Improving Aggregate Diversity in Recommender Systems

A Project Report

submitted by

AISHWARYA P

*in partial fulfilment of the requirements
for the award of the degree of*

BACHELOR OF TECHNOLOGY

under the guidance of
Dr. B. Ravindran

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY, MADRAS.**

May 2015

THESIS CERTIFICATE

This is to certify that the thesis entitled **Improving Aggregate Diversity in Recommender Systems**, submitted by **Aishwarya P**, to the Indian Institute of Technology, Madras, for the award of the degree of **Bachelor of Technology**, is a bona fide record of the research work carried out by her under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

B. Ravindran
Research Guide
Associate Professor
Dept. of Computer Science and Engineering
IIT-Madras, 600 036

Place: Chennai

Date:

ACKNOWLEDGEMENTS

I would like to thank the Indian Institute of Technology, Madras for giving me the opportunity to work on a project and my guide Dr B. Ravindran, for allowing me to work with him and his invaluable help throughout the project, right from the identification of my project topic, to helping mould solution approaches and identifying new directions to work in when a line of thought had to be abandoned. I would also like to thank Dr Umang for suggesting a line of thought that led to two of our major solution approaches. I would also like to thank Mr. Chelliah from Flipkart for his interest in my project. My gratitude also goes to my classmate Vaishnavh for helping me many times during the course of the project, from identifying mistakes in some of my proofs, to suggesting ideas for others and providing motivation when I needed it. Thanks are also due to many others in RISE lab for their valuable suggestions and questions at various times. Finally, I would like to thank my parents and my friends, particularly Aarati, Dhivya, Spandana and Kartik for their invaluable support especially when I was in difficult situations.

ABSTRACT

In recent years, research in recommender systems has turned away from the accurate prediction of ratings, to objectives that focus on the properties of the lists of recommended items. One of these is aggregate diversity - a measure that compares the relative number of times different items are recommended across users in the system. This is a relatively unexplored objective and prior works primarily use item coverage as a metric to evaluate aggregate diversity. In this project, we propose a new metric - intersection distance to evaluate the aggregate diversity of a recommender system and demonstrate its advantages over other possible metrics such as coverage or entropy. We also propose a range of techniques to generate recommendation lists for users that optimize intersection distance, given ratings from a standard recommender system, including simple item-selection heuristics, mathematical re-ranking methods and a gradient-based solution.

TABLE OF CONTENTS

LIST OF TABLES

CHAPTER 1

Introduction

1.1 Motivation

Recommender systems are a class of web applications that aim to predict user responses to actions. They have become fairly ubiquitous today with the advent of online shopping sites such as Amazon and Flipkart. Such systems are also popularly used to recommend content, such as movies, videos, music and news. The most common model used in recommender systems involves the estimation of a utility matrix - there are two classes of entities, generally referred to as users and items, and the system estimates a utility for a user-item pair. In most cases, the utility corresponds to a rating on a fixed scale and is learned from past ratings of users for items [31].

Traditionally, recommender systems have focussed only on improving the accuracy of predicted ratings. This was particularly influenced by the fact that one of the largest spurts in recommender systems research was caused by the Netflix Prize - a competition that required participants to design a recommender system for predicting ratings of users for movies, based on data from Netflix - an online DVD rental and video streaming service. The goal of the task was to solely minimize the root mean squared error between the predicted and known ratings.

However, studies such as those by Mc Nee et al [27] show that this approach based solely on ratings is not sufficient to make recommender systems useful or satisfying to customers. For example, a travel recommendation system that only ranks places you have already visited or a movie recommendation system that only displays the most popular blockbusters is not likely to be very useful to a customer.

To address this issue, researchers have expanded their focus to optimize other objectives in conjunction with rating accuracy. These typically measure the goodness of the list of items recommended to a user. Some commonly studied objectives in this regard include the diversity of the list, which examines some measure of dissimilarity among recommended items, novelty, which tries to measure how different is an item with respect to what the user already knows and serendipity, which measures the system's ability to recommend to the user items that he/she would not have been able to find in its absence. [8, 22, 43]

A commonly studied direction is the user's perspective of diversity which aims at increasing the dissimilarity between items in the recommendation list of a user. [22, 43]. This view is based on the intuition that users typically have a number of preferences and would prefer items satisfying a mix of these. Variety prevents people from getting bored with the class of items typically recommended to them, causing them to use the system longer. While the vast majority of the literature has focused on the user's perspective of diversity, the equally important notion of aggregate diversity has not received much attention. Aggregate diversity refers to the system's perspective of diversity, measuring the relative number of times different items get recommended [1].

Consider the recommendation system for an online store, which recommends products to users. It is well known that the sales of products typically show a long tail behaviour, as in figure 1.1.

