be or not kept, there are other techniques that can help into taking a decision. In this case, the business understanding, and the application of the model is sufficient. Regarding the count of ratings or text reviews, one column is sufficient for the content-based model.

In the other hand, the comments column can be also dropped as, by knowledge of the GoodReads app, these notes don't really give us relevant information, while the number of votes does.

The higher the number of votes, the higher the review is shown on the book page. So, at the end, we drop ***text_reviews_count*** and ***review_n_comments.***

3.2.3. Outlier detection

Thanks to the descriptive statistics of the numerical columns, we can easily see some outliers or even wrong data on the columns such as:

	count	mean	std	min	25%	50%	75%	max
text_reviews_count	270146.0	3075.922316	10593.143457	1.00	107.00	451.00	1875.00	142645.0
average_rating	270146.0	3.855400	0.286123	1.57	3.68	3.86	4.06	5.0
num_pages	270146.0	330.567108	148.261292	0.00	276.00	335.00	388.00	52015.0
publication_year	270146.0	2011.962346	9.564905	2.00	2011.00	2012.00	2014.00	2021.0
book_id	270146.0	12203.512871	8363.987709	0.00	5091.25	11177.00	18894.00	29712.0
ratings_count	270146.0	49350.781796	277918.912270	0.00	537.00	3211.00	17717.00	4899965.0
user_id	270146.0	477.317832	304.741963	0.00	210.00	453.00	737.00	1049.0
review_rating	270146.0	3.523254	1.328369	0.00	3.00	4.00	4.00	5.0
review_n_votes	270146.0	2.473844	14.507852	-3.00	0.00	0.00	1.00	1366.0
review_n_comments	270146.0	0.597699	3.872690	-2.00	0.00	0.00	0.00	394.0
Days_read	270146.0	8.202757	69.783418	-2969.00	1.00	1.00	3.00	7322.0

Figure 24. Descriptive statistics of the data frame

- There are books with 0 pages.
- In ***publication_year*** the minimum value is 2.
- In ***review_n_votes*** there are values below 0, this is not possible in the app as there are no dislike bottom.
- In ***Days_read*** we have values that goes up to -2969 and high as 7322 (meaning a user took 20 years to read a book, it could be possible but as the app was lunch in December 2006, we understand it as a wrong data).

For the column ***Days_read*** we apply to different methods, the first one is a filter that keeps values higher or equal to 1 and smaller or equal to 365 days. This method kept 99.58% of the data. The second method was to apply the IQR method, filtering by the upper and lower limit, keeping 87.57% of the data (this means that the lowest value was -2 days). Therefore, to keep the maximum data, while having coherent values, we apply the first method.

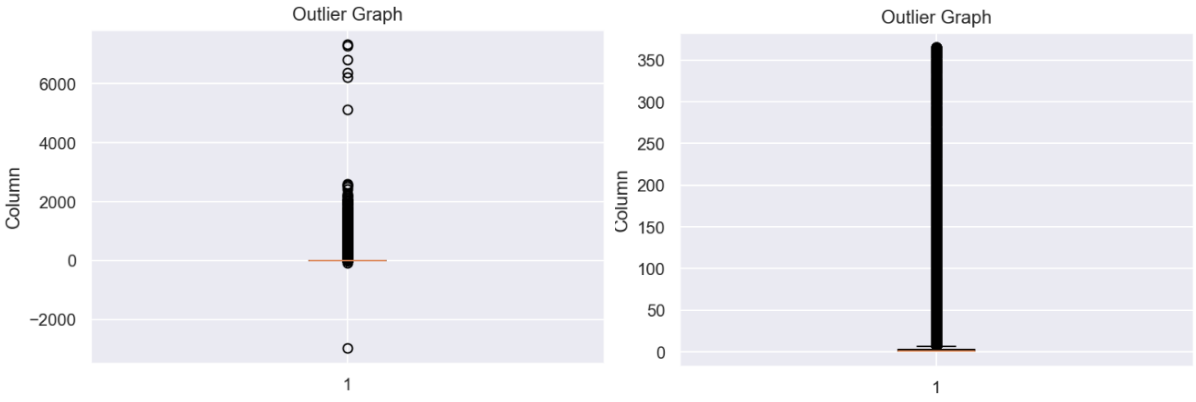

Figure 25. Outlier graph of Days_read before and after the outlier removal.

We do the same for ***publication_year***, the result of the IQR method kept 93.47% of the data while the filter method (1600 <= Year <= 2021) kept 99.58% of the data. We applied the second method.

For ***num_pages*** we applied a filter: 50 <= Number of pages <= 1000 (poetry books have fewer pages but are neither an outlier nor wrong data). For ***ratings_count*** we can see how the standard deviation

is tremendously high. Therefore, we apply a filter to keep only the books with less or equal 600 thousand ratings. Finally, we also applied a filter to remove those records with a number of votes lower than 0. The final data frame has 1050 unique users, 28467 unique books and 9953 unique authors.

	count	mean	std	min	25%	50%	75%	max
average_rating	259630.0	3.849871	0.285217	1.57	3.68	3.86	4.05	5.0
num_pages	259630.0	336.399961	96.646160	50.00	280.00	336.00	388.00	987.0
publication_year	259630.0	2012.043539	3.839683	1946.00	2011.00	2013.00	2014.00	2021.0
book_id	259630.0	12403.342487	8364.582267	0.00	5219.00	11487.00	19378.00	29712.0
ratings_count	259630.0	25593.158915	65258.238402	0.00	553.00	3216.00	16933.00	550237.0
user_id	259630.0	477.365809	304.674687	0.00	210.00	452.00	736.00	1049.0
review_rating	259630.0	3.516381	1.326729	0.00	3.00	4.00	4.00	5.0
review_n_votes	259630.0	2.479113	14.142742	0.00	0.00	0.00	1.00	1366.0
Days_read	259630.0	4.668220	17.390919	1.00	1.00	1.00	3.00	365.0

Figure 26. After the outlier removal, the final descriptive statistics of the dataframe.

3.2.4. Data Visualization

One of the key aspects of the CRISP-DM is the data understanding, therefore, before applying any modelling techniques we visualized the data to have a better picture of the database.

For example, the scatter plot of the publication year and the ratings count gives us the spread of the data: the newest books have a higher number of ratings in general. This is expected as the app was lunch in 2006 and people tend to read recent novelties rather than classic literature, (Tobar, 2013).

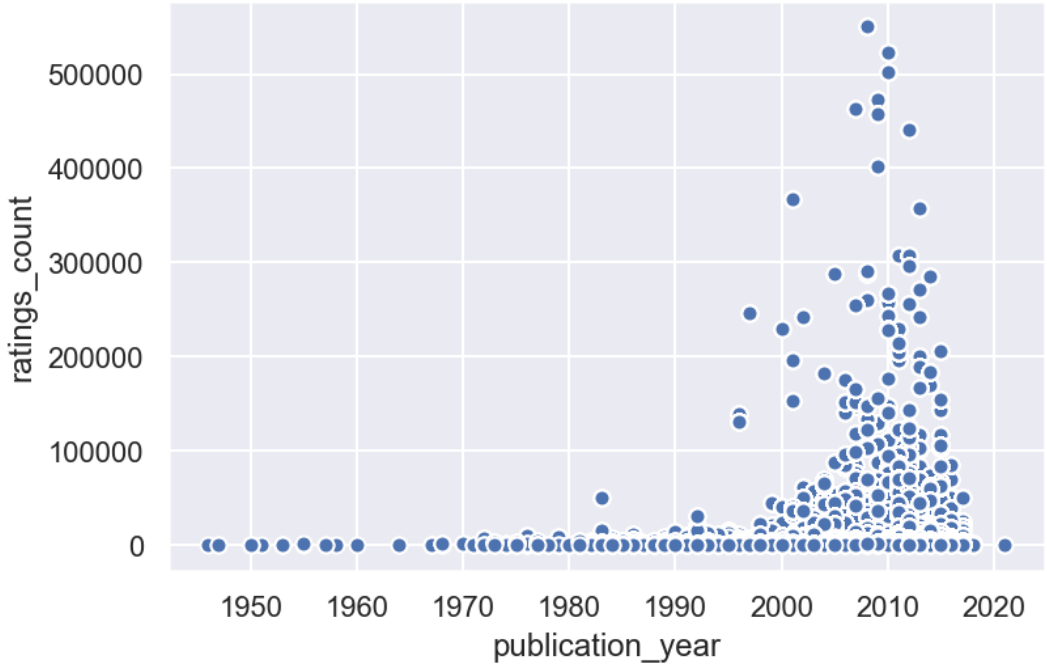

Figure 27. Scatter plot of year of publication versus number of ratings.

In the other hand, we can see that the distribution of the publication year of the books have grown, Figure 28, matching with our understanding of the growth of the industry nowadays. The dataset was updated in 2019 explaining the drop of values in the most recent years.

