Machine Learning : Publisher Recommendation Model

Master Thesis

zur Erlangung des Grades

Master of Science (M.Sc.)

im Studiengang Data Science

am Department Mathematik der
Friedrich-Alexander-Universität Erlangen-Nürnberg

vorgelegt am **7.Juli 2023**

von **Mohammed Quaidjohar Husain**

Prüfer: Prof. Dr. DhC. Enrique Zuazua

Acknowledgement

I would like to express my deepest gratitude to all the individuals and organizations who have supported me throughout the process of completing my master's thesis on "Machine Learning : Publisher Recommendation Model." Their guidance, encouragement, and assistance have played a pivotal role in the successful completion of this research endeavor.

I am immensely grateful to my thesis supervisor, Prof. Dr. Enrique Zuazua, for their unwavering support, expert guidance, and invaluable feedback. Their deep knowledge and insights in the field of machine learning have greatly contributed to shaping this research project. I am thankful for their constant encouragement, patience, and dedication in helping me navigate the complexities of this topic.

I would like to extend my heartfelt gratefulness to the members of my company AWIN and my company supervisor Ioana-Livia Stanila for their valuable insights, continuous support, and thoughtful suggestions throughout my thesis.

I would want to thank my family for their consistent support, love, and understanding as I pursued my academic goals. Their unwavering encouragement and confidence in my abilities have been a constant source of power.

In conclusion, I am profoundly grateful to all those who have contributed directly or indirectly to the completion of this master's thesis. Your support and encouragement have been invaluable, and I am honored to have had the opportunity to work on this research project.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Recommendation System

The exponential rise of digital content in recent years, along with the spread of internet platforms, has led to an excessive influx of data from all sources, which causes an information overload issue. Finding your way through this enormous sea of data may be difficult, time-consuming, and even intimidating. By offering consumers individualized recommendations and materials that are catered to their unique interests and preferences, recommendation systems have evolved as a solution to this issue. Systems that make recommendations to users based on past preferences, interactions with and engagement with products, etc. are known as recommender systems. The internet transmits a lot of data, and much of it reveals information about the user's search activities. The data used to make recommendations can come from articles and user-generated content, or it can come from explicit evaluations when readers are requested to review articles. Customers may receive helpful ideas from them that direct them to their intriguing products. Such apps might suggest books, CDs, and other products on Amazon.com, television shows, and movies on Netflix for instance. A wide range of businesses, including e-commerce, media, entertainment, and more, have acknowledged the value of recommendation systems. To give individualized suggestions, enhance client retention, increase sales, and increase revenue, businesses are investing more and more in the creation and improvement of recommendation algorithms. A recommendation system's performance depends on its capacity to comprehend user preferences, foresee their needs, and provide precise and timely recommendations.

Recommendation systems are one example of unsupervised learning [YLA21]. Without explicit labels or targets, the computer learns patterns or structures in the data in unsupervised learning. Similarly to this, the objective of a recommendation system using implicit ratings is to find trends or preferences in user behavior to provide tailored recommendations. In this thesis, we used an unsupervised clustering algorithm called k-means clustering, which can be used in recommendation systems to group comparable persons or products based on their characteristics or behavior. Then, by proposing

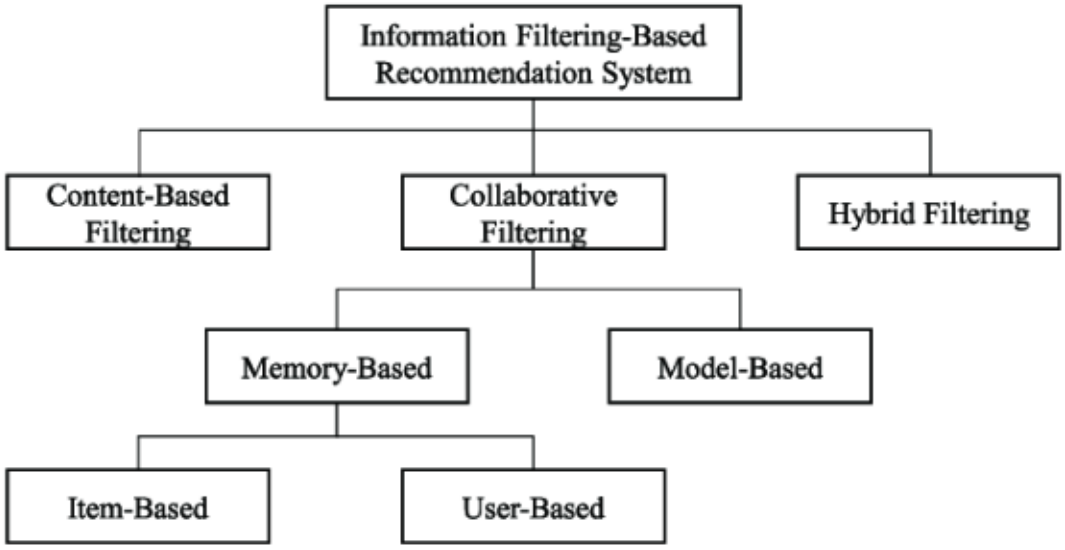

Figure 1.1: Classification of Recommendation Systems [KJC20]

products that are well-liked inside a specific cluster but haven't been interacted with by a specific user, these clusters can be leveraged to generate suggestions.

The categorization of the recommendation system [KJC20] based on information filtering is shown in Fig. 1.1. The recommendation system is classified into content-based, collaborative filtering, and hybrid filtering. Collaborative filtering is further divided into memory-based and model-based models. In order to recommend products or information, collaborative filtering makes use of user data to spot trends and similarities among users. Contrarily, content-based filtering concentrates on examining the characteristics and attributes of things to suggest equivalent ones. Hybrid strategies combine the two methodologies to offer a more thorough and precise recommendation.

1.2 Motivation

Information seeking has become difficult over the past few decades due to the availability of data on the Internet and the sheer number of online users. It has become challenging for consumers to get and keep track of the most pertinent information in the area. Since the development of the first publication on various recommender system techniques including content-based filtering and collaborative filtering in the mid-1990s, recommender systems have changed how people find and assess products that are available on the Web. A significant study area that has drawn numerous researchers during the past ten years is recommender systems. Different academics have put forth various strategies and methods concerning recommender systems. Many of these include explanations of how the current recommender systems operate, and others make suggestions for how they might be improved. Research in the field of recommender systems has continuously increased from the year 2015 to 2019 as shown in Fig. 1.2, with the highest

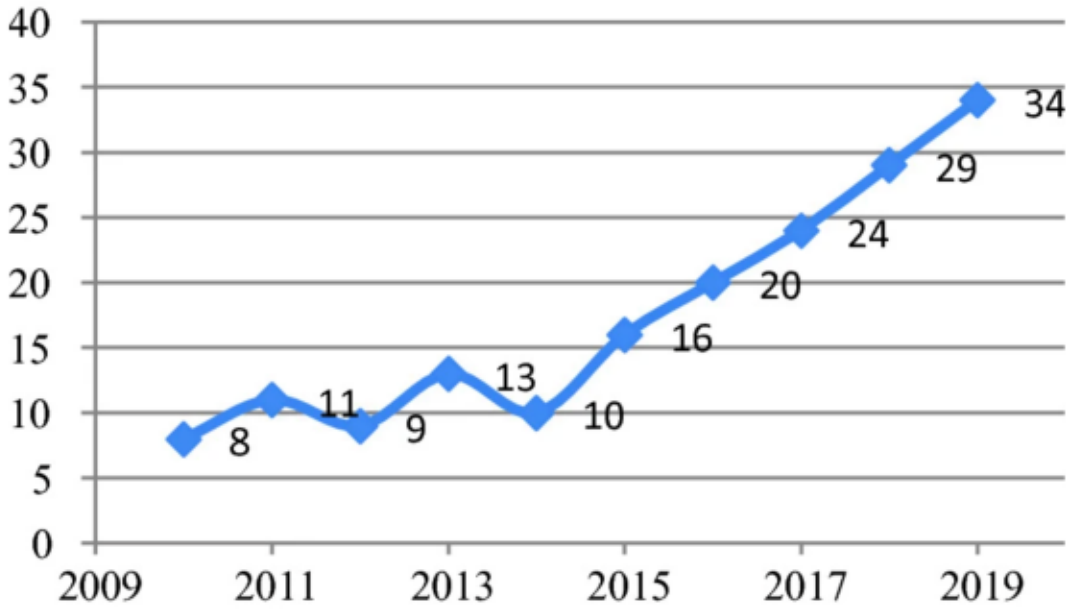

Figure 1.2: Research in Recommendation Systems from 2010 to 2019 [SAK21]

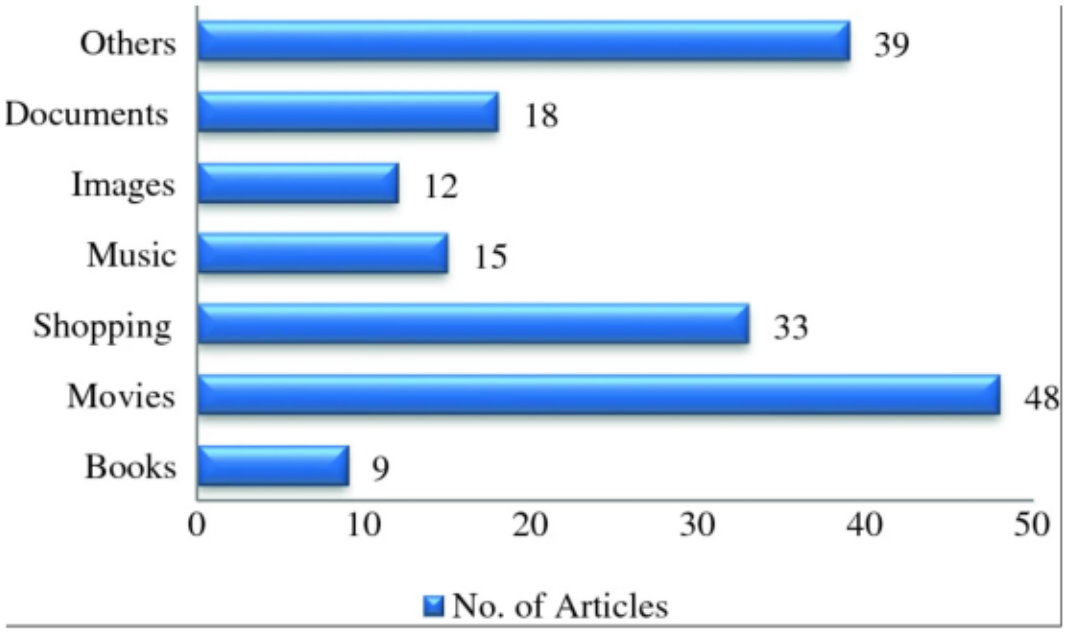

Figure 1.3: Recommendation System research paper distribution in different domains [SAK21]

number of studies occurring in the year 2019, according to a study analysis in [SAK21]. Such a sharp rise suggests that many scholars are exploring this particular research area in the field of recommender systems and contributing their original theories regarding recommender systems.

