RECOMMENDER SYSTEM USING K-NN ALGORITHM

A PROJECT REPORT

Submitted by

ARPIT MANOCHA [Reg No: RA1811003010264]
RISHABH PRAKASH [Reg No: RA1811003010272]

Under the guidance of

Mrs. Poornima S.E., Ph.D
(Professor, Department of Computer Science & Engineering)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE

of

FACULTY OF ENGINEERING AND TECHNOLOGY

SRM
UNIVERSITY
(Under section 3 of UGC Act 1956)

S.R.M. Nagar, Kattankulathur, Kancheepuram District

October 2020

SRM UNIVERSITY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that this project report titled "**RECOMMENDER SYSTEM USING K-NN ALGORITHM**" is the bonafide work of " **ARPIT MANOCHA [Reg No: RA1811003010264], RISHABH PRAKASH [Reg No: RA1811003010272]**", who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE	**SIGNATURE**
Mrs. Poornima S.E., Ph.D **GUIDE** Professor Dept. of Computer Science & Engineering	———————- **HEAD OF THE DEPARTMENT** Dept. of Computer Science
Signature of the Internal Examiner	Signature of the External Examiner

ABSTRACT

Recommendation system can be defined as a system that produces individual recommendations (a personalized way of possible options) as an output based on their previous choices which are considered an input by the system. Most of the products that we use today are powered by recommendation system. A movie recommendation is important in our social life due to its strength in providing enhanced entertainment. Such a system can suggest a set of movies to users based on their interest, or the popularities of the movies. Although, a set of movie recommendation systems have been proposed, most of these either cannot recommend a movie to the existing users efficiently or to a new user by any means. In this project we propose a recommendation system using the K-NN algorithm that has the ability to recommend movies to a new user as well as the others. It mines movie databases to collect all the important information, such as, genres, keywords, required for recommendation.Also we have analysed how much weight should be given to which feature because it will increase the efficiency of the K-NN algorithm to give the most suitable result.

ACKNOWLEDGEMENTS

We would like to express our deepest gratitude to our guide, Mrs.Poornima S.E. her valuable guidance, consistent encouragement, personal caring, timely help and providing me with an excellent atmosphere for doing research. All through the work, in spite of her busy schedule, she has extended cheerful and cordial support to us for completing this research work.

Author

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The explosive growth in the amount of available digital information and the number of visitors to the Internet have created a potential challenge of information overload which hinders timely access to items of interest on the Internet. Information Retrieval systems, such as Google, DevilFinder and Altavista have partially solved this problem but prioritization and personalization (where a system maps available content to user's interests and preferences) of information were absent. This has increased the demand for recommender systems more than ever before. Recommender systems are information filtering systems that deal with the problem of information overload by filtering vital information fragment out of large amount of dynamically generated information according to user's preferences, interest, or observed behavior about item. Recommender system has the ability to predict whether a particular user would prefer an item or not based on the user's profile.

Recommender systems are beneficial to both service providers and users. They reduce transaction costs of finding and selecting items in an online shopping environment. Recommendation systems have also proved to improve decision making process and quality. In e-commerce setting, recommender systems enhance revenue for the fact that they are effective means of selling more products. Recommender system can also be used to recommend different types of movies of different genres. Various entertainment websites such as Netflix, Amazon Prime, Hotstar, etc. uses recommendation system to predict the content which will be preferred by the user. It enhances the user's experience too.

In this model we have created a recommender engine which makes use of K-NN algorithm to find out the top 5 similar movies to the movie that the user has give as input.We have discussed in detail about the K-NN algorithm in this report.

CHAPTER 2

TYPES OF RECOMMENDATION SYSTEMS

Recommender system is defined as a decision making strategy for users under complex information environments. Also, recommender system was defined from the perspective of E-commerce as a tool that helps users search through records of knowledge which is related to users' interest and preference. Recommender system was defined as a means of assisting and augmenting the social process of using recommendations of others to make choices when there is no sufficient personal knowledge or experience of the alternatives . Recommender systems handle the problem of information overload that users normally encounter by providing them with personalized, exclusive content and service recommendations.

2.1 Content based filtering

Content-based technique is a domain-dependent algorithm and it emphasizes more on the analysis of the attributes of items in order to generate predictions. When documents such as web pages, publications and news are to be recommended, content-based filtering technique is the most successful. In content-based filtering technique, recommendation is made based on the user profiles using features extracted from the content of the items the user has evaluated in the past. Items that are mostly related to the positively rated items are recommended to the user.
CBF uses different types of models to find similarity between documents in order to generate meaningful recommendations. It could use Vector Space Model such as Term Frequency Inverse Document Frequency (TF/IDF) or Probabilistic models such as Naïve Bayes Classifier, Decision Trees or Neural Networks to model the relationship between different documents within a corpus. These techniques make recommendations by learning the underlying model with either statistical analysis or machine learning techniques.

Content-based filtering technique does not need the profile of other users since they do not influence recommendation. Also, if the user profile changes, CBF technique still has the potential to adjust its recommendations within a very short period of time. The major disadvantage of this technique is the need to have an in-depth knowledge and description of the features of the items in the profile.The information source that content-based filtering systems are mostly used with are text documents. A standard approach for term parsing selects single words from documents.

2.1.1 Merits

• There is no requirement of much of the user's data.
• We just need item data that enable us to start giving recommendations to users.
• A content-based recommender engine does not depend on the user's data, so even if a new user comes in, we can recommend the user as long as we have the user data to build his profile.
• It does not suffer from a coldstart

2.1.2 Demerits

• Items data should be in good volume.
• Features should be available to compute the similarity.

2.2 Collaborative filtering

Collaborative filtering is a domain-independent prediction technique for content that cannot easily and adequately be described by metadata such as movies and music. Collaborative filtering technique works by building a database (user-item matrix) of preferences for items by users. It then matches users with relevant interest and preferences by calculating similarities between their profiles to make recommendations. Such users build a group called neighbourhood. An user gets recommendations to those items that he has not rated before but that were already positively rated by users in his neighbourhood. Recommendations that are produced by CF can be of either prediction or recommendation. Prediction is a numerical value, Rij, expressing the predicted score of item j for the user i, while Recommendation is a list of top N items that the user will like the most. The technique of collaborative filtering can be divided into two categories: memory-based and model-based.