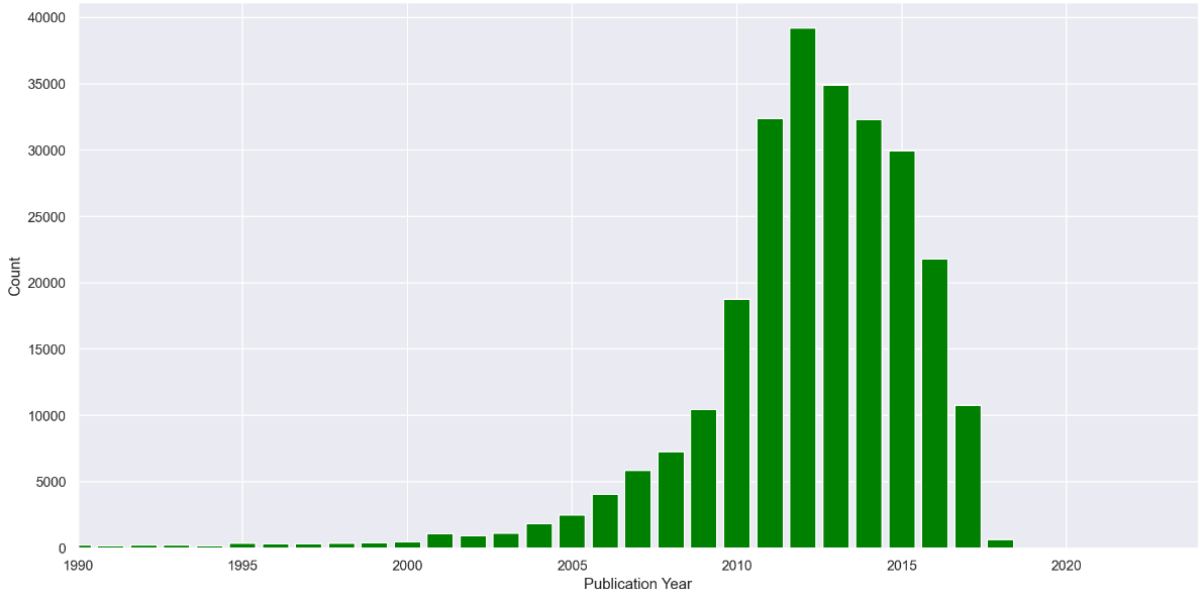

Figure 28. Bar plot of the distribution of the publication years.

Another useful graph is the distribution of the average rating of a book, Figure 29. We can see that most books have an average between 3.5 and 4.3, following a gaussian distribution, which is expected in ratings.

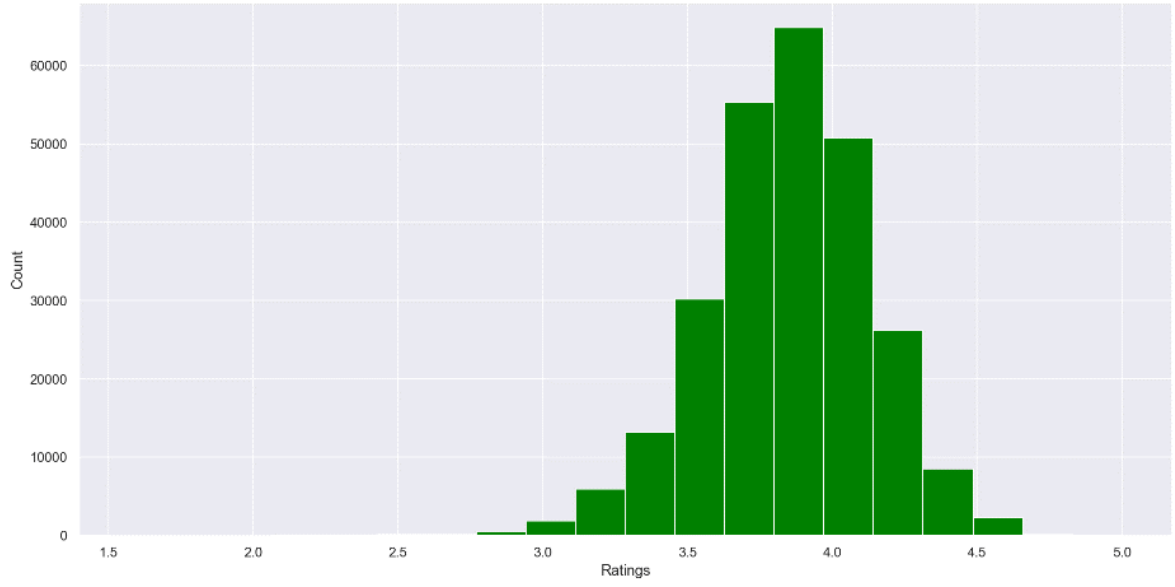

Figure 29. Average rating distribution graph

In the other hand, most users rate between 3 and 5 stars, which correlates with the output of the average graph, Figure 30.

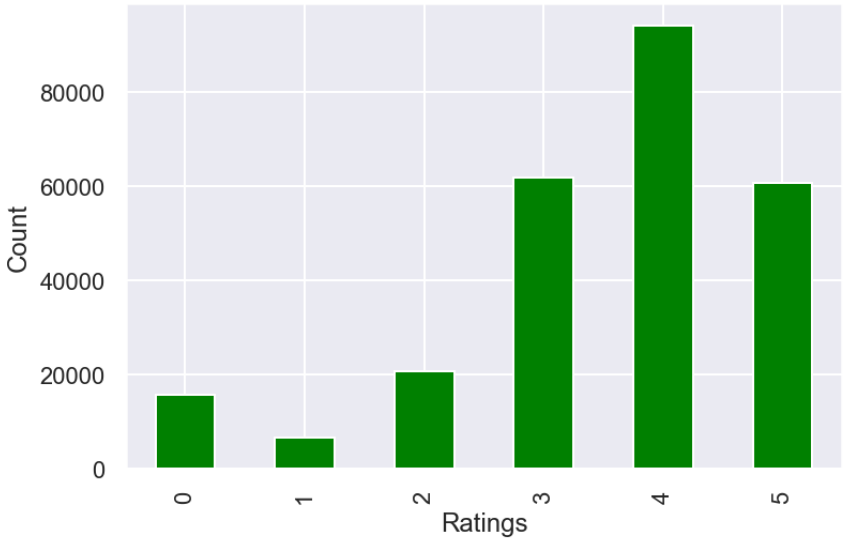

Figure 30. Bar plot of the user's rating of the books.

3.2.5. Split into Training, Validation and Test

It is known that the biggest problems with training a Recommendation system is that they are unable to generalize the model because it was given little information, underfitting, or, they have too much and rather than detecting the underline patter it adjusts extremely to the seen data, resulting in overfitting. To avoid this, and to ensure that our recommended systems work properly we split the database in Training, Validation and Test.

The subset of **training** is used to train the data given to the recommended, these are visible data for the model. The results of this subset do not represent the real performance of the model.

The subset of **validation** is used to try different models that have been trained in the training subset. This is helpful to adjust parameters of the models. This data has not been seen by the model to be able to know if the model is underfitting, overfitting and the overall evaluation of the models can be compared.

The subset of **test** are data not visible for the model, with this subset is measured the real performance of the recommendations. In this subset we apply precision metrics such as the Mean Average Precision, mAP, the Normalized Discount Cumulative Gain, nDCG and the F1-score.

In Machine Learning systems, the split of data is generally split into 80% training, 20 % test, but also in case of Big Data, it can be enough 98% training, 1% validation and 1% test, (Brownlee, 2020).

For our data collection we have decided to split into 80% training and 20% test. And for the training set, as the main objective is to maximize it to be able to train the models with more information while not reducing the data in the test set, we have used the k-fold cross validation.

This divides the training subset into *K* splits, then one is chosen for the validation and the rest for training. The next step is training the model with the training set and see the performance with the validation. One done, another partition will be the validation set and the rest the training and so on. This process will be done K times. Once it finishes it will calculate the mean error.

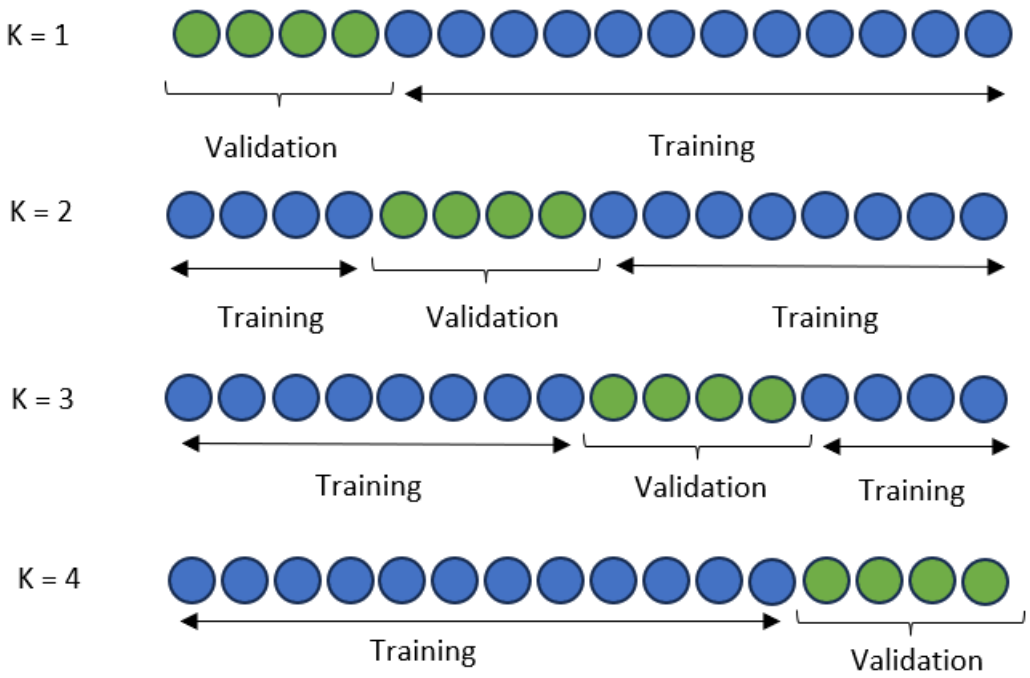

Figure 31. Example of K-folds cross Validation for K = 4.

Finally, for recommended systems with collaborative filtering is important to take note of the level of sparsity of the data. An ideal matrix is one that isn't too sparse. However, in the case of books, it is normal for user to have not read every single book available, neither rate everything that has been, so the matrix is expected to be sparse.