The findings of the distribution of all 174 research papers (as shown in Fig. 1.2) that were chosen from the year 2010 to 2019 are displayed in Fig. 1.3. According to this data, the majority of research papers have been published in the areas of recommender systems for movies (48 out of 174 articles, or 27.58%) and recommender systems for shopping (33 out of 174 articles, or 18.96%). This study also suggests that the majority of research publications are written for these two fields since they have more useful and practical applications than the other fields. This specific distribution also suggests that there aren't many study publications in the fields of books, photographs, documents, and music.

1.3 Problem Definition

Advertisers struggle mightily in the contemporary digital environment to successfully engage and reach their target demographic. Because there is so much content available, advertisers must develop creative strategies to engage with potential customers and increase return on investment. The chance to investigate how intelligent technologies may transform focused marketing strategies is compellingly presented by a thesis that focuses on a publisher recommendation system for advertisers. With the help of a sophisticated publisher recommendation engine, advertisers may reach a sizable audience of engaged readers who fit the profile of their target market. The system may monitor user behavior, preferences, and interactions to give highly relevant publisher suggestions by utilizing data analytics and machine learning algorithms. A publisher recommendation system creates opportunities for advertising and publishers to collaborate effectively. The system fosters collaboration and chances for both parties to thrive by bringing together marketers and publishers who operate in related industries. Publishers gain from greater revenue opportunities through advertising agreements, while advertisers can use the recommended publishers' current audience base to boost their reach. These collaborative relationships not only promote creativity but also result in a win-win situation for all stakeholders. It's essential to stay one step ahead of the competition in the competitive world of advertising. By using a publisher recommendation system, advertisers can access a large database of publishers who operate in similar industries, giving them a competitive advantage. With the help of this information, advertisers may reach out to up-and-coming publishers, delve into undiscovered markets, and spot cutting-edge advertising opportunities.

1.4 Thesis Objectives

Although recommender systems have successfully used collaborative filtering, there are still several issues that are known as Scalability concerns, Sparsity problems, and Cold-

start problems [Sar+01]. When new user first registers for the system, they experience the cold start issue. Since there is insufficient information regarding its rating, it is challenging to locate comparable users and things. Users and items are growing quickly in many large-scale e-businesses. New items cannot be suggested until some users vote for them, and the same is true for new goods. The accuracy of a recommendation system that compares users is compromised by this issue. Although the content-based strategy always suggests comparable products, one of its drawbacks is that it lacks innovation. While both content-based and collaborative filtering approaches have their advantages, both fail to provide quality recommendations under specific conditions. While content-based and collaborative filtering strategies have their benefits, under certain circumstances neither is successful in producing high-quality recommendations.

This thesis incorporates the k-means clustering technique with an alternating least squares algorithm and makes use of implicit ratings to foretell the recommendations after a comprehensive evaluation. First, the k-means clustering method is used to group similar publishers based on the sectors they are working in. Then the publisher clusters are linked with the advertisers based on the advertiser sector lying in the publisher clusters. The few implicit ratings are trained on alternating least squares to generate more advertiser publisher ratings. These ratings have been added to the result and several criteria (like sectors, sub-sectors, policy definitions, region, membership, etc) set by advertisers are implemented on the results. Then the final results are sorted in descending order based on ratings to give top publisher recommendations for advertisers. The proposed method can greatly speed up the suggestion and generate improved recommendation quality, according to experimental data. The tasks performed during the thesis are the following: 1) Understand and analyze the given data i.e. publishers, advertisers, region, ratings (Implicit), transactions, clicks, etc. 2) study of state-of-the-art machine learning models used in existing recommendations algorithm i.e. content-based [RP19], collaborative filtering [Neg21] and k means clustering [Hao21] models. 3) implementation of models on a given advertiser publisher dataset and generating recommendations according to criteria set by advertisers.

1.5 Thesis Organisation

The next chapter will provide a literature review of the different recommendation system methods available. Chapter 3 will give a technical overview of the different recommendation system models i.e. content-based, collaborative filtering, k-means clustering, and the algorithm used for advertiser publisher ratings with alternating least square algorithm used to generate more ratings. In Chapter 4, we will discuss the data used in this thesis. Chapter 5 will describe the methodology used by the system in detail. Finally, the results of the k-means clustering models over traditional content-based and collaborative filtering methods will be explained and discussed in Chapter 6. This thesiss conclusion in Chapter 7 and future work are discussed in Chapter 8.

Chapter 2

Literature Review

Since the first studies on collaborative filtering appeared in the middle of the 1990s, Recommendation systems have grown in significance as a field of study [Hil+95]. Over the past decade, there has been a lot of work done in both the industry and academia to develop novel ways to recommend systems. In [AT05], it has been described that recommendation systems are usually categorized into three main approaches i.e. content-based, collaborative, and hybrid recommendation.

The survey [Ko+22] examines the research trends that connect the business components of these services with the highly technologically sophisticated technical features of recommendation systems employed in numerous service domains. Additionally, the application service fields that used recommendation systems were categorized, and each field's research on the recommendation system model and approach was examined. In addition, enormous volumes of data on applied services, used by recommendation systems, were gathered between 2010 and 2021, without taking journal rankings into account, and examined along with several research on recommendation systems, as well as data on the applied services area industry. Initially, the Content-Based Filtering recommendation model, one of the earliest models to be employed, has increasingly been used by itself in the recommendation system model. Additionally, even though research on collaborative filtering has become more popular since 2014, it was discovered that research on hybrid systems that can supplement the advantages and disadvantages of both Content-Based Filtering and Collaborative Filtering recommendation models has grown significantly. Nevertheless, depending on the service application industry, there are some situations where using Content-Based Filtering and Collaborative Filtering is more appropriate. Studies on techniques including Text Mining, K nearest neighbor, Clustering, Matrix Factorization, and Neural Networks were examined with the filtering model of the recommendation system. The technologies of text mining and clustering, which analyze user or location data of a comparable group for recommendations, have been thoroughly investigated over a long period.

2.1 Content-Based Filtering

The personalized recommender system that uses content-based filtering methods to offer brief articles regarding home improvements is discussed in the article [Met00]. This type of area necessitates a very dynamic user model that can only be learned by positive feedback. Due to its efficiency and dynamic nature, the relevance feedback technique appears to be a strong contender for learning such a user model. According to the test results, slightly more than one out of every two recommendations made by the personalized recommendation system is appropriate. The fact that multiple terms can typically be used to represent the same idea and that many terms have multiple meanings has a negative impact on the findings. Because of this, the user profile is less accurate, especially given that the collection's documents are generally brief and the user typically only selects a small number of documents on a given subject. The vector space model could be enhanced to produce better outcomes. Based on users' prior usage and the content profiles of retrieved objects, content-based techniques suggest related objects. But extracting material from media files like movies, music, and videos is a highly difficult and time-consuming operation.

Particularly with regard to news, information overload currently precludes daily effective information gathering in these domains. This is increasingly important in relation to the internet and news websites since the quantity of newly added news is frequently used as the primary metric for determining how good a news portal is. A recommendation, particularly a tailored one, is the traditional approach typically employed to address information overload. A strategy for quick content-based news recommendation based on cosine-similarity search and efficient news representation is presented in the article [KB10]. The largest electronic daily in Slovakia published the results of the experiments using the suggested methodology.

2.2 Collaborative Filtering

Collaborative filtering [Sch+07] is one of the powerful personalization tools driving the adaptive web. Using other people's opinions to filter or evaluate items is a method known as collaborative filtering. Its technology facilitates the filtering of substantial amounts of data while bringing together the views of numerous connected and large online communities. The information must involve people in the loop who examine the data and distill their thoughts into information that software systems can quickly process to be filtered based on complicated variables.

Collaborative filtering is the method described in the paper [Das+07] for producing individualized recommendations for Google News readers. It produces suggestions using three methods: probabilistic latent semantic indexing, collaborative filtering using MinHash clustering, and visitation data. It uses a linear model to aggregate recommendations from various methods. Since the method is content-agnostic and hence domain-independent, it may be quickly and readily modified for use with various applications and languages. The document also provides a thorough explanation of the

algorithms and system configuration and presents the outcomes of testing the Google News recommendations engine.

In 1992 at the Xerox Palo Alto Research Center, the term "Collaborative Filtering" was first used in an email handling system named Tapestry [Gol+92]. The growing usage of electronic mail, which causes users to receive a deluge of incoming documents, is what spurred the development of Tapestry. Filtering is how Tapestry manages the massive influx of documents. By allowing people to mark documents and then allowing filtering based on those annotations, Tapestry enables collaborative filtering, which is different from existing systems in that it believes humans provide the most reliable evaluation of texts. Because annotations are unknown when documents are received, Tapestry filters must repeatedly run over the full document database in addition to testing incoming documents.

2.2.1 Model-Based Collaborative Filtering

One of the papers on model-based collaborative filtering [BHK13] from 1998, suggested using user preferences to foretell what other subjects or goods a potential new user could find interesting. This paper presents a number of algorithms created for this goal, such as vector-based similarity calculations, correlation coefficient-based algorithms, and statistical Bayesian methods. The results show that Bayesian networks with decision trees at each node and correlation methods outperform Bayesian clustering and vector similarity methods across a wide range of situations. The best approach between correlation and Bayesian networks depends on the characteristics of the dataset, the nature of the application, and the availability of votes to make projections.