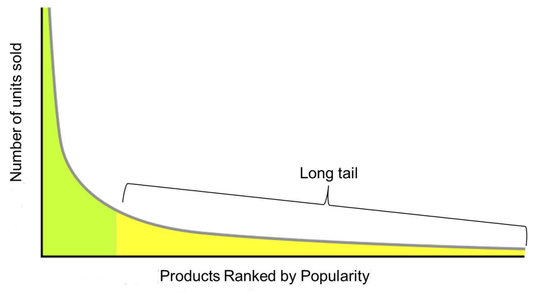

Figure 1.1: Long tail distribution seen in online marketplaces [a]

[a]Source: `http://organicmedialab.com/2013/01/30/3-dimensions-of-smart-economy/`

The owners of the store would like to see their niche products bought by a reasonable number of customers. By increasing the fraction of times such items get recommended, the system is more likely to be able to help users to find items they would otherwise not have come across, increasing user satisfaction and ensuring sales of more available products. This could potentially increase both users' interest in the system and the percentage of conversions from browse to purchase.

Alternately, consider the requirements of the owner of a movie rental service. Here, the seller has a certain number of DVDs and would like, at any point of time, to have as many of them rented out as possible. If a small number of very popular movies are being demanded by a lot of users, many of them are unlikely to be able to rent the movie they desire and hence may choose to switch to another service. However, if the less popular DVDs are also recommended to users, there is a higher chance that some users would choose these instead and hence reduce the competition for the more popular DVDs. This would also, as in the product recommendation case, be likely to be better able to cater to the desires of users with niche tastes.

Consider a different domain - the recommendation of papers to be reviewed, to experts. In this case, having a high aggregate diversity would imply that all papers get recommended roughly equal number of times. This would ensure that papers get reviewed in a load-balanced manner. Here, to ensure that a paper gets a sufficient number of reviews, it may even be desirable to recommend it to experts who are not perfectly aligned with the topic discussed in the paper.

One of the early works to demonstrate the importance of aggregate diversity is that of Fleder et al.[18], that examined the effects of generalized recommendation models on aggregate diversity (there referred to as sales diversity), as measured by the Gini Index. Zhang et al. [39] also use a two phase model of a recommendation system, that separates the tasks of rating prediction and formation of recommendation lists, to demonstrate the usefulness of randomized methods to improve the recommendation diversity.

A trivial way to increase the aggregate diversity of a recommender system is to provide random recommendations. However, such a system is unlikely to provide recommendations that are relevant for users. Aggregate diversity needs to be optimized in conjunction with a suitable measure of relevance to ensure that the resultant recommendation system is useful. Some works that examine this problem include those by Adomavicius et al [1, 2], that directly optimize the coverage of a recommender system, that is, the total number of items recommended at least once.

1.2 Contributions of this project

Due to its ease of computation and natural links with the idea of aggregate diversity, coverage has been in use as a metric for aggregate diversity. Some alternatives would be probability-based measures such as entropy or Gini impurity. In this work, we show that these measures do not satisfy some basic properties that we would expect to be satisfied by a metric for aggregate diversity. The primary contributions of this work are -

- A metric for aggregate diversity based on the histogram intersection distance [12], which is a special case of the Earth Mover's distance [33]. This metric does not suffer from the drawbacks faced by probability-based measures.
- Algorithms for optimizing aggregate diversity based on the intersection distance. We empirically demonstrate that the algorithms proposed outperforms standard baselines for the task.

To the best of our knowledge, apart from Adomavicius et al. [2], this is the only work to optimize aggregate diversity directly. Further, this is the first method to optimize intersection distance.

1.3 Outline

In the next chapter, we present a short survey of works in literature relevant to this project. Following this in chapter 3 is a justification of the need for a new metric for aggregate diversity and a discussion of the properties that make intersection distance suitable for the task. Chapter 4 introduces some first-cut approaches to improving the intersection distance of recommender systems. In chapter 5, an alternate direction that was explored, that attempted to use ideas from uctions to solve this problem, are outlined. Following this, in chapter 6, we describe two improtant solution approaches that cast the problem of optimizing intersection distance as a min cost flow problem. The solution approaches in chapters 4-6 are two phase methods. In the first phase, they make use of a baseline recommender to obtain predicted ratings for user-item pairs. The main method involves ordering items appropriately into recommendation lists for users of high aggregate diversity without compromising on relevance. However, in chapter 7, we suggest an alternate approach - optimizing the parameters of a recommender system directly so that the lists obtained from it have high relevance and diversity. Finally chapter 8 outlines our conclusions and some directions for future work.

CHAPTER 2

Literature Review

2.1 Diversity in Data Mining Tasks

Diversity is a requirement of many tasks in data mining. A class of applications in which there has been considerable emphasis on diversity is web search. These tasks take a query as input from the user and return a list of items (documents, links, products) relevant to the query. Typically this list is sorted in decreasing order of relevance. The need for diversity of search results is primarily because the intent behind a query is often ambiguous. A diverse set of results is more likely to provide at least one item the user actually intended to retrieve via the query. Typically, users need only one or two relevant results. So it is desirable to produce results that are relevant for different interpretations of the query. Here, increase of diversity can be also viewed as redundancy penalization [3, 14].