The usual sparsity in recommended system is up to a 99%, (Guibing, 2013). In our case, the sparsity of the matrix of **book_id, user_id** and **review_rating** is 99.13%. This is not optimal, as the result of the Alternating Least Squares algorithm, ALS, would be a matrix with most of its rows and columns with null values. To fix this, we filtered our database for the 10000 books with more ratings. This way we select those values that are more relevant, and the matrix is less sparse. We can do this as we have the identifier of each book. After filtering we obtain a level of dispersion of 97.81%.

3.3. Modelling: Collaborative Filtering and Content Based.

The main objective of a recommender system is to predict the user's preference for a set of items based, in this case, on previous ratings. The two most popular approaches are Collaborative Filtering and Content Based.

Throughout this study, both approaches were trained to see which models for each approach were more ideal for our database. In the case of collaborative filtering, the model and memory based were analysed.

3.3.1. Collaborative Filtering

Model based Recommender is the approach were the model tries to predict the user's ratings. The models apply dimensionality reduction methods to reduce the dimension of the matrix to avoid an abundance of missing values.

The algorithms studied were:

- Singular Value Decomposition, SVD: a matrix factorization technique that splits the user-item interaction matrix into a low rank matrix. It tries to learn the underlying patterns for items and users, while predicting the missing ratings. The items that are recommended are those with the highest predicted rating for the user, (Nurfadillah, 2022).
- Non-negative Matrix Factorization, NMF: is similar to the previous one but imposes the constraint that all factors should be positive. Are relevant where non-negative and interpretability are important, (Pramoditha, 2023).
- Alternating Least Squares, ALS: is an optimization algorithm that alternates between item and user factors. Used for large-scale recommender systems.

Memory Based Recommender is the technique that apply a statistical approach to the dataset to calculate the predictions. They can be divided into user-based filtering and item-based filtering. The closest items or users are calculated by using cosine similarity, Pearson correlation or mean squared difference, msd.

The algorithms studied were:

- K-Nearest Neighbours, KNN: it finds the k nearest neighbours of a user based on their ratings and recommend items that the neighbours have liked. Can be applied the other way around, search for the closest items based on user ratings and recommend an item that are highly rated by the neighbours (Pedregosa et al., 2011c).
- KNNBasic: applies KNN using msd, pearson or cosine similarity to find similar users or items and make recommendations based on their ratings (Pedregosa et al., 2011b).
- KNNwithMeans: applies KNN considering the mean of the items/users and applies msd to handle rating biases (Li, 2018).
- KNNwithZScore: normalizes the item or user ratings using z-score normalization and applies the msd, cosine or pearson correlation for the similarity (Alharbi et al., 2021).

Consider that all models were studied with an item-based approach and a user-based approach.

Moreover, many algorithms for model and memory based collaborative filtering were studied, but only the models with the lowest RMSE and MSE where further analysed with the Grid Search Cross Validation. This technique computes the accuracy of the metrics for an algorithm on various combinations of parameters over a cross-validation procedure. This is helpful for finding the best configuration of parameters. Those algorithms that shown an improvement in the results were compared and studied, shown in chapter **Results and Discussion**.

3.3.2. Content Based

Is one of the two most common techniques for a recommender system. It uses the content of the entities to find other relevant items similar to it. In other words, it searches for the keywords related to the product to find another with similar keywords to recommend to the user.

The choice is made considering concise information of the book, in this case, the title, author, description and publisher.

Moreover, as the input in this technique is the book information and not the ratings, it is needed further preprocessing such as joining the first and second name of the authors in one single string by '_', the same with the publisher, so the techniques take them as one word. Then, the description, authors, publisher is joined into a single string which column is called 'description'. This dataset is then saved.

The algorithms studied, regarding NLP techniques, where:

- TF-IDF: is a vector algorithm that measures the originality of words by comparing the number of times the word appears in a document with how much that same word appears in all documents. Transforming words into a meaningful representation of numbers, (Chaudhary, 2020).
- Doc2Vec: is a paragraph vector that represents the overall meaning or theme of a document useful for documents clustering or classification tasks where documents have to be represented as a vector, (Nayak, 2019).

Once we have the NLP techniques, we apply the following algorithms to generate clusters in order to recommend books similar to the one input by the user:

- K-means: is an unsupervised learning algorithm which partitions the data into clusters in a way the data point within the cluster is as similar as possible while differs from the other clusters. Each cluster is represented by its centroid, central point, and works through an iterative process until all data points are closer to their cluster's centroids rather than to the others, (Lopez, 2023).
- Gaussian Mixture: is a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters, (Pedregosa et al., 2011a).

Finally, in order to see if a book will be liked, we use the following techniques to predict this categorical problem:

- SVM: the Support Vector Machine objective is to find the hyperplane which distinctly classifies the data points in an N-Dimensional space (being N the number of features), (Gandhi, 2018).
- Gaussian Naïve Bayes: is a probabilistic classification algorithm that applies bayes theorem, (Vats, 2022).
- Decision Trees: is both a classification and regression algorithm, which allow us to clearly see the feature and decisions the technique applies, simple to understand, interpret and visualize, (Gupta, 2017)

For our analysis we have decided that those words that appear less than 10 times considering all documents are not relevant as they will be too rare and those that appear in more than 60% of the documents (being common tokens) are removed as we consider them not relevant. Also, English stop words are removed.

Finally, the vector that represents all the books in the database is generated and furthered compared within each vector based on the cosine similarity. Meaning that those books that have the same keywords will have a value of 1 and those who are highly different will have a value closer to 0.

3.4. EVALUATION

As we have seen in the previous subchapter, regarding Collaborative Filtering, the ratings are numerical, from 1 to 5. In order to evaluate the different approaches of the recommender system, we measure the MAE and the RMSE of each model.

In the case of MAE:

$$MAE = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

- y_i = prediction
- x_i = true value
- n = total number of data points

For RMSE:

$$RMSE = \sqrt[2]{\frac{\sum_{i=1}^{n} (y_i - x_i)}{n}}$$

- y_i = prediction
- x_i = true value
- n = total number of data points

Once the models have been evaluated, we keep those with the lowest RMSE, and we further analyse them by improving their parameters. Again, they are evaluated by the MAE and RMSE and compare the validation and test results. Trying to avoid overfitting the models.

In the other hand, regarding Content Based, the models try to classify the books into two categories: liked or not liked. Books with an average rating above 3.5 are good books, therefore liked, and those below that rating are not that liked, therefore, not liked.

So, to evaluate these models, we compare the precision, recall and f1-score. Take into account that, as we want true books classified as 'liked' we focused on the models with better precision to penalize the false positives. Accuracy is not used as a metric as the database is unbalanced and there are divided 80/20 and it would give us useful insight regarding the models, (Galli, 2023).

Regarding accuracy:

$$Accuracy = \frac{TN + TP}{TN + FP + TP + FN}$$

- TN = True Negatives
- TP = True Positives
- FP = False Positives
- FN = False Negatives

For Precision:

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

For recall:

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

For f1-score:

$$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Finally, to evaluate the clusters we use the following metrics:

Total Within-Cluster sum of squares (WCSS) which represents the sum of squared distances between each data point within a cluster and the centroid. Meaning that lower values of WCSS indicate more compact and well-defined clusters.

Total Between-Cluster Sum of Squares (BCSS) that measures the distances between centroids of all the clusters. The higher the BCSS, the better the separation between clusters.

Thirdly, the cluster consistency which measures how well the clusters are separated from one another and finally the silhouette coefficient which indicates how similar a data point is to its own cluster compared to the others.

4. RESULTS AND DISCUSSION

In this chapter it is discussed the implementation of recommender system in the following manner: Firstly, the collaborative filtering techniques and their results, being explained in detail the one with the best performance. Secondly, the implementation of the content-based with NLP.

In order to check if the models are actually learning meaningful patterns from the data, we compare the RMSE and MAE of the models with the one obtained by the Normal Predictor. This is a random algorithm that generates predictions by randomly sampling ratings from a normal distribution within the rating range. It does not take into account the item or user information and makes random predictions.

Model	MAE	RMSE	Fit time	Time
Normal Predictor	1.355038	1.081123	0.145967	0.381324

Table 2. Evaluation of the Normal Predictor model used as a baseline.

Also, to compare the different models we have a BaselineOnly model, this is to keep in mind the RMSE and MAE value we want to improve for our different techniques. As we want the forecast value to be the closest to the actual rating, we need the MAE to be lower, as a lower value indicates a better fit. Therefore, the meaning of the 1.35 MAE in the normal predictor is that there is a difference of 1.35 between the actual and predicted value on average. The same for RMSE, the lower its value, the less inaccuracy in the model and better the predictions. A 0 value would indicate that the expected and actual value matches precisely (Olumide, 2023).

Model	MAE	RMSE	Fit time	Time
BaselineOnly	0.847319	0.665300	0.244027	0.199528

Table 3. Baseline model evaluation results

4.1. COLLABORATIVE FILTERING

As shown in the Collaborative Filtering chapter, we have studied different approaches, model, and memory models. In the first place, the model techniques obtained the following results:

Model	Used Based	RMSE	MAE	Fit time	Test Time
KNNBasic	True	0.907305	0.684861	0.356751	4.240210
KNNBasic	False	0.904573	0.683260	0.321304	4.033887
KNNWithMeans	True	0.836539	0.637056	0.499262	5.039770
KNNWithMeans	False	0.835805	0.636808	0.410775	4.352333

KNNWithZScore	True	0.835600	0.632968	0.481613	5.263409
KNNWithZScore	False	0.835269	0.632591	0.415480	4.528687
KNNBaseline	True	0.831365	0.633121	0.577348	5.244984
KNNBaseline	False	0.832201	0.633435	0.615601	5.009452

Table 4. Results obtainend in the validation set for Model Techniques

The lowest RMSE is the KNNBaseline item-based model, while the best MAE is for the KNNWithZscore item-based but is really closed by the KNNBaseline item-based. Regarding further improvements, we will apply Grid Search to this model.

