PROJECT REPORT

MOVIE RECOMMENDATION ENGINE

NAME-ADITYA KUMAR NAME-AARTI

ROLL-2001920130011 ROLL-200192010002

GL BAJAJ INSTITUTE OF TECHNOLOGY AND MANAGEMENT

Contents

Chapter 1

Introduction

As the development of information technology and Internet, people enter the era of information overload from that of information deficiency gradually. The report is the result of the Master Thesis in Information and Communication Technology.

1.1 Background

In the era of information overload, it is very difficult for users to get information that they are really interested in. And for the content provider, it is also very hard for them to make their content stand out from the crowd. That is why many researchers and companies develop Recommender System to solve the contradiction. The mission of Recommender System is to connect users and information, which in one way helps users to find information valuable to them and in another way push the information to specific users. This is the win-win situation for both customers and content providers.

1.2 Problem Statement

For building a recommender system from scratch, we face several different problems. Currently there are a lot of recommender systems based on the user information, so what should we do if the website has not gotten enough users. After that, we will solve the representation of a movie, which is how a system can understand a movie. That is the precondition for comparing similarity between two movies. Movie features such as genre, actor and director is a way that can categorize movies. But for each feature of the movie, there should be different weight for them and each of them plays a different role for recommendation. So we get these questions:

- How to recommend movies when there are no user information.
- What kind of movie features can be used for the recommender system.
- How to calculate the similarity between two movies.
- Is it possible to set weight for each feature.

1.3 Goals

The goals of this thesis project is to do the research of Recommender Systems and find a suitable way to implement it. There are many kinds of Recommender Systems but not all of them are suitable for one specific problem and situation. Our goal is to find a new way to improve the classification of movies, which is the requirement of improving content-based recommender systems.

1.4 Methodology

In order to achieve the goal of the project, the first process is to do enough background study, so the literature study will be conducted. The whole project is based on a big amount of movie data so that we choose quantitative research method. For philosophical assumption, positivism is selected because the project is experimental and testing character. The research approach is deductive approach as the improvement of our research will be tested by deducing and testing a theory. Ex post facto research is our research strategy, the movie data is already collected and we don't change the independent variables. We use experiments to collect movie data. Computational mathematics is used data analysis because the result is based on improvement of algorithm. For the quality assurance, we have a detail explanation of algorithm to ensure test validity. The similar results will be generated when we run the same data multiple times, which is for reliability. We ensure the same data leading to same result by different researchers.[12]

1.5 Ethics

Movie information is the only part that may have ethics problem. However, all the information we get for research is from public database such as Wikipedia and our own movie database. So there are no data confidentiality and user privacy problems.

1.6 Outline

In the first chapter, background of the project is introduced. After that, the layout of the report is as follows: Chapter 2 provides an overview of related work on recommender systems. I will introduce the mainstream approaches and some famous business recommender systems. Chapter 3 shows the basic principles of our recommender system and how we improve it. Chapter 4 I will illustrate how recommendation works by a specific movie. Chapter 5 is the validation part, I will present our test result to illustrate the improvement of our approach. In the end, Chapter 6 and 7 is the conclusion and possible future work respectively.

Chapter 2

Overview of Related Work

Recommender system is a very hot research topic in recent years. Many researchers raised a lot of different recommendation approaches. The most famous category of these approaches is:

- Content-based Recommendation.
- Collaborative-filtering Recommendation.
- Hybrid Recommendation.

2.1 Content-based Recommendation

Content-based recommendation is an important approach in recommender systems. The basic idea is to recommend items that are similar with what user liked before[21]. The core mission of content-based recommender system is to calculate the similarity between items. There are a lot of methods to model item and the most famous one is Vector Space Model[2]. The model extracts keywords of the item and calculate the weight by TF-IDF. For example, set k_i as the ith keyword of item d_j, w_{ij} is the weight of k_i for d_j, then the content of d_j can be defined as:

$$Content(d_j) = \{w_{1j}, w_{2j}, ...\} \tag{2.1}$$

As we talked before, content-based recommender system recommends items that are similar with what user liked before. So the tastes of a user can be modeled according to the history of what the user liked. Consider $ContentBasedProfile(u)$ as the preference vector of user u, the definition is:

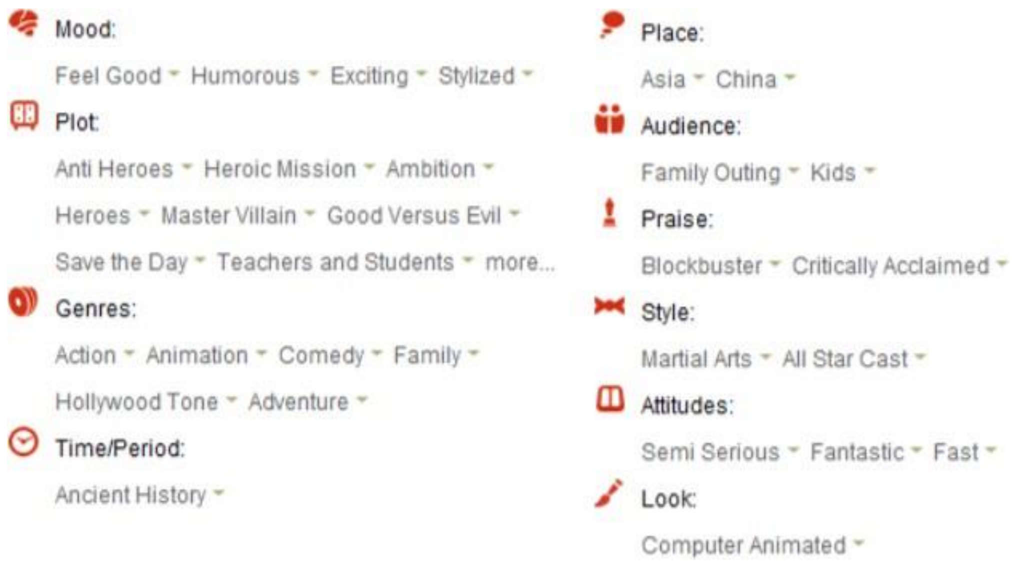
$$ContentBasedProfile(u) = \frac{1}{|N(u)|} sum_{d \in N(u)} Content(d) \tag{2.2}$$

$N(u)$ is what the user u liked before. After calculating content vector $Content(.)$ and content preference vector $ContentBasedProfile(.)$ of all users, given any user

u and an item d, how the user like the item is defined as the similarity between $ContentBasedProfile(u)$ and $Content(d)$:

$$p(u, d) = sim(ContentBasedProfile(u), Content(d)) \tag{2.3}$$

Using keywords to model item is an important step for many recommender systems. But extracting keywords of an item is also a difficult problem, especially in media field, because it is very hard to extract text keywords from a video. For solving this kind of problem, there are two main ways. One is letting experts tag the items and another one is letting users tag them. The representative of expert-tagged systems are Pandora for music and Jinni for movies. Let's take Jinni as an example, the researchers of Jinni defined more than 900 tags as movie gene, and they let movie experts to make tags for them. These tags belong to different categories, including movie genre, plot, time, location and cast. Figure 2.1 is from *Jinni*, which are the tags for movie Kung Fu Panda. As we can see from the figure, the tags of Kung Fu Panda are divided into ten categories totally, *Mood*, *Plot*, *Genres*, *Time*, *Place*, *Audience*, *Praise*, *Style*, *Attitudes* and *Look*. These tags contain all aspects of movie information, which can describe a movie very accurately.



Mood:
Feel Good · Humorous · Exciting · Stylized
Plot:
Anti Heroes · Heroic Mission · Ambition
Heroes · Master Villain · Good Versus Evil
Save the Day · Teachers and Students · more...
Genres:
Action · Animation · Comedy · Family
Hollywood Tone · Adventure
Time/Period:
Ancient History
Place:
Asia · China
Audience:
Family Outing · Kids
Praise:
Blockbuster · Critically Acclaimed
Style:
Martial Arts · All Star Cast
Attitudes:
Semi Serious · Fantastic · Fast
Look:
Computer Animated

Figure 2.1. Tags for Kung Fu Panda

Compared with expert-tagged system, user-tagged system is applied more widely. The representative websites are *Delicious* and *Flickr*. The feature of user-tagged system is the tags are more diversity than that of expert-tagged system. But the weakness is that the tags are of lower quality, even there are a lot of wrong tags. So in the user-tagged systems, there are two main problems, one is tag recommendation[34], which means when a user tags an item, the system can recommend some relative tags for him to choose. The purpose is first to be convenience for users and second it can increase the quality of tags. Another question is how to recommend items based on tags(tag-based recommendation[33]). After items are tagged, the simplest recommendation approach is to use tags as keywords of the item, and recommend by the content-based algorithm.

