University of Kerbala
College of Computer Science & Information Technology
Computer Science Department

Exploiting Explicit and Implicit Feedback Information for Recommendation System Improvement

A Thesis

Submitted to the Council of the College of Computer Science and Information Technology / University of Kerbala in Partial Fulfillment of the Requirements for the Master Degree in Computer Science

Written by

Hussain Jubair Oudah Mujibej

Supervised by

Assist. Prof. Dr. Mohsin Hasan Hussein

2023 A.D. 1444 A.H.

بسم الله الرحمن الرحيم

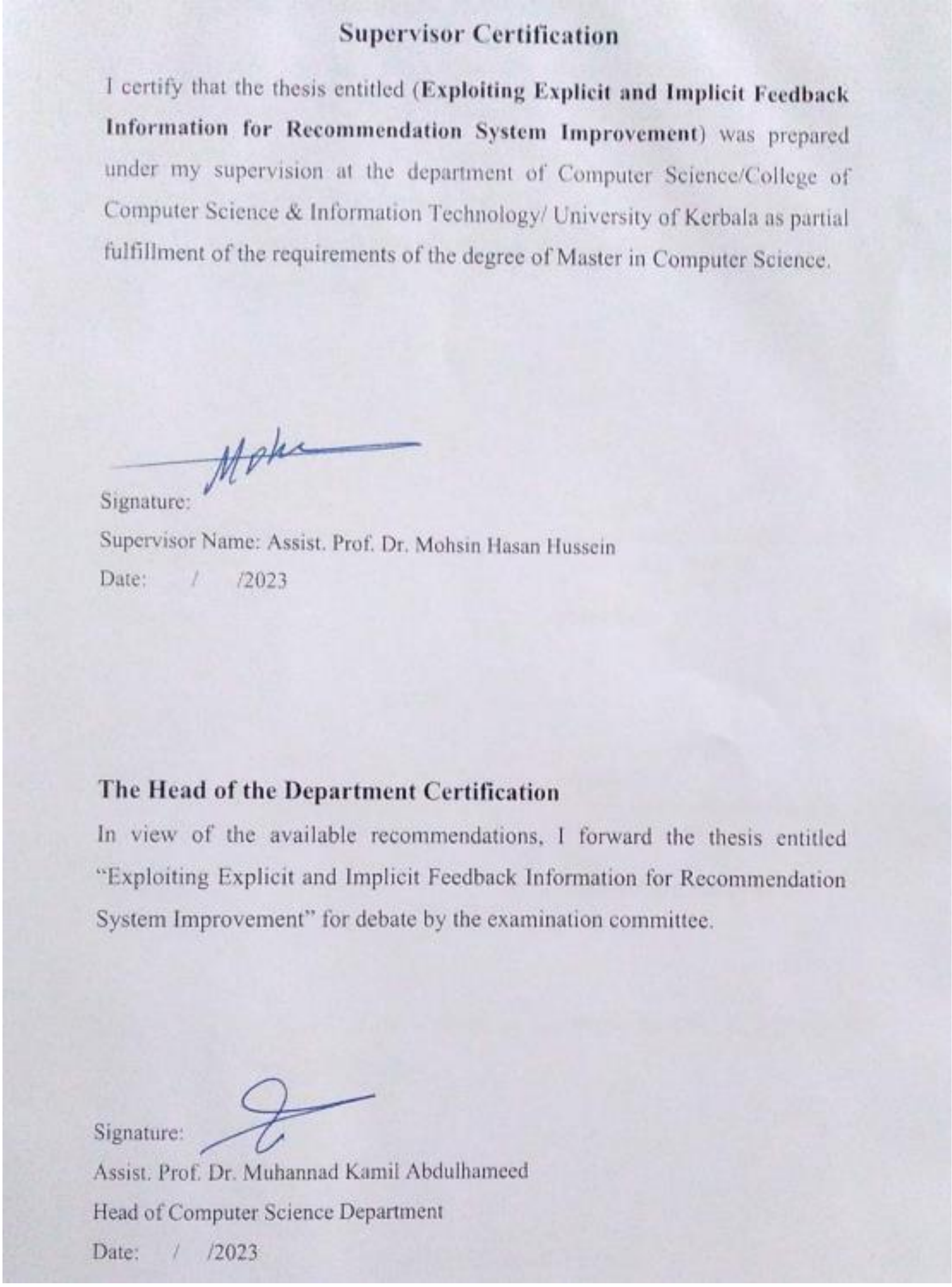
﴿ شَهِدَ اللَّهُ أَنَّهُ لَا إِلَٰهَ إِلَّا هُوَ وَالْمَلَائِكَةُ وَأُولُو الْعِلْمِ قَائِمًا بِالْقِسْطِ ۚ
لَا إِلَٰهَ إِلَّا هُوَ الْعَزِيزُ الْحَكِيمُ ﴾

صدق الله العظيم

سورة آل عمران

آية ١٨

Supervisor Certification



Supervisor Certification

I certify that the thesis entitled (**Exploiting Explicit and Implicit Feedback Information for Recommendation System Improvement**) was prepared under my supervision at the department of Computer Science/College of Computer Science & Information Technology/ University of Kerbala as partial fulfillment of the requirements of the degree of Master in Computer Science.

Signature:

Supervisor Name: Assist. Prof. Dr. Mohsin Hasan Hussein

Date: / /2023

The Head of the Department Certification

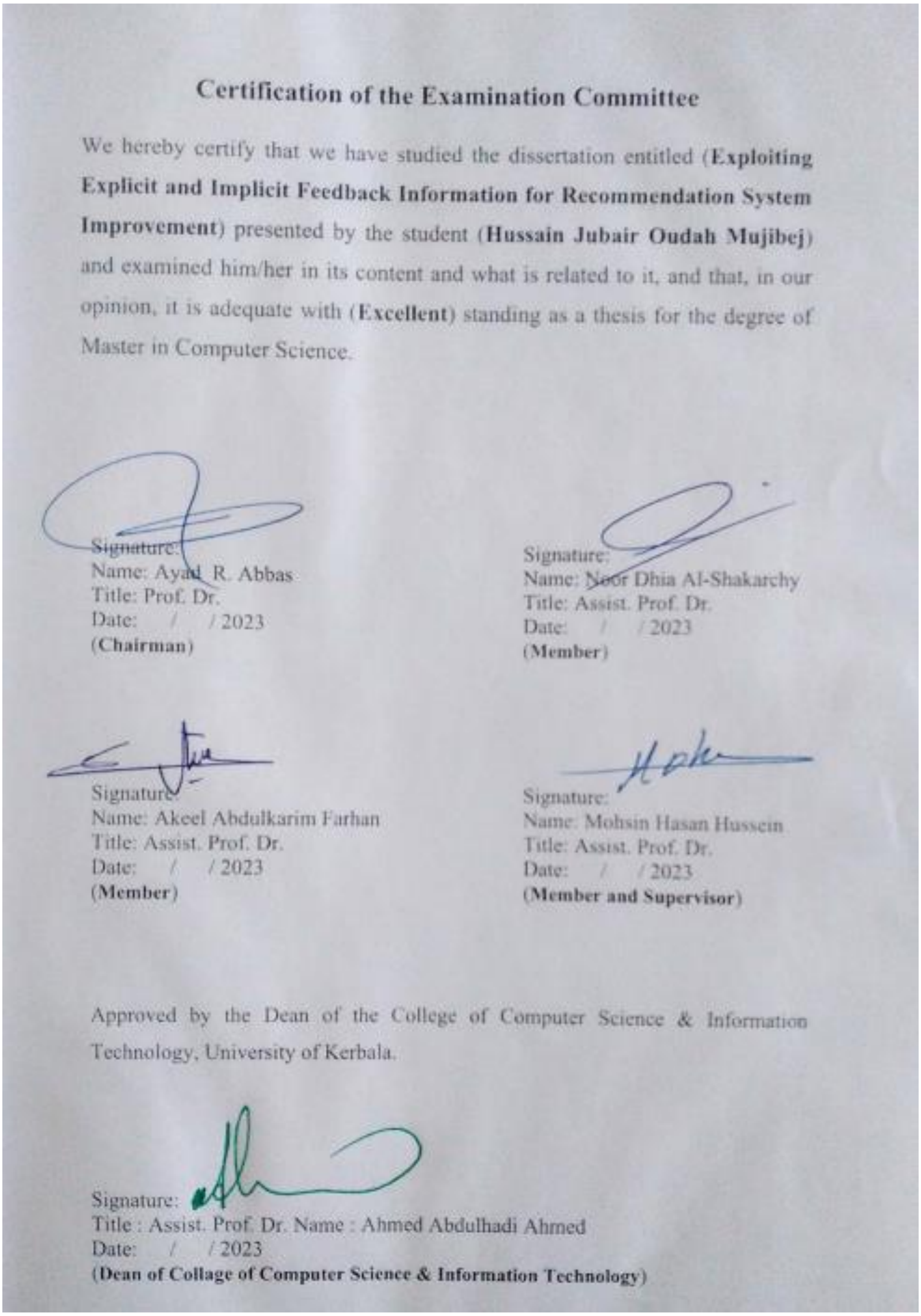
In view of the available recommendations, I forward the thesis entitled "Exploiting Explicit and Implicit Feedback Information for Recommendation System Improvement" for debate by the examination committee.

Signature:

Assist. Prof. Dr. Muhannad Kamil Abdulhameed

Head of Computer Science Department

Date: / /2023



Certification of the Examination Committee

We hereby certify that we have studied the dissertation entitled (**Exploiting Explicit and Implicit Feedback Information for Recommendation System Improvement**) presented by the student (**Hussain Jubair Oudah Mujibej**) and examined him/her in its content and what is related to it, and that, in our opinion, it is adequate with (**Excellent**) standing as a thesis for the degree of Master in Computer Science.

Signature:
Name: Ayad R. Abbas
Title: Prof. Dr.
Date: / / 2023
(Chairman)

Signature:
Name: Noor Dhia Al-Shakarchy
Title: Assist. Prof. Dr.
Date: / / 2023
(Member)

Signature:
Name: Akeel Abdulkarim Farhan
Title: Assist. Prof. Dr.
Date: / / 2023
(Member)

Signature:
Name: Mohsin Hasan Hussein
Title: Assist. Prof. Dr.
Date: / / 2023
(Member and Supervisor)

Approved by the Dean of the College of Computer Science & Information Technology, University of Kerbala.

Signature:
Title : Assist. Prof. Dr. Name : Ahmed Abdulhadi Ahmed
Date: / / 2023
(Dean of Collage of Computer Science & Information Technology)

Dedication

This work is dedicated to my mother (Allah rest her soul).

Acknowledgement

First, I would like to thank Allah for his blessings in facilitating the completion of this work in the best way.

Secondly, after Allah, I would like to thank my dear mother, who always did everything in her power and wished me to excel in my studies, may Allah have mercy on her soul.

All my sincere thanks and gratitude to my supervisor, Assist. Prof. Dr. Mohsin Hasan Hussein, for his valuable guidance and encouragement during the development of my work.

I would also like to express my thanks to all the professors and teachers who made an effort that help me to reach this stage. As well as I do not forget to thank my director and co-workers for facilitating the procedures in order to perform this study. Many thanks to all my relatives, my friends, and my partners for their assistance.

Finally, I would like to send special thanks and gratitude to all positive persons who supported and encouraged me in the most difficult times.

Abstract

Recommendation systems (RSs) are intelligent information filtering systems that deal with the information overload problem. Recently, social trust information has become a significant additional factor in obtaining high-quality recommendations. In addition, textual review information plays a core role in many RS methods that can improve the accuracy of recommendations.

RS techniques suffer from problems such as sparsity, cold-start, and class imbalance, which reduce their recommendation accuracy. In this work, two approaches have been proposed for RS improvement. They are based on implicit feedback inferences and explicit feedback ratings. The first one is the trust-based prediction model (TPM) and the other is offered with two tracks of review-based recommendation models (RRMs).

In TPM, explicit and implicit trust relations are obtained depending on the trust propagation attribute. They are incorporated to benefit from more ratings of trustworthy neighbors to alleviate the rating sparsity and cold-start problems. Accordingly, the weighted voting technique of the ensemble classifier is applied for the election of the most appropriate trusted neighbors' ratings. The K Nearest Neighbor (KNN) method is employed with a linear combination of original and trust-elected ratings to obtain the best coverage and accuracy of the prediction.

In RRMs, on the other hand, two recommendation models are developed based on the presence and absence of textual reviews in the test set. In particular, five essential steps have shaped these models: data preprocessing, text classification, topic modeling, text similarity, and the step of deducing adjustment weights to mitigate the rating sparsity and class imbalance problems. The Naïve Bayes model is used to integrate the adjustment weights as implicit feedback for recommending the preferable items to the target user.

Extensive experiments have been conducted using five real-world datasets: FilmTrust, Epinions, Musical Instruments, Automotive, and Amazon

Instant Video. The experimental results show that the proposed TPM and RRMs significantly outperform all methods compared in both rating prediction and Top-N recommendation tasks. Specifically, in TPM, the improvement ratios ranged approximately from 4% as a minimum to 20% and 10% as a maximum on FilmTrust and Epinions respectively, in terms of the F-measure of the rating prediction task. While, in RRMs, the recommendation accuracy improved by about 10% as a minimum to 61%, 26%, and 22% as a maximum on Musical Instruments, Automotive, and Amazon Instant Video respectively, in terms of the F1-measure of the Top-N recommendation task.

Declaration Associated with this Thesis

Some of the works presented in this thesis have been published or accepted as listed below.

1) The research paper entitled: “Exploiting Textual Reviews for Recommendation Systems Improvement” is presented at the 1st International Conference on Data Science and Intelligent Computing (ICDSIC 2022) held on 1-2 November 2022 and accepted to publish in IEEE Xplore digital library.

2) The research paper under the title “Exploiting Social Trust via Weighted Voting Strategy for Recommendation Systems Improvement” obtained preliminary acceptance for publication in the Karbala International Journal of Modern Science (KIJOMS) indexed in Scopus (Q2).

Table of Contents

List of Tables

List of Figures

List of Algorithms

List of Abbreviations

Abbreviation	Description
ADRS	Adaptive Deep learning-based method for Recommendation System
AIV	Amazon Instant Video
ALBERT	A Lite Bidirectional Encoder Representations from Transformers
AM	Automotive
AODR-MCNN	Aspect-based Opinion mining Deep learning-based method for Recommender system using Multichannel Convolutional Neural Network
ASCF	Aspect Sentiment Collaborative Filtering
BERT	Bidirectional Encoder Representations from Transformers
CB	Content-based Filtering
CF	Collaborative-Filtering
DistilBERT	Distil Bidirectional Encoder Representations from Transformers
EIMerge	Explicit Implicit Merge
FN	False Negative
FP	False Positive
FST	Factored user and item Similarity model with social Trust
GRU-BFGS	Gated Recurrent Unit with Broyden Fletcher Goldfarb Shanno
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol

iMAE	inverse Mean Absolute Error
IMDb	Internet Movie Database
IRAD	Information Radius
IRGNN	Item Relationship Graph Neural Networks
ITRA	Implicit Trust Recommendation Approach
JSD	Jensen Shannon divergence
KLD	Kullback Leibler Divergence
KNN	K Nearest Neighbor
LDA	Latent Dirichlet Allocation
MAE	Mean Absolute Error
MI	Musical Instruments
MF	Matrix Factorization
NBC	Naïve Bayes Classifier
NBCF	Naïve Bayes Collaborative Filtering
nDCG	Normalized Discounted Cumulative Gain
NLP	Natural Language Processing
N2VSCDNNR	Node2Vec and Spectral Clustering based on Dynamic Nearest-Neighbors Recommender system
PCC	Pearson’s Correlation Coefficient
RBP	Review-based Profile
RC	Rating Coverage
RMSE	Root Mean Square Error
RoBERTa	Robustly Optimized Bidirectional Encoder Representations from Transformers Pre-training Approach
RPCC	Reliable Pearson’s Correlation Coefficient
RRMs	Review-based Recommendation Models

RSs	Recommendation Systems
RUM	Recommender system with external User Memory networks
SA	Sentiment Analysis
TF-IDF	Term Frequency-Inverse Document Frequency
TN	True Negative
TP	True Positive
TPCF	Topic Profile Collaborative Filtering
TPM	Trust-based Prediction Model
Word2Vec	Word to Vector

1.1 General Introduction

In this section, a general introduction about recommendation systems (RSs) is presented. After that, section 1.2 describes the problem statement addressed in this thesis. In section 1.3 the aim of this thesis is offered. In addition, section 1.4 shows the work motivation and related research questions. The contributions are listed for each proposed model in section 1.5. Further, the recent studies related to this thesis are touched on. Specifically, section 1.6 list all the corresponding works in two parts. The first part depicts the trust-aware methods compared to the proposed trust-based RS. Similarly, the second part completes the review-oriented techniques that encouraged us to achieve the proposed review-based RS. Finally, thesis organization is outlined in the last section.

Every great invention has a downside. The Internet and the development of the Web paved the way for a massive amount of information to be circulated and accessible by many more people. Information overload is a term that arose with the development of information technology and the growth of commercial and service enterprises in e-commerce and social media. This issue means that many items will be provided to users, causing them may not to be able to decide a quick, comfortable decision according to their interests and requirements [1], [2]. However, generating a tremendous amount of information can be utilized in various ways [3]. Furthermore, users' choices face irrelevant ample information that prevents them from locating the information they want [4].

A recommender system (RS) is one of the intelligent information filtering systems. It is used to offer exciting items to users based on their historical behavior, saving their effort and time searching. Therefore, RS helps them deal with information overload problem [5]. RSs gained increasing attention from researchers in some disciplines, such as Machine Learning, Information Retrieval, and Human-Computer Interaction. This interest paved the way for RSs to enter industrial domains, such as e-commerce marketing and the movie industry [6].

Generally, recommendation approaches consist of three types: Content-based Filtering (CB), Collaborative Filtering (CF), and Hybrid strategies. CF is considering the most common and applicable approach in RS studies due to its simplicity and applicability to a wide range of items, not only textual data. Nevertheless, it suffers from several drawbacks, namely, sparsity of ratings and cold-start of a new user or item [7]. The CF approach needs a sufficient number of ratings from a user on an item to make an effective prediction [8]. Additionally, a concealed problem that commercial RS suffer is the natural bias toward positive ratings (also known as the class imbalance) [9], [10].

Many solutions were suggested to mitigate these problems due to their influence on recommendation accuracy. One of these solutions is a hybrid RS that combines the good features of different techniques. Another solution is to exploit users' demographic information. A more promising solution is using social information such as trust relations to enrich the sparse nature of the rating matrix [11]. Bobadilla et al. [12] mentioned that the increased progress of RS parallel with the Web made social information significantly reduce the RS problems. Further, another promising solution is utilizing users' opinions in a textual form as comments (reviews) on items that aroused their interest [13]. Besides ratings, users' textual reviews will serve as a resource replete with details that will significantly help elicit their preferences [14]. Textual reviews assist people in facilitating their decision-making process about an item or contributes to answering any questioning raised by them [15].

In general, RS research addresses two main problems (rating sparsity and cold-start). In this thesis, an concealed problem is highlighted which is almost not received the same attention by RS researchers. A class imbalance problem arises in the RS field when its task is handled as a classification problem. Although the bulk of studies tends to solve this problem by the data-level method [10], [16]. However, In this thesis, the algorithm-level method is adopted [17]. For two reasons, the first is that a simple classifier as Naïve Bayes is used; hence there is no need to preprocess the data with a data-level method according to [18]. The second reason is because that the RS data is originally scarce in the rating matrix; thus, taking a sample of it will amplify the rating sparsity problem.

Most state-of-the-art research uses one of the two RS-lifesaving solutions for making accurate recommendations. These are either social information or textual reviews as additional information. The former emphasizes the exploitation of trust relations to compensate for the lack of ratings in the rating matrix by inferring more reliable links between like-minded users. The latter, on the other hand, focuses on analyzing these knowledge-rich textual reviews to provide an adequate understanding of most users’ volatility and moods, hence linking what they write with their numerical assessments. Notably, each solution is employed in a separate approach in this thesis.

1.2 Problem Statement

There are many studies that have addressed the rating sparsity and cold-start problems by exploiting additional information such as demographic information and social information. In the context of RS, fewer studies have addressed the class imbalance problem by either data-level methods (i.e., Over-sampling and Under-sampling) or algorithm-level methods.

All the above problems are addressed in this thesis. The rating sparsity means that users have rated only a very few percentages of items as to all available items in the system's user-item rating matrix. Therefore, it became very challenging to infer the relationships (similarity) between users which will be led to inaccurate recommendations.

The cold-start means that new users of the system have few or no rating records; or new items which have few, or no ratings assigned to them yet. The CF approach needs a sufficient number of ratings from a user on an item to make an effective prediction. This will not operate with newcomers due to their few ratings in the system.

The class imbalance problem is common in Machine Learning classifiers. Such a problem is also known as the natural bias towards positive ratings in the context of RS. It influenced the successful commercial RSs when there is an uneven distribution of ratings. The reason for this problem is that most users naturally rate the items positively, causing the distribution of ratings to be uneven (i.e., the number of positive ratings outweighs negative ones). This problem will cause a decline in the accuracy of recommendation models when the RS problem is viewed as a classification problem.

1.3 The Aim of the Thesis

This thesis aims to improve the accuracy of traditional RSs through the most successful recent methods within its field. The following two tasks are accomplished:

1) Proposing a predictive model (trust-based RS) based on explicit/implicit trust relations through adopting the election idea by the weighted voting of ensemble classifiers within the framework of memory-based CF (user-based).
2) Proposing a recommender model (review-based RS) that utilizes advanced methods of textual information analysis through using a Naïve Bayes model as a model-based CF.

1.4 Motivation and Research Questions

Many works have been proposed to improve the accuracy of RSs by reducing their associated problems (i.e., sparsity, cold-start, and class imbalance). Some of these works benefited from trust network information and others from valuable information provided in textual reviews as additional implicit information besides explicit numerical ratings. It can be said that these studies are the motivation behind thinking about selecting the best functions (i.e., trust network expansion, weighted voting technique, text classification, topic modeling, text similarity), thus formulating the following research questions.

In a prediction task of a trust-based RS, it is essential to answer the following research questions:

1) Can the election idea by the weighted voting technique give better accuracy results than the commonly used weighted average

technique when used in conjunction with inferred implicit trust relations?

2) If a linear combination of mixing two prediction results of original and trust-elected ratings is formed by involving a contribution weight, would this increase the prediction coverage of predicted ratings while improving its accuracy?

In a recommendation task of a review-based RS, it is necessary to highlight the following research questions:

3) After adding review opinion polarity value to the topic's probabilities vector extracted from the exact text, does this reflect the best semantic similarity between written reviews?
4) Hence, will this combination help us obtain appropriate adjustment weights that will equilibrate the natural bias towards the positive rating (i.e., class imbalance) found in most e-commerce platforms by exploiting a Naïve Bayes-based CF model?

1.5 Contributions

The main contributions of this thesis are:

for the trust-based RS,

1) A practical model quotes from similar algorithms and is applied to mitigate the well-known problems of RSs by using an unprecedented method called election by weighted voting in parallel with exploiting the trust propagation attribute within users' relations network.
2) A comprehensive strategy applies by linearly integrating two prediction results of original and trust-elected ratings using a contribution weight for promoting both accuracy and coverage of

the prediction task. Additionally, to validate the claim that pure ratings can probably support trust-elected inferred ratings.

for the review-based RS,

3) The textual review polarity and topic probabilities of text combine to improve the computation of text similarity after only important topics had been filtered.
4) Four adjustment weights with a positive/negative influence deduce to improve the user-based Naïve Bayes CF model. Two of them are employed for modifying the target item's rating priors, while the other two are handed out to alter the target user's ratings conditional probabilities.

1.6 Related Work

Most existing works on recommendation algorithms have been arranged for solving the well-known problems of RS by using whatever available auxiliary information, such as rare demographic, abundant social relations, or rich textual reviews. Thus, much trust-based and review-based research have been presented in the literature to recognize user preferences better.

1.6.1 Trust-Aware Methods

The trust-based recommender system is a distinct shape of RSs that uses trust links to enhance performance accuracy. At first, the trust concept was utilized in social science, but today it joined computer science, specifically in the RS field. In such a domain, trust can be defined as "one's faith in others' aptness to provide valuable ratings concerning the preference of the target” [19] (F. Roy, 2020). In the following, some related trust-aware methods will be described.

G. Guo et al. in [20] (2017) suggested an item recommendation model. Its idea that the trust network was utilized to provide social relations of users, which positively influenced their tastes. The model enters the user’s direct (explicit) trust with an adjustable similarity method to make a recommendation. However, the flaw in their work was that they did not make use of indirect (implicit) trust relations.

Additionally, it has been confirmed that the trust propagation feature could be productively applied to decrease prediction deviation, thus enhancing the rating prediction results. According to this, the social trust relations and confidence evaluation of ratings have been used to return the most similar trusted neighbors, depending on the transitive property of the node (user) in the social trust graph. Then, the rating prediction results can be computed through the merged ratings of trusted neighbors [21] (G. Guo et al., 2014).

EIMerge approach was offered in [22] (H. Zhao et al., 2018). It exploited explicit and implicit trust to reduce the rating sparsity by merging the ratings of the trusted neighbors after aggregating them using Pearson’s Correlation Coefficient (PCC) as a trust similarity metric. As a result, it forms a new user profile that it uses to compute rating prediction via the classic CF function. A noticeably similar approach (ITRA) was presented in [4] (2020), where Y. Li et al. also utilize the implicit trust information of a trusted network to enhance rating prediction accuracy. The first step is to aggregate the trust neighbors for every user in the set of users using the trust expansion strategy. Next, they compute the trust similarity between the target user and every other user based on the collected candidate items, and then apply the PCC metric. Finally, a trust weighting approach has been used to boost the trust weight of the candidate users who will contribute to the final prediction phase.

The last three studies are the motivation behind implementing the proposed idea of the rating election. Since each method has advantages and disadvantages, the proposed model has benefited from the former and avoided the latter. Commonly, other similar methods have been submitted in the trust-aware RS field which are focused on exploiting the trust network information, in particular the implicit trust links, to refine the recommendation results. Although these algorithms can slightly improve the recommendation performance, they fall into some practical issues, such as low prediction accuracy and poor prediction coverage.

Most previous work relied on social information to exploit them as additional information that helps to improve the recommendation accuracy. In particular, the focus has been on the social trust relations to mitigate RS problems because it reflects the actual impression of similar thoughts.

In this thesis, the idea of the rating election by the weighted voting of ensemble classifiers [23] (Z. H. Zhou, 2012) is considered an alternative to the weighted average technique adopted in most CF methods. Where the idea is embodied as follows: the trusted neighbors are assumed the classifiers and the rating scales are the classes (labels) to be voted on to be included in the profile of the target user by the weighted voting technique. Moreover, the strength of mixing original and trust-elected ratings is showed in a rating prediction evaluation by involving a contribution weight in a linear combination process. Additionally, indirect implicit trust relations are utilized by reaching a three-step depth as the best choice to set beneficial information. PCC has also been used as the trust similarity metric [24] (Y. Mu et al., 2018), and the classic CF as the prediction function through implementing the KNN algorithm by selecting Top-K trusted neighbors [12] (J. Bobadilla et al., 2013). This memory-based CF method is conducted to perform better performance in the field of trust-based RS. The following table shows the general information of each trust-aware method.

Table 1.1: Trust-Aware Methods

Method	Problem	Task	Datasets	Year	Metrics
Merge [21]	Rating Sparsity and Cold Start	Rating Prediction	FilmTrust, Epinions, Flixster	2014	MAE, RC, F1
EIMerge [22]	Rating Sparsity and Cold Start	Rating Prediction	FilmTrust	2018	MAE, RC, F1
ITRA [4]	Rating Sparsity and Cold Start	Rating Prediction	FilmTrust, Epinions, Douban	2020	MAE, RMSE, RC
FST [19]	Rating Sparsity	Top-N Recomme ndation	FilmTrust, Epinions, Ciao	2020	Precision, Recall, F1

1.6.2 Review-Oriented Techniques

A review-based RS has the aptness to keep pace with and significantly mitigate the well-known problems of conventional RS. In particular, the main attention is on research based on building a user profile (i.e., user-based CF) because it has the advantage of improving or inferring ratings, which are in turn often missing in the user-item rating matrix.