While the memory models obtained:

Model	RMSE	MAE	Fit time	Test Time
SVD	0.824361	0.625217	1.823370	0.479682
SVD++	0.828658	0.631144	27.751368	12.078696
NMF	0.876282	0.668436	3.065825	0.421257
CoClustering	0.884133	0.685007	3.610716	0.387943
SlopeOne	0.841342	0.636858	2.811051	6.117374

Table 5. Results for the validation set for the Memory models.

The lowest RMSE and MAE, even compared to the model algorithm, is the SVD. So, in this case, this model will be further studied with the Grid Search method.

4.1.1. Adjustment of the parameters

For the best model in memory based, SVG, we apply the Grid Search Cross Validation for the following hyperparameters:

- **'n_epochs'** specifies the number of iterations the algorithm must do over the entire training set. Each iteration updates the model's parameters based on the ratings observed. A higher number of epochs will result in a better understanding of data but can cause overfitting. By default, is 20.
- **'lr_all'** is the learning rate which the rest of parameters will be updated in each iteration. A low learning rate will slow down the model but will help the precision of the parameters. By default, is 0.005.
- **'reg_all'** is the regularization parameter, helps prevent overfitting by adding a penalty term. By default, is 0.02.
- **'n_factors'** the number of latent factors used to represent the items and user in the model. They capture patterns on the features. Increasing this value will help detect the underlying

characteristics but will increase the complexity of the model and training time. By default, is 100.

- **'biased'** is a Boolean parameter to include or not bias terms in the model. It can account for systematic biases in the ratings (users that tend to rate too low or too high compared to others). By default, is True.

After applying Grid Search, we obtained that the best score was for:

- n_factors=150
- n_epochs=35
- lr_all=0.008
- reg_all=0.1:

Model	RMSE	MAE	Fit time	Test Time
SVD Baseline	0.824361	0.625217	1.823370	0.479682
SVD_GS	0.805564	0.616461	8.596994	0.397599

Table 6. Results of SVD model with Grid Search applied.

For the best model in model based, KNNBaseline, we apply the Grid Search Cross Validation for the following hyperparameters:

- **'k'** is the number of neighbors to consider. By default, is 40.
- **'min_k'** is the minimum number of neighbors to consider. By default, is 1.
- **'sim_options'** are the similarity options which include:
    - **'name'** is the similarity metric like msd, pearson or cosine. By default, is msd.
    - **'user_based'** whether to compare item to item or user to user similarity. By default, is True.
    - **'min_support'** is the minimum number of common items or users to compute the similarity. By default, is 1.
    - **'shrinkage'** parameter used in the calculation of the similarities. Can reduce the influence of rare or common items. By default, is None.
- **'bsl_options'** is the baseline options:
    - **'method'** for computing the baseline estimation: sgd or als. By default, is als.
    - **'reg'** is the regularization parameter. By default, is 0.02.
    - **'learning_rate'** by default is 0.005.
    - **'n_epochs'** by default is 10.

The best parameters were: k=45, min_k=5, shrinkage=20, reg=0.1, method=sgd, learning_rate=0.008, n_epochs=35 and user_based=False. Those parameters that are not specified is because they take the default value.

Model	User Based	RMSE	MAE	Fit time	Test Time
KNNBaseline	False	0.831365	0.633121	0.577348	5.244984
KNNBaseline_GS	False	0.805678	0.609007	3.763212	12.08570

Table 7. Results of KNNBaseline model with Grid Search applied.

4.1.2. Comparison of the best model: train and test predictions

In this part, we are only going to compare the best models, SVG and KNNBaseline for item based to the RMSE and MAE obtained by the test set.

For SVG we obtain the following results:

Model	RMSE	MAE
SVD_GS Val	0.805564	0.616461
SVD_GS Test	0.804104	0.616259

Table 8. Comparison of the SVD results in the validation and in the test set.

While in KNNBaseline:

Model	RMSE	MAE
KNNBaseline_GS Val	0.805678	0.609007
KNNBaseline_GS Test	0.808901	0.613226

Table 9. Comparison of the KNNBaseline results in the validation and in the test set.

The approach with the lowest RMSE was the SVD algorithm. Therefore, this model was the one that predicted better the rating of the items based on the users.

4.2. CONTENT-BASED FILTERING

For this recommendation system, it is needed a different approach as the model will compute the similarities between the books in the dataset and recommend based on this. Therefore, it won't consider the user's preference or past ratings. It will recommend based on the similarity of the books content. So, even though the database is the same, the information that is used as input is not and therefore the results will be compared within the models that have used the same data.

4.2.1. Clusters Methods

In is important in this technique that the models are able to generate clusters that are consistent within and distinct from each other. Therefore, the first thing we look for is the ideal number of clusters for the model using the elbow method.

The first approach, TF-IDF, had the elbow at k=32 while the Doc2Vec approach obtained it at k=41. This method helps us decide the optimal number of clusters by seeing where the cost (or inertia) by number of clusters begin to look like an 'elbow', (Mudadla, 2023). We use kElbowVisualizer to extract the optimal number of clusters. We can see it in Figure 32 and Figure 33:

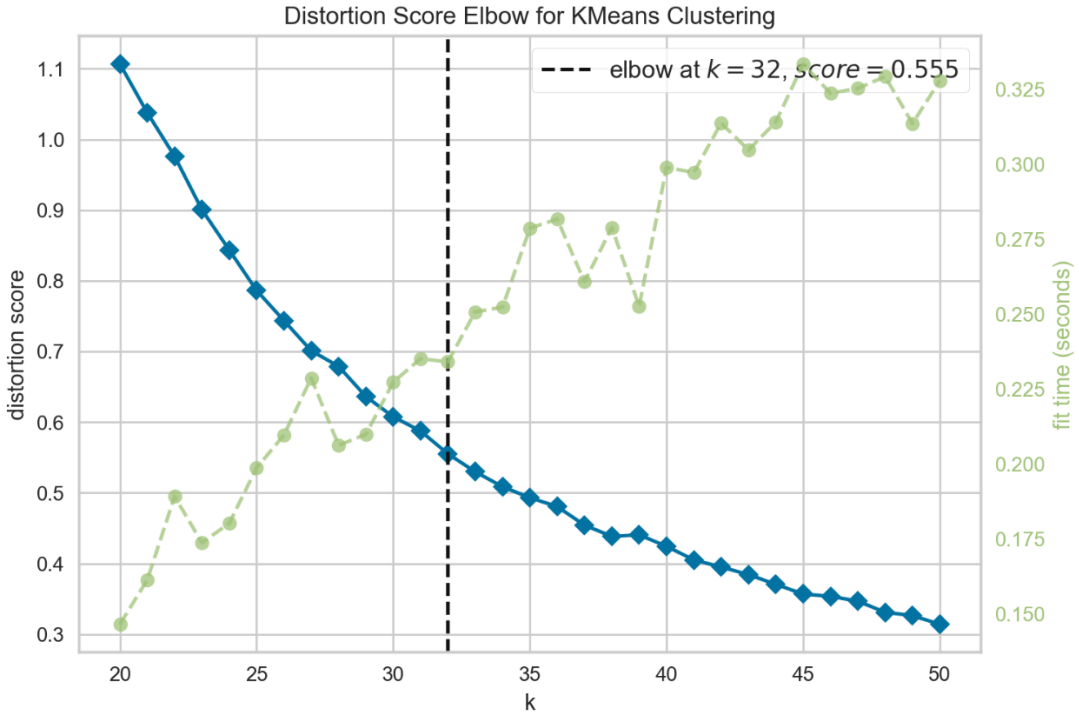

Figure 32. Distortion Score in Elbow Method for KMeans Clustering for TF-IDF

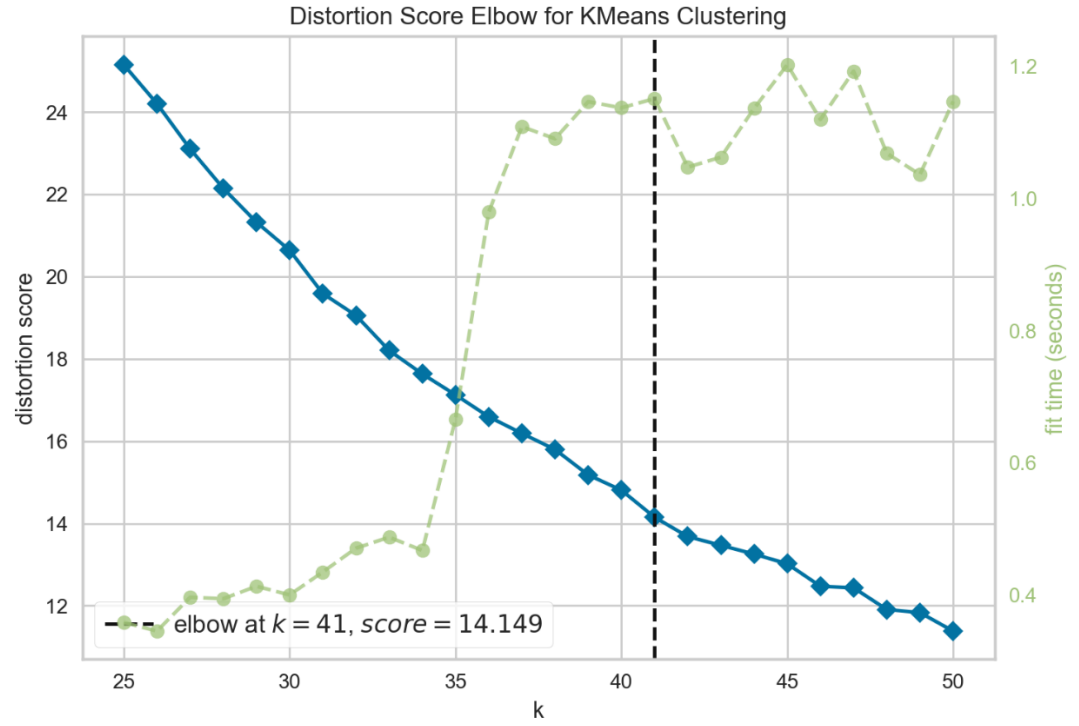

Figure 33. Distortion Score Elbow for KMeans Clustering for Doc2Vec

Next, we obtain these results when evaluating the clusters:

Model		WCSS	BCSS	Cluster Consistency	Silhouette Coefficient
K means	TFIDF	0.10457	66.51178	0.00157	0.45621
	Doc2Vec	1.87093	447.53971	0.00418	0.40327
Gaussian Mixture	TFIDF	0.57016	13.29805	0.04288	0.31754
	Doc2Vec	1.09914	435.09514	0.00253	0.39715

Table 10. Evaluation of the consistency and silhouette of the clusters in Content-Based.

4.2.2. Evaluation of the classification problem

Regarding the classification problem, being a book generally liked or not in order to recommend a similar book that is input, we obtained the following results:

Models	Class	Precision	Recall	F1-score	Macro avg
SVM + TF-IDF	0	0.00	0.00	0.00	0.45
	1	0.81	1.00	0.89	
SVM + Doc2Vec	0	0.00	0.00	0.00	0.45
	1	0.81	1.00	0.89	
Gaussian Naïve Bayes + TFIDF	0	0.25	0.48	0.33	0.54
	1	0.85	0.68	0.75	
Gaussian Naïve Bayes + Doc2Vec	0	0.19	0.30	0.23	0.49
	1	0.81	0.70	0.75	
Decision Tree + TFIDF	0	0.38	0.39	0.39	0.62
	1	0.86	0.85	0.85	
Decision Tree + Doc2Vec	0	0.70	0.10	0.17	0.54
	1	0.83	0.99	0.90	

Table 11. Validation results for Content-Based Models

Models	Class	Precision	Recall	F1-score	Macro avg
Decision Tree + TFIDF	0	0.37	0.38	0.0.38	0.61
	1	0.86	0.84	0.85	

Table 12. Test results for Content-Based best models

The best model in terms of F1-Score was the Decision Tree with TF-IDF, which has the following confusion matrix:

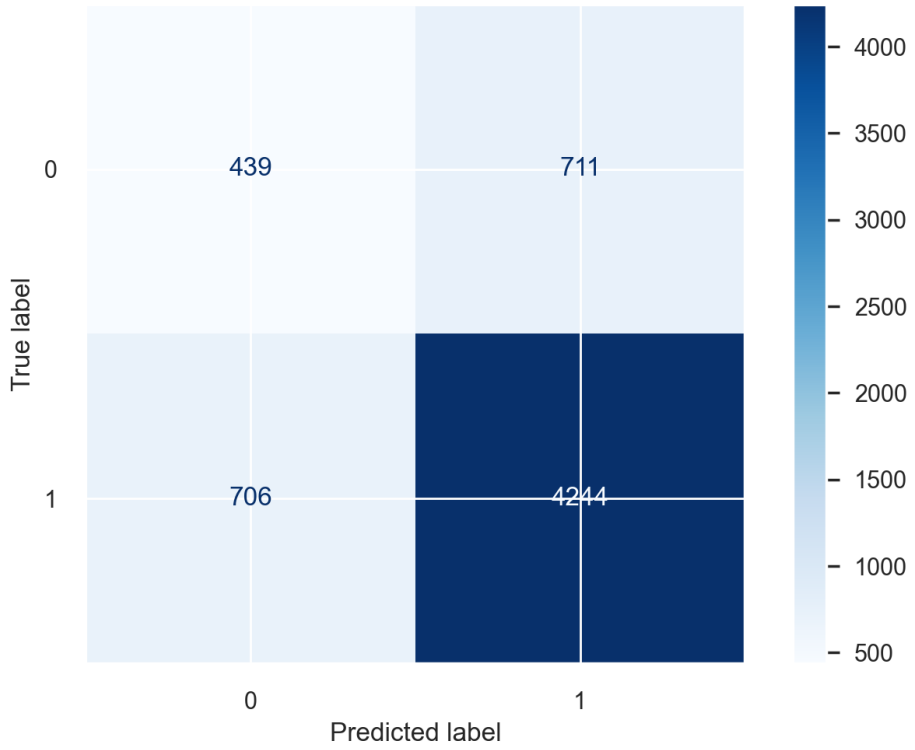

Figure 34. Confusion Matrix for Decision Tree with TF-IDF

4.3. Discussion

4.3.1. Collaborative Filtering

As shown in the results, the best models for predicting the ratings are the SVD and the KNNBaseline for this application and dataset. It was also clear that there was no overfitting as the test set RMSE was practically the same as the validation set for the SVD model and slightly higher (+0.003223) in the KNNBaseline.

Another important note is that the KNNBaseline model was quicker to fit the training but much slower processing the unseen data. Overall, the SVD was quicker to implement and give results and had slightly better performance.

However, it was noticed that the model had a hard time predicting those values that were not that common in the dataset as the 1 and 2 rating. In the other hand, it had a lower absolute error rate in those most common: 3 and 4. To improve this behavior we could implement sampling techniques to increase the data with those least common ratings. We can see this in the following graph:

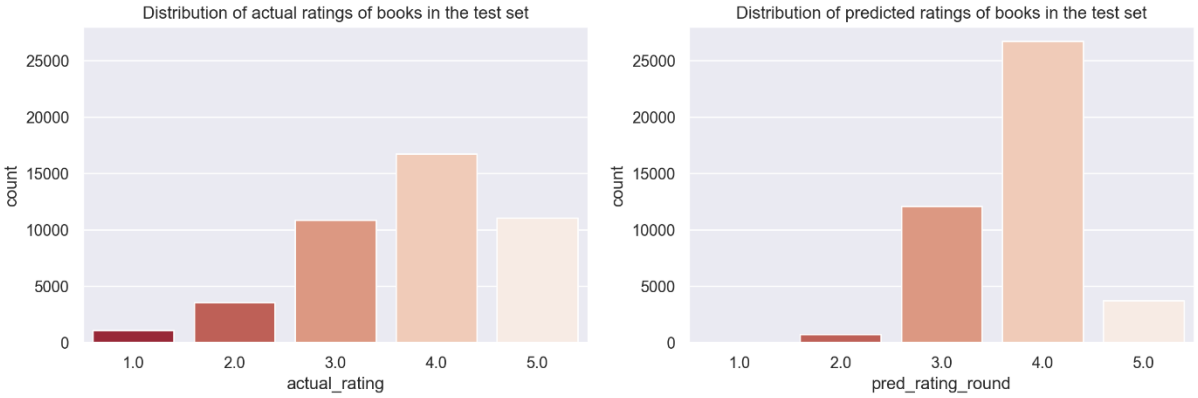

Figure 35. Distribution of actual and predictive ratings for SVD model.

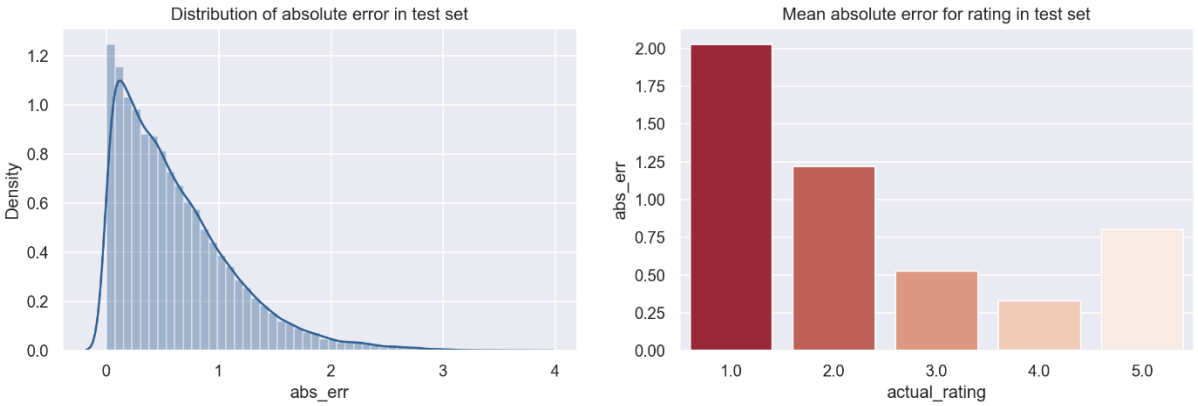

Figure 36. Distribution of absolute error in test set for SVD model.

And the same happens, although slightly better in the KNNBaseline model:

Figure 37. Distribution of actual and predictive ratings for KNNBaseline model.