2.2.1 Memory based techniques

The items that were already rated by the user before play a relevant role in searching for a neighbor that shares appreciation with him. Once a neighbour of a user is found, different algorithms can be used to combine the preferences of neighbours to generate recommendations. Due to the effectiveness of these techniques, they have achieved widespread success in real life applications. Memory-based CF can be achieved in two ways through user-based and item-based techniques. User based collaborative filtering technique calculates similarity between users by comparing their ratings on the same item, and it then computes the predicted rating for an item by the active user as a weighted average of the ratings of the item by users similar to the active user where weights are the similarities of these users with the target item. Item-based filtering techniques compute predictions using the similarity between items and not the similarity between users. It builds a model of item similarities by retrieving all items rated by an active user from the user-item matrix, it determines how similar the retrieved items are to the target item, then it selects the k most similar items and their corresponding similarities are also determined. Prediction is made by taking a weighted average of the active users rating on the similar items k. Several types of similarity measures are used to compute similarity between item/user. Similarity measure is also referred to as similarity metric, and they are methods used to calculate the scores that express how similar users or items are to each other. These scores can then be used as the foundation of user- or item-based recommendation generation. Depending on the context of use, similarity metrics can also be referred to as correlation metrics or distance metrics.

2.2.2 Model-based techniques

This technique employs the previous ratings to learn a model in order to improve the performance of Collaborative filtering Technique. The model building process can be done using machine learning or data mining techniques. These techniques can quickly recommend a set of items for the fact that they use pre-computed model and they have proved to produce recommendation results that are similar to neighbourhood-based recommender techniques. Examples of these techniques include Dimensionality Reduction technique such as Singular Value Decomposition (SVD), Matrix Completion Technique, Latent Semantic methods, and Regression and Clustering. Model-based techniques analyze the user-item matrix to identify relations between items; they use these relations to compare the list of top-N recommendations. Model based techniques resolve the sparsity problems associated with recommendation systems.

2.2.3 Pros and Cons of collaborative filtering techniques

Pros

Collaborative Filtering has some major advantages over CBF in that it can perform in domains where there is not much content associated with items and where content is difficult for a computer system to analyze (such as opinions and ideal).
CF technique has the ability to provide serendipitous recommendations, which means that it can recommend items that are relevant to the user even without the content being in the user's profile.

Cons

• Cold start: For a new user or item, there isn't enough data to make accurate recommendations.
• Scalability: In many of the environments in which these systems make recommendations, there are millions of users and products. Thus, a large amount of computation power is often necessary to calculate recommendations.

• Sparsity: The number of items sold on major e-commerce sites is extremely large. The most active users will only have rated a small subset of the overall database. Thus, even the most popular items have very few ratings.

2.3 Hybrid filtering

Most recommender systems now use a hybrid approach, combining collaborative filtering, content-based filtering, and other approaches. There is no reason why several different techniques of the same type could not be hybridized. Hybrid approaches can be implemented in several ways: by making content-based and collaborative-based predictions separately and then combining them; by adding content-based capabilities to a collaborative-based approach (and vice versa); or by unifying the approaches into one model (for a complete review of recommender systems). Several studies that empirically compare the performance of the hybrid with the pure collaborative and content-based methods and demonstrated that the hybrid methods can provide more accurate recommendations than pure approaches.These methods can also be used to overcome some of the common problems in recommender systems such as cold start and the sparsity problem, as well as the knowledge engineering bottleneck in knowledge-based approaches.
Netflix is a good example of the use of hybrid recommender systems.The website makes recommendations by comparing the watching and searching habits of similar users (i.e., collaborative filtering) as well as by offering movies that share characteristics with films that a user has rated highly (content-based filtering).

Some hybridization techniques include:
• Weighted: Combining the score of different recommendation components numerically.
• Switching: Choosing among recommendation components and applying the selected one.
• Mixed: Recommendations from different recommenders are presented together to give the recommendation.
• Feature Combination: Features derived from different knowledge sources are com-

bined together and given to a single recommendation algorithm.

• Feature Augmentation: Computing a feature or set of features, this is then part of the input to the next technique.

• Cascade: Recommenders are given strict priority, with the lower priority ones breaking ties in the scoring of the higher ones.

• Meta-level: One recommendation technique is applied and produces some sort of model, which is then the input used by the next technique.

CHAPTER 3

K-NEAREST NEIGHBORS ALGORITHM

3.1 Definition

- KNN is a non-parametric method used for classification and regression.In pattern recognition, the k-nearest neighbors algorithm (k-NN) is a non-parametric method proposed by Thomas Cover used for classification and regression.In both cases, the input consists of the k closest training examples in the feature space. The output depends on whether k-NN is used for classification or regression
- k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until function evaluation. Since this algorithm relies on distance for classification, normalizing the training data can improve its accuracy dramatically.
- Both for classification and regression, a useful technique can be to assign weights to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of 1/d, where d is the distance to the neighbor.
- The neighbors are taken from a set of objects for which the class (for k-NN classification) or the object property value (for k-NN regression) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required.

3.2 Mathematics related to the Algorithm

- Assume a dataset $[(X_1, Y_1), (X_2, Y_2), ...(X_n, Y_n)]$ where $X_i \epsilon R^d$ and $Y_i \epsilon R$. We define X_i as the feature set and Y_i, as the class label.
- Given a new data point, $x \epsilon R^d$. Let,

$$d(X_1, x) < d(X_2, x) < d(X_3, x) < d(X_4, x)..... < d(X_n, x) \tag{3.1}$$

$$where,\ d(A, B) = 1 - \frac{A.B}{||A||\ ||B||} \tag{3.2}$$

we can produce a label y for x such that,

$$y = mode(Y_1, Y_2, Y_3, ..., Y_n) \tag{3.3}$$

- In our problem, we don‘t have class labels (Y) and it is a rank ordering problem where we are ordering our movies, books or any other stuff by comparing the features of the input data point to all the movies in the dataset.
- We define d(a,b) as the distance between the two coordinates in the plot of the features that we are going to use. It can be calculated using any suitable distance measure. The three approaches that we studied are:
  - Cosine Distance
  - Jaccard Similarity
  - Euclidean Distance
- We have chosen cosine distance in our implementation since this is a popularly used distance measure which can be used to compare text documents by looking for common words between the documents. In our implementation, the features are textual in nature and thus, we assessed this measure to be the most appropriate.