2.2 Collaborative-filtering Recommendation

Collaborative-filtering recommendation is the most famous algorithm in recommender systems. This algorithm models user's taste according to the history of user behavior. GroupLens published the first paper[31] about collaborative filtering and the paper raised user-based collaborative filtering. In 2000, Amazon came up with item-based collaborative filtering in their paper[20]. These two algorithms are very famous in business recommender systems.

2.2.1 User-based collaborative-filtering

In user-based collaborative filtering, it is considered that a user will like the items that are liked by users with whom have similar taste. So the first step of user-based collaborative-filtering is to find users with similar taste. In collaborative filtering, the users are considered similar when they like similar items. Simply speaking, given user u and v, $N(u)$ and $N(v)$ are items set liked by u and v respectively. So the similarity of u and v can be simply defined as:

$$s_{uv} = \frac{|N(u) \cap N(v)|}{|N(u) \cup N(v)|} \tag{2.4}$$

There are a lot of similarity algorithm, Equation 2.4 is one of them. User u's likeability for item i can be calculated by:

$$p_{ui} = \sum_{v \in S(u,k) \cap N(i)} s_{uv} p_{vi} \tag{2.5}$$

User/Item	Item A	Item B	Item C	Item D
User A	!		!	recommend
User B		!		
User C	!		!	!

Table 2.1. User-based CF

Table 2.1 is an example of User-based CF recommendation. According to the interest history of User A, only User C can be the neighbor of him, so Item D will be recommended to User A.

2.2.2 Item-based collaborative-filtering

Item-based collaborative-filtering is different, it assumes users will like items that are similar with items that the user liked before. So the first step of item-based collaborative-filtering is to find out items that are similar with what the user liked before. The core point of item-based collaborative-filtering is to calculate the similarity of two items. Item CF considers that items that are liked by more same users, the more similar they are. Assume $N(i)$ and $N(j)$ are user sets who like i and j respectively. So the similarity of i and j can be defined as:

$$s_{ij} = \frac{|N(i) \cap N(j)|}{|N(i) \cup N(j)|} \tag{2.6}$$

User u's likeability of item i can be calculated by:

$$p_{ui} = \sum_{j \in S(i,k) \cap N(u)} s_{ij} p_{uj} \tag{2.7}$$

Table 2.2 is an example of Item-based CF recommendation. According to the interest history of all the users for Item A, people who like Item A like Item C as well, so we can conclude that Item A is similar with Item C. While User C likes Item A, so we can deduce that perhaps User C likes Item C as well.

User/Item	Item A	Item B	Item C
User A	!		!
User B	!	!	!
User C	!		recommend

Table 2.2. Item-based CF

User-based and Item-based collaborative-filtering algorithms are all neighborhood-based algorithm, there are also a lot of other collaborative-filtering algorithms. Hoffman raised Latent Class Model in this paper[13], the model connects user and item by latent class, which considers that a user will not become interested in items directly. Instead, a user is interested in several categories that contain items, so the model will learn to create the categories according to user's behavior. On top of Latent Class Model, researchers came up with Matrix Decomposition Model, which is called Latent Factor Model[4] as well. There are a lot of models based on matrix decomposition and they mostly came from Netflix Prize Competition, such as RSVD[28], SVD++[18] and so on.

Besides Matrix Decomposition Model, Graph Model is widely applied in collaborative-filtering. Baluja introduced graph model of co-view behind the recommender algorithm of YouTube in [3] and also raised a broadcast algorithm on graph to measure how much a user like an item. This literature[27] research how to increase serendipity of recommendation result by means of the analysis of the path between nodes in the graph. Mirza[26] systematically studied recommendation problems based on graph model and point out the essence of the recommendation is to connect user and item. The graph is the natural method for that. [10] studies similarity algorithms between the nodes of the graph and compares the recommendation precision of different algorithms.

2.3 Hybrid Recommender Systems

Hybrid Recommender System is more and more popular currently. Combining collaborative filtering and content-based filtering can be more effective by recently research. There are many ways to implement hybrid recommender systems: simply combine the result of CF and CB recommendations, add CF capability to a CB method.[1]

There are seven hybridization methods:

- Weighted: Add scores from different recommender components.
- Switching: Choose methods by switching in different recommender components.
- Mixed: Show recommendation result from different systems.
- Features Combination: Extract features from different sources and combine them as a single input.
- Feature Augmentation: Calculate features by one recommender and put the result to the next step.
- Cascade: Generate a rough result by a recommender technique and recommend on the top of the previous result.
- Meta-level: Use the model generated by one recommender as the input of another recommender technique.

Although there are many combinations theoretically, it is not always efficacious for a specific problem. The most important principle of hybrid recommendation is to avoid or make up the weakness of every single recommender technique.[6]

2.4 Comparison

Each approach has its advantage and disadvantage, and the effects are different as well for different dataset. The approach may not suitable for all kinds of problems because of the algorithm itself. For example, it is hard to apply automate feature extraction to media data by content-based filtering method. And the recommendation result only limits to items the user ever chose, which means the diversity is not so good. It is very hard to recommend for users who never choose anything. Collaborative filtering method overcomes the disadvantage of mentioned before somehow. But CF based on big amount of history data, so there are problems of sparsity and cold start. In terms of cold start, as collaborative filtering is based on the similarity between the items chosen by users, there are not only new user problem[30], but also new item problem, which means it is hard to be recommended if the new item has never been recommended before[1]. The comparison is in Table 2.3[11].

2.5 Famous Recommender Systems

What is the difference between recommender system and search engine is that recommender system is based on the behaviors of user. There are a lot of websites using recommender system in the world. Personalized recommender system analyzes a huge amount of user behavior data and provides personalized content to different users, which improves the click rate and conversions of the website[36]. The fields that widely use recommender system are e-commerce, movie, video, music, social network, reading, local based service, personalized email and advertisement.

2.5.1 E-Commerce

The most famous e-commerce website, Amazon, is the active application and promoter of recommender system. The recommender system of Amazon reaches deeper into all kinds of products[32]. Figure 2.2 is the recommendation list of Amazon.

Apart from personalized recommendation list, another important application of recommender system is relevant recommendation list. When you buy something in Amazon, the relevant goods will be shown below [15]. Amazon has two kinds of relevant recommendation, one is customers who bought this item also bought in Figure 2.3.

Another is what other items do customers buy after viewing this item in Figure 2.4. The difference between the two recommendations is the calculation of the different user behaviors. The most important application of relevant recommendation is cross selling. When you are buying something, Amazon will tell you what other customers who bought this item also bought and let you decide whether buy it at the same time[7]. If you do, the goods will be packed and provide a certain discount.

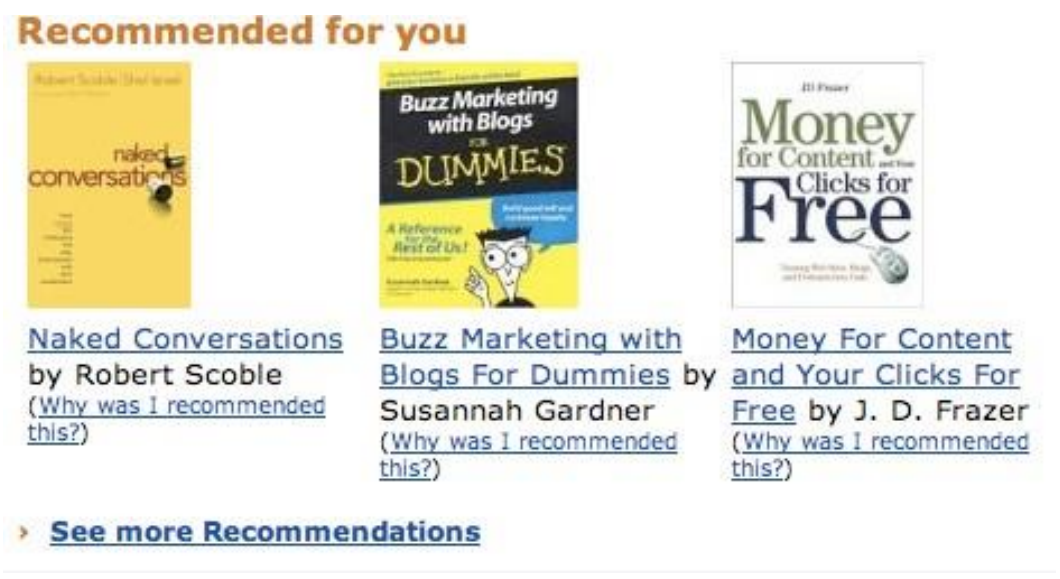
2.5.2 Movie and Video website

Personalized recommender system is a very important application for movie and video website, which can help users to find what they really like among the vast of videos. Netflix is the most successful company in this field[24]. Amazon and it are

Recommendation algorithms	Advantages	Disadvantages
Content based	Recommendation result is intuitive and easy to interpret; No need for users? access history data; No new item problem and no sparsity problem; Supported by the mature technology of classification learning.	Limited by the features extraction methods; New user problem; The training of classifier needs massive data; Poor scalability.