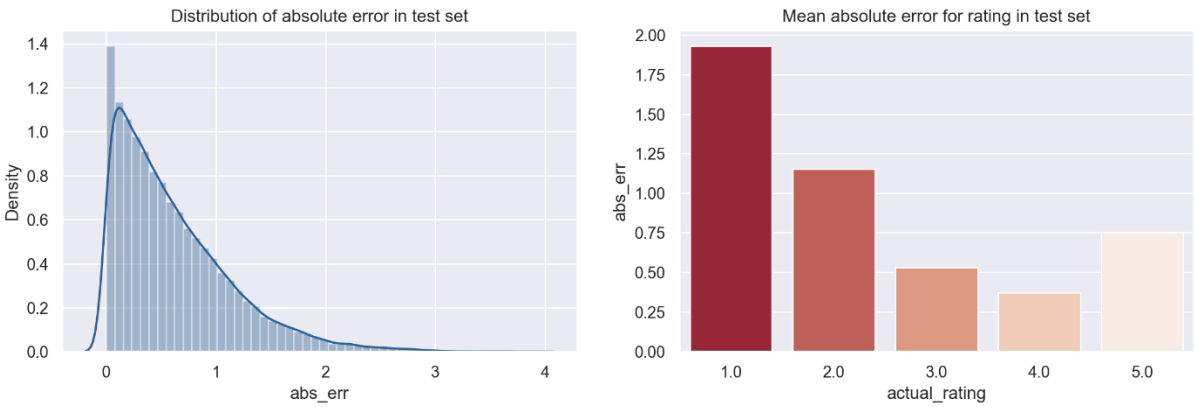

Figure 38. Distribution of absolute error in the test set for KNNBaseline.

The main problem given by the models is that those ratings that are not common are not correctly predicted. This means that the RMSE is higher as it has bigger errors in those ratings. However, the model does predict similar in the test set as in the training, so the model is capable of learning the patterns of the data and apply it correctly on new users and items.

Therefore, in future works, the model should be further developed taking into account those ratings that the model fails to predict.

4.3.2. Content-based

Firstly, based on the results shown in Table 10, we can conclude the following: both kmeans + TFIDF and gaussian mixture + TFIDF has the lower values for Within-Cluster Sum of Squares, which indicate closer, more compact clusters. While kmeans + doc2vec has the higher Between-Cluster Sum of Squares which indicate better separation between clusters. The highest model for cluster consistency which was Gaussian Mixture +tfidf approach, while the highest silhouette coefficient, meaning, better separated clusters, was kmeans + TFIDF.

Therefore, the best NLP for extracting keywords from the description input was the TF-IDF approach.

While, the best clustering approach for this specific problem is not that simple. In this case, we want the books inside each cluster to be statistically similar (theme, author, keywords and so on) to make good recommendations. Therefore, for our application it is best a consistent and homogeneous cluster, the model which suits best the content-based clustering is the Gaussian Mixture with TF-IDF.

Although, in a similar way, if we want more diverse recommendations even if they are not closely related, the model with K-means and TF-IDF will be the most suitable for it. The recommendations will be different to the expected ones, but we will have a more interesting list for the user.

We can see the clusters in Figure 39 for the K-means with TF-IDF which are far better separated and spread from one another while in Figure 40, the Gaussian Mixture with TF-IDF, the clusters are slightly overlapped.

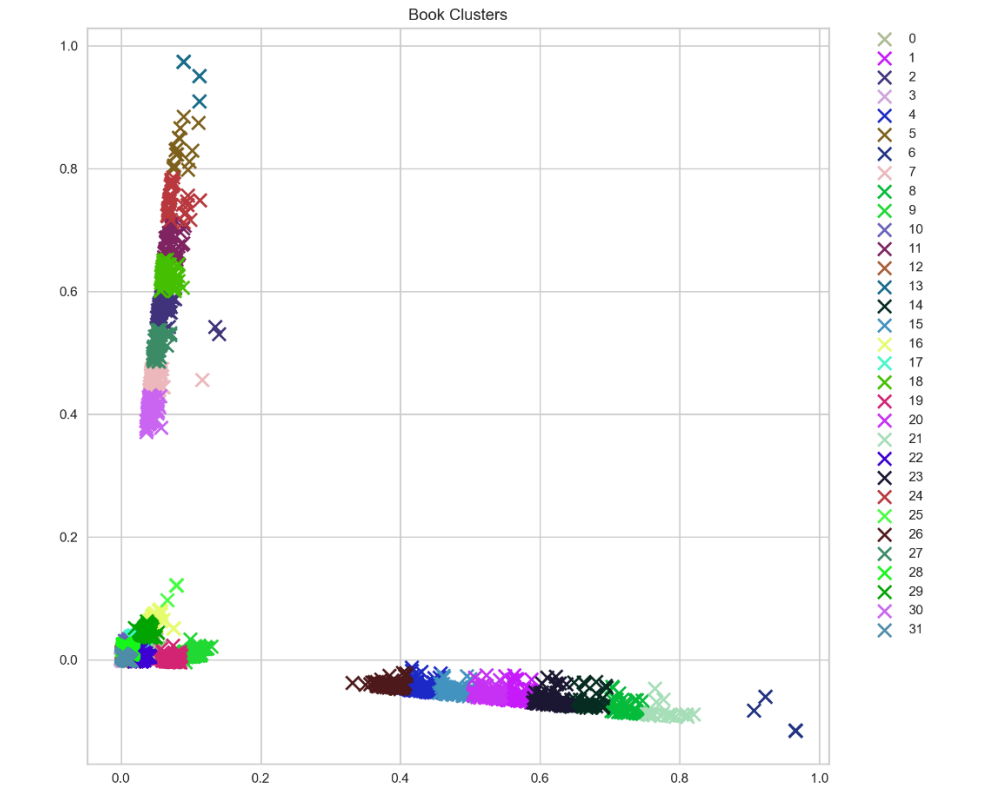

Figure 39. Cluster distribution for K-means with TFIDF

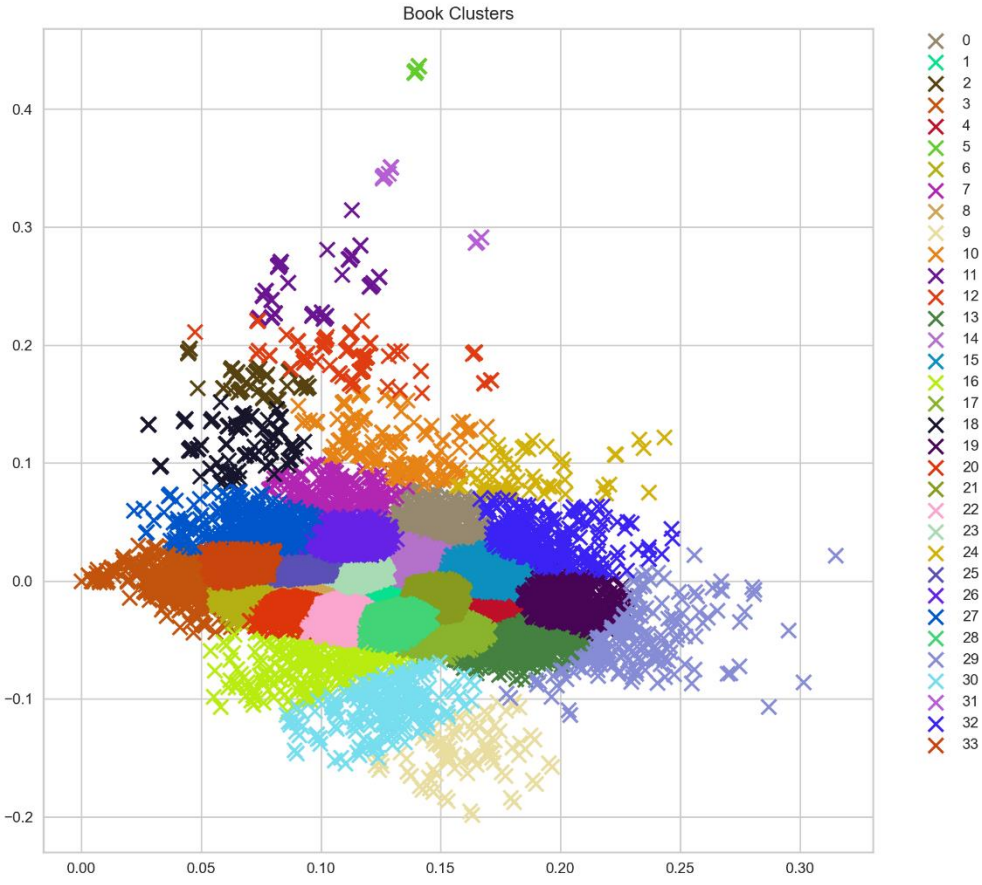

Figure 40. Cluster distribution for Gaussian Mixture with TF-IDF

Regarding the classification problem, as shown in Table 11, all the models where capable of getting most of the items of class 1 correctly, but only the decision tree was able to also classify the true negatives. Therefore, its accuracy was the highest as it also was good predicting the true positives, but the metric most useful in this case, the F1-score, was also the highest.

The model didn't show overfitting as it obtained nearly the same results in the test set.

4.3.3. Models applied to a random User

In order to further evaluate the recommender system with the model SVD, we choose a random user_id, in this case 709, and see the books that the user read for the training and compare them with the 5 top books predicted by the model and the actual 5 top books of the test set.

