# **Abstract**

*The objective of many businesses today is to give online consumers helpful product suggestions in order to enhance their website usage. People frequently choose or buy a new product based on suggestions from friends, comparisons of similar products, or user reviews. It is necessary to develop a recommender system in order to automate all of these operations. Even when the client is unaware of it, recommender systems use tools to make suggestions that best meet their needs. Based on previous behaviour, the offers of customised content entice users to return to the website. In this essay, a Netflix movie recommendation system will be constructed. The dataset used in this study includes more than 17K movies and 500K+ customers. A content-based recommender system for the Netflix movie dataset is presented in this research. The diverse and distinctive qualities of the movie are used to create the features employed in the system. The main focus of our research is that the movie can be more correctly represented by TF-IDF and Cosine similarity*

Table of Contents

[**Chapter 1 Introduction** 5](#_Toc110109299)

[**1.2 Objectives** 6](#_Toc110109300)

[**1.3 Hypothesis** 6](#_Toc110109301)

[**1.4** **Research Methodology** 6](#_Toc110109302)

[**1.5 Document Structure** 6](#_Toc110109303)

[**2 Literature Review** 7](#_Toc110109304)

[**2.1 Recommendation system types** 7](#_Toc110109305)

[**2.1.1 Collaboration filtering** 7](#_Toc110109306)

[2.1.2 Content Based Recommender System 7](#_Toc110109307)

[**2.1.3 Context Based Recommender System** 8](#_Toc110109308)

[**2.2 Collaborative filtering** 8](#_Toc110109309)

[2.2.1 Memory-Based Collaborative Filtering 9](#_Toc110109310)

[**2.2.2 Model-based CF** 9](#_Toc110109311)

[**2.2.3 Hybrid Collaborative Filtering Techniques** 9](#_Toc110109312)

[2.3 Content-Based Recommender System 10](#_Toc110109313)

[2.4 Context-Based Recommender System 11](#_Toc110109314)

[**Chapter 3 Methodology** 14](#_Toc110109315)

[**3.1 Introduction** 14](#_Toc110109316)

[**3.2 Dataset Description** 14](#_Toc110109317)

[**3.3 Tools and technologies used** 14](#_Toc110109318)

[**3.4 Data Cleaning** 14](#_Toc110109319)

[**3.5 Exploratory Data Analysis (EDA)** 14](#_Toc110109320)

[**3.6 Training, Testing, and Validating results** 18](#_Toc110109321)

[**3.7 Summary of Chapter** 18](#_Toc110109322)

[**Chapter 4 Experiment and Result** 19](#_Toc110109323)

[**4.1 Experiment** 19](#_Toc110109324)

[**4.1.1 Feature Representation** 20](#_Toc110109325)

[**4.1.2 S**imilarity 22](#_Toc110109326)

[**4.2 Result** 24](#_Toc110109327)

[**Chapter 5 Conclusion and Future Work** 26](#_Toc110109328)

[**5.1 Conclusion** 26](#_Toc110109329)

[**5.2 Future Work** 26](#_Toc110109330)

[**References** 28](#_Toc110109331)

**Table of Figures**

[Fig 2.1 Pearson correlation similarity 8](#_Toc110109823)

[Fig 2.2. Collaborative Filtering Recommender System 10](#_Toc110109824)

[Fig 2.3 Content-Based Recommender System 11](#_Toc110109825)

[Fig 2.4. Contextual Information Dimensions 13](#_Toc110109826)

[Fig 3.1 number of movies and tv shows 15](#_Toc110109827)

[Fig 3.2 TV rating 15](#_Toc110109828)

[Fig 3.3 movie release year 16](#_Toc110109829)

[Fig 3.4 Directors with the most number of Movies/TV Shows. 16](#_Toc110109830)

[Fig 3.5 Countries with the most movies/series. 17](#_Toc110109831)

[Fig 3.6 Distribution of number of seasons for TV Shows. 17](#_Toc110109832)

[Fig 4.1 first 5 rows of the dataset 19](#_Toc110109833)

[Fig 4.2 attribute data types 20](#_Toc110109834)

[4.3 TF-IDF Matrix. 20](#_Toc110109835)