Collaboration filtering	No need for professional knowledge;
	Performance improving as the increasing of the user number;
	Automatic;
	Easy to find user?s new interesting point;
	Complex unstructured item can be processed. eg. Music, Video, etc.
		Sparsity problem;
		Poor scalability;
		New user and new item problem;
		The recommendation quality limited by the history data set.

Table 2.3. Comparison of contented-based and collaborative filtering


Recommended for you

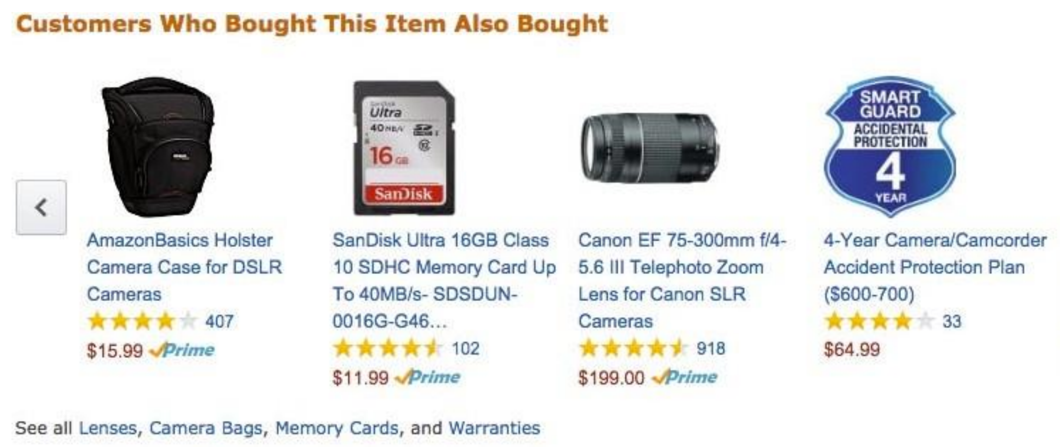
Naked Conversations by Robert Scoble (Why was I recommended this?)

Buzz Marketing with Blogs For Dummies by Susannah Gardner (Why was I recommended this?)

Money For Content and Your Clicks For Free by J. D. Frazer (Why was I recommended this?)

See more Recommendations

Figure 2.2. Personalized recommendation of Amazon


Customers Who Bought This Item Also Bought

AmazonBasics Holster Camera Case for DSLR Cameras ★★★★☆ 407 $15.99 Prime

SanDisk Ultra 16GB Class 10 SDHC Memory Card Up To 40MB/s- SDSDUN-0016G-G46... ★★★★½ 102 $11.99 Prime

Canon EF 75-300mm f/4-5.6 III Telephoto Zoom Lens for Canon SLR Cameras ★★★★½ 918 $199.00 Prime

4-Year Camera/Camcorder Accident Protection Plan ($600-700) ★★★★☆ 33 $64.99

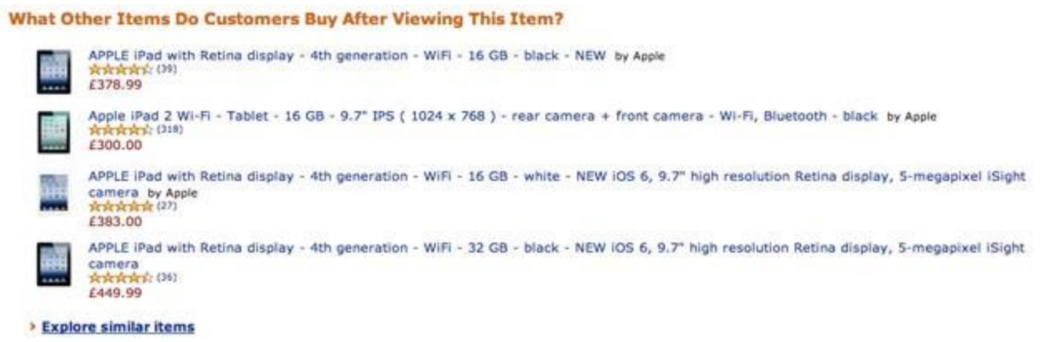
See all Lenses, Camera Bags, Memory Cards, and Warranties

Figure 2.3. Relevant Recommendation, Customers Who Bought This Item Also Bought

the two most representative companies in recommender systems.

Figure 2.5 is the recommendation page of Netflix. We can find that the recommendation result consists of the following parts.

- The title and poster of the movie.
- The feedback of the user, including Play, Rating and Not Interested.
- Recommendation reason.



What Other Items Do Customers Buy After Viewing This Item?

APPLE iPad with Retina display - 4th generation - WiFi - 16 GB - black - NEW by Apple
(39)
£378.99

Apple iPad 2 Wi-Fi - Tablet - 16 GB - 9.7" IPS (1024 x 768) - rear camera + front camera - Wi-Fi, Bluetooth - black by Apple
(318)
£300.00

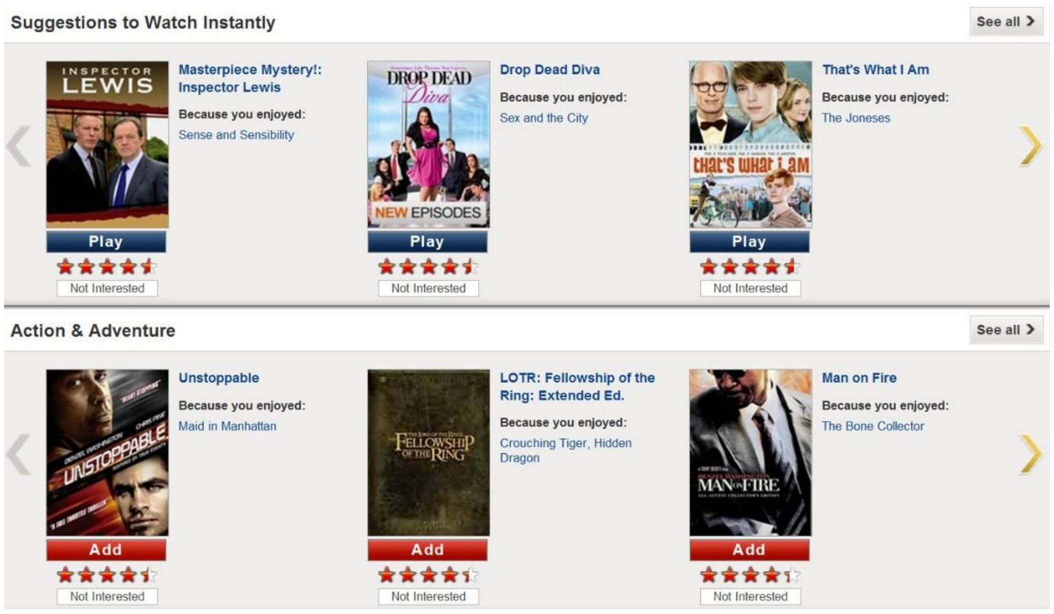
APPLE iPad with Retina display - 4th generation - WiFi - 16 GB - white - NEW iOS 6, 9.7" high resolution Retina display, 5-megapixel iSight camera by Apple
(27)
£383.00