3.2.1 Euclidean Distance

The Euclidean distance between two points, the length of a line segment between the two points. It can be calculated from the Cartesian coordinates of the points using the equation:

$$d(p,q) = \sqrt{(p_1 - q1)^2 + (p_2 - q2)^2 + ... + (p_i - qi)^2 + ... + (p_n - qn)^2} \tag{3.4}$$

3.2.2 Cosine Distance

We can calculate the similarity between two vectors using the cosine of the angle between the two vectors, vectors of A and B. If the vectors are closer, then small will be the angle and large will be the cosine. Since the cosine similarity ranges [0, 1], we can calculate the distance using 1 - Cosine Similarity. Cosine Distance can be computed using:

$$d(A,B) = 1 - \frac{A.B}{||A||\,||B||} \tag{3.5}$$

3.2.3 Jaccard Similarity

The Jaccard Similarity measures similarity between finite sample sets, and is defined as the size of the intersection divided by the size of the union of the sample sets. We can calculate Jaccard Similarity as:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{3.6}$$

3.2.4 Pros and Cons of the Algorithm

Pros

- K-NN is pretty intuitive and simple: K-NN algorithm is very simple to understand and equally easy to implement. To classify the new data point K-NN algorithm reads through whole dataset to find out K nearest neighbors.
- K-NN has no assumptions: K-NN is a non-parametric algorithm which means there are no assumptions to be met to implement K-NN. Parametric models like linear regression has lots of assumptions to be met by data before it can be implemented which is not the case with K-NN.
- No Training Step: K-NN does not explicitly build any model, it simply tags the new data entry based learning from historical data. New data entry would be tagged with majority class in the nearest neighbor.
- It constantly evolves: Given it's an instance-based learning; k-NN is a memory-based approach. The classifier immediately adapts as we collect new training data. It allows the algorithm to respond quickly to changes in the input during real-time use.
- Very easy to implement for multi-class problem: Most of the classifier algorithms are easy to implement for binary problems and needs effort to implement for multi class whereas K-NN adjust to multi class without any extra efforts.
- Can be used both for Classification and Regression: One of the biggest advantages of K-NN is that K-NN can be used both for classification and regression problems.
- One Hyper Parameter: K-NN might take some time while selecting the first hyper parameter but after that rest of the parameters are aligned to it.

Cons

- K-NN slow algorithm: K-NN might be very easy to implement but as dataset grows efficiency or speed of algorithm declines very fast.
- Curse of Dimensionality: KNN works well with small number of input variables but as the numbers of variables grow K-NN algorithm struggles to predict the output of new data point.

- K-NN needs homogeneous features: If you decide to build k-NN using a common distance, like Euclidean or Manhattan distances, it is completely necessary that features have the same scale, since absolute differences in features weight the same, i.e., a given distance in feature 1 must means the same for feature 2.
- Optimal number of neighbors: One of the biggest issues with K-NN is to choose the optimal number of neighbors to be consider while classifying the new data entry.
- Imbalanced data causes problems: k-NN doesn't perform well on imbalanced data. If we consider two classes, A and B, and the majority of the training data is labeled as A, then the model will ultimately give a lot of preference to A.
- This might result in getting the less common class B wrongly classified. Outlier sensitivity: K-NN algorithm is very sensitive to outliers as it simply chose the neighbors based on distance criteria.
- Missing Value treatment: K-NN inherently has no capability of dealing with missing value problem.

CHAPTER 4

MODEL METHODOLOGY

4.1 Dataset

- Source of our dataset is: https://github.com/codeheroku/Introduction-to-Machine-Learning/blob/master/Building
- This dataset contains the following features of a movie:
  - index
  - budget
  - genre
  - homepage
  - id
  - keywords for the movie
  - original language
  - original title
  - overview
  - popularity
  - production companies
  - production countries
  - revenue
  - movie length
  - spoken language
  - tagline
  - vote average
  - vote count
  - cast
  - crew
  - director
- The movie dataset contains a dataset of 5808 movies.
- The dataset is cleaned too before using it for the project to avoid error due to dataset.

4.2 Approach

- For our movie prediction we have used 4 features which are Genres, Director ,cast and keywords.
- We have also analysed that which feature have how much effect on the movie prediction by the recommendation engine, by changing the weightage of each of the features one by one and taking various cases each giving different weightage to different features.
- The algorithm we have used for this recommendation problem i.e. a rank ordering problem, is KNN Algorithm in which the similarity between points is calculated using cosine distance.
- Our recommendation model finds out the nearest 10 movies to the movie entered as input by the user using coordinates of different movies(calculated using their features) and print them in order.

CHAPTER 5

FEATURE SELECTION

- We have selected 4 features from the dataset of the movie that are Genres, Director, Cast and keywords based on which we are trying to predict the similar movies to the movie the user has entered.
- We have tested the importance of the selected features for the KNN algorithm.
- To test the most suitable weightage that should be given to different features we did some observations and analysis on some movie sample.
- In our observation we selected a sample movie from the dataset and created a list of 3 movies recommendations based on subjective oversight and google search results.
- For Example, we took a sample movie named: Man Of Steel which have the following different features:
  - Title: Man Of Steel
  - Genre: Action Adventure Fantasy Science Fiction
  - Director Name: Zack Snyder
  - Cast: Henry Cavill Amy Adams Michael Shannon Kevin Costner Diane Lane
  - Keywords: saving the world dc comics superhero based on comic book superhuman
- Expected Recommendations: Superman, Batman Vs Superman, Watchmen. These movies were selected as expected due to their similarity with the movie, Man Of Steel and also they were recommended by the google recommendation Engine.
- We expect the above movies to be included in the top 10 recommendations by the model, especially among the top 5. So, we have defined the following error metrics to test the model performance,

$$Average\ Rank = \frac{Rank(Movie_1) + Rank(Movie_2) + Rank(Movie_3)}{3} \tag{5.1}$$

$$Top\ 5\ Rate = \frac{Number\ of\ expected\ movies\ in\ top\ 5 * 100}{3} \tag{5.2}$$

  We define a default rank of 100 if a movie isn't selected in top 10 to penalize the recommendation.