In the thesis by Sandoval [36], an attempt is made to extend techniques and metrics that enhance diversity in web search to recommendations. Specifically, they use variants of MMR [10] and IA-Select [3] to increase diversity. In web search, diversity is typically associated with subtopics, categories or intents. To have an equivalent concept in recommendation systems, they define an *Aspect space*, which can contain information about item features, additional contextual information or make use of latent features obtained by matrix factorization of the utility matrix. This formulation is found to result in good performance according to a known baseline metric - expected intra-list similarity [43]. However, this interpretation of diversity is from a user's perspective. From the system's perspective, as it is not a ranked list being evaluated, the generalization of web search measures is not as straightforward.

Another notion of diversity can be seen in search tasks on graphs. Such tasks typically have to identify nodes of high importance, as indicated by the network structure. The most well-known such measure of is PageRank. However, some tasks require the identification of nodes that are sufficiently important but also spread out in the graph. Here, the notion of diversity is used to indicate that the returned nodes should not be neighbours and in fact should be reasonably well-separated from each other in the graph [28, 42]. However, this is not easily extendable to recommendation systems.

2.2 Diversity and novelty in Recommendation Systems

Diversity in recommender systems has attracted a significant amount of interest in recent years. The perspective of diversity considered in a majority of these works is the user's perspective, which aims at increasing the dissimilarity between items in the recommendation list of a user. This is done in a number of ways, from re-ranking strategies to minimizing a weighted combination of relevance and diversity metrics [22, 40] to extending techniques used in the diversification of web search results to recommender systems [11, 38].

A survey by Nguyen et al. [30] examines diversity in content recommended to a user over time to check for the presence of the filter bubble effect in MovieLens data. The filter bubble effect refers to the potential of online personalization to isolate people from a diversity of viewpoints (or content). The authors wished to check whether the recommendations received and content consumed became more narrow with time. Movies are represented by a tag genome - a vector where each dimension corresponds to a user-defined tag and the feature value of a movie along that dimension is the relevance of the tag to the movie. Content diversity is then measured as the average pairwise distance between movies in a list. Also, user satisfaction is measured using average rating. They assume that a recommendation is responsible for a rating if it is between 3 hours and 3 months before the rating. According to this model, they divide people into those who follow recommendations and those who do not. They observe a drop in recommendation diversity and correspondingly, that of consumed content accompanied by a drop in average rating.

Another persepective on diversity can be seen in the work of Lathia et al. [25]. This checks for the presence of temporal diversity in recommendations made by collaborative filtering, measured by the overlap of recommendations made using ratings up to a certain time-stamp, for many consecutive windows. It was observed that kNN-based recommender systems had the highest temporal diversity. Another unsurprising observation was that a user was more likely to observe diverse results if he/she returned to the system after a long gap. The authors suggest alternating between recommendation algorithms for consecutive time windows or random re-ranking as solutions to this issue.

In the work of Kawame et al. [24], an attempt is made to improve the serendipity of the system, that is, they try to increase the number of items recommended that the user would struggle to find without the aid of the system. This is done by identifying and following *innovators* - users who tend to be the first to try out a new item. They also identify novel items by estimating the probability of buying one item after buying another. It is found that these techniques also improve the diversity of the system as measured by Gini coefficient, item and user coverage.

2.3 Aggregate Diversity in Recommendation Systems

A compelling justification for the importance of examining aggregate diversity as a metric for recommendation systems can be seen in [18]. They use a ball-in-urn model to simulate the effects of some common recommendation paradigms and prove that these typically result in highly unbalanced sales - as measured by the Gini index. The model is extended to a full fledged user and item model which studied by simulation. It was seen that even on this model, sales showed a very poor Gini index.

Another work that demonstrates the importance of improving aggregate diversity is that of Zhang et al. [39]. Diversity here is measured in terms of the concentration index - a measure similar to the Gini index. They model recommendation systems as two phase systems. The first phase computes a similarity value between an item and user (utility/rating) and the second phase is a suitable probabilistic model that determines whether or not to recommend an item, given its similarity value. They suggest randomization methods and increasing the number of items recommended to a user as solutions to improve diversity.

A first-cut attempt at improving aggregate diversity can be seen in Adomavicius et al. [1]. Here, the top-M items in a user's list are suitably re-ranked so that the top-N, ($N < M$) are sufficiently diverse across users. Relevance of the ratings is maintained by placing a threshold on the predicted rating for the item by the user. The suggested methods use a decrease in average rating, number of rates, awareness, or relative variants of these indicators to determine that the item is less well-known and hence, diverse. Diversity was measured in terms of the number of items and the number of *long tail* items recommended. In [2], a more formal solution to the problem was proposed, making use of a max flow problem to provably maximize item coverage.