APPLE iPad with Retina display - 4th generation - WiFi - 32 GB - black - NEW iOS 6, 9.7" high resolution Retina display, 5-megapixel iSight camera
(36)
£449.99

› Explore similar items

Figure 2.4. Relevant Recommendation, What Other Items Do Customers Buy After Viewing This Item

It can be illustrated that the recommendation algorithm is similar with Amazon according to the recommendation reason of Netflix. Netflix declared that 60% of their users find movies that they are interested in by the recommender system.[19]



Suggestions to Watch Instantly — See all

Masterpiece Mystery!: Inspector Lewis
Because you enjoyed:
Sense and Sensibility

Drop Dead Diva
Because you enjoyed:
Sex and the City

That's What I Am
Because you enjoyed:
The Joneses

Action & Adventure — See all

Unstoppable
Because you enjoyed:
Maid in Manhattan

LOTR: Fellowship of the Ring: Extended Ed.
Because you enjoyed:
Crouching Tiger, Hidden Dragon

Man on Fire
Because you enjoyed:
The Bone Collector

Figure 2.5. Netflix Recommender System

As the biggest video website in America, YouTube has a huge amount of videos that uploaded by users. So the information overload is a serious problem for YouTube. In the latest paper[8] of YouTube, in order to prove the availability of personalized recommender system, researchers of YouTube ever made an experience comparing the click rate of personalized recommendation list and popular list.

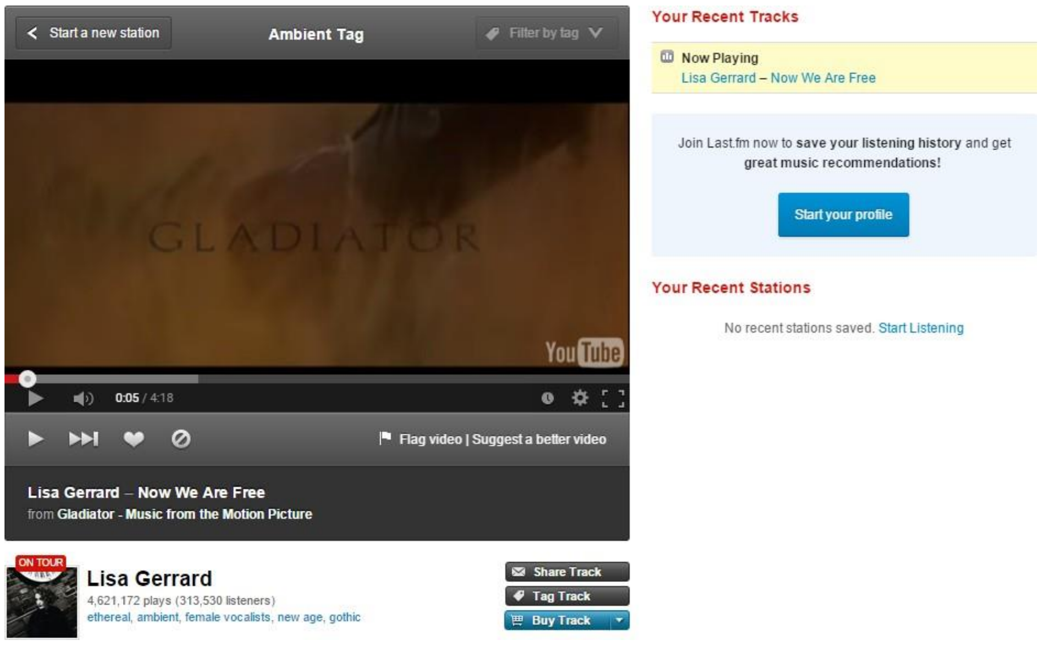
The result showed that click rate of personalized recommendation list is twice of that of popular list.

2.5.3 Internet Radio

The successful application of personalized recommender system needs two requirements, one is information overload because if users can find what they like easily, there is no reason to use recommender systems. The second is the user doesn't have clear requirements. Because they will use search engine directly if they do.

Under the two requirements, recommender system is very suitable for personalized Internet radio. First of all, people cannot listen to all the music in the world and find which one they like. Secondly, people often do not want to listen to a specific music, they wish to listen to whatever musics that match their mood at that moment.[16]

There are a lot of personalized Internet radios, such as Pandora and Last.fm. Figure 2.6 is the main page of Last.fm.


Start a new station
Ambient Tag
Filter by tag
GLADIATOR
YouTube
0:05 / 4:18
Flag video | Suggest a better video
Lisa Gerrard – Now We Are Free
from Gladiator - Music from the Motion Picture
ON TOUR
Lisa Gerrard
4,621,172 plays (313,530 listeners)
ethereal, ambient, female vocalists, new age, gothic
Share Track
Tag Track
Buy Track
Your Recent Tracks
Now Playing
Lisa Gerrard – Now We Are Free
Start your profile
Your Recent Stations
Start Listening

Join Last.fm now to save your listening history and get great music recommendations!

No recent stations saved.

Figure 2.6. Last.fm

Chapter 3

Recommender System for Vionel

3.1 Datasets

The datasets in this project are from Vionel database and public place such as Wikipedia. The data is in JSON format and each JSON object represents one movie information. We store the whole data in MongoDB, which is a cross-platform document-oriented database.

Each item of the data has a IMDb id as a unique identifier. There are now 8 features that are used in the project. They are *director*, *actor*, *genre*, *keyword*, *theme*, *scene* and *location*. Note that not all movies have complete features, some of the movies may miss some feature information, and this fact has been considered in the algorithm.

3.2 Feature Extraction

There are a lot of benefits to represent document as vector space model, for example we can measure cosine similarity and even extent to Clustering and Classification. However, vector space model does not have the ability to solve these two typical problems: a word has multiple meaning and a meaning has multiple words[22].

We ever consider to build a thesaurus, but it is too much time consuming. There are two mainstream approach, one is lexical co-occurrence and another is using shallow parsing to analyze the grammatical relation or syntax dependence between vocabulary[22]. Generally speaking, lexical co-occurrence is more robust and grammatical relation is more accurate.

In the previous vector space, we only pay attention on the frequency of a single word. But the word co-occurrence is a very important information as well, which based on a fact that the appearance of two or more co-occurrence words in the document is not occasional. Latent Semantic Indexing is an approach to explore the inner semantic relations[9][23]. The latent semantic indexing is to map the co-occurrence words to the same dimensional space. The co-occurrence words are considered semantically related.

Compared with our previous vector space, latent semantic space has less dimensions. That is because many words are mapped to the same dimension. Therefore, latent semantic indexing is a kind of dimensionality reduction method. The process of dimensionality reduction is mapping objects in high-dimensional space to low-dimensional space.

We can transform previous vector to term-document matrix, which is a $M * N$ matrix C consists of M terms and N documents. Each row of the matrix represents a term and each column of the matrix represents a document.

3.2.1 Singular Value Decomposition

Singular Value Decomposition is an important matrix decomposition in linear algebra. Let us see a theorem first.

Assume r is the *rank* of $M * N$ matrix C, so the SVD(Singular Value Decomposition) of matrix C is:

$$C = USV^T \tag{3.1}$$

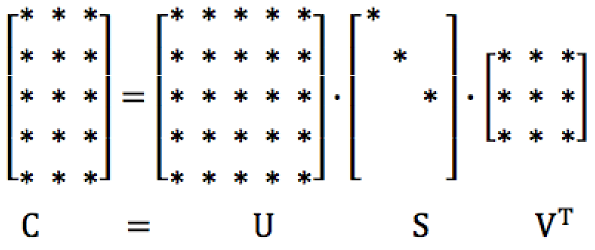


Figure 3.1. Singular Value Decomposition

Thereinto, the columns of U are orthonormal and so are columns of V. S is a diagonal matrix and elements on diagonal are singular value of C.

3.2.2 Matrix Low Rank Approximation

This is the process to solve matrix low rank approximation problem:

1. Given C, construct SVD by Equation 3.1 and decompose it: $C = USV^T$
2. Set $r - k$ minimum singular value on the diagonal of S zero, then get S_k
3. Calculate $C_k = US_kV^T$ as the approximation of C

3.2.3 Application of LSI

Matrix $S_k V^T$ is $k \times N$, k is the rank after using LSI, N is the total number of documents. Each column of matrix is the new coordinate of corresponding document on dimension-reduced space.