- We excluded each feature and analyzed the impact on Average Rank and Top 5 rate. It was observed that both the metrics declined on exclusion of these features except for the feature, "Cast". The results can be observed in figures 5.1 and 5.2.

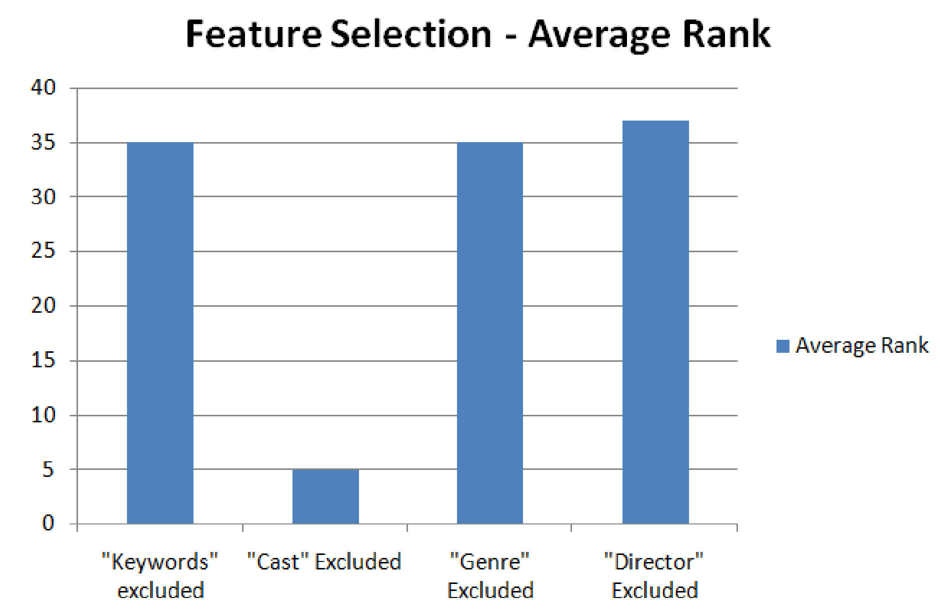


Figure 5.1: Impact of feature exclusion on Average Rank.

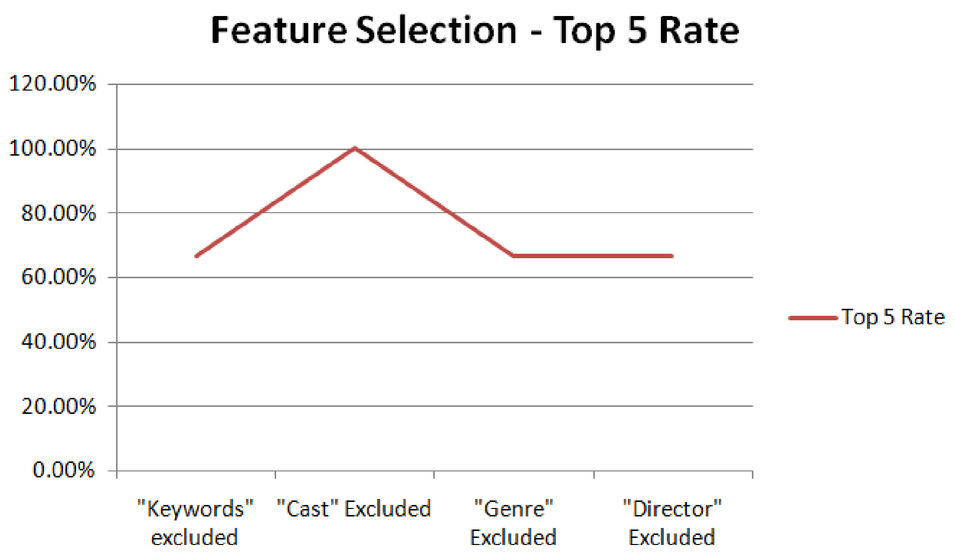
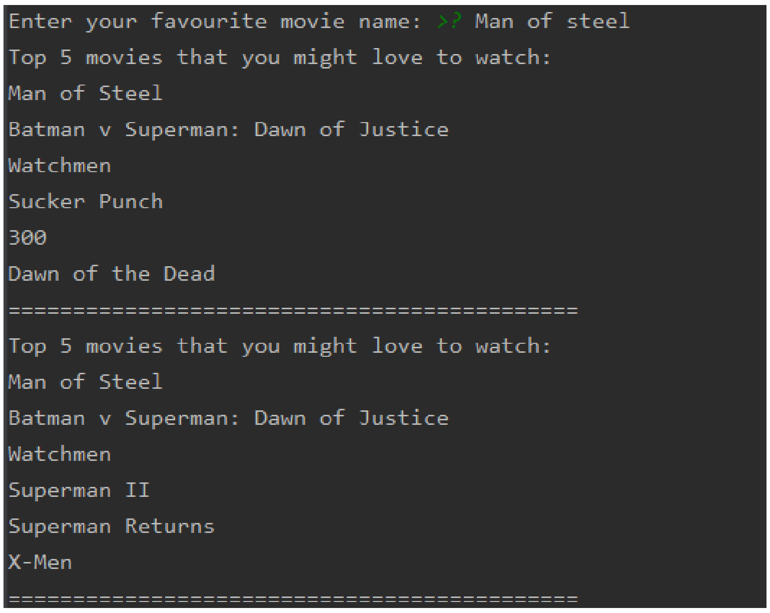


Figure 5.2: Impact of feature exclusion on top 5 rate.