[Fig 4.4 Vector Space Model of documents 21](#_Toc110109836)

[Fig 4.5 cosine similarity 23](#_Toc110109837)

[Fig 4.6 cosine similarity function 23](#_Toc110109838)

[Fig 4.7 get recommendation function. 24](#_Toc110109839)

[Fig 4.8 recommendation for Blood & Water movie 25](#_Toc110109840)

**Table of Formulas**

[Formula 4.1 term frequency 21](#_Toc110110066)

[*Formula 4.2 IDF* 22](#_Toc110110067)

[*Formula 4.3 Normalization function* 22](#_Toc110110068)

# **Chapter 1 Introduction**

Netflix is a firm that manages a large library of television shows and films that may be seen online at any time (computers or TV). Users pay a monthly fee to have access to the site, which makes this company profitable. Clients, on the other hand, have the option to discontinue their memberships at any moment (Amatriain, 2013). As a result, it's critical for the company to retain people interested in the platform. This is where recommendation systems come into play; providing useful ideas to users is critical (Ricci et al., 2010). The popularity of recommendation systems is growing among service providers because they help to increase the number of items sold, provide a diverse selection of items, increase user satisfaction and loyalty to the company, and they are very useful in gaining a better understanding of what the user wants (Ricci et al., 2010). The user will be able to make better judgments from a wider range of cinematographic products as a result.

The recommender systems consider not only information about the consumers, but also information about the items they consume, as well as comparisons with other products etc. (Hahsler, 2014). Nonetheless, there are a plethora of algorithms that can be used to create a recommendation system. For example, I Popularity, which only recommends the most popular goods. (ii) Collaborative Filtering, which looks for patterns in user behavior to generate user-specific recommendations (Breese, Heckerman, and Kadie, 1998); (iii) Content-based Filtering, which recommends items based on similar information the user has liked or used in the past (description, topic, etc) (Aggarwal, 2016); (iv) Hybrid Approaches, which combines the two algorithms mentioned above (Adomavicius and Tuzhilin, 2005).

Selecting the method that best suits the analysis is not straightforward, and neither does improving the obvious expand the user's preference into nearby regions. As a result, the primary types of recommender algorithms will be discussed in this paper, along with the benefits and drawbacks of each algorithm to provide a better knowledge of how they work. As a result, many algorithms will be tried in the end in order to determine which one works best for Netflix consumers.

This research is based on real Netflix user data and the ratings they've given to the films they've seen. The database comprises 17,770 files, one for each movie, with consumer ratings on a five-star system ranging from 1 to 5. In addition, the movie file contains the year of release as well as the title of the film.

## **1.2 Objectives**

The following are the project's key goals and objectives:

* Analyse the current state of the art in terms of how other researchers are approaching the problem of utilizing machine learning to recommend movies to viewers.
* Create a machine learning model that uses similarity to recommend movies to users.
* Examine the model that has been proposed.

## **1.3 Hypothesis**

How effective is a machine learning model in suggesting movies to users?

## **Research Methodology**

In order to recommend movies to users, the collaborative filtering approach was applied in this project's study methodology. We'll be working with the Netflix movie recommender dataset, which can be found here. Anaconda would be installed and used to create, train, and test our model on our local system. Then we'd evaluate our model and present the results.

## **1.5 Document Structure**

The following sections of the paper are described in detail: We present a complete review of the literature on recommendation systems in Chapter 2, including types, algorithms, and related studies. In Chapter 3, we outline our research strategy, methodology, and data collection methods. Chapter 4 details our implementation and discussion. In Chapter 5, we present our findings, recommendations, and future work.

# **2 Literature Review**

Collaborative filtering, content-based filtering, and hybrid filtering are the three methods used by recommendation systems. A content-based recommender system uses either explicit (ratings) or implicit input from the user (by clicking on a link). This information is gathered by the system, which creates user profiles for each user. Recommendations are created using user profiles. When using content-based filtering, only one user's profile is looked at to make recommendations. Based on users' prior behaviour, the system tries to propose items that are comparable to that item. Collaborative filtering, as opposed to content-based filtering, identifies users whose preferences match those of a certain user. It then makes recommendations for items or other products by figuring out whether the given user will enjoy the thing that other users enjoy since their tastes are similar. Due to the fact that each of these approaches has strengths and weaknesses of its own, a hybrid approach was developed to address these issues. There are numerous types of hybrid filtering. To generate the result, we can either combine both filters into one model or start with content-based filtering and then feed the results to a collaborative recommender (or vice versa).