The previous reading of the user was mainly fantasy, science fiction and romance:

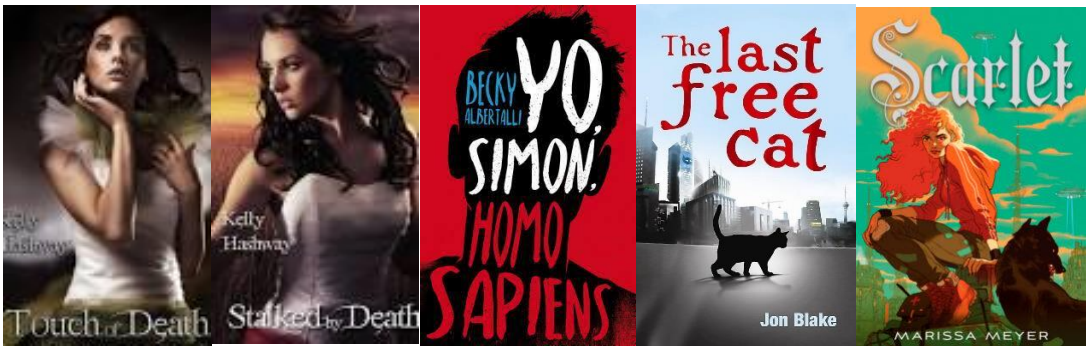

Figure 41. Training Top Books of user_id 709

While the predictive 5 recommendations books for the user are the following which are mainly fantasy with romance.

Figure 42. Predicted Top 5 Books for the user 709.

While the real top 5 books rated by the user in the test set are:

Figure 43. Actual top 5 books for the user 709 in the test set.

The two books that the model didn't predict accurately are from similar genres as the ones in the train and test: romance and fantasy. Therefore, even they are not the most rated by the user, they are relevant based on the description of the book and genre.

Moreover, we can apply the NLP model, Gaussian Mixture with TF-IDF to find which 5 books this model will recommend based on the books in Figure 41. The following recommendations are based on the recommended books for each actual top rating of the user, Figure 44. As it can be seen are books of similar genres: romance, supernatural and fantasy.

Figure 44. Recommended books based on the actual top-rated books of the user.

As we can see the content-based model gives recommendations which are relevant and similar to the input while does not recommend the same author as the collaborative filtering. Although it does not take into account if the book recommended is in a series, being able to recommend a book that is not the first in the series.

5. CONCLUSION

In this thesis project it has been developed a recommender system applied to books based on collaborative filtering and content-based models, in order to study which approach could be more interesting and effective. Its aim is to help the decision-making of users when choosing their next story to enjoy. This can be done as the RS try to personalize at maximum the services they offer to each user while being feed with enormous quantities of data. The objective is to improve the performance of the model while having better recommendations for the user.

The GoodReads database has been used to develop Collaborative Filtering models, this means recommending a book based on previous ratings of the user. Therefore, it there is a user similar to another, it will recommend books based on their similar taste. To do so we have seen that the SVG was the best approach for it, although its main problem was the computational expense. It takes longer to compute the recommendations although these ones are far better as it detects the underlying patterns of the users and items.

While the Content-based, tries to predict based on factors in the implicit data what the user is looking for and offer something similar to it. The problem with this approach is the computational time that it takes to create the word embeddings and the selection of the keywords. And, as this problem wanted to give specific recommendations to the user based on their preference, this approach failed to offer that level of personalization. Even though the recommendations that it gave were more diverse and interesting that the ones received by the Collaborative Filtering. Regarding the classification model, this helps recommending books that are overall liked by the reading community. This model could also be used with genre classification to give more precise recommendations.

However, the overall conclusion is that, with the SVD algorithm obtained a 0.8 RMSE and we were able to obtain a model capable of making useful recommendations, having acceptable results. This is undeniable important as the main objective of a recommender system is to improve the sales, in this case, of books for a publishing house, bookstore or even improve the services of a public library. As shown in the Literature review, recommender systems are useful to maximizes sales, capture new clients, improves the satisfaction of the consumers and the relationship between the company and the user. However, we should always keep in mind that the data is the one that makes a great difference between a useful or an unacceptable RS. It is needed a sufficient number of users and items to develop a recommender system and it can be difficult to obtain.

Therefore, while implementing a recommendation system, we need sufficient and accurate data and try to fix the problem of new user and items into the built model. This is because items with less ratings can be overlooked by the model and users with insufficient data can receive irrelevant recommendations.

In conclusion, several recommenders' systems have been correctly applied and evaluated, keeping the SVD model, which was the most suitable based on their performance and execution time for this specific problem and database. Also, it leaves the door open for further improvements.

6. LIMITATIONS AND RECOMMENDATIONS FOR FUTURE WORKS

As explained in the Conclusions, the objective of this thesis was to implement a recommendation system and compare which of all the different approaches could be better for book recommendations. As shown in the results chapters, the best models had issues to predict those ratings that were not that common on the database and the NLP approach had a long execution time to compute the model.

Therefore, while applying the different techniques and algorithms for the recommender system, it aroused some improvements and ideas that could have an impact on the models, obtaining a better performance. This involves an implementation of newer technologies like the use of the cloud or adding new information by crossing the database with a new one.

The proposed improvements for future works could be:

- Use a repository in the cloud like Azure of Microsoft or Amazon Web Services, AWS, so the recommender system could be store completely on the cloud.
- Implement the developed code in Spark rather than in pandas. This would improve the performance and time of the general program. This is a key point as one of the problems was the time consuming of some models (the content-based, for example).
- Apply techniques to increase the percentage of ratings with value 1, 2 and 5 so the model can learn better to predict them.
- Apply techniques so new items are also weighted and the number of ratings in an existing item does not determine the model.
- Apply Neural Networks to the database so the model not only learns about existing ratings, but by authors, publishing house and more variables.
- Apply oversampling for the minority class in Content-Based models.
- Using Hybrid techniques to for the best model in collaborative filtering and in Content-Based.
- Moreover, using a database with more recent data and with continuous updates so the input in the recommended system is updated and in real time.
- Regarding the content-based model, it could be interesting to cross the database with another that has further information of the books like its genre.
- Create a website so users can input a title and receive recommendations based on that book. This can be done by applying the model created in the content-based NLP model.

7. REFERENCES

Aleecia McDonald, & Lorrie Faith Cranor. (2012). *Beliefs and Behaviors: Internet Users' Understanding of Behavioral Advertising*. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=1989092

Alharbi, B., Assiri, F., & Alharbi, B. (2021). A Comparative Study of Student Performance Prediction using Pre-Course Data A Comparative Study of Student Performance Prediction using Pre-Course Data. *Advances in Distributed Computing and Artificial Intelligence Journal*. http://adcaij.usal.es

Alzubi, J., Nayyar, A., & Kumar, A. (2018). Machine Learning from Theory to Algorithms: An Overview. *Journal of Physics: Conference Series*, *1142*, 012012. https://doi.org/10.1088/1742-6596/1142/1/012012

Ayub, M., Ghazanfar, M. A., Maqsood, M., & Saleem, A. (2018). A Jaccard base similarity measure to improve performance of CF based recommender systems. *International Conference on Information Networking*, *2018-January*, 1–6. https://doi.org/10.1109/ICOIN.2018.8343073

Bluepi. (2016). *Demystifying Hybrid Recommender Systems and their Use Cases*. https://www.bluepiit.com/blog/demystifying-hybrid-recommender-systems-and-their-use-cases/

Bobadilla, J., Ortega, F., Hernando, A., & Alcalá, J. (2011). Improving collaborative filtering recommender system results and performance using genetic algorithms. *Knowledge-Based Systems*, *24*(8), 1310–1316. https://doi.org/10.1016/J.KNOSYS.2011.06.005

Brownlee, J. (2020, August). *Train-Test Split for Evaluating Machine Learning Algorithms*. https://machinelearningmastery.com/train-test-split-for-evaluating-machine-learning-algorithms/

Burke, R. (2002a). Hybrid recommender systems: Survey and experiments. *User Modelling and User-Adapted Interaction*, *12*(4), 331–370. https://doi.org/10.1023/A:1021240730564/METRICS

Burke, R. (2002b). Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction*, *12*(4), 331–370. https://doi.org/10.1023/A:1021240730564

Burke, R., Felfernig, A., & Göker, M. H. (2011). Recommender Systems: An Overview. *AI Magazine*, *32*(3), 13–18. https://doi.org/10.1609/AIMAG.V32I3.2361

Cedro. (2021). *La industria editorial mundial: algunas cifras*. Cedro. https://www.cedro.org/blog/articulo/blog.cedro.org/2021/09/16/industria-editorial-mueve-cifras-millonarias

Chandak, M., Girase, S., & Mukhopadhyay, D. (2015). Introducing Hybrid Technique for Optimization of Book Recommender System. *Procedia Computer Science*, *45*, 23–31. https://doi.org/10.1016/j.procs.2015.03.075

Chaudhary, M. (2020, April 24). *TF-IDF Vectorizer scikit-learn*. Medium. https://medium.com/@cmukesh8688/tf-idf-vectorizer-scikit-learn-dbc0244a911a

Cheng, W., Yin, G., Dong, Y., Dong, H., & Zhang, W. (2016). Collaborative Filtering Recommendation on Users' Interest Sequences. *PLoS ONE*, *11*(5). https://doi.org/10.1371/JOURNAL.PONE.0155739

Chiang, J. (2021, June 26). *7 Types of Hybrid Recommendation System*. https://medium.com/analytics-vidhya/7-types-of-hybrid-recommendation-system-3e4f78266ad8

Deng, C., Ji, X., Rainey, C., Zhang, J., & Lu, W. (2020). Integrating Machine Learning with Human Knowledge. *IScience*, *23*(11), 101656. https://doi.org/10.1016/J.ISCI.2020.101656

Didur, I. (2021, December 14). *Recommender systems: use cases, choosing one for your business*. Data Root Labs. https://datarootlabs.com/blog/recommender-systems-use-cases-choosing-one-for-your-business

Dwiastuti, M. (2017). *Weighted hybrid technique for recommender system*. 12050. https://doi.org/10.1088/1742-6596/930/1/012050