The work suggests a brand-new Bayesian technique [BCT13] for personalized recommendations. This method uses multivariate Gaussian distributions to characterize user preferences and the things that are being recommended, and Normal-Inverse Wishart priors to model the recommendation agent's assumptions about different user types. The method was tested using the MovieLens dataset in the domain of movie suggestions. The chosen strategy is anticipated to outperform approaches that depend on user ratings in sparse datasets.

2.2.2 Memory-Based Collaborative Filtering

The collaborative filtering technique-based recommender system may experience scalability problems, cold start problems, and problems with new users and items. The most widely used and effective algorithm in the CF sector is K-Nearest Neighbors. Based on specific similarity metrics, the kNN algorithm estimates the "closeness" between items. Cold start issues with users and items are another issue with the traditional K-Nearest Neighbor Technique. According to the study [TS15], the recommender system creates recommendations for the user by integrating transaction data filtering and rating predictions based on user demographics, and item similarity. The weighted sum of the ratings generated from the transaction data, user data, and item data constitutes the final rating. The benefit of the suggested system is that the recommender system can

handle a "new user" or "new item" cold start. Additionally, compared to conventional collaborative filtering based on the K-Nearest Neighbor method, the system has low mean squared error and root mean square error (or root-mean-square deviation).

2.3 Hybrid Filtering

Scalability is one of the major problems that memory-based algorithms currently have, according to the authors of [Yu+02]. This study focuses on common user preference datasets with lots of missing values. To increase the effectiveness and precision of the memory-based method, they suggested four innovative instance reduction strategies, dubbed TURF1 through TURF4, as a preprocessing stage before the memory-based collaborative filtering approach. The main concept is to make predictions using a properly chosen set of pertinent events. The experiment demonstrated that the suggested methods not only significantly increased prediction speed but also predicted accuracy. The suggested method consists of four phases; the first algorithm adds training instances with novel profiles to the training set incrementally, starting with a small number of training instances. Filtering is carried out by the second algorithm, which adds the instance with the most logical profile to the training set. To select examples with a strong rational and novel profile, the third algorithm works in conjunction with earlier algorithms. The possibility for storage reduction is taken into account in the final algorithm.

The study [Ras+06] suggests a hybrid memory-based and model-based collaborative filtering strategy that focuses primarily on enhancing the quality of the recommendations. Their CLUSTKnn technique is an easy-to-use algorithm that works well with large datasets and is reasonably priced. The initial step of the process is to develop a simple but effective clustering model, which greatly compresses the data. Then, recommendations are promptly produced by employing a straightforward Nearest Neighbor-based methodology.

2.4 Naive Bayes Classifiers

Naive Bayes classifier [RSA22], one of the most efficient kinds of machine learning and artificial intelligence algorithms now in use, is used to increase user and object trustworthiness confidence and enhance recommendation accuracy. The Naive Bayes classifier's implementation is easy because it only requires the class node to be linked to every other attribute's node. The method is demonstrated to be workable because, when applied to an online dataset of a social network, it achieved a prediction accuracy of 89% with a confidence level of roughly 0.89. The use of a Bayesian classifier in a recommender system is effective, although lack of data or improper access to data may be the primary factor in incorrect classifier implementation. It is known to greatly reduce errors and can involve a large set of training data.

The Naive Bayes classifier method is being utilized in [Pan+20] to classify data. The YouTube video comment dataset, which was compiled from well-known YouTube videos, was used in this study. The Naive Bayes methodology, a probabilistic analysis

method, is used to examine every record in the dataset, and the outcome is then shown as a list. The Naive Bayes Classifier is used to determine the overall number of positive and negative reviews in the provided review collection. The final step is to base the recommendation system on categorized positive and negative remarks. The suggested method might be used to analyze the sentiments in any type of review dataset.

2.5 K-Means Clustering

The methods utilized in a comparative analysis [SNP17] included k-mean clustering, collaborative filtering, hybrid content-collaborative filtering, and naive Bayes classifier. The algorithms were fully utilized to attain the highest level of precision, and they have provided a thorough comparative study. The 10K, 50K and 100K MovieLens datasets were compared for accuracy. The dataset's sparsity varies. For instance, the 100K MovieLens dataset has 1682 films from 19 distinct genres, 943 users, and 100K ratings. These algorithms' analysis is illustrated using a precision metric. It applied the aforementioned algorithms and turned 30% of each test user's previously viewed movies into previously unseen flicks. Regarding their precision rates, all the algorithms discussed in this study are contrasted. This thorough examination shows each one's advantages and disadvantages in various MovieLens dataset iterations. The best precision is provided by K Means clustering and Naive Bayes among all of these methods.

In another such study [Cho+12], the recommendation system uses K-means clustering of item based category on Recency, Frequency, Monetary (RFM). To represent the characteristics of the item and identify those with a high likelihood of being purchased, the study suggested a personalized recommendation system based on the RFM approach employing k-means clustering of item categories. The implicit technique, which avoids complex questions and answers, is more convenient and straightforward for the system to use in this proposed system to reduce consumers' search effort than the explicit method, which relies on rating data and is frequently used in the existing system.

Clustering is used as a first stage in a contemporary recommender system design in a review-based study [BK21] to enhance overall performance. The data sparsity of user-preference matrices changes in user preferences over time, and boosting the diversity, consistency, and reliability of suggestions are all concerns that can be resolved by using clustering in recommendation systems. The examination of the scholarly literature on recommender systems and clustering models is the main emphasis of this review. The nearest neighbor query range is not conducive to a real-time recommendation, and the traditional collaborative filtering recommendation algorithm based on user rating is very sparse and not a good predictor of user interest because the user changes over time. The conclusion is that as in any area of machine learning, algorithms for recommender systems generally change and become more complex. Clustering is a good preventative measure before using recommendation algorithms. However, both the clustering model and the recommender algorithm now have a role in overall effectiveness.

The authors of [DM11] designed a new collaborative filtering algorithm. The most effective algorithm for recommender systems is collaborative filtering. An intelligent

system that can assist users in finding interesting goods is a recommender system. It employs information filtering and data mining methods. Users receive suggestions based on the tastes of their neighbor's thanks to collaborative filtering. However, it has issues with accuracy and scalability. The new method produced a recommendation for an active user based on user clustering. Users are categorized according to their interests using the k-means clustering technique. It then creates a recommendation using a novel technique called a voting algorithm. The scalability issue affects the collaborative filtering algorithm. The time it takes to generate suggestions, particularly for user-based algorithms, increases as the user-Item dataset grows. The results demonstrate that the suggested method is less time-consuming and more accurate than the traditional one.

Chapter 3

Algorithms

This chapter will provide a technical overview of different recommendation system models that were used in this thesis along with explanations of the algorithms for Content-Based, Collaborative Filtering, Alternating Least Square, and K-Means Clustering. Also, this chapter will include the recommendation system ratings that have been used by the model as an important feature to produce top publisher recommendations for advertisers and publishers.

3.1 Content-Based Filtering

The content-based filtering [Met00] uses the ratings or points rated by users towards the item and evaluates the similarities between the items to predict the recommendations. An implicit or explicit rating from the user is the basis for a content-based model's operation. By using the information, we build a profile of the user, which is then used to make suggestions to the user. As the user adds more information or acts more frequently on the recommendation, the engine improves in accuracy. The basic goal of content-based techniques is to attempt to develop a model that can account for the observed user-item interactions based on the "features" that are already accessible.

Making recommendations doesn't require any user data at this time. Contrary to collaborative filtering and other machine learning algorithms, content-based filtering does not require user data to generate recommendations and it is the advantage of the algorithm over others as stated in [BGB15]. To recommend things that have not yet been put by any users, content-based recommenders systems are appropriate. For new users, this will be useful. Because the approach depends on matching the features or properties of a database object with the user's profile, content-based recommenders can be extensively customized to the user's preferences, including recommendations for specialized products. To the user, recommendations are transparent. Highly relevant recommendations convey an air of openness to the user, increasing their degree of trust in the suggestions made. Systems for content-based filtering are typically simpler to develop. Compared to other filtering systems meant to replicate user-to-user recommendations, the data science underpinning a content-based filtering system is rather simple.

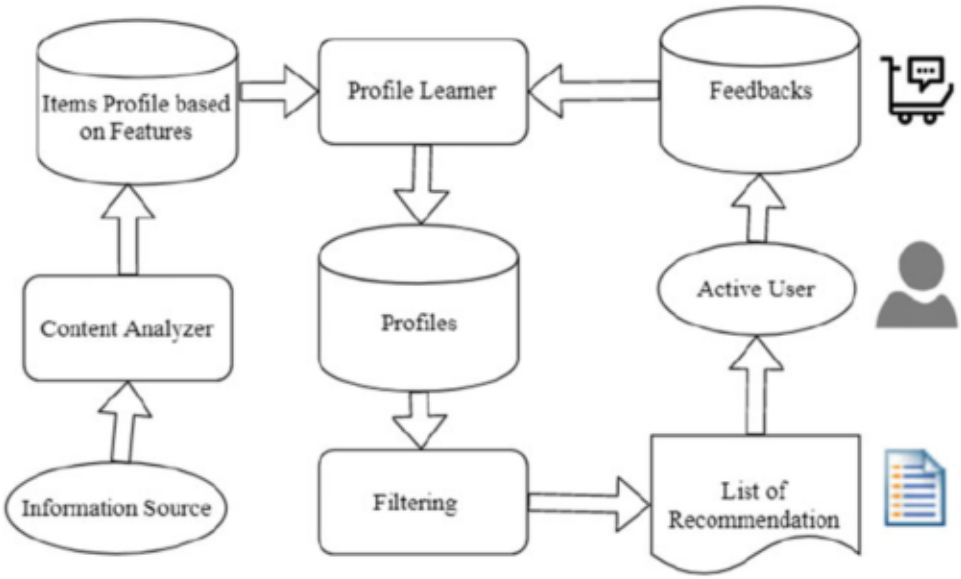

Figure 3.1: Workflow of Content-Based Filtering Model [RP19]

As stated in [BGB15], generating the qualities for things in some sectors is a challenging undertaking task for the content-based model. It suffers from an overspecialization issue because it promotes the same kinds of products. Because users do not often score the items (as in collaborative filtering), it is more difficult to get user input on content-based recommendations. As a result, it is impossible to know whether the recommendation is accurate. Fig.3.1 shows the workflow of a content-based model.