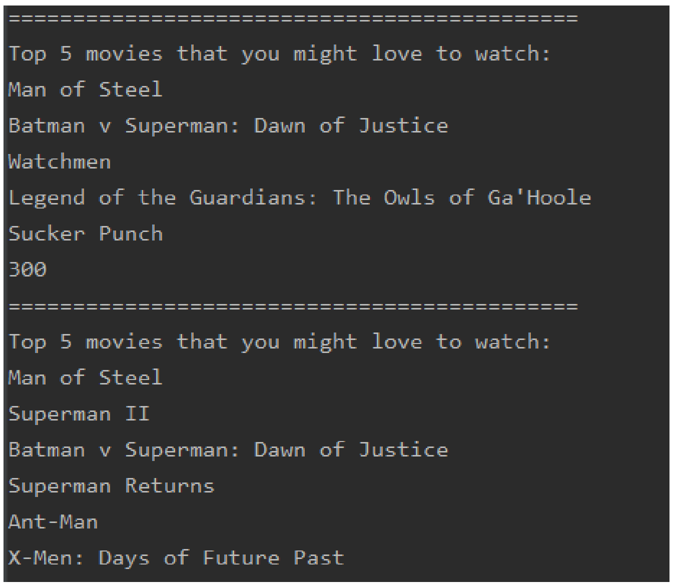
```
Enter your favourite movie name: >? Man of steel
Top 5 movies that you might love to watch:
Man of Steel
Batman v Superman: Dawn of Justice
Watchmen
Sucker Punch
300
Dawn of the Dead
==========================================
Top 5 movies that you might love to watch:
Man of Steel
Batman v Superman: Dawn of Justice
Watchmen
Superman II
Superman Returns
X-Men
==========================================
```

Figure 5.3: Movie recommendations on exclusion of "Keyword" and "Cast".

- The model recommendations on exclusion of these features are also provided for reference. Assessment of model recommendations are provided below:
  - Exclusion of "Keyword": The unique recommendations for this scenario, "300", "Sucker Punch" and "Dawn of the Dead" were primarily similar to the test movie in terms of the features, "Director" and "Genre". Due to the exclusion, these features had higher weights than normal. Please refer to figure 5.3 for the list of recommendations.
  - Exclusion of "Cast": The recommendations on this exclusion are impacted equally by "Director", "Keyword" and "Genre". Please refer to figure 5.3 for the list of recommendations.
  - Exclusion of "Genre": This scenario lead to the inclusion of the movie, "Legend of the Guardians: The Owls of Ga'Hoole", which we have assessed to be noisy. Even though it's a fantasy film, it's different from the sample movie. Please refer to figure 5.4 for the list of recommendations.
  - Exclusion of "Director": Recommended movies are impacted heavily by genre and keywords. Please refer to figure 5.4 for the list of recommendations.

- We created different scenarios by weighing each feature differently. The results from the analysis can be observed in 5.5 and 5.6. We have observed that the features "Keyword" and "Director" have higher than usual impact on the error metrics.

- Focusing on the conclusion of analysis we found out the features namely: "keyword" and "director" have higher than usual impact on the K-nearest neighbor algorithm and therefore they should be assigned higher weights. The impact of exclusion of the feature "Cast" didn't seem to have a major impact but we have chosen to keep this feature and assign a lower weight to it.



```
============================================
Top 5 movies that you might love to watch:
Man of Steel
Batman v Superman: Dawn of Justice
Watchmen
Legend of the Guardians: The Owls of Ga'Hoole
Sucker Punch
300
============================================
Top 5 movies that you might love to watch:
Man of Steel
Superman II
Batman v Superman: Dawn of Justice
Superman Returns
Ant-Man
X-Men: Days of Future Past
```

Figure 5.4: Movie recommendations on exclusion of "Keyword" and "Cast".

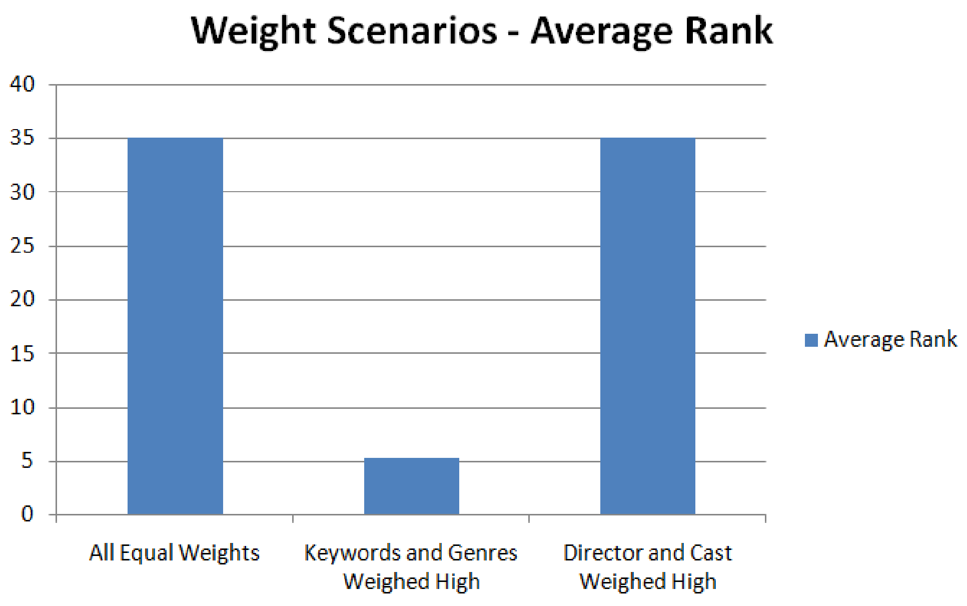


Figure 5.5: Average Rank for different weight scenarios.