SVD is first raised by Deerwester[9] in Information Retrieval. LSI represent the documents on a new dimension-reduced space, which is actually the linear combination of each dimension on old space. It can be considered as a soft clustering.

Feature extraction is not the main task of the thesis, so I just introduced the basic principle simply, which is with the help of other research team in VionLabs.

3.3 Feature Representation

Vector Space Model

Before being deep into feature representation, it is very important to get the idea of document representation. Assume there are several documents and each document consists of one single sentence. Then we can represent the document as a model in Figure 3.2, which is called Vector Space Model. Consider each feature of a movie as a term, then a feature can represented by this model. It is obviously that some features are more important than others and the importance is the weight in the similarity calculation.

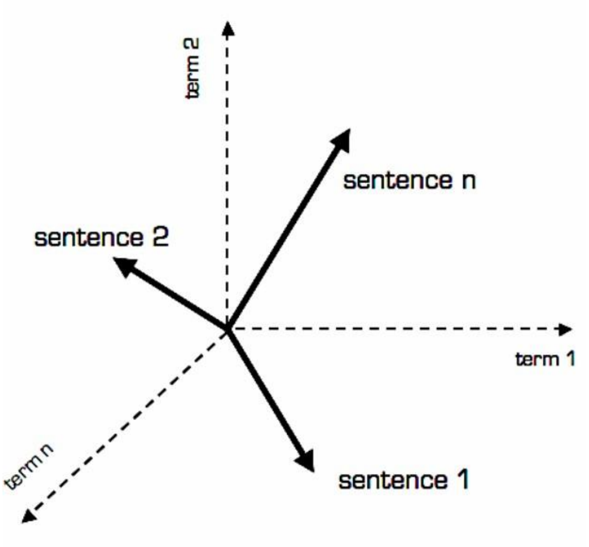


Figure 3.2. Vector Space Model of documents

3.4 TF-IDF

TF-IDF is short for term frequency-inverse document frequency that is a common weighting technique for information retrieval and text mining, which reflects how important a word is for a document[29]. The importance of a word increases proportionally to the times the word appears in the document, but it also decreases inversely proportional to the frequency the word appears in the whole corpus.

TF-IDF consists of TF and IDF, which are Term Frequency and Inverse Document Frequency respectively. TF represents the frequency a word appears in the document. The main idea of IDF is: if a term appears more in other documents, the term will be less important.

3.4.1 Term Frequency

Term frequency is about how many times a term t_i appears in document d_j, which can be represented by $TF(t_{ij})$. In the condition of removing stop-words, the more t_i appears in the document, the more important term t_i is for the document. It can be defined as:

$$TF(t_{ij}) = \frac{N(t_i, d_j)}{N(d_j)} \tag{3.2}$$

$N(t_i, d_j)$ is the number of times t_i appears in d_j and $N(d_j)$ is the total number of terms in document d_j

3.4.2 Inverse Document Frequency

In order to understand inverse document frequency, let us see what document frequency is. Document frequency is how many times term t_i appears in all documents C, which is represented by $N(t_i, C)$. The more term t_i appears in all documents C, the weaker term t_i can represent document d_j.

Inverse document frequency means that the represent ability of term t_i for document d_j and its amount in all documents $N(t_i, C)$ is inverse proportion, which is represented by $IDF(t_i)$:

$$IDF(t_i) = \log \frac{N(C)}{N(t_i, C)} \tag{3.3}$$

$N(C)$ is the total amount of documents, $IDF(t_i)$ decrease with the increase of $N(t_i, C)$. The less $N(t_i, C)$ is, the more representative t_i is for d_j.

3.4.3 Normalization

In order to reduce the inhibition of stop words, we will normalize each variable. After normalization, the calculation of $TF-IDF$ is:

$$weight_{TF-IDF}(t_{ij}) = \frac{TF(t_{ij}) \times IDF(t_i)}{\sqrt{\sum_{j=1}^{s}[TF(t_{ij}) \times IDF(t_i)]^2}} \tag{3.4}$$

Equation 3.4 base on the principle: The term that is more representative for a document is the word that appears in the document more often and less often in other documents.

3.5 Weakness of TF-IDF

The weakness of TF-IDF is mainly because of the fact that TF-IDF does not consider the same category of documents.

1. The concept of inverse document frequency is that the more number of times term t_i appears in document, the more representative term t_i is for the document, which does not consider the situation that the terms in same category. If the number of term t_i increases in the same category, it means that this t_i can represent the category very well. So the term should be given higher weight. But in Equation 3.3, the IDF will decrease when the number of term t_i in category C_l $N(t_i, C_l)$ increases. So we conclude: the weight of terms appearing frequently in same category should be strengthened.

2. For term t_1 and t_2, the appearance of t_1 is more average than t_2 in documents, so t_1 is more representative than t_2. If t_2 only appears in one or two documents of the category and almost does not appear in others, it can be considered that t_2 is of lower importance for the category. This kind of terms is not representative and they should be given lower weight. TF-IDF cannot solve this situation neither.

3.6 Improvement of TF-IDF

We researched related papers and found that the improvement approach is limited according to our analysis above.

Paper [25] introduces a new parameter to represent the in-category characteristic, which give us ideas about how to handle situation of terms in same category. TF-IDF focus on the ability to differentiate different documents, which ignores the fact that the term appears in the documents belonging to the same category. So in-category term frequency is something important for the improvement.

In paper [35], we know that term weight is positive correlated to their frequency. But it also points out that the higher frequency term maybe distributed in a part of the document. Such terms are not so representative for the whole document. The paper comes up the idea to solve the situation.

According to the analysis of Section 3.5 and research of related topic, we will improve TF-IDF base on the following two rules.

1. The number of term appearing in same category increases, the higher weight the term should be.
2. If the appearance of term in documents of a category is more even, the higher weight the term should be.

We came up with an improved feature weight algorithm $TF-IIDF-DC$:

$$weight_{TF-IIDF-DC}(t_{ij}) = TF \times IIDF \times DC \tag{3.5}$$

TF is term frequency, $IIDF$ is the improved inverse term frequency and DC is distribution coefficient.

3.6.1 Improved IDF

Consider the first rule mentioned above, we make the transformation:

$$IDF = \log \frac{N(C)}{N(t_i, C_l) + N(t_i, \overline{C_l})} \tag{3.6}$$

$N(t_i, C_l)$ represents how many times term t_i appears in category C_l, $N(t_i, \overline{C_l})$ represents how many times term t_i appears in other categories.

If a term mostly appears in a same category and less in others, the weight of the term for the category should be higher. So we introduce a coefficient λ for concentration, which is the in-category term frequency mentioned before:

$$\lambda = \frac{N(t_i, C_l)}{N(C_l)} \tag{3.7}$$

$N(C_l)$ is the total number of documents in category C_l, it is obvious that $\lambda \leq 1$. After plugging in concentrative coefficient λ, the IIDF is:

$$\begin{aligned} IIDF &= \log\left[\frac{N(C)}{N(t_i, C_l) + N(t_i, \overline{C_l})} \times \lambda\right] \\ &= \log\left[\frac{N(t_i, C_l)}{N(t_i, C_l) + N(t_i, \overline{C_l})} \times \frac{N(C)}{N(C_l)}\right] \end{aligned} \tag{3.8}$$

After a simple transformation:

$$IIDF = \log\left[\frac{1}{1+\frac{N(t_i,\bar{C_l})}{N(t_i,C_l)}} \times \frac{N(C)}{N(C_l)}\right] \tag{3.9}$$

$N(C)$ and $N(C_l)$ are constants, so we can see that when the number of term t_i appearing in category C_l increases, the $N(t_i, C_l)$ increases so that IIDF will increase. When $N(t_i, \bar{C_l})$ which is the number of term t_i appearing in other categories increases, IIDF will decrease. $N(t_i, \bar{C_l})$ is 0 when t_i only appear in category C_l, then IIDF gets the maximum value $\log\frac{N(C)}{N(C_l)}$. So t_i is the most representative for category C_l at this moment. When t_i appears in every document of category C_l, then $N(t_i, C_l) = N(C_l)$ and $N(t_i, C_l) + N(t_i, \bar{C_l}) = N(C)$, IIDF is 0 now. t_i is not representative for category C_l in this situation.