Algorithm 1: Content-Based Filtering

Data: Publishers, Advertisers Data
Result: AdvertiserKey, PublisherKey, Ratings
while *for all advertisers* **do**
  join advertiser policy definitions and publisher verticals;
  create advertiser and publisher profiles;
  from publisher recommendation ratings, take the top 10 publishers for the advertiser and call the cosine similarity function;
  Filtering publishers on the basis of features (like region, subsector, policy definitions, etc.) ;
end

3.1.1 Cosine Similarity

The vector similarity approach, which may represent each document as a vector based on the frequency of words as described in [MW07], was initially employed in the disciplines of text mining and natural language processing to determine the similarities between two documents. Cosine similarity can be used to compute similarity in recommendations as well. In user space, the item features can be thought of as vectors, and the angle's cosine can be used to measure how similar two items are. To determine the difference between two or more vectors, it measures the cosine angle's inner product space. Due to its rapid evaluation and consideration of user ratings as vectors, cosine similarity is a well-liked similarity metric. However, one disadvantage is that the different grading scales are not taken into account.

Algorithm 2: Cosine Similarity

Data: u, v
Result: result
Function `cossim(u,v):`
    result = float(float(v.dot(u)) / (v.norm(2) * u.norm(2)));
    return *result*;

3.2 Collaborative Filtering

The most common and widely used recommendation method is collaborative filtering [Neg21]. It is a technique for a recommendation system that is created by user cooperation.

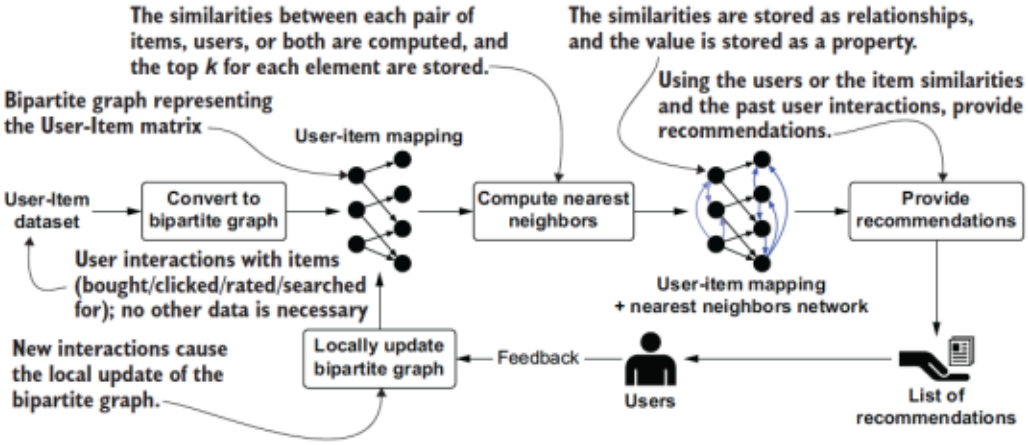

Figure 3.2: Workflow of Collaborative Filtering Model [Neg21]

In collaborative filtering as shown in Fig.3.2, we frequently identify similar users and suggest things that they enjoy. Instead of using the characteristics of the item to propose it in this type of system, we group users into the bucket of like-minded people and then make recommendations based on the preferences of its bucket for each one. When a newly constructed website or community has few brand-new users and few user connections, it creates a potential cold-start scenario. Large data sets and a variety of application domains, including finance, weather forecasting, environmental sensing, and e-commerce, were used in collaborative filtering algorithms. It offers suggestions for a variety of settings with billions of consumers and products. Therefore, computing recommendations frequently require a significant amount of processing power.

Collaborative filtering is further divided into two main categories i.e. memory-based filtering and model-based filtering.

3.2.1 Memory-Based Filtering

Memory-based techniques [RP19] are simple and simple to use. The most well-known method is memory-based neighborhood-based filtering, which forecasts preferences by referring to persons or objects that are comparable to the user or object being searched for. The neighborhood technique's precision and effectiveness are significantly impacted by the method used to determine how similar users or products are. These methods are further divided into item-based filtering methods and user-based filtering methods.

User-Based Filtering

The user-based filtering [IFO15] technique compares users' preferences for the same item to determine how similar they are, and it also determines the active user's expected choice of items.

Item-Based Filtering

The similarity between items is used in the item-based filtering [IFO15] algorithms to create predictions. The method finds similarities between the obtained things and the target item by retrieving all the items that have been rated by an active user. To anticipate the active user's preference for the target item, it then chooses the top N most comparable things.

3.2.2 Model-Based Filtering

Model-based methods anticipate a user's preference for a product by using data mining and machine-learning techniques. These models are made up of pertinent features and their weights. To generate the models, the raw user-item dataset must first be analyzed offline. During the recommendation phase, only the developed model is required to make recommendations. It improves the performance of the model.

Algorithm 3: Collaborative Filtering

Data: Publishers, Advertisers Data
Result: AdvertiserKey, PublisherKey, Ratings
while *for all advertisers* **do**
  join advertiser policy definitions and publisher verticals;
  create advertiser and publisher profiles;
  from publisher recommendation ratings, take ratings and join them with advertiser details;
  select the top 20 publishers for the advertiser Filtering publishers on the basis of features (like region, subsector, policy definitions, etc.) ;
  then find top publisher's top advertisers;
  then model again does a second set of filtering on features;
end

3.3 Alternating Least Square (ALS)

The matrix factorization process is known as alternating least squares [CA19] which is a type of model-based collaborative filtering that runs independently in parallel. It is designed for large-scale collaborative filtering issues and is implemented in Apache Spark ML. Applications requiring quick data transformations and analytics output are ideally suited for Apache Spark. It performs gradient descent with the item matrix first, holding the user matrix fixed, and then performs gradient descent with the user matrix to minimize two loss functions alternately. The underlying training data from a cluster of machines is used to perform its gradient descent algorithm in parallel across numerous partitions. The algorithm is called Matrix Factorization. A huge matrix is broken down into products of matrices via matrix factorization. From Fig.3.3, we can notice how the users and movies rating are broken down into two separate matrixes of user and movies (item) and then they are used to generate user and movie relationships.

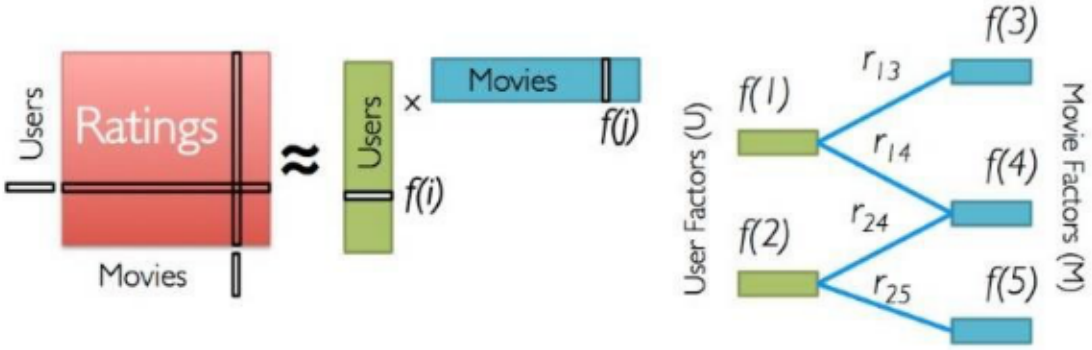

Figure 3.3: Workflow of Alternating Least Square Model [CA19]

The Alternating Least Square method can be used for two things i.e it can be independently used for developing a recommendation system on Apache Spark MLlib whereas it can be used to generate more ratings from publisher recommendations rating data as we have fewer ratings for advertisers and publishers, so we try to generate more ratings based on features. In this thesis, the ratings generated by the ALS model are further used by the k-means clustering model to sort out top recommendations based on descending-order ratings.

Algorithm 4: Alternating Least Squares (ALS)

Input: advertiser-publisher matrix p, rank r, regularization parameters λ, maximum iterations max_iter
Output: advertiser and publisher latent factor matrices A and P
Initialize advertiser and publisher latent factor matrices A and P;
for $iter = 1$ *to* max_iter **do**
  for *each advertiser* a *and each publisher* p **do**
    Update advertiser latent factor a and publisher latent factor p by solving the least squares problem;
  end
end

3.4 K-Means Clustering

The k-means clustering [NXY10] method is an iterative, numerical, unsupervised, and non-deterministic approach. The approach has proven to be quite effective in many real-world applications since it is quick and easy to use, and it can give strong clustering results. However, it is excellent for creating globular clusters. While doing so, this technique lessens the influence of solitary points and 'noise,' improving the effectiveness of clustering. However, the complexity of time is not improved by this strategy.

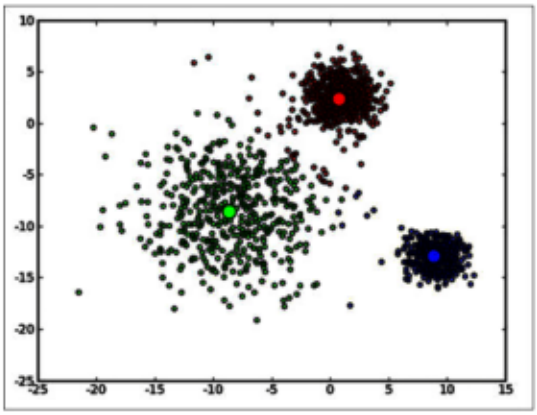

Figure 3.4: Graph plot of K-Means Clustering Model [Jai10]

As shown in Fig.3.4, the clustering method successfully locates target objects to clusters that are comparable to the object and seeks to combine similar objects into one cluster. Since all of the objects in each cluster are similar to the target object, it is easy to identify the closest neighbors in each cluster, reducing the dimension of the data and speeding up processing. Each cluster has a centroid, which is the average value of all the cluster's components. A cluster's centroid is the point at which all of its members travel, and it is updated after each iteration as can be seen in workflow in Fig.3.5. The centroid keeps changing until a saturation point is reached, at which point it stops changing.