## **2.1 Recommendation system types**

### **2.1.1 Collaboration filtering**

Collaborative filtering (CF) is mostly based on the previous data of the user that is available to the system (Su and Khoshgoftaar, 2009), (Zhang, 2015) and employs the numerical reviews provided by the user. The user profile is aided by the historical data that is readily available, and the item profile is created using the data that is readily available about the item. A recommendation system is created using both the user profile and the item profile. Collaborative filtering is very popular now thanks to the Netflix Competition (Bennett & Lanning 2007), (Zhang, 2015). The most fundamental and straightforward technique for obtaining recommendations and making forecasts about the sales of a product is collaborative filtering. Although it does have some drawbacks, these drawbacks have spurred the creation of new strategies and tactics.

### **2.1.2 Content Based Recommender System**

Content-based (Lops, de Gemmis and Semeraro, 2010) systems emphasize product features and seek to build user profiles based on past reviews as well as profiles of the item based on its features and the reviews it has received (Titov and McDonald, 2008) (Zhang, 2015). It has been noted that reviews typically pair user opinions and product features (Zhang, 2015), (Lops, de Gemmis and Semeraro, 2010), (Ding, Liu and Yu, 2008). It has been noticed that user evaluations typically start with a specific feature and end with the user's assessment of the product. The sparsity issue that collaborative filtering-based recommendation systems encounter is helped by content-based recommendation systems.

### **2.1.3 Context Based Recommender System**

Context-based recommender systems accomplish this by extending the user/item convention to the user's circumstances in order to take into account contextual information (Panniello et al., 2009). This makes it easier to do away with the tedious procedure of requesting a ton of personal information from the user.

## **2.2 Collaborative filtering**

By using the previously known preferences of numerous users, collaboration filtering is a method for forecasting the preferences of individuals (Resnick and Varian, 1997) and the other is the object. The key issues that Collaborative Filtering addresses are data sparsity, scalability, and cold start problem. It uses cosine and Pearson correlation. Three key methods are introduced by CF: memory, which is utilized to combine CF with other recommendation techniques, and their processing power (Zhang, 2015).

**Cosine similarity:** Given two attribute vectors, A and B, the magnitude, cosine similarity, and cos (product are all as ) (Marx, 2013).

Text

Description automatically generated

Fig 2.1 Pearson correlation similarity

Text

Description automatically generated

N is the total number of attributes, and (x, y) denotes the data objects ) (Marx, 2013).

### 2.2.1 Memory-Based Collaborative Filtering

Each user is a member of a group made up of individuals with shared interests (Sarwar, Karypis, Konstan and Reidl, 2001). Memory-based CF is represented by User-based CF and Item-based CF. It scales well with associated items and is simple to use. The content of the things being recommended doesn't need to be taken into account. Memory-based CF has a number of drawbacks, including the cold start problem, sparsity, and their reliance on human ratings (Su and Khoshgoftaar, 2009).

### **2.2.2 Model-based CF**

Complex patters which are based on training data, are recognized by designing and developing the models (such as data mining algorithms, machine learning) and then intelligent predictions are made for CF tasks for the real-world data which are based on learnt models (Breese, Heckerman, and Kadie, 1998) (Hirsh, and Cohen, 1998). It gives an intuitive rationale for recommendations. Model-building is an expensive procedure. Other disadvantage of model-based CF is that it loses useful information for dimensionality reduction techniques (Su and Khoshgoftaar, 2009).

### **2.2.3 Hybrid Collaborative Filtering Techniques**

To achieve better outcomes, hybrid recommender systems integrate several collaborative approaches and other recommender techniques (often content-based approaches). Using a hybrid method can help you avoid a number of issues, including cold-start, data sparsity, and scalability (Adomavicius and Tuzhilin, 2005). The following are various ways to combine CF with other recommender techniques:

Hybrid Recommenders Combining CF and Other Recommender Systems.