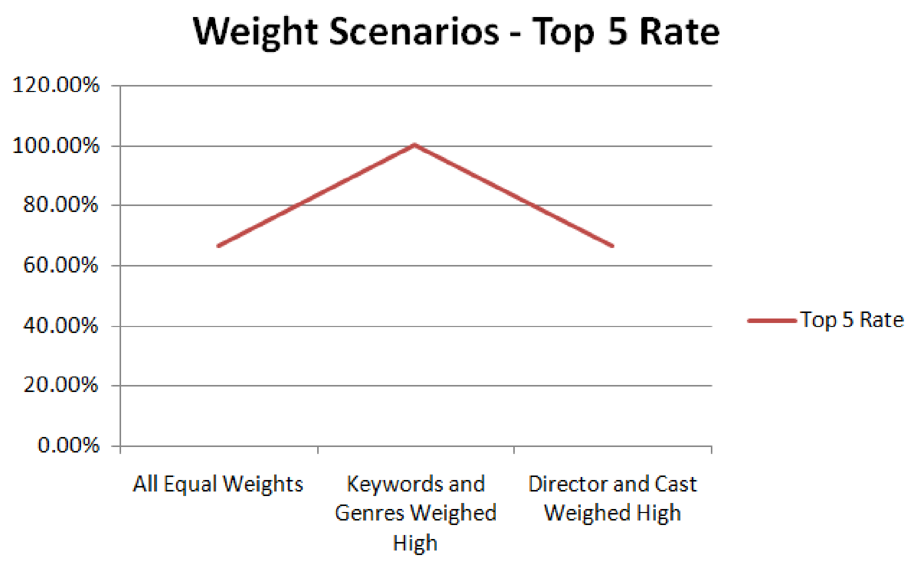


Figure 5.6: Top 5 rate for different weight scenarios.

CHAPTER 6

INFERENCES/CONCLUSION

- The best choice of k depends upon the data; generally, larger values of k reduces effect of the noise on the classification,[8] but make boundaries between classes less distinct.
- K-NN Algorithm is one of the most suitable approach for rank-ordering problems and also it is widely accepted.
- Feature Selection is one of the most important steps involved in building a machine learning algorithm as we analysed in our model.
- We observed a significant impact to the model output on inclusion/exclusion of certain features.
- We also applied weights to the features to understand their relative importance. This helped in improving the model accuracy as we saw in our graph.
- Distance Measures: We studied about the different distance measures for the purpose of this project.

CHAPTER 7

FUTURE ENHANCEMENTS

- K-NN slow algorithm: K-NN might be very easy to implement but as dataset grows efficiency the speed of algorithm declines very fast.
- Optimal number of neighbors: One of the biggest issues with K-NN is to choose the optimal number of neighbors to be consider while classifying the new data entry.
- Imbalanced data causes problems: k-NN doesn't perform well on imbalanced data. If we consider two classes, A and B, and the majority of the training data is labeled as A, then the model will ultimately give a lot of preference to A. This might result in getting the less common class B wrongly classified.
- Cosine similarity works best when there are a great many (and likely sparsely populated) features to choose from. Under these conditions, Euclidean methods tail off in terms of their sensitivity. Likewise, cosine similarity is less optimal under spaces of lower dimension.
- The selection of features is the most important step for this type of model, if there is any error in selection of features that might decrease the efficiency of the model.
- It is very important to assign correct weightage to each of the feature being used as it affects the prediction strongly.
- For weightage selection we need to analyse some sample data like we did in our project hence it takes little longer time to assign the correct weightage.

CHAPTER 8

REFERENCES

1. Sang-Ki Ko, Sang-Min Choi, Hae-Sung Eom , Jeong-Won Cha, Hyunchul Cho, Laehyum Kim, and Yo-Sub Han :
   A Smart Movie Recommendation System Content-based method uses item-to-item similarity. If a user like B, we recommend A that is similar to B. Association method also uses item-to-item similarity. In this method, we do not decide whether actually they are similar or not. If items have high correlation with each other, we decide that they are similar. Demographic method and collaborative method use people-to- people similarity both. Demographic method needs actual features of people to decide whether they are similar. Collaborative method uses correlation between users. They propose a movie recommendation system based on genre correlation. Choi and Han suggested the method that users should input their favourite movie genres into the recommendation system manually and the system calculates the recommendation points. On the other hand, their recommendation system uses movie lists as input and obtains the genres of movies in lists and thus the preferences of users. (They profile movie preferences of users.) This step assumes that users would prefer genres appeared in the list to other genres.

2. Yibo Wang, Mingming Wang, and Wei Xu :
   A Sentiment-Enhanced Hybrid Recommender System for Movie Recommendation As mentioned before, this paper uses collaborative filtering and content-based hybrid recommender systems. Collaborative filtering and content-based approaches can compensate for the shortcomings of each other, thus ensuring the accuracy and stability of the recommender system. On the one hand, collaborative filtering can make up for the lack of personalization of content-based method; on the other hand, content-based method can make up for the flaw of collaborative filtering method whose scalability is relatively weak. In general, the hybrid recommendation method is first executed based on user data and movie data to achieve a preliminary recommendation list. Then sentiment analysis is implemented to optimize the preliminary list and get the final recommendation list. Furthermore, on the basis of the hybrid recommendation framework, this paper fully considers the efficiency of the recommender system. In the process of recommending movies, this paper focuses on the user's reviews on movies.

3. YuanHuang: A news recommendation engine driven by collaborative reader behavior
   As a first step, the engine identifies readers with similar news interests based on their behavior of retweeting articles posted on Twitter. The data was collected from articles posted on Twitter by three different publishers (New York Times, Washington Post, and Bloomberg) between May 23rd and June 2nd, 2017. I also collected information from the Twitter profiles of the users retweeting the articles.

By looking at how many news posts two users share in common, I can define a cosine similarity score for the users. This similarity score enables the construction of a network by weighting the links between users.