3.6.2 Distribution Coefficient

If the term t_i is distributed evenly in every document of a category, it is obvious that t_i is representative for the category. We define a distributed coefficient:

$$DC = 1 - \frac{\frac{1}{N(C_l)-1}\sum_{j=1}^{N(C_l)}\left[TF(t_{ij}) - \overline{TF}(t_i, C_l)\right]^2}{N(C_l) \times \overline{TF}(t_i, C_l)^2} \tag{3.10}$$

$\overline{TF}(t_i, C_l)$ is the average number of term t_i appearing in documents of category C_l.

$$\overline{TF}(t_i, C_l) = \frac{1}{N(C_l)}\sum_{j=1}^{N(C_l)} TF(t_{ij}) \tag{3.11}$$

When t_i appears in every document of C_l, it is obvious that $TF(t_{ij}) = \overline{TF}(t_i, C_l)$. Then DC is 1 which is the maximum value at the time. That means t_i is the most representative for category C_l. When term t_i only appears in one document of category C_l, we can easily know:

$$\sum_{j=1}^{N(C_l)}\left[TF(t_{ij}) - \overline{TF}(t_i, C_l)\right]^2 = (N(C_l)-1)\overline{TF}(t_i, C_l)^2 + \left[TF(t_{ij}) - \overline{TF}(t_i, C_l)\right]^2 \tag{3.12}$$

Then Equation 3.12 is substituted in Equation 3.10. We can see:

$$
\begin{aligned}
DC &= 1 - \frac{\frac{1}{N(C_l)-1}\sum_{j=1}^{N(C_l)}[TF(t_{ij}) - \overline{TF}(t_i, C_l)]^2}{N(C_l) \times \overline{TF}(t_i, C_l)^2} \\
&= 1 - \frac{\frac{1}{N(C_l)-1}[(N(C_l)-1)\overline{TF}(t_i, C_l)^2 + [TF(t_{ij}) - \overline{TF}(t_i, C_l)]^2]}{N(C_l) \times \overline{TF}(t_i, C_l)^2} \\
&= 1 - \frac{\frac{1}{N(C_l)-1}[(N(C_l)-1)\overline{TF}(t_i, C_l)^2 + [N(C_l) \times \overline{TF}(t_i, C_l) - \overline{TF}(t_i, C_l)]^2]}{N(C_l) \times \overline{TF}(t_i, C_l)^2} \\
&= 1 - \frac{\frac{1}{N(C_l)-1}[(N(C_l)-1) \times \overline{TF}(t_i, C_l)^2 + (N(C_l)-1)^2 \times \overline{TF}(t_i, C_l)^2]}{N(C_l) \times \overline{TF}(t_i, C_l)^2} \\
&= 1 - \frac{1 + [N(C_l)-1]}{N(C_l)} \\
&= 0
\end{aligned}
\tag{3.13}
$$

So DC is 0 when term t_i only appears in one document of category C_l, which means it is not representative for C_l.

3.6.3 Normalization

Finally, we got the equation for calculating weight of movie features after the normalization.

$$
weight(t_{ij}) = \frac{TF \times IIDF \times DC}{\sqrt{\sum_{i=1} (TF \times IIDF \times DC)^2}} \tag{3.14}
$$

3.7 Similarity

Calculate the similarity between movies is the objective of content-based recommender systems. The content can be anything such as text, video and image. In our project, each movie is represented by a feature vector. I will introduce the cosine similarity algorithm then.

Cosine Similarity is the most popular measurement for document similarity. In order to calculate the similarity between two features, we can calculate the cosine of the angle between the feature vector using Equation 3.15.

$$
similarity = \cos(\vartheta) = \frac{A \cdot B}{||A||||B||} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}} \tag{3.15}
$$

The range of the similarity is between -1 and 1. -1 means that the direction of the two vectors are totally opposite and 1 means they are in the same direction. The cosine similarity is 0 if the two vectors have no relationship. For text matching, it is obviously that the weights are non-negative, so the range should be 0 to 1 in our case. Here is an example for illustrating cosine similarity.[14]

Given two sentences:

- A: I like watching TV, but I don't like watching films.
- B: I don't like watching TV and films.

How can we calculate the similarity between the two sentences? The basic idea is: the more similar the words used by the two sentences are, the more similar the sentences are.

First step: Word segmentation.

- A: I / like / watching / TV, but / I / don't / like / watching / films.
- B: I / don't / like / watching / TV / and / films.

Second step: List all the words.

- I, like, watching, TV, but, don't, films, and

Third step: Word frequency calculation.

- A: I 2, like 2, watching 2, TV 1, but 1, don't 1, films 1, and 0.
- B: I 1, like 1, watching 1, TV 1, but 0, don't 1, films 1, and 1.

Forth step: Get word frequency vector.

- 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 3.15.

$$\cos(\vartheta) = \frac{2\times1+2\times1+2\times1+1\times1+1\times0+1\times1+1\times1+0\times1}{\sqrt{2^2+2^2+2^2+1^2+1^2+1^2+1^2+0^2}\times\sqrt{1^2+1^2+1^2+1^2+0^2+1^2+1^2+1^2}}$$

The value is 0.85 so that the two sentences are much similar.

Chapter 4

Experiment

In this chapter, we will introduce how to implement the content-based recommender system based on the principle mentioned. After that, we will test the system and give out the result to prove the improvement of our system.

4.1 Dataset

All the movie data we used is from IMDb, Wikipedia and our own database in VionLabs. In the end we get 178356 movies and related information. For the perspective of recommender system, a movie can be described by a collection of features, which can be genres, actors, directors and so on.

- *Director* : The director is from IMDb, most of movies only have one director, but some of them have two or more.
- *Actor* : A movie normally has a lot of actors, but most of them is useless for recommender system and bring disadvantageous effects. So we only get three main actors for one movie. They are from IMDb as well.
- *Keyword*: We use LSI to extract keywords from Wikipedia plot, which is under the help of the colleagues in VionLabs.
- *Release Year*: This is when the movie is released and data is from IMDb.
- *Vionel Theme*: Theme is a kind of keyword that describes movies in a different perspective, such as *Time Travel* and *Comic Book*. They are defined by VionLabs.
- *Language*: Language is from IMDb, which is the language that occurs in the movie.
- *Location*: Location is from IMDb, which is where the movie happens.

- *Vionel Scene*: Scene of the movie is analyzed by other research in our team. We will recognize the background of every frame in the movie by machine learning. For example, bar, hall room, store are what we recognized.

4.2 Category

In the scenario of movie, we will divide the movies into 23 categories by the normal genres. Table 4.1 shows the categories we used. Each movie in the case is a document, which is represented by the eight features described in Section 4.1. As we said before, the movie is represented by Vector Space Model, each feature for the movie is a term in the document.

Sci-Fi	Crime	Romance	Animation	Music
Comedy	War	Horror	Adventure	News
Biography	Thriller	Western	Mystery	Short
Drama	Action	Documentary	Musical	History
Family	Fantasy	Sport		

Table 4.1. Categories

In many other content-based recommender systems, the genre is used as vector to calculate similarity. But this is only one aspect of the movie and there are a lot of other features of movie such as background, actor, etc. So we add more features and some of them are very unique because they are extracted by our own research.

But we didn't simply add features together to calculate TF-IDF, we have discussed the reasons in Section 3.5. The genre is the natural feature that we can use as category. Figure 4.1 shows the distribution of genres in the movie database. Compared with the principle mentioned before, each genre is a category for documents. The number of document in each category is shown in the figure. Each document contains many terms which are features in our case, they are described in Section 4.1.