Hybrid Recommenders Including CF and Content-Based Features

Hybrid Recommenders that Combine CF Algorithms (Su and Khoshgoftaar, 2009).

Fig. 1 shows the Collaborative Filtering Recommender System flowchart. It demonstrates how collaborative filtering simply takes into account numerical ratings provided by various users before returning a list of suggested products. The user reviews are kept in a database for future use and forecasting. User 1 and User 6 in the illustration exhibit comparable behavior; thus, their profiles are located in the same area, which denotes shared interests. The reviews of User 1 can be used to anticipate a review for a product that User 6 has not yet given a rating for. Thus, considering the information at hand, a prediction is produced regarding User 6's review of product C Recommendations are derived from these forecasts and offered to the user.

Diagram

Description automatically generated

Fig 2.2. Collaborative Filtering Recommender System

## 2.3 Content-Based Recommender System

It is a method where unique user profiles are taken into consideration. It increases user interest and makes predictions about whether they would like to eat at a specific restaurant or see a specific movie (Hirsh, and Cohen, 1998). It is also known as adaptive filtering since it adapts to users' preferences and offers suggestions based on their areas of interest. It shows the contrast between the content of the item and the content of items the user is interested in. Better user profiles are created by collecting feedback from current users using the Bayesian hierarchical model (Zhang and Koren, 2007). To contrast pure CF and pure Content-base, content based collaborative filtering is more frequently used. In CF, the sparsity issue is solved by applying content-based prediction to transform a sparse user filled matrix into a full user rated matrix (Melville, Mooney and Nagarajan, 2002). A content-based recommendation system's information flow is shown in Fig. 2. As input, pertinent entities and relations of an item are maintained together. The main characteristics of items are taken from the item ontology. The content-based recommender system takes into account item characteristics, user reviews, and data from user modeling. Several recommended items are provided as output after applying.

Diagram

Description automatically generated

Fig 2.3 Content-Based Recommender System

## 2.4 Context-Based Recommender System

Although collaborative filtering techniques or recommender systems based on the contents and attributes of things have seen significant success, they still have room for improvement. To provide more relevant suggestions, recommender systems need a higher level of personalization. Users' contextual data is also taken into account while creating a recommender system in order to do this. The terms "context" and "user's status" refer to the time, place, region, and environment of the user. A recommender system that incorporates contextual data can better understand the circumstances around any person, location, or object that is relevant to its prediction capabilities (Lee and Lee, n.d.), (Levi et al, 2012).

It assists in gathering data on a specific group of people, and this data is essential for improving the user suggestions and improving the effectiveness of the system. Recommender systems need the user's current situational information, which context-based recommender systems directly obtain through a variety of methods (such GPS) (Yang and Wang, 2009) without bothering the user. The flow of contextual data in a context-based recommender system is shown in Fig. 3. The system receives as input from the user's location data, social data, time of day, and weather data, which are all considered to be contextual data. The user's position is recorded and an estimated address for them is determined. A user's social account can be asked for permission to access their social data.

The person's device can be used to determine the time of day, which is important for determining the most appropriate recommendations for the user. Finding the location's weather can also be done with the aid of access. A system processes all the data gathered, and sentiment analysis is performed. Following processing, the system outputs a user-friendly list of attractions. The system produces the desired effects based on how it is applied. If the program is intended for a tourist, suggested locations, the shortest routes, or lodgings may be provided.

Depending on whether or not they vary over time, contextual elements can be either dynamic or static.

1. **Dynamic:** when the circumstances are unstable because they change over time. By receiving direct customer feedback, they might alter. User feedback is typically used to improve the user profile so that recommendations are more accurate. The major problem is that a system should be able to determine when to transition to a new underlying context model if it is thought of as dynamic.
2. **Static:** The contextual elements are steady because they don't alter over time. For instance, while deciding whether to buy a cell phone, the context may include the time of day, the reason for the purchase, and these two aspects alone.

Depending on what is being viewed, contextual elements can be fully observable, partly observable, or unobservable.