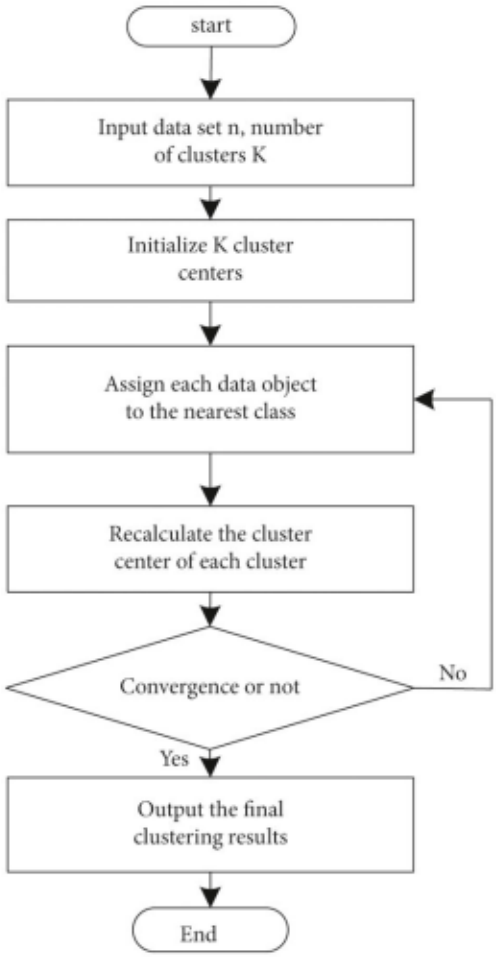

Figure 3.5: Workflow of K-Means Clustering Model [Hao21]

Algorithm 5: K-Means Clustering

Data: Publishers, Advertisers Data
Result: AdvertiserKey, PublisherKey, Ratings
select k (number of clusters) and select k centroids randomly ;
while *for all publishers* **do**
    calculate the distance of the point to k centroids;
    assign the point to the nearest centroid and update the cluster;
    update the new centroid;
end
while *for all advertisers* **do**
    assign the advertiser to the particular clusters of publishers;
    from publisher recommendation ratings, take ratings and join them with advertiser details;
    Filtering publishers on the basis of features (like region, subsector, policy definitions, etc.) ;
end

3.5 Recommendation System Ratings

The recommendation systems need a rating or score parameter to calculate the top recommendations for a user. There are two types of ratings i.e.

Explicit Ratings

Ratings in which a user expresses their opinion of a product out loud. The user may be given the option to choose their favorite foods from a pre-selected list or be given a list of previously eaten items.

Implicit Rating

Ratings are produced automatically by a model using one or more algorithms (in this example, MinMax Scaling) to calculate and offer a rating in accordance with the weights assigned to the various columns.

The model doesn't have pre-defined ratings for advertisers and publishers, the publisher recommendation ratings were built to generate implicit ratings. For generating the ratings, six months of data on transactions, clicks, sales, conversion rates, Average Order Value, Cost Per Acquisition, and different features of publishers and advertisers were used to prepare the dataset. Then all the data are combined (union). Re-scale each feature individually to a common range [min, max] linearly using column summary statistics, which is also known as min-max normalization or re-scaling. Now, it uses MinMax scaling available in the Pyspark ML library. The values are scaled between min=0 and max=1.

Since we do not have any explicit rating given by an advertiser to the publisher, the following data points are used to calculate the implicit rating which indicates the effectiveness of membership between an advertiser and publisher. We use the following data points as mentioned in Table 3.1 to calculate the rating

Data Points	Calculation	Weights	Description
AdvertiserKey	-	-	Advertiser who has a membership
PublisherKey	-	-	Publisher who has a membership
Clicks	Sum of clicks	1	Number of clicks generated by an Advertiser-Publisher pair
Transactions	Sum of transaction count	2	Number of transactions generated by an Advertiser-Publisher pair
Average Order Value (AOV)	Sales/Transactions	3	Average Order Value of each transaction
Cost Per Acquisition (CPA)	Commission/Sales	3	Cost to acquire customer
ConversionRate	Transactions/Clicks	3	Metric for successful conversion of customers from clicks to transactions

Table 3.1: Data Points of Rating Calculation

Algorithm 6: Publisher Recommendation Ratings

Data: Publishers, Advertisers Data
Result: AdvertiserKey, PublisherKey, Ratings
*rating = (col(Clicks*1)) + (col('Transactions')*2) +*
*(col('AverageOrderValue')*3) +(col('CostPerAcquisition')*3) +*
*(col('ConversionRate')*3)* ;

Chapter 4

Data

The dataset used during the thesis was acquired from AWIN company. The dataset is private and that's why most details are hidden. In this chapter, we will discuss the different data used in the model implementation in this thesis.

4.1 Advertiser

The Advertiser acts as a User of the recommendation system. The dataset has a total of 40,422 advertisers that were used in the model. The fields of the Advertiser table are shown in Table 4.1.

Attribute	**Type**
Advertiser	string
AdvertiserKey	integer
AdvertiserID	integer
AdvertiserSector	string
AdvertiserSubSector	string
AdvertiserRegion	string

Table 4.1: Advertiser Table Attributes

4.2 Publisher

The Publisher acts as an Item set of the recommendation system. The dataset has a total of 747,116 publishers that were used in the model. The fields of the Publisher table are shown in Table 4.2. The publisher Country field acts similarly to the Advertiser Region field as both state the location of the publisher or advertiser respectively.

Attribute	Type
PublisherKey	integer
PublisherID	integer
Publisher	string
PublisherVertical	string
PublisherSubVertical	string
PublisherTypeGroup	string
PublisherWebSite	string
PublisherJoinDate	date
PublisherCountry	string
PublisherSector	string
PublisherSubSector	string

Table 4.2: Publisher Table Attributes

4.3 Recommendation System Ratings

The advertiser publisher ratings i.e. Implicit Ratings are generated through Recommendation System Ratings. These ratings are fewer in numbers that's why they have been ingested into the ALS model to get more frequent ratings for advertisers and publishers in the proposed methodology. The ratings were generated for 10,891 advertisers and 85,326 publishers only. Fig.4.1 shows the example of data of ratings that we used in the ALS model.

	AdvertiserKey	PublisherKey	ratings
1	398	5972916	70.49059486580927
2	8482	223634	48.407108770634885
3	142182	60695	45.310309439874715
4	49510	229873	38.239613059326324
5	9439	230324	35.91811341387904
6	5313	53163	34.96810829667407
7	186549	214219	25.0494568942562
8	16313	6487020	23.877787389428732
9	1988	3581028	22.939764590841634
10	32664	223590	21.718506940851984
11	30399	229361	20.330765406397866
12	32894	230478	19.938104350051045

Figure 4.1: Recommendation System Ratings Data

4.4 Feedback Score

The feedback score data directly comes when the Advertiser scores the Publishers according to the relevance of the nature of work similar to each other out of 5. This data as shown in Fig.4.2 is directly used in the model to avoid low scores of advertiser-publisher pairs.

	AdvertiserKey	PublisherKey	FeedbackScore
1	1115	3456422	3
2	1496	1209921	2
3	3655	365801	1
4	3655	373608	3
5	3655	5905008	2
6	3655	5913078	2
7	3696	6474772	1
8	3899	352589	3
9	3899	366947	1
10	3899	1171081	1

Figure 4.2: Feedback Score Data

Chapter 5

Methodology

Numerous recommendation systems have been put forth in recent years with significant outcomes. Although content-based and collaborative filtering has shown to be extremely effective in practice, the techniques and their learning process have several drawbacks. In this thesis, we have proposed a novel solution to overcome the problem by using the clustering method. In this chapter, we discuss the methodology used in this thesis considering the K means clustering algorithm and ALS model.

The proposed methodology in this thesis, as shown in Fig.5.1, is divided into nine steps. In this chapter, we will discuss these steps in detail.

5.1 Preprocessing of Data

As discussed in Chapter 4 about the dataset used in the model, this data is preprocessed before sending it to the K means clustering algorithm or any further use. The preprocessing step includes removing publishers and advertisers that are inactive, whose sectors and sub-sectors are undefined, and publishers whose verticals, sub-verticals, and type groups are undefined. These are important attributes that would be used further in the model, so it's necessary to remove unwanted garbage data for both advertisers and publishers.

5.2 Publisher Data

After cleaning data, we create a dataframe with publisher data with PublisherKey and Sectors as shown in Fig.5.2. These two attributes are needed to give input to K means clustering algorithms as we create clusters for publishers based on the sectors they are classified into.