In order to benefit from the valuable information available in textual reviews, there are many advanced textual analysis techniques can be utilized to obtain significant textual features, such as topics mentioned by the reviewers in their comments and other useful features.

The work in [25] (B. Ray et al., 2021) revealed the design of a hotel RS that helped people choose a place that suited their vacation requirements and financial affordability. The system leveraged sentimentally analyzing the textual reviews using a Bidirectional Encoder Representations from Transformers (BERT) model. However, the flaw of this model is that it is costly to implement and operates slowly on large-scale corpora.

Avoiding such additional costs, V. Sanh et al. in [26] (2019) presented a model based on the idea of distilling knowledge (i.e., training a small model called the student to imitate the behavior of a large teacher model) during the pre-training phase. They developed a DistilBERT model by making it 60% faster and 40 % smaller with 97% language comprehension capabilities than the original BERT model. DistilBERT is the improved version of the BERT model that is lighter and cheaper to pre-train on larger corpora with an approximately identical performance of textual reviews classification.

The previous two works used one of the recent advanced models in natural language processing (NLP), the BERT model, to classify the polarity of reviews written on items. The former exploited the BERT model to improve the hotel RS prediction accuracy. The latter developed the original BERT model and made it available to researchers interested in building review-based RS.

In the work of [27] (C. C. Musat et al., 2013), a topic profile CF approach was offered to estimate ratings and then rank items personalized to the target user. The idea of this approach was to build a user profile that contains the topics the user mentions in his/her comments. It then compared the profiles to determine the user’s credit for the collaboration. However, its drawback was that it did not incorporate the opinion direction in the user profile.

In RS domains of textual reviews such as hotels and restaurants, where the topics (i.e., aspects) are predefined, there will be no need to extract them. While in other areas, when dealing with the unstructured nature of textual reviews, the large volume of their vocabulary, and the diversity of reviewers in their writing style, resorting to the Latent Dirichlet Allocation (LDA) model is the most appropriate solution. This model was presented firstly as a novel technique in [28] (D. M. Blei et al., 2002).

As for the work in [29] (M. Oppermann et al., 2021), a CB recommender model was implemented to assist analysts in finding and managing visualization workbooks related to topics of their interest. The researchers tried four NLP models to measure topic-based text similarity. After conducting a user study, it was concluded that using the LDA model followed by the Jensen Shannon divergence (JSD) metric was the closest combination to align human judgment in deciding the relevance.

R. Habib and M. T. Afzal at [30] (2019) offered a paper RS that would rely on analyzing in-text citations in the logical sections of research papers (e.g., introduction, literature review, methodology). Due to the difficulty of evaluating the work using a user study because of the large dataset size, JSD was adopted as an automated evaluation method. In particular, JSD was used to automatically rank papers by calculating the distance between every two probability distribution vectors.

As mentioned in [31] (J. Wang and Y. Dong, 2020), it usually complements the LDA model with the JSD metric, especially when there is a purpose of calculating the semantic similarity of texts. This study motivated us to use both these successive functions in the proposed models.

The recent three studies have focused on leveraging the LDA model or similar output (i.e., topic probability distributions) followed by the JSD metric when there is a need to compute the similarity of textual data.

A probabilistic model was constructed in [32] (P. Valdiviezo-Diaz et al., 2019) based on the Bayes theorem. Three model types have been formed: user-based, item-based, and hybrid. The model lacks accuracy when the rating distribution is uneven (i.e., positive ratings outweigh negative ratings). Therefore, it can be developed by exploiting textual reviews and adding them as adjustment weights to equilibrate the natural bias towards the positive ratings (i.e., class imbalance) of products on successful e-commerce platforms such as Amazon.

The previous work relied on textual reviews to exploit them as additional information that helps to improve the recommendation accuracy. These text reviews have the most prominent role in guiding the decision-making process. In this thesis, the direction is to use a mixture of ideas presented in the related studies presented in this section to increase the recommendation accuracy of commercial RS. Therefore, a combination of DistilBERT, LDA, JSD, and Naïve Bayes is used as a model-based CF method to achieve optimum performance in the field of review-based RS. The following table shows the general information of each review-oriented technique.

Table 1.2: Review-Oriented Techniques

Method	Problem	Task	Datasets	Year	Metrics

TPCF [27]	Data Sparsity	Rating Prediction	TripAdvisor	2013	MAE
Section-based Bibliographic Coupling [30]	Recommending Research Papers	Item Ranking	Two Datasets collected from CiteSeer	2019	JSD, Spearman correlation coefficient
NBCF [32]	Providing Explanations	Predictions	MovieLens, FilmTrust, Yahoo, BookCrossing	2019	MAE, Precision, Recall, nDCG
Hotel Recommendation System [25]	Assist users' decision-making process	Sentiment Classification	A crawled dataset from Trip advisor	2021	Precision, Recall, F1
VizCommender [29]	The difficulty in finding relevant information	Text-Based Similarity	Tableau Public VizRepo, Tableau corporate VizRepo	2021	LDA and JSD

1.7 Thesis Organization

The thesis is divided into five chapters. Each chapter begins with a short background that underlines the key contributions and offers an impression of the chapter. The summaries of the chapters are as follows:

Chapter 2 offers the definition, concepts, and types of RS. It also highlights the social trust theory, analysis techniques of textual reviews, and Naïve Bayes CF model.

Chapter 3 dissects the methodology of the proposed system.

Chapter 4 demonstrates the evaluation results and the discussion of the proposed system.

Chapter 5 articulates conclusion statements and future trends.

2.1 Overview

This chapter explains the Recommendation System (RS) with its definition, basic concepts, and types. In addition, it sheds light on trust theory as well as its propagation property and advanced analysis techniques of textual data. Finally, an overview is made of the Naïve Bayes model and the datasets used.

In particular, section 2.2 gives an RS introduction and feedback information as additional information (social and textual data). It is followed by section 2.3 which explains common RS issues. Next, section 2.4 lists the traditional types of RS and their divisions. Section 2.5, in turn, describe similarity measures. Subsequently, section 2.6 depicts the trust-based RS and how to exploit trust propagation feature. In addition, section 2.7 focuses on the review-based RS, particularly, the advanced methods that deal with textual data and how to benefit from it. Finally, section 2.8 provides a description of the evaluation metrics that used in this thesis.

2.2 Recommendation Systems Definition and Concepts

Due to the abundance of information across Web platforms, it has become quite difficult for users to quickly get exactly what they want. This has formed a challenge related to the information overload problem for such platforms [1], [2].

The way to deal with such a problem is to resort to recommendation systems (RSs). These are software techniques that will act on behalf of users in performing the most challenging part to provide automatic recommendations of the items appropriate for them [5].

Providing such recommendations is not enough. It should be very accurate as well to the point that it will make it easier for users to quickly make decisions and choose their favorite items without being surrounded by other information useless to them.

RSs focus on how to collect feedback information from their users because of the scarcity of the ratings they provided. The recommendation technique used determines how to collect this information and how it is perfectly utilized. Usually, RSs explicitly ask users to show their preferences. This explains rating dearth due to people’s reluctance to do so. Alternatively, these systems implicitly deduce their users’ interests by observing their interaction history.

2.2.1 Explicit/Implicit Feedback Information

Generally, RS is pivotally based on constructing user profiles containing helpful information and preferences that will serve the way it works. In order to do this, it is assumed that this system has full knowledge of every component that is or will be present when it provides personalized recommendations. As mentioned earlier, the way of collecting and disseminating the preferences is determined by the applicable recommendation technique. Two general ways exist for obtaining this information in RSs. The system can access user preferences by implicitly observing their interaction behavior. However, it can provide explicit tools to allow users to express their opinions under a particular scale [33].

Since traditional RSs lack sufficient information to provide practical recommendations, these systems might exploit additional information in its explicit form or implicit inference from it. So, what is this additional information that can be used?

In general, RSs attempt to summarize the choices of their users on a pool of available items (e.g., products, movies, musical tools) in the form of useful information. So, this information exists either explicitly (i.e., any direct action taken by the user, such as rating an item) or implicitly (i.e., observing and deducing the users' actions without their direct intervention). RSs may also attempt to take advantage of demographic information (e.g., age, race, gender, occupation) if it is available. In addition, these systems might employ social media information, such as social relations (i.e., trust relations) or text reviews (i.e., items' textual comments) as information that enhances system performance. Recently, the reliance on these last two forms of information coincided with the development of Web 2.0, where users can publish content, interact with it and share it with others [3], [14], [34].

The simplicity of explicit feedback is its most important feature. Still, its drawback lies in requests to users to do a direct action requiring an effort to rate the items, which is often not attractive enough to win their attention. On the contrary, the inferred score from the users' actions of their own, such as following a user or expressing an opinion in a text comment, will represent the indirect implicit feedback. This explains that the process is managed remotely by monitoring users without their obvious involvement [35].

2.2.2 User-Item Rating Matrix

The rating matrix is the basic building block of RS in which the rating sparsity is observed. Its original ratings also evaluate the system performance. Users express their preferences by evaluating various items representing the system's information domain. In the context of RSs, this expression is called “rating”. The rating matrix is built from users’ ratings of their attracted items when the system asks them. The request for these evaluations is in several forms including numerical scales (e.g., [0.5-4.0], [1-5], [1-10]) and binary scales (e.g., like and dislike). Still others are unary assessments, as in e-commerce platforms such as “been sold” [6].

Three main components that make up the rating matrix are user, item, and rating (i.e., the user’s preference for an item). All these ratings will be collected in this matrix, which is usually sparse due to the majority’s reluctance to explicitly evaluate items. This leads to a state of non-rating, which will remain unknown (i.e., NaN value). Figure 2.1 illustrates the user-item rating matrix for four users and six items, where $U = \{u_1, \dots, u_m\}$ and $I = \{i_1, \dots, i_n\}$ represent sets of all users and all items in the system, respectively. $R_{m \times n}$ is the matrix of ratings $r_{u,i}$. Note that the blank cells are the missing ratings and the question mark (?) represents a missing rating that will be predicted in an example later.

Users	Items					
	i_1	i_2	i_3	i_4	i_5	i_6
u_1	1		3	5		2
u_2	4	2	1	3	5	4
u_3	1	4			4	
u_4	?	5				

Figure 2.1: A Rating Matrix on a 1–5 Rating Scale

When users are attracted to a particular item, they will rate it. For instance, in Amazon’s five-star rating system, 4 or 5 stars express a positive interest in a product of various categories, while 1 star or 2 stars mean the opposite. 3 stars represent a neutral opinion, but some studies add it to positive or negative feedback depending on the study notion. In the event of lacking ratings, algorithms of RSs will act to predict these blanks in the matrix and recommend items with a positive estimated rating to the target users.

2.2.3 Additional Information

In many cases, RS researchers tend to develop the traditional systems with additional information available in system-related datasets. Some works have benefited from the demographic information, especially to address cold-start issues [36]. But due to its scarcity in many applications' databases, it is rarely relied upon [37]. In contrast, social or text information is much more available and frequently used due to the recent development of Web [12], [13]. Both kinds of information have become accessible to everyone because of the facilities provided by the electronic platforms.

The last two types of additional information (i.e., social and text information) have been used in this thesis in both explicit and implicit feedback forms. In the following, each type of such information will be explained in a separate section, respectively.

2.2.3.1 Social Information

Social information is collected by Web applications (i.e., social media) that work online through their users' interaction with each other. They try to connect their users' profiles and form a network of user relationships. Social media can provide its users with capabilities such as sharing content, cooperation, and opinion expression to serve as a bridge of communication between them. Thus, it has led to an increase in social activities, which have provided the social information more like a wealth to raise the performance of traditional RSs that lack sufficient information [38].

Social platforms are the primary source of social information and the place where people influence each other's decision-making by increasing their mutual interaction. Therefore, an individual’s social relationships (e.g., friends, followers, or trusted people) will significantly affect their interests and preferences. Social correlation theories support this point. They summarized that individuals with similar tastes are likely to have a prior social relationship [38].

Social information is often implied as social relations between users of RS. These relations are of various types, including symmetrical such as friendship, or asymmetrical such as follow or trust [38]. Each person’s interests are supposed to be influenced by their trustees. Therefore, trust relations are preferred over other relations because they represent reliable links that give an actual indication of a relationship’s importance and its outcomes for an effective recommendation.

2.2.3.2 Text Information

The emergence of World Wide Web (i.e., Web 2.0) and social networks helped to produce and share content easily. Web resources and social media enabled users to express their opinions in abundance. At present, visitors can access websites, and provide textual comments by expressing their ideas about a specific product. Hence, such textual reviews can assist other readers in making a future order. Therefore, this rich information (i.e., textual reviews) helps consumers in facilitating their decision-making process [15], [25]. The availability and spread of textual information no longer have the focus, given platforms like Web forums, e-commerce sites, and social networks which have fertile land to provide such information with high flexibility. This motivates and enables users to participate and show their opinions about favorable social and marketing events [15].

User-generated reviews are material full of details about favorable items. So, the contents of these reviews can help customers and companies producing those items. For instance, after purchasing an item, the buyers can describe their experience and mention advantages and/or disadvantages to entice or warn others [39]. Numerical ratings and textual reviews are two crucial elements for users. Yet, they are more critical for e-commerce enterprises to prepare the appropriate plan to promote their products in a better way. Textual reviews analysis contributes to identify spots to be improved in the products offered by commercial enterprises [39].

Generally, textual reviews will provide a significant benefit, but will also be accompanied by difficulties. One of the difficulties it will pose is the arduous task of the consumers interested in settling on a decision by looking at each product's attributes for all its comments. Other difficulties are related to the diversity of the reviewers' opinions as each individual describes items in the way his/her realizes and how much needs them. This diversity will generate complexity in analyzing and extracting useful features from texts [25]. In addition, related systems will also face a massive amount of common text vocabulary, including those related to the target field and rare ones related to the language level of reviewers. In this regard, these systems are more likely to encounter many writing forms that are due to each individual's distinctive style from the other. Furthermore, user-generated reviews typically come unorganized, so the systems cannot interpret and benefit from them. Many of these issues will make the task difficult when addressing textual information [15], [40].

Some systems attempt to recommend the most influential reviews associated with an item to their users and let them decide. Others, however, analyze the informative textual information to infer users' preferences for the features they like about an item. The attention is concentrated on the latter in this thesis. According to [25], the continuous development of computer science, especially in data science and data mining, has provided advanced algorithms that will help overcome the related difficulties to benefit from the rich details of textual reviews.

2.3 Problems of Recommendation Systems

The rating sparsity and cold-start are the main problems associated with RSs that studies attempt to mitigate due to the difficulty of addressing them. In addition, another problem that is highlighted in this thesis is commonly known as class imbalance.

2.3.1 Rating Sparsity

The rating matrix of RS that serves as a system knowledge base is almost full of missing values because although many items are available, users rate very few of them, usually only the interesting ones to them. This made the base seem almost empty, and thus it is difficult to come up with valuable recommendations. In the context of RSs, this is known as a rating sparsity problem [7].

This problem posed a real challenge to obtain accurate recommendations. Reasonable solutions have been proposed to overcome this obstacle, from exploiting demographic information to merging some methods to get a hybrid approach that addresses it [12].

In the proposed trust-based prediction model (TPM), the rating sparsity problem is addressed by leveraging the ratings of the most trustworthy neighbors that are elected to fill the rating matrix blanks to mitigate the severity of their impact on prediction accuracy. In particular, an election method is conducted inspired by the ensemble classifiers [23], especially the type of weighted voting. The idea is embodied as follows:

the trusted neighbors are assumed the classifiers and the rating scales are the classes to be voted on to be included in the profile of the target user by the weighted voting technique. Moreover, the strength of mixing original and trust-elected ratings is showed in a rating prediction evaluation by involving a contribution weight in a linear combination process. Additionally, after computing it, indirect implicit trust is leveraged from by reaching a three-step depth as the best choice to set beneficial information. Indeed, eliciting implicit trust relations by adopting trust propagation has considerably helped reduce the rating sparsity problem. Besides, Pearson's Correlation Coefficient (PCC) has also been used as the trust similarity metric [24] and the classic memory-based CF as the prediction function through the implementation of the K-Nearest Neighbors (KNN) algorithm by selecting the top K trusted neighbors [12].

2.3.2 Cold Start

The cold-start problem constitutes one of the significant difficulties in RSs. It occurs when a new user or a new item enters the system so that they remain without adequate information for deducing an effective recommendation pattern. That is, there are two types of cold-start: one for newcomer users and the other for items newly added to the system [41].

Most treatments of this problem resorted to content related to new users or items, which was used with the rating data as a hybrid system to reduce the impact of this difficulty. Other solutions have been exploited such as demographic information or social information [12].

In the proposed trust-based prediction model (TPM), the cold-start problem is addressed by leveraging the trusted neighbors' ratings that are elected to be added in the cold-users profiles. The linear combination process of mixing original and trust-elected ratings and adopting trust propagation attribute have substantially supported to mitigate this problem by achieving a high rating coverage results.

2.3.3 Class Imbalance

Most RS research generally focuses on addressing the two above problems. However, a concealed problem is highlighted which is almost not received the same attention from RS researchers. This issue is called class imbalance, it is prevalent in the Machine Learning field and points to uneven distribution of classes in data (i.e., one majority class outweighs another minority class) [9], [10], [18]. Such a problem is conditioned, and it occurs in the RS field when its task is handled as a classification problem.

Such a problem is preferred to call as “the natural bias toward positive ratings” in the context of RS. The reason of this problem is that most users naturally rate the items positively, causing the distribution of ratings to be uneven (i.e., the number of positive ratings outweighs negative ones). For illustration, TV show audiences intrinsically resort to delivering positive ratings rather than negative ones because they may continually focus on following particular shows and not watching others [9]. In addition, human nature often rewards positive items and neglects negative ones. The issue here is a decline in the accuracy of recommendation models when the RS problem is viewed as a classification problem [10]. Furthermore, It is very likely to cause users dissatisfaction with the system because the wrong suggestion of a negative item has more impact than no suggestion of a positive item [9].

The majority of studies manage to solve this problem by the data-level method [10], [16]. It means modifying the labels of the training set by standard sampling methods (Over-sampling and Under-sampling) to decrease the data imbalance. Differently, in this thesis, the algorithm-level method is adopted [17], which plans the training process to be adjusted by inferring penalty weights to reduce bias towards the majority labels and expand the role of minority labels. Two reasons for this choice. The first is that an uncomplicated classifier such as Naïve Bayes is employed, hence according to [18], they stated no need for altering the distribution of data labels with a data-level method when a simple classifier is used. The second reason is that the RS matrix originally lacks rating data; consequently, sampling it will worsen the problem. In the proposed review-based recommendation models (RMMs), the class imbalance problem is addressed by the algorithm-level method.

2.4 Types of Recommendation Systems

In general, RSs are of three main types: content-based (CB), collaborative filtering (CF), and hybrid systems. Figure 2.2 shows the main types of RSs.

CB means recommending items depending on the properties content of items or users. CF, however, assumes that users who share similar tastes on some items will likely share the same tastes on others in the future. For instance, two users, *A* and *B*, watched and rated the same movies with identical ratings. After that, a new movie is produced that user *A* saw and enjoyed, because both users have a history of similar choices, this movie will be recommended to user *B*. Lastly, the hybrid system combines the above types (i.e., CB and CF) in various ways to benefit from the advantages and reduce the shortcomings of both [3].

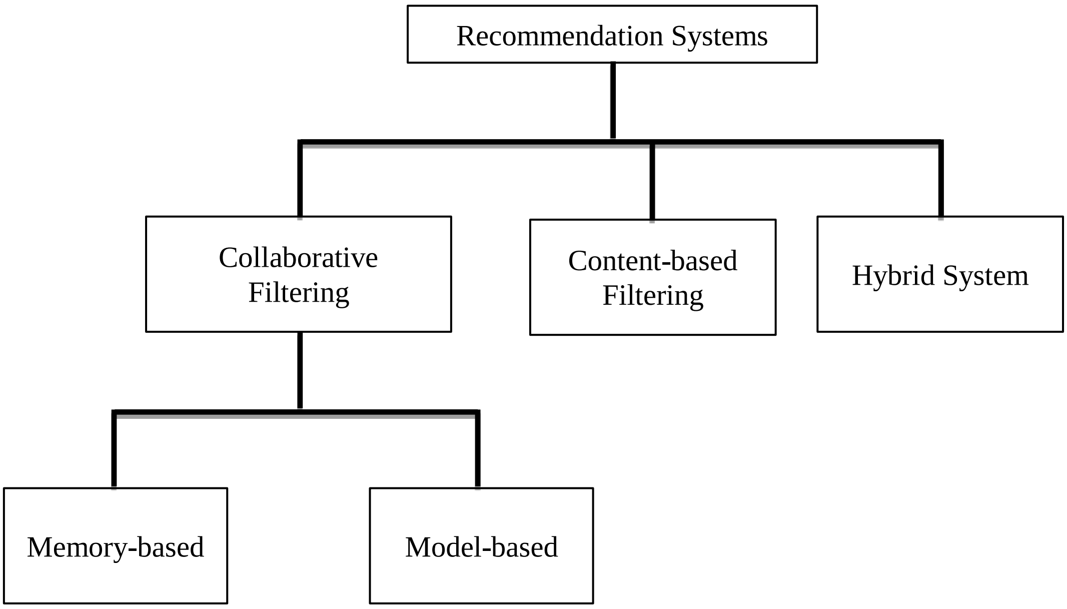


Figure 2.2: *Main Types of RSs* [11]

2.4.1 Collaborative Filtering Branches

CF techniques rely on computing the similarity between users or items or both to predict the item preferred to a target user [3]. CF is considered one of the most common and applicable approaches in RS due to its simplicity and applicability to a wide range of items, not only the textual ones as in CB. Yet, it suffers from several problems such as the sparsity of ratings and cold-start [7]. Still, the most prominent type of RSs is CF, either hybridized with the CB or stand alone.

The popularity of CF is due to its simplicity. In addition, the databases for this type are available more than other types. This led to CF wide applicability and exploitation in most previous works. After all, CF works even if there are only numerical ratings for its implementation and evaluation of its algorithms. Further, CF is subdivided into two branches: memory-based and model-based approaches [42].

2.4.1.1 Memory-based CF

Memory-based CF approaches work directly on the user-item rating matrix for predicting its blank locations and recommending the preferred items. These approaches typically need to compute users' and/or items' similarities by utilizing the primary similarity measures on the known ratings [42].

One of the most popular memory-based CF algorithms is the K-Nearest Neighbors (KNN). It relies on selecting the k elements closest to the target to complete the missing rating estimation. Three categories used in this algorithm are either focus on users (user-based CF), items (item-based CF), or both (hybrid one). The first category calculates predictions and recommendations based on the similarities of users' ratings. The second, on the other hand, is in the same pattern but on items' ratings. Finally, the third one combines the above two [12].

The basic steps of the KNN algorithm in any of the three categories above are as follows:

1) Using the appropriate similarity metric to calculate the degree of similarities between entities of the RS (i.e., users and/or items).
2) Choosing the K closest entities to the target one (i.e., selecting the neighborhood).
3) Calculating the missing value by inserting the K entities ratings into the traditional CF prediction function (i.e., weighted average technique).

After executing its steps, the KNN algorithm can recommend the positive estimated items to the target user when adopting personal recommendations as in the user-based CF [43].

2.4.1.2 Model-based CF

In order to build a RS prediction model, model-based methods operate on explicit and/or implicit information to learn the model that will improve the performance of traditional RSs [42]. In Machine Learning models, each model has a training phase on the available data. Hence, there is a test phase of the generality of these models with new validation data. The same applies to the model-based CF. Latent factors, matrix factorization, and Bayesian models are among the most successful models in recent work of RSs [12]. A standard Bayesian model is known as a Naïve Bayes classifier (NBC) which is used in the Machine Learning branch of Artificial Intelligence as a model for classifying records into one of the predefined classes.

The proposed review-based models have relied on the NBC. This classifier is selected for its simplicity in justifying and understanding the results, as well as its competitive performance against the other classifiers. NBC can be defined as "a supervised multi-class classification algorithm based on applying Bayes' theorem with the 'naïve' assumption of conditional independence between every pair of variables" [32]. The NBC can be adopted in CF in a statistical way that views its prediction task as a classification problem [44].

Thus, this classifier is employed in the work related to RSs as a model-based CF. It can compute user and/or item preferences' probability by relying on existent feedback in the system rating matrix. Given predefined ratings (i.e., classes) and an independence condition between user or item ratings (i.e., conditional probability), an NBC can estimate the unknown preference. $X = \{x_1, x_2, . . ., x_n\}$ as a set of independent variables, and $C = \{c_1, c_2, . . ., c_m\}$ as the predefined classes for which the posterior probability is computed. In this model, X will represent the users' ratings on items, and C will be each of the available ratings (i.e., 1 and 0) for binary classification. The classification score *P(C|X)* will be calculated from items' prior probability of each class *P(C)* and users' likelihood *P(X|C)* as in Eq. (2.1).

$$P(C|X) \propto \; P(C) \prod_{i=1}^{n} P(x_i|C) \tag{2.1}$$

Then, according to Eq. (2.2), the outcome that maximizes *P(C|X)* will be regarded as the output class [32].

$$\hat{y} = \underset{y}{argmax}\, P(C|X) \tag{2.2}$$

NBC computes the probability P that user *u* will rate an item *i* with a possible rating value $r_{u,i}$, given the ratings available in the rating matrix. P is calculated by the prior probability and likelihood according to Eq. (2.1).

The user-based NBC is used where prior probability and likelihood are obtained through the items rated by each user. Given *U* as a set of users, *I* as a set of items, $r_{u,i}$ as the available rating value, and *NaN* as the missing rating value. Thus, the prior probabilities distributions can be computed based on the ratings of each item, the likelihood distributions based on the ratings of a user considering the ratings of another one, and the classification score by multiplying the above two terms as in Eqs. (2.3), (2.4), and (2.5) respectively [32]:

$$P(r_i = y) = \frac{\#\{\, u \in U \mid r_{u,i} = y\} + \alpha}{\#\{\, u \in U \mid r_{u,i} \neq NaN\} + \#R\,.\,\alpha} \tag{2.3}$$

where $P(r_i = y)$ is the probability of item *i* rated as *y* by any user, α as constant parameter to avoid zero probabilities, and *#R* as the number of the available ratings (i.e., 1 and 0).

$$P\left(r_j = k \mid r_i = y\right) = \frac{\#\{\, u \in U \mid r_{u,j} = k \,\&\, r_{u,i} = y\} + \alpha}{\#\{\, u \in U \mid\ r_{u,j} \neq NaN \,\&\, r_{u,i} = y\} + \#R\ .\ \alpha} \qquad (2.4)$$

where $P\left(r_j = k \mid r_i = y\right)$ as the probability of item j being rated as k, knowing that the rating of item *i* is y. Note that the optimum value of $\alpha = 1$ according to the extensive experiments of RRMs (see section 4.2.2).

$$P\left(r_{u,i} = y\right) \propto P(r_i = y) \prod_{j \in I_u} P\left(r_j = k \mid r_i = y\right) \qquad (2.5)$$

where $P\left(r_{u,i} = y\right)$ as the classification score of the user u on item i as rating *y* according to $I_u = \left\{j \ \in I \mid r_{u,j} \neq NaN\right\}$ which is the set of items rated by u.

For each specific class label (i.e., rating), it will be assumed that the features (i.e., entities in the RS (user and item)) are independent. Hence, after calculating these features' probabilities (priors and likelihood), NBC will classify the rating of the maximum probability as the estimated class. Algorithm 2.1 illustrates the steps of the Naïve Bayes-based CF prediction model [32].

Since noisy data has no influence on the NBC probabilities' computational process, distinguished results are obtained from this Bayesian method with advanced performance.

Algorithm 2.1: Naïve Bayes-based CF Algorithm

```
Input:
    ➢ Rating Matrix $R$, Users Set $U$, Items Set $I$
    ➢ Setting the Constant Parameter $\alpha$ according to the model experiments
Output:
    ➢ Recommending Top-N highest probability items to the target user
Begin
    ➢ Preprocessing Rating Matrix $R$
    For each $u$ in $U$
      For each $i$ in $I$
        Calculate the prior probability of item $i$ according to Eq. (2.3)
        For each $j$ rated by user $u$   // $I_u = \{j \in I \mid r_{u,j} \neq NaN\}$
          Calculate the likelihood of item $j$ given item $i$ according to Eq. (2.4)
        End for $j$
       Compute the posterior probability of user $u$ on item $i$ according to Eq. (2.5)
      End for $i$
      Recommend Top-N highest probability items to user $u$
    End for $u$
End.
```

However, most rating distributions were provided toward the positive direction in the successful e-commerce platforms (e.g., Amazon), posing the problem of natural bias toward positive ratings (i.e., class imbalance) in this domain. Given the large percentage of positive ratings compared to the few negative ones, the results will often be satisfactory when using the Bayesian model regardless of uneven rating distributions. Therefore, when the RS task is handled as a classification problem, a decline in the accuracy of recommendation models ensues. This class imbalance issue is what has been addressed in the proposed review-based models.

2.5 Similarity Metrics

The idea of obtaining similarity values between RS’s entities (i.e., user and item) is almost inherent to memory-based CF methods and may also be included in the procedures of model-based CF methods.

Despite the flaw of applying the similarity idea as it consumes a lot of time and memory when dealing with large dimensions of a rating matrix. However, its effectiveness overwhelms this defect, especially when conducting the research experiments offline [12].

Among the famous metrics used to calculate the weight of rating similarity between users or items of the system is Pearson’s Correlation Coefficient (PCC) within CF. While there are also metrics of text similarity that are included within review-based RSs, the most prominent is the Jansen Shannon Divergence (JSD) [31]. These metrics will be further highlighted in the following subsections.

2.5.1 Pearson’s Correlation Coefficient

Pearson’s correlation coefficient (PCC) is one of the most outstanding similarity metrics in Machine Learning and Data Mining. Its name is often associated with memory-based CF methods. It measures the statistical correlation between two variables (i.e., two vectors of values) based on the covariance of data points. It can provide information about the direction and magnitude of the relationship [35].

Inspired by Papagelis et al. [45], who used the very common PCC as a similarity measure to identify implicitly trusted neighbors for an active user. Therefore, to calculate the degree of similarity between two users in the context of RS, the most appropriate solution is to use PCC [46], as in the following equation:

$$rsim_{u,v} = \frac{\sum_{i\in I_{u,v}}(r_{u,i}-\bar{r}_u)(r_{v,i}-\bar{r}_v)}{\sqrt{\sum_{i\in I_{u,v}}(r_{u,i}-\bar{r}_u)^2}\sqrt{\sum_{i\in I_{u,v}}(r_{v,i}-\bar{r}_v)^2}} \tag{2.6}$$

where $rsim_{u,v}$ is the degree of similarity value between user u and user v, $r_{u,i}$ is the actual rating of user u on an item i, $\bar{r}_u$ is the average of ratings belonging to user u, and $I_{u,v}$ are the co-rated items of both u and v. In order to add the asymmetric property of trust between users, the overall similarity $Rsim_{u,v}$ is computed by averaging the common item set $CIS_{u,v}$ and original rating similarity value $rsim_{u,v}$ (see section 3.3(B)). According to [47], the rating similarity threshold was determined ($\theta_{Rsim} = 0.707$) and the minimum number of co-rated items between the respective users should be greater than another threshold ($\theta_{I_{u,v}} = 2$) for inferring implicit trust relations between the target user and his/her trusted neighbors.

In addition to the values of original rating similarity $rsim_{u,v}$ and the trust between users, a third factor called social confidence is required to obtain a voting weight $VW_{u,v}$ that will be exploited in the rating election process (see section 3.3(C)). Social confidence is calculated as the ratio of similar mutual users to the target user, as in the following equation [37]:

$$\mathrm{SC_{u,v}} = \begin{cases} \frac{|TN_u \cap TN_v|}{|TN_u|}, & if\ |TN_u| > 0 \\ 0, & otherwise \end{cases} \tag{2.7}$$

where $\mathrm{SC}_{u,v}$ is social confidence between the two users u, v, and its range is in $[0, 1]$.

Since the proposed trust-based prediction model (TPM) requires validating the elected values before adding them to the elected rating matrix. Therefore, the reliability value of the elected rating $\tilde{r}_{u,j}$ of the active user’s unknown items $j \in I'_u$ can be computed as follows [32]:

$$RL_{u,j} = \frac{\max(VW^j_{u,v})}{\sum_{r \in R^s} VW^j_{u,v}},\ \forall\, v \in TN_u \tag{2.8}$$

where $RL_{u,j}$ is the reliability of the elected rating $\tilde{r}_{u,j}$ on an item $j \in I'_u$, which is in the interval $(0, 1]$, and I'_u is the set of items that are not rated by target user u. R^s is the rating scale which is [1-5] in Epinions and [0.5-4.0] in FilmTrust, TN_u are the trusted neighbors of user u in the trusted network of users, and $VW_{u,v}$ is the voting weight of v that involved in producing the elected rating of u (i.e., $\tilde{r}_{u,j}$).

It was motivated by the work [21] that used PCC metric to calculate the similarity degree between users based on their shared numerical ratings. The reliability values $RL_{u,j}$ is included in the standard PCC equation (refer Eq. (2.6)) to reduce the impact of less reliable elected ratings. So, an improved version of PCC is designated as Reliable PCC (RPCC). Thus:

$$Tsim_{u,v} = \frac{\sum_{i \in I_{u,v}} RL_{u,i}(\tilde{r}_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I_{u,v}} RL_{u,i}{}^2(\tilde{r}_{u,i} - \bar{r}_u)^2}\sqrt{\sum_{i \in I_{u,v}}(r_{v,i} - \bar{r}_v)^2}} \tag{2.9}$$

where $Tsim_{u,v} \in [-1, 1]$ denotes the trust similarity between u and v. It is important to note that $Tsim_{u,u} = 1$ due to the user u being part of his trusted neighbors TN_u. $RL_{u,i}$ is the rating reliability of the elected rating $\tilde{r}_{u,i}$ on an item $i \in I'_u$, which is in the interval $(0, 1]$. Regarding the actual ratings, their reliability will be the highest $RL_{u,j} = 1$ for all items $j \in I_u$.

2.5.2 Jansen Shannon Divergence

When talking about the similarity of text reviews, it differs from the similarity of numerical ratings. In order to calculate such degrees of similarity, first the text is preprocessed to transform it into a computable formula, and then the appropriate metric completes the job. Among the common text metrics used to compare the distributions of two texts for judging their similarity value are Kullback Leibler Divergence (KLD) and Jensen Shannon Divergence (JSD) [31].

JSD is one of the most successful metrics that work in parallel with the Latent Dirichlet Allocation (LDA) model because it calculates the distance between the probability distributions of the texts that are usually LDA outputs. It has been known by other names such as total divergence to the average or information radius (IRAD) [31]. Since this metric determines the difference, the lower its value, the more similar the two texts are. It depends mainly on KLD metric (aka, relative entropy), which is included in its mathematical formula as shown in the two equations below:

$$JSD(\vec{A},\vec{B}) = \frac{1}{2}\, KLD\left(\vec{A},\frac{\vec{A}+\vec{B}}{2}\right) + \frac{1}{2}\, KLD\left(\vec{B},\frac{\vec{A}+\vec{B}}{2}\right) \qquad (2.10)$$

where $JSD(\vec{A},\vec{B})$ is the distance value between the two text probability vectors $\vec{A}\ and\ \vec{B}$.

$$KLD(\vec{V},\vec{M}) = \sum_{i=1}^{n} \vec{V}^{i} \log\frac{\vec{V}^{i}}{M^{i}} \qquad (2.11)$$

Where $KLD(\vec{V},\vec{M})$ is the relative entropy value between one of the text probability vectors ($\vec{V}$ either $\vec{A}$ or $\vec{B}$) and $\vec{M}$ is the average vector of the two related vectors ($\vec{A}$ and $\vec{B}$). n represents the dimension of each vector. Algorithm 2.2 shows the JSD metric function [31], which has been utilized in the proposed review-based recommendation models (RRMs).

Algorithm 2.2: JSD Metric Function

```
Input:
    ➢ Two Probability Vectors $A$, $B$
Output:
    ➢ Distance Degree $DD$
Begin
    ➢ Compute the median vector $M$
      $M \longleftarrow \frac{A+B}{2}$
    ➢ Calculate the entropy of $M$ with each $A$ and $B$
      $AM \longleftarrow$ KLD ($A$,$M$)  according to Eq. (2.11)
      $BM \longleftarrow$ KLD ($B$,$M$)  according to Eq. (2.11)
    ➢ Check if the entropy values are negatives
      If $AM < 0$
        $AM \longleftarrow 0$
      End if
      If $BM < 0$
        $BM \longleftarrow 0$
      End if
    ➢ Apply the JSD equation then compute square root for JSD value
      $JSD \longleftarrow \frac{1}{2} AM + \frac{1}{2} BM$ // refer to Eq. (2.10)
      $DD \longleftarrow$ sqrt ($JSD$)
End.
```

2.6 Trust-Based RS

A trust-based RS can be defined as a conventional recommendation engine that adopts trust relations as additional information. In other words, it means reinforcing the system with explicit and implicit trust relations.

In systems with no trust information, their user associations are obtained based on only existent ratings of being the only data available. While, in trust-based RSs, trust correlations will be exploited as additional information, both explicit and implicit, along with rating information to collect more relations among the users of these systems.

The traditional steps of trust-based RSs are summed up in building the user relations graph (i.e., trust relations network) first, hence taking advantage of the ratings of network members to compensate for the missing evaluations of the target user [19]. Thus, the rating sparsity ratio is reduced in the user-item rating matrix, resulting in obtaining distinctive prediction accuracy.

A network typically consists of links connecting its members, these links describe the relationship between every two members. The types of relations most frequently utilized in RS projects are friendship, follow, and trust. Some of these relations are directly assigned by the parties, namely, friendship and follow. Trust relations, on the other hand, can indirectly be deduced from the activities of the parties concerned. Furthermore, friendship relations are symmetric (i.e., user *A* is a friend of user *B* and vice versa). However, relations such as follow and trust are not necessarily so. Trust relations are among the most effective relations applied in the context of recommendation engines. Generally, a graph is the usual means of representing and depicting a network of trust relations. An undirected graph is adopted in establishing symmetrical links such as friendship relations. Yet, asymmetric links such as trust relations are represented by a directed graph [11].

In this thesis, trust relations among users are exploited to enrich the sparsity nature of the rating matrix. For an overview, trust definition and its properties will be presented. Guo in [48] noted that, “trust is defined as one’s belief towards the ability of others in providing valuable ratings”. In [22], the authors listed the four distinct properties of trust theory as follows:

1) Asymmetry: it means that if one user trusts the other, it cannot guarantee the opposite. This property is adopted in the proposed trust-based prediction model (TPM).
2) Dynamism: the trust relation changes over time, and it is not static.

3) Context-Dependence: if user A trusts someone in buys a car, they probably might not be reliable in recommending movies to A.
4) Transitivity: this property has the notion that if user A trusts user B, and user B trusts user C, then A is more likely to trust C to some extent. It is involved in the proposed trust-based prediction model (TPM). In this model, the trust relation degree between two users (A and C) after propagating their trust is computed as inversely proportional to their shortest distance $sd_{A,C}$ (see section 3.3(B)), which is calculated using the breadth-first search algorithm [49].

The last feature, also known as the trust propagation attribute, had been used in recent research [4], [21], [22]. Its effectiveness is confirmed in mitigating the errors of missing ratings estimating. Thus, it had the credit for improving the overall prediction accuracy.

There are also two types of trust. Explicit trust refers to the trust degree between two users and takes a value of 1 or 0 if no trust exists. According to the trust propagation feature, implicit trust can be inferred and calculated from the inverse of the shortest distance between every two users connected in a network of trust relations. Since the implicit transitivity of social users has been ignored in most CF methods, many like-minded users and their similar choices are hard to obtain. This results in ineffective recommendations [4].

Furthermore, people as social members in the real world prefer to connect with others (trustworthy ones) of their trust circle links before they decide to watch a movie or buy the desired product. Therefore, the recommendations provided from trustees are more convincing than those suggested by stranger sellers. As mentioned earlier, social networks are considered a promising solution not only to connect people and establish societies, but also their information (i.e., trust relations) is critical to have better and more precise recommendations [8].

2.7 Review-Based RS

A review-based RS can be defined as a traditional recommendation engine exploiting rich knowledge extracted from the textual information utilized as additional information. That is, it uses explicit textual features in an implicit inference to enhance the system’s performance.

Traditional RSs suffer from the poor collection of their users’ hidden preferences. Thus, the overall performance will decline. In other words, conventional algorithms based on rating logs recommend without prioritizing the user's underlying preferences [50]. However, the review-based RS will address this because text reviews will tell the system why the intended user interacted with an item. Yet, ratings will not explain it. They just show whether a user likes an item or not. Therefore, this system will own the main reasons about the latent preferences of its users. Hence, it acquires the ability to detect them with greater accuracy [51].

The value of textual reviews is reflected in the fact that they are the second most important element after the numerical ratings that the consumer will look at when inquired about a specific item. As for [13], it refers to textual information-based systems’ ability to minimize CF problems. These systems have invested with the most spectacular model-based CF methods, including attribute-based MF and probabilistic MF [51]. Similarly, in this thesis, review-based recommendation models

(RRMs) are applied that depend on the Bayes theorem-based CF model, specifically the user-oriented type.

For the model-based CF methods to be able to deal with the textual reviews after they are collected, the most crucial task in text mining and Natural Language Processing (NLP) must be taken. It is the preprocessing of texts. The reason for proceeding with this step is the noisy formats arising from writing reviews (e.g., HTML tags, hyperlinks, numbers, dates, and ineffective words). All these forms should be disposed of because they will not be influential when joining the next steps, such as classifying texts or extracting topics. The unnecessary computational operations should also be avoided to save time and memory, thus making textual reviews generalized for any upcoming procedure [25]. In addition, maintaining such uninformative formats may cause problems related to accurate processing. Therefore, the preprocessing step is essential to improve the performance and quality of later analysis techniques [15].

Most research on this step focuses on the basic operations that will be mentioned next, while others include additional operations according to the type of issue and the need for computational resources. In this thesis, the common operations of text preprocessing step are:

1) Case folding: a procedure in which all words are converted to lowercase or uppercase to avoid the repetition of any word written in diverse shapes but still carries the same meaning which must be used once in the later stages. The option to convert to lowercase is the most appropriate (aka, lowercasing stage). Therefore, it has been adopted in this thesis.
2) Tokenization: a function in which the text is segmented into small pieces (i.e., terms), often called tokens. The segmentation is based

on the spaces between the words in the text. Each token will be represented as a unique variable in the subsequent stages. In this thesis, any token of a length fewer than three characters (i.e., an ineffective word) is excluded to keep off any additional useless terms in the next operations.

3) Removal of stop-words: a necessary operation to delete frequently repeated words that do not carry semantic information for the text (i.e., removing them does not affect the meaning). These are words such as ‘a’, ‘the’, ‘this’ etc., as well as conjunctions, prepositions, and pronouns. It is worth noting that all texts should be checked after this operation because the removal process may yield an empty text as some texts may contain only stop-words.
4) Stemming or Lemmatization: two operations are used alternately, and their purpose is to transform words to their original base by removing suffixes from them. Stemming differs in that it cuts off the added part of the word without considering the meaning after that (i.e., it is possible to reach unintelligible words). In contrast, lemmatization lies in the limits of returning the word to its root with keeping the explicit meaning. In this thesis, the latter is preferred to use instead of the former to preserve the semantic value of the words and reduce the vocabulary large domain.
5) Additional text preprocessing operations: these are added as needed depending on the nature of the problem. An example is the operation of deleting punctuation marks and special characters including digits, short words, HTTP web links, and HTML tags. Other operations involve combining the review title within its text,

substituting accented characters, and expanding the contractions of words (i.e., returning abbreviations to their primary words). The last operation is added as abbreviations will become meaningless after being transformed into lowercase within the case folding stage. So, they must be converted before that. All these additional operations will be further explained in section 3.4.1(A).

After implementing text preprocessing as the first step for the proposed review-based recommendation models (RRMs), the other important step is how to represent the filtered text in a digital form (i.e., converting text into its topics' probability vector). This is done for enabling the used analysis techniques to flexibly handle the text [52].

2.7.1 Common Text Analysis Techniques

To benefit from the valuable information that exists in user-written textual reviews, many advanced analysis techniques have been utilized for tasks such as text classification and topic modeling. Extracting opinions and analyzing or classifying texts are especially intended to obtain important textual features such as topics mentioned by the reviewers in their comments and other useful features [13].

Many studies specializing in text processing use data representation models to represent the text in digital form. Among the most common models are bag of words, TF-IDF, and Word2Vec. These are followed by Machine Learning classifiers, Deep Learning technologies, or pre-trained models such as Logistic Regression, Naïve Bayes, Support Vector Machine, Random Forest, or BERT, etc. to accomplish the process of analyzing and then classifying texts. This is one of the most common procedures used in RSs based on textual information [25],[43], [52]–[54].

Recently, the emergence of advanced pre-trained models in the field of word processing or NLP has led to their frequent use due to their high accuracy. These are BERT and its variants: RoBERTa [55], ALBERT, and DistilBERT [56]. In addition to the usual using of the LDA model for extracting textual features (i.e., topics), it is, according to a previous study [29], one of the closest algorithms to human judgment. In the following subsections, these models will be explained in detail.

2.7.1.1 BERT Model

The Bidirectional Encoder Representations from Transformers (BERT) model is one of the most pre-trained models of general-purpose language representation, especially in NLP tasks, that achieved the most convincing state-of-the-art performance [57]. It is of two types: the "base cased" and the "base uncased". A binary or multi-classification method can be harnessed on BERT according to the problem kind. Basically, it is used for the task of text classification or what is known as sentiment analysis (SA) for textual reviews [25].

The BERT model has been pre-trained on a concatenation of two large English corpora: Toronto Book Corpus and Wikipedia [58]. It is fine-tuned to reach the most appropriate results by opting for many important parameters including optimizer type, loss function, constant factor ratios such as learning rate, batch size, and many others. A tokenizer of BERT is also utilized to segment the textual reviews into individual small pieces called 'tokens' [25].

In particular, each token will be represented by an embedding (i.e., token embedding) that converts it to digital form. In addition, special token embeddings are inserted to separate sentences from each other.

They are placed at the sentence beginning and end. Another embedding called ‘segment embedding’ is applied to tokens to specify which segment the token belongs to. Through a transformer encoder, the location of each token in the sequence is determined by applying positional embedding. An element-wise summation is made of each token for the three embeddings (token, segment, and positional) to obtain a single representation that will act as an input. An additional classification layer is attached at the top of the transformer to deal with the output embedding. Finally, a SoftMax function will be used on the output logs of the transformer to get the probability distribution of the predefined classes.

Generally, the BERT model transformer consists of two main parts: an encoder and a decoder [25]. In specific, it contains a bundle of 12 encoders and decoders stacked over each other. Each encoder consists of two layers: self-head attention and feed-forward neural network. In the corresponding decoder, such two layers are preceded by a layer called masked self-head attention to deal with the same input sequence of the peer encoder, but in a shifted sequence. Each layer is followed by a normalization layer that will normalize its input across features. Thus, many attention mechanisms will be contained in this transformer. Since each word gives a different meaning depending on its context in the text, the role of self-attention layers is to encode each word differently according to its context.

The procedure of the BERT transformer can be summarized in two steps:

a) After feeding the summed representation of token, segment, and positional embeddings to the first encoder, it will be encoded and moved to the next encoder.

b) Then, the output of the last encoder in the stack will be transmitted to each decoder within the collection.

It is worth noting that the BERT model needs a fixed length of the input sequence. Therefore, finding the appropriate maximum length depends on word distributions in all textual reviews of the dataset. At last, a SoftMax function will be applied to the output of the last hidden layer of the BERT model to get the estimated probability distributions for the text classification task.

2.7.1.2 DistilBERT Model

Applying large-scale pre-trained models such as BERT to NLP tasks has become an essential widespread tool. The credit is attributed to the concept of transfer learning, which is intended to train models on a large data corpus, and then apply them to others in various fields [55], [57], [59]. The BERT model has become very popular due to the high performance they offer when trained on a large corpus, in addition to the fascinating ability they provide in NLP real-time applications. However, the implementation challenge lies in two things. The first is the high cost of expanding its computational (i.e., training budget) scope, whereas the second is its need for more computing and memory resources that may hinder its adoption in real-time applications [60]. So, the incentive was to create light models that are less expensive and give approximately the same performance.

The work in [26] presented a DistilBERT model which is smaller than the original BERT model. It works similarly to BERT as a general-purpose language representation model giving roughly the same performance on common tasks. The authors of [26] exploited the idea of knowledge distillation [61]. It means to train a small model (the student) to imitate as possible the exact behavior of a large model (the teacher). Furthermore, to include induction biases in their model, they combined three loss functions. They are distillation loss (teacher-student cross-entropy loss), supervised training loss (BERT classic loss), and cosine embedding loss (teacher-student cosine distance loss). Therefore, they produced a 60% faster, 40% smaller model with 97% language comprehension potentialities. So, their model is cheaper and requires fewer resources to pre-train on a larger corpus.

DistilBERT is trained on the same English corpus (Wikipedia and Toronto Book Corpus). It has the same general architecture as the BERT model, but the emphasis was on reducing the number of attention layers to half through copying a layer and ignoring the next for the purpose of the initialization process as shown in Figure 2.3. These attention layers first introduced in [62] represent the core of learning the BERT model. In addition to classifying IMDb movies, the DistilBERT model was conducted for other tasks, such as question answering on mobile devices to further study the model performance [26].

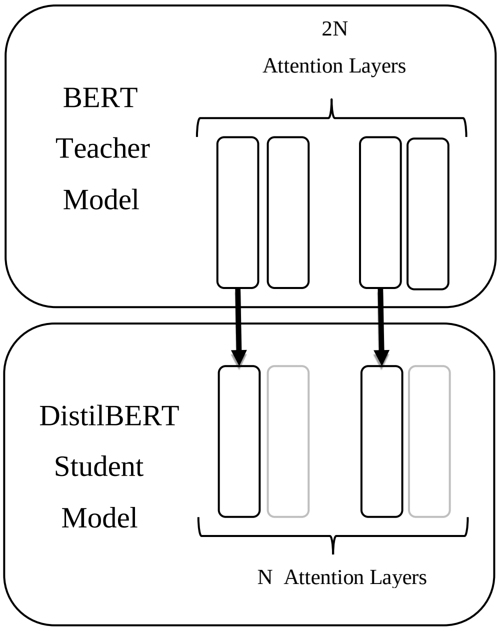


Figure 2.3: DistilBERT Initialization Process[1]

The remainder of the knowledge distillation operation is obvious. The DistilBERT model trains just like the BERT model as previously stated in section 2.7.1.1. The only difference is that two models (teacher and student) are operated simultaneously. Fortunately, the teacher (i.e., BERT) layers do not require updating since backpropagation loss is only passed to the student (i.e., DistilBERT) layers. Still, the combination of triple loss functions necessitates its implementation.

Finally, the authors employed the General Language Understanding Evaluation benchmark to evaluate the generalization and natural language understanding abilities of DistilBERT on 9 data sets related to natural language comprehension systems [63]. Therefore, the DistilBERT model is the preferred use for the text classification step of the proposed review-based recommendation models (RRMs) due to its high performance compared to the other BERT model types [56].

[1] https://towardsdatascience.com/distillation-of-bert-like-models-the-theory-32e19a02641f

2.7.1.3 LDA Model

Latent Dirichlet Allocation (LDA) is a three-level hierarchical probabilistic random model for processing a text corpus. In other words, it assigns latent topics to text documents based on the Dirichlet distributions of their words. Three components are exploited within the random generation process of this model. They are documents (i.e., textual reviews), their constituent words, and the topics to be extracted. The LDA's basic notion is that each text document will be represented by a mixture of distributions of K latent topics. These topics, in turn, will be represented by a set of word distributions. LDA will be trained on the words of all documents to calculate each K topic's probability for each document [28].

Since LDA model assumes that there are a specific number of latent topics within each text document, the words of this text will contribute to determine the ratio of each topic within the document. That means each document will have a probability distribution of all its topics. In turn, each topic will have a probability distribution of all the contributing words [31]. Due to the random assignment of words to each topic at the time of the topic modeling initiation, the same word may appear in different documents under different topics.

The general steps of the LDA model [64] are portrayed in Figure 2.4, and as follows:

1) Before the initial step, a number of *K* topics should be assumed for all documents. Practically, it will be necessary to experiment with several tests to reach the best number of *K* topics.
2) The initial step involves looping through *M* documents, and then randomly assigning each *W* word in each document to one of the *K*

topics. This step is performed only once. It might be that the same word will be assigned to different topics in different documents.

3) For each *D* document, iterate on each *W* word to calculate the local probability $P(T_k \mid D_i)$ and global probability $P(W_j \mid T_k)$. The local probability is the ratio of words in the current document assigned to each topic. The global probability, however, is the ratio of documents in which the target word was assigned to each topic.
4) Calculating the assignment probability $P(W_j \mid T_k , D_i)$ (i.e., attributing *W* to T_k) by multiplying the local and global probabilities yielded by the previous step.
5) One more iterating on each *W* word within each *D* document to re-assign this word to new topic according to the highest assignment probability $P(W_j \mid T_k , D_i)$ calculated in the previous step.
6) Checking the number of pre-determined iterations. If it is not reached, the process will be repeated back from step 3. Otherwise, the process will be terminated.

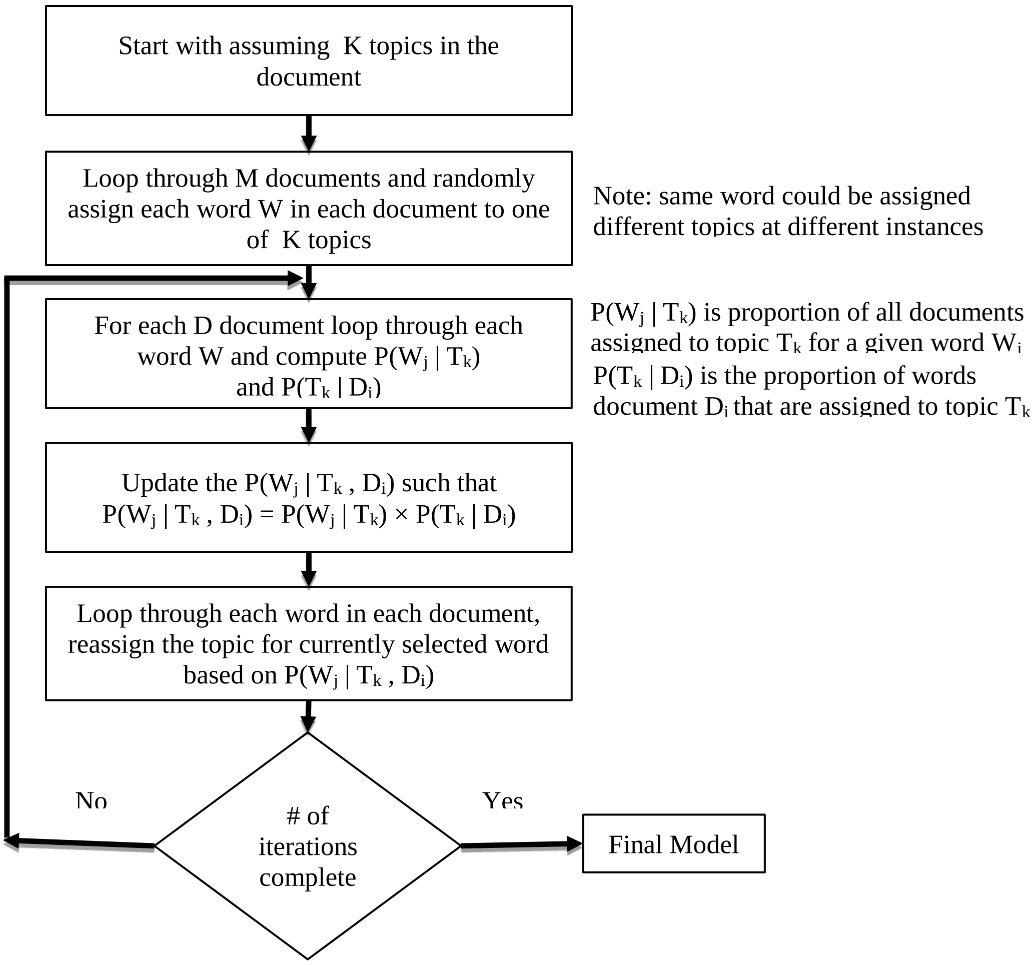

Start with assuming K topics in the document
Loop through M documents and randomly assign each word W in each document to one of K topics
Note: same word could be assigned different topics at different instances
For each D document loop through each word W and compute $P(W_j \mid T_k)$ and $P(T_k \mid D_i)$
$P(W_j \mid T_k)$ is proportion of all documents assigned to topic T_k for a given word W_i
$P(T_k \mid D_i)$ is the proportion of words document D_i that are assigned to topic T_k
Update the $P(W_j \mid T_k , D_i)$ such that $P(W_j \mid T_k , D_i) = P(W_j \mid T_k) \times P(T_k \mid D_i)$
Loop through each word in each document, reassign the topic for currently selected word based on $P(W_j \mid T_k , D_i)$
# of iterations complete
No
Yes
Final Model

Figure 2.4: LDA General Steps[2]

In order to extract features from texts, discover underlying ones, or look for relationships between text documents, topic modeling is one of the most dominant methods in data mining. In particular, topic modeling can be defined as a model that can automatically detect and extract topics based on the words of the textual document. Topics can be visualized as keywords that will describe the entire document. Topic modeling is one of the unsupervised learning methods that several standard techniques were used to implement. These include Latent Semantic Analysis, Probabilistic Latent Semantic Analysis, Correlated Topic Model, and Latent Dirichlet Allocation (LDA). The last is the most

[2] https://medium.com/analytics-vidhya/topic-modelling-using-lda-aa11ec9bec13

common one. In addition, multiple disciplines benefit from LDA-based topic modeling. These are political science, software engineering, medical/biomedical science, crime science, and linguistic science [65]. However, LDA is not only a favorite for a topic modeling task, but also often exploited in conjunction with the JSD metric (refer to section 2.5.2) to achieve semantic text similarity tasks [29], [31].

2.8 Evaluation Performance of RSs

Acquiring recommendations of excellent quality is the ultimate objective of any RS. Measuring the success of RS algorithms requires a crucial procedure, namely, performance evaluation to compare the proposed system with other standard systems. Most research tends to test the performance of its methods through offline evaluation. It avoids online evaluation and crowdsourced user study because of the high cost and the limitation of asking a group of people (e.g., experts) for an opinion on large data [6], [30]. Due to the regular cost that researchers can afford, offline evaluation is almost essential for any work related to RS [66].

Generally, RS accuracy is evaluated through new data (i.e., test data) after training the system model on the data available to create an objective function. Before that, the dataset will be divided into a specified number of N folds (usually N = 5), and then the evaluation is carried out according to the N test sets generated (i.e., cross-validation process) [15]. In other words, the entire dataset is split into N equal parts. At each time, one fold of each N subset is taken as a test set, while the remaining parts (N-1) as a training set. This procedure is run N times, in which one different fold is evaluated each time. In the end, the whole performance of the system is estimated by taking the average results of all runs. Thus, the evaluation will involve all data and reduce the divergence in evaluation results for each fold of the test data.

Two types of tasks are performed within the domain of RS research [67]: the rating prediction task and the item recommendation task (aka, Top-N Recommendation). The first task is concerned with estimating the values of empty cells within the rating matrix to reach the lowest possible error. The second task, however, seeks to find and recommend the most favorable items to the active user. Each task has different evaluation metrics for testing the proposed system's performance. In the following subsections, the most common metrics for each of the RSs' tasks will be explained separately.

2.8.1 Rating Prediction Task Metrics

There are two standard metrics for the rating prediction task: Mean Absolute Error (MAE) and Root Mean Square Error (RMSE).

MAE is used to check the system's predictability and how much error it causes (i.e., the error represents the inverse value of the system's accuracy). More precisely, this metric computes the average absolute variance between the predicted ratings and their peers of actual ratings of the test set, as shown in the equation below [22]:

$$MAE = \frac{\sum_u \sum_i |\hat{r}_{u,i} - r_{u,i}|}{M} \tag{2.12}$$

where $r_{u,i}$ and $\hat{r}_{u,i}$ are the actual and estimated ratings, respectively. M represents the number of all predictable ratings from the test set.

RMSE differs from MAE in that it calculates the square of the error between the expected value and its actual counterpart, and then computes the square root of the output. The RMSE formula is illustrated by the following equation [4]:

$$RMSE = \sqrt{\frac{\sum_u \sum_i (\hat{r}_{u,i} - r_{u,i})^2}{M}} \tag{2.13}$$

both metrics impact the prediction accuracy, as they calculate the error values, the lower the values, the better the prediction results.

Another important metric is called Rating Coverage (RC). It gives the ratio of test ratings covered in the prediction process. In other words, it measures the proportion of the test ratings that are predictable to all available test ratings [22]. Thus:

$$RC = \frac{M}{N} \tag{2.14}$$

where M is the number of all predictable ratings, N is the number of all test ratings. Many works come with distinct accuracy, but poor coverage. So, the RC metric is essential to indicate the actual prediction performance.

Since prediction coverage and accuracy (i.e., the inverse of error) are two important metrics to measure the overall productivity of a system, they can be represented together through an aggregate measure. According to [68], F-measure (aka, F1) is a metric that considers both coverage and accuracy of prediction to measure the overall performance in a balanced manner. It is calculated as follows:

$$F1 = \frac{2 \, . \, iMAE \, . \, RC}{iMAE + RC} \tag{2.15}$$

where iMAE is the inverse MAE as formulated in [21]. It is the accuracy of prediction normalized by the maximum r_{max} and minimum r_{min} numbers in the rating scale. Thus:

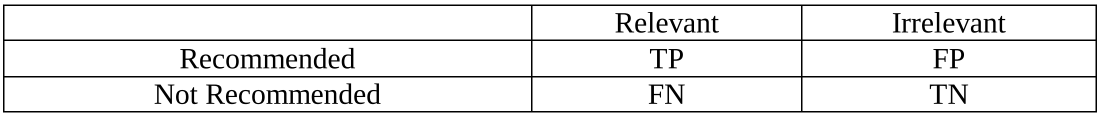
$$iMAE = 1 - \frac{MAE}{r_{max} - r_{min}} \tag{2.16}$$

the system accuracy is better when the iMAE values are higher.

2.8.2 Top-N Recommendation Task Metrics

Top-N recommendation task means recommending the preferred items for target users by converting the numerical scales (i.e., 1-5, 1-10) to the binary scale (i.e., like and dislike) using a threshold value.

Three well-known classification metrics have been considered to be among the most critical performance metrics for the Top-N recommendation task of RS. They are Precision, Recall, and F1-measure [19]. Figure 2.5 depicts the confusion matrix, which indicates the terms used in the metrics equations below.



	Relevant	Irrelevant
Recommended	TP	FP
Not Recommended	FN	TN

Figure 2.5: Confusion Matrix

According to Figure 2.5, True Positive (TP) defines the number of recommended relevant items, while True Negative (TN) describes the number of items that are not recommended and irrelevant. False Positive (FP) represents the number of irrelevant items that are wrongly recommended, whereas False Negative (FN) expresses the number of relevant items, but not recommended.

Since Precision and Recall are among the best metrics of the classification task, they will be adopted in this thesis after depicting the task of the proposed model as a classification problem. Thus, they are used to estimate the accuracy of the recommendation in proposed review-based recommendation models (RRMs).

In particular, Precision defines the number of correct recommended items (TP) to all recommended ones within the Top-N list (TP + FP) and as in its following equation:

$$Precision@N = \frac{1}{U_t} \sum\nolimits_u \frac{|recommended\ items\ that\ are\ relevant|}{|recommended\ items|} \quad (2.17)$$

on the contrary, Recall determines the number of correct recommended items (TP) among all relevant (positive-rated) items that existed in the test set (TP + FN), as depicted in the following equation:

$$Recall@N = \frac{1}{U_t} \sum\nolimits_u \frac{|recommended\ items\ that\ are\ relevant|}{|relevant\ items|} \quad (2.18)$$

where U_t is the number of test users.

Inevitably, there will be a trade-off between the two metrics mentioned above when selecting N of items for overall recommendation evaluation. Therefore, a third comprehensive metric called F1-measure is set up to balance the two metrics, as shown in the equation below:

$$F1@N = \frac{2\ .\ Precision@N\ .\ Recall@N}{Precision@N + Recall@N} \quad (2.19)$$

3.1 Overview

This chapter presents the architecture of the proposed system, which consists of two approaches: a trust-based prediction model (TPM) and review-based recommendation models (RRMs). Section 3.2 presents a description of the five datasets used in this thesis. In addition, the TPM will be explained in section 3.3. Lastly, in section 3.4, RRMs will be described separately.

The general architecture consists of two proposed approaches: a trust-based prediction model (TPM) and review-based recommendation models (RRMs). Each model consists of multiple steps and is trained separately from the other and tested on different datasets (see Tables 3.1 and 3.2) under various evaluation metrics according to each model's task. Both types of models differ in their preprocessing step, however, they share the same processing phase that is related to the user-item rating matrix (i.e., converting *NaN* values into zeros). These proposed models will be clarified in the following sections. It is worth noting that the equations mentioned in this chapter have been formulated according to the author of this thesis's ideas.

3.2 Description of Datasets

In this thesis, five datasets are used to train and test two approaches: a trust-based prediction model (TPM) and review-based recommendation models (RRMs). Tables 3.1 and 3.2 describe the datasets used in each approach individually.

Table 3.1: Trust-oriented Datasets Description

Datasets	#Users	#Items	#Ratings	#Trust Relations	Rating Sparsity	Avg. Rating	Trust Sparsity	Avg. Trust
FilmTrust	1508	2071	35497	1853	98.86%	23.53	99.58%	3.04
Epinions	49 K	139 K	664 K	487 K	99.98%	16.55	99.97%	14.35

Table 3.2: Review-oriented Datasets Description

Datasets	#Users	#Items	#Ratings #Reviews	Rating Sparsity	Avg. Reviews	Min, Max Reviews
Musical Instruments	1429	900	10261	99.2%	7.18	(5, 163)
Automotive	2928	1835	20473	99.6%	6.99	(5, 169)
Amazon Instant Video	5130	1685	37126	99.5%	7.23	(5, 455)

In the TPM, two datasets are obtained for the experiment and evaluation purposes, namely, FilmTrust[3] and Epinions[4] [21]. It is worth noting that due to memory limitations in the local machine, a representative sample is taken from the large-scale Epinions dataset by randomly selecting 1,923 users who recorded 27,013 ratings on 4,221 items. The sampled dataset of Epinions comes with a rating sparsity of (99.66%) with an average rating of about 14.04, and 250888 of trust relations with a sparsity ratio of (99.60%) and 31.92 as average trust. In addition to the rating information contained in these datasets, there are explicit trust relations acquired as additional information. Two files for each of the above datasets: the ratings file and the trust relations file.

In RRMs, three datasets from Amazon[5] are obtained for the experiment and evaluation purposes. They are Musical Instruments, Automotive, and Amazon Instant Videos, which were first introduced in [69]. These datasets contain product textual reviews and numerical rating information from a repository of RSs datasets. Specifically, the columns included in the dataset file are reviewer ID, product ID, ratings, review

[3] https://guoguibing.github.io/librec/datasets.html#filmtrust

[4] http://www.trustlet.org/datasets/downloaded_epinions/

[5] http://jmcauley.ucsd.edu/data/amazon/links.html

texts, helpfulness votes, and timestamps. That is, one file contains all the information for each of the datasets above. All Amazon datasets include customer activities for various categories of products reviewed by users of Amazon.com from May 1996 to July 2014. They are 5-core datasets in which each user and item has a minimum of 5 pieces of feedback information.

It is worth noting that each one of the five datasets contains either trust relations information or textual reviews information in addition to the numerical ratings information. The details of all these datasets will be described in the next chapter.

As shown in Table 3.1 for the datasets: FilmTrust and Epinions, the number of explicit trust relations is provided because they are used to verify the proposed TPM. Whereas for Amazon datasets: Musical Instrument (MI), Automotive (AM), and Amazon Instant Video (AIV), as shown in Table 3.2, the count of textual reviews generated by users is provided because they are utilized to validate the proposed RRMs. The columns of the number of ratings and the number of reviews are merged into one column because they have an equal count.

Some statistics of the tables above are calculated through a function within the preprocessing stage of the proposed system. These are the sparsity ratio of the matrices of the user-item ratings (aka, rating matrix) and trust relations (aka, adjacency matrix), as well averages of ratings, trust relations, and textual reviews, besides count the min/max number of reviews. Such statistics are unavailable in the description of the datasets on their websites.

3.3 The Proposed Trust-based Prediction Model

This model is built based on both explicit trust relations which already exist in the dataset and the implicit ones that are inferred depending on the propagation attribute of the trust theory (refer to section 2.6). It is important to note that the procedure used in Merge and EIMerge methods are followed, but by applying the weighted voting technique (rating election) rather than the weighted average technique for electing (merging) ratings. The basic idea is leveraging from the top K trusted neighbors of the trust network through electing their ratings for an active user to boost his/her preference profile. Figure 3.1 illustrates the steps of the prediction model which deal with the trust links that connect users to enrich the sparsity nature of the user-item rating matrix.

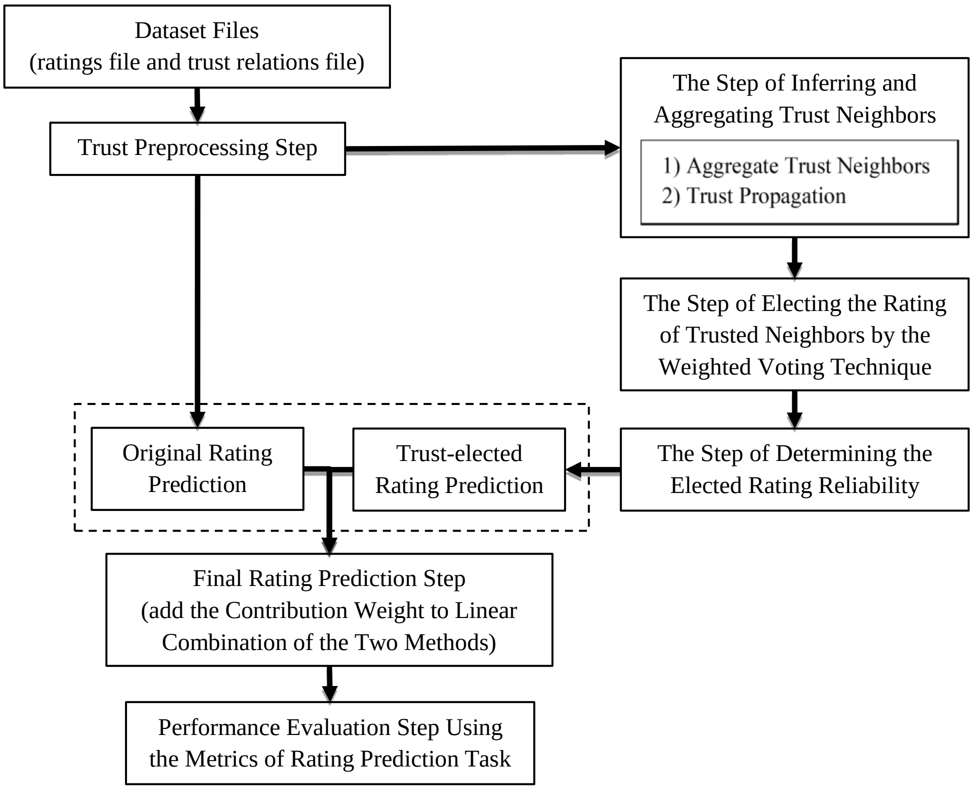

Dataset Files
(ratings file and trust relations file)
Trust Preprocessing Step
The Step of Inferring and Aggregating Trust Neighbors
1) Aggregate Trust Neighbors
2) Trust Propagation
The Step of Electing the Rating of Trusted Neighbors by the Weighted Voting Technique
The Step of Determining the Elected Rating Reliability
Original Rating Prediction
Trust-elected Rating Prediction
Final Rating Prediction Step
(add the Contribution Weight to Linear Combination of the Two Methods)
Performance Evaluation Step Using the Metrics of Rating Prediction Task

Figure 3.1: Steps of the Trust-based Prediction Model (TPM)

The above model is depicted with the following steps:

A. Trust Preprocessing Step

This step of the proposed TPM, consist of two phases: the first phase mandatory due to memory constraints since a representative sample from the Epinions dataset is taken to compress its vast size. This representative sample is made by randomly selecting 1,923 users who recorded 27,013 ratings on 4,221 items. The random sampling process is done because it is impossible to represent the rating matrix from the original numbers of users and items due to memory limitations of the local machine. The second phase is conducted on the users' trust relations network after the first step. It was necessary to delete any link unrelated to the sampled users to avoid redundant computations that would not affect the final result.

Then, after building the user-item rating matrix R from the rating file of FilmTrust and Epinions datasets, each NaN value is converted to zero to avoid any failure in the calculations later.

B. The Step of Inferring and Aggregating Trust Neighbors

Due to the high sparsity of the trust relations (adjacency) matrix, it is similar to the user-item rating matrix in terms of sparsity, as illustrated in Table 3.1. Thus, more implicit trust relations among users must be inferred and aggregated to define their trusted neighbors. To achieve this, two phases of this step are listed as follows:

1) Aggregate Trust Relations Phase

Since the very common PCC is utilized as a similarity metric to identify implicitly trusted neighbors (refer to section 2.5.1). The PCC calculation formula (refer to Eq. 2.6) is used to get the original similarity value $rsim_{u,v}$ between user u and user v.

In order to provide the asymmetric property of trust theory for the inferred trust relations, the value of common items set related to the target user is calculated, then combine it with the result of Eq. (2.6) to obtain the average value between them, as shown in the following two equations:

$$\mathrm{CIS}_{\mathrm{u,v}} = \frac{|I_u \cap I_v|}{|I_u|} \tag{3.1}$$

$$Rsim_{u,v} = \frac{(rsim_{u,v} + \mathrm{CIS}_{\mathrm{u,v}})}{2} \tag{3.2}$$

where $\mathrm{CIS}_{\mathrm{u,v}}$ is the common items set between user u and user v, and I_u, I_v are the set of items that are rated by both users respectively, $Rsim_{u,v}$ is the overall similarity value among the two respective users. To infer implicit trust relations between the network users, $Rsim_{u,v}$ value is used as the inferred trust value as shown in Eq. (3.3). The formula of inferred (implicit) trust value is as follows:

$$T_{INF_{u,v}} = \begin{cases} Rsim_{u,v}, & if\ Rsim_{u,v} > \theta_{Rsim} \land |I_{u,v}| > \theta_{I_{u,v}} \\ 0, & otherwise \end{cases} \tag{3.3}$$

where $T_{INF_{u,v}}$ is the inferred (implicit) trust degree of user u in user v, and θ_{Rsim}, $\theta_{I_{u,v}}$ are the rating similarity threshold and the rating number threshold respectively (previously determined in section 2.5.1).

2) Trust Propagation Phase

In this phase, the aim is to find the indirectly trusted neighbors by diffuse trust in the network of trust, depending on the transitive property of trust theory. This result in additional valuable information which can be utilized to improve the recommendation accuracy. However, going deeper to propagate inside the trust network may not likely provide useful information. According to the experiment of the proposed TPM, the best gain is in adopting 3-step propagation as shown in Eq. (3.4), to avoid meaningless exploration and useless consumption, especially when dealing with large-scale datasets (e.g., Epinions). The indirect (implicit) trust relation can be inferred between any two users in the trust network. The propagated (implicit) trust degree between two users is inversely proportional to their shortest distance (refer to section 2.6). The propagated (implicit) trust value $T_{Prop_{u,v}}$ is computed as follows:

$$T_{Prop_{u,v}} = \frac{1}{sd_{u,v}}, \text{ s.t. } \left|sd_{u,v}\right| \leq 3 \tag{3.4}$$

where $sd_{u,v}$ is the shortest distance between u and v, and $T_{Prop_{u,v}} \in [0, 1]$. From now on, the symbol $T_{u,v}$ is used to denote each of the explicit trust relations given in the dataset, and implicit trust relations that called the inferred ($T_{INF_{u,v}}$) and the propagated ($T_{Prop_{u,v}}$), both obtained from Eqs. (3.3) and (3.4). Any user trusted more by another with trust value ($T_{u,v}$) than a pre-defined threshold ($\theta_{T_{u,v}} = 1/4$) is regarded as a trusted neighbor, as in the following formula:

$$TN_u = \left\{v \mid T_{u,v} > \theta_{T_{u,v}}, \text{ where } v \in U\right\} \tag{3.5}$$

the trusted neighbors of user u in the network of users U are TN_u, with $\theta_{T_{u,v}}$ referring to the trust relation threshold.

The trust relations network (directed graph) builds mainly from the explicit trust links available in the database to form the initial appearance of a directed graph, as shown in Figure 3.2. After that, the trust propagation feature is employed to spread out the network and fill the adjacency matrix (aka, trust relations matrix), which corresponds to the graph by the inferred implicit links, as shown in Figure 3.3. With this, the circle of user trustees will be expanded to include the largest possible number, whose ratings, after applying the idea of rating election, will mitigate the sparsity in the rating matrix. Figures 3.4 and 3.5 show an example of the trust relations matrix before and after the propagation of trust relations. It is worth noting that each user fully trusts himself. Therefore, the main diagonal of the trust relations matrix is appeared in values equal to 1, as shown in bold.

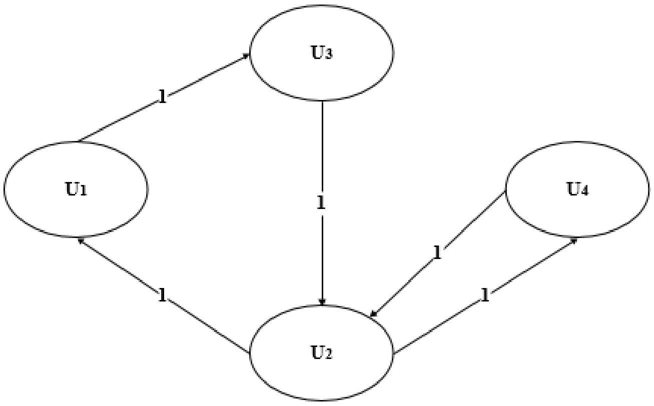


Figure 3.2: Trust Relations Network on Four Users Before Propagation

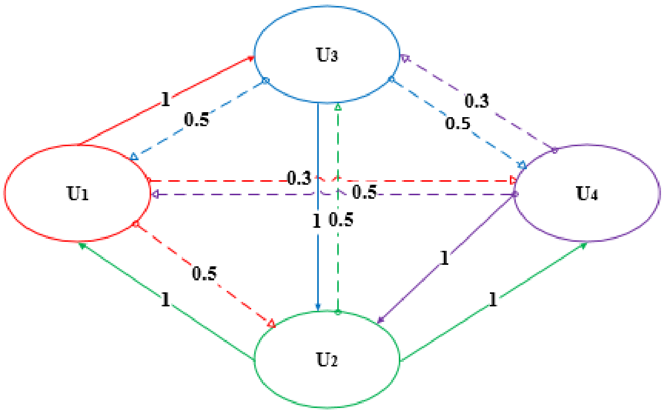


Figure 3.3: Trust Relations Network on Four Users After Propagation

Trustors	Trustees			
	u_1	u_2	u_3	u_4
u_1	**1**		1	
u_2	1	**1**		1
u_3		1	**1**	
u_4		1		**1**

Figure 3.4: Trust Relations Matrix on Four Users Before Propagation

Trustors	Trustees			
	u_1	u_2	u_3	u_4
u_1	**1**	**0.5**	1	**0.3**
u_2	1	**1**	**0.5**	1
u_3	**0.5**	1	**1**	**0.5**
u_4	**0.5**	1	**0.3**	**1**

Figure 3.5: Trust Relations Matrix on Four Users After Propagation

According to Figures 3.4 and 3.5, let us take u_4 as the target user, hence observe his/her relationships before and after leveraging the propagation of trust relations. Also observe his/her ratings, as shown previously in Figure 2.1, before applying the election of trustworthy users' ratings. Specifically, u_4 has one trust relation with u_2, as shown in Figures 3.2 and 3.4. Then in Figures 3.3 and 3.5, his/her relations have increased to include u_1 and u_3 as TN_{u4} (refer to Eq. (3.5)), but with a relative trust degree calculated from the inverse of the shortest path between them in the network (refer to Eq. (3.4)). Recalling from the rating matrix (refer to Figure 2.1) that u_4 has only one rating on i_2 with a value of 5. Accordingly, the ratings of his/her trustees (u_2, u_1, and u_3) will be elected over the other unknown items like i_1 to fill in his/her missing ratings (as the cell with question mark (?) in Figure 2.1). Note that, the trustees of u_4, whose are (u_2, u_1, and u_3), all rated the i_1 with (4, 1, 1) values, respectively. Thus, rating 4 is chosen as the rating with the highest voting weight among the trustees' ratings on i_1 as follows:

The candidate ratings for u_4 on i_1 from u_1, u_2, $u_3 \rightarrow \{1,4\}$

The shortest paths of u_4 to his/her trustees → $sd_{u4,u1} = 2$,

$sd_{u4,u3} = 3$,

$sd_{u4,u2} = 1$

The voting weight of rating 1 → $T_{u4,u1} + T_{u4,u3}$

$$= \frac{1}{\mathrm{sd}_{u4,u1}} + \frac{1}{\mathrm{sd}_{u4,u3}}$$

$$= \frac{1}{2} + \frac{1}{3}$$

$$= 0.5 + 0.3 = 0.8$$

The voting weight of rating 4 → $\mathrm{T}_{\mathrm{u4,u2}} = \frac{1}{\mathrm{sd}_{u4,u2}}$

$$= \frac{1}{1} = 1$$

So, the elected rating of u_4 on i_1 is equal to 4 (i.e., $\tilde{r}_{u4,i1} = 4$).

This simple example illustrates the trust propagation and the idea of rating election in general. The whole algorithm includes other factors that will be further illustrated in Algorithm 3.1.

The reason for employing the trust propagation attribute as shown in the above example is to extend the trust relations network (i.e., enriching the trust relations matrix). Thus, reducing the sparsity of trust links that will positively affect the rating sparsity resulting in better prediction outcomes. Otherwise, if the proposed prediction model only depends on explicit trust relations, a high ratio of rating matrix blanks will remain empty (i.e., NaN values). Therefore, it cannot estimate the rating of target users causing low prediction coverage and producing ineffective recommendations.

C. The Step of Electing the Rating of Trusted Neighbors

Algorithm 3.1 illustrates the proposed rating election algorithm, in which the ratings of trusted neighbors are elected as a single value on some items j that an active user u did not rate, and these $j \in I'_u$ are rated at least by one trusted neighbor. Therefore, a voting weight of a trusted neighbor representing the relation degree with the active user must be accumulated to achieve the election process. In addition to the rating similarity value calculated in Eq. (2.6) and the trust value calculated in Eqs. (3.3) and (3.4), the factor of social confidence $SC_{u,v}$ is computed as in Eq. (2.7) to obtain the voting weight. The social confidence represents the indicator of the trustworthiness of the users concerning each other. Hence, a linear combination of the three factors ($T_{u,v}, rsim_{u,v}, SC_{u,v}$) with parameters α, β, and Υ is required to obtain the voting weight $VW_{u,v}$ between an active user u and every trusted neighbor v, as follows:

$$VW_{u,v} = \alpha \, . \, T_{u,v} + \beta \, . \, rsim_{u,v} + \Upsilon \, . \, SC_{u,v} \tag{3.6}$$

where $\alpha, \beta,$ and Υ represent how much each factor contributes to the combination. These constants are tuned according to the TPM experiments, their values are 0.5, 0.3, 0.2 on FilmTrust and 0.7, 0.2, 0.1 on Epinions, respectively. It is important to involve all three factors instead of only trust value for having a significant influence on prediction results and avoiding tie votes. Moreover, it is worth mentioning that only the positive rating similarity $rsim_{u,v} > 0$ is considered in the formula above (Eq. (3.6)). After saving the voting weight between every active user and his/her trusted neighbors for every rating r in the rating scale, the elected value $\tilde{r}_{u,j}$ can be voted as the rating r that is associated with the maximum voting weight. Thus:

$$\tilde{r}_{u,j} = \mathrm{r}_{\arg\underset{j}{max} \sum_{v}^{TN_u} VW_{u,v}^{j}} \tag{3.7}$$

where $\tilde{r}_{u,j}$ is the elected rating of an active user u on an item $j \in I'_u$, and $VW_{u,v}^{j}$ is the voting weight of the item j of every trusted neighbor v that belongs to TN_u.

Algorithm 3.1: Rating Election Algorithm

```
Input:
    ➢ Rating Matrix $R$, Rating Scale $R^s$, Users Set $U$, Items Set $I$
    ➢ Trust Direct Graph $dg$
    ➢ Setting the parameters $\alpha$, $\beta$, $\Upsilon$ according to the experiments
Output:
    ➢ Elected Rating Matrix $E$
Begin
    ➢ Preprocessing Rating Matrix $R$
    ➢ Preparing Empty Elected Rating Matrix $E$
    ➢ Preparing Count Dictionary $C$
    ➢ Collecting Trusted Neighbors $TN$ and Trust Values $T$ from Direct Graph $dg$
    ➢ Computing Rating Similarity by Eq. (2.6)
    ➢ Computing Social Confidence by Eq. (2.7)

    For each $u$ in $U$
      For each $j$ in $I$
        If $R_{u,j}$ == 0                // The missing rating of target user needs an election
          For each $r$ in $R^s$
            $Count \longleftarrow 0$
              For each $n$ in $TN_u$
                If $R_{n,j} > 0$         // Trust Neighbor has rate the target item
                  Calculate $Count$ $C_{u,j}$ of all $TN_u$ based on the output of Eq. (3.6)
                End if
              End for $n$
            Save $C_{u,j}$ of current rating $r$ in Count Dictionary $C$
          End for $r$
          $E_{u,j} \longleftarrow r_{\text{argmax}} C_{u,j}$    // Add elected rating $r$ of maximum $C_{u,j}$ to $E$ as in Eq. (3.7)
        Else                             // Target user has a rating for the target item
          $E_{u,j} \longleftarrow R_{u,j}$              //Add the original rating $R_{u,j}$ to $E$
        End if
      End for $j$
    End for $u$
End.
```

D. The Step of Determining the Elected Rating Reliability

The elected rating reliability $RL_{u,j}$ must be computed to ensure the elected rating certainty for approbation of the elected ratings. It can be defined as the system certainty in the elected rating. Two factors are necessary to obtain the rating reliability. First is the maximum voting weight involved in choosing the elected rating. Second is all voting weights values associated with each candidate rating r in the rating scale. The reliability value of the elected rating $\tilde{r}_{u,j}$ is previously computed in

Eq. (2.8). Regarding the actual ratings, their reliability will be the highest $RL_{u,i} = 1$ (i.e., 100% reliable) for all items $i \in I_u$. The notion of this reliability is that the less reliable an elected value is, the more liable it is to be inaccurate.

E. Final Rating Prediction Step

By the end of the rating election process, every item $i \in I$ will be associated with two values: actual rating $r_{u,i}$ or elected rating $\tilde{r}_{u,i}$ and the corresponding rating reliability $RL_{u,i}$. Thus, a new preference profile is provided for every user $u \in U$. The user-based CF algorithm can be applied for rating prediction depending on this new profile. Therefore, two steps are conducted next.

First, the trust similarity $Tsim_{u,v}$ among users is calculated by RPCC (refer to Eq. (2.9)) which is similar to PCC, but with incorporating the rating reliability value to reduce the impact of less reliable elected ratings. It is important to note that $Tsim_{u,u} = 1$ due to the user u being part of his/her trusted neighbors TN_u. Since the trust similarity values $Tsim_{u,v}$ ranged in $[-1, 1]$, a trust similarity threshold ($\theta_{Tsim_{u,v}} = 0$) can filter only the positive trust relations among users. The filtered users are then added to the group of nearest neighbors NN_u. Thus:

$$NN_u = \left\{ v | Tsim_{u,v} > \ \theta_{Tsim_{u,v}} , \ \text{where} \ v \in U \right\} \qquad (3.8)$$

it is noteworthy that the KNN method (refer to section 2.4.1.1) is selected to determine the neighborhood which is optimal at K = 25 according to the proposed TPM experiments.

Second, the prediction of unrated items can be computed by collecting all the ratings of u's nearest neighbors NN_u on the target item j and multiplied by their trust similarity with the active user u. The similarity weight assures more influence for the most like-minded neighbors to active users. Hence, the CF prediction formula is applied. Thus:

$$\bar{r}_{u,j} = \bar{r}_{\mathrm{u}} + \frac{\sum_{v \in NN_u} Tsim_{u,v} \cdot (r_{v,j} - \bar{r}_{\mathrm{v}})}{\sum_{v \in NN_u} |Tsim_{u,v}|} \tag{3.9}$$

where $\bar{r}_{u,j}$ refers to the predicted value of unrated item j.

Further, an examination of the impact of mixing original ratings and trust-elected ratings on the CF prediction outcome is included. Therefore, with the above method based on the trust similarity values, another method is presented that depends only on the values of original rating similarity calculating using Eq. (2.6) and repeating Eqs. (3.8) and (3.9) by replacing trust similarity $Tsim_{u,v}$ with the original rating similarity $rsim_{u,v}$ to obtain the prediction result $\bar{\bar{r}}_{u,j}$ from them. Then, the results of the two methods are combined linearly by incorporating a contribution weight $ContW$ ranging from 0 to 1 and observing the results to set the appropriate weight that gives the best possible prediction and coverage, as follows:

$$\hat{r}_{u,j} = \begin{cases} \bar{r}_{u,j} \cdot (1 - ContW) + \bar{\bar{r}}_{u,j} \cdot ContW\ , & if\ \bar{r}_{u,j} > 0 \ \wedge\ if\ \bar{\bar{r}}_{u,j} > 0 \\ \bar{\bar{r}}_{u,j}\ , \ if\ \bar{r}_{u,j} = 0 \ \wedge\ if\ \bar{\bar{r}}_{u,j} > 0 & \\ \bar{r}_{u,j}\ , \ if\ \bar{r}_{u,j} > 0 \ \wedge\ if\ \bar{\bar{r}}_{u,j} = 0 & \\ 0,\ otherwise & \end{cases} \tag{3.10}$$

where $\hat{r}_{u,j}$ refers to the final predicted value of unrated item j after involving the two prediction results $\bar{r}_{u,j}$ and $\bar{\bar{r}}_{u,j}$, which are the predicted values obtained from the trust similarity and from the original rating similarity, respectively. Note from the above equation that if one of the two outcomes is greater than zero, it is taken in full without adding the contribution weight to them. Otherwise, when they are unavailable (both are zero), the last result will be zero.

F. Performance Evaluation Step Using the Metrics of Rating Prediction Task

The performance evaluation is the last step of the proposed trust-based prediction model (TPM). It represents an important step to test the effectiveness of the proposed model. According to the rating prediction task that accomplished in this model, metrics like (MAE, RMSE, and RC) as described in section 2.8.1, are used to assess the estimation quality and coverage of the proposed TPM.

At this step, the evaluation is done on the test set after building the proposed model on the training set. That is, after dividing the dataset into the training and test sets with the usual ratio of 1:4. In particular, 80% of the dataset is devoted to the purpose of building and generalizing the model to be able to predict and classify test ratings which represent 20% of the ratings of each trained user. In the rating prediction task, the estimated ratings resulting from applying the prediction model to the ratings of the training set are compared with their counterparts of actual ratings in the test set to check the amount of prediction error.

The following example depicts the performance evaluation step for the task mentioned above. According to the user-item rating matrix previously shown in Figure 2.1, assume that Figures 3.6 and 3.7 represent the training set and the test set of the rating matrix, respectively.

Users	Items					
	i_1	i_2	i_3	i_4	i_5	i_6
u_1			3	5		2
u_2	4	2	1	3		
u_3	1				4	
u_4		5				

Figure 3.6: The Training set of the Rating Matrix

Users	Items					
	i_1	i_2	i_3	i_4	i_5	i_6
u_1	1					
u_2					5	4
u_3		4				
u_4						

Figure 3.7: The Test Set of the Rating Matrix

Now, let the estimated values of all test users (u_1, u_2, u_3) for test items (i_1, i_2, i_5, i_6) after conducting the proposed TPM be (1.1, 4.5, 4, 3), respectively. Then MAE, RMSE, and RC metrics are computed between the actual ratings of test users (1, 5, 4, 4) and the estimated ones (1.1, 4.5, 4, 3) according to Eqs. (2.12), (2.13), and (2.14) respectively.

3.4 The Proposed Review-based Recommendation Models

This section deals with user-generated textual reviews, i.e., user comments on exciting items. Such rich textual information can be used to improve the recommendation accuracy of the proposed system (refer to section 2.7).

It is worth clarifying that the proposed review-based model is built on two different cases. The first case includes the availability of textual reviews within the test set. In the second case, however, the test set does not contain these reviews (i.e., users did not submit their comments), but it only contains numerical ratings. Thus, there are two diverse tracks upon which the proposed review-based recommendation model (RRM) is operated. In the first track, the textual reviews of the test set are compared to those within the training set. Whereas, In the second track, the model compares the textual reviews of the target user with those of the target item within the training set. The aim in both tracks is to derive the appropriate adjustment weights that will improve the model's accuracy based on the Naïve Bayes classifier for the Top-N recommendation task. Choosing one of the two models within the proposed system depends on whether the target user has written a review on the item or not. In other words, if this user has a comment on the target item within the test set, RRM$_1$ will be selected, otherwise, the system will enter RRM$_2$.

3.4.1 The Proposed RRM with Test Reviews

Figure 3.8 shows the steps of the recommendation model (RRM$_1$) when textual reviews are available within the test set. The below model is depicted with the following steps:

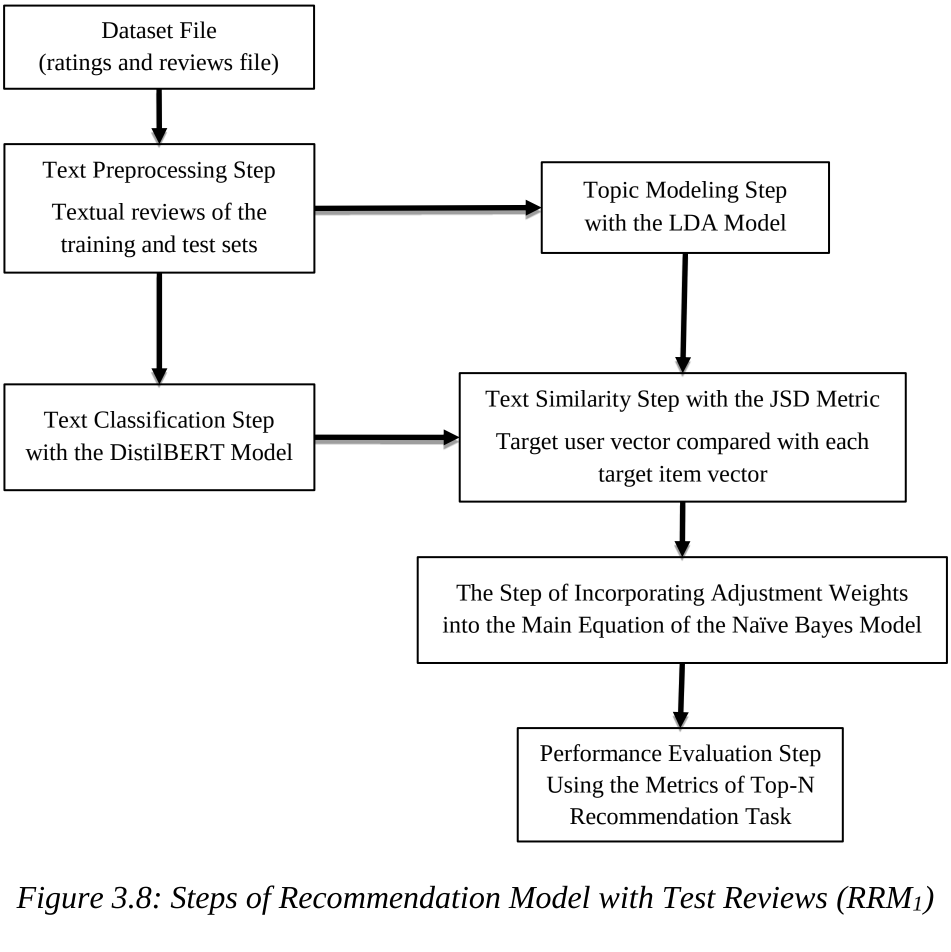

Dataset File
(ratings and reviews file)
Text Preprocessing Step
Textual reviews of the training and test sets
Topic Modeling Step with the LDA Model
Text Classification Step with the DistilBERT Model
Text Similarity Step with the JSD Metric
Target user vector compared with each target item vector
The Step of Incorporating Adjustment Weights into the Main Equation of the Naïve Bayes Model
Performance Evaluation Step Using the Metrics of Top-N Recommendation Task

Figure 3.8: Steps of Recommendation Model with Test Reviews (RRM$_1$)

A. Text Preprocessing Step

This step consists of four phases necessary to perform the review-based recommendation model (RRM_1), as shown in Figure 3.9.

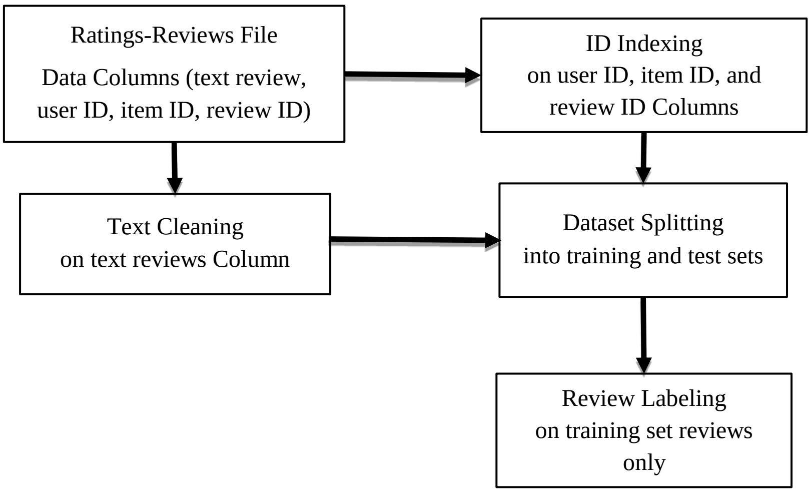

Ratings-Reviews File
Data Columns (text review, user ID, item ID, review ID)
ID Indexing
on user ID, item ID, and review ID Columns
Text Cleaning
on text reviews Column
Dataset Splitting
into training and test sets
Review Labeling
on training set reviews only

Figure 3.9: Phases of Text Preprocessing

Firstly, the text cleaning phase includes the common preprocessing operations of the text, such as case folding, stop-words removal, tokenization, and lemmatization, as previously explained in section 2.7. In addition, further extra preprocessing operations on the text reviews have been employed as follows:

1) Combining the words of the review title within its text to increase the number of weighted words and avoids empty reviews that may result after applied some operations that involve delete actions.
2) Substituting accented characters: this procedure is done before the case folding procedure to achieve the uniformity, in which the marked characters are replaced with their alternatives, the regular characters, such as: café → cafe; naïve → naive.
3) Expanding the contractions: it means reverting the abbreviated words (i.e., a contraction is a shortened form of a group of words) to their original form before removing the punctuation marks. Thus, it allows

deleting the meaningless words created from the removing process, such as: aren’t → arent; here the word (aren’t) will be converted to (are not). This procedure is beneficial for better text classification.

4) Removing punctuation marks, special characters, and trailing whitespace that may be present at the beginning or end of the text, in addition to extra spaces between the text’s words.
5) Deleting digits and short words, i.e., any word that does not exceed two letters because it does not represent a weighted word. This operation must occur after the tokenization function.
6) Excluding HTTP Web links and HTML tags.

The second phase (i.e., ID indexing) involves converting all user/item identifiers (IDs) into ordered integers since they are available as string types in the original dataset. Moreover, adding ordered integers IDs to user-generated reviews (dataset rows) facilitates the process of indexing and recalling them in the computational functions.

After performing all previous operations, the third phase (i.e., dataset splitting) divides the data into five folds as training and test sets with a ratio of 1:4 (i.e., 80% for training and 20% for testing) to train and test the proposed model. It is worth noting that the text reviews of the test set are utilized in this model’s track (the current section), while in return, they are eliminated from the other track of the model (see section 3.4.2).

The final phase of preprocessing step is responsible for labeling each textual review of the training set, which is an essential procedure for applying supervised classification of text in the second step of the RRM$_1$. Specifically, the training reviews are labeled with a rating of less than or equal to three ($r \leq 3$) as negative reviews and the remaining (i.e., with a rating of more than three ($r > 3$)) as positive reviews. This stage is necessary to perform binary classification of text reviews. This distribution is used, that is counting the score three as a negative rating, to balance the low number of negative reviews with the massive count of positive reviews in the datasets. The ultimate goal of the proposed system is to either recommend the item or not, therefore the labeling phase by converting the multiple rating scales into a binary scale (i.e., like and dislike) is the most appropriate procedure.

B. Text Classification Step with the DistilBERT Model

After completing the data preprocessing step, the test set reviews are required to be classified based on the training of the DistilBERT model (refer to section 2.7.1.2) on the training set reviews as shown in Figure 3.10, for carrying through RRM$_1$.

The DistilBERT model first requires that the training set reviews are prepared by specifying a maximum fixed length of text. Then, these reviews are encoded, by the model tokenizer, to be associated with labels (i.e., classes) in a temporary dataset ready for the subsequent process.

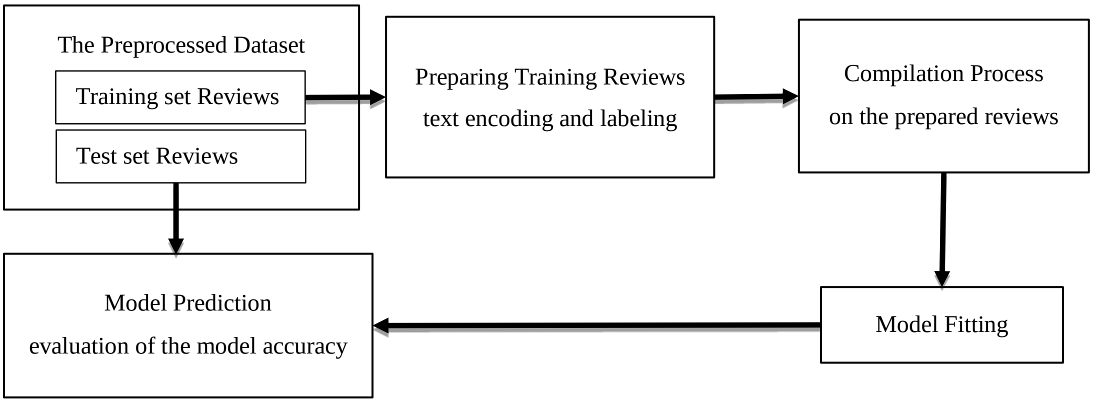


Figure 3.10: Process of DistilBERT Model

After that, the compilation process of the model begins by setting its parameters which will contribute to the model’s fitting process on the prepared training reviews. Such parameters include specifying batch size, the number of epochs, type of classification, loss function, and optimizer with its constants: learning rate and epsilon value. These are a sequence classification, a sparse categorical cross entropy as loss function, and an Adam optimizer with the best-tuned values of batch size, number of epochs, learning rate, epsilon factor, and the maximum length of text (see section 4.2.2(II)).

Finally, the model is fitted to the prepared training reviews and its classification accuracy is examined on the test set reviews, thereby evaluating its performance and equipping it to be employed in the text similarity step of RRM_1.

C. Topic Modeling Step with the LDA Model

In this crucial step, the topic probability distributions of the user-generated reviews are obtained by using the LDA model previously explained in section 2.7.1.3 and its process is illustrated graphically in Figure 3.11.

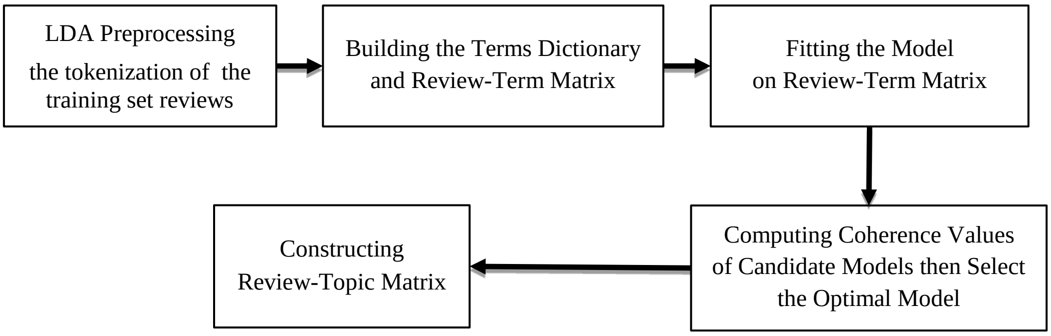

LDA Preprocessing
the tokenization of the training set reviews
Building the Terms Dictionary and Review-Term Matrix
Fitting the Model on Review-Term Matrix
Computing Coherence Values of Candidate Models then Select the Optimal Model
Constructing Review-Topic Matrix

Figure 3.11: Process of LDA Model

The LDA model will handle the tokenized copy of the training set reviews to build the dictionary of terms and review-term matrix. Then the parameters of training the LDA model on textual reviews are set to reach the best coherent result from the topic probability distributions for each review of the training set. Specifically, the model is tuned on a range of K topics from 5 to 50 (see section 4.2.2(III)).

After conducting the experiments over the specified range of K topics, the model with the optimal number of topics will be selected based on the maximum coherence value, which is calculated during the training of each candidate model. Lastly, a review-topic matrix is constructed based on the optimum model that trained on all textual reviews of the training set. Later in the text similarity step, the chosen model will be adopted in constructing review-topic vectors for the test set reviews to compare them with the constructed review-topic matrix.

D. Text Similarity Step with the JSD Metric

This step is the most significant in the proposed model, which is consisting of two main parts: the first for calculating the semantic similarity of the texts, followed by the algorithm for deducing the adjustments weights. These weights contribute to balance the classifications (i.e., class imbalance) of the Naïve Bayes model used in the next step.

The results of the text classification and topic modeling steps on the test set reviews represent the input to the text similarity step that is performed using several operations, as shown in Figure 3.12.

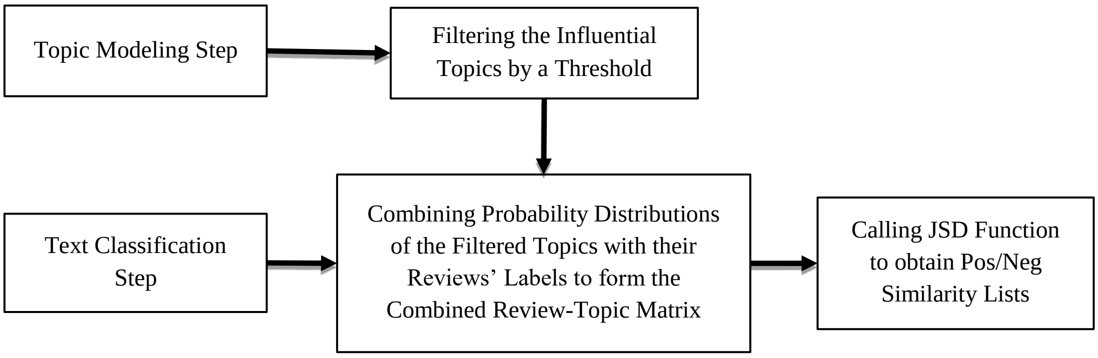

Topic Modeling Step
Filtering the Influential Topics by a Threshold
Text Classification Step
Combining Probability Distributions of the Filtered Topics with their Reviews' Labels to form the Combined Review-Topic Matrix
Calling JSD Function to obtain Pos/Neg Similarity Lists

Figure 3.12: Text Similarity Step

The text similarity step initiates by filtering the ineffective topics of the topic probability distributions that result from the topic modeling step, after dynamically determining the influential topics' threshold θ that depends on K number of topics of the LDA model, as follows:

$$\theta = \frac{1}{K} \tag{3.11}$$

where K is the number of topics of the chosen LDA model.

The next operation of the text similarity step is to combine the probability distributions of the filtered topics with their reviews' labels to form the combined review-topic matrix of all reviews necessary to implement the JSD metric as the last operation. The following figure illustrates an example of the combining operation:

Set	Target	Text Review	Label
Test	User	I have had this pedal for about 15 years, just replaced the old one a month ago. It distorts sound, not quite the way you would imagine though. This effect sounds really good …	**?**
Training	Item	This was the first pedal I ever bought when I started playing guitar 15 years ago. It's still in my primary set up! The sound it puts out is very balanced …	**1**
		This is a cheap piece of junk that does what it says, it distorts, You want something to make your guitar sound like junk, this will do it. Why pedals that make your guitar sound like a piece of junk are the most popular …	**0**

Figure 3.13: Sample of Comments (Textual Reviews) from the MI Dataset

According to Figure 3.13, the availability of three textual reviews is observed. One of them, from the test set belongs to the target user, will refer to it as *Urev*. The remaining two reviews, both from the training set which are related to the target item, will refer to them as $Irev_1$ and $Irev_2$, respectively. Also notice next to each comment is the label column where its values are known in the training set such as $Irev_1$ as a positive comment (i.e., 1) and the other as negative (i.e., 0). As for the comment *Urev* of the test set, its label is remains unknown (i.e., ?) until processed in the text classification step.

According to Figure 3.12, where the text similarity step starts with the end of topic modeling step for all text reviews and is then followed by filtering only the important topics. Suppose this step is done with K = 5 topics, so each review will have a topic probability vector as follows:

$Urev = <0.62, 0.004, 0.206, 0.004, 0.102>$

$Irev_1 = <0.003, 0.537, 0.003, 0.269, 0.139>$

$Irev_2 = <0.48, 0.109, 0.08, 0.003, 0.267>$

before performing the combining operation, it is necessary to predict the label of *Urev* during the step of text classification and assume its outcome to be equal to one (i.e., positive review). Then it will be possible to perform the combining operation as the following:

$Urev = <0.62, 0.004, 0.206, 0.004, 0.102, \mathbf{1}>$

$Irev_1 = <0.003, 0.537, 0.003, 0.269, 0.139, \mathbf{1}>$

$Irev_2 = <0.48, 0.109, 0.08, 0.003, 0.267, \mathbf{0}>$

finally, calling the JSD function to calculate the degree of difference between *Urev* and $Irev_1$, $Irev_2$ respectively. Thus:

JSD (*Urev, Irev*$_1$) = 0.21

JSD (*Urev, Irev*$_2$) = 0.47

the JSD metric, previously illustrated in Algorithm 2.2, calculates the distance degree between the combined vectors of the target user reviews and the combined vectors of the target item reviews that provided in the review-topic matrix (refer to section 3.4.1(C)). After that, JSD scores are inverted to obtain the corresponding similarity degrees JSD_{sim} for each textual review written by a user on an item within the test set, hence collecting them in similarity lists as follows:

$$JSD_{sim} = 1 - JSD_{dist} \tag{3.12}$$

where JSD_{dist} is the distance value resulted from the JSD function.

Whereas the adjustment weights elicitation algorithm shown in Algorithm 3.2 is operated on the similarity lists resulting from the text similarity step. In particular, for each review written by the target user, there are two lists: one positive resulting from a comparison with positive reviews of the target item, and the other negative resulting from a comparison with negative reviews of the same item. The following equations show the process of calculating the adjustment weights:

$$PPW = \begin{cases} avg(PosList_{sim}) & , \; if \; PosList_{sim} \neq \emptyset \\ \zeta/(\#R \, . \, \zeta) \; , & otherwise \end{cases} \tag{3.13}$$

$$PLW = \begin{cases} max(PosList_{sim}) & , \; if \; PosList_{sim} \neq \emptyset \\ \zeta/(\#R \, . \, \zeta) \; , & otherwise \end{cases} \tag{3.14}$$

$$NPW = \begin{cases} avg(NegList_{sim}) & , \; if \; NegList_{sim} \neq \emptyset \\ \zeta/(\#R \, . \, \zeta) \; , & otherwise \end{cases} \tag{3.15}$$

$$NLW = \begin{cases} max(NegList_{sim}) & , \; if \; NegList_{sim} \neq \emptyset \\ \zeta/(\#R \, . \, \zeta) \; , & otherwise \end{cases} \tag{3.16}$$

where ζ is a constant parameter to avoid zero weight value, $\#R$ is the number of the available ratings (i.e., 1 and 0), Ø refers to the empty set, and PPW, PLW, NPW, NLW are positive prior, positive likelihood, negative prior, negative likelihood weights respectively. Note that the optimum value of $\zeta = 1$ according to the extensive experiments of RRMs (see section 4.2.2).

Four adjustment weights are computed for each review within the test set. Two of those weights (PPW, NPW) will contribute to calculate item priors, and the other two (PLW, NLW) will have the role in computing the likelihood of the user conditional probability within Eq. (2.5) of the Naïve Bayes model. Thus:

$$P(r_{u,i} = y) \propto \begin{cases} P(r_i = y) \,.\, PPW \prod_{j \in I_u} P(r_j = k \mid r_i = y) \,.\, PLW \;, & if\ y = 1 \\ P(r_i = y) \,.\, NPW \prod_{j \in I_u} P(r_j = k \mid r_i = y) \,.\, NLW \;, & if\ y = 0 \end{cases} \qquad (3.17)$$

Note that this step (i.e., text similarity with JSD) differs in this track (RRM_1) from the other track (RRM_2) of the proposed recommendation models.

Algorithm 3.2: Adjustment Weights Elicitation Algorithm

```
Input:
    ➢ Test Reviews TSR and Training Reviews TNR
    ➢ Setting the parameters #R, $\zeta$ according to the experiments
Output:
    ➢ Four Adjustment Weights PPW, NPW, PLW, NLW
Begin
    ➢ Preparing two similarity lists PosSimList, NegSimList
    ➢ Getting target User Ut and target Item It from the test set

        For each TSR of Ut and It
          Collect PosTNR and NegTNR        // Separate Pos/Neg Training Reviews of It

          For each TNR in PosTNR
            PosSimList ⟵ JSD (TSR, TNR)    // Call JSD function (Algorithm 2.2)
          End for TNR

          If PosSimList is not empty
            PPW ⟵ mean (PosSimList)      // Compute average of positive similarity values
            PLW ⟵ max (PosSimList)       // Pop maximum positive similarity value
          Else
            PPW ⟵ $\zeta / (\#R \times \zeta)$        // Get the default weight
            PLW ⟵ $\zeta / (\#R \times \zeta)$        // Get the default weight
          End if

          For each TNR in NegTNR
            NegSimList ⟵ JSD (TSR, TNR)   // Call JSD Function (Algorithm 2.2)
          End for TNR

          If NegSimList is not empty
            NPW ⟵ mean (NegSimList)     // Compute average of negative similarity values
            NLW ⟵ max (NegSimList)      // Pop maximum negative similarity value
          Else
            NPW ⟵ $\zeta / (\#R \times \zeta)$      // Get the default weight
            NLW ⟵ $\zeta / (\#R \times \zeta)$       // Get the default weight
          End if

        End for TSR
End.
```

E. The Step of Incorporating Adjustment Weights into the Main Equation of the Naïve Bayes Model

The last step of RRM$_1$ includes incorporating the adjustment weights deduced from the previous step to Eq. (2.5) of the Naïve Bayes model as shown above in Eq. (3.17), based on the output of the adjustment weights elicitation algorithm (refer to Algorithm 3.2).

After training the Naïve Bayes model on the training set ratings, the testing phase begins by fetching the adjustment weights. These weights are used in the testing phase along with the test set ratings to reach the most balanced classification score.

The model's performance is evaluated through the standard metrics of the recommendation task: Precision, Recall, and F1-measure after offering the Top-N of test items relevant to each test user. It is worth noting that each Top-N list arrange its items according to the highest probability of positive classifications.

F. Performance Evaluation Step Using the Metrics of Top-N Recommendation Task

The performance evaluation is the last step of the proposed review-based recommendation models (RRMs). It represents an important step to test the effectiveness of the proposed models. According to the Top-N recommendation task that accomplished in these models, measures such as (Precision, Recall, and F1-measure) as explained in section 2.8.2, are used to estimate the recommendation accuracy of RRMs.

At this stage, the evaluation is done on the test set after building the proposed model on the training set. That is, after dividing the dataset into the training and test sets with the usual ratio of 1:4. In particular, 80% of the dataset is devoted to the purpose of building and generalizing the model to be able to predict and classify test ratings which represent 20% of the ratings of each trained user. In the task of Top-N recommendation, the items of the recommended list obtained from implementing the recommendation model (RRM_1 or RRM_2) are matched with the items stored in the test set to determine the relevant ones only.

The following example depicts the performance evaluation step for the task mentioned above. Figures 2.1, 3.6, and 3.7 represent the user-item rating matrix, the training set and the test set of the rating matrix, respectively.

Now, let u_2 be selected as the target user to apply the above metrics to his/her ratings. Remember that items with a rating value greater than three, are considered positives (relevant), otherwise (i.e., rating value less than or equal to three) are considered negatives (irrelevant). After applying the recommendation metrics (Precision, Recall, and F1-measure) according to Eqs. (2.17), (2.18), and (2.19), assume the recommended items for user u_2 are assumed i_3, i_5, and i_6. Given these recommended items and the test items of u_2 shown in Figure 3.7, so i_5 and i_6 are considered as relevant to u_2.

3.4.2 The Proposed RRM without Test Reviews

It mentioned earlier that the proposed recommendation model builds on two cases, where the first (i.e., RRM_1), as previously described, depends on the presence of comments in the test set. In this subsection, the other case (i.e., RRM_2) is addressed, which includes the presence of textual reviews only in the training set. Figure 3.14 shows the steps of the recommendation model of the second track when no textual reviews are available within the test set. The difference between the two tracks of the proposed recommendation model is explained by their steps as follows:

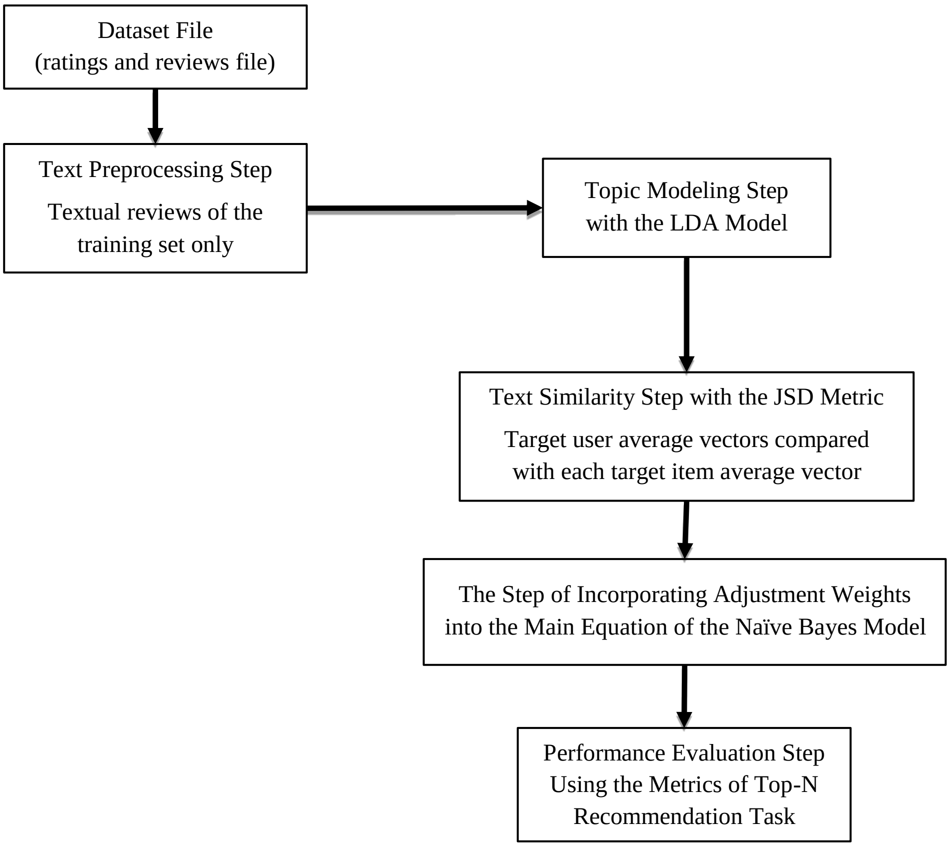

Dataset File
(ratings and reviews file)
Text Preprocessing Step
Textual reviews of the training set only
Topic Modeling Step with the LDA Model
Text Similarity Step with the JSD Metric
Target user average vectors compared with each target item average vector
The Step of Incorporating Adjustment Weights into the Main Equation of the Naïve Bayes Model
Performance Evaluation Step Using the Metrics of Top-N Recommendation Task

Figure 3.14: Steps of Recommendation Model without Test Reviews (RRM$_2$)

A. Text Preprocessing Step

This step is previously explained in section 3.4.1(A) and illustrated in Figure 3.9. The only difference in this track (i.e., RRM$_2$) from the previous one is that the textual reviews are excluded from the test set to check the accuracy of the model's performance if no user feedback was available.

B. Topic Modeling Step with the LDA Model

To remind, the text classification step is not required in RRM$_2$ because the textual reviews of the test set are excluded from the model process. Thus, the topic modeling step that utilizes the LDA model is implemented only on the textual reviews of the training set in this model, as explained earlier in section 3.4.1(C) and illustrated in Figure 3.11.

C. Text Similarity Step with the JSD Metric

This step is previously clarified in section 3.4.1(D) and shown in Figure 3.12 and Algorithms 2.2 and 3.2, but the difference of this step in RRM_2 is the absence of the test set reviews. Therefore, the essential difference between the two counterparts (i.e., text similarity step in RRM_1 and RRM_2) is only in the parties of the text similarity process. Earlier, in RRM_1 the test review (combined vector) of the target user is compared with each of the target item's positive/negative training reviews (combined vectors) to derive positive/negative adjustment weights. As for this step here in RRM_2, the positive/negative training reviews of the target user is collected separately and then compared as combined vectors with each of the positive and negative training reviews of the target item, respectively, for the same purpose (i.e., derive positive/negative adjustment weights).

The result of the collection process of positive/negative training reviews of the target user is saved only in two vectors (positive and negative). Where the positive vector represents the average topic probability distributions of all positive reviews, while the negative vector represents the average topic probability distributions of all negative reviews, as shown in the following example:

assume the target user has four textual reviews represented by the following topic probability vectors:

$+Urev_1 = <0.62, 0.004, 0.206, 0.004, 0.102, \mathbf{1}>$

$+Urev_2 = <0.003, 0.537, 0.003, 0.269, 0.139, \mathbf{1}>$

$-Urev_3 = <0.48, 0.109, 0.08, 0.003, 0.267, \mathbf{0}>$

$-Urev_4 = <0.36, 0.15, 0.08, 0.003, 0.329, \mathbf{0}>$

thus, the averaged vectors are:

$+Uavg = <0.312, 0.271, 0.105, 0.136, 0.121, \mathbf{1}>$

$-Uavg = <0.42, 0.13, 0.08, 0.003, 0.248, \mathbf{0}>$

let the target item have the following two vectors:

$Irev_1 = <0.004, 0.628, 0.002, 0.35, 0.24, \mathbf{1}>$

$Irev_2 = <0.57, 0.218, 0.07, 0.004, 0.358, \mathbf{0}>$

thereby, the positives vectors and the negative ones are compared separately. Thus:

JSD ($+Uavg, Irev_1$) $= 0.73$

JSD ($-Uavg, Irev_2$) $= 0.62$

D. The Step of Incorporating Adjustment Weights into the Main Equation of the Naïve Bayes Model

The adjustment weights elicitation algorithm performed in RRM$_1$ shown in Algorithm 3.2 is repeated here to obtain the four adjustment weights. Finally, as a reminder, the last step of this track (RRM$_2$) represented by incorporating adjustment weights to the Naïve Bayes model is a similar copy of its counterpart in the previous track (RRM$_1$) of the proposed model (refer to section 3.4.1(E)).

The RRM$_2$ performance is evaluated as shown in section 3.4.1(F), through the standard metrics of the recommendation task: Precision, Recall, and F1-measure after offering the Top-N of test items relevant to each test user.

4.1 Overview

Discussion of the results of extensive experiments will be displayed in this chapter for the following proposed approaches:

1) Trust-based Prediction Model (TPM)
2) Review-based Recommendation Models (RRMs)

The experiments of the proposed system are presented in section 4.2 with the above two approaches. The prediction model has been tested on two real-world datasets: FilmTrust and Epinions in subsection 4.2.1. As for Top-N recommendation, RRMs are conducted on three categories of Amazon datasets which are Musical Instruments (MI), Automotive (AM), and Amazon Instant Video (AIV) in subsection 4.2.2, in addition to the experiments related to DistilBERT and LDA models respectively. All five datasets are previously described in chapter three (refer to section 3.2). All models' performance has been compared with similar recent RS algorithms implemented on the same datasets. It should be noted that two F-measures are used for each task of RS. The first F-measure obtained from the weighted average between accuracy and coverage of the prediction, was employed for the rating prediction task. The other (i.e., F1-measure), on the other hand, obtained from the weighted average between Precision and Recall metrics, was used for the Top-N recommendation task.

4.2 Proposed System Experiments

The proposed models have been managed in two cases regarding utilizing the additional information as implicit feedback. Trust relations are used as an implicit extra factor besides explicit numerical ratings to mitigate the sparsity problem of the rating matrix in the TPM.

Differently, Textual reviews are utilized as an implicit additional component in addition to the explicit numerical ratings to balance the natural bias toward positive ratings (i.e., class imbalance) in the RRMs.

4.2.1 Trust-based Prediction Model Experiments

A trust-based prediction model (TPM) has been proposed for the rating prediction task by utilizing the propagation property of trust relations and adopting the weighted voting technique to mitigate the rating matrix’s sparsity. This model is tested on two datasets (FilmTrust and Epinions). It is important to say that the model also utilizes explicit ratings in its training process. Thus, the prediction model depends on the original ratings already available in the rating matrix as explicit feedback and trust-elected ratings as implicit feedback inferred from trustworthy neighbors.

For evaluation of the prediction accuracy of the proposed model, three methods (Merge [21], EIMerge [22], ITRA [4]), that utilized the merge mechanism of ratings in their implementation are selected to be the baselines methods for comparison with the proposed model. In addition, some values that are not provided in their experiments are calculated, such as the value of the F-measure between accuracy and coverage of prediction.

Extensive experiments on the used datasets are performed to show the proposed model’s efficiency compared with the other counterparts. These experiments are conducted on a 64-bit OS Windows 10 Pro, Intel® Core™ i3 3120M CPU 2.50 GHz, 4.00 GB of RAM (3.82 GB usable). So, a representative sample from the large-scale Epinions dataset is taken due to RAM limitations. The proposed TPM has been applied to FilmTrust and Epinions datasets. In the following, TPM results will be exhibited and discussed on both datasets, respectively.

a) Experiments on FilmTrust dataset

For the FilmTrust dataset, Table 4.1 depicts the results of the proposed model's 5-fold cross-validation process, besides the optimal weights representing the best experiment within each fold. Table 4.2 portrays the MAE, RMSE, RC, and F-measure values of the comparison methods against the average values of the best trials of the proposed model, in addition to the improvement ratios of the TPM over the comparison methods.

Table 4.1: The Best Values of each Fold on FilmTrust (TPM)

#Folds	Optimal Weight	Metrics			
		MAE	RMSE	RC (%)	F-measure
1	0.9	0.42211	0.59245	96.41	0.91980
2	0.8	0.42570	0.60347	96.18	0.91821
3	1.0	0.42052	0.59421	**96.75**	**0.92158**
4	1.0	0.41378	0.58515	95.99	0.91916
5	0.9	**0.41276**	**0.58030**	96.20	0.92029

Based on the outcomes stated in Table 4.1, which are not significantly different for all five folds, this justifies the ability of the proposed model to generalize over various test sets.

Table 4.2: Averages Values of TPM Against the Comparison Methods and The Improvement Ratio in terms of F-measure on FilmTrust

Metrics	Methods			
	Merge [21]	EIMerge [22]	ITRA [4]	TPM
MAE	0.708	0.6819	0.9050	**0.41897**
RMSE	N/A	N/A	1.0756	**0.59112**
RC (%)	95.06	95.23	69.24	**96.31**
F-measure	0.8674	0.8726	0.7160	**0.91981**
Improvement Ratio (F-measure)	**5.24**	**4.72**	**20.38**	**-**

For all previously mentioned comparison methods, the available results are recorded on FilmTrust as found in their respective papers and illustrated in Table 4.2. In particular, the ITRA model was the worst on FilmTrust in terms of all metrics. The TPM and ITRA were the only two models that calculated the RMSE metric in their experiments, accordingly, the performance of the proposed model (TPM) significantly outperforms the ITRA model on FilmTrust in terms of RMSE and other metrics also.

EIMerge method was tested only on FilmTrust and achieved better MAE value relative to the others, however, it ranked behind the proposed TPM as the second-best. Similarly, its rating coverage is high among all and very close to the Merge method, but also it came behind the proposed model in terms of RC metric. Uniquely, any prediction available obtained from both the original and trust-elected ratings is leveraged, as shown in Eq. (3.10), which is the reason for the best prediction coverage of the proposed TPM (see Figure 4.1).

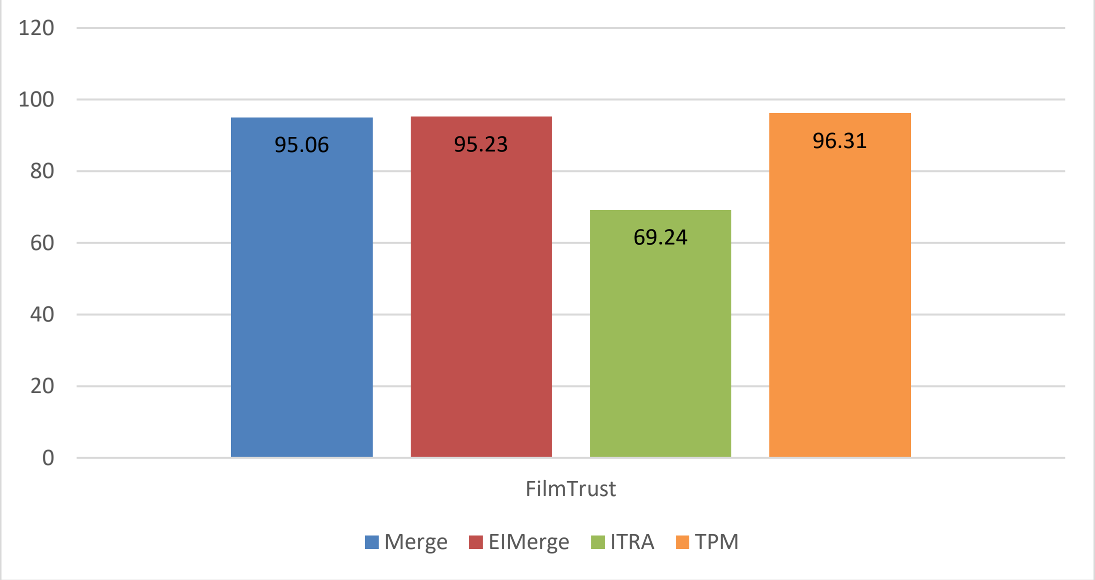


Figure 4.1: The Rating Prediction Coverage Ratios on FilmTrust

According to the results shown in Table 4.2, the proposed TPM has superior prediction outcomes on the FilmTrust dataset. For the sake of comparison, the F-measure value of the ITRA model is calculated due to its unavailability in its research. For a better look at the overall improvement that the proposed TPM achieves, the improvement ratios are calculated and compared with the other methods in terms of the F-measure, as illustrated in the last row of Table 4.2. So, the more positive the difference between the proposed model and the others, the more improvements are achieved.

The reasons for the significant improvement of the proposed TPM against its counterparts are eliciting implicit trust relations and adopting the property of trust propagation, both considerably helped reduce the sparsity of the rating matrix. In addition, conducting the idea of the rating election by the weighted voting technique that elects ratings of trustworthy neighbors to enrich users' preference profiles, greatly assists in decreasing the noise of the weighted average technique used mainly in CF methods. Finally, the mixing process of original and trust-elected ratings has a core role in contributing positively by extending the prediction coverage and increasing prediction accuracy, due to how Eq. (3.10) is formulated, which takes the available results of two methods to compute the final prediction.

b) Experiments on Epinions dataset

For the Epinions dataset, Table 4.3 shows the outcomes of the 5-fold cross-validation process performed by the proposed TPM besides the optimal weights representing the best experiment within each fold. Table 4.4 displays all four metrics' values of the comparison methods (Merge and ITRA) versus the average values of the best trials of the proposed model, in addition to the improvement ratios of the TPM over the comparison methods.

Table 4.3: The Best Values of each Fold on Epinions (TPM)

#Folds	Optimal Weight	Metrics			
		MAE	RMSE	RC (%)	F-measure
1	1.0	0.61811	**0.83628**	97.78	**0.90683**
2	1.0	0.62860	0.86454	**97.82**	0.90548
3	1.0	0.62586	0.86261	97.43	0.90420
4	1.0	0.62257	0.85885	97.69	0.90579
5	1.0	**0.61770**	0.84964	97.69	0.90649

The values stated in Table 4.3 indicate relatively close results for all the five-fold trials, thus asserting the proposed TPM generalization capability on different test sets.

Table 4.4: Averages Values TPM Against the Comparison Methods and The Improvement Ratio in terms of F-measure on Epinions

Metrics	Methods		
	Merge [21]	ITRA [4]	TPM
MAE	0.820	0.7181	**0.62257**
RMSE	N/A	**0.8200**	0.85438
RC (%)	80.02	86.98	**97.68**
F-measure	0.7976	0.8444	**0.90576**
Improvement Ratio (F-measure)	**10.82**	**6.14**	**-**

For all previously mentioned comparison methods, the available results are recorded on Epinions as found in their respective papers and illustrated in Table 4.4. Note that this table contained only Merge and ITRA as the comparison methods because the EIMerge method was not tested on the Epinions dataset. The Merge method had the worst performance based on only MAE and RC (because RMSE was not included in its evaluation). Again, the TPM and ITRA were the only two models that considered the RMSE metric in their experiments, accordingly, the performance of ITRA slightly outperforms the proposed model (TPM) on Epinions unlike on the previous data set (FilmTrust).

The reason of this is that the proposed model achieved an RC value more considerable than the one achieved by the ITRA model. Therefore, the proposed model is better regarding F-measure, as shown in Table 4.4. So, the performance was competitive between the two models on both datasets. Leveraging any prediction available obtained from both the original and trust-elected ratings as formulated in Eq. (3.10) is why the proposed TPM has the best prediction coverage (see Figure 4.2).

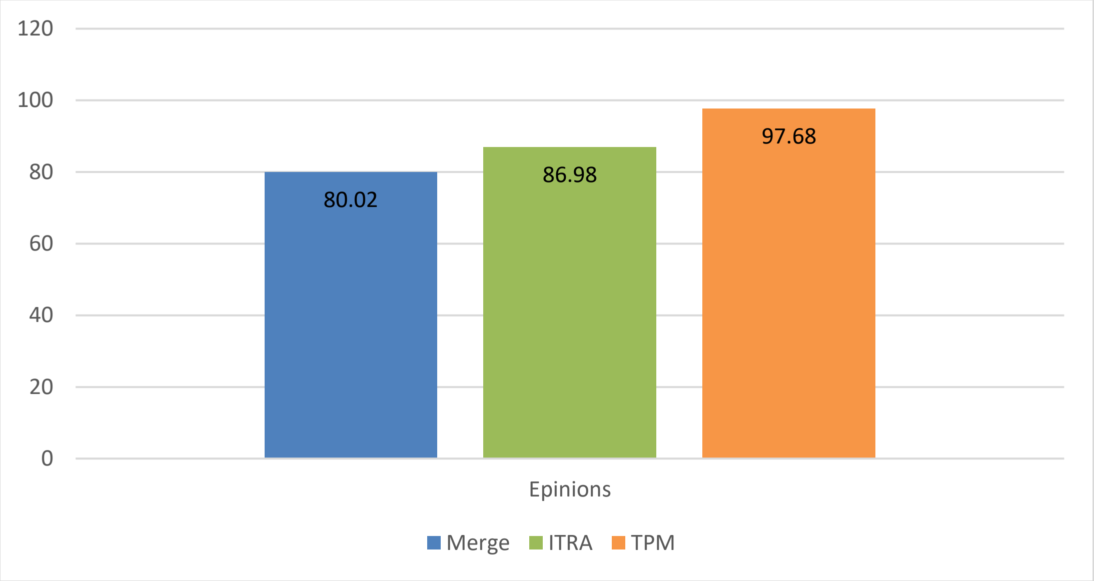


Figure 4.2: The Rating Prediction Coverage Ratios on Epinions

According to the upshots shown in Table 4.4, the proposed TPM accomplish the best estimation results on the Epinions dataset. To compare, the value of the F-measure of the ITRA model is computed due to it not being recorded in its research. Additionally, the improvement ratios that the proposed TPM achieves against the comparison methods are summed up in terms of the F-measure, as shown in the last row of Table 4.4. That is a good insight into the overall improvement that the proposed model achieved.

The notion of the rating election by the weighted voting technique, the propagation property of trust theory, inferring implicit trust relations, and the mixing of original and trust-elected ratings all together have contributed to the high improvements of the proposed TPM over the comparison methods.

c) Final prediction result based on the Contribution Weight

The main step for obtaining the final rating prediction of the TPM is to calculate the results of two methods. By depending on the nearest neighbors with their similarity values of the original ratings obtained from Eq. (2.6) and the trust-elected ratings obtained from Eq. (2.9). Hence, both results are separately included in Eq. (3.9), then incorporating an essential parameter (i.e., contribution weight), which is used in the linear combination process of both methods (refer to Eq. (3.10)) to obtain the final prediction.

This work includes examining the impact of mixing the original and trust-elected ratings on CF prediction results. Accordingly, a contribution weight is involved as its values ranged from 0 to 1 in increments of 0.1 on both FilmTrust and Epinions in terms of F-measure and RMSE metric, as shown in Figures 4.3 and 4.4, respectively.

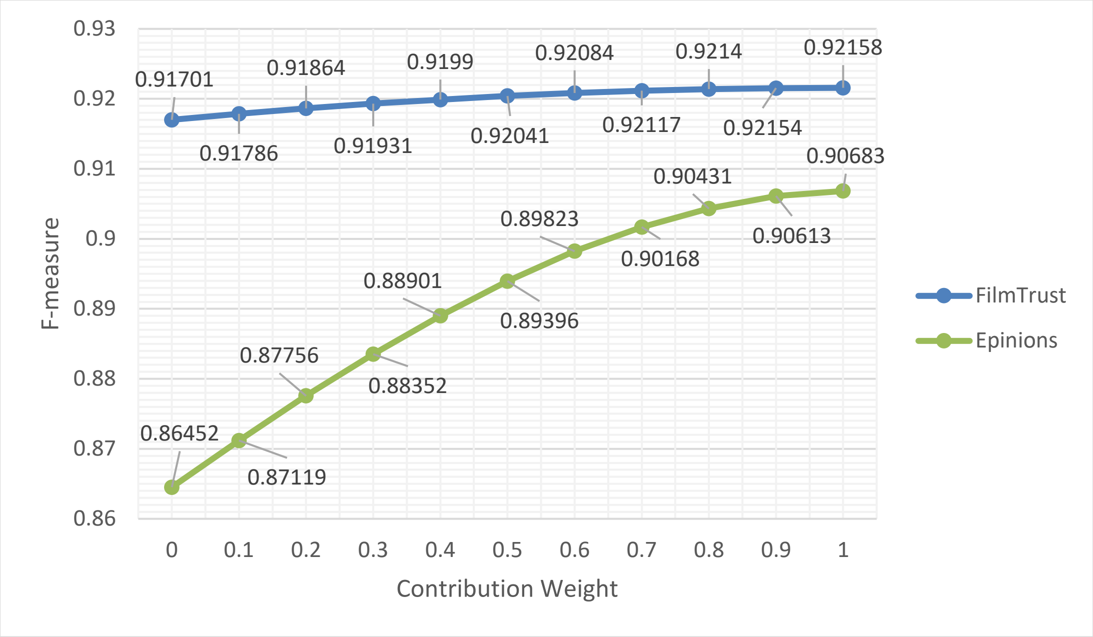


Figure 4.3: The Contribution Weight in terms of F-measure on FilmTrust and Epinions

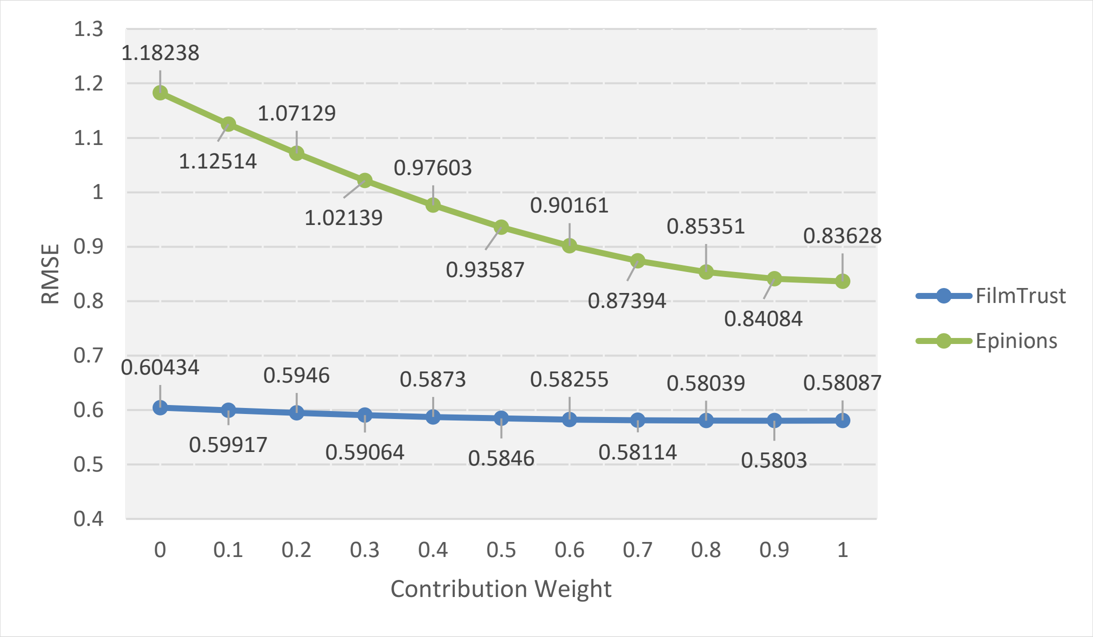


Figure 4.4: The Contribution Weight in terms of RMSE on FilmTrust and Epinions

The experiments concluded that the optimal value of the contribution weight is either 0.8, 0.9, or 1 on both datasets. That means the more prominent this weight, the more accurate the results and vice versa. In this situation, the outcome of the final prediction will depend, to a large extent, on the results of the original rating method. According to Eq. (3.10), if the score of the original rating method cannot be predicted, the trust-elected rating method's score is used as an alternate. Similarly, this occurs in reverse when the score of the trust-elected rating method cannot be estimated.

The final prediction result follows the same pattern when applied to both FilmTrust and Epinions. In terms of all measures (MAE, RMSE, RC, and F-measure), the best result occurred when the contribution weight value was high (i.e., 0.8, 0.9, or 1). Thus, the prediction coverage is utterly stable because of how Eq. (3.10) is formulated, which takes the available results of two methods to calculate the final prediction values.

4.2.2 Review-based Recommendation Models Experiments

Two review-based recommendation models (RRMs) have been proposed for the Top-N recommendation task based on whether to incorporate the textual reviews of the test set in inferring the adjustment weights: namely, RRM with test reviews and RRM without test reviews. These models are carried out on three Amazon datasets (i.e., MI, AM, and AIV). Intentionally, the test set reviews are excluded for the second model (RRM without test reviews) to check the proposed model capability. Accordingly, RRMs have been implemented as RRM_1 on the textual reviews of both training and test sets and RRM_2 only on the training set reviews. It is important to say that both models also utilize explicit ratings in their training process. Thus, in such models, the recommendation engine depended on the numerical rating as explicit feedback and the adjustment weights inferred from the textual reviews as implicit feedback.

For evaluation and comparison purposes, several recent works [70]–[77] are surveyed that used the same datasets (MI, AM, AIV) and then their results are recorded to be compared to the proposed models' results. In addition, some values that were not available in these works are computed, such as the value of the F1-measure.

I. RRMs Experiments

As the textual reviews are available in the test set, the first RRM (with test reviews) has been applied to both training and test reviews of MI, AM, and AIV datasets. Since the textual reviews are intentionally excluded from the test set, the second RRM (without test reviews) has been only conducted on the training reviews for the same datasets. It is worth noting that the following results were calculated when setting $\alpha = 1$ as the best value to avoid zero probabilities, as indicated previously in Eqs. (2.3) and (2.4). At the same, ζ also set equal to 1 (i.e., $\zeta = 1$) as the finest value to avoid zero weights, as pointed out previously in Eqs. (3.13), (3.14), (3.15), and (3.16). Next, RRMs results will be presented and discussed in the following subsections on the three datasets, respectively.

a) Experiments on Musical Instruments (MI) dataset

For the MI dataset, Tables 4.5 and 4.6 illustrate the upshots of the 5-fold cross-validation obtained by the two proposed RRMs. Table 4.7 displays the Precision and Recall values for the comparison methods against the average values of the proposed models. F1-measure values are added to Table 4.8 for exhibiting all three metrics of the proposed models versus their counterparts, in addition to the improvement ratios presented regarding the F1-measure.

The P@N, R@N, and F1@N in the tables below mean Precision, Recall, and F1 values are computed when the number of items is no more than N in the recommended lists.

Table 4.5: The Best Values of each Fold on MI (RRM$_1$)

#Folds	N	Metrics		
		P@N	R@N	F1@N
1	8	0.86794	0.89661	0.88204
2	8	0.86217	0.88418	0.87304
3	8	**0.88767**	**0.90661**	**0.89704**
4	9	0.87064	0.89128	0.88084
5	9	0.86801	0.89035	0.87904

Table 4.6: The Best Values of each Fold on MI (RRM$_2$)

#Folds	N	Metrics		
		P@N	R@N	F1@N
1	8	0.86579	0.89410	0.87971
2	8	0.86711	0.89903	0.88279
3	8	**0.87903**	**0.90401**	**0.89135**
4	9	0.87039	0.89468	0.88237
5	9	0.86002	0.88684	0.87323

Based on the output values revealed in Tables 4.5 and 4.6, the outcomes of the five folds are reasonably close to each other in each experiment, which explains the generalization ability of the proposed RRMs over every test partition from the MI dataset.

Table 4.7: Averages Values of RRMs Against the Comparison Methods on MI Dataset in terms of Precision and Recall

Metrics	Methods				
	RBP [70]	AODR-MCNN [71]	ADRS [73]	RRM$_1$	RRM$_2$
P@N	N/A	0.0989	0.0904	**0.87129**	0.86847
R@N	0.5788	N/A	N/A	0.89381	**0.89573**

Table 4.8: Averages Values of RRMs Against the Comparison Methods in terms of all Metrics and The Improvement Ratios in terms of F1-measure on MI Dataset

Metrics	Methods

	IRGNN [74]	N2VSCDNNR [75]	ASCF [76]	RRM$_1$	RRM$_2$
P@N	0.7977	0.7792	0.2235	**0.87129**	0.86847
R@N	0.7586	0.8145	0.4123	0.89381	**0.89573**
F1@N	0.7777	0.7965	0.2899	**0.88240**	0.88189
Improvement Ratio of RRM$_1$ (F1@N)	**10.47**	**8.59**	**59.25**	**-**	-
Improvement Ratio of RRM$_2$ (F1@N)	**10.42**	**8.54**	**59.20**	**-**	-

As shown in Table 4.7, massive progress in results has been calculated using the proposed RRMs in terms of both Precision and Recall. Similarly, in Table 4.8, the outcomes are superior to other methods in all three primary metrics, as the F1-measure is appended to the table. In particular, AODR-MCNN, ADRS, and ASCF methods obtained the worst outcome regarding Precision and Recall. Some other methods, such as IRGNN and N2VSCDNNR, did not record the F1 value, so it is required to calculate it. Accordingly, their performance was competitive. In specific, IRGNN has better Precision value compared with all methods except the proposed RRMs. The reason of this is that combining the polarity and topic probabilities of the text is leveraged to obtain better text semantic similarity, thus, better recommendation accuracy. Resulting in the proposed models outperforming all methods mentioned above in terms of all metrics. Notably, N/A refer to the unavailability of the metric’s score in the respective research papers.

According to the last two rows of Table 4.8, the improvement ratios that the proposed RRMs achieve compared with the other methods in terms of F1-measure are spectacular. Where the more positive the ratios, the more improvement are achieved. The significant improvement of RRMs is because choosing the best-advanced techniques for processing texts, besides utilizing the classic classification model (i.e., Naïve Bayes classifier) after incorporating the implicit inferences (i.e., adjustment weights) in its central equation.

b) Experiments on Automotive (AM) dataset

For the AM dataset, Tables 4.9 and 4.10 depict the outcomes of the 5-fold cross-validation performed by the proposed RRMs, respectively. Table 4.11 shows the comparison methods' Precision and Recall values against the proposed models' average values. F1-measure values are appended to Table 4.12 for exhibiting all three metrics of the proposed model versus other methods, in addition to the improvement ratios presented in terms of F1-measure.

The P@N, R@N, and F1@N in the tables below mean Precision, Recall, and F1 values are computed when the number of items is no more than N in the recommended lists.

Table 4.9: The Best Values of each Fold on AM (RRM$_1$)

#Folds	N	Metrics		
		P@N	R@N	F1@N
1	10	0.87175	0.89170	0.88161
2	10	0.86799	0.88732	0.87755
3	10	0.87146	**0.89435**	0.88276
4	10	0.86021	0.88035	0.87016
5	11	**0.87396**	0.89424	**0.88398**

Table 4.10: The Best Values of each Fold on AM (RRM$_2$)

#Folds	N	Metrics		
		P@N	R@N	F1@N
1	9	0.86502	0.89130	0.87796
2	9	0.86477	0.88559	0.87506
3	10	0.86421	0.89344	0.87858
4	10	0.85809	0.88351	0.87062
5	11	**0.86859**	**0.89356**	**0.88090**

Based on the outcomes presented in Tables 4.9 and 4.10, the results of the five folds are pretty close, which explains the generalization ability of the proposed RRMs over different test sets.

Table 4.11: Averages Values of RRMs Against the Comparison Methods on AM Dataset in terms of Precision and Recall

Metrics	Methods			
	RBP [70]	AODR-MCNN [71]	RRM_1	RRM_2
P@N	N/A	0.1075	**0.86907**	0.86414
R@N	0.3327	N/A	**0.88959**	0.88948

Table 4.12: Averages Values of RRMs Against the Comparison Methods in terms of all Metrics and The Improvement Ratios in terms of F1-measure on AM Dataset

Metrics	Methods		
	RUM [72]	RRM_1	RRM_2
P@N	0.842	**0.86907**	0.86414
R@N	0.501	**0.88959**	0.88948
F1@N	0.622	**0.87921**	0.87662
Improvement Ratio of RRM_1 (F1@N)	**25.72**	**-**	-
Improvement Ratio of RRM_2 (F1@N)	**25.46**	**-**	-

As shown in Table 4.11, tremendous progress in results has been calculated using the proposed RRMs in terms of both Precision and Recall. Similarly, in Table 4.12, the outcomes of the proposed models are better than the other method in terms of all three metrics. In specific, AODR-MCNN and RBP held the worst performance at Precision and Recall values, respectively. Uniquely, RUM is the only method that calculated all metrics in its experiment. Still, it places as the second-best behind the proposed RRMs. Indeed, tackling the natural bias toward positive ratings (i.e., class imbalance) had the role of surpassing the proposed models to its counterparts. Note, N/A refer to an unavailable score.

The last two rows of Table 4.12 show that the proposed models (RRM_1, RRM_2) achieve improvement ratios compared with the RUM method is better by about 25% in terms of the F1-measure. The remarkable improvement is because utilizing the Naïve Bayes model after incorporating the adjustment weights in its main equation, besides choosing the best-advanced techniques for preprocessing texts.

c) Experiments on Amazon Instant Video (AIV) dataset

For the AIV dataset, Tables 4.13 and 4.14 show the results of the 5-fold cross-validation achieved by the proposed RRMs, respectively. Table 4.15 illustrates the Precision and Recall values for the comparison methods against the average values of the proposed models. F1-measure values are supplied in Table 4.16 for exhibiting all three metrics of the proposed models versus other methods that recorded them, in addition to the improvement ratios offered concerning the F1-measure.

The P@N, R@N, and F1@N in the tables below mean Precision, Recall, and F1 values are computed when the number of items is no more than N in the recommended lists.

Table 4.13: The Best Values of each Fold on AIV (RRM_1)

#Folds	N	Metrics		
		P@N	R@N	F1@N
1	10	0.80772	0.82516	0.81635
2	10	0.77730	0.79625	0.78666
3	10	**0.81089**	**0.83427**	**0.82241**
4	10	0.80820	0.83124	0.81956
5	9	0.79969	0.82089	0.81015

Table 4.14: The Best Values of each Fold on AIV (RRM_2)

#Folds	N	Metrics		
		P@N	R@N	F1@N

1	10	**0.78665**	**0.80844**	**0.79739**
2	10	0.77071	0.79663	0.78346
3	10	0.78179	0.80581	0.79362
4	10	0.78291	0.80759	0.79506
5	10	0.77516	0.79803	0.78643

Based on the values displayed in Tables 4.13 and 4.14, the results of the five folds are reasonably close, which explains the generalization ability of the proposed RRMs over various test partitions of the AIV dataset.

Table 4.15: Averages Values of RRMs Against the Comparison Methods on AIV Dataset in terms of Precision and Recall

Metrics	Methods				
	RBP [70]	AODR-MCNN [71]	ADRS [73]	RRM$_1$	RRM$_2$
P@N	N/A	0.1282	0.0928	**0.80076**	0.77944
R@N	0.7369	N/A	N/A	**0.82156**	0.80330

Table 4.16: Averages Values of RRMs Against the Comparison Methods in terms of all Metrics and The Improvement Ratios in terms of F1-measure on AIV Dataset

Metrics	Methods		
	GRU-BFGS [77]	RRM$_1$	RRM$_2$
P@N	0.60	**0.80076**	0.77944
R@N	0.59	**0.82156**	0.80330
F1@N	0.5950	**0.81103**	0.79119
Improvement Ratio of RRM$_1$ (F1@N)	**21.60**	**-**	-
Improvement Ratio of RRM$_2$ (F1@N)	**19.62**	**-**	-

As shown in Table 4.15, enormous progress in results has been computed using the proposed RRMs in terms of both Precision and Recall. Similarly, in Table 4.16, the outcomes of the proposed models are competitive with the other method in terms of all three metrics. Regarding Precision and Recall values, the worst methods were ADRS, AODR-MCNN, and GRU-BFGS, with poor outcomes. Differently, the RBP method has a better Recall value on this dataset (i.e., AIV). Again, the proposed RRMs have the best performance on the third consecutive dataset. Sure, the involvement of the textual reviews within the proposed models had a significant role in weightage of the recommendation accuracy over other counterparts. In addition, choosing the best-advanced techniques for analyzing and modeling texts and measuring their similarity, besides utilizing the classic Naïve Bayes classifier, had the most noticeable impact on the dominance of the proposed models compared to other methods. Notably, N/A refer to an unavailable score.

According to the last two rows of Table 4.16, the improvement ratios that the proposed models ($RRM_{1,}$ RRM_2) achieve compared with the GRU-BFGS method in F1-measure are reasonable at about 21% and 19% for each model respectively. The significant improvement is because selecting the best-advanced techniques for preprocessing the textual reviews, and then incorporating their implicit inferences (adjustment weights) in the main formula of the Naïve Bayes model. Again, for comparison purposes, the F1 value is calculated for GRU-BFGS method because of unavailability in its study.

II. DistilBERT Experiments

Recalling section 3.4.1(B), in which the step of classifying the text using the DistilBERT model is explained. In this section, the experiment that was conducted to tune the DistilBERT model as a classifier for test set reviews of the RRM_1 will be presented.

As mentioned earlier that the DistilBERT model works on a fixed length of texts. Therefore, the optimal value that the classification model settled on, is selected to be equal to 250 as the maximum fixed length of text. This is because of the various word distributions of reviews among the used datasets and the limitation of the practical environment. So, any extra token will be ignored automatically in the training process.

Preparing textual reviews follows setting the basic parameters to fit the model within its training process, as previously shown in Figure 3.10. As mentioned earlier, the DistilBERT model is tuned on several parameters. These are a sequence classification, a sparse categorical cross entropy as loss function, and an Adam optimizer with the best-tuned values of batch size, number of epochs, learning rate, epsilon value, and the maximum length of text. In order to clarify the model fitting process, a unique symbol will refer to each collection of parameter values as in the following plots (Figures 4.5, 4.6, and 4.7) on each MI, AM, and AIV datasets, respectively. This symbol will represent the horizontal axis, and the vertical axis of each plot will represent the model's classification accuracy. Table 4.17 shows the values of the parameters corresponding to each symbol in the related plots.

Table 4.17: The Symbols of Fitting Parameters

Symbol	Meaning
a	MAX_LEN = 250, BATCH_SIZE = 16, N_EPOCHS = 2, LR = 5e-05, EPS = 1e-08
b	MAX_LEN = 250, BATCH_SIZE = 16, N_EPOCHS = 2, LR = 5e-05, EPS = 1e-09
c	MAX_LEN = 250, BATCH_SIZE = 16, N_EPOCHS = 2, LR = 6e-05, EPS = 1e-08
d	MAX_LEN = 250, BATCH_SIZE = 16, N_EPOCHS = 2, LR = 6e-05, EPS = 1e-09
e	MAX_LEN = 250, BATCH_SIZE = 16, N_EPOCHS = 3, LR = 5e-05, EPS = 1e-08
f	MAX_LEN = 250, BATCH_SIZE = 16, N_EPOCHS = 3, LR = 5e-05, EPS = 1e-09
g	MAX_LEN = 250, BATCH_SIZE = 16, N_EPOCHS = 3, LR = 6e-05, EPS = 1e-08
h	MAX_LEN = 250, BATCH_SIZE = 16, N_EPOCHS = 3, LR = 6e-05, EPS = 1e-09

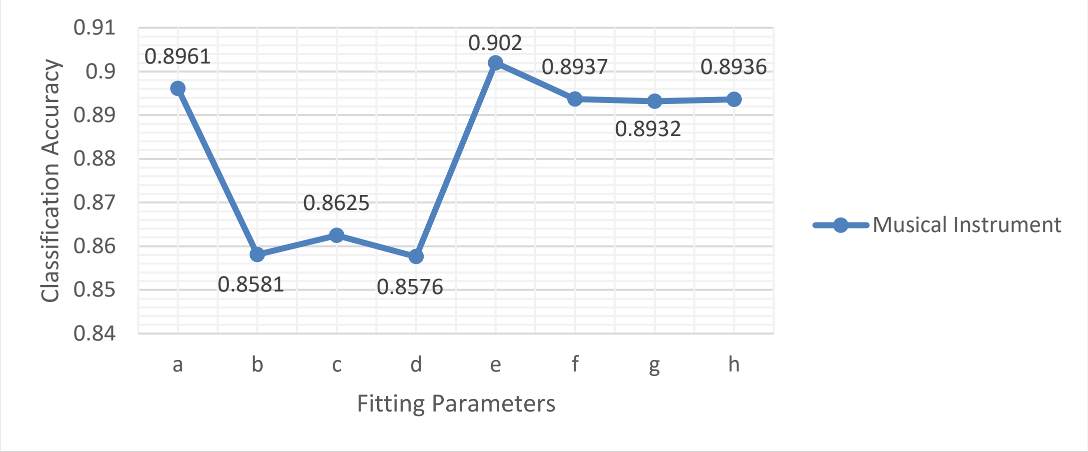


Figure 4.5: DistilBERT Classification Accuracy on MI Dataset

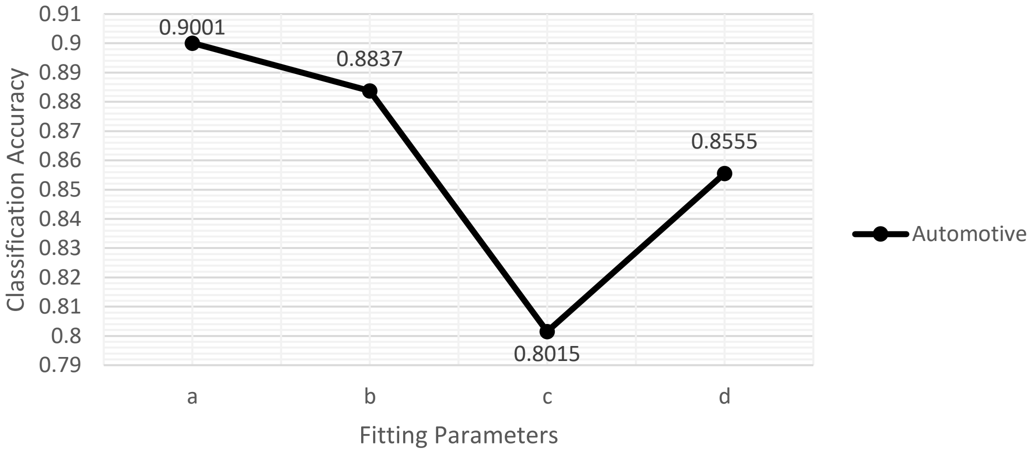


Figure 4.6: DistilBERT Classification Accuracy on AM Dataset

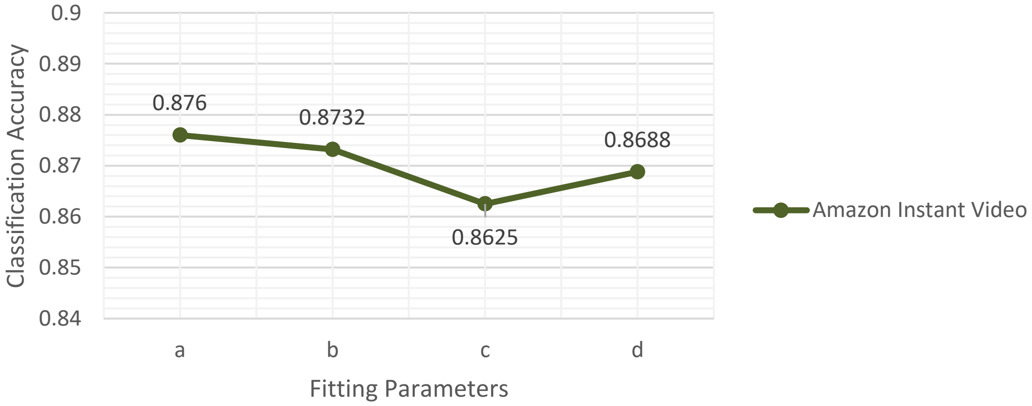


Figure 4.7: DistilBERT Classification Accuracy on AIV Dataset

Finally, after fitting the model with various parameters on the training set reviews and checking its classification accuracy on the test set reviews, the best-parametrized model will be employed in the remaining steps of RRM$_1$ when the evaluation is satisfied.

III. LDA Experiments

In this section, the experiment of choosing the best model for extracting the latent topic probability distributions based on several K topics will be presented, as mentioned earlier in section 3.4.1(C). The step of topic modeling using the LDA model is essential in this work, as the model steps are previously shown in Figure 3.11. To remember, the LDA model is conducted on the tokenized copy of the textual reviews to build the topic-review matrix for training other reviews to generate their topic-review vectors for the required reviews (i.e., test set reviews in case of RRM$_1$). To ensure the best probability distributions of topics related to each review, the LDA model is tested with a different number of K topics ranging from 5 to 50, as shown in Figures 4.8, 4.9, and 4.10 for each MI, AM, and AIV datasets, respectively.

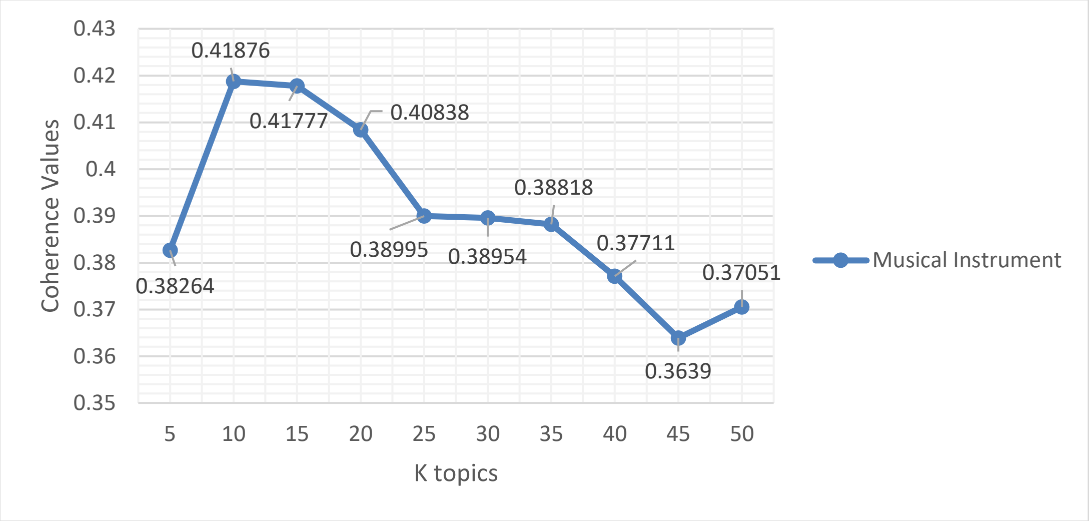


Figure 4.8: LDA Coherence Values on MI Dataset

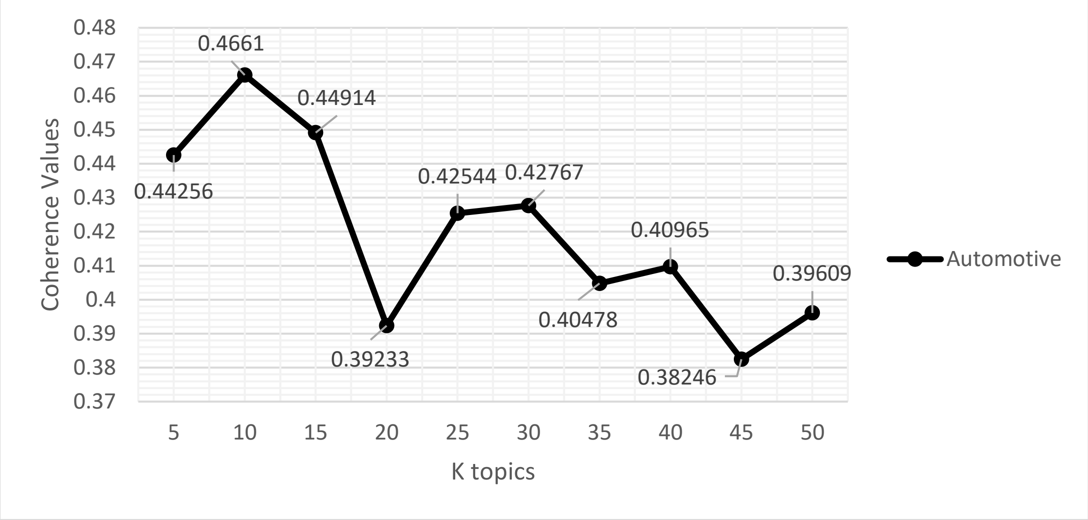


Figure 4.9: LDA Coherence Values on AM Dataset

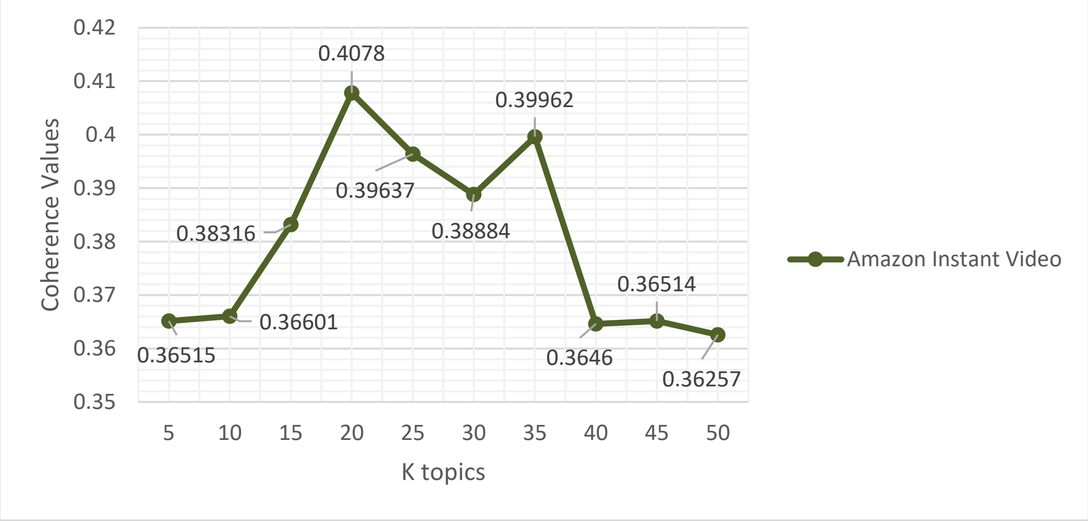


Figure 4.10: LDA Coherence Values on AIV Dataset

The optimal model will be selected from the candidate models according to the maximum coherence value. Thus, this model will to be used in the subsequent steps of RRM_1 and RRM_2. According to Figure 4.8, the maximum coherence value intersects with ten topics, thus, the model with ten topics on a MI dataset is picked as the optimal model. Similarly, the same procedure is applied to other datasets. In particular, Figures 4.9 and 4.10 show that the models with 10 and 20 topics are chosen as the optimal models on AM and AIV datasets, respectively.

5.1 Conclusions

This section presents the conclusions arrived at in this thesis. After that, section 5.2 points out future work.

In the following, the conclusions of the proposed models are presented separately. The conclusion related to the trust-based prediction model (TPM) will be demonstrated first, and then the ones related to the review-based recommendation models (RRMs). The following main conclusions will be presented based on the results obtained from the corresponding models on related datasets.

In the first proposed model (TPM), the advantage of some successful functions used in the latest CF methods is taken. The idea of the rating election by weighted voting is derived from ensemble classifiers and implemented to improve prediction accuracy and increase coverage in the proposed model. In particular, regarding the TPM, the following conclusions have been reached:

1) The process of inferring implicit trust relations and adopting the trust propagation property has considerably helped reduce the sparsity problem of the user-item rating matrix.
2) The idea of the rating election by the weighted voting technique, which involves electing the ratings of trustworthy neighbors to enrich users' preference profiles, has greatly assisted in decreasing the noise of the weighted average technique used mainly in CF methods. Additionally, calculating the reliability of the elected ratings with RPCC has led to more balanced results.
3) Concerning the impact of mixing original and trust-elected ratings on the accuracy and coverage of the prediction, it has found that such a combination contributes positively to extend the prediction

coverage and increases prediction accuracy. This is due to the way Eq. (3.10) is formulated, which makes use of the available results of two methods to compute the final prediction.

4) The extensive experiments on two real-world datasets, showed that the proposed TPM outperformed all comparison methods in terms of prediction coverage and accuracy.

In the second proposed model (RRM), however, the most advanced techniques for analyzing and modeling text besides text similarity metrics and utilizing the Naïve Bayes classifier through carrying out the two tracks of the proposed model, are employed to improve their recommendation accuracy. In specific, regarding the RRMs, the following conclusions have been drawn:

1) Combining the polarity and topic probabilities of the text greatly assisted in obtaining better text semantic similarity, and hence a better model performance.
2) Tackling the class imbalance problem (i.e., natural bias toward the positive ratings) contributed to advance the proposed RRMs by inferring the appropriate adjustment weights that equilibrated this bias.
3) The extensive experiments on three Amazon datasets, depicted that the proposed RRMs surpassed all comparison methods in terms of Top-N recommendation.

5.2 Future Work

1. Developing the TPM by exploiting the other properties of trust theory: dynamism and context dependence. These aspects might

provide the system users with more trustworthy neighbors, thus improve the accuracy of RS.

2. Extending the RRMs to include trust relations in addition to the textual reviews. Such additional information is utilized to reach the best possible performance. In other words, incorporating the review-based RS with other types of recommenders, such as trust-aware RS, might be a potent combination of a hybrid RS.
3. Employing other rating similarity metrics to infer the nearest trustworthy neighbors and compare the results with the TPM results.
4. Applying other text similarity metrics to figure out the most similar features and compare the results with the RRMs results.
5. Building a weighted trust relations network from scratch based on implicit and explicit users' feedback and exploiting it for the improvement of RS.
6. Exploiting advanced rule-based methods to manipulate the textual reviews to deduce the optimal weights and employ them in review-based RS improvement.
7. Depending on timestamp information, text reviews can be filtered from other fake ones to avoid wrong recommendations to the target users within RRMs, because the incorrect suggestion of an item is worse than no correct suggestion of that item.

REFERENCES

[1] C. Zhang, M. Yang, J. Lv, and W. Yang, “An improved hybrid collaborative filtering algorithm based on tags and time factor,” *Big Data Min. Anal.*, vol. 1, no. 2, pp. 128–136, 2018, doi: 10.26599/BDMA.2018.9020012.

[2] S. Forouzandeh, M. Rostami, and K. Berahmand, “Presentation a Trust Walker for rating prediction in recommender system with Biased Random Walk: Effects of H-index centrality, similarity in items and friends,” *Eng. Appl. Artif. Intell.*, vol. 104, p. 104325, 2021, doi: 10.1016/j.engappai.2021.104325.

[3] V. K. Singh, M. Mukherjee, and G. K. Mehta, “Combining collaborative filtering and sentiment classification for improved movie recommendations,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2011, vol. 7080 LNAI, pp. 38–50, doi: 10.1007/978-3-642-25725-4_4.

[4] Y. Li, J. Liu, J. Ren, and Y. Chang, “A Novel Implicit Trust Recommendation Approach for Rating Prediction,” *IEEE Access*, vol. 8, pp. 98305–98315, 2020, doi: 10.1109/ACCESS.2020.2997040.

[5] S. M. Ali, G. K. Nayak, R. K. Lenka, and R. K. Barik, “Movie Recommendation System Using Genome Tags and Content-Based Filtering,” in *Lecture Notes in Networks and Systems*, vol. 38, Springer Nature Singapore Pte Ltd., 2018, pp. 85–94.

[6] M. D. Ekstrand, J. T. Riedl, and J. A. Konstan, “Collaborative filtering recommender systems,” *Found. Trends Human-Computer Interact.*, vol. 4, no. 2, pp. 81–173, 2010, doi: 10.1561/1100000009.

[7] H. Koohi and K. Kiani, “A new method to find neighbor users that improves the performance of Collaborative Filtering,” *Expert Syst. Appl.*, vol. 83, pp. 30–39, 2017, doi: 10.1016/j.eswa.2017.04.027.

[8] X. Chao and C. Guangcai, “Collaborative Filtering and Leaders’ Advice Based Recommendation System for Cold Start Users,” *PervasiveHealth Pervasive Comput. Technol. Healthc.*, pp. 158–164, 2020, doi: 10.1145/3404555.3404644.

[9] M. Krstić and M. Bjelica, “Impact of class imbalance on personalized program guide performance,” *IEEE Trans. Consum. Electron.*, vol. 61, no. 1, pp. 90–95, 2015, doi: 10.1109/TCE.2015.7064115.

[10] W. Shafqat and Y. C. Byun, “A Hybrid GAN-Based Approach to Solve Imbalanced Data Problem in Recommendation Systems,” *IEEE Access*, vol. 10, pp. 11036–11047, 2022, doi: 10.1109/ACCESS.2022.3141776.

[11] M.H. Hussein, W. S. Bhaya, and H. N. Nawaf, “Exploiting Social Influence for Recommendation System Improvement,” University of Babylon, Babylon, 2017.

[12] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, “Recommender systems survey,” *Knowledge-Based Syst.*, vol. 46, pp. 109–132, 2013, doi: 10.1016/j.knosys.2013.03.012.

[13] L. Chen, G. Chen, and F. Wang, “Recommender systems based on user reviews: the state of the art,” *User Model. User-adapt. Interact.*, vol. 25, no. 2, pp. 99–154, 2015, doi: 10.1007/s11257-015-9155-5.

[14] Y. Ma, G. Chen, and Q. Wei, “Finding users preferences from large-scale online reviews for personalized recommendation,” *Electron. Commer. Res.*, vol. 17, no. 1, pp. 3–29, 2017, doi: 10.1007/s10660-016-9240-9.

[15] F. Abbasi and A. Khadivar, “Collaborative Filtering Recommendation System through Sentiment Analysis,” *Turkish Journal of Computer and Mathematics Education*, vol. 12, no. 14, pp. 1843–1853, 2021.

[16] L. Yu, S. Pei, C. Zhang, S. Liang, X. Bai, N. Chawla, and X. Zhang, “Addressing Class-Imbalance Problem in Personalized Ranking,” *ACM*, vol. 37, no. 4, 2020, [Online]. Available: http://arxiv.org/abs/2005.09272.

[17] J. M. Johnson and T. M. Khoshgoftaar, “Survey on deep learning with class imbalance,” *J. Big Data*, vol. 6, no. 1, 2019, doi: 10.1186/s40537-019-0192-5.

[18] A. J. Costa, M. S. Santos, C. Soares, and P. H. Abreu, “Analysis of Imbalance Strategies Recommendation using a Meta-Learning Approach,” in *7th ICML Workshop on Automated Machine Learning (AutoML), AutoML 2020*, 2020.

[19] F. Roy and M. Hasan, “Comparative Analysis of Different Trust Metrics of User-User Trust-Based Recommender System,” Comp. Sci., vol. 23, pp. 337–375, 2020, https://doi.org/10.7494/csci.2022.23.3.4227.

[20] G. Guo, J. Zhang, F. Zhu, and X. Wang, “Factored similarity models with social trust for top-N item recommendation,” *Knowledge-Based Syst.*, vol. 122, pp. 17–25, 2017, doi: 10.1016/j.knosys.2017.01.027.

[21] G. Guo, J. Zhang, and D. Thalmann, “Merging trust in collaborative filtering to alleviate data sparsity and cold start,” *Knowledge-Based Syst.*, vol. 57, pp. 57–68, 2014, doi: 10.1016/j.knosys.2013.12.007.

[22] H. Zhao, Y. Zhang, and Y. Xiao, “A new collaborative filtering algorithm with combination of explicit trust and implicit trust,” in *13th International Conference on Computer Science and Education, ICCSE 2018*, 2018, pp. 319–323, doi: 10.1109/ICCSE.2018.8468763.

[23] Z. H. Zhou, *Ensemble methods: Foundations and algorithms*, 1st ed. New York: Chapman and Hall/CRC, 2012.

[24] Y. Mu, X. Liu, and L. Wang, “A Pearson’s correlation coefficient based decision tree and its parallel implementation,” *Inf. Sci. (Ny).*, vol. 435, pp. 40–58, 2018, doi: 10.1016/j.ins.2017.12.059.

[25] B. Ray, A. Garain, and R. Sarkar, “An ensemble-based hotel recommender system using sentiment analysis and aspect categorization of hotel reviews,” *Appl. Soft Comput.*, vol. 98, no. xxxx, p. 106935, 2021, doi: 10.1016/j.asoc.2020.106935.

[26] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter,” (The 5th EMC2 - Energy Efficient Training and Inference of Transformer Based Models) in *33rd Conference on Neural Information Processing Systems, NeurIPS 2019*, pp. 2–6, 2019, [Online]. Available: http://arxiv.org/abs/1910.01108.

[27] C. C. Musat, Y. Liang, and B. Faltings, “Recommendation using textual opinions,” *IJCAI Int. Jt. Conf. Artif. Intell.*, pp. 2684–2690, 2013.

[28] D. M. Blei, A. Y. Ng, and M. T. Jordan, “Latent dirichlet allocation,” *Adv. Neural Inf. Process. Syst.*, vol. 3, pp. 993–1022, 2002.

[29] M. Oppermann, R. Kincaid, and T. Munzner, “VizCommender: Computing text-based similarity in visualization repositories for content-based recommendations,” *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 2, pp. 495–505, 2021, doi: 10.1109/TVCG.2020.3030387.

[30] R. Habib and M. T. Afzal, “Sections-based bibliographic coupling for research paper recommendation,” *Scientometrics*, vol. 119, no. 2, pp. 643–656, 2019, doi: 10.1007/s11192-019-03053-8.

[31] J. Wang and Y. Dong, “Measurement of text similarity: A survey,” *Inf.*, vol. 11, no. 9, pp. 1–17, 2020, doi: 10.3390/info11090421.

[32] P. Valdiviezo-Diaz, F. Ortega, E. Cobos, and R. Lara-Cabrera, “A Collaborative Filtering Approach Based on Naïve Bayes Classifier,” *IEEE Access*, vol. 7, pp. 108581–108592, 2019, doi: 10.1109/ACCESS.2019.2933048.

[33] Y. Hu, F. Xiong, D. Lu, X. Wang, X. Xiong, and H. Chen, “Movie collaborative filtering with multiplex implicit feedbacks,” *Neurocomputing*, vol. 398, no. xxxx, pp. 485–494, 2020, doi: 10.1016/j.neucom.2019.03.098.

[34] Z. Ji, H. Pi, W. Wei, B. Xiong, M. Wozniak, and R. Damasevicius, “Recommendation Based on Review Texts and Social Communities: A Hybrid Model,” *IEEE Access*, vol. 7, pp. 40416–40427, 2019, doi: 10.1109/ACCESS.2019.2897586.

[35] F. Ricci, B. Shapira, and L. Rokach, *Recommender systems handbook, Second edition*. 2015.

[36] F. S. Gohari, F. S. Aliee, and H. Haghighi, “A significance-based trust-aware recommendation approach,” *Inf. Syst.*, vol. 87, p. 101421, 2020, doi: 10.1016/j.is.2019.101421.

[37] R. Barzegar Nozari and H. Koohi, “Novel implicit-trust-network-based recommendation methodology,” *Expert Syst. Appl.*, vol. 186, p. 115709, 2021, doi: 10.1016/j.eswa.2021.115709.

[38] J. Tang, X. Hu, and H. Liu, “Social recommendation: a review,” *Soc. Netw. Anal. Min.*, vol. 3, no. 4, pp. 1113–1133, 2013, doi: 10.1007/s13278-013-0141-9.

[39] M. V. K. Kiran, R. E. Vinodhini, R. Archanaa, and K. Vimalkumar, “User specific product recommendation and rating system by performing sentiment analysis on product reviews,” 2017, doi: 10.1109/ICACCS.2017.8014640.

[40] G. Ganun, Y. Kakodkar, and A. Marian, “Improving the quality of predictions using textual information in online user reviews,” *Inf. Syst.*, vol. 38, no. 1, pp. 1–15, 2012, doi: 10.1016/j.is.2012.03.001.

[41] A. M. A. Al-Sabaawi, H. Karacan, and Y. E. Yenice, “Svd++ and clustering approaches to alleviating the cold-start problem for recommendation systems,” *Int. J. Innov. Comput. Inf. Control*, vol. 17, no. 2, pp. 383–396, 2021, doi: 10.24507/ijicic.17.02.383.

[42] A. M. Ahmed Al-Sabaawi, H. Karacan, and Y. E. Yenice, “Exploiting implicit social relationships via dimension reduction to improve recommendation system performance,” *PLoS One*, vol. 15, no. 4, pp. 1–18, 2020, doi: 10.1371/journal.pone.0231457.

[43] M. R. Petrusel and S. G. Limboi, “A Restaurants Recommendation System: Improving Rating Predictions Using Sentiment Analysis,” in *Proceedings - 21st International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, SYNASC 2019*, 2019, pp. 190–197, doi: 10.1109/SYNASC49474.2019.00034.

[44] D. P. He, Z. L. He, and C. Liu, “Recommendation algorithm combining tag data and naive bayes classification,” in *Proceedings - 2020 3rd International Conference on Electron Device and Mechanical Engineering, ICEDME 2020*, 2020, pp. 662–666, doi: 10.1109/ICEDME50972.2020.00156.

[45] M. Papagelis, D. Plexousakis, and T. Kutsuras, “Alleviating the sparsity problem of collaborative filtering using trust inferences,” in *Herrmann, P., Issarny, V., Shiu, S. (eds) Trust Management. iTrust 2005. Lecture Notes in Computer Science*, 2005, vol. 3477, pp. 224–239, doi: 10.1007/11429760_16.

[46] G. Adomavicius and A. Tuzhilin, “Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions,” *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, 2005, doi: 10.1109/TKDE.2005.99.

[47] A. E. Castro Sotos, S. Vanhoof, W. van den Noortgate, and P. Onghena, “The transitivity misconception of Pearson’s correlation coefficient,” *Stat. Educ. Res. J.*, vol. 8, no. 2, pp. 33–55, 2009, doi: 10.52041/serj.v8i2.394.

[48] G. Guo, “Integrating trust and similarity to ameliorate the data sparsity and cold start for recommender systems,” in *RecSys 2013 - Proceedings of the 7th ACM Conference on Recommender Systems*, 2013, pp. 451–454, doi: 10.1145/2507157.2508071.

[49] S. Beamer, K. Asanović, and D. Patterson, “Direction-Optimizing Breadth-First Search,” in *SC ’12: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, 2013, pp. 137–148, doi: 10.1155/2013/702694.

[50] T. Hadad, “Review Based Rating Prediction,” pp. 1–9, 2016, [Online]. Available: https://arxiv.org/abs/1607.00024.

[51] M. Jiang, D. Song, L. Liao, and F. Zhu, “A Bayesian recommender model for user rating and review profiling,” *Tsinghua Sci. Technol.*, vol. 20, no. 6, pp. 634–643, 2015, doi: 10.1109/TST.2015.7350016.

[52] K. Iliopoulou, A. Kanavos, A. Ilias, C. Makris, and G. Vonitsanos, *Improving Movie Recommendation Systems Filtering by Exploiting User-Based Reviews and Movie Synopses*, vol. 585 IFIP. Springer International Publishing, 2020.

[53] N. Liang, H. T. Zheng, J. Y. Chen, A. K. Sangaiah, and C. Z. Zhao, “TRSDL: Tag-aware recommender system based on deep learning-intelligent computing systems,” *Appl. Sci.*, vol. 8, no. 5, pp. 1–15, 2018, doi: 10.3390/app8050799.

[54] L. Jiang, L. Liu, J. Yao, and L. Shi, “A hybrid recommendation model in social media based on deep emotion analysis and multi-source view fusion,” *J. Cloud Comput.*, vol. 9, no. 1, pp. 1–16, Dec. 2020, doi: 10.1186/s13677-020-00199-2.

[55] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, “RoBERTa: A Robustly Optimized BERT Pretraining Approach,” no. 1, 2019, [Online]. Available: http://arxiv.org/abs/1907.11692.

[56] J. Huwail and S. Imran, “An empirical comparison of machine learning methods for text-based sentiment analysis of online consumer reviews,” *Int. J. Res. Mark.*, vol. 39, no. 1, pp. 1–19, 2021, doi: 10.1016/j.ijresmar.2021.10.011.

[57] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” *NAACL HLT 2019 - 2019 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. - Proc. Conf.*, vol. 1, no. Mlm, pp. 4171–4186, 2018.

[58] Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler, “Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books,” *Proc. ICCV*, pp. 19–27, 2015.

[59] I. S. Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, “Language Models are Unsupervised Multitask Learners,” *Tech. report, OpenAI*, 2019, [Online]. Available: http://arxiv.org/abs/2007.07582.

[60] E. Strubell, A. Ganesh, and A. McCallum, “Energy and Policy Considerations for Deep Learning in NLP,” *57th Annu. Meet. Assoc. Comput. Linguist.*, vol. 34, no. 09, pp. 13693–13696, Jun. 2019, [Online]. Available: http://arxiv.org/abs/1906.02243.

[61] G. Hinton, O. Vinyals, and J. Dean, “Distilling the Knowledge in a Neural Network,” *NIPS 2014 Deep Learn. Work.*, pp. 1–9, 2015, [Online]. Available: http://arxiv.org/abs/1503.02531.

[62] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” *Adv. Neural Inf. Process. Syst.*, pp. 5999–6009, 2017.

[63] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, “GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding,” *ICLR*, pp. 353–355, 2018, doi: 10.18653/v1/w18-5446.

[64] X. Wei and W. B. Croft, “LDA-based document models for ad-hoc retrieval,” *Proc. Twenty-Ninth Annu. Int. ACM SIGIR Conf. Res. Dev. Inf. Retr.*, vol. 6, no. 11, pp. 178–185, 2016, doi: 10.1145/1148170.1148204.

[65] H. Jelodar, Y. Wang, C. Yuan, X. Feng, X. Jiang, Y. Li, and L. Zhao, “Latent Dirichlet allocation (LDA) and topic modeling: models, applications, a survey,” *Multimed. Tools Appl.*, vol. 78, no. 11, pp. 15169–15211, 2019, doi: 10.1007/s11042-018-6894-4.

[66] L. Feng and G. Wei-wei, “Recommendation Algorithm Based On Tag Time Weighting,” *2018 3rd Int. Conf. Smart City Syst. Eng.*, pp. 755–758, 2018, doi: 10.1109/ICSCSE.2018.00162.

[67] A. H. Jadidinejad, C. MacDonald, and I. Ounis, “Unifying explicit and implicit feedback for rating prediction and ranking recommendation tasks,” in *ICTIR 2019 - Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval*, 2019, pp. 149–156, doi: 10.1145/3341981.3344225.

[68] M. Jamali and M. Ester, “TrustWalker: A random walk model for combining trust-based and item-based recommendation,” in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 397–405, doi: 10.1145/1557019.1557067.

[69] J. McAuley and J. Leskovec, “Hidden Factors and Hidden Topics: Understanding Rating Dimensions with Review Text,” pp. 165–172, 2013, doi: 10.1145/2507157.2507163.

[70] R. K. Chaurasiya and U. Sahu, “Improving Performance of Product Recommendations Using User Reviews,” *3rd Int. Conf. Work. Recent Adv. Innov. Eng. ICRAIE 2018*, vol. 2018, no. November, pp. 1–4, 2018, doi: 10.1109/ICRAIE.2018.8710414.

[71] A. Da’u, N. Salim, I. Rabiu, and A. Osman, “Weighted aspect-based opinion

mining using deep learning for recommender system," *Expert Syst. Appl.*, vol. 140, 2020, doi: 10.1016/j.eswa.2019.112871.

[72] X. Chen, H. Xu, Y. Zhang, J. Tang, Y. Cao, Z. Qin, and H. Zha, "Sequential recommendation with user memory networks," *WSDM 2018 - Proc. 11th ACM Int. Conf. Web Search Data Min.*, vol. 2018-Febua, pp. 108–116, 2018, doi: 10.1145/3159652.3159668.

[73] A. Da'u, N. Salim, and R. Idris, "An adaptive deep learning method for item recommendation system," *Knowledge-Based Syst.*, vol. 213, p. 106681, 2021, doi: 10.1016/j.knosys.2020.106681.

[74] W. Liu, Y. Zhang, J. Wang, Y. He, J. Caverlee, P.K. Chan, D.S. Yeung, and P. Heng, "Item Relationship Graph Neural Networks for E-Commerce," *IEEE Trans. Neural Networks Learn. Syst.*, pp. 1–15, 2021, doi: 10.1109/TNNLS.2021.3060872.

[75] J. Chen, Y. Wu, L. Fan, X. Lin, H. Zheng, S. Yu, and Q. Xuan, "N2VSCDNNR: A Local Recommender System Based on Node2vec and Rich Information Network," *IEEE Trans. Comput. Soc. Syst.*, vol. 6, no. 3, pp. 456–466, 2019, doi: 10.1109/TCSS.2019.2906181.

[76] J. Zhang, D. Chen, and M. Lu, "Combining sentiment analysis with a fuzzy kano model for product aspect preference recommendation," *IEEE Access*, vol. 6, pp. 59163–59172, 2018, doi: 10.1109/ACCESS.2018.2875026.

[77] A. Suresh and M. J. Carmel Mary Belinda, "Online product recommendation system using gated recurrent unit with Broyden Fletcher Goldfarb Shanno algorithm," *Evol. Intell.*, no. 0123456789, 2021, doi: 10.1007/s12065-021-00594-x.

الخلاصة

نظم التوصية (RSs) هي أنظمة ذكية لتصفية المعلومات والتي تتعامل مع مشكلة التحميل الزائد للمعلومات. في الآونة الأخيرة ، أصبحت معلومات (social trust) الثقة الاجتماعية عاملاً إضافيًا مهمًا في الحصول على توصيات عالية الدقة بالإضافة إلى ذلك ، تلعب معلومات (textual review) المراجعة النصية دورًا أساسيًا في العديد من أساليب نظم التوصية التي يمكنها تحسين دقة التوصيات.

تعاني تقنيات نظم التوصية من مشاكل مثل (class imbalance) و (sparsity) و (cold-start) ، مما يقلل من دقة توصياتها. في هذا العمل ، تم اقتراح نموذجين لتحسين نظام التوصية وهذين النموذجين يستندان إلى استدلالات التغذية الراجعة الضمنية والتقييمات الصريحة. النموذج الأول هو نموذج التنبؤ المستند إلى الثقة (TPM) ويتم تقديم الآخر بنوعين من نماذج التوصية القائمة على المراجعة (RRMs).

في TPM، يتم الحصول على علاقات ثقة صريحة وضمنية بناءً على خاصية نشر الثقة ثم دمج هذه العلاقات للاستفادة من تقييمات اكثر للجيران الجديرين بالثقة للتخفيف من مشكلتي (cold-start) و (sparsity). وفقًا لذلك ، يتم تطبيق تقنية (weighted voting) المستمدة من (ensemble classifier) لانتخاب تقييمات أفضل الجيران الموثوق بهم. يتم استخدام طريقة (K Nearest Neighbor) من خلال دمج خطي للتقييمات الأصلية والتقييمات المنتخبة من خلال الثقة بواسطة تضمين وزن المساهمة (contribution weight) للحصول على أفضل تغطية ودقة للتنبؤ.

من ناحية أخرى، في RRMs، تم تطوير نموذجين للتوصية بناءً على وجود وغياب المراجعات النصية في مجموعة الاختبار. على وجه الخصوص، شكلت خمس خطوات أساسية هذه النماذج: المعالجة المسبقة للبيانات، تصنيف النص، نمذجة الموضوع، تشابه النص، و خطوة استنتاج أوزان التعديل وبالتالي التخفيف من مشكلتي (class imbalance) و (sparsity). يتم استخدام نموذج (Naïve Bayes) لدمج أوزان التعديل المستنتجة كتعليقات ضمنية للتوصية بالعناصر الأكثر تفضيلًا للمستخدم المستهدف.

تم إجراء تجارب مكثفة على النماذج المقترحة باستخدام خمس مجموعات من البيانات: (FilmTrust)، (Epinions)، (Musical Instruments)، (Automotive)،

و (Amazon Instant Video). أظهرت النتائج التجريبية أن (TPM و RRMs) المقترحة تفوقت بشكل ملحوظ على جميع الطرق التي تمت المقارنة معها في كل من مهمتي (Rating Prediction و Top-N Recommendation). على وجه التحديد، في TPM، تراوحت نسب التحسين تقريبًا من 4% كحد أدنى إلى 20% و 10% كحد أقصى على (FilmTrust) و (Epinions) على التوالي، من حيث مقياس F1 لمهمة (Rating Prediction). بينما، في RRMs، تحسنت دقة التوصيات بحوالي 10% كحد أدنى إلى 61% و 26% و 22% كحد أقصى على (Musical Instruments)، (Automotive)، و (Amazon Instant Video) على التوالي ، من حيث مقياس F1 لـ مهمة (Top-N Recommendation).

جامعة كربلاء
كلية علوم الحاسوب وتكنولوجيا المعلومات
قسم علوم الحاسوب

استغلال معلومات التغذية الراجعة الصريحة والضمنية لتحسين نظام التوصية

رسالة ماجستير

مقدمة الى مجلس كلية علوم الحاسوب وتكنولوجيا المعلومات / جامعة كربلاء وهي جزء من متطلبات نيل درجة الماجستير في علوم الحاسوب

كتبت بواسطة
حسين جبيرعودة مجيبج

بإشــراف
أ.م.د محسن حسن حسين

2023 م **1444 هـ**