1. **Fully observable**: When recommendations are provided, the whole structure and values of contextual elements are explicitly understood.
2. **Partially observable**: A certain amount of knowledge about the contextual circumstances is explicit.
3. **Unobservable**: There is no specific information on contextual elements in it. (Savage, Baranski, Chavez and Höllerer, 2011).

Table

Description automatically generated

Fig 2.4. Contextual Information Dimensions

# **Chapter 3 Methodology**

## **3.1 Introduction**

In this chapter, we outline our methodology, the dataset's description, data exploration, the division between training and validation, and the project's implementation.

## **3.2 Dataset Description**

One of the most well-known media and video streaming services is Netflix. They offer more than 8000 movies and TV episodes on their platform, and as of the middle of 2021, they had more than 200 million subscribers worldwide. This tabular dataset includes listings for all of the Netflix movies and TV episodes, together with information about the actors, directors, ratings, release year, duration, and other factors. The dataset is available freely [here](https://www.kaggle.com/datasets/shivamb/netflix-shows).

## **3.3 Tools and technologies used**

The following resources and technologies were employed during the research for this dissertation:

**Anaconda:** With Anaconda, machine learning models may be trained, created, tested, and assessed. In this setting, machine learning techniques can be used at will. The construction of our model, as well as its subsequent training, testing, and evaluation, benefited from the Anaconda architecture. Because installing several machine learning Python packages using Anaconda is so simple, we chose it over Google Colab. Anaconda offers a GUI for running machine learning models (Graphical User Interface).

**Tableau:** Using this tool, Tableau data visualization is simple. Data that is visualized makes it simpler to understand and generate fresh ideas. For our research, exploratory data analysis was done using Tableau. Because to its simplicity and ability to quickly adapt and combine data, Tableau was chosen over other data visualization tools including PowerBI and data visualization libraries like Matplotlib, Seaborn, or R GGplot. Our model's time series are forecast using Tableau as well.

## **3.4 Data Cleaning**

In the process of cleaning the data, we looked for any missing values and replaced the numeric data types with their mean and the categorical data types with their mode. Additionally, we looked for outliers but couldn't find any.

## **3.5 Exploratory Data Analysis (EDA)**

Chart, bar chart

Description automatically generated

Fig 3.1 number of movies and tv shows

In the Netflix dataset, there are more movies than TV shows, with about 6000 movies and over 2000 tv shows.

Chart, bar chart

Description automatically generated

Fig 3.2 TV rating

The ratings chart clearly shows that TV-MA (Mature Audiences) programming is well-liked by its viewers. Then it goes down to TV-PG ( Parental Guidance) and TV-14 (for content that some parents and guardians may deem inappropriate for children under 14).

Chart, bar chart

Description automatically generated

Fig 3.3 movie release year

According to this chart, the years 2018 and 2017 saw the greatest output of available content. Not many contents will be created in 2019, 2020, or 2021, which is an interesting fact. The Covid-19 pandemic is largely to blame for this.

Chart, bar chart

Description automatically generated

Fig 3.4 Directors with the most number of Movies/TV Shows.

Fig 3.4 shows the directors with the most number of movies/tv shows. Rajiv Chilaka has the most number of movies/tv shows directed by him.

Chart

Description automatically generated

Fig 3.5 Countries with the most movies/series.

Fig 3.5 shows the countries with the most movies/series. The United States has the most movies and series, then India and the united kingdom comes second and third respectively.

Chart

Description automatically generated with medium confidence

Fig 3.6 Distribution of number of seasons for TV Shows.

Fig 3.6 shows the distribution of number of seasons for TV Shows. Most Netflix tv shows only lasted for 1 season. There are only a few shows who got to multiple seasons, this is true as Netflix are famous for cancelling tv shows after just one season.

## **3.6 Training, Testing, and Validating results**

First, we split our dataset in half, using 80% of it for training and the remaining 20% for evaluation. The sample size split was the same for all of our models. We applied the dataset for the validation set using 20% of the training data.

## **3.7 Summary of Chapter**

We have described our project's approach and methodology in this chapter, and we will go into more detail about the project's implementation in the part that follows.