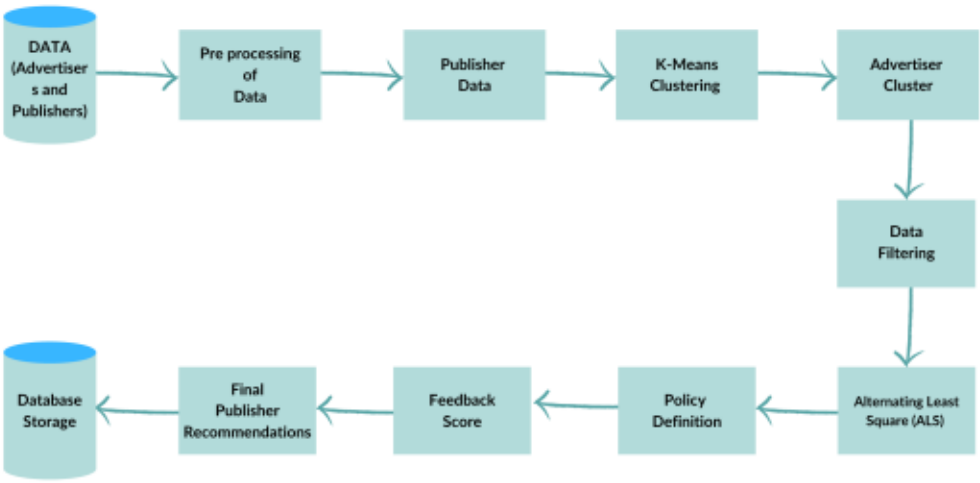

Figure 5.1: Publisher Recommendation System Workflow

PublisherKey	Sector
1201	[Retail & Shopping]
1274	[Travel]
1645	[Telco's & Services, Retail & Shopping, Financ...
1959	[Finance & Insurance, Travel]
2212	[Travel, Retail & Shopping, Finance & Insuranc...
...	...
7267788	[Telco's & Services, Retail & Shopping, Financ...
7267813	[Retail & Shopping]
7267839	[Retail & Shopping]
7267859	[Retail & Shopping]
7267866	[Retail & Shopping, Travel]

Figure 5.2: Publisher Data

5.3 K-Means Clustering

By speeding up the neighborhood construction in huge datasets, clustering algorithms are included in recommendation systems to increase their time performance. Large item sets are grouped according to publishers in this thesis. Publishers in the same cluster frequently operate in comparable industry sectors, therefore collaborative filtering techniques can enhance the quality of recommendations by utilizing these traits. In addition, the clustering technique eliminates items with different characteristics that can have an impact on the precision of the forecast. The accuracy, however, may differ between various clusters because the suggestion quality is cluster dependent.

In this thesis, by choosing one point for each cluster, the k-means algorithm initializes the k number of clusters. We chose the value of k as four for our experiment, as there were four distinct sectors for both publishers and advertisers. The closest centroid of the cluster is used to group each location. The centroids of the k clusters are modified after all points have been grouped, reassigning all points to the new closest centroids. Up until all of the points stay in the same cluster for a number of steps, this process is repeated. Fig.5.3 describes the division of publishers into four clusters with the count of publishers present in each cluster while Fig.5.4 depicts the example of publisher division into clusters with PublisherKey, its equivalent sector, and cluster label according to sector.

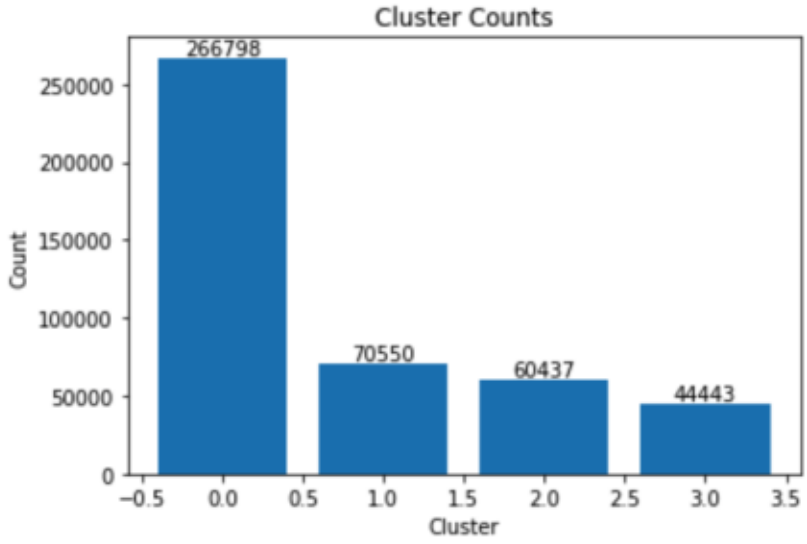

Figure 5.3: Cluster wise Publishers Count

	PublisherKey	Sector	ClusterLabel
1	1201	Retail & Shopping	0
2	1274	Travel	3
3	1645	Travel,Retail & Shopping,Finance & Insurance,Telco's & Services	2
4	1959	Finance & Insurance,Travel	3
5	2212	Travel,Retail & Shopping,Finance & Insurance,Telco's & Services	2
6	2721	Travel,Retail & Shopping,Telco's & Services	1
7	2848	Travel,Retail & Shopping,Finance & Insurance,Telco's & Services	2

Figure 5.4: Publisher Clustering

5.4 Advertiser Cluster

The publisher data have been divided into four clusters based on sectors. Now, the advertiser data consisting of AdvertiserKey and Sector is joined with these clusters based on advertiser sectors because the value of the publisher's sector and advertiser sectors are the same. This way we can have proper recommendations of publishers for advertisers working in the same sectors. Fig.5.5 shows the example of AdvertiserKey, PublisherKey, Publisher Sector, and Cluster in which the Advertiser and Publisher lie together.

AdvertiserKey	PublisherKey	Sector	ClusterLabel
2738	1201	Retail & Shopping	0
2738	7910	Retail & Shopping	0
2738	9441	Retail & Shopping	0
2738	14302	Retail & Shopping,Travel	0
2738	15254	Retail & Shopping	0
...	...	...	...
7410	7258839	Retail & Shopping	0
7410	7258898	Retail & Shopping	0
7410	7258900	Retail & Shopping,Travel	0
7410	7258901	Retail & Shopping	0
7410	7259261	Retail & Shopping	0

Figure 5.5: Advertiser Publisher Cluster

5.5 Data Filtering

The advertiser has linked with the respected cluster and list of publishers in that cluster. The advertisers need publisher filtering. In this step of the model, we check that the advertisers and publishers are in the same sub-sectors (this is another feature than sectors), and region (country of advertiser and publisher like United Kingdom, Germany, etc.). In this step, we also add a new column named Rank which has integer values if Rank equals 1 means the advertiser region and publisher region are the same as well as the advertiser subsector and publisher subsector are the same. Whereas if the advertiser region is not equal to the publisher region but the advertiser subsector and publisher subsector are the same, then it would Rank as 3. In this thesis, we work on recommendations whose Ranks are set to 1 as shown in Fig.5.6. In this filtering of data, we also removed advertiser publisher pair that are already present in membership i.e. they already are working together according to the data present. So it's useless to recommend a publisher to the advertiser, with whom they are already working.

	AdvertiserKey	PublisherKey	Rank
1	51063	267138	1
2	7410	211967	1
3	51063	3430214	1
4	7410	361871	1
5	7410	5596346	1
6	8	6770952	1
7	7410	177817	1
8	7410	146525	1
9	7410	7205828	1
10	2738	3431350	1

Figure 5.6: Data Filtering of Advertisers and Publishers

5.6 Alternating Least Square (ALS)

The model needs to recommend top publishers, so for that, we need some criteria like scores or ratings. But we don't have any feedback from advertisers for publishers. The implicit rating is already being calculated for some of the advertiser publisher pair based on transactions, sales, clicks, etc. as per the algorithm presented in section 3 of this thesis. But this data is very limited as it's only for advertisers and publishers who have worked in the past or have membership among them. So, in this step, we try to train that limited data and generate more ratings for advertisers and publishers with the help of the ALS model. The algorithm is already discussed in section 3 of this thesis. The parameters as shown in Table 5.1 are used to train the ALS model. The ALS model is

used to generate 80,000 publisher ratings for each advertiser. The threshold of 80,000 was set after running several instances to get the best recommendations ratings. The output generated by this ALS model is shown in Fig.5.7. These ratings are then joined with the advertiser's and publisher's data that is filtered in the previous step.

Parameters	Value	Description
userCol	AdvertiserKey	Users in the Recommendation system
itemCol	PublisherKey	Items in the Recommendation system
ratingCol	rating	Implicit calculated rating
nonnegative	TRUE	Use/Dont use non-negative constraints for least squares
implicitPrefs	TRUE	Use/Dont ALS variant adapted for implicit feedback data
coldStartStrategy	drop	Drop any rows in the DataFrame of predictions that contain NaN values
rank	15	Number of latent factors in the model
maxIter	30	maximum number of iterations to run
regParam	1.0	The regularization parameter in ALS
alpha	20	Governs the baseline confidence for implicit feedback

Table 5.1: Parameters used for the ALS Model

	AdvertiserKey	PublisherKey	rating
1	463	53163	0.8172706
2	463	170760	0.7757128
3	463	52900	0.75483114
4	463	188216	0.73911536
5	463	154037	0.73569655
6	463	67225	0.7291111
7	463	57130	0.71004117

Figure 5.7: Advertiser Publisher Ratings through ALS

5.7 Policy Definition

Advertisers have policy definitions for the publishers they want to work with. In this step, we have created a many-to-one relationship of a Publisher Sub Vertical with an Advertiser Policy Definition as shown in Fig.5.8. This relationship has been made based on data groups we already have. This is the same relationship used in the existing model of the publisher recommendation system.

PublisherSubVertical	AdvertiserPolicydefinition
Retargeting Display	BehaviouralRetargeting
Cashback	Cashback
UGC & Social Platforms	Community
Comparison Sites	Content
Editorial	
Shopping Content	
Sub Networks	
Comparison Shopping Service (CSS)	
Content Creators & Influencers	
Discount & Promotions	DiscountCode
E-Mail	Email
Loyalty	Loyalty
Social Rewards	
Media Brokers	MediaBrokers
Ad Networks	
SEM	Search

Figure 5.8: Advertiser Policy Definition and Publisher Sub Verticals Mapping

5.8 Feedback Score

The recommendation system needs a feedback system to work accurately. So, in this step, we consider the feedback score and eliminate the publisher recommendations for an advertiser whose rate is less than equal to 2. When the advertisers were presented with publisher recommendations, they have the option to rate the publisher according to their relevance. If the advertiser rates the publisher less than equal to 2, it's removed from the list of recommendations and then the list of recommendations is forwarded to the next step.

5.9 Final Publisher Recommendations

The commutation of the recommendation system is done in the above steps. Now, the final publisher recommendations for each advertiser are sorted in descending order based on the advertiser publisher ratings as shown in Fig.5.9. The results are saved back to the database.

AdvertiserKey	PublisherKey	rating	Rank
2738	169276	0.88105106	1
2738	99970	0.8475774	1
2738	3396823	0.77533364	1
2738	376322	0.7119581	1
2738	6317913	0.7116713	1
2738	371400	0.6597692	1
2738	6374279	0.64574546	1
2738	381079	0.6367331	1
2738	6250465	0.6275494	1
2738	7090365	0.55209464	1
2738	188919	0.5313114	1
2738	6087483	0.52250063	1
2738	3329046	0.51322144	1

Figure 5.9: Final Result of Publisher Recommendation

Chapter 6

Results

In this chapter, we discuss the computing environment used by the model during this thesis. Other than this, we also discuss the evaluation metrics to evaluate our model over the existing model. We use MAE, RMSE, and manual evaluation feedback.

6.1 Computing Environment

In this thesis, the model was implemented on Microsoft Azure Databricks [Eta19]. An analytics service called Databricks is built on the Apache Spark open-source initiative. Apache Spark is a platform for batch and real-time processing. Because of its capacity to analyze enormous volumes of data, streaming capabilities, graph processing, machine learning, and interactive query engine, Apache Spark is highly well-liked among data scientists. Cluster computing in memory is offered by Spark. A brand-new platform for big data analytics and machine learning is called Azure Databricks. Data engineers, data scientists, and business analysts can all work together on a single tool thanks to the notebook in Azure Databricks. The cluster used in the Azure Databricks had 256-1024 GB Memory and 32-128 Cores for workers. The driver consists of 256 GB of Memory and 32 Cores. The Runtime was 11.3x-cpu-ml-scala2.12.

The functions were coded in Apache Spark and Python. Apache Spark [BJK18] is a respectable distributed memory-based computing system for machine learning. In terms of computing, Spark is better than Hadoop. Apache Spark is a rapid, easy-to-use tool for managing large amounts of data that has been tested with graph processing, streaming SQL, and machine learning modules. Using Spark and its machine learning library MLlib, we can simply apply machine learning algorithms to massive data, making it straightforward. The Python API PySpark further simplifies this process. Python [van06] is an interpreted, object-oriented programming language that can be extended. It supports a wide range of applications, including interactive web browsers and basic text processing routines. The standard library that is given with the language and significantly improves its immediate usability is not covered in the Python Reference Manual, even though it contains a detailed description of the syntax and semantics of the language.

6.2 Evaluation Metrics

The evaluation of recommendation system will be evaluated by three metrics including Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and manual feedback from the business analysts. The most popular assessment metrics in the realm of recommendations are MAE and RMSE. In a research work [ZMB15], the authors utilized RMSE to assess their unique Java-Based Context-aware Recommendation algorithms while the authors of [CYL15] employed RMSE to assess music recommendation in a real-world scenario. The collaborative filtering with facial expressions for online video recommendation was evaluated in [16] using MAE. The fuzzy preference tree-based recommender system for the customized Business-to-Business E-services was also evaluated by [WZL15] using MAE. In general, RMSE and MAE are straightforward to compute and comprehend.

According to article [CD14], it is proposed that the MAE would be a superior metric for that purpose because the RMSE is not a good predictor of average model performance and may be a deceptive indicator of average error. When model errors follow a normal distribution, RMSE is more acceptable to employ than MAE since it is unambiguous in its meaning. Anyone metric can only project the model errors in one direction, and as a result, it can only highlight one particular feature of the error characteristics. Therefore, we assess our model using both RMSE and MAE. The outcomes of the recommendations have been compared with those of the existing, non-clustered recommendation approach.

6.2.1 Mean Absolute Error (MAE)

A statistic called mean absolute error is used to calculate the average of all absolute value discrepancies between the actual and anticipated rating. The accuracy increases with a decrease in MAE. According to the rating system of the application being assessed, the MAE can often vary from 0 to infinity, with infinity representing the highest mistake. Additionally, according to some studies, if the rating scores are integers, rounding the prediction findings will lower the MAE error. The following formula as stated in [Raj+22] can be used to calculate the Mean Absolute Error :

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i|$$

where:

y_i : actual value of the i-th rating,

$\hat{y}_i$: predicted value of the i-th rating,

n : total number of ratings.

From Fig.6.1, we can see that MAE is smaller for the new model than the existing model. The smaller the values, the more accurately the model works and the better the recommendations.

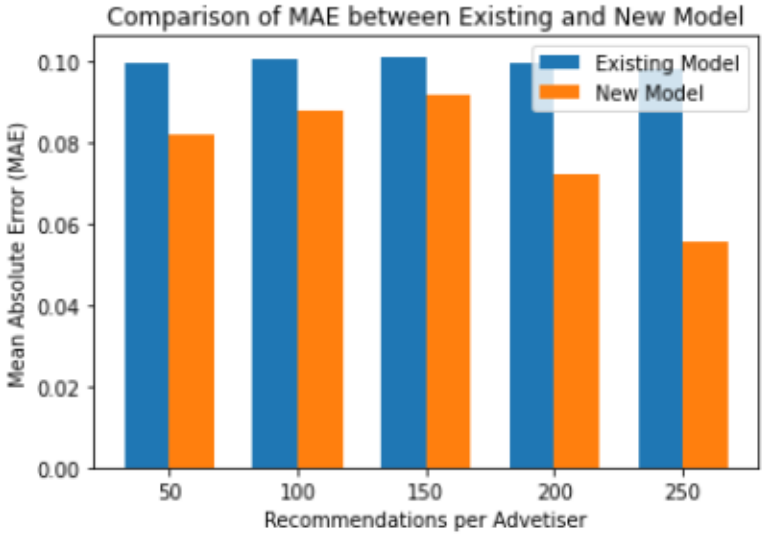

Figure 6.1: Comparison of MAE between Existing and New Model

6.2.2 Root Mean Square Error (RMSE)

In order to get the square root of the result, the root mean square error first computes the mean value of all the squared differences between the true and predicted ratings. Big errors may consequently have a considerable impact on the RMSE rating, making the RMSE metric most beneficial when noticeably big errors are undesirable. The RMSE evaluation approach, as opposed to MAE, places more attention on larger absolute errors. The smaller the RMSE value is, the better accurate the recommendation results will be. The following formulas as stated in [SDK22] are used to calculate the Root Mean Square Error between true and expected ratings:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

where:

y_i : actual value of the i-th rating,

$\hat{y}_i$: predicted value of the i-th rating,

n : total number of ratings.

From Fig.6.2, we can see that RMSE is almost the same for both of the models i.e. for existing and new ones. For both of the models, the RMSE is smaller, indicating both models have good recommendations.

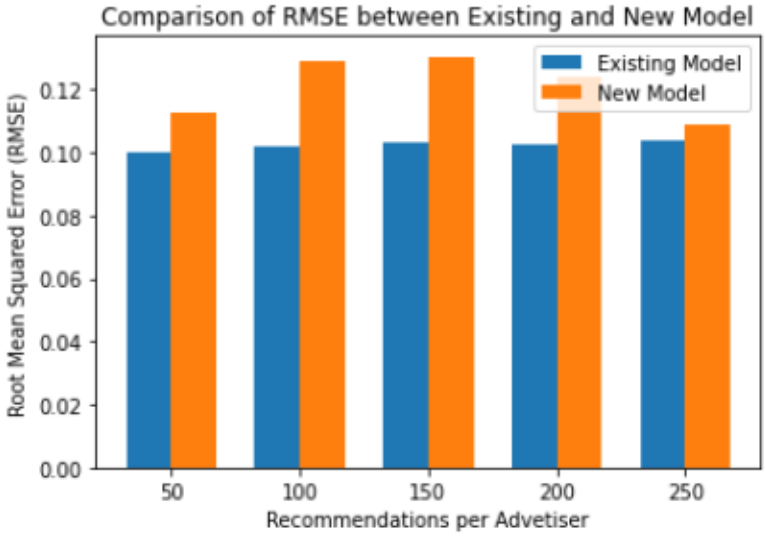

Figure 6.2: Comparison of RMSE between Existing and New Model

6.2.3 Manual Evaluation

The recommendations for existing and new models were sent out to Product Team (who works closely with advertisers and publishers) at AWIN. We have sent them the top 10 recommendations for 6 advertisers to compare. The team manually checked the recommendations across most of the Advertisers sent, the content on their website was relevant for most Advertisers, and most of the publisher recommendations for the new model had a decent level of performance (not super high but a good mixture) and higher than the existing recommendations. Also, they compared the lists to the top 30,000 publishers, In new recommendations, 56/60 (consisting of 10 recommendations for 6 advertisers i.e. 60) were in the top 30,000 compared to 30/60 in the existing recommendations.

Chapter 7

Conclusion

In this thesis, we implemented the K means clustering with the ALS model in which the clusters were divided based on sectors of publishers, and advertisers were joined with publishers based on the sector clusters. Besides this implicit ratings were used with the ALS model to generate more data and then ingested with advertiser publisher results to give the final results with top recommendations based on ratings in descending order. The proposed method which was developed based on K means clustering overcomes problems like data sparsity, scalability, and effectiveness.

With the overall evaluation using MAE, RMSE, and Manual evaluation feedback, it is evident that the new recommendations generated through the K mean clustering and ALS models together give better publisher recommendations compared to existing ones. This satisfies the aim of the thesis to present a Machine Learning model that works better and more accurately than the existing model. Additionally, from a business perspective, this thesis presents a solution for advertisers with publisher recommendations with better interests, the same working sectors, and regions. The method assists advertisers in making decisions, boosts the volume of company sales, and improves advertiser satisfaction and royalties.

Chapter 8

Future Work

The recommendation system was implemented using the K-Means clustering method but every model has their drawbacks like finding an optimal cluster. In the future, more work can be done by analyzing the data of publishers and advertisers to make the system more effective. There can be new features or parameters included in the implicit rating calculation (recommendation system ratings) as its being an important feature of the system. Experiment with different algorithms like deep learning (neural networks if possible), decision trees, and other clustering methods. A study on the mentioned models with a comparison to the proposed recommendation system method can also be implemented.

Bibliography

Books

[RP19] S. Raghuwanshi and R. Pateriya. *Recommendation Systems: Techniques, Challenges, Application, and Evaluation: SocProS 2017, Volume 2*. Jan. 2019, pp. 151–164. DOI: 10.1007/978-981-13-1595-4_12.

[Neg21] A. Negro. *Graph-Powered Machine Learning*. 1st. 2021.