Efe. (2022). *El libro en España: el 86% de los títulos vende menos de 50 ejemplares al año*. Zenda. https://www.zendalibros.com/el-libro-en-espana-el-86-de-los-titulos-vende-menos-de-50-ejemplares-al-ano/

Felfernig, A., & Burke, R. (2008). Constraint-based recommender systems: Technologies and research issues. *ACM International Conference Proceeding Series*. https://doi.org/10.1145/1409540.1409544

Fleder, D. M., & Hosanagar, K. (2009). *Blockbuster Culture's Next Rise or Fall: The Impact of Blockbuster Culture's Next Rise or Fall: The Impact of Recommender Systems on Sales Diversity Recommender Systems on Sales Diversity*. https://doi.org/10.1287/mnsc.1080.0974

Galli, S. (2023, March 13). *Dealing with inbalanced Datasets in Machine Learning*. Train in Data. https://www.blog.trainindata.com/machine-learning-with-imbalanced-data/

Gandhi, R. (2018). *Support Vector Machine - Introduction to Machine Learning Algorithms*. Towards Data Science. https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

GeeksforGeeks. (2020). *ML - Content Based Recommender System*. https://www.geeksforgeeks.org/ml-content-based-recommender-system/

Goodreads Datasets. (2022). https://mengtingwan.github.io/data/goodreads.html

Guibing, G. (2013). *Improving the performance of recommender systems by alleviating the data sparsity and cold start problems*. Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence. https://dl.acm.org/doi/10.5555/2540128.2540617

Gupta, P. (2017). *Decision Trees in Machine Learning*. Towards Data Science. https://towardsdatascience.com/decision-trees-in-machine-learning-641b9c4e8052

Hotz, N. (2023, January 19). *What is CRISP DM?* Data Science Process Alliance. https://www.datascience-pm.com/crisp-dm-2/

IBM. (2023). *Real-world machine learning use cases*. https://www.ibm.com/topics/machine-learning

Isinkaye, F. O., Folajimi, Y. O., & Ojokoh, B. A. (2015). Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal*, *16*(3), 261–273. https://doi.org/10.1016/J.EIJ.2015.06.005

Jayaram, Mr. D., Prasad, Dr. G. N. R., & Gupta, I. (2022). Book recommendation system Just Read It! *IJARCCE*, *11*(6). https://doi.org/10.17148/IJARCCE.2022.11602

Karypis, G. (2001). *Evaluation of Item-Based Top- N Recommendation Algorithms* . 247. https://doi.org/10.1145/502585.502627

Konstan, J. A., & Riedl, J. (2012). Recommender systems: From algorithms to user experience. *User Modeling and User-Adapted Interaction*, *22*(1–2), 101–123. https://doi.org/10.1007/S11257-011-9112-X

Kurmashov, N., Latuta, K., & Nussipbekov, A. (2016). Online book recommendation system. *Proceedings of the 2015 12th International Conference on Electronics Computer and Computation, ICECCO 2015*. https://doi.org/10.1109/ICECCO.2015.7416895

Lam, S. K., & Riedl, J. (2004). Shilling recommender systems for fun and profit. *Thirteenth International World Wide Web Conference Proceedings, WWW2004*, 393–402. https://doi.org/10.1145/988672.988726

Lambert, J. (2014). The problem of too many books, or, publishing history. *Michigan Quarterly Review*, *53*(3). http://hdl.handle.net/2027/spo.act2080.0053.320

Li, S. (2018). *Building and Testing Recommender Systems With Surprise*. Towards Data Science. https://towardsdatascience.com/building-and-testing-recommender-systems-with-surprise-step-by-step-d4ba702ef80b

Lopez, D. (2023). *Introduction to K-Means Clustering*. Pinecone. https://www.pinecone.io/learn/k-means-clustering/

Manrique Sabogal, W. (2015, July 7). *Feria del Libro 2015: ¿Cómo sería un mundo sin libros?* El País. https://elpais.com/cultura/2015/06/07/actualidad/1433692845_240405.html

Martinez-Plumed, F., Contreras-Ochando, L., Ferri, C., Hernandez-Orallo, J., Kull, M., Lachiche, N., Ramirez-Quintana, M. J., & Flach, P. (2021). CRISP-DM Twenty Years Later: From Data Mining Processes to Data Science Trajectories. *IEEE Transactions on Knowledge and Data Engineering*, *33*(8), 3048–3061. https://doi.org/10.1109/TKDE.2019.2962680

Medina, E. L. (2023, May 25). *The 10 leading companies in artificial intelligence and machine learning in 2023*. Crónicatech. https://cronica.tech/tecnologia/software/las-empresas-lideres-en-inteligencia-artificial-y-machine-learning-en-2023/

Mudadla, S. (2023). *What is the purpose of the elbow method in K-means clustering, and how does it help determine the optimal number of clusters?* . Medium. https://medium.com/@sujathamudadla1213/what-is-the-purpose-of-the-elbow-method-in-k-means-clustering-and-how-does-it-help-determine-the-6c939dc914c3

Nayak, M. (2019). *An Intuitive Introduction to Document Vector (Doc2Vec)*. Towards AI. https://pub.towardsai.net/an-intuitive-introduction-of-document-vector-doc2vec-42c6205ca5a2

Noblit, C. (2023). How Readers Pick What to Read Next. *WrittenWord Media*.

Nurfadillah, R. (2022, January 25). *Get to know with surprise*. Medium. https://medium.com/tiket-com/get-to-know-with-surprise-2281dd227c3e

Olumide, S. (2023). *Root Mean Square Error (RMSE): What You Need To Know* . Arize AI. https://arize.com/blog-course/root-mean-square-error-rmse-what-you-need-to-know/

Orús, A. (2023). *Industria mundial del libro*. Statista. https://es.statista.com/temas/11254/industria-del-libro-en-el-mundo/

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011a). Scikit-Learn: Gaussian Mixture Models. *Journal of Machine Learning Research*, *12*, 2825–2830.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011b). Scikit-learn: k-NN inspired algorithms. *Journal of Machine Learning Research*, *12*, 2825–2830. https://surprise.readthedocs.io/en/stable/knn_inspired.html

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011c). Scikit-Learn: Nearest Neighbors. *Journal Of Machine Learning Research*. https://scikit-learn.org/stable/modules/neighbors.html

Pramoditha, R. (2023). *Non-Negative Matrix Factorization (NMF) for Dimensionality Reduction in Image Data*. https://towardsdatascience.com/non-negative-matrix-factorization-nmf-for-dimensionality-reduction-in-image-data-8450f4cae8fa

Resnick, P. (2008). Manipulation-Resistant Recommender Systems through Influence Limits. *ACM SIGecom Exchanges*, *7*.

Saiz, Y. (2012, June 13). *Los beneficios de la lectura*. La Vanguardia. https://www.lavanguardia.com/estilos-de-vida/20120613/54312096470/los-beneficios-de-la-lectura.html

Sandvig, J. J., Bhaumik, R., Ramezani, M., Burke, R., & Mobasher, B. (2008). *A Framework for the Analysis of Attacks Against Social Tagging Systems*. www.spurl.net

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. *Proceedings of the 10th International Conference on World Wide Web, WWW 2001*, 285–295. https://doi.org/10.1145/371920.372071

Speer, N. K., Reynolds, J. R., Swallow, K. M., & Zacks, J. M. (2009). Reading Stories Activates Neural Representations of Visual and Motor Experiences. *Psychological Science*, *20*(8), 989–999. https://doi.org/10.1111/j.1467-9280.2009.02397.x

Tobar, H. (2013, September 6). *People often lie about reading classic novels, survey finds*. Los Angeles Times. https://www.latimes.com/books/jacketcopy/la-et-jc-people-lie-often-about-reading-classic-novels-survey-finds-20130906-story.html

Toch, E., Wang, Y., & Cranor, L. F. (2012). Personalization and privacy: a survey of privacy risks and remedies in personalization-based systems. *User Modeling and User-Adapted Interaction*, *22*(1–2), 203–220. https://doi.org/10.1007/s11257-011-9110-z

TresB. (2023). *Qué es el FOMO, la epidemia del siglo XXI*. El Mundo. https://www.elmundo.es/como/2023/09/05/64f7083ae9cf4a9d458b459a.html

Tsuji, K., Takizawa, N., Sato, S., Ikeuchi, U., Ikeuchi, A., Yoshikane, F., & Itsumura, H. (2014). Book Recommendation Based on Library Loan Records and Bibliographic Information. *Procedia - Social and Behavioral Sciences*, *147*, 478–486. https://doi.org/10.1016/j.sbspro.2014.07.142

Tucci, L. (2023). *What Is Machine Learning and Why Is It Important?* TechTarget. https://www.techtarget.com/searchenterpriseai/definition/machine-learning-ML

Van, M., & Brynjolfsson, A. E. (1997). *Electronic Communities: Global Village or Cyberbalkans?*

Vats, R. (2022). *Gaussian Naive Bayes: What You Need to Know?* https://www.upgrad.com/blog/gaussian-naive-bayes/

Wu, H. C., Wing, R., Luk, P., & Kwok, ; K L. (2008). Interpreting TF-IDF Term Weights as Making Relevance Decisions. *ACM Transactions on Information Systems*, *26*. https://doi.org/10.1145/1361684.1361686

Xu, H., Ren, S., Yan, J., & Li, Z. (2020). Research and design of recommendation in book circle system. *Journal of Physics: Conference Series*, *1629*(1), 012063. https://doi.org/10.1088/1742-6596/1629/1/012063

Zahrawi, M., & Mohammad, A. (2021). Implementing Recommender Systems using Machine Learning and Knowledge Discovery Tools. *Knowledge-Based Engineering and Sciences*, *2*(2), 44–53. https://doi.org/10.51526/KBES.2021.2.2.44-53

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa