FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Empirical study of the behavior of several recommender system methods on SAPO Videos

Guaicaipuro Alberto Oliveira Neves

FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Carlos Manuel Milheiro de Oliveira Pinto Soares

Co-supervisor: Tiago Daniel Sá Cunha

Junho 29, 2015

Empirical study of the behavior of several recommender system methods on SAPO Videos

Guaicaipuro Alberto Oliveira Neves

Mestrado Integrado em Engenharia Informática e Computação

Junho 29, 2015

Resumo

Nos últimos anos, a internet tornou-se numa ferramenta indispensável para qualquer empresa ou cidadão comum. O que levou à uma enorme quantidade de informações estar disponível aos seus utilizadores. Esta sobrecarga de informação tornou-se num problema urgente que faz com que o utilizador não consiga manter o controle dos seus próprios interesses. Para resolver este problema, os sistemas de recomendação são desenvolvidos para sugerir automaticamente itens que sejam do interesse dos utilizadores.

Existem várias estratégias de recomendação sendo as mais usualmente utilizadas o *collaborative filtering e o content-based filtering*. Ainda assim existem ainda muitas outras formas de recomendação das quais podemos identificar: *Social based filtering, Social tagging filtering, Knowledge-based filtering, hybrid filtering, context-aware filtering and time-aware filtering.*

Esta tese tem como objetivo realizar um estudo empírico sobre recomendação de vídeos no site do Sapo. A motivação para este trabalho foca-se em avaliar qual a melhor estratégia para o problems proposto.

Para realização deste estudo, é necessário fazer um levantamento de diferentes ferramentas de recomendação, recolher e preparar os dados a serem utilizados na plataforma experimental. Para este efeito o RiVaL toolkit é utilizado, de forma a executar diferentes estratégias de ferramentas distintas, avaliando-as sempre das mesma forma usando métricas de avaliação que mais se adequarem ao problema.

A avalição deste estudo empírico teve por base três métricas (Precision, RMSE, NDCG), tentando encontrar padrões dos resultados das mesmas em diferentes execuções do módulo experimental.

Depois do trabalho realizado conclui-se que as filtragens realizadas ao dataset original tem um grande impacto na performance final dos algoritmos. Obtendo-se no geral melhores resultados a nível de precisão e NDCG e piorando os resultados de RMSE.

Existem três algoritmos que merecem destaque, Item-based usando como similaridade Pearson's correlation, que obtém bons resultados na ferramenta Apache Mahout. No que diz respeito ao LensKit as estratégias de Matrix factorization tem sempre boa performance nas diferentes métricas. Pelo lado negativo a similaridade de cosine obtém sempre má performance em ambas ferramentas. No final, conclui-se que mesmo tendo um controlo de como os dados são tratados e avaliados em diferentes ferramentas os seus resultados não são totalmente comparáveis.

Abstract

In the last years, the internet became an indispensable tool for any company or internet user, which led to a huge amount of information being at every internet user's disposal. This information overload became a pressing problem making the user unable to keep track of his own interests. To solve this issue, recommender systems are developed to automatically suggest items to users that may fit their interests.

The are a numerous amount of different strategies used being the most popular collaborative filtering and content based filtering. Some others can be found in current bibliography about the subject like: 1) Social based filtering, 2) Social tagging filtering, 3) Knowledge-based filtering, 4) hybrid filtering, 5) context-aware filtering and 6)time-aware filtering. This last ones will not be focused in the state of the art because they are not applied in the empirical study.

This thesis aims to do an empirical study regarding recommender systems strategies for the Sapo Videos website. The motivation for this work lays with assessing which is the best strategy for the proposed problem, that leads to finding the best tool and evaluation metrics. There are a lot of different tools and metrics to implement and evaluate this kind of strategies finding the best one will point out that best strategy.

To accomplish this study it will be necessary to survey different recommendation tools, collect and prepare the data to be used on the experimental plataform. For this effect RIVAL toolkit is used allowing the use of different recommendation frameworks and ensuring control over the evaluation process.

The evaluation process used three common metrics (Precision, RMSE, NDCG), leading to patterns in their comparison and in different executions of the experimental module.

The first thing to notice is that the dataset filtrations have a huge impact on the performance, being that for precision and NDCG seems to only improve by increasing the filtering thresholds

In the end, it was concluded that even so the data was handled and evaluated the same way for the different frameworks, the results are not directly compared between them.

Agradecimentos

Gostaria de deixar o meu muito obrigado a todas as pessoas que me acompanharam ao longo do meu percurso académico, e que de algum modo, contríbuiram para a realização da minha dissertação, sendo este um marco importante, agradecer a quem me ajudou a chegar até aqui.

Agradeço ao Prof. Dr. Carlos Soares pela orientação e acompanhamento, pela partilha de conhecimento e pela compreensão durante o longo percurso percorrido na realização desta dissertação.

Agradeço também ao meu co-orientador e amigo Eng. Tiago Cunha por todo o trabalho, empenho e troca de conhecimentos. Acima de tudo agradeço a disponibilidade e acompanhamento contínuo duranto este ano.

Agradeço bastante a minha mãe pela paciência demostrada, pelos conselhos e pelo apoio prestados para a concretização desta etapa da minha vida.

Agradeço a minha tia pelo trabalho, e pelo empenho que sempre demonstrou num papel de uma excelente segunda mãe.

Agradeço também a todos os professores com quem tive oportunidade conhecer, e com quem aprendi algo para a minha formação futura.

Agora que irei agradecer aos meus amigos, a quem tive oportunidade de me cruzar não só durante a minha vida académica mas tabém fora dela, poupem-me os formalismos.

A vocês a família que escolhi para me acompanhar nestes longos sete anos, a quem conheci nos meus anos enquanto membro da AEFEUP. Aos ditos velhos, Ricardo Martins, Brunho Canhoto e Hugo Carvalho, por todos os ensinamentos e confiança que depositaram em mim.

Aos que comigo entraram e me acompanharam durante todos esses anos, ao quarteto de química, Mimi Mendes, Maria Afonso, Guida Santos e Inês Guimarães, obrigado pela companhia, pelos conselhos e pelos jantares quando estava desalojado. Ao Ricardo Rocha, Pedro Kretschmann, Tiago Baldaia e Diogo Moura, obrigado pelo companheirismo, pela ajuda e pelo apoio demonstrado nestes longos anos.

Aos que a seguir vieram e mesmo assim tenho o prazer de ter como amigos, José Pedro Nunes, Rita Magalhães, Bruno Guimarães, João Brochado e especialmente ao Bruno Sousa, por serem excelentes amigos e por toda a ajuda que sempre me prestaram.

Não podia concluir estes agradecimentos sem mais uma vez agradecer individualmente a ti Ricardo Martins, companheiro de todas as horas, porque nem todos os irmãos são de sangue.

A vocês agradeço,

Guaicaipuro Neves

À memória da melhor avó que qualquer pessoa podia desejar...

"There are two rules for success:
1) Never tell everything you know."

Roger H. Lincoln

Contents

List of Figures

List of Tables

Abbreviations and Symbols

CF	Collaborative Filtering
CBF	Content-based Filtering
MF	Matrix Factorization
SVD	Singular Value Decomposition
ALS	Alternating Least Squares
URL	Uniform Resource Locator
RMSE	Root Mean Square Error
NDCG	Normalized Discounted Cumulative Gain
CSV	Comma Separated Values

Chapter 1

Introduction

The expansion of the internet and the advent of Web 2.0 allowed users to do more than just access information. Instead of merely reading, a user is invited to comment on published articles, or create a user account or profile on the site. Major features of Web 2.0 include social networking sites, user created Web sites, self-publishing platforms, tagging, and social bookmarking (i.e Youtube, Facebook, Amazon, Blogs). Users can provide the data that is on a Web 2.0 site and exercise some control over it, transforming users from passive consumers to active content producers.

The recommendations generated by these systems aim to provide end users with suggestions about products or services that are likely to be of their interest.

There are several ways to achieve these recommendations: collaborative filtering methods recommend items that similar users like, while content-based filtering methods recommend items similar to those that the user liked in the past. A combination of different strategies can also be applied.

This has tremendously increased the amount of information that is available to users (Zanardi & Capra, 2008). With this information growth, recommender systems are gaining momentum, because it allows companies to make a more personalized approach for user item interactions.

The combination of personalized recommendations and the pure search and browsing, is a key method for information retrieval nowadays, since it allows users to handle the huge amount of information available in an efficient and satisfying way (Davidson et al., 2010).

1.1 Overview

There is a big number of tools that implement the different recommendation strategies, leading to the question of what is the best one to use in the specific problem. It is necessary to study them and try to understand which is better for the task at hand. An empirical study is a way of gaining knowledge regarding this question by means of direct and indirect observation or experience.

The results of such a study must be adequately evaluated. In recommender systems, the most common evaluation methodologies used in empirical studies are two methods, online evaluation (using users to directly evaluate the recommendations, and offline evaluation (using a set of evaluation metrics for the specific problem). There are numerous amount distinct metrics and each of them evaluated very different aspects of the recommendation (Bobadilla, Ortega, Hernando, & Gutiérrez, 2013).

1.2 Motivation

The purpose of this thesis is to give Sapo Videos[1] an idea of what is the best method or methods to make recommendations for their data. These kind of systems give companies like Sapo a huge competitive advantage because it allows its user to find videos they enjoy watching in a easy and direct way, which otherwise the user probably would not find. YouTube for example has a very powerful recommendation system in order to keep users entertained and engaged.

For a good use of recommendation system it is imperative that these recommendations are updated regularly and reflect a user's recent activity on the site, consequently these systems increase the number of users, a more importantly their loyalty.

As a scientific contribution this aims to give a comparison on the conclusions obtained by other similar empirical studies and if they apply using the data from Sapo Videos.

It is of the utmost importance that an empirical study is carried as a first step for recommendation system to be created at Sapo Videos. In a world where every minute there are hours of videos uploaded to the internet, this may be an important contribution for Sapo to thrive.

1.3 Goals

This thesis aims to do an empirical study of the behavior of different recommendation methods on the Sapo Videos data. This empirical study will need firstly the identification several recommender system methods that are representative of the different strategies.

Before the experimental structure implementation it is necessary to collect the data from Sapo Videos and prepare it to run the experiments. For this experimental structure the RiVaL toolkit was selected, because it includes three popular recommendation frameworks (Apache Mahout, LensKit, MyMediaLite).

Afterwards, the methods have to be evaluated with the data from Sapo Videos using appropriate metrics. The end goal of this thesis is to find patterns in the results obtained, discussing this results and comparing them with other empirical studies.

This will allow the gain of new conclusion of which is the best method for Sapo videos to proceed.

1.4 Thesis Stucture

Besides the introduction, these dissertation contains 4 more chapters.

[1] http://videos.sapo.pt/

In Chapter 2, the state of the art regarding recommender systems is described, as well as the most commons strategies of recommendation, how they are evaluated and some empirical studies already done on the subject.

Chapter 3, gives us a detailed insight on how the empirical study described in this thesis was conducted, how the data needed was collected and prepared for the recommendation toolkit used. This toolkit is then described, namely explaining how it works and how it fits in the implementation architecture developed.

In chapter 4, the results obtained are presented and discussed. These conclusions are then compared to those obtained in another study, also made with RiVaL toolkit.

Chapter 5 presents the conclusions of this thesis and some suggestions of future work.

Chapter 2

Recommender Systems

This transformation of users from passive consumers to active producers of content generated an amount of data online which makes it impossible for the users to keep up with. To deal with this problem, recommender systems arose to automatically suggest items that may interest the user (Bobadilla et al., 2013; Yang, Guo, Liu, & Steck, 2014). Although the roots of recommender systems can be traced back to the extensive work in other scientific areas, recommender systems appeared as an independent research in the mid-1990s (Adomavicius & Tuzhilin, 2005; Yang et al., 2014). Recommender systems have become extremely common in recent years, and are applied in a variety of applications. The most popular ones are movies, music, news, books, research articles, and products in general, as it is presented in Figure 2.1.

Website	Items recommended
Amazon	Books/ other products
Facebook	Friends
WeFollow	Friends
MovieLens	Movies
Nanocrowd	Movies
Jinni	Movies
Findory	News
Digg	News
Zite	News
Meehive	News
Netflix	DVDs
CDNOW	CDs/DVDs
eHarmony	Dates
Chemistry	Dates
True.com	Dates
Perfectmatch	Dates
CareerBuilder	Jobs
Monster	Jobs
Pandora	Music
Mufin	Music
StumbleUpon	Websites

Figure 2.1: Websites using Recommender Systems
(Lü et al., 2012)

Recommender Systems collect information on the preferences of its users for a set of items in an explicit (user's rating) or implicit form (user's behaviour) and make use of different types of information to provide its users with predictions and recommendations of items (Bobadilla et al., 2013).

In this chapter it is described the general concept of a recommender systems. We then explain in more detail the most common strategies of recommendation and what type of data, what algorithms are used and how they recommend an item.

The end of the chapter is focused on how recommender systems are evaluated and it is shown some conclusion of empirical studies on the subject.

2.1 Taxonomy

In this section it will be discussed the most common recommendations taxonomy, the most popular strategies and how they evolved to resolve typical recommendation problems.

2.1.1 Collaborative Filtering

Collaborative Filtering works under the assumption that if two users are similar they probably like the same items. The key idea behind collaborative filtering is to use the feedback from each individual user as a base for recommendations. This feedback can be distinguished between explicit feedback, where the user assigns a rating to an item, or implicit feedback, when for instance the user clicks on a link or sees a video (Yang et al., 2014).

Formally collaborative filtering can be represented by: the utility $u(c, s_i)$ of item s for user c is estimated based on the utilities $u(c_j, s_i)$ assigned to item s by those users $c_j \in C$ who are "similar" to user c.

In typical collaborative filtering based recommender systems the input data is a collection of user-item interactions. It is usually represented as an $U \times I$ user-item rating matrix, such as, u_n represents de users and i_n the number of items as we see in figure 2.2 (Yang et al., 2014; Sarwar, Karypis, Konstan, & Riedl, 2000).

	i_1	i_2	i_3	i_4	i_5	i_6	i_7
u_1	5		1	5			2
u_2	4	1		5		4	1
u_3	5		1		5	5	1
u_4			5			3	
u_5	2						5
u_6		2				5	

Figure 2.2: User-Item rating matrix
(Bobadilla et al., 2013)

Furthermore a widely accepted taxonomy divides recommender systems into memory-based and model-based methods. Essentially the former are heuristics that make rating predictions based on the entire collection of previously rated items by the users, whereas the latter use the collection of ratings to learn a model, which is then used to make recommendations (Bobadilla et al., 2013; Yang et al., 2014; Adomavicius & Tuzhilin, 2005). The most common memory-based algorithms in collaborative filtering are user-based nearest neighbor and item-based nearest neighbor. User-based and Item-based kNN recommend items for a particular user by, calculating the similarity between users or items respectively using similarity measures. The most common metrics for this type of methods are Pearson's correlation or Cosine similarity:(Sarwar et al., 2000)

Pearson's correlation – Similarity between two users u_0 and u_1, whereas v_k and w_k, are the vectors of u_0XS_i and u_1XS_i. It is measured by calculating:

$$sim(u_0,u_1) = \frac{\sum_{k=1}^{K}(v_k-\overline{v})(w_k-\overline{w})}{\sqrt{\sum_{k=1}^{K}(v_k-\overline{v})^2\sum_{k=1}^{K}(w_k-\overline{w})^2}} \tag{2.1}$$

Cosine measure – In this case users u_0 and u_1 are represented as two vectors, the similarity between them is measured by computing the cosine of the angle between them:

$$sim(u_0,u_1) = \frac{v.w}{||v||w||} \tag{2.2}$$

But it can also be found the use of other similarity measures as part of the list of possible ones to use in these algorithms. As it will be seen the analysis of results some of this may include Tanimoto coefficient and the familiar Euclidean distance.

Using the selected similarity measure we produce a neighborhood, for each user or item. User-based nearest neighbor then predicts the missing rating of a user u to an item s with the formula 2.3.

$$pred(u,s) = \overline{R}_u + \frac{\sum_{n\subset neighbours(u)} userSim(u,n).(R_{ni}-\overline{R}_n)}{\sum_{n\subset neighbours(u)} userSim(u,n)} \tag{2.3}$$

Being *Sim* the function of similarity used to calculate de similarity between users or items, r_{ui} is the rating of the user user u to an item i and r_u is the average value of recommendations for user u.

On the other hand Item based nearest neighbor predictions take into account the ratings users gave to similar items, using the equation: 2.4.

$$pred(u,s) = \frac{\sum_{j\in ratedItems(u)} Sim(s,j).R_{ui}}{\sum_{j\in ratedItems(u)} Sim(s,j)} \tag{2.4}$$

Despite collaborative filtering being a very popular approach it may potentially pose some problems including sparsity, scalability and the cold-start problem. Sparsity is the problem that appears because each user has typically provided only a few ratings and cannot cover the entire spectrum of items. This affects negatively the number of items for which recommendations can be produced. To reduce high sparsity problems in recommender system databases, certain studies have used dimensionality reduction techniques. Some reduction methods are based on Matrix Factorization (Bobadilla et al., 2013). Matrix factorization characterizes both items and users by vectors of factors implicit from item ratings of a certain user, being that the major challenge of these methods is computing the mapping of each item and user factor vectors (Koren, Bell, & Volinsky, 2009). After this mapping is complete the recommender system can easily estimate the rating to a given item.

Some of the most established techniques for identifying factors is singular value decomposition (SVD). Applying SVD in the collaborative filtering domain requires factoring the user-item rating matrix. Then there was an urge to improve the matrix factorization strategies. To optimize these strategies, the two most used approaches are the Stochastic gradient descent (SGD) and Alternating least squares (ALS)(Koren et al., 2009).

The Scalability problems appear since the computational complexity of these methods grows linearly with the number of users and items. This has a huge impact on typical commercial applications that can grow to several millions user and/or items(Sarwar et al., 2000)(Deshpande & Karypis, 2004). Some solutions appeared like Hadoop and Spark[1] for distributed processing, reducing the processing times of Map reduce algorithms. An example is MLI, an application programming interface designed to address the challenges of building data mining algorithms in a distributed setting.(Sparks & Talwalkar, 2013)

The cold start problem occurs when it is not possible to make reliable recommendations due to the initial lack of ratings and this problem can be sub-divided in new community, refers to the difficulty when starting up a recommendation system, in obtaining a sufficient amount of data for making reliable recommendations. New item problem, becomes evident when the new items are entered in the recommendation system do not usually have initial ratings, and therefore, they are not likely to be recommended. New user problem, appears since new users have not yet provided any rating in the recommendation system, they cannot receive any personalized recommendations (Bobadilla et al., 2013).

2.1.2 Content-Based Filtering

Content-based filtering makes recommendations based on user choices made in the past, and it follows the principle that items with similar attributes will be rated similarly (Bobadilla et al., 2013). For example, if a user likes a web page with the words "car", "engine" and "gasoline", the content-based recommender system will recommend pages related to the automotive world.

Content-based filtering has become more important due to the increase of social networks. Recommender systems show a clear trend to allow users to introduce content, such as comments, critiques, ratings, opinions and labels as well as to establish social relationship links. (Adomavicius & Tuzhilin, 2005) states that content-based recommendation methods, the utility $u_{c,s}$ of the item s for the user c is estimated based on the utilities u_{c,s_i} assigned by the user c to the items $s_i \in S$ that are similar to item s.

[1] http://mahout.apache.org/

In a video recommendation application, in order to recommend movies to user u, the content-based filtering recommender system tries to understand the common features among the videos of the user u has rated highly in the past. One problem is that sometimes the item descriptions are unstructured. One common case is when items are represented by documents and it is important to organize the data in such a way that it is possible to conduct the computation of similarities, to address these problems. Researchers have applied data-mining and natural language processing techniques to automate the process of mining features from items as can be seen in Figure 2.3 (Dumitru et al., 2011).

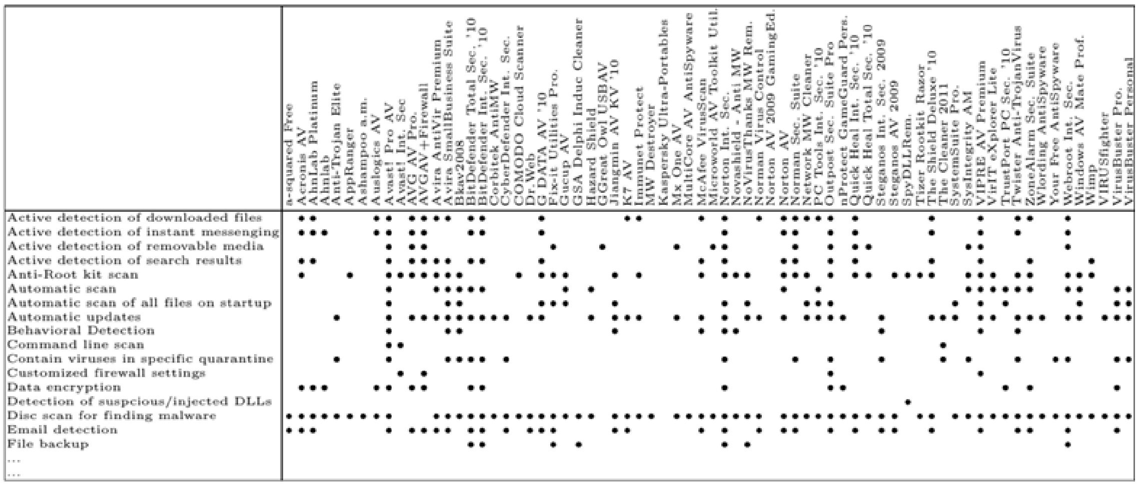


Figure 2.3: Example of features for selection

After the data is structured it is necessary to make a user profile relatively to the user's preferred items. There are two main sources of information, a model of the user's preferences that is basically a description of what types of items interest the user. A history of the user's interactions with the recommendation system, this interactions can be implicit or explicit(Pazzani & Billsus, 2007). Explicit interactions relies on something similar to "like" and "dislike" buttons in the user interface. By using them, the user can state her opinion on the current recommendation. Implicit interactions can for example observe the time the user is actually watching the program (Dumitru et al., 2011).

The key purpose for content-based filtering is to determine whether a user will like a specific item. This task is solved traditionally by using heuristic methods or classification algorithms, such us: rule induction, nearest neighbors, decision trees, association rules, linear classifiers, and probabilistic methods (Pazzani & Billsus, 2007; Bobadilla et al., 2013).

Two of the most important problems for content-based filtering are limited content analysis and overspecialization. The first problem emerges from the difficulty in extracting automated information from some types of content (e.g., images, video, audio and text), which can diminish the quality of the recommendations and introduce a large overhead.

The second one refers to the occurrence of users only receiving recommendations for items that are very similar to the items they liked. This means that, users are not receiving recommendations of items they would like but are unknown. Some authors refer this as serendipity (Bobadilla et al., 2013).

2.1.3 Hybrid Filtering

Because of the problems presented above, it is more common to find the combinations of Content-based filtering and Collaborative filtering which are known as Hybrid Filtering. Collaborative filtering solves content-based's filtering problems because it can function in any domain, it is less affected by overspecialization and it acquires feedback from users. Content-based filtering adds the following qualities to Collaborative Filtering, improvement to the quality of the predictions (because they are calculated with more information), and reduced impact from the cold-start and sparsity problems. A proper hybrid recommendation algorithm can be devised and applied to fit the type of information available in a specific domain (Jeong, 2010).

Content-Based and Collaborative Filtering can be combined in different ways. The different alternatives as shown in Figure 2.4 (Bobadilla et al., 2013). There are several different approaches to hybridization. It is possible to (A) implement collaborative and content-based methods separately and combine their predictions, (B) incorporate some content-based characteristics into a collaborative approach, (C) build a general unifying model that incorporates both content-based and collaborative characteristics and (D) include some collaborative characteristics into a content-based approach. The most challenging way is to construct a unified model (Bobadilla et al., 2013; Christakou, Vrettos, & Stafylopatis, 2007).

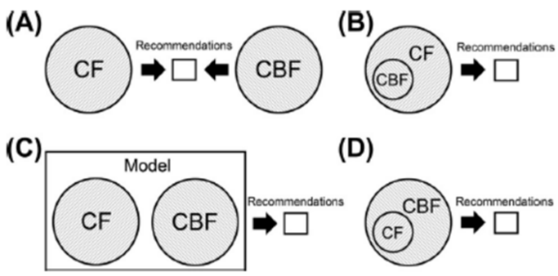


Figure 2.4: Different alternatives to combine Content-based and Collaborative Filtering (Bobadilla et al., 2013)

For example, (Melville, Mooney, & Nagarajan, 2002) uses hybridization in the movie recommendation domain. The basic approach uses content-based predictions to convert a sparse user ratings matrix into a full ratings matrix and then uses CF to provide recommendations. The research developed by (Christakou et al., 2007) considered a combination of CF and CBF, as the two approaches are proved to be almost complementary. A user evaluates films that he/she has seen on a discrete scale. This information allows the system to learn the preferences of the user and subsequently, construct the user's profile. We take into consideration two elements: the content of films that individuals have already seen and the films that persons with similar preferences have liked, as a result, they enhance both performance and reliability.

(Jeong, 2010) proposes a hybrid algorithm combining a modified Pearson's correlation coefficient-based collaborative filtering and distance-to-boundary (DTB) based content-based filtering. The study focused on developing a hybrid approach that suggests a high-quality recommendation method for a tremendous volume of data. The results indicated that this hybrid model performed better than the pure recommender system and by integrating both CB and CF strategies, the personalization engine provided a powerful recommendation solution.

Lastly, (Saveski & Mantrach, 2014) introduced a new method for Email Recipient recommendation that combines the content and collaborative information in a unified matrix factorization framework while exploiting the local geometrical structure of the data.

2.2 Evaluation

With the growth of the recommender systems research, assessing the performance of recommender systems became an important factor of success and more importantly, a way to gain a better understanding of recommender system behavior. This leads to an increase of evaluation approaches in an effort to determine the best approach and their individual strengths and weaknesses. Evaluating recommender systems and their algorithms can be very difficult for two reasons. First, different algorithms perform differently on different data sets. Secondly, the goals for which an evaluation is performed may be very different (Herlocker, Konstan, Terveen, & Riedl, 2004).

(Beel, Genzmehr, Langer, Nürnberger, & Gipp, 2013) state that finding the best recommender systems methods is not simple, and separates evaluation into three main methods: user studies, online evaluation and offline evaluation. User studies works on the basis that users explicitly rate recommendations generated by different algorithms, basically they quantify the user's satisfaction with the recommendations. In addition to this, user studies ask the user to rate a single aspect of the recommender system, but this method is not frequently used.

In online evaluation, recommendations are shown to real users of the system during their session. In this method users do not rate recommendations. Instead recommender systems capture how often the user accepts a recommendation. This acceptance is normally measured in click-through rate (CTR), which measures the ratio of clicks to impressions of an online item. The online method serves to implicitly measure the user's satisfaction.

Offline evaluations operates using datasets containing past users behavior from which some information has been removed (i.e the information removed can be the rating a user gave to an item). Afterwards, results are obtained by analyzing their ability to recommend/predict this missing information. (Campos, Díez, & Cantador, 2013; Herlocker et al., 2004; Beel et al., 2013). This process can be seen in Figure 2.6, basically the dataset is divided into training and testing datasets. The training set is sent to the recommendation model to train it on how the users will rate the item. The testing dataset set to try to predict the rating of the items the users has not rated yet or that have been removed for this purpose. To assess the performance of the recommendations the evaluations metrics are applied on the recommendations generated 2.5.

This means that we must split the data into training and testing datasets. The main differences are that the evaluation metrics must suit the problem at hand and one must split the test data into hidden and observable instances.

Name	Symbol	Preference	Scope	Rank	L
MAE	MAE	Small	Rating accuracy	No	No
RMSE	$RMSE$	Small	Rating accuracy	No	No
Pearson	PCC	Large	Rating correlation	No	No
Spearman	ρ	Large	Rating correlation	Yes	No
Kendall's Tau	τ	Large	Rating correlation	Yes	No
NDPM	$NDPM$	Small	Ranking correlation	Yes	No
Precision	$P(L)$	Large	Classification accuracy	No	Yes
Recall	$R(L)$	Large	Classification accuracy	No	Yes
F_1-score	$F_1(L)$	Large	Classification accuracy	No	Yes
AUC	AUC	Large	Classification accuracy	No	No
Ranking score	RS	Small	Ranking accuracy	Yes	No
Half-life utility	$HL(L)$	Large	Satisfaction	Yes	Yes
Discounted Cumulative Gain	$DCG(b, L)$	Large	Satisfaction and precision	Yes	Yes
Rank-biased Precision	$RBP(p, L)$	Large	Satisfaction and precision	Yes	Yes
Hamming distance	$H(L)$	Large	Inter-diversity	No	Yes
Intra-similarity	$I(L)$	Small	Intra-diversity	No	Yes
Popularity	$N(L)$	Small	Surprisal and novelty	No	Yes
Self-information	$U(L)$	Large	Unexpectedness	No	Yes
Coverage	$COV(L)$	Large	Coverage and diversity	No	Yes

Figure 2.5: Recommender System evaluation metrics
(Lü et al., 2012)

In order to evaluate Recommender Systems, several metrics are proposed (Bobadilla et al., 2013; Lü et al., 2012). Figure 2.5 shows some of the most common evaluation metrics and on what type of dataset they are most suited, as for on which aspect of the recommendation it evaluates.

Here, we focus on three main evaluation strategies. Firstly RMSE is a typically metric used to measure the error on the rating predictions and it is calculated by (Herlocker et al., 2004; Jiang, Liu, Tang, & Liu, 2011):

$$RMSE = \sqrt{\frac{\sum_{k=1}^{N}(p_{ui} - r_{ui})^2}{N}} \tag{2.5}$$

Precision and NDCG on the other hand are used for classification accuracy measurement, precision is the fraction of retrieved documents that are relevant, this is the fraction of true positives. The formula used to calculate precision is (Gunawardana & Shani, 2009):

$$Precision = \frac{TP}{TP + FP} \tag{2.6}$$

NDCG is the ratio between the DCG and the IDCG, which is the maximum possible gain value for user u (Baltrunas, Makcinskas, & Ricci, 2010), it measures the usefulness, or gain, of an item based on its position in the result list. It is calculated with:

$$NDCG_k^u = \frac{DCG_k^u}{IDCG_k^u} \tag{2.7}$$

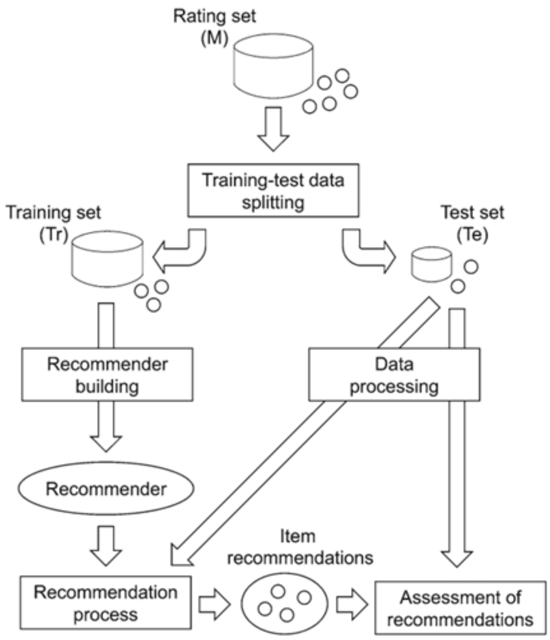


Figure 2.6: Schematic view of the stages followed in an offline evaluation protocol for RS (Campos et al., 2013)

2.3 Empirical Studies

Table 2.1 presents some empirical studies on the area of recommender systems, which have to be analyzed to find the main conclusion of the experiments.

In this case it is analyzed which datasets were used, which algorithms for recommendations they tried and some of the main conclusion their drew from they study.

(Adomavicius & Zhang, 2012) tried to understand the stability of the most common collaborative filtering algorithms.

(Ekstrand et al., 2014) used online evaluation metrics trying to prove that, offline metrics fail to capture much of what will impact the user's recommendation experience.

The only empirical studies found that used content-based filtering was (Cantador, Bellogín, & Vallet, 2010), that compared three methods of content-based filtering (FIDF Cosine-based Similarity, BM25-based Similarity and BM25 Cosine-based Similarity), on two different data types (Delicious and Last.fm).

As it is shown most of the studies experiment only with collaborative filtering, this may be due to the fact that is still on of the most used methods for recommendations and there is huge amount of public tools that implement collaborative filtering algorithms.

Table 2.1: Empirical studies

Paper	Data types	Algorithms	Main Conclusions
(Adomavicius & Zhang, 2012)	Movielens, Joker	Item Average; User Average; User-ItemAverage; Item-based kNN; User-based kNN; Matrix Factorization (SVD)	Model-based techniques are more stable than memory-based collaborative filtering heuristics; Normalizing rating data before applying any algorithms not only improves accuracy for all recommendation algorithms, but also plays a critical role in improving their stability;
(Ekstrand, Harper, Willemsen, & Konstan, 2014)	Movielens	Item-based CF; User-based CF; Matrix Factorization (SVD)	Item-based CF and SVD performed very similarly, with users preferring them in roughly equal measure; Offline metrics fail to capture much of what will impact the user's experience with a recommender system.
(Said & Bellogín, 2014)	MovieLens, Yelp	User-based CF; Item-based CF; Matrix factorization(SVD)	Different frameworks implement and evaluate the same algorithms in distinct ways leading to the relative performance of two or more algorithms evaluated under different conditions becoming essentially meaningless.;
(Vargas & Castells, 2011)	Delicious, Last.fm	TF-IDF Cosine-based Similarity; BM25-based Similarity; BM25 Cosine-based Similarity	In general,the models focused on user profiles outperformed the models oriented to item profiles; Regarding cosine-based models, by performing a weighting scheme that exploits the whole folksonomy, clearly enhance the classic frequency profile representation;

2.4 Frameworks

Table 2.2 represents a list of public frameworks that implement several recommendations strategies and methods. As it is shown all the frameworks implement only collaborative filtering strategies. This is because collaborative filtering is still the most used method of recommendation.

There is a tendency for the most recent frameworks to implement matrix factorization algorithms, since it attenuates some of the cold start and sparsity problems that the base collaborative filtering algorithms have.

We could not find any framework that implements content-based methods. This may be caused by the fact that most Content-based filtering techniques use traditionally text mining or classification algorithms with a vector space model strategy (Pazzani & Billsus, 2007; Bobadilla et al., 2013).

This empirical study is focused on the two different frameworks that will be used: Apache Mahout[10], is a project of the Apache Software Foundation focused primarily in the areas of collab-

[10]http://mahout.apache.org/

Table 2.2: Recommendation frameworks

Name	Strategies	Algorithms	Datasets	Evaluation Metrics
LensKit [2]	Collaborative Filtering	User/Item-based CF;Matrix Factorization	MovieLens 100K	normalized discounted cumulative gain; actual length of the top-N list
PREA [3]	Collaborative Filtering	User/item-based CF;Slope One;Matrix Factorization	N/A	Root of the Mean Square Error (RMSE);Mean Absolute Error (MAE); Normalized Mean Absolute Error (NMAE);Asymmetric Measures; Half-Life Utility (HLU);Normalized Discounted Cumulative Gain (NDCG)
Duine [4]	Collaborative Filtering	User-based kNN	MovieLens 100K	Root of the Mean Square Error (RMSE); Mean Absolute Error (MAE);
MyMediaLite [5]	Collaborative Filtering	User/Item based CF	MovieLens 1M/10M	Mean Absolute Error (MAE); Root of the Mean Square Error (RMSE);Area Under Curve(AUC); MAP; Negative Discount Comulative Gain(NDCG)
Crab [6]	Collaborative Filtering	User/Item based CF	N/A	Precision;Recall
Apache Mahout [7]	Collaborative Filtering	User/Item based CF; Matrix Factorization(SVD, ALS)	N/A	Average Absolute Difference Error (AADE); Root of the Mean Square Error (RMSE)
SVDFeature [8]	Collaborative Filtering	Matrix Factorization (SVD)	N/A	Root of the Mean Square Error (RMSE)
PredictionIO [9]	Collaborative Filtering	Matrix Factorization (SVD, ALS, PCA)	N/A	Mean Square Error (MSE);

orative filtering, clustering and classification. LensKit[11], is a free open source software developed primarily by researchers at Texas State University and GroupLens Research at the University of Minnesota.

2.5 Summary

This chapter focus on recommender systems, the most used recommendation strategies, how this systems performance is evaluated, other empirical studies on the subject and existing recommendation frameworks. Collaborative filtering is the most commonly used strategy and it works on the assumption that if two users are similar they probably like the same items. It used the feedback of each individual user as a basis for recommendations. The feedback can be distinguished into two different categories implicit and explicit feedback. The first algorithms to appear to achieve recommendations where Item/User-based kNN. They calculate the similarity between items or users, using similarity measures (i.e Pearson's correlation, Cosine similarity). Although this strategy possess problems like scalability the cold start problem.

To resolve these problems other strategy is presented. Content-based filtering, it makes recommedations based on the choices that the user made in the past. In order to recommend items to and user, the content-based filtering strategy tries to understand the common features among the items that the user rated highly in the past. After this data is structured it is necessary to make the users profile relatively to the user's preferred items. The key purpose for content-based filtering is to determine whether a user will like a specific item. This task is solved traditionally by using heuristic methods or classification algorithms, such us: rule induction, nearest neighbors, decision trees, association rules, linear classifiers, and probabilistic methods.

A combination of the strategies present above is also possible, it is known as hybrid filtering. This type of strategy emerged in a urge diminish the problems other recommendation strategies.

Assessing the performance of recommender systems became an important factor of success and more importantly, a way to gain a better understanding of recommender system behavior. This leads to an increase of evaluation approaches. These evaluation approaches can be divided into two categories. Online evaluation, in which recommendations are shown to real users of the system during their session. In this method users do not rate recommendations. Instead recommender systems capture how often the user accepts a recommendation.

Whereas in offline evaluation methods operates using datasets containing past users behavior from which some information has been removed. Afterwards, results are obtained by analyzing their ability to recommend/predict this missing information.

Analyzing other empirical studies made in the field of recommender systems, we try to understand what are the main findings other authors take so they can be later compared to the ones made in this work.

[11] http://lenskit.org/

To the success of this empirical study it is important to understand which recommendation frameworks are most used. From this frameworks it is analyzed what strategies are implemented in each one, what algorithms and the evaluations metrics they have implemented for use.

Chapter 3

Experimental Methodology

This chapter describes the whole experimental procedure needed to run the experiments. The process has several steps that are described in more detail in the sections bellow.

3.1 SAPO Data

This section describes in full detail the first step in this empirical study, specifically how the data was collected and prepared for the experiments. It explains how the data was collected and processed from the SAPO servers.

The data preparation subsection characterizes how the data had to be altered so it would be accepted experimental module and followed the premise that should be handled the same way for the different frameworks.

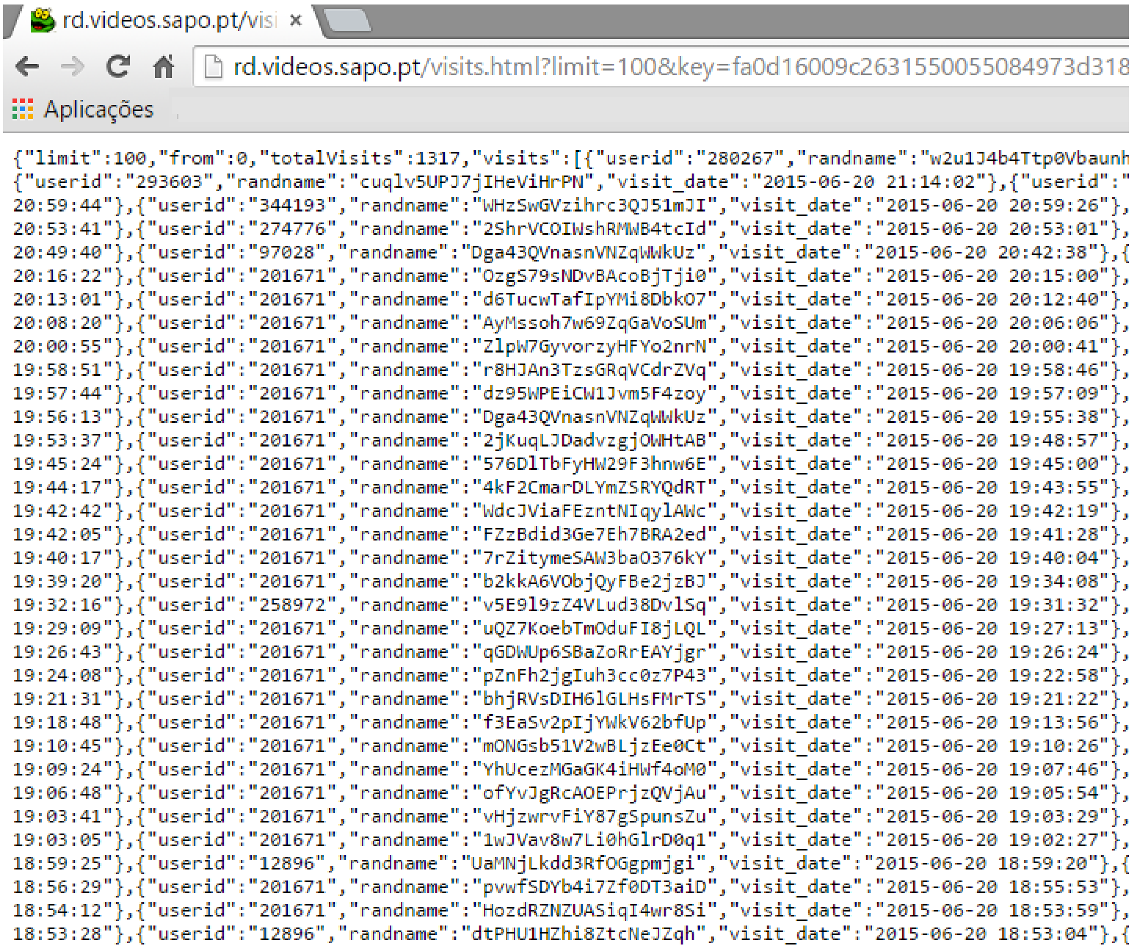
3.1.1 Data Collecting

The data was collected through a RSS feed provided by SAPO. To access it, it was necessary to set up a VPN to the Portugal Telecom intranet.

Figure 3.1 shows the aspect of the raw data stored for user-video interactions. It is important to notice that the data feed provided only keeps stored the data from the last three days of interactions. This limitation has the effect of only providing on average 1000 observations, so it was necessary to develop a solution that collected the data of long periods of time. Storing this three days of interactions of data in a database at a time to get a final dataset with enough observations to run the experiment.

We developed a Python script running on a Linux virtual machine using crontab to run once everyday storing only the new interactions. The script connected to the VPN and run through the feed comparing all the interactions present avoiding collecting duplicate information.

As it can be seen on figure 3.1, the information provided is an integer variable *userid* when 0 the user is not a registered user, a string *randname* as figure it is an unique string the appears on the end of the video URL and the full date when the user saw the video.



{"limit":100,"from":0,"totalVisits":1317,"visits":[{"userid":"280267","randname":"w2u1J4b4Ttp0Vbaunh
{"userid":"293603","randname":"cuqlv5UPJ7jIHeViHrPN","visit_date":"2015-06-20 21:14:02"},{"userid":"
20:59:44"},{"userid":"344193","randname":"WHzSwGVzihrc3QJ51mJI","visit_date":"2015-06-20 20:59:26"},
20:53:41"},{"userid":"274776","randname":"2ShrVCOIWshRMWB4tcId","visit_date":"2015-06-20 20:53:01"},
20:49:40"},{"userid":"97028","randname":"Dga43QVnasnVNZqWWkUz","visit_date":"2015-06-20 20:42:38"},{
20:16:22"},{"userid":"201671","randname":"OzgS79sNDvBAcoBjTji0","visit_date":"2015-06-20 20:15:00"},
20:13:01"},{"userid":"201671","randname":"d6TucwTafIpYMi8DbkO7","visit_date":"2015-06-20 20:12:40"},
20:08:20"},{"userid":"201671","randname":"AyMssoh7w69ZqGaVoSUm","visit_date":"2015-06-20 20:06:06"},
20:00:55"},{"userid":"201671","randname":"ZlpW7GyvorzyHFYo2nrN","visit_date":"2015-06-20 20:00:41"},
19:58:51"},{"userid":"201671","randname":"r8HJAn3TzsGRqVCdrZVq","visit_date":"2015-06-20 19:58:46"},
19:57:44"},{"userid":"201671","randname":"dz95WPEiCW1Jvm5F4zoy","visit_date":"2015-06-20 19:57:09"},
19:56:13"},{"userid":"201671","randname":"Dga43QVnasnVNZqWWkUz","visit_date":"2015-06-20 19:55:38"},
19:53:37"},{"userid":"201671","randname":"2jKuqLJDadvzgjOWHtAB","visit_date":"2015-06-20 19:48:57"},
19:45:24"},{"userid":"201671","randname":"576DlTbFyHW29F3hnw6E","visit_date":"2015-06-20 19:45:00"},
19:44:17"},{"userid":"201671","randname":"4kF2CmarDLYmZSRYQdRT","visit_date":"2015-06-20 19:43:55"},
19:42:42"},{"userid":"201671","randname":"WdcJViaFEzntNIqylAWc","visit_date":"2015-06-20 19:42:19"},
19:42:05"},{"userid":"201671","randname":"FZzBdid3Ge7Eh7BRA2ed","visit_date":"2015-06-20 19:41:28"},
19:40:17"},{"userid":"201671","randname":"7rZitymeSAW3baO376kY","visit_date":"2015-06-20 19:40:04"},
19:39:20"},{"userid":"201671","randname":"b2kkA6VObjQyFBe2jzBJ","visit_date":"2015-06-20 19:34:08"},
19:32:16"},{"userid":"258972","randname":"v5E9l9zZ4VLud38DvlSq","visit_date":"2015-06-20 19:31:32"},
19:29:09"},{"userid":"201671","randname":"uQZ7KoebTmOduFI8jLQL","visit_date":"2015-06-20 19:27:13"},
19:26:43"},{"userid":"201671","randname":"qGDWUp6SBaZoRrEAYjgr","visit_date":"2015-06-20 19:26:24"},
19:24:08"},{"userid":"201671","randname":"pZnFh2jgIuh3cc0z7P43","visit_date":"2015-06-20 19:22:58"},
19:21:31"},{"userid":"201671","randname":"bhjRVsDIH6lGLHsFMrTS","visit_date":"2015-06-20 19:21:22"},
19:18:48"},{"userid":"201671","randname":"f3EaSv2pIjYWkV62bfUp","visit_date":"2015-06-20 19:13:56"},
19:10:45"},{"userid":"201671","randname":"mONGsb51V2wBLjzEe0Ct","visit_date":"2015-06-20 19:10:26"},
19:09:24"},{"userid":"201671","randname":"YhUcezMGaGK4iHWf4oM0","visit_date":"2015-06-20 19:07:46"},
19:06:48"},{"userid":"201671","randname":"ofYvJgRcAOEPrjzQVjAu","visit_date":"2015-06-20 19:05:54"},
19:03:41"},{"userid":"201671","randname":"vHjzwrvFiY87gSpunsZu","visit_date":"2015-06-20 19:03:29"},
19:03:05"},{"userid":"201671","randname":"1wJVav8w7Li0hGlrD0q1","visit_date":"2015-06-20 19:02:27"},
18:59:25"},{"userid":"12896","randname":"UaMNjLkdd3RfOGgpmjgi","visit_date":"2015-06-20 18:59:20"},{
18:56:29"},{"userid":"201671","randname":"pvwfSDYb4i7Zf0DT3aiD","visit_date":"2015-06-20 18:55:53"},
18:54:12"},{"userid":"201671","randname":"HozdRZNZUASiqI4wr8Si","visit_date":"2015-06-20 18:53:59"},
18:53:28"},{"userid":"12896","randname":"dtPHU1HZhi8ZtcNeJZqh","visit_date":"2015-06-20 18:53:04"},{

Figure 3.1: RSS feed from SAPO Videos data

whenever a user that is not registered or is not logged in the website watches a video, the *userid* variable appears as 0. As this users can not be used for recommendation this potentially deceases the quality of the data collected as they have to be removed from the dataset.

3.1.2 Data Preparation

In this section it will be explained the improvements to the raw data, for example the users that were not registered while watching the videos need to be cut from the final dataset. As it will be explained in full detail, in the experimental methodology section some more changes needed to be made to the final dataset for it to run through the experimental structure already implemented, RiVaL toolkit 3.2.

RiVal toolkit was chosen to ensure the data was prepared, handled and evaluated in the same way, regardless of the method, algorithm or framework chosen. Some of the things to take into account for the dataset to run through RiVaL are that firstly as mentioned already the data collected was only implicit and RiVaL handled only explicit datasets in CSV format, making it impossible to use the data collected without changes. The item identification, in this case the randname, had to be a long variable and finally instead of a viewing date it was needed a timestamp instead.

The solution found was to build a new Python script that loaded the data from the database, handled these changes and saved that data on a csv file and when saving to CSV. Since we are working with implicit feedback(i.e user views), we introduced a new column with the number 1 on every line. Using Python direct conversion from full date directly to timestamp one problem was resolved.

Now the biggest issue was converting a randname variable to a long one. The key to solving this problem was to create an hash map that for every new line it would search the hash map to see if it was a new randname, in other words a new video. If it was it would add it to the hash map incrementing a new ID. If it already existed on the hash map it would give the new line the ID of the first time the randname appeared. The outcome is visible on Figure 3.2.

	A	B	C	D	E
1	254998	2	1	1429045525	1YZqE8W14qaShcrWGhZU
2	20621	2	1	1428212936	1YZqE8W14qaShcrWGhZU
3	20621	3	1	1428213137	MaPhA1QTEDtzbhlkJEza
4	320678	5	1	1428224917	Bz5FcHCcN8KEOvaDbz9z
5	32672	6	1	1428225376	ptjyvzzkyvDUnhvGrszl
6	32672	7	1	1428225527	ZfkkPHaDrzkZfCeR3ljv
7	32672	8	1	1428225642	lQnzr3KdGrfrBZd5TIpd
8	140729	3	1	1428226843	MaPhA1QTEDtzbhlkJEza

Figure 3.2: Aspect of final dataset in CSV format (UserId, New VideoID, Rating, Timestamp, Randname)

3.1.3 Exploratory Data Analysis

In this subsection it will be presented an analysis of the quality of the data as for some general statistics about it. The final dataset collected after the unregistered users had a total of 15939 observations, this is user-video interactions.

Of this it can be found 1008 unique users and 5880 unique videos the compose the total number of interactions. Observing below on figure 3.3, about the user interactions, the histogram presents the distribution of this users relating the number of interactions made present on the dataset. For example it can be seen that more than 800 users watched between 0 and less than 20 videos, and normally it is only between 0 and 5.

For the videos figure 3.4 gives some insights on the video distribution through the data. As it can be seen the for videos the distribution is even worse, showing that almost all the videos only are watched between 0 and less then 5 times. In conclusion making the user-item rating matrix for the data really sparse, being approximately 0.269

To resolve this problem the dataset was filtered trying to improve the global results. The process in which the data was filtered is explained in the next chapter 4.

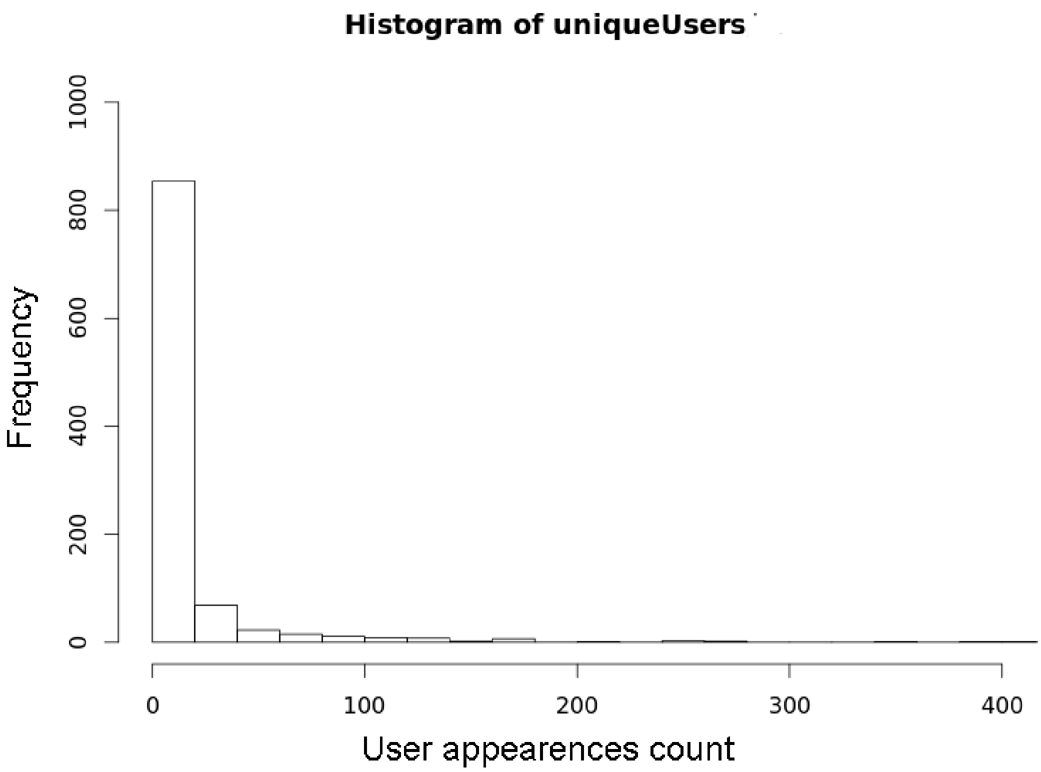


Figure 3.3: User distribution in dataset

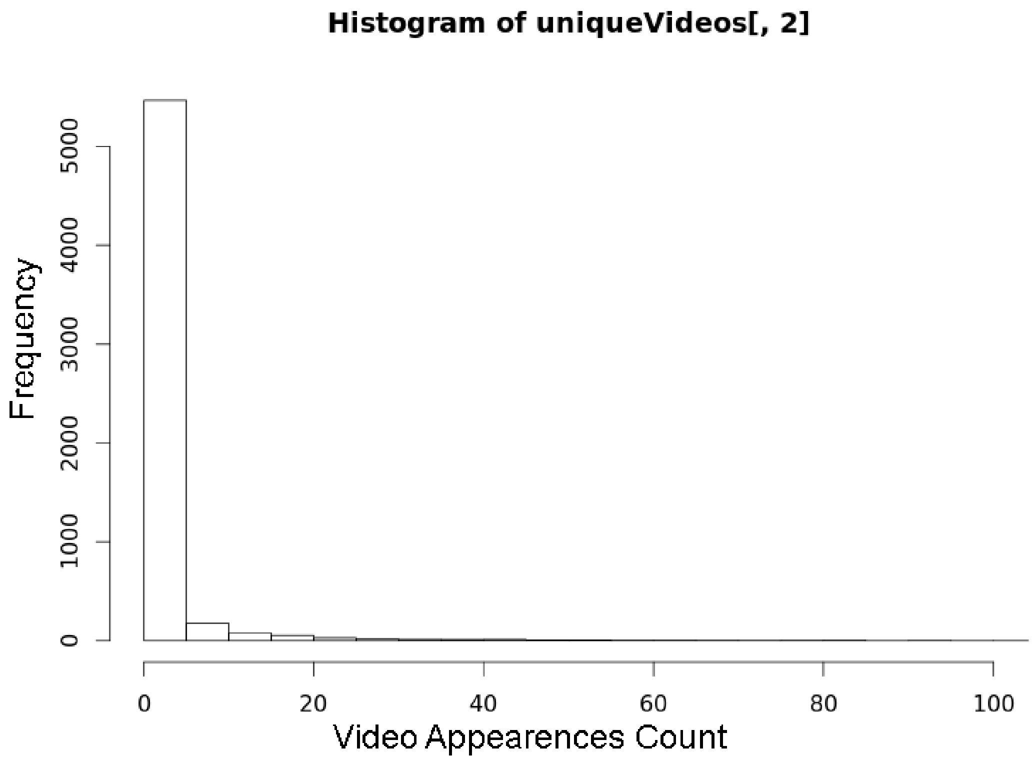


Figure 3.4: Video distribution in dataset

3.2 Implementation Details

This section describes with great detail the whole experimental structure. Starting how RiVaL toolkit 3.2.1 works and ensured a fine-grained control over the evaluation process.

The final subsection will explain full process behind the total implementation from the data collecting to the final results.

3.2.1 RiVaL Toolkit

The best solution found to accomplish this empirical study and the objectives proposed on Chapter 2 was RiVaL toolkit, an open source program developed in Java that allows a subtle control of the complete evaluation process (Said & Bellogín, 2014). RiVaL has integrated three main recommendation frameworks (Apache Mahout, LensKit and MyMediaLite), although in this empirical study we considered only two, Apache Mahout nad LensKit. The reason for leaving MyMediaLite out is because the version offered for download on the official website[1] did not had MyMediaLite available for use. Being that the documentation about RiVaL was weak not allowing understanding on why MyMediaLite was missing.

The recommendation process for RIVAL can be defined in four stages, i) data splitting; ii) item recommendation; iii) candidate item generation; iv) performance measurement. Of this four stages only three are performed by RiVaL, since it is not a recommendation framework. Step (iii) is not performed by RiVal, but can be performed by any of the three integrated frameworks (Mahout and LensKit).

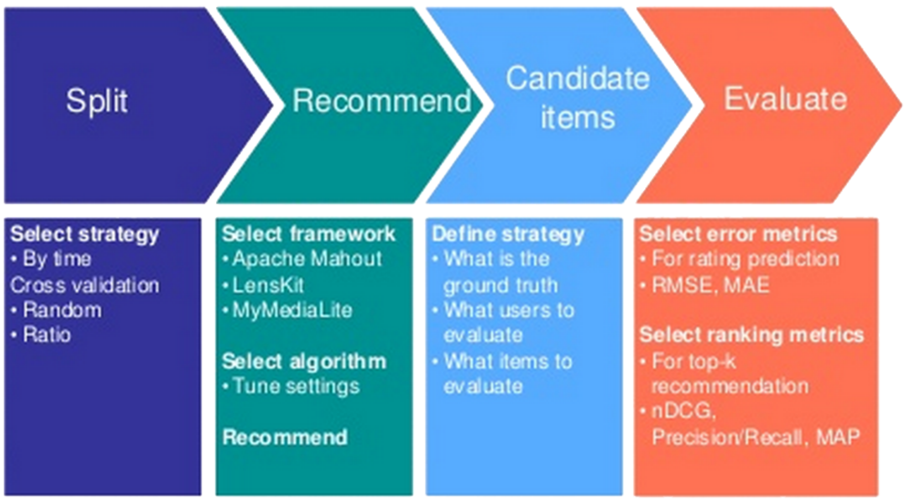


Figure 3.5: RIVAL's Work flow

[1] http://rival.recommenders.net/

In this case, steps (i), (iii), and (iv) are performed in the toolkit. As for step (ii) the preferred recommendation framework is given the data splits generated in the previous step and the recommendations produced by the framework are then given as input to step (iii) of the stages. Figure 3.5 illustrates RiVal's four stages.

To execute RiVaL, the evaluation selected, the algorithms, and the specific framework to use are specified in property files.

The example in Figure 3.6 shows the a configuration of RIVAL for the recommendation step executing an User-Based algorithm using cosine similarity with a neighborhood size of 50 of the LensKit recommendation framework.

```
recommender=org.grouplens.lenskit.knn.user.\
        UserUserItemScorer
similarity=org.grouplens.lenskit.vectors.\
        similarity.CosineVectorSimilarity
neighborhood=50
training=./trainset.scv
test=./testset.csv
output=./results.csv
framework=lenskit
```

Figure 3.6: Example of RIVAL recommendation configurations

RiVal toolkit had to suffer a set of changes for the Sapo dataset to run, never compromising the normal functioning of the work flow. This changes were mainly because RiVal was developed and tested with good datasets, this means no sparsity problems and always a good number of observations, believing that for every user on the dataset their would be a possible recommendation.

This led to, when running experiments, RiVaL could not handle users without possible recommendations and stopped running throwing exceptions errors. In the recommendation step it was added different conditional clauses to validate if the user had possible item recommendation, only then proceeding to the next step.

After RiVaL was running correctly for our data, it was needed to find out out algorithms would run for the distinct frameworks. For Apache Mahout the GenericItemBased (Item Based Knn), GenericUserBased (User Based Knn) with different similarity measures (Pearson's Correlation, Cosine Similarity, Tanimoto Coefficient and Spearman Rank). It also implemented Matrix Factorization types with distinct factorization types(FunkSVD, ALSW, Plus Plus factorizer and Rating SGD).

LensKit on the other hand has a fewer number of available algorithms running ItemItemScorer (Item Based Knn), UserUserScorer (User Based Knn) with Spearman Rank, Cosine similarity and Pearson's correlation as similarity measures.

3.2.2 System Architecture

To best explain the implementation system, figure 3.7 illustrates all the stages. The start of this process involves collecting the data from the RSS feed provided by SAPO videos.

Detailed explanation on how this how made can be found on 3.1.1 subsection.

This collected data was then stored in a database. The next step of the implementation was preparing the data and storing it on CSV file (more details on this step are established in 3.1.2). From this final dataset, some new datasets were created. The idea behind this new datasets and their explanation are explained on chapter 4.

The final stage in the all process was the experimentation process. In this stage all the different datasets, were submitted to RiVaL toolkit, subsection 3.2.1, and the evaluation results stored in the database for further processing.

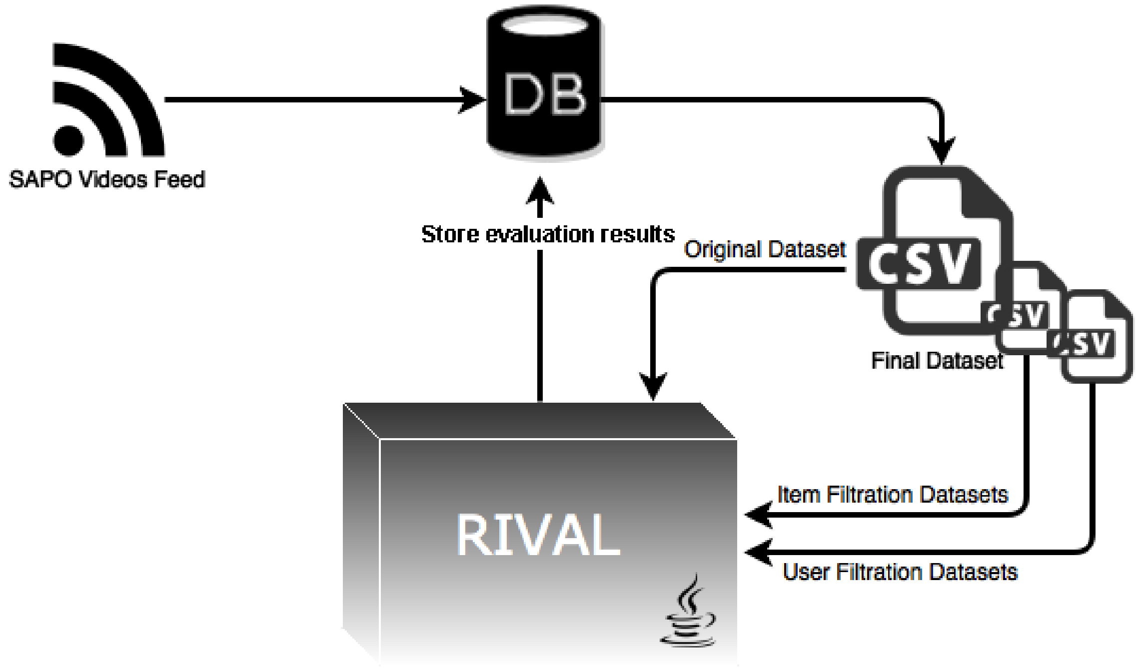


Figure 3.7: System Architecture

Chapter 4

Analysis and Discussion of Results

This chapter focus on the results obtained, it will be presented the result tables highlighting in green the best result and in red the worst result for each of the different metrics and frameworks in the different datasets. Afterwards conclusions are drawn from the results and why those happened.

4.1 Dataset Analysis

In the context of obtaining better results and producing a more in depth empirical study, ten different datasets were created from the original one. Because of the sparsity in data of the original dataset the results obtained were not satisfactory. These new datasets were created by filtering the original by video, selecting just the videos that had more than a certain number of visualizations, and by User, choosing users that had more than a certain number of videos seen. The selected thresholds were one, two, five, ten, twenty and fifty for both users and videos. These datasets are labeled as video_1 through video_50 for video filtering and user_1 through user user_50 for user filtering.

Another evaluation regarded was the variation of the distinct initial algorithm parameters like neighborhood size and the number of factors in the SVD strategies, in order to study their effects in the overall performance.

The whole empirical study was developed using three main strategies: User Based CF, Item Based CF and Matrix Factorization. Furthermore, different similarity classes and factorizer types were used. It is important to note that from the two frameworks available on RiVaL, Apache Mahout seems more extensive in their selection of strategies and algorithms when compared to LensKit.

The next sections present the results obtained using all these strategies and have been divided in, results of the original, video filtering and user filtering datasets.

4.1.1 Original Dataset

The tables presented below show the results for the different algorithms of the distinct frameworks. The evaluation metrics used, precision, RMSE and NDCG, there can be separated in two wider groups, both precision and NDCG are used to directly find a list of item to recommend,as a classification accuracy metric. RMSE on the other hand is used to predict the rating the a user would give to a specific item, a rating accuracy metric.

Table 4.1: Mahout Collaborative Filtering Original dataset

Algorithm	**NDCG**	**RMSE**	**Precision**	**NBsize**	**Similarity Class**
GenericItemBased	0,0240347	14,12554848	0,003645686	50	Tanimoto Coefficient
GenericItemBased	0,021813772	14,12555919	0,002123459	50	Euclidean Distance
GenericItemBased	0,021251396	14,12877938	0,002054042	50	Cosine Similarity
GenericItemBased	0,081575895	19,3978057	0,026406879	50	Pearson's Correlation
GenericUserBased	0,113550369	15,38540189	0,023660826	50	TanimotoCoefficient
GenericUserBased	0,105711383	16,18242996	0,015641249	50	Cosine Similarity
GenericUserBased	0,089444303	17,58941012	0,019419683	50	Euclidean Distance
GenericUserBased	0,131209389	19,48957225	0,032576487	50	Pearson's Correlation

Table 4.2: Mahout Matrix Factorization Original dataset

Algorithm	**NDCG**	**RMSE**	**Precision**	**Factorizer Type**	**Iteractions**	**Factors**
SVD	0,132614237	13,08272318	0,021578258	ALSWR	50	10
SVD	0,125891765	13,0947511	0,02134621	FunkSVD	50	10
SVD	0,123367101	13,03957183	0,02127789	SVDPlusPlus	50	10
SVD	0,075635674	13,17950689	0,013553344	RatingSGD	50	10
SVD	0,135204589	13,09583287	0,021243526	FunkSVD	50	30
SVD	0,132685476	13,09829961	0,021459288	RatingSGD	50	30
SVD	0,135005957	13,09281137	0,021653886	SVDPlusPlus	50	30
SVD	0,136016267	13,12582943	0,021359499	ALSWR	50	30
SVD	0,139377297	13,10819402	0,021236987	FunkSVD	50	50
SVD	0,137198693	13,13950122	0,022549697	RatingSGD	50	50
SVD	0,136445361	13,12344064	0,022443315	SVDPlusPlus	50	50
SVD	0,138968716	13,13464736	0,021567416	ALSWR	50	50

Table 4.3: LensKit Collaborative Filtering Original dataset

Algorithm	**NDCG**	**RMSE**	**Precison**	**Nbsize**	**Similarity Class**
UserUserScorer	0,052756987	13,39676723	0,007624473	50-150	Pearson's Correlation
UserUserScorer	0,042849141	13,2675676	0,007568793	50-150	Cosine Similarity
ItemItemScorer	0,012243574	13,29499233	0,000612559	50-150	Cosine Similarity
ItemItemScorer	0,049213951	18,58679494	0,011486631	50-150	Spearman Rank
ItemItemScorer	0,061317066	17,95627876	0,011355201	50-150	Pearson's

The first observation found after analyzing the results from the original dataset, is that the change in the initial parameters in the generic collaborative filtering techniques, did not change the outcome of the results. Only in the matrix factorization strategies, changing the parameters altered the outcome. For LensKit it seems that increasing the number of factors reduces the performance

Table 4.4: LensKit Matrix Factorization Original dataset

Algorithm	NDCG	RMSE	Precision	Factorizer Type	Iteractions	Factors
FunkSVD	0,062440759	14,14141109	0,009459459	FunkSVD	50	50
FunkSVD	0,099069374	14,16353719	0,011238299	FunkSVD	50	30
FunkSVD	0,134704152	14,1786524	0,012453622	FunkSVD	50	10

of the algorithms. On the other hand, Apache Mahout behavior is not stable, resulting in the fact that sometimes the increase of factor improves the algorithms results and sometimes it doesn't.

Generally it can be seen that the best strategy for recommendation in the original dataset both for Apache Mahout and LensKit is based on matrix factorization. The type of factorization though is not unanimous being that FunkSVD and ALS obtain the best results.

On the negative side, Item Based methods combined with cosine similarity measures rank last for the two frameworks on almost every metric evaluated.

4.1.2 Video Filtering

The tables in this subsection show the results for the video filtering datasets, as the number of user-item interactions and the sparsity are reduced their results will be presented individually.

Table 4.5: Mahout Collaborative Filtering Videos_1

Algorithm	NDCG	RMSE	Precision	NBSize	Similiraty Class
GenericItemBased	0,0212513962	14,2679383015	0,0020840416	50	Cosine Similarity
GenericItemBased	0,0815758948	19,5728241970	0,0294068786	50	Pearson's Correlation
GenericItemBased	0,0218137720	14,2622365919	0,0022159516	50	Euclidean Distance
GenericItemBased	0,0240346996	14,2070184848	0,0036567621	50	Tanimoto Coefficient
GenericUserBased	0,1312093894	19,9713895723	0,0354865192	50	Pearson's Correlation
GenericUserBased	0,1057113827	16,2785506073	0,0177412491	50	Cosine Similarity
GenericUserBased	0,0894443033	17,7010744101	0,0194196829	50	Euclidean Distance
GenericUserBased	0,1135503686	15,6128852002	0,0246608263	50	Tanimoto Coefficient

Some brief conclusion, that will be discussed more thorough in the next section, can be taken after all this results. Firstly it can be seen, as in the original dataset, that the initial parameters variation only affects the matrix factorization strategies.

Still on this subject it is to notice that the variation of the number of factors in the LensKit framework for the matrix factorization algorithms tends to almost always improve the performance in the evaluation metrics.

This does not happen frequently on the Apache Mahout framework. Usually the increase of the number of factors does not work in favor of better recommendations.

For Apache Mahout framework the combination of Item Based strategy with the pearson's correlation strategy seems to be the best in terms of precision, and the SVD recommendation strategy in NDCG and RMSE, although here the factorization type seems to change a bit in which one is the best.

Table 4.6: Mahout Matrix Factorization Videos_1

Algorithm	NDCG	RMSE	Precision	Factorizer	Iteractions	Factors
SVD	0,125891765	13,1419368	0,021478541	FunkSVD	50	10
SVD	0,075635674	13,26686894	0,015185873	RatingSGD	50	10
SVD	0,123367101	13,14669053	0,021563982	SVDPlusPlus	50	10
SVD	0,132614237	13,13637859	0,021814113	ALSWR	50	10
SVD	0,1352045893	13,1487615329	0,0238278226	FunkSVD	50	30
SVD	0,1326854755	13,1615260596	0,0227379193	RatingSGD	50	30
SVD	0,1350059568	13,1527947137	0,0226538857	SVDPlusPlus	50	30
SVD	0,1360162669	13,1557500431	0,0220673395	ALSWR	50	30
SVD	0,1393772968	13,1521255096	0,0235751587	FunkSVD	50	50
SVD	0,1371986929	13,1519711218	0,0234919697	RatingSGD	50	50
SVD	0,1364453611	13,1599653600	0,0232401503	SVDPlusPlus	50	50
SVD	0,1389687157	13,1551377364	0,0232415579	ALSWR	50	50

Table 4.7: LensKit Collaborative Filtering Videos_1

Algorithm	NDCG	RMSE	Precison	Nbsize	Similarity Class
UserUserScorer	0,013346787	13,13546895	0,007647289	50	Cosine Similarity
UserUserScorer	0,052756987	13,39676723	0,007544726	50	Pearson's Correlation
ItemItemScorer	0,061317066	17,95627876	0,013383565	50	Pearson's Correlation
ItemItemScorer	0,013297403	13,44419579	7,55E-04	50	Cosine Similarity
ItemItemScorer	0,049213951	18,58679494	0,011486631	50	Spearman Rank

Table 4.8: LensKit Matrix Factorization Videos_1

Algorithm	NDCG	RMSE	Precison	Factorizer	Iteractions	Factors
SVD	0,062440759	14,18221109	0,009459459	FunkSVD	50	50
SVD	0,099069374	14,18344997	0,01518283	FunkSVD	50	30
SVD	0,134704152	14,1956524	0,016216216	FunkSVD	50	10

On the other hand in the LensKit framework it is not so easy to find the best approach for each individual evaluation metric. It can be said that Item based strategy with cosine similary is a bad strategy to obtain a good precision, and Matrix factorization strategy is a good all around strategy for this framework.

In general it is clear that for video filtration datasets types Item based strategies are a good way to go, this is an expected result because the number of sparse elements is reduced.

4.1.3 User Filtering

This subsection presents the results of how the different strategies and algorithms behaved in the user filtering datasets. The evaluation metrics used are the same as in the other analysis.

Table 4.9: Mahout Collaborative Filtering Users_1

Algorithm	NDCG	RMSE	Precison	Nbsize	Similarity Class
GenericItemBased	0,017175921	16,01310287	0,001278781	50	Cosine Similarity
GenericItemBased	0,077697845	20,48474437	0,026948669	50	Pearson's Correlation
GenericItemBased	0,01658242	16,0083438	0,001533883	50	Euclidean Distance
GenericItemBased	0,024032787	15,97350116	0,002174908	50	Tanimoto Coefficient
GenericUserBased	0,193427039	21,80282736	0,107653061	50	Pearson's Correlation
GenericUserBased	0,077978159	18,84034312	0,012044316	50	Cosine Similarity
GenericUserBased	0,066654683	16,17781762	0,014621505	50	Euclidean Distance
GenericUserBased	0,103787542	16,37209262	0,024934999	50	Tanimoto Coefficient

Table 4.10: Mahout Matrix Factorization Users_1

Algorithm	NDCG	RMSE	Precison	Factorizer	Iteractions	Factors
SVD	0,108213086	14,91851637	0,020230408	FunkSVD	50	10
SVD	0,06325301	15,03543943	0,014700144	RatingSGD	50	10
SVD	0,10382793	14,92124018	0,020313637	SVDPlusPlus	50	10
SVD	0,106385091	14,91754256	0,020393382	ALSWR	50	10
SVD	0,107423826	14,9254806	0,021198959	FunkSVD	50	30
SVD	0,104560149	14,91946073	0,020953918	RatingSGD	50	30
SVD	0,106339582	14,91554964	0,021517939	SVDPlusPlus	50	30
SVD	0,104123463	14,92665467	0,020394543	ALSWR	50	30
SVD	0,108436542	14,92599922	0,02103947	FunkSVD	50	50
SVD	0,063603291	15,03230174	0,014701305	RatingSGD	50	50
SVD	0,108265567	14,91736896	0,021760657	SVDPlusPlus	50	50
SVD	0,107730393	14,91877184	0,021360772	ALSWR	50	50

Firstly, for Apache Mahout, as expected the User based algorithms in the user filtered datasets work better in general than any other algorithm, obtaining an overall better precision and NDCG results.

In terms of the similary measures it follows the same pattern as in the video filtering strategy, being that Pearson's correlation obtains the best results overall and cosine similary combined with a item based strategy is one of the worst.

Table 4.11: LensKit Collaborative Filtering Users_1

Algorithm	NDCG	RMSE	Precison	Nbsize	Similarity Class
UserUserScorer	0,034267686	15,64856226	0,007981481	50	Cosine Similarity
UserUserScorer	0,039401081	15,11053226	0,008039516	50	Pearson's Correlation
ItemItemScorer	0,038881931	19,84727894	0,009901232	50	Pearson's Correlation
ItemItemScorer	0,004591458	15,14708959	3,22E-04	50	Cosine Similarity
ItemItemScorer	0,037076904	19,38975846	0,012163321	50	Spearman Rank

Table 4.12: LensKit Matrix Factorization Users_1

Algorithm	NDCG	RMSE	Precison	Factorizer	Iteractions	Factors
FunkSVDItemScorer	0,106151567	14,0095381	0,014397906	FunkSVD	50	10
FunkSVDItemScorer	0,091129654	13,99858901	0,01289267	FunkSVD	50	30
FunkSVDItemScorer	0,06933908	13,99761578	0,01197644	FunkSVD	50	50

Similarly to the other datasets LensKit does not demonstrate a clear pattern on which is the worst although we can identify the Matrix factorization strategies as being the best in of all the results all across the evaluation metrics. Item based strategy with cosine similarity seems to be one of the worst.

In terms of the variation of parameters we can see that the neighborhood size continues to have no effect on the final result of the metrics. This can maybe explained by the high sparsity of the data, the increase of neighborhood size only increases the number ou line or columns (of the user-item rating matrix) taken into account. With this sparse data the new cells added, either lines ou columns, are probably blank, not changing the final performance of the algorithm.

On the other hand the number of factors on the Matrix factorization strategies has an impact on the results. On Apache Mahout framework there is not a correlation between the increase of the factors and the metrics result.

On LensKit side the increase of the number factors continues to only decrease the performance of the algorithms on precison and NDCG and improve de results of RMSE.

4.2 Discussion and Comparison of Results

4.2.1 Discussion

The first thing to notice is that the dataset filters have a huge impact on the performance. In particular, precision and NDCG seems to only improve by increasing the filtering thresholds. This can be explained by the amount of sparsity on the raw data. Reducing the number of total observations in favor of having more of one individual user or video made it easier for the different algorithms to both gain more information and predict more adequate items to an user.

The results RMSE on the other hand suggests otherwise. Looking closely it can be seen that it gets worst proportional to the reduction of the total number of observations of the filtered datasets. These pattern is unexpected, to understand it, more experiments would be necessary. These RMSE results may also have another plausible explanation. As it was already discussed in chapter 3, the type of data collected from SAPO videos consists of implicit feedback. Since all the algorithms presented in RiVaL work with explicit feedback, it is difficult for the algorithms to predict ratings correctly.

Another thing to point out is that for the generic collaborative filtering algorithms, changing the parameter neighborhood size did not modify the final results. We observe that the neighborhood size only increases the number of lines and columns considered for neighbourhood calculation. However, if these lines still maintain a considerable amount of sparsity the final performance of the algorithms do not change.

In terms of similarity measures for item and user based strategies, cosine similarity obtains bad results in most of the evaluation metrics for both frameworks. Since cosine similarity calculates the cosine of the angle formed between the two user vectors and with the sparsity of the data collected the angle between two similar is still so big that accomplishing a successful recommendation is hard.

4.2.2 Empirical Study Comparison

Although different datasets where used, it is important to compare the results obtained here with the empirical study carried out by the developers of RiVaL using this same framework(Said & Bellogín, 2014), in its discussion some arguments were presented.

Firstly it is clear that even though the evaluated frameworks presented on RiVal, implement the same recommendation algorithms in a controlled ambient, the results still can't be compared. The performances of algorithms implemented in one framework can not be compared to the performance of the same algorithm in another.

This conclusion presented in (Said & Bellogín, 2014), seems to be true no only for the public datasets (Movielens 100k, Movielens 1M, Yelp2013) as for the SAPO videos dataset collected for this project. Taking for example the results from the original dataset on Item-based algorithm for the Apache Mahout framework with the same similarity measure and initial parameters, the results for NDCG, RMSE and precision were 0.08, 19.39 and 0.03 respectively.

Using the same variables on LensKit framework for NDCG, RMSE and precision the results were 0.06, 17.96 and 0.01 respectively, the results are very distinct, and this happens for all algorithms.

Proving further evidence in favor of the authors perspective that when it comes to performance comparison of recommendation algorithms, a standardized evaluation is crucial. To objectively and definitively characterize the performance of an algorithm, a controlled evaluation, with a defined evaluation protocol is extremely necessary.

Also (Said & Bellogín, 2014) stated that there are large differences in recommendation quality, not just for the distinct algorithms, but especially for the same algorithms implemented in the different frameworks presented on RiVaL.

For example, not all the frameworks are able to deal equally with sparse data. For instance, made it impossible for RiVaL to run some of the algorithms implemented in the frameworks in our datasets.

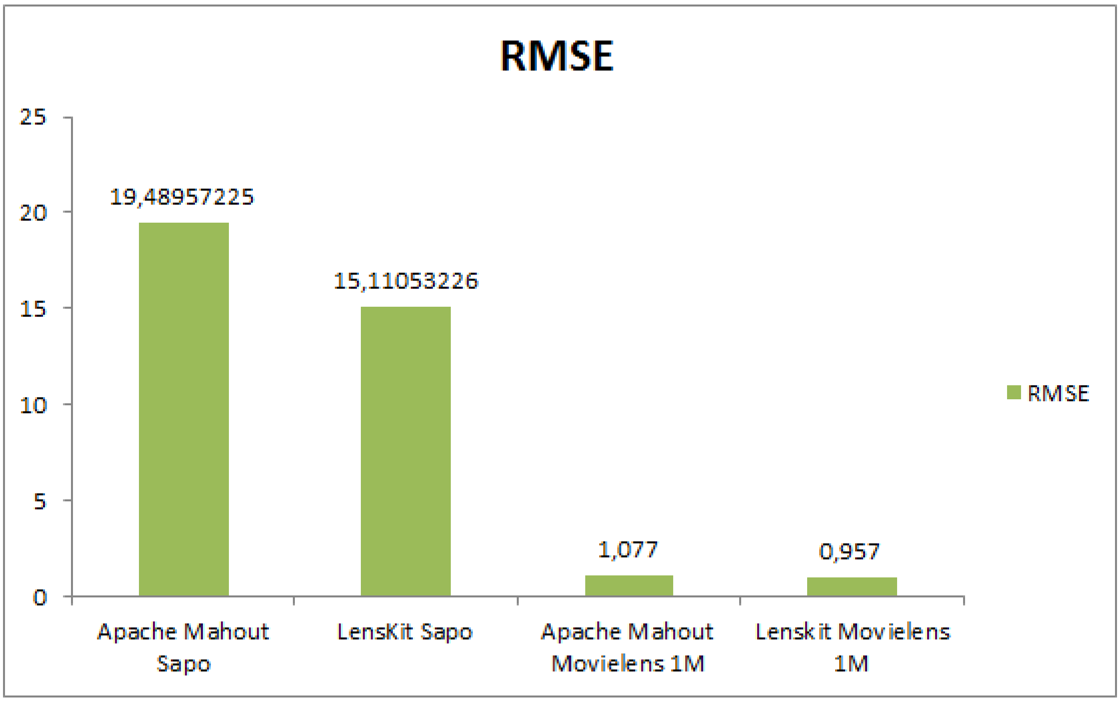


Figure 4.1: RMSE comparison

Observing Figures 4.1 and 4.2, that show the comparison of results between the Sapo data (original dataset) and the Movielens used in their empirical study, only on the metrics they used. These results give further evidence of some of the arguments already presented in this empirical study.

Firstly the data collected from Sapo videos is nowhere near perfect, having a poorer performance in both metrics. Secondly it is noticeable that in their experiments, even obtaining better results, LensKit seems to obtaining worse performance when compared to Apache Mahout.

However, when it comes to recommender systems, given that much of the progress is measured in terms of higher or lower evaluation metrics. It seems intuitive that some form of controlled evaluation, this is for all the different frameworks evaluating their algorithms in the same way, could lead to a bigger understanding of recommender system algorithms qualities in general.

Concluding it is very interesting to see that some of the conclusions demonstrated by (Said & Bellogín, 2014) are present in the results of this empirical study. This controlled evaluation environment where the experiments were run, still gives evidence that even if we are talking of the same algorithms they are implemented very differently leading to very different results.

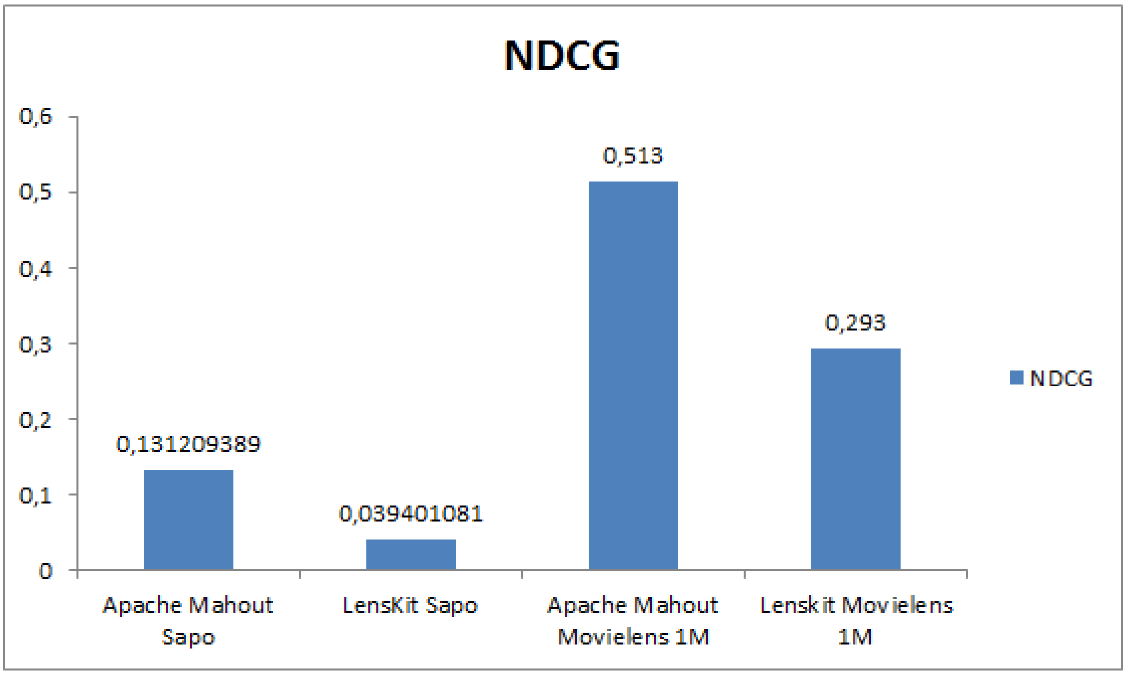


Figure 4.2: NDCG comparison

Chapter 5

Conclusion and Future work

This thesis aim to provide Sapo with an understanding of the potential of recommender systems to increase the interest of the SAPO videos site to their users. This goal could only be accomplished by first completing a various steps. To begin, that data necessary should be collected and prepared to run several experiments with different methods.

Next several recommendation frameworks were surveyed. Finally the data should run through the experimental system, using different algorithms from distinct frameworks, maintaining control over the evaluation process. Furthermore, the results obtained would be analyzed in search of patterns that provide information about the different methods.

This serves the purpose of identifying if RiVaL would behave similarly with very different datasets and if the findings extracted from their results would also be present in the results of this empirical study.

For this experimental procedure that should include more than one of the frameworks surveyed, RiVaL was selected. It allowed the use of two different frameworks (Apache Mahout and LensKit) and it was developed as a four step recommendation process, where only the recommendation process was computed using libraries from the frameworks.

Leaving RiVal responsible for the whole evaluation process. Ensuring that the results obtained from different could compared in the same atmosphere.

Finally all the results obtained were discussed. With the purpose of finding patterns that would lead to conclusions on which method performed best. For a more in depth analysis this conclusions were than compared to the ones obtained by another empirical study using RiVaL.

This serves the purpose of identifying if RiVaL would behave similarly with very different datasets and if the findings extracted from their results would also be present in the results of this empirical study.

In conclusion we have shown, that for this data, there is not a best approach. Although we can argue that for a best performance in precision and NDCG, the data must be filtered, because the original data was so sparse it decreases the performance of the algorithms. To obtain better RMSE results Sapo should start collecting not just the implicit feedback of the user but also the explicit, users should rate the videos they watch.

All this can give a better understanding of the behavior of recommender system not only on Sapo but for the recommendation of videos.

It was also seen that the two main findings of another empirical study were also proved here. Even though the frameworks implement similar algorithms, there exist large differences in the reported recommendation quality. Not only that, but these same algorithms handle the data very differently. For example there is a big discrepancies in the number of the amount of algorithms run by Apache Mahout when compared to LensKit.

5.1 Future Work

In the attempt to prevent evaluation discrepancies in recommender systems and limit its effects, a possible continuation of this work is improve the toolkit used in this study. This improvement could be in the number of frameworks available, increasing the number of evaluation metrics ready for use or different methods of preparing and splitting the data. These improvements would point to and open source toolkit that can formally compare different recommendation strategies.

Another line of work is to build on frameworks such as Rival to create a recommendation as a service. This implies the developing of a web service providing recommendations based on previously developed models.

Whenever a new recommendation is requested the web service would search for best possible method in the previous saved models, with the intention of giving the best recommendation possible for this new request. With the increase of recommendation requests executed, the overall performance of the service should improve.

Chapter 6

Result Tables

6.1 Video Filtering Dataset

Table 6.1: Mahout Collaborative Filtering Videos_2

Algorithm	**NDCG**	**RMSE**	**Precision**	**Nbsize**	**Similarity Class**
GenericItemBased	0,024373779	14,31102345	0,003284535	50	Cosine Similarity
GenericItemBased	0,078865128	20,3932726	0,029898119	50	Pearson's Correlation
GenericItemBased	0,022154043	14,31867362	0,003288288	50	Euclidean Distance
GenericItemBased	0,028347699	14,25899895	0,005469219	50	Tanimoto Coefficient
GenericUserBased	0,110537468	19,96689811	0,025469925	50	Pearson's Correlation
GenericUserBased	0,094541424	16,7855072	0,013830224	50	Cosine
GenericUserBased	0,095096532	17,95699985	0,021002438	50	Euclidean Distance
GenericUserBased	0,128753639	15,75750302	0,027738737	50	Tanimoto Coefficient

Table 6.2: Mahout Matrix Factorization Videos_2

Algorithm	**NDCG**	**RMSE**	**Precision**	**Factorizer**	**Iteractions**	**Factors**
SVD	0,139114095	15,28995369	0,023463174	FunkSVD	50	10
SVD	0,100529247	15,41378189	0,015524168	RatingSGD	50	10
SVD	0,143890208	15,28602183	0,023734747	SVDPlusPlus	50	10
SVD	0,138405389	15,3000333	0,02391786	ALSWR	50	10
SVD	0,145935305	15,30080581	0,025359953	FunkSVD	50	30
SVD	0,142241678	15,30317033	0,024635974	RatingSGD	50	30
SVD	0,13929393	15,30842325	0,023914928	SVDPlusPlus	50	30
SVD	0,146505944	15,30382723	0,024817783	ALSWR	50	30
SVD	0,152594549	15,30273343	0,026082629	FunkSVD	50	50
SVD	0,146374986	15,30805525	0,024637603	RatingSGD	50	50
SVD	0,149111351	15,31050485	0,024548816	SVDPlusPlus	50	50
SVD	0,152339808	15,30786787	0,026082303	ALSWR	50	50

Table 6.3: LensKit Collaborative Filtering Videos_2

Algorithm	NDCG	RMSE	Precison	Nbsize	Similarity Class
UserUserScorer	0,02356569	15,13545763	0,002457455	50	Cosine Similarity
UserUserScorer	0,067074368	15,50943582	0,009481615	50	Pearson's Correlation
ItemItemScorer	0,060945789	18,88321309	0,013909206	50	Pearson's Correlation
ItemItemScorer	0,022359428	15,52834915	0,002075004	50	Cosine Similarity
ItemItemScorer	0,050126349	20,1947561	0,013882443	50	Spearman Rank

Table 6.4: LensKit Matrix Factorization Videos_2

Algorithm	NDCG	RMSE	Precison	Factorizer	Iteractions	Factors
SVD	0,106198859	14,64361562	0,013268893	FunkSVD	50	50
SVD	0,125966414	14,64606926	0,01572935	FunkSVD	50	30
SVD	0,142370261	14,66118998	0,018453427	FunkSVD	50	10

Table 6.5: Mahout Collaborative Filtering Videos_5

Algorithm	NDCG	RMSE	Precison	Nbsize	Similarity
GenericItemBased	0,030571622	15,27794491	0,004178251	50	Cosine Similarity
GenericItemBased	0,083569167	20,41336504	0,028511328	50	Pearson's Correlation
GenericItemBased	0,034548062	15,26724878	0,005370509	50	Euclidean Distance
GenericItemBased	0,039714885	15,20614826	0,007309488	50	Tanimoto Coefficient
GenericUserBased	0,117020069	19,97378396	0,027819549	50	Pearson's Correlation
GenericUserBased	0,089437639	16,33798363	0,014562118	50	Cosine Similarity
GenericUserBased	0,12914195	17,23509267	0,022348185	50	Euclidean Distance
GenericUserBased	0,154975082	15,68009625	0,024762609	50	Tanimoto Coefficient

Table 6.6: Mahout Matrix Factorization Videos_5

Algorithm	NDCG	RMSE	Precison	Factorizer	Iteractions	Factors
SVD	0,164745849	16,44536709	0,025506073	FunkSVD	50	10
SVD	0,10257476	16,60942397	0,017307692	RatingSGD	50	10
SVD	0,181192413	16,45664519	0,026619433	SVDPlusPlus	50	10
SVD	0,168965877	16,46154919	0,026315789	ALSWR	50	10
SVD	0,18204337	16,45512758	0,028036437	FunkSVD	50	30
SVD	0,171516546	16,46576142	0,027226721	RatingSGD	50	30
SVD	0,177533491	16,45726023	0,027327935	SVDPlusPlus	50	30
SVD	0,181399046	16,46627547	0,027732794	ALSWR	50	30
SVD	0,185106877	16,46990705	0,028441296	FunkSVD	50	50
SVD	0,180815888	16,47512167	0,028137652	RatingSGD	50	50
SVD	0,178726204	16,4725251	0,027530364	SVDPlusPlus	50	50
SVD	0,1767486	16,47107668	0,027530364	ALSWR	50	50

Table 6.7: LensKit Collaborative Filtering Videos_5

Algorithm	**NDCG**	**RMSE**	**Precison**	**Nbsize**	**Similarity Class**
UserUserScorer	0,029577844	16,4578677	0,002125797	50	Cosine Similarity
UserUserScorer	0,073694605	16,79103911	0,011639676	50	Pearson's Correlation
ItemItemScorer	0,061390839	18,64718276	0,012604183	50	Pearson's Correlation
ItemItemScorer	0,031343912	16,78951709	0,002631579	50	Cosine Similarity
ItemItemScorer	0,050054272	19,45567622	0,013251333	50	Spearman Rank

Table 6.8: LensKit Matrix Factorization Videos_5

Algorithm	**NDCG**	**RMSE**	**Precison**	**Factorizer**	**Iteractions**	**Factors**
FunkSVDItemScorer	0,143429992	16,14734931	0,016232465	FunkSVD	50	50
FunkSVDItemScorer	0,163623332	16,15156809	0,016332665	FunkSVD	50	30
FunkSVDItemScorer	0,180827424	16,17362477	0,020240481	FunkSVD	50	10

Table 6.9: Mahout Collaborative Filtering Videos_10

Algorithm	**NDCG**	**RMSE**	**Precison**	**Nbsize**	**Similarity Class**
GenericItemBased	0,050710769	17,48616598	0,006304126	50	Cosine Similarity
GenericItemBased	0,104973273	20,90359303	0,035686739	50	Pearson's Correlation
GenericItemBased	0,056739131	17,48875834	0,006645423	50	Euclidean Distance
GenericItemBased	0,058057297	17,44647253	0,011584361	50	Tanimoto Coefficient
GenericUserBased	0,132311436	19,31948413	0,029944673	50	Pearson's Correlation
GenericUserBased	0,100125354	16,35576272	0,016018307	50	Cosine Similarity
GenericUserBased	0,13941313	18,3716735	0,026010425	50	Euclidean Distance
GenericUserBased	0,17473019	16,80059348	0,029938519	50	Tanimoto Coefficient

Table 6.10: Mahout Matrix Factorization Videos_10

Algorithm	**NDCG**	**RMSE**	**Precison**	**Factorizer**	**Iteractions**	**Factors**
SVD	0,178676932	16,33501775	0,029394706	FunkSVD	50	10
SVD	0,12374382	16,51955538	0,018941791	RatingSGD	50	10
SVD	0,172636705	16,33573958	0,028016978	SVDPlusPlus	50	10
SVD	0,175946657	16,33197226	0,029165612	ALSWR	50	10
SVD	0,187307549	16,35567623	0,029050406	FunkSVD	50	30
SVD	0,18274348	16,35595655	0,030083043	RatingSGD	50	30
SVD	0,176409135	16,3607074	0,029509649	SVDPlusPlus	50	30
SVD	0,187414008	16,34572549	0,031345566	ALSWR	50	30
SVD	0,186991781	16,36579752	0,029968364	FunkSVD	50	50
SVD	0,185188464	16,36691373	0,029623273	RatingSGD	50	50
SVD	0,185977673	16,36705516	0,030542023	SVDPlusPlus	50	50
SVD	0,189401091	16,36488966	0,03031161	ALSWR	50	50

Table 6.11: LensKit Collaborative Filtering Videos_10

Algorithm	**NDCG**	**RMSE**	**Precison**	**Nbsize**	**Similarity Class**
UserUserScorer	0,044576719	16,12454266	0,00418192	50	Cosine Similarity
UserUserScorer	0,095944155	16,70987306	0,015731572	50	Pearson's Correlation
ItemItemScorer	0,075635711	19,12380764	0,016083607	50	Pearson's Correlation
ItemItemScorer	0,044475388	16,72543963	0,00482205	50	Cosine Similarity
ItemItemScorer	0,066751236	19,58135247	0,015634526	50	Spearman Rank

Table 6.12: LensKit Matrix Factorization Videos_10

Algorithm	**NDCG**	**RMSE**	**Precison**	**Factorizer**	**Iteractions**	**Factors**
SVD	0,146125499	18,08173109	0,015789474	FunkSVD	50	50
SVD	0,19153816	18,08843161	0,020251716	FunkSVD	50	30
SVD	0,211986692	18,11257357	0,021624714	FunkSVD	50	10

Table 6.13: Mahout Collaborative Filtering Videos_20

Algorithm	NDCG	RMSE	Precison	Nbsize	Similarity Class
GenericItemBased	0,074551788	21,06262875	0,015305118	50	Cosine Similarity
GenericItemBased	0,170948549	23,30713583	0,054919656	50	Pearson's Correlation
GenericItemBased	0,083752263	21,04738407	0,01734252	50	Euclidean Distance
GenericItemBased	0,088757157	21,02964289	0,02379101	50	Tanimoto Coefficient
GenericUserBased	0,142255715	23,36848257	0,03459191	50	Pearson's Correlation
GenericUserBased	0,106434934	20,4771949	0,018681319	50	Cosine Similarity
GenericUserBased	0,1655734	20,39662966	0,03221983	50	Euclidean Distance
GenericUserBased	0,216258387	19,82835844	0,04025511	50	Tanimoto Coefficient

Table 6.14: Mahout Matrix Factorization Videos_20

Algorithm	NDCG	RMSE	Precison	Factorizer	Iteractions	Factors
SVD	0,217211901	21,79068395	0,035714286	FunkSVD	50	10
SVD	0,15472838	22,00651057	0,029258242	RatingSGD	50	10
SVD	0,215072948	21,80710854	0,035851648	SVDPlusPlus	50	10
SVD	0,159317836	21,30589553	0,024725275	ALSWR	50	10
SVD	0,226204456	21,80507715	0,038598901	FunkSVD	50	30
SVD	0,231683496	21,80837399	0,039697802	RatingSGD	50	30
SVD	0,233537981	21,8038943	0,038873626	SVDPlusPlus	50	30
SVD	0,23244908	21,80355167	0,039835165	ALSWR	50	30
SVD	0,226424732	21,82313624	0,039697802	FunkSVD	50	50
SVD	0,235936617	21,80988908	0,040247253	RatingSGD	50	50
SVD	0,230743604	21,82486173	0,03956044	SVDPlusPlus	50	50
SVD	0,23884007	21,81500362	0,03956044	ALSWR	50	50

Table 6.15: LensKit Collaborative Filtering Videos_20

Algorithm	NDCG	RMSE	Precison	Nbsize	Similarity Class
UserUserScorer	0,061279809	21,12354658	0,010834646	50	Cosine Similarity
UserUserScorer	0,1386322	22,22645188	0,027884615	50	Pearson's Correlation
ItemItemScorer	0,121427871	21,40392442	0,02982906	50	Pearson's Correlation
ItemItemScorer	0,062947415	22,21651746	0,010851648	50	Cosine Similarity
ItemItemScorer	0,109285801	22,45877779	0,030644847	50	Spearman Rank

Table 6.16: LensKit Matrix Factorization Videos_20

Algorithm	NDCG	RMSE	Precison	Factorizer	Iteractions	Factors
SVD	0,187862758	21,93054312	0,025686813	FunkSVD	50	50
SVD	0,206010901	21,94227927	0,027335165	FunkSVD	50	30
SVD	0,22816264	21,9719689	0,02760989	FunkSVD	50	10

Table 6.17: Mahout Collaborative Filtering Videos_50

Algorithm	NDCG	RMSE	Precison	Nbsize	Similarity Class
GenericItemBased	0,358331129	35,28577951	0,083333333	50	Cosine Similarity
GenericItemBased	0,36888953	40,7497227	0,097155412	50	Pearson's Correlation
GenericItemBased	0,360577006	35,21463823	0,085460993	50	Euclidean Distance
GenericItemBased	0,369549616	35,19319797	0,085815603	50	Tanimoto Coefficient
GenericUserBased	0,288577022	29,11332497	0,039615385	50	Pearson's Correlation
GenericUserBased	0,187640811	31,94961879	0,030576923	50	Cosine Similarity
GenericUserBased	0,37048421	31,93904193	0,05780294	50	Euclidean Distance
GenericUserBased	0,424521424	31,02240638	0,064784427	50	Tanimoto Coefficient

Table 6.18: Mahout Matrix Factorization Videos_50

Algorithm	NDCG	RMSE	Precison	Factorizer	Iteractions	Factors
SVD	0,278424493	32,13771294	0,046730769	FunkSVD	50	10
SVD	0,384350012	27,97558355	0,058076923	RatingSGD	50	10
SVD	0,395233101	31,59339007	0,054230769	SVDPlusPlus	50	10
SVD	0,368964925	31,60762656	0,050192308	ALSWR	50	10
SVD	0,404318366	31,64875357	0,054423077	FunkSVD	50	30
SVD	0,400419465	31,61428617	0,055384615	RatingSGD	50	30
SVD	0,394127763	31,59202099	0,054230769	SVDPlusPlus	50	30
SVD	0,392493488	31,61519984	0,055384615	ALSWR	50	30
SVD	0,407909062	31,62903851	0,054615385	FunkSVD	50	50
SVD	0,396072036	31,65982429	0,055384615	RatingSGD	50	50
SVD	0,396606889	31,64988851	0,055192308	SVDPlusPlus	50	50
SVD	0,40852989	31,62722753	0,055192308	ALSWR	50	50

Table 6.19: LensKit Collaborative Filtering Videos_50

Algorithm	NDCG	RMSE	Precison	Nbsize	Similarity Class
UserUserScorer	0,242554757	32,58796135	0,037777778	50	Cosine Similarity
UserUserScorer	0,359180282	32,60897257	0,051923077	50	Pearson's Correlation
ItemItemScorer	0,364383438	34,37476138	0,057612118	50	Pearson's Correlation
ItemItemScorer	0,27277698	32,27826861	0,04	50	Cosine Similarity
ItemItemScorer	0,383625085	34,454775	0,061003861	50	Spearman Rank

Table 6.20: LensKit Matrix Factorization Videos_50

Algorithm	NDCG	RMSE	Precison	Factorizer	Iteractions	Factors
SVD	0,299809667	31,74746217	0,037307692	FunkSVD	50	50
SVD	0,331675372	31,76375406	0,04	FunkSVD	50	30
SVD	0,377098696	31,80326517	0,0425	FunkSVD	50	10

6.2 User Filtering Dataset

Table 6.21: Mahout Collaborative Filtering Users_2

Algorithm	**NDCG**	**RMSE**	**Precison**	**Nbsize**	**Similarity Class**
GenericItemBased	0,009803037	14,65824749	0,001289343	50	Cosine Similarity
GenericItemBased	0,078607863	19,82033926	0,031015625	50	Pearson's Correlation
GenericItemBased	0,00998516	14,64887434	0,001546412	50	Euclidean Distance
GenericItemBased	0,010797994	14,61058071	0,001547411	50	Tanimoto Coefficient
GenericUserBased	0,223424513	21,94277062	0,106190625	50	Pearson's Correlation
GenericUserBased	0,081233613	20,39979407	0,015469762	50	Cosine Similarity
GenericUserBased	0,078841399	17,47064147	0,021295835	50	Euclidean Distance
GenericUserBased	0,101641602	15,36736213	0,025745456	50	Tanimoto Coefficient

Table 6.22: Mahout Matrix Factorization Users_2

Algorithm	NDCG	RMSE	Precison	Factorizer	Iteractions	Factors
SVD	0,127078212	14,96230747	0,023870142	FunkSVD	50	10
SVD	0,07307877	15,07345942	0,017493639	RatingSGD	50	10
SVD	0,130454903	14,95401671	0,024824341	SVDPlusPlus	50	10
SVD	0,129527394	14,95752968	0,024055922	ALSWR	50	10
SVD	0,132127144	14,96936999	0,025106263	FunkSVD	50	30
SVD	0,132309348	14,96350316	0,025580471	RatingSGD	50	30
SVD	0,133292996	14,96329429	0,02510554	SVDPlusPlus	50	30
SVD	0,128436636	14,96710015	0,02624841	ALSWR	50	30
SVD	0,135053268	14,9718579	0,024820003	FunkSVD	50	50
SVD	0,076877918	15,08919488	0,01749147	RatingSGD	50	50
SVD	0,135978789	14,97097672	0,025864562	SVDPlusPlus	50	50
SVD	0,134062495	14,97162421	0,026058293	ALSWR	50	50

Table 6.23: LensKit Collaborative Filtering Users_2

Algorithm	NDCG	RMSE	Precison	Nbsize	Similarity Class
UserUserScorer	0,040546254	15,20512295	0,009808727	50	Cosine
UserUserScorer	0,040546876	15,25756853	0,00968797	50	Pearson's Correlation
ItemItemScorer	0,040858312	18,34032527	0,01326397	50	Pearson's Correlation
ItemItemScorer	0,004730485	15,2642729	3,81E-04	50	Cosine
ItemItemScorer	0,037039564	20,95980682	0,014056979	50	Spearman Rank

Table 6.24: LensKit Matrix Factorization Users_2

Algorithm	NDCG	RMSE	Precison	Factorizer	Iteractions	Factors
SVD	0,116105696	14,24463766	0,018196721	FunkSVD	50	10
SVDr	0,105799147	14,23174497	0,017459016	FunkSVD	50	30
SVD	0,091006061	14,23083315	0,014918033	FunkSVD	50	50

Table 6.25: Mahout Collaborative Filtering Users_5

Algorithm	NDCG	RMSE	Precison	Nbsize	Similarity Class
GenericItemBased	0,01024767	14,75202378	0,002135914	50	Cosine Similarity
GenericItemBased	0,070550341	19,75533062	0,037607763	50	Pearson's Correlation
GenericItemBased	0,011220036	14,75621927	0,002286972	50	Euclidean Distance
GenericItemBased	0,013912738	14,70223483	0,002584864	50	Tanimoto Coefficient
GenericUserBased	0,181586409	21,38261642	0,109145833	50	Pearson's Correlation
GenericUserBased	0,074919808	18,64097115	0,02055812	50	Cosine Similarity
GenericUserBased	0,082099245	17,46184593	0,027730527	50	Euclidean Distance
GenericUserBased	0,090690823	15,57553368	0,033622619	50	Tanimoto Coefficient

Table 6.26: Mahout Matrix Factorization Users_5

Algorithm	**NDCG**	**RMSE**	**Precison**	**Factorizer**	**Iteractions**	**Factors**
SVD	0,130845311	13,80003207	0,033237084	FunkSVD	50	10
SVD	0,078460455	13,93147711	0,023289254	RatingSGD	50	10
SVD	0,135202891	13,79395753	0,033780885	SVDPlusPlus	50	10
SVD	0,130727295	13,79617647	0,033511702	ALSWR	50	10
SVD	0,139741527	13,80322532	0,037945076	FunkSVD	50	30
SVD	0,137169948	13,80611993	0,037405623	RatingSGD	50	30
SVD	0,136839081	13,8036755	0,036327802	SVDPlusPlus	50	30
SVD	0,136960092	13,80611529	0,037273748	ALSWR	50	30
SVD	0,141275791	13,81128186	0,038209912	FunkSVD	50	50
SVD	0,079809963	13,93435041	0,023693573	RatingSGD	50	50
SVD	0,138938223	13,80627488	0,038355916	SVDPlusPlus	50	50
SVD	0,143869914	13,80716726	0,039025071	ALSWR	50	50