4.3 Document

As we discussed before, the document in our case is the movie which contains many features. The movie will be represented by vector space model in the experiment. In Section 4.1, we introduced the features that are used to model the movie. The vector space model is like this kind of format:

$$MovieModel = [Directors, Actors, Keywords, ReleaseYear, VionelThemes, \\ Languages, Locations, VionelScenes] \tag{4.1}$$

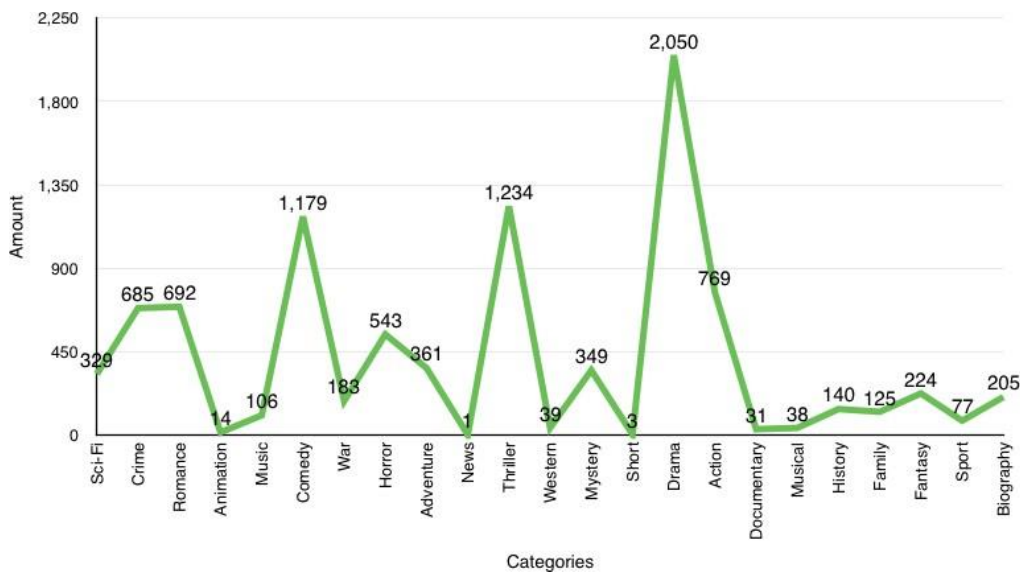


Figure 4.1. Category Distribution

A movie can have multiple directors and actors, so the vector is pretty long generally. Here we use movie *The Dark Knight* to illustrate the model.

Directors	Christopher Nolan
Actors	Christian Bale Heath Ledger Aaron Eckhart
Keywords	Keywords
Release Year	2008
Vionel Themes	Business and finance Comic book Bromance Cars and racing Mafia & Gangsters Revenge
Language	English, Mandarin
Location	USA, UK
Vionel Scene	Basement, Bar

Table 4.2. Information of The Dark Knight

Table 4.2 is the basic information of movie *The Dark Knight*. I don't list keywords in the table because there are 63 keywords for the movie, which is difficult to show them in the table.

We can get a very long vector after the calculation according to Equation 3.14, which is the model for the movie. Each weight represents the importance of a feature for the movie. In order to show it intuitively, I present it in Figure 4.2.

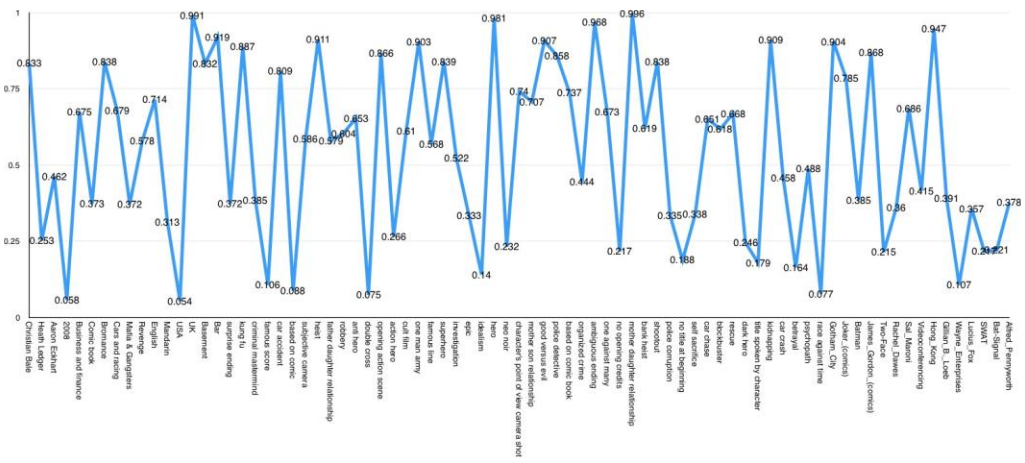


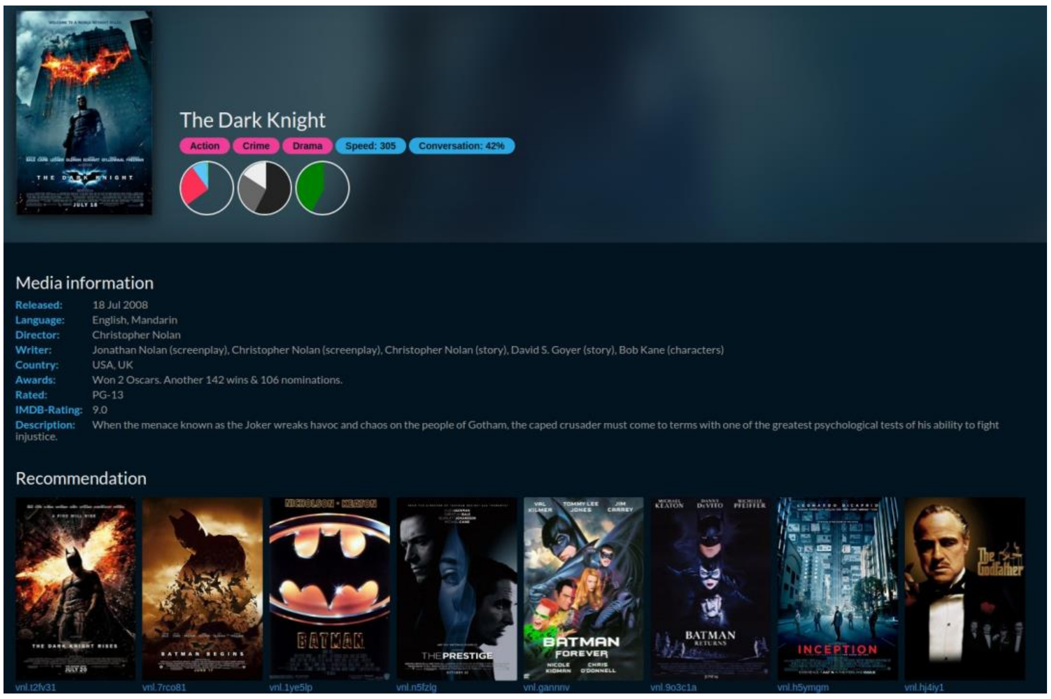
Figure 4.2. Feature Weight of The Dark Knight

There are 80 features totally for movie *The Dark Knight*, the number is calculated by our TF-IIDF-DC, which shows the importance of each feature. From this perspective, we can see that if features of a movie have similar distribution, it means that the two movies are similar.

Actually, one feature is one dimension for the model, I list all features in one dimension in the figure just because multiple-dimension is hard to show by figure.

4.4 Result

In our case, feature to movie is term to document. We can easily convert movie to vector space model which can be used to calculate the similarity. After previous calculation, every movie in the database can be represented by a vector. Then we use cosine similarity discussed in Section 3.7 to calculate similarity for each movie. Figure 4.3 shows the final recommendation of The Dark Knight, which is a screenshot of our current demo system.



The Dark Knight

Action Crime Drama Speed: 305 Conversation: 42%

Media information

Released:	18 Jul 2008
Language:	English, Mandarin
Director:	Christopher Nolan
Writer:	Jonathan Nolan (screenplay), Christopher Nolan (screenplay), Christopher Nolan (story), David S. Goyer (story), Bob Kane (characters)
Country:	USA, UK
Awards:	Won 2 Oscars. Another 142 wins & 106 nominations.
Rated:	PG-13
IMDB-Rating:	9.0
Description:	When the menace known as the Joker wreaks havoc and chaos on the people of Gotham, the caped crusader must come to terms with one of the greatest psychological tests of his ability to fight injustice.

Recommendation

vnl.t2fv31 vnl.7rco81 vnl.1ye5lp vnl.n5fzlg vnl.gannnv vnl.9o3c1a vnl.h5ymgm vnl.hj4iy1

Figure 4.3. Recommendation of The Dark Knight

Chapter 5

Validation

In order to illustrate the improvement of our recommender system, we will firstly prove our improved TF-IDF can generate better weight for features of the movie. In this chapter, we introduce a classification approach to validate the effect of movie model that is generated by different weight.

5.1 k-NN

k-Nearest Neighbors is a classical algorithm in Machine Learning. *k* represents its nearest *k* items.