2.4 Diversity Metrics in Literature

The choice of diversity metric depends strongly on the application being considered. Although there have been attempts to derive a uniform class of metrics, many are still in use. A detailed assessment of the same is present in the thesis of Sandoval [36]. They can be broadly classified as -

- Those that assume that the topic(s) a query refers to can be divided into dsitinct subtopics and aim to maximize the number of subtopics that get listed in the top results. These include number of subtopics, subtopic precision, subtopic recall and weighted subtopic precision

- Metrics that penalize redundancy in search results, such as α-nDCG.
- Intent-Aware metrics - These aim to generalize more standard search metrics to cover the possible intents of a query.

Text summarization is also seen as a common platform for evaluating the diversity of search methods. The technique is used in an information-retrieval based text summarization system and the quality of the resultant summary is assumed to be better if the search results are more diverse [10, 28, 42]. Variants of coverage are also popular as a measure of diversity [28].

Diversity can also be measured using measures of inequality found in economics. These are outlined in the technical report by Travis Hale [20]. Some basic measures include the range of values, range ratio and the McLoone's index. These do not capture much information about the distribution of values. The coefficient of variation calculates the "peakedness" of a distribution but since it is unbounded, it is practically difficult to determine what is a "good" value when using this metric. Some other measures such as the Gini index and the Theil's T statistic are similar to search diversity metrics. It is also possible to assign different weights to different parts of the distribution using measures such as the Atkinson's index and the Generalized Entropy Measure. However, these are cumbersome to compute and optimize. The choice of parameters becomes an additional problem.

2.5 Earth Mover's Distance

Earth Mover's Distance (EMD) was first introduced as a metric by Rubner et al. in [33]. EMD is a distance measure between two histograms/distributions that attempts to calculate the minimal amount of work that must be performed to transform one distribution into the other by moving *distribution mass* around. It is defined using the solution to the following transportation problem - say we have a ground distance measure c_{ij} between the i^{th} component of the first histogram x and the j^{th} component of the second histogram y. Find the matrix F^* of flows that is the optimal solution to -

$$\begin{aligned}
&\min_F \sum_i \sum_j c_{ij} f_{ij} \\
&\sum_j f_{ij} = x_i \, \forall \, i \\
&\sum_i f_{ij} = y_j \, \forall \, j \\
&f_{ij} \geq 0 \quad \forall \, i, \, \forall \, j
\end{aligned}$$

Then,

$$EMD(x,y) = \frac{\sum_i \sum_j c_{ij} f^*_{ij}}{\sum_i \sum_j f^*_{ij}} = \frac{\sum_i \sum_j c_{ij} f^*_{ij}}{\sum_i y_i} \tag{2.1}$$

EMD is a popular metric for image processing applications. Since it has a relatively generic formulation, it can be used to examine many vector transformations. For example, in Cohen et al. [15], they attempt to compute a transformation to a distribution that minimizes its EMD to another, using an EM-like algorithm. This is motivated by problems in image matching, where external factors that prevent two images of the same object from being exact matches of each other transform the features of the image in a fixed manner.

Cha et al. [13] consider a special case of EMD, called *Minimum Difference of Pairwise Assignments*, motivated by the fact that vector-based distance measures such as L_p norms and probabilistic measures like KL-divergence, Bhattacharya distance and Matusita distance only examine the overlapping portions of the histograms. EMD can examine non-overlapping portions of distributions but it is generic and requires solving a transportation problem. For MDPA, closed form expressions can be obtained that can be computed in polynomial time. These expressions are derived as follows. Each histogram is viewed as a multiset of measurements. If the bin height is h for a bin of value v, the multiset will have h instances of value v and so on. Elements from the two sets are then grouped in pairs such that the sum of pairwise distances is minimized. This minimum sum of pairwise distances is the MDPA. MDPA is metric and there exists a method of normalizing MDPA for histograms not of the same length. The authors also propose polynomial-time algorithms for calculating MDPA for different types of histograms. The also show that for nominal histograms, the MDPA is the histogram intersection distance [12], which is equal to half the l_1 distance between the two histograms.

2.6 Parameterized Recommendation Systems

A number of recommendation systems assume that ratings are functions of some suitable parameters, thus reducing the task of the system to identify that set of parameters which best fit the known ratings. These methods use different functions - deterministic or probabilistic, attempt to minimize different error terms and use different techniques to optimize parameters. A commonly used class of models is the class of matrix factorization-based models. In these models, the rating matrix R of dimension $M \times N$ is assumed to be equal to the product of two matrices - a user factor matrix U of dimension $M \times K$ and an item factor matrix V of dimension $N \times K$, that is, $R = U^T V$. The singular value decomposition

(SVD) gives the best such K-rank approximation to R in terms of squared error. However, due to the presence of missing entries in R, the matrices U and V have to be learnt by techniques such as stochastic gradient descent (SGD) [6].

However, methods such as SGD are prone to overfitting. Generative models, which assume a probabilistic model for ratings, attempt to overcome this defect. One of the simplest such methods is Probabilistic Matrix Factorization [34]. A rating is assumed to be normally distributed around the value obtained using standard matrix factorization. The parameters are learnt my maximizing the log-likelihood of the known ratings. This was then extended in [35] by adding Gaussian priors to the user and item factors, whose parameters then become the ones to be learnt. Further, instead of a standard variational method, this work makes use of Gibbs sampling to optimize parameters.

An example of how parameterization of a recommendation system has been exploited to optimize ranking is the work on Bayesian Personalized Ranking by Rendle et al. [32]. They try to address the fact that standard recommendation techniques such as Matrix Factorization or Adaptive kNN are optimized for the task of rating prediction, not item ranking. They aim to learn a total ranking $>_u$ over all items for each user u. It is assumed that all observed items are preferred over all non-observed items to obtain a posterior probability of the system parameters to maximize. That is, if Θ is the set of parameters of the underlying recommendation model, then the optimal ranking is the one that maximizes $Pr(\Theta \mid >_u)$. This criterion is optimized by stochastic gradient descent on the parameters Θ.

2.7 Auctions

During the course of this project, a couple of attempts were made to model the problem of aggregate diversity in recommendation systems as an auction. This would enable the use of results from game theory to provide guarantees.

In the simplest auction model, there are a number of bidders and a single good. Two broad classes of auctions are efficient and optimal auctions. Efficient auctions aim to sell the good to the bidder with maximum valuation for it whereas optimal auctions attempt to sell the good in such a manner that the seller's revenue gets maximized [37].

A multi-unit auction differs from an ordinary auction only in that instead of one good to be sold, there are k identical units of a good to be sold. Bidders are allowed to bid for any number of units up to k. When all bidders demand only a single unit, a generalized second price auction is efficient. Some other popular multi-unit auctions are the sequential auction and the random sampling optimal single price auction [37].

In the case of combinatorial auctions, there are M different bidders and N different goods. Any bidder can bid for any combination of goods. The Winner Determination Problem (WDP) - identifying the bidders to which each good must be sold to result in efficiency, in a combinatorial auction, is reducible to the Set Packing problem and is hence NP-complete. It cannot even be approximated uniformly. However, some special combinatorial auctions have a WDP that reduces to a linear program that can be solved in polynomial time [37].

One such tractable subclass of combinatorial auctions is multi-item auctions. Here, there is a constraint that every bidder can receive at most one item. Thus, bidders place bids only on individual items. The work of Demange et al. [16] designs a mechanism to find an equilibrium price vector - a vector of prices for each item such that every bidder can either be assigned some item which gives them maximum positive surplus or does not have any item which results in positive surplus and all unsold items are priced at their reserve price. The algorithm in fact finds the minimal such vector, that is no price can be decreased without loss of equilibrium. A computationally more efficient approximate solution is also provided, which is additive $B\delta$-approximate where B is the maximum number of bidders and δ is a parameter.

There are many works that provide approximate solutions for different classes of combinatorial auctions. One such work is that of Bartal et al. [5]. This deals with multi-unit combinatorial auctions - auctions where there are n different goods and k_i units for each good i. The paper examines a special case of such auctions where for each item, there are k units and a bidder either wants 0 units or $x \in [\theta, \Theta]$ units. This covers the case where a bidder wants at most one unit of each item. They show that approximating the WDP to within a factor of $\mathcal{O}\left(n^{\frac{1-\epsilon}{k+1}}\right)$ is NP-hard by a reduction from the Maximal Independent Set problem. They also propose an approximate algorithm with a good approximation ratio of $\mathcal{O}\left(\frac{1}{\Theta}\left(\frac{n}{\theta}\right)^{\frac{\Theta}{1-2\Theta}}\right)$, which simplifies to $\mathcal{O}\left(kn^{\frac{1}{k-2}}\right)$ in the case of k-duplicates (exactly k units of each item).

Another is the work of [26] which deals with a branch and bound solution for multi-unit combinatorial auctions. The algorithm, CAMUS, aims to find the solution for an optimal auction - one that maximizes revenue for the seller. Being a branch and bound solution, it is exponential in the worst case but was found to work well experimentally.

In auctions, it is often desirable to introduce budgets for bidders as this better models the real-life scenario. One such attempt is by Fiat et al. [17]. Their auction setup is as follows - there are n distinct items, each bidder a is interested in a set of items S_a, has the same valuation v_a for every item in this set and a budget b_a. In this setting, they find a Pareto optimal allocation of items, with associated prices. The solution is Pareto optimal in the sense that there is no other allocation and price combination for which all bidders have at least the same surplus (total valuation - total price) and the seller gets at least the same revenue and at least one bidder's surplus or the seller's revenue strictly improves.

2.8 Order Statistics of Probability Distributions

Given random variables $X_1, X_2, \ldots X_n$, the k-th order statistic is the k-th smallest of these values. There are results for calculating the order statistics of IID random variables or non-identical uniformly distributed variables. However, in parameterized recommendation systems, we typically deal with non-identical Gaussian random variables. Some works that deal with such variables include that of Nadarajah et al. [29] which gives a formula for the PDF of the max and min of 2 normal random variables (neither independent nor identical). Bromiley et al. [7] demonstrate how to obtain the PDF of a product of non-identical independent random variables and perform convolutions on them. Other works are more generic, such as those of Cao et al. [9] and Balakrishnan et al. [4]. The former derives a recurrence relation that can be used to obtain successive order statistics of independent, non-identical random variables and the latter proves some identities followed by distributions of order statistics of non-independent, non-identical random variables. However, these are difficult to adapt to some distributions. An approximate solution is found in the work of Janjoom et al. [23]. They approximate the Gamma and Normal random variables using the Burr type XII distribution and hence derive expression for single moments of their order statistics.

CHAPTER 3

A Metric for Aggregate Diversity

The goal of aggregate diversity is to measure the extent to which different items get recommended by a system. A metric to evaluate aggregate diversity must consider

1. A system that recommends more items to be more diverse than one that recommends fewer items.
2. A system that recommends most items only to one or two users to be less diverse than one which recommends many items to a large fraction of users

The first condition is a natural requirement associated with the notion of aggregate diversity. To understand the necessity of the second condition, consider the viewpoint of an online retailer. If a product is recommended to only one user, it is quite possible that he/she may not buy it. Recommending it to more users increases the probability that some user buys the product. Thus it is necessary that a recommender system achieves a more equitable distribution over the number of recommendations of different products.

3.1 Need for a Different Metric for Aggregate Diversity

Coverage, defined as the total number of items recommended across all users, has been used in literature as a measure of aggregate diversity. However, consider the following example - there are two systems each of which have an inventory of 100 items. The first recommends one item to all users and the remaining 99 to only one user each. The second system recommends each item to 5 users. Coverage will not be able to differentiate between the two systems but the second system is preferable according to the second requirement discussed for aggregate diversity. Thus although coverage is necessary for high diversity, it is not sufficient.

A possible solution to this drawback is to use a metric like Gini impurity of the normalized vector of the number of times each item gets recommended, henceforth referred to as the normalized count vector. An entry corresponding to an item in this vector can be interpreted as the probability that it gets recommended to some user. The Gini impurity for a probability distribution $\mathbf{p}$ is given by [21] -

$$Gini(\mathbf{p}) = 1 - \sum_i p_i^2 \tag{3.1}$$

A higher Gini impurity score is indicative of increased diversity. However, a simple example can be used to show that a system of higher coverage need not receive a higher score. Consider the case when there are five items and two systems give the following normalized count vectors - $[0.5, 0.5, 0, 0, 0]$ and $[0.75, 0.24, 0.01, 0, 0]$, whose respective Gini impurity scores are 0.5 and 0.3798. The first system is declared as more diverse, despite the second system having a higher coverage of three items, as opposed to the two items recommended by the first system.

Information theoretic metrics such as the entropy of the normalized count vector also appear to be an intuitive choice of metric for aggregate diversity. A high entropy would indicate that the vector is close to the uniform distribution and hence has high aggregate diversity. The entropy H of a probability vector $\mathbf{p}$ is given by

$$H(\mathbf{p}) = -\sum_i p_i \log_2 p_i$$

However, for the same example, the entropy of the first system is 1 but that of the second is a lower value of 0.87, even though it has a higher coverage. Thus neither Gini index nor entropy satisfy the required criteria.

Another possible probability-based metric for aggregate diversity is the KL divergence of the normalized count vector with respect to the uniform distribution, given by,

$$KLD(\mathbf{p}||\mathbf{q}) = \sum_i p_i \ln \frac{p_i}{q_i}$$

$KLD(\mathbf{p}||\mathbf{q})$ is defined only if $q_i = 0 \Rightarrow p_i = 0 \; \forall \; i$. As recommendation systems rarely have full coverage, it is necessary to use the uniform distribution as q. Then if n is the number of items available,

$$\begin{aligned} KLD(\mathbf{p}||\mathbf{q}) &= \sum_i p_i \; \ln\left(\frac{p_i}{q_i}\right) = \sum_i p_i \; \ln\left(\frac{p_i}{1/n}\right) \\ &= ln(n)\sum_i p_i + ln(2)\left(\sum_i p_i \log_2 p_i\right) \\ &= ln(n) - ln(2)H(\mathbf{p}) \end{aligned}$$

Thus, if a distribution has higher entropy, it will also have lower KL divergence to the uniform distribution. However, the earlier example is a case where a system of lower entropy is to be identified as more diverse. This cannot be done using KL divergence.

3.2 Earth Mover's Distance and Histogram Intersection Distance

The Earth Mover's Distance (EMD) is a distance measure between two histograms/distributions that attempts to calculate the minimal amount of work that must be performed to transform one distribution into the other by moving *distribution mass* around [33]. It is defined using the solution to the following transportation problem. Given a ground distance measure c_{ij} between the i^{th} component of the first histogram x and the j^{th} component of the second histogram y. Let F^* be the optimal solution to the following optimization problem in the matrix F, with elements f_{ij}

$$\begin{aligned}
&\min_F \sum_i \sum_j c_{ij} f_{ij} \\
&\sum_j f_{ij} = x_i \; \forall \, i \\
&\sum_i f_{ij} = y_j \; \forall \, j \\
&f_{ij} \geq 0 \quad \forall \, i, \, j
\end{aligned}$$

Then,

$$EMD(x,y) = \frac{\sum_i \sum_j c_{ij} f^*_{ij}}{\sum_i \sum_j f^*_{ij}} = \frac{\sum_i \sum_j c_{ij} f^*_{ij}}{\sum_i y_i} \tag{3.2}$$

Some advantages of EMD as a distance measure in general are,

- The optimization problem used is a transportation problem and hence a special case of the min-cost flow problem.
- EMD is a metric if the distributions are of equal length and the ground distances are metric.
- It is robust in comparison to other histogram matching techniques.
- It can be used for partial matching.

If x and y are probability distributions, $\sum_i x_i = \sum_j y_j = 1$. So equation 3.2 reduces to,

$$EMD(x,y) = \sum_i \sum_j c_{ij} f^*_{ij}$$

which is just the objective function value of the optimization problem.

When considering the EMD between the normalized count vector and the uniform distribution, the ground distance chosen should be such that the comparison of values corresponding to each item are weighted equally. Also, no comparison is to be done between components corresponding to different items. This is achieved using a 0-1 cost function.

$$c_{ij} = \begin{cases} 1 & : i \neq j \\ 0 & : i = j \end{cases}$$

With this cost function, the optimal value of the transportation problem becomes a closed form expression. When x and y are probability distributions, this is given by

$$EMD(x, y) = 1 - \sum_i \min(x_i, y_i) \tag{3.3}$$

Then, a closed form expression can be obtained for EMD as follows,

$$\begin{aligned} \sum_i \sum_j c_{ij} f_{ij} &= \sum_i \sum_{j \neq i} f_{ij} = \sum_i \left(\sum_j f_{ij} - f_{ii} \right) \\ &= \sum_i (x_i - f_{ii}) = 1 - \sum_i f_{ii} \end{aligned}$$

So, to minimize the objective function value, we need to maximize $\sum_i f_{ii}$. But,

$$\begin{aligned} \sum_i f_{ij} = y_j &\quad\Rightarrow\quad f_{ii} \leq y_i \\ \sum_j f_{ij} = x_i &\quad\Rightarrow\quad f_{ii} \leq x_i \end{aligned}$$

Hence the maximum value of f_{ii} is $\min(x_i, y_i)$. This is independent of the value of $f_{i'i'}$ for any other $i' \neq i$. Hence the minimum value of the objective is $1 - \sum_i \min(x_i, y_i)$. This is equal to the histogram intersection distance between the distributions x and y defined in [12].

Intuitively, we can explain this as follows. When defining EMD, one distribution is viewed as piles of earth and the other as holes. The ground distance c_{ij} is the distance from the i^{th} pile of dirt to the j^{th}

hole and the EMD is the minimum amount of work that needs to be done to move all the earth into holes. For a 0-1 cost function, the i^{th} pile of earth is inside the i^{th} hole (so that no cost is incurred in moving earth from this pile to this hole) and the cost of moving a unit of earth from any pile to any other hole is 1. Then the minimum cost of the operation is equal to the amount of dirt that towers above the corresponding holes, which is equal to the difference between the total amount of dirt and the amount of dirt that is inside holes. When the distributions are probability distributions, the total amount of dirt is 1 (as $\sum_i x_i = \sum_i y_i = 1$) and for each hole-pile pair, the amount of dirt inside the hole is the minimum of the height of the pile and depth of the hole, which is $\min(x_i, y_i)$, which results in equation 3.3.

When y is the uniform distribution of n components, the histogram intersection distance to x, denoted as $D_I(x)$ becomes

$$D_I(x) = 1 - \sum_i \min\left(x_i, \frac{1}{n}\right) \tag{3.4}$$

where n is the number of items.

Henceforth, we refer to the histogram distance to the uniform distribution as simply intersection distance.

3.3 Properties of Intersection Distance

3.3.1 Relation with Coverage

Lemma 1. *Let x and x' be two normalized count vectors, such that the coverage of x' is greater than that of x. Then $D_I(x') \leq D_I(x)$.*

Proof. Recall that coverage of the system is the total number of items recommended across all users, that is, $\sum\limits_i I[x_i \neq 0]$. Thus, the coverage of a vector x is the number of non-zero components. To analyze the effect of increasing coverage, it is sufficient to analyze the effect of changing the value of any one x_i from 0 to $\epsilon > 0$. Then, this operation can be repeated any number of times to simulate the effect of any increase in coverage.