# **Chapter 4 Experiment and Result**

## **4.1 Experiment**

In this chapter, we present how we conducted our experiment, the result and discussion/analysis. At first, we import our libraries and read the dataset with panda, to convert it to a data frame and then display the first 5 rows.

Text

Description automatically generated

Fig 4.1 first 5 rows of the dataset

Next, we display the data type of the attributes of our dataset. Most of the attributes were strings (objects), with the release year the only numeric attribute. Fig 4.2 captures the data types of our attributes

Graphical user interface, text, application, table

Description automatically generated

Fig 4.2 attribute data types

Next, we performed some exploratory data analysis (EDA), the results of our exploratory data analysis are presented in chapter 3 of this document.

### **4.1.1 Feature Representation**

Next, we define our TF-IDF matrix for our content based recommender system. The TF-IDF(Term Frequency-Inverse Document Frequency (TF-IDF) ) score is the frequency of a word occurring in a document, down-weighted by the number of documents in which it occurs. This is done to reduce the importance of words that occur frequently in plot overviews and therefore, their significance in computing the final similarity score.

Graphical user interface, text, application

Description automatically generated

4.3 TF-IDF Matrix.

#### **4.1.1.1Vector Space Model**

It is crucial to understand document representation before delving deeply into feature representation. Assume there exist many documents, each of which is composed of just one sentence. The document can then be represented as a model, known as a vector space model, in Figure 4.4. If you think of each movie feature as a phrase, this model can be used to represent a feature. Obviously, certain features are more significant than others, and the significance determines how much weight each feature receives in the similarity calculation.

Diagram

Description automatically generated

Fig 4.4 Vector Space Model of documents

#### **4.1.1.2 TF-IDF**

TF-IDF, which stands for term frequency-inverse document frequency, is a popular weighting method for text mining and information retrieval that indicates the significance of a word for a document. A word's significance increases in direct proportion to the number of times it appears in the text, but it also diminishes in direct proportion to the number of times it appears across the entire corpus.

Term Frequency and Inverse Document Frequency, or TF and IDF, respectively, make up TF-IDF. TF stands for the frequency with which a word appears in the text. IDF's key tenet is that a term will have less significance if it appears more frequently in other papers.

#### **4.1.1.3 Term Frequency**

Term frequency, which can be denoted by TF, refers to how frequently a term ti appears in a document dj (tij). When stop-words are removed, the more frequently a term (ti) appears in a document, the more significant that term is to the document. It is characterized as:

*TF*(*tij*) = *N*(*ti, dj*)

*N*(*dj*)

Formula 4.1 term frequency

*N (ti, dj) represents the number of times that ti appears in dj, and N (dj) is the total number of terms in the document dj.*

#### **4.1.1.4 Inverse Document Frequency**

Let's first define document frequency in order to comprehend inverse document frequency. The number of times the term ti appears across all documents C, which is denoted by N, is known as the document frequency (ti, C). The weaker term ti may represent document dj, the more frequently it appears in all papers C.

Inverse document frequency, denoted as IDF (ti), signifies that the relationship between the term ti's capacity to represent the document dj and the sum of all documents N (ti, C) is inverse.

*IDF* (*t* ) = log *N* (*C*)

*i N* (*ti, C*)

Formula 4.2 IDF

The total number of documents is N (C), and IDF (ti) decreases as N (C) increases (ti, C). The more the representation ti has for dj, the less N (ti, C) there is.

#### **4.1.1.5 Normalization**

We shall normalize each variable to lessen the inhibition caused by stop words. The calculation of TF IDF following normalization is:

*T F* (*tij* ) × *I DF* (*ti* )

*weightT F* −*IDF* (*tij* ) = s *n*

P [*TF* (*tij* ) × *IDF* (*ti*)]2

*j*=1

Formula 4.3 Normalization function

Equation 4.4 is based on the idea that a term is more indicative of a document when it appears in it more frequently than it does in other texts.

### **4.1.2 S**imilarity

Next, we calculate the similarities between movies. We use the cosine similarity function to calculate the similarities between movies.