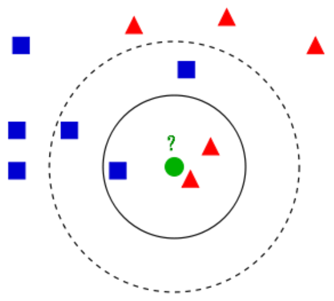


Figure 5.1. k-NN

Figure 5.1 is from Wikipedia of k-NN, we can see there are two types of items, one is blue rectangle and another is red triangle. The green cycle is the item to be classified.

In this case, if $k = 3$, the nearest items to the green cycle are two red triangles and one blue rectangle. This three items vote and the green cycle belongs to red triangle category. If $k = 5$, the nearest items to the green cycle are two red triangles and three blue rectangles. Then the five items vote and the green cycle should belong to the blue rectangle category.

5.2 Evaluation

For our movie case, we split all the movie data into two sets. One is used as unclassified set which is test set and another is classified set which is validation set.

For each movie in test set, we will calculate the similarity between it with all the movies in validation set. Then we can find the most k similar movies for each movie in the test set. So the movie to be classified should belong to the category that appears most in the k movies.

The k is set to 15 in our project which is decided by a lot of experience. We cannot make sure 15 is the best for our project but it is enough for the validation.

After that, every movie in test set is classified, which means every movie in test set is classified with a genre in our case. We have known the right genre of the movie, so we will find out the right and wrong classifications and measure the improvement by metrics in following section.

5.3 Evaluation Metrics

Precision and Recall are two measurements for statistics, which are used to evaluate the quality of statistic result. Precision is used to calculate the ratio of related documents with selected documents. Recall is used to calculate the ratio of related documents with all related documents in selected documents. Below is the definition of the precision and recall under the context of our movie case.

Assume TP_i represents the number of test documents belonging to C_i and they are classified to C_i as well. FP_i represents the number of test documents that do not belong to C_i are classified to C_i. FN_i represents the number of test documents belonging to C_i are classified to other categories. So the Precision and Recall in category C_i is defined by:

$$P = \frac{TP_i}{TP_i + FP_i} \tag{5.1}$$

$$R = \frac{TP_i}{TP_i + FN_i} \tag{5.2}$$

Generally we should comprehensively consider precision and recall, then we introduce F-Measure, which is defined in Equation 5.3.

$$F = \frac{2 \times P \times R}{P + R} \tag{5.3}$$

5.4 Analysis

In order to show the improvement of our TF-IIDF-DC, we split the data into 80% as training data and 20% as testing data. Both TF-IDF and TF-IIDF-DC are calculated and evaluated by Equations above. The result is in Table 5.1.

	TF-IDF			**TF-IIDF-DC**		
Category	**P**	**R**	**F**	**P**	**R**	**F**
Sci-Fi	86.64	83.56	85.07	89.32	85.51	87.37
Crime	78.57	75.98	77.25	85.54	81.19	83.31
Romance	70.65	67.87	69.23	76.76	69.34	72.86
Animation	78.78	77.32	78.04	85.34	81.49	83.37
Music	70.45	67.29	68.83	77.45	72.83	75.07
Comedy	80.67	75.19	77.83	88.76	82.73	85.64
War	77.87	72.87	75.29	85.91	77.34	81.40
Horror	82.34	80.17	81.24	89.21	84.65	86.87
Adventure	86.32	83.76	85.02	88.75	84.35	86.49
News	70.44	67.93	69.16	73.79	69.72	71.70
Biography	69.98	62.48	66.02	74.39	68.93	71.56
Thriller	79.89	78.28	79.08	84.18	80.38	82.24
Western	75.45	70.32	72.79	77.65	73.76	75.66
Mystery	71.22	68.95	70.07	76.85	71.45	74.05
Short	70.39	67.87	69.11	74.28	69.89	72.02
Drama	89.87	85.69	87.73	92.48	88.41	90.40
Action	80.43	76.21	78.26	84.56	81.44	82.97
Documentary	73.93	70.48	72.16	78.58	75.65	77.09
Musical	70.22	69.28	69.75	73.23	71.28	72.24
History	76.89	73.47	75.14	81.74	79.34	80.52
Family	73.49	70.34	71.88	76.43	72.67	74.50
Fantasy	73.68	71.94	72.80	77.98	74.87	76.39
Sport	77.23	73.45	75.29	80.64	75.34	77.90

Table 5.1. Comparison of TF-IDF and TF-IIDF-DC(%)

As we can see in Table 5.1 and Figure 5.2, the Precision, Recall and F-Measure by TF-IIDF-DC are all higher than that of TF-IDF. From the experiment, TF-IIDF-DC strengthens the weight of representative terms and weaken terms of no use in the category, which is so-called noise.

The most important factor for content-based recommender systems is feature. How to describe a movie is the most important task because the more accurate a movie is described, the better results recommender system generates. So from this perspective, we have proved the improvement of our approach in content-based

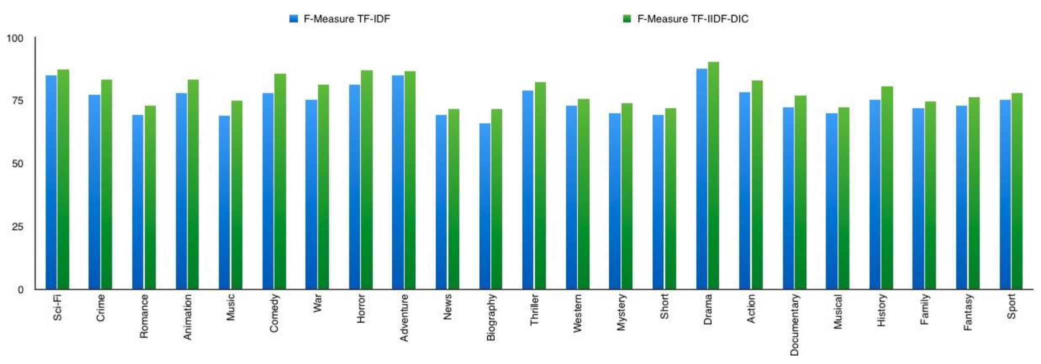


Figure 5.2. Comparision of F-Measure

recommender system.

Chapter 6

Conclusion

Recommender system has become more and more important because of the information overload. For content-based recommender system specifically, we attempt to find a new way to improve the accuracy of the representative of the movie.

For the problems we mentioned at beginning, firstly, we use content-based recommender algorithm which means there is no cold start problem. In Section 4.1, we list all the features in our recommender system. Some of them are from other research team in the company, so the features are diversity and more accurate than others. Then we introduced the cosine similarity which is commonly used in industry. For the weight of features, we introduced TF-IIDF-DC which improve the representative of the movie.

This master thesis introduces a content-based recommender system for the movie website of VionLabs. The features used in the system are extracted from various aspects of the movie, which are diversity and unique. We introduce a new approach for setting weight for these features, the movie can be represented more accurately by TF-IIDF-DC which is the key point of our research.

In the end of the project, we use k-NN and various metrics to evaluate the improvement of the new approach. It is illustrated that the new approach contributes positively according to the evaluation.

Chapter 7

Future Work

Recommender system has developed for many years, which ever entered a low point. In the past few years, the development of machine learning, large-scale network and high performance computing is promoting new development in this field. We will consider the following aspects in future work.

- Use collaborative filtering recommendation.
  After getting enough user data, collaborative filtering recommendation will be introduced. As we discussed in Section 2.2, collaborative filtering is based on the social information of users, which will be analyzed in the future research.[37]

- Introduce more precise and proper features of movie.[1]
  Typical collaborative filtering recommendation use the rating instead of object features. In the future we should extract features such as color and subtitle from movie which can provide a more accurate description for movie.

- Introduce user dislike movie list.
  The user data is always useful in recommender systems. In the future we will collect more user data and add user dislike movie list. We will input dislike movie list into the recommender system as well and generate scores that will be added to previous result. By this way we can improve the result of recommender system.

- Introduce machine learning.
  For future study, dynamic parameters will be introduced into recommender system, we will use machine learning to adjust the weight of each feature automatically and find the most suitable weights.

- Make the recommender system as an internal service.
  In the future, the recommender system is no longer a external website that will be just for testing. We will make it as an internal APIs for developers to invoke. Some movie lists in the website will be sorted by recom