[MW07] C. Ma and J. Wu. *Data Clustering: Theory, Algorithms, and Applications*. Vol. 20. Jan. 2007. DOI: 10.1137/1.9780898718348.

[Eta19] L. Etaati. *Azure Databricks*. June 2019, pp. 159–171. DOI: 10.1007/978-1-4842-3658-1_10.

[van06] G. van. *Python Library Reference*. Jan. 2006.

Articles

[YLA21] A. Yassine, M. LAZAAR, and M. Al Achhab. "Intelligent recommender system based on unsupervised machine learning and demographic attributes." In: *Simulation Modelling Practice and Theory* 107 (Feb. 2021), p. 102198. DOI: 10.1016/j.simpat.2020.102198.

[KJC20] S. Kang, C. Jeong, and K. Chung. "Tree-Based Real-Time Advertisement Recommendation System in Online Broadcasting." In: *IEEE Access* 8 (Jan. 2020), pp. 192693–192702. DOI: 10.1109/ACCESS.2020.3031925.

[Hao21] H. Hao. "An English Online Homework Tutoring System Supported by Internet Database." In: *Journal of Mathematics* 2021 (Dec. 2021), pp. 1–12. DOI: 10.1155/2021/5960185.

[AT05] G. Adomavicius and A. Tuzhilin. "Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions." In: *IEEE Transactions on Knowledge and Data Engineering* 17.6 (2005), pp. 734–749. DOI: 10.1109/TKDE.2005.99.

[Ko+22] H. Ko et al. "A Survey of Recommendation Systems: Recommendation Models, Techniques, and Application Fields." In: *Electronics* 11.1 (Jan. 2022), p. 141. DOI: 10.3390/electronics11010141.

[Met00] R. Meteren. "Using Content-Based Filtering for Recommendation." In: (June 2000).

[Gol+92] D. Goldberg et al. "Using collaborative filtering to weave an information tapestry." In: *Commun. ACM* 35 (1992), pp. 61–70.

[BHK13] J. S. Breese, D. Heckerman, and C. M. Kadie. "Empirical Analysis of Predictive Algorithms for Collaborative Filtering." In: *CoRR* abs/1301.7363 (2013). arXiv: 1301.7363.

[TS15] S. K. Tiwari and S. K. Shrivastava. "An Approach for Recommender System by Combining Collaborative Filtering with User Demographics and Items Genres." In: *International Journal of Computer Applications* 128.13 (Oct. 2015). Published by Foundation of Computer Science (FCS), NY, USA, pp. 16–24.

[Ras+06] A. Rashid et al. "ClustKNN: a highly scalable hybrid model- and memory-based CF algorithm." In: (Sept. 2006).

[RSA22] K. Rrmoku, B. Selimi, and L. Ahmedi. "Application of Trust in Recommender Systems - Utilizing Naive Bayes Classifier." In: *Computation* 10.1 (Jan. 2022), p. 6. DOI: 10.3390/computation10010006.

[Pan+20] C. Pandian et al. "Recommendation System Using Naive Bayes Classifier." In: *International Research Journal of Engineering and Technology (IRJET)* 07 (Mar. 2020).

[SNP17] S. Sahu, A. Nautiyal, and M. Prasad. "Machine Learning Algorithms for Recommender System - a comparative analysis." In: *International Journal of Computer Applications Technology and Research* 6 (Feb. 2017), pp. 97–100. DOI: 10.7753/IJCATR0602.1005.

[BGB15] P. B.Thorat, R. Goudar, and S. Barve. "Survey on Collaborative Filtering, Content-based Filtering and Hybrid Recommendation System." In: *International Journal of Computer Applications* 110 (Jan. 2015), pp. 31–36. DOI: 10.5120/19308-0760.

[IFO15] F. Isinkaye, Y. Folajimi, and B. Ojokoh. "Recommendation systems: Principles, methods and evaluation." In: *Egyptian Informatics Journal* 16.3 (2015), pp. 261–273. DOI: https://doi.org/10.1016/j.eij.2015.06.005.

[Jai10] A. Jain. "Data Clustering: 50 Years Beyond K-Means." In: *Pattern Recognition Letters* 31 (June 2010), pp. 651–666. DOI: 10.1016/j.patrec.2009.09.011.

[BJK18] R. Bandi, A. Jayavel, and R. Karthik. "Machine Learning with PySpark - Review." In: *Indonesian Journal of Electrical Engineering and Computer Science* 12 (Oct. 2018), pp. 102–106. DOI: 10.11591/ijeecs.v12.i1.pp102-106.

[16] "Collaborative filtering with facial expressions for online video recommendation." In: *International Journal of Information Management* 36.3 (2016), pp. 397–402. DOI: https://doi.org/10.1016/j.ijinfomgt.2016.01.005.

[WZL15] D. Wu, G. Zhang, and J. Lu. "A Fuzzy Preference Tree-Based Recommender System for Personalized Business-to-Business E-Services." In: *Trans. Fuz Sys.* 23.1 (Feb. 2015), pp. 29–43. DOI: 10.1109/TFUZZ.2014.2315655.

[CD14] T. Chai and R. Draxler. "Root mean square error (RMSE) or mean absolute error (MAE)? Arguments against avoiding RMSE in the literature." In: *Geoscientific Model Development* 7 (June 2014), pp. 1247–1250. DOI: 10.5194/gmd-7-1247-2014.

Proceedings

[SAK21] M. 'Sharma, L. Ahuja, and V. Kumar. "'Study and Classification of Recommender Systems: A Survey'." In: *'International Conference on Innovative Computing and Communications'*. Ed. by D. 'Gupta et al. 'Springer Singapore', 2021, pp. 1153–1163.

[Sar+01] B. Sarwar et al. "Item-Based Collaborative Filtering Recommendation Algorithms." In: *Proceedings of the 10th International Conference on World Wide Web*. Hong Kong, Hong Kong: Association for Computing Machinery, 2001, pp. 285–295. DOI: 10.1145/371920.372071.

[Hil+95] W. Hill et al. "Recommending and Evaluating Choices in a Virtual Community of Use." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '95. Denver, Colorado, USA: ACM Press/Addison-Wesley Publishing Co., 1995, pp. 194–201. DOI: 10.1145/223904.223929.

[KB10] M. Kompan and M. Bieliková. "Content-Based News Recommendation." In: *E-Commerce and Web Technologies*. Ed. by F. Buccafurri and G. Semeraro. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 61–72.

[Sch+07] B. Schafer et al. "Collaborative Filtering Recommender Systems." In: Jan. 2007.

[Das+07] A. S. Das et al. "Google News Personalization: Scalable Online Collaborative Filtering." In: *Proceedings of the 16th International Conference on World Wide Web*. New York, NY, USA: Association for Computing Machinery, 2007, pp. 271–280. DOI: 10.1145/1242572.1242610.

[BCT13] K. Babas, G. Chalkiadakis, and E. Tripolitakis. "You Are What You Consume: A Bayesian Method for Personalized Recommendations." In: RecSys '13. Hong Kong, China: Association for Computing Machinery, 2013, pp. 221–228. DOI: 10.1145/2507157.2507158.

[Yu+02] K. Yu et al. "Instance Selection Techniques for Memory-Based Collaborative Filtering." In: *Proc. of the Second Siam Intl. Conf. on Data Mining, SDM*. 2002.

[Cho+12] Y. S. Cho et al. "Implementation of personalized recommendation system using k-means clustering of item category based on RFM." In: *2012 IEEE International Conference on Management of Innovation and Technology (ICMIT)*. 2012, pp. 378–383. DOI: 10.1109/ICMIT.2012.6225835.

[BK21] I. Beregovskaya and M. Koroteev. "Review of Clustering-Based Recommender Systems." In: 2021. arXiv: 2109.12839 [cs.IR].

[DM11] G. M. Dakhel and M. Mahdavi. "A new collaborative filtering algorithm using K-means clustering and neighbors' voting." In: *2011 11th International Conference on Hybrid Intelligent Systems (HIS)*. 2011, pp. 179–184. DOI: 10.1109/HIS.2011.6122101.

[CA19] M. Cetin and S. Ayvaz. "A Negative Similarity Based Hybrid Recommender System Using Apache Spark." In: Oct. 2019, pp. 166–172. DOI: 10.1145/3369114.3369152.

[NXY10] S. Na, L. Xumin, and G. Yong. "Research on k-means Clustering Algorithm: An Improved k-means Clustering Algorithm." In: *2010 Third International Symposium on Intelligent Information Technology and Security Informatics*. 2010, pp. 63–67. DOI: 10.1109/IITSI.2010.74.

[ZMB15] Y. Zheng, B. Mobasher, and R. Burke. "CARSKit: A Java-Based Context-Aware Recommendation Engine." In: *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*. 2015, pp. 1668–1671. DOI: 10.1109/ICDMW.2015.222.

[CYL15] S. Y. Chou, Y. H. Yang, and Y. C. Lin. "Evaluating Music Recommendation in A Real-world Setting: on Data Splitting and Evaluation Metrics." In: *IEEE International Conference on Multimedia and Expo (ICME)*. IEEE. June 2015, pp. 1–6.

[Raj+22] A. S. Rajawat et al. "Chapter six - Renewable energy system for industrial internet of things model using fusion-AI." In: *Applications of AI and IOT in Renewable Energy*. Ed. by R. N. Shaw et al. Academic Press, 2022, pp. 107–128. DOI: https://doi.org/10.1016/B978-0-323-91699-8.00006-1.

[SDK22] M. Singh, R. K. Dubey, and S. Kumar. "Chapter 15 - Vehicle telematics: An Internet of Things and Big Data approach." In: *Artificial Intelligence and Machine Learning for EDGE Computing*. Ed. by R. Pandey et al. Academic Press, 2022, pp. 235–254. DOI: https://doi.org/10.1016/B978-0-12-824054-0.00019-8.

Declaration

I hereby certify that I have written this thesis independently and have not used any sources or aids other than those specified, that all posts work, literally or analogously, from other sources were adopted, are identified as such and that the work in in the same or a similar form has not yet been submitted to an examination authority.

Erlangen, den 7.Juli 2023 .....

Mohammed Quaidjohar Husain