4. J. Ben Schafer , Dan Frankowski , Jon Herlocker , and Shilad Sen :

   Collaborative Filtering Recommender Systems One of the potent personalization technologies powering the adaptive web is collaborative filtering. Collaborative filtering (CF) is the process of filtering or evaluating items through the opinions of other people. CF technology brings together the opinions of large interconnected communities on the web, supporting filtering of substantial quantities of data. In this chapter we introduce the core concepts of collaborative filtering, its primary uses for users of the adaptive web, the theory and practice of CF algorithms, and design decisions regarding rating systems and acquisition of ratings. We also discuss how to evaluate CF systems, and the evolution of rich interaction interfaces. We close the chapter with discussions of the challenges of privacy particular to a CF recommendation service and important open research questions in the field.Over the past decade, collaborative filtering algorithms have evolved from research algorithms intuitively capturing users' preferences to algorithms that meet the performance demands of large commercial applications.

5. Khalid Haruna,Maizatul Akmar Ismail ,Damiasih Damiasih,Joko Sutopo,Tutut Herawan :

   A collaborative approach for research paper recommender system Research paper recommenders emerged over the last decade to ease finding publications relating to researchers' area of interest. The challenge was not just to provide researchers with very rich publications at any time, any place and in any form but to also offer the right publication to the right researcher in the right way. Several approaches exist in handling paper recommender systems. However, these approaches assumed the availability of the whole contents of the recommending papers to be freely accessible, which is not always true due to factors such as copyright restrictions. This paper presents a collaborative approach for research paper recommender system. By leveraging the advantages of collaborative filtering approach, we utilize the publicly available contextual metadata to infer the hidden associations that exist between research papers in order to personalize recommendations. The novelty of our proposed approach is that it provides personalized recommendations regardless of the research field and regardless of the user's expertise. Using a publicly available dataset, our proposed approach has recorded a significant improvement over other baseline methods in measuring both the overall performance and the ability to return relevant and useful publications at the top of the recommendation list.

6. Mark Rethana :Building a Song Recommendation System using Cosine Similarity and Euclidean Distance

   Streaming services have given us an unlimited amount of music options from tons of different eras. Personally, this has made it overwhelming to decide what to listen to when exploring different generations. To fix this I set out to build a program that would add songs to my Spotify library I was most likely to enjoy for any artist I wanted.My first step was to create a user profile for myself based off my listening history. I could then compare this profile to every song by a given artist and add the most similar songs to my library. The Spotify API lets you access all of your most listened to songs by long term, medium term and short term streams. Each list is about 100 songs long so I decided to use these 300 songs to create my user profile, since they would be an accurate depiction of what type of music I enjoy the most. I used a python library called Spotify that simplifies the process of working with the Spotify API. Below is the code I used to connect to my personal Spotify account through Spotipy, create a dictionary of all my most listened to songs andadd feature values defined by Spotify to each song.

7. Ankit Kamal KishoreKhera , Dr. Chris Tseng, Dr.SoonTee Teoh: Online Recommendation System

   The goal of this project is to study recommendation engines and identify the shortcomings of traditional recommendation engines and to develop a web based recommendation engine by making use of user based collaborative filtering (CF) engine and combining context based results along with it. The system makes use of numerical ratings of similar items between the active user and other users of the system to assess the similarity between users' profiles to predict recommendations of unseen items to active user. The system makes use of Pearson's correlation to evaluate the similarity between users. The results show that the system rests in its assumption that active 9 users will always react constructively to items rated highly by similar users, shortage of ratings of some items, adapt quickly to change of user's interest, and identification of potential features of an item which could be of interest to the user. This project will focus on making use of context based approach in addition to CF approach to recommend quality content to its users. It would be exploiting available contextual information, analyzing and summarizing user queries, and linking the metadata like tags and feedback to a richer information model to recommend content. The project also aims at using soft computing technologies to create an automated process and develop an intelligent web application. The System would benefit those users who have to scroll through pages of results to find relevant content.

8. Mohit Soni , Shivam Bansal: Movie Recommendation System

   A recommendation engine filters the data using different algorithms and recommends the most relevant items to users. It first captures the past behaviour of a customer and based on that, recommends products which the users might be likely to buy. If a completely new user visits an e-commerce site, that site will not have any past history of that user. So how does the site go about recommending products to the user in such a scenario? One possible solution could be to recommend the best selling products, i.e. the products which are high in demand. Another possible solution could be to recommend the products which would bring the maximum profit to the business. Three main approaches are used for our recommender systems. One is Demographic Filtering i.e They offer generalized recommendations to every user, based on movie popularity and/or genre.

The System recommends the same movies to users with similar demographic features. Since each user is different , this approach is considered to be too simple. The basic idea behind this system is that movies that are more popular and critically acclaimed will have a higher probability of being liked by the average audience. Second is content-based filtering, where we try to profile the users interests using information collected, and recommend items based on that profile. The other is collaborative filtering, where we try to group similar users together and use information about the group to make recommendations to the user.

9. Sidnooma Christian Kabore, CarlesFarreTost:Design and implementation of a recommender system as a module for Liferay portal

   In preparation for designing the recommender engine, we decided to analyze the different approaches in the literature with the intention of learning and understanding the advantages and shortcomings of each approach. After our analysis we decided to follow a hybrid approach that combines both collaborative filtering and content-based filtering techniques. Such an approach will enable to overcome the limits of content-based and collaborative filtering, leveraging the advantages of both techniques. Two-part hybrid recommenders are quite successful, but under certain domain and data characteristics different hybrids may achieve unlike results. For our intended recommender system we have decided to try out a way of introducing content-based descriptors into collaborative filtering by combining the content features (keywords and tags) and user ratings. To be more specific content features will be utilized to strengthen collaborative filtering.