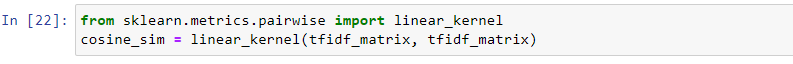


Fig 4.5 cosine similarity

Since it is independent of magnitude and can be calculated rather quickly, the Cosine similarity score is used. In text analysis, the cosine similarity measure of similarity is frequently used to assess document similarity.

Content based systems aim to compare the similarities of films. Any type of content, including text, video, and images, can be used. Each movie in our project is represented by a feature vector. Then, I'll describe the cosine similarity algorithm.

The most often used method of determining document similarity is cosine similarity. Equation 4.5 can be used to determine the cosine of the angle between the feature vectors in order to determine how similar two features are to one another.

Graphical user interface

Description automatically generated with low confidence

Fig 4.6 cosine similarity function

The similarity's range is between -1 and 1. -1 indicates that the two vectors are moving in complete opposite directions, and 1 indicates that they are moving in the same direction. If there is no relationship between the two vectors, the cosine similarity is 0. Since the weights are non-negative for text matching, the range in our situation should be 0 to 1. Here is an example that shows how cosine similarity works.

Given are two phrases:

• A: I enjoy watching television, but not film.

B: I detest watching TV and listening to music.

How can we determine how similar the two sentences are? The fundamental tenet is that sentences are more similar the more closely related the terms in the two phrases are.

Segmenting words is the first stage.

I enjoy watching TV, but I don't enjoy watching the news.

• B: I dislike watching TV and using the internet. List each word as the second step.

• I enjoy watching TV but I don't drink and drive. Calculating word frequencies is the third stage.

• A: I enjoy watching television, but I don't watch it much.

• B: I like watching TV, but I don't want to drink, so 1. Get a word frequency vector in the fourth stage.

• A: [2, 2, 2, 1, 1, 1, 1, 0]

• B: [1, 1, 1, 1, 0, 1, 1, 1]

Then we can calculate the cosine of the two vectors by Equation 4.5.

2 × 1 + 2 × 1 + 2 × 1 + 1 × 1 + 1 × 0 + 1 × 1 + 1 × 1 + 0 × 1

cos(*θ*) = √

22 + 22 + 22 + 12 + 12 + 12 + 12 + 02 × √12 + 12 + 12 + 12 + 02 + 12 + 12 + 12

Since the value is 0.85, the two statements are very similar to one another.

Finally, we define a get recommendation function that takes a movie title and calculate the consume similarity of movies similar to the title and return them to the user.

Text

Description automatically generated

Fig 4.7 get recommendation function.

## **4.2 Result**

In this instance, term to document is feature to movie. The vector space model, which may be used to determine similarity, can be simply converted from movie format. Each movie in the database can be represented by a vector thanks to earlier calculations. Then, to determine how similar each movie is to the others, we utilize the cosine similarity discussed in Section 4.1.2. The fi suggestion for the film Blood and Water is displayed in Figure 4.8.

Text

Description automatically generated

Fig 4.8 recommendation for Blood & Water movie

# **Chapter 5 Conclusion and Future Work**

## **5.1 Conclusion**

The importance of recommender systems is growing as a result of the information overload. We specifically try to create a new technique to increase the representative of the movie's accuracy for content-based recommender systems.

A content-based recommender system for the Netflix movie dataset is presented in this research. The diverse and distinctive qualities of the movie are used to create the features employed in the system. The main focus of our research is that the movie can be more correctly represented by TF-IDF and Cosine similarity, so we develop a novel method for allocating weight to these features.

## **5.2 Future Work**

There has never been a low moment in the history of the recommender system. Large-scale networks and high-performance computing have propelled new research in this area thanks to the rise of machine learning. We'll take into account the following considerations in our next work.

Utilize collaborative filtering recommendations: Once we have a sufficient amount of user data, we will add collaborative filtering recommendations. Collaborative filtering is based on user social information, which will be examined in subsequent study, as we stated in Chapter 2.