Table 6.27: LensKit Collaborative Filtering Users_5

Algorithm	**NDCG**	**RMSE**	**Precison**	**Nbsize**	**Similarity Class**
UserUserScorer	0,044041991	14,04594038	0,01360626	50	Cosine Similarity
UserUserScorer	0,041254564	14,04691246	0,012434266	50	Pearson's Correlation
ItemItemScorer	0,03468459	18,43376961	0,019045968	50	Pearson's Correlation
ItemItemScorer	0,002675222	14,08146477	6,72E-04	50	Cosine Similarity
ItemItemScorer	0,035625632	19,08055598	0,018938663	50	Spearman Rank

Table 6.28: LensKit Matrix Factorization Users_5

Algorithm	**NDCG**	**RMSE**	**Precison**	**Factorizer**	**Iteractions**	**Factors**
SVD	0,104839324	14,66006172	0,023366834	FunkSVD	50	10
SVD	0,095480559	14,64421518	0,02160804	FunkSVD	50	30
SVD	0,087526446	14,64014834	0,020728643	FunkSVD	50	50

Table 6.29: Mahout Collaborative Filtering Users_10

Algorithm	**NDCG**	**RMSE**	**Precison**	**Nbsize**	**Similarity Class**
GenericItemBased	0,005117327	14,4991081	0,001064778	50	Cosine Similarity
GenericItemBased	0,068314781	20,26785416	0,044428232	50	Pearson's Correlation
GenericItemBased	0,005844455	14,49641472	0,001273914	50	Euclidean Distance
GenericItemBased	0,010768458	14,46405137	0,004256383	50	Tanimoto Coefficient
GenericUserBased	0,208803511	20,82761776	0,110427632	50	Pearson's Correlation
GenericUserBased	0,070474608	17,91264338	0,025249771	50	Cosine Similarity
GenericUserBased	0,101915243	16,99337596	0,040241404	50	Euclidean Distance
GenericUserBased	0,129597879	15,838004	0,048111989	50	Tanimoto Coefficient

Table 6.30: Mahout Matrix Factorization Users_10

Algorithm	**NDCG**	**RMSE**	**Precison**	**Factorizer**	**Iteractions**	**Factors**
SVD	0,162020624	16,01573918	0,045445194	FunkSVD	50	10
SVD	0,110422598	16,14743493	0,03794108	RatingSGD	50	10
SVD	0,17462771	16,01287955	0,046667215	SVDPlusPlus	50	10
SVD	0,169367497	16,0169059	0,045651745	ALSWR	50	10
SVD	0,174078833	16,01967014	0,047869487	FunkSVD	50	30
SVD	0,175176095	16,0195531	0,047876893	RatingSGD	50	30
SVD	0,176482921	16,02701228	0,047468729	SVDPlusPlus	50	30
SVD	0,179148311	16,02267391	0,048280118	ALSWR	50	30
SVD	0,177299088	16,02622009	0,049484858	FunkSVD	50	50
SVD	0,112419826	16,15680161	0,037530448	RatingSGD	50	50
SVD	0,179915552	16,01988697	0,048892363	SVDPlusPlus	50	50
SVD	0,183213088	16,02516927	0,049494733	ALSWR	50	50

Table 6.31: LensKit Collaborative Filtering Users_10

Algorithm	**NDCG**	**RMSE**	**Precison**	**Nbsize**	**Similarity Class**
UserUserScorer	0,07551937	16,13630422	0,024169684	50	Cosine Similarity
UserUserScorer	0,078989141	16,5657868	0,028254624	50	Pearson’s Correlation
ItemItemScorer	0,04777581	19,38070415	0,028696586	50	Pearson’s Correlation
ItemItemScorer	0,003020593	16,30392451	8,06E-04	50	Cosine Similarity
ItemItemScorer	0,036337546	19,35876778	0,025298614	50	Spearman Rank

Table 6.32: LensKit Matrix Factorization Users_10

Algorithm	**NDCG**	**RMSE**	**Precison**	**Factorizer**	**Iteractions**	**Factors**
SVD	0,155097559	15,69777532	0,041666667	FunkSVD	50	10
SVD	0,144645938	15,67827355	0,03875969	FunkSVD	50	30
SVD	0,133716232	15,67199283	0,036046512	FunkSVD	50	50

Table 6.33: Mahout Collaborative Filtering Users_20

Algorithm	**NDCG**	**RMSE**	**Precison**	**Nbsize**	**Similarity Class**
GenericItemBased	0,004248623	15,05510404	6,80E-04	50	Cosine Similarity
GenericItemBased	0,08318154	21,93072572	0,061069767	50	Pearson’s Correlation
GenericItemBased	0,008821376	15,05228598	0,001710063	50	Euclidean Distance
GenericItemBased	0,006015387	14,99377963	0,001369927	50	Tanimoto Coefficient
GenericUserBased	0,228434804	22,22633584	0,126723058	50	Pearson’s Correlation
GenericUserBased	0,138790661	18,61255903	0,058922093	50	Cosine Similarity
GenericUserBased	0,128149289	17,48000535	0,061284691	50	Euclidean Distance
GenericUserBased	0,144968578	16,6393978	0,072723562	50	Tanimoto Coefficient

Table 6.34: Mahout Matrix Factorization Users_20

Algorithm	NDCG	RMSE	Precison	Factorizer	Iteractions	Factors
SVD	0,18350432	16,8151609	0,069	FunkSVD	50	10
SVD	0,153185175	16,95888928	0,06	RatingSGD	50	10
SVD	0,182907765	16,80214584	0,067333333	SVDPlusPlus	50	10
SVD	0,18654654	16,79150887	0,066666667	ALSWR	50	10
SVD	0,198596939	16,81658433	0,072333333	FunkSVD	50	30
SVD	0,188165105	16,81267913	0,069	RatingSGD	50	30
SVD	0,190979137	16,81638649	0,072	SVDPlusPlus	50	30
SVD	0,194934475	16,82064565	0,071666667	ALSWR	50	30
SVD	0,192352292	16,82911639	0,071666667	FunkSVD	50	50
SVD	0,155286719	16,95690803	0,059333333	RatingSGD	50	50
SVD	0,197594207	16,82161781	0,073333333	SVDPlusPlus	50	50
SVD	0,19481147	16,82890571	0,072	ALSWR	50	50

Table 6.35: LensKit Collaborative Filtering Users_20

Algorithm	NDCG	RMSE	Precison	Nbsize	Similarity Class
UserUserScorer	0,087579059	16,94764622	0,04	50	Cosine Similarity
UserUserScorer	0,089575463	17,01274174	0,041111111	50	Pearson's Correlation
ItemItemScorer	0,070897877	20,99347661	0,0425868	50	Pearson's Correlation
ItemItemScorer	0,001587474	17,1142796	6,67E-04	50	Cosine Similarity
ItemItemScorer	0,058631307	21,50665117	0,034891304	50	Spearman Rank

Table 6.36: LensKit Matrix Factorization Users_20

Algorithm	NDCG	RMSE	Precison	Factorizer	Iteractions	Factors
SVD	0,172347903	16,92138425	0,057467532	FunkSVD	50	10
SVD	0,162569697	16,89617458	0,056493506	FunkSVD	50	30
SVD	0,161548846	16,88599408	0,055844156	FunkSVD	50	50

Table 6.37: Mahout Collaborative Filtering Users_50

Algorithm	NDCG	RMSE	Precison	Nbsize	Similarity Class
GenericItemBased	0,002334645	17,29502695	0,002173913	50	Cosine Similarity
GenericItemBased	0,085602806	26,06114909	0,086862981	50	Pearson's Correlation
GenericItemBased	0,003205005	17,2878666	0,002898551	50	Euclidean Distance
GenericItemBased	0,003655243	17,23348598	0,002898551	50	Tanimoto Coefficient
GenericUserBased	0,202337608	23,51454458	0,16254162	50	Pearson's Correlation
GenericUserBased	0,185347069	19,57740293	0,098550725	50	Cosine Similarity
GenericUserBased	0,16554576	19,45726024	0,09057971	50	Euclidean Distance
GenericUserBased	0,189229462	19,33571127	0,102898551	50	Tanimoto Coefficient

Table 6.38: Mahout Matrix Factorization Users_50

Algorithm	NDCG	RMSE	Precison	Factorizer	Iteractions	Factors
SVD	0,205397807	16,73933938	0,100724638	FunkSVD	50	10
SVD	0,247127205	16,53919649	0,11884058	RatingSGD	50	10
SVD	0,245729217	16,52106384	0,108695652	SVDPlusPlus	50	10
SVD	0,229534317	16,53266774	0,110869565	ALSWR	50	10
SVD	0,223190202	16,56208308	0,10942029	FunkSVD	50	30
SVD	0,23096636	16,54709543	0,113043478	RatingSGD	50	30
SVD	0,236243051	16,55640347	0,120289855	SVDPlusPlus	50	30
SVD	0,22731153	16,56842841	0,113043478	ALSWR	50	30
SVD	0,23681477	16,57847756	0,11884058	FunkSVD	50	50
SVD	0,202299463	16,76002846	0,099275362	RatingSGD	50	50
SVD	0,231357304	16,56795792	0,110869565	SVDPlusPlus	50	50
SVD	0,235745167	16,56781835	0,118115942	ALSWR	50	50

Table 6.39: LensKit Collaborative Filtering Users_50

Algorithm	NDCG	RMSE	Precison	Nbsize	Similarity Class
UserUserScorer	0,156275537	16,89692489	0,071014493	50	Cosine Similarity
UserUserScorer	0,193414021	16,934325	0,076778769	50	Pearson's Correlation
ItemItemScorer	0,058592432	25,47332362	0,067171946	50	Pearson's Correlation
ItemItemScorer	6,80E-04	17,1166718	0,001449275	50	Cosine Similarity
ItemItemScorer	0,045110784	25,46886409	0,057273756	50	Spearman Rank

Table 6.40: LensKit Matrix Factorization Users_50

Algorithm	NDCG	RMSE	Precison	Factorizer	Iteractions	Factors
SVD	0,229070276	16,92849045	0,102898551	FunkSVD	50	10
SVD	0,22999558	16,89607692	0,105072464	FunkSVD	50	30
SVD	0,229344715	16,88245992	0,097826087	FunkSVD	50	50

References

Adomavicius, G., & Tuzhilin, A. (2005). Toward the Next Generation of Recommender Systems : A Survey of the State-of-the-Art and Possible Extensions. *IEEE Trans. on Knowl. and Data Eng.*, *17*(6), 734–749.

Adomavicius, G., & Zhang, J. (2012, November). Stability of Recommendation Algorithms. *ACM Transactions on Information Systems*, *30*(4), 1–31.

Baltrunas, L., Makcinskas, T., & Ricci, F. (2010). Group Recommendations with Rank Aggregation and Collaborative Filtering. In *Acm conference on recommender systems* (pp. 119–126).

Beel, J., Genzmehr, M., Langer, S., Nürnberger, A., & Gipp, B. (2013). A comparative analysis of offline and online evaluations and discussion of research paper recommender system evaluation. *Proceedings of the International Workshop on Reproducibility and Replication in Recommender Systems Evaluation - RepSys '13*, 7–14.

Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013, July). Recommender systems survey. *Knowledge-Based Systems*, *46*, 109–132.

Campos, P. G., Díez, F., & Cantador, I. (2013, February). Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. *User Modeling and User-Adapted Interaction*, *24*(1-2), 67–119.

Cantador, I., Bellogín, A., & Vallet, D. (2010). Content-based recommendation in social tagging systems. In *Acm conference on recommender systems* (pp. 237–240). New York, New York, USA: ACM Press.

Christakou, C., Vrettos, S., & Stafylopatis, A. (2007). A Hybrid Movie Recommender System based on Neural Networks. *International Journal on Artificial Intelligence Tools*, *16*(5), 771–792.

Davidson, J., Livingston, B., Sampath, D., Liebald, B., Liu, J., Nandy, P., … Lambert, M. (2010). The YouTube video recommendation system. *Proceedings of the fourth ACM conference on Recommender systems - RecSys '10*, 293.

Deshpande, M., & Karypis, G. (2004, January). Item-based top-N recommendation algorithms. *ACM Transactions on Information Systems*, *22*(1), 143–177.

Dumitru, H., Gibiec, M., Hariri, N., Cleland-Huang, J., Mobasher, B., Castro-Herrera, C., & Mirakhorli, M. (2011). On-demand feature recommendations derived from mining public product descriptions. *Proceeding of the 33rd international conference on Software engineering - ICSE '11*, 181.

Ekstrand, M. D., Harper, F. M., Willemsen, M. C., & Konstan, J. A. (2014). User perception of differences in recommender algorithms. In *Acm conference on recommender systems* (pp. 161–168). New York, New York, USA: ACM Press.

Gunawardana, A., & Shani, G. (2009). A Survey of Accuracy Evaluation Metrics of Recommendation Tasks. *The Journal of Machine Learning Research*, *10*, 2935–2962. doi: 10.1145/1577069.1755883

Herlocker, J. L., Konstan, J. a., Terveen, L. G., & Riedl, J. T. (2004, January). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, *22*(1), 5–53.

Jeong, M. K. (2010, September). A Hybrid Recommendation Method with Reduced Data for Large-Scale Application. *IEEE Transactions on Systems, Man, and Cybernetics*, *40*(5), 557–566. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5456227 doi: 10.1109/TSMCC.2010.2046036

Jiang, Y., Liu, J., Tang, M., & Liu, X. (2011, July). An Effective Web Service Recommendation Method Based on Personalized Collaborative Filtering. In *Ieee international conference on web services* (pp. 211–218). Ieee.

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, 42–49.

Lü, L., Medo, M., Yeung, C. H., Zhang, Y.-C., Zhang, Z.-K., & Zhou, T. (2012, October). Recommender systems. *Physics Reports*, *519*(1), 1–49.

Melville, P., Mooney, R., & Nagarajan, R. (2002). Content-boosted collaborative filtering for improved recommendations. *AAAI/IAAI*, 187–192.

Pazzani, M., & Billsus, D. (2007). Content-Based Recommendation Systems. *The Adaptive Web*, *4321*, 325–341.

Said, A., & Bellogín, A. (2014). Comparative recommender system evaluation: benchmarking recommendation frameworks. *Proceedings of the 8th ACM Conference on . . . .*

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2000). Analysis of recommendation algorithms for e-commerce. *Proceedings of the 2nd ACM conference on Electronic commerce*.

Saveski, M., & Mantrach, A. (2014). Item cold-start recommendations: learning local collective embeddings. *Proceedings of the 8th ACM Conference on Recommender systems*.

Sparks, E., & Talwalkar, A. (2013, October). MLI: An API for distributed machine learning. *Data Mining (ICDM), ...* Retrieved from http://arxiv.org/abs/1310.5426http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6729619

Vargas, S., & Castells, P. (2011). Rank and relevance in novelty and diversity metrics for recommender systems. In *Acm conference on recommender systems* (pp. 109–116). New York, New York, USA: ACM Press.

Yang, X., Guo, Y., Liu, Y., & Steck, H. (2014, March). A survey of collaborative filtering based social recommender systems. *Computer Communications*, *41*, 1–10.

References

Zanardi, V., & Capra, L. (2008, October). Social ranking: uncovering relevant content using tag-based recommender systems. In *Proceedings of the 2008 acm conference on recommender systems - recsys '08* (p. 51). New York, New York, USA: ACM Press.