10. SamiaHaimoura: Building a Book Recommendation System using Matrix Factorization and SV Decomposition

    We're now about to create a matrix model using the Matrix Factorization method with the Single Value Decomposition model (SVD). You can find a number of excellent technical resources online to describe the models in deeper ways, but I'm going to break it down to you in simple terms here.What we'll do next is call the pivot function to create a pivot table where users take on different rows, books different columns, and respective ratings values within that table with a shape (m*n).We'll need to create a set of training data — What our training data consists of is essentially smaller matrices that are factors of the ratings we want to predict. For this we'll set another matrix X which is the transposition of the resulting matrix, also called invertible matrix.I evaluated this model just by having a look at the list of books the algorithms spit out, and since I had already read (and really liked) some of the titles recommended, and run the model through other people for fun to see if they would agree for the most part- I thought I'd call it a day. Of course, this is not the right way to do it if you have to provide sharp performance metrics.

11. Mahak Dhanda and Vijay Verma Recommender System for Academic Literature with Incremental Dataset:

    The massive hike in the extent of information over the web has turned it arduous for the users to search for the information admissible to them. Recommender

System (RS) has come out as a revolutionary concept to ride out through this situation14. It is a tool (software) that provides the users with the suggestions of information that may be useful to them. These suggestions may turn out helpful to the users in many scenarios where decision making is involved ex. selecting books to read, movies to watch etc. A lot of techniques are available for recommendation which is majorly categorized as collaborative filtering7 and content-based filtering5. Collaborative filtering works on the concept of finding out similar users so as to make recommendations, while content-based techniques work on the basis of similarity in features of the item and the user.The proposed technique works in two stages 1) first stage is used to filter out the papers pertinent to the topic of interest of the user based on their content; 2) second stage makes sure that the personalized requirements of the user are satisfied (on the basis of usability assigned to the papers with the help of input taken from the user).

12. A collaborative approach for research paper recommender system:

    The objective of this doctoral thesis is to develop an effective user-modeling approach based on mind maps. To achieve this objective, we integrate a research-paper recommender system in our mind-mapping and reference-management software Docear. The recommender system builds user models based on the users' mind maps, and recommends research papers based on the user models. As part of our research, we identify several variables relating to mind-map- based user modeling, and evaluate the variables' impact on user-modeling effectiveness with an offline evaluation, a user study, and an online evaluation based on 430,893 recommendations displayed to 4,700 users. We show that user modeling based on mind maps performs about as well as user modeling based on other items, namely the research articles users downloaded or cited. Our findings let us to conclude that user modeling based on mind maps is a promising research field, and that developers of mind-mapping applications should integrate recommender systems into their applications. Such systems could create additional value for millions of mindmapping users.

13. Docear, Magdeburg, Germany: Recommendation Engine

    A "recommendation approach" is a model of how to bring a recommendation class into practice. For instance, the idea behind CF can be realized with user-based CF [222], content- boosted CF [223], and various other approaches [224]. These approaches are quite different but are each consistent with the central idea of CF. Nevertheless, these approaches to represent a concept are still vague and leave room for speculation on how recommendations are calculated. A "recommendation algorithm" precisely specifies a recommendation approach. For instance, an algorithm of a CBF approach would specify whether terms were extracted from the title of a document or from the body of the text, and how terms are processed (e.g. stop-word removal or stemming) and weighted (e.g. TF-IDF). Algorithms are not necessarily complete. For instance, pseudo-code might contain only the most important information and ignore basics,such as weighting schemes. This means that for a particular recommendation approach there might be several algorithms. Finally, the "implementation" is the actual source code of an algorithm that can be compiled and applied in a recommender system. It fully details how recommendations are generated and leaves no room for speculation. It is therefore the most specific idea about how recommendations might be generated.

14. Bela Gipp, Jöran Beel, Christian Hentschel: Recommendation System

    In this paper we present Scienstein, a hybrid recommender system, which uses both content- based and collaborative-based techniques. We believe that this approach has the potential to alleviate the problem of finding relevant research papers. Instead of solely relying on text mining, Scienstein combines citation analysis, implicit ratings, explicit ratings, author analysis and source analysis to a recommender system with a user-friendly GUI. Currently, Scienstein is in the development stage and open for cooperation. The first part of this paper gives an overview of related work including a discussion of the advantages and disadvantages of existing approaches. The main partintroduces Scienstein and discusses the technologies used. The focus lies on a hybrid recommender approach, which combines content-based and collaborative-based approaches. It shows that many of the disadvantages of existing systems become obsolete by combining known concepts with new ones.

15. Shraddha B. Shinde, Mrs. M. A. Potey Recommender System Evaluation Using Coverage:

    We studied a range of common metrics used for the evaluation of recommendation systems in software engineering. Based on a review of current literature, derived a set of dimensions that are used to evaluate individual recommendation systems or in comparing it against the current state of the art. We used coverage, F-Measure metrics to improvise the performance of recommender system for Research paper. Evaluation results shows the disparity between three common recommendation frameworks. Even though the frameworks implement similar algorithms, there exist large differences in the reported recommendation quality. It have been evaluated three types of recommendation algorithms (user-based and item-based CF and SVD- based matrix factorization) using popular and publicly available DBLP dataset. Finally, the content of this report can be used for understanding the evaluation criteria for recommendation systems and this can be improve the decisions when selecting a specific recommendation system for a software development project. In futurethe system performance can be enhanced by using new metrics rather than existing metrics for evaluating the recommender system

16. Beijing, Haidian District, Zhong Guan Cun South Street, No.5 Yard, Beijing Institute of Technology, China: Recommendation System

    The recommendation system is based on recommended techniques. The recommended technique, also known as personalized information filtering, is used to predict whether a given user will like a particular project (predictive problem) or to identify a set of N items of interest to a given user (top-N recommendation) problem).According to the model, it is divided into nearest neighbor model, hidden factor model and graph model. It can also be divided into ecommerce domain recommendation, social network domain recommendation, multimedia domain recommendation, mobile application domain recommendation, cross-domain recommendation, etc., depending on the application domain. The recommended system classification framework. The left branch of the frame diagram is the application area of the recommendation system, such as books, texts, pictures, movies, music, shopping, TV programs and others. The right branch is used by the recommendation system. Data mining techniques such as association rules, clustering, decision trees, K-nearest neighbors, link analysis, neural networks, regression, and heuristics.