Introduce more accurate and appropriate movie attributes. Typical group recommendation uses the rating rather than object features. In the future, we should extract attributes from movies like color and subtitles that can give a more accurate description of the film.

introduce the user dislike movie list: In recommender systems, user data is usually helpful. We will continue to gather user information and add a list of movies that users dislike. In order to generate scores that will be added to the previous result, we will also input a list of movies that we despise into the recommender system. By doing this, we can enhance the performance of the recommender system.

Make use of machine learning: In a future study, we'll add dynamic parameters to the recommender system. We'll utilize machine learning to automatically modify each feature's weight and filter the best weights.

# **References**

* Asher Levi, Osnat (Ossi) Mokryn, Christophe Diot and Nina Taft, 2012 “Finding a Needle in a Haystack of Reviews: Cold Start Context-Based Hotel Recommender System”,6th ACM conference on Recommender Systems.
* Bennett, J., & Lanning, S. 2007. The Netflix Prize.
* Chumki Basu, Haym Hirsh, and William Cohen, 1998. Recommendation as classification: using social and content-based information in recommendation. In Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence (AAAI '98/IAAI '98). American Association for Artificial Intelligence, USA, 714–720.
* Ding, X., Liu, B. and Yu, P., 2008. A holistic lexicon-based approach to opinion mining. Proceedings of the international conference on Web search and web data mining - WSDM '08,.
* Fan Yang and Zhi-Mei Wang, 2009. A mobile location-based information recommendation system based on GPS and WEB2.0 services. W. Trans. on Comp. 8, 4 (April 2009), 725–734.
* G. Adomavicius and A. Tuzhilin, 2005. “Toward the next generation of recommender systems: A survey of the state-of-the-artand possible extensions,” IEEE Trans. on Knowledge and Data Eng., vol. 17, pp. 734–749.
* Ivan Titov and Ryan McDonald, 2008. A Joint Model of Text and Aspect Ratings for Sentiment Summarization. In Proceedings of ACL-08: HLT, pages 308–316, Columbus, Ohio. Association for Computational Linguistics.
* John S. Breese, David Heckerman, and Carl Kadie, 1998. Empirical analysis of predictive algorithms for collaborative filtering. In Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence (UAI'98). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 43–52.
* Lee, J. and Lee, J., n.d. Context Awareness by Case-Based Reasoning in a Music Recommendation System. Ubiquitous Computing Systems, pp.45-58.
* Lops, P., de Gemmis, M. and Semeraro, G., 2010. Content-based Recommender Systems: State of the Art and Trends. Recommender Systems Handbook, pp.73-105.
* Norma Saiph Savage, MaciejBaranski, Norma Elva Chavez and Tobias Höllerer, 2011. “Advances in Location-Based Services I’m feeling LoCo: A Location Based Context Aware Recommendation System”, Springer.
* Panniello, U., Tuzhilin, A., Gorgoglione, M., Palmisano, C. and Pedone, A., 2009. Experimental comparison of pre- vs. post-filtering approaches in context-aware recommender systems. Proceedings of the third ACM conference on Recommender systems - RecSys '09,.
* Paul Marx, 2013 - Providing Actionable Recommendations: A Movie Recommendation Algorithm with Explanatory Capability, Joseph EulVerlag.
* Prem Melville, Raymond J. Mooney and RamadassNagarajan, 2002. “Content-Boosted Collaborative Filtering for Improved Recommendations”,pp. 187 – 192.
* Resnick, P. and Varian, H., 1997. Recommender systems. Communications of the ACM, 40(3), pp.56-58.
* Sarwar, B., Karypis, G., Konstan, J. and Reidl, J., 2001. Item-based collaborative filtering recommendation algorithms. Proceedings of the tenth international conference on World Wide Web - WWW '01,.
* Su, X. and Khoshgoftaar, T., 2009. A Survey of Collaborative Filtering Techniques. Advances in Artificial Intelligence, 2009, pp.1-19.
* Zhang, Y. and Koren, J., 2007. Efficient bayesian hierarchical user modeling for recommendation system. Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '07,.
* Zhang, Y., 2015, February. Incorporating phrase-level sentiment analysis on textual reviews for personalized recommendation. In Proceedings of the eighth ACM international conference on web search and data mining (pp. 435-440).