Since $\sum_i x_i = 1$, if x_i increases from 0 to $\epsilon > 0$, $\exists\ x_j > \epsilon$ that decreases to $x_j - \epsilon$. Let x' be the distribution after the change. Then, the following cases arise -

- **Case 1:** $x'_j > \frac{1}{n}$

$$\begin{aligned} D_I(x) &= 1 - \sum_k \min\left(x_k, \frac{1}{n}\right) \\ &= 1 - \sum_{k \neq i,j} \min\left(x_k, \frac{1}{n}\right) - min\left(x_j, \frac{1}{n}\right) \\ &= 1 - \sum_{k \neq i,j} \min\left(x_k, \frac{1}{n}\right) - \frac{1}{n} \end{aligned}$$

Also, $x_j > x'_j \Rightarrow x_j > \frac{1}{n}$. Then,

$$\begin{aligned} D_I(x') &= 1 - \sum_k \min\left(x'_k, \frac{1}{n}\right) \\ &= 1 - \sum_{k \neq i,j} \min\left(x_k, \frac{1}{n}\right) - min\left(x'_j, \frac{1}{n}\right) - \epsilon \\ & \quad (as\ x_k = x'_k\ \forall\ k \neq i, j) \\ &= 1 - \sum_{k \neq i,j} \min\left(x_k, \frac{1}{n}\right) - \frac{1}{n} - \epsilon \end{aligned}$$

Hence $D_I(x') < D_I(x)$.

- **Case 2:** $x_j \geq \frac{1}{n}, x'_j < \frac{1}{n}$

$$\begin{aligned} D_I(x) &= 1 - \sum_{k \neq i,j} \min\left(x_k, \frac{1}{n}\right) - \frac{1}{n} \\ D_I(x') &= 1 - \sum_{k \neq i,j} \min\left(x_k, \frac{1}{n}\right) - min\left(x'_j, \frac{1}{n}\right) - \epsilon \\ &= 1 - \sum_{k \neq i,j} \min\left(x_k, \frac{1}{n}\right) - x'_j - \epsilon \\ &= 1 - \sum_{k \neq i,j} \min\left(x_k, \frac{1}{n}\right) - (x_j - \epsilon) - \epsilon \\ &= 1 - \sum_{k \neq i,j} \min\left(x_k, \frac{1}{n}\right) - x_j \end{aligned}$$

Since $x_j \geq \frac{1}{n}$,

$$\begin{aligned} & 1 - \sum_{k \neq i,j} \min\left(x_k, \frac{1}{n}\right) - x_j \\ & \qquad \leq 1 - \sum_{k \neq i,j} \min\left(x_k, \frac{1}{n}\right) - \frac{1}{n} \\ & \Rightarrow D_I(x') \leq D_I(x) \end{aligned}$$

- **Case 3:** $x_j < \frac{1}{n}$
  Since $x_j > x'_j$, this implies that $x'_j < \frac{1}{n}$. Then,

$$\begin{aligned}
D_I(x) &= 1 - \sum_k \min\left(x_k, \frac{1}{n}\right) \\
&= 1 - \sum_{k \neq i,j} \min\left(x_k, \frac{1}{n}\right) - min\left(x_j, \frac{1}{n}\right) \\
&= 1 - \sum_{k \neq i,j} \min\left(x_k, \frac{1}{n}\right) - x_j \\
D_I(x') &= 1 - \sum_k \min\left(x'_k, \frac{1}{n}\right)
\end{aligned}$$

$$\begin{aligned}
&= 1 - \sum_{k \neq i,j} \min\left(x_k, \frac{1}{n}\right) - min\left(x'_j, \frac{1}{n}\right) - \epsilon \\
&= 1 - \sum_{k \neq i,j} \min\left(x_k, \frac{1}{n}\right) - x'_j - \epsilon \\
&= 1 - \sum_{k \neq i,j} \min\left(x_k, \frac{1}{n}\right) - (x_j - \epsilon) - \epsilon \\
&= 1 - \sum_{k \neq i,j} \min\left(x_k, \frac{1}{n}\right) - x_j
\end{aligned}$$

  Hence $D_I(x) = D_I(x')$.

$\square$

3.3.2 Relation with l_1 Distance

Lemma 2. *The histogram intersection distance between two distributions x and y is equal to half the l_1 distance between them.*

Proof.

$$\begin{aligned}
\sum_i \min(x_i, y_i) &= \sum_i \frac{\mid x_i + y_i \mid - \mid x_i - y_i \mid}{2} \\
&= \sum_i \frac{\mid x_i + y_i \mid}{2} - \sum_i \frac{\mid x_i - y_i \mid}{2} \\
&= \frac{1}{2}\left(\sum_i (x_i + y_i)\right) - \frac{D_{l_1}(X,Y)}{2} \text{ (as } x_i, y_i \geq 0 \text{)}
\end{aligned} \tag{3.5}$$

$$
\begin{aligned}
&= \frac{\sum_i x_i}{2} + \frac{\sum_i y_i}{2} - \frac{D_{l_1}(X,Y)}{2} \\
&= \frac{n}{2} + \frac{n}{2} - \frac{D_{l_1}(X,Y)}{2} \\
\Rightarrow \quad & n - \sum_i \min(x_i, y_i) = \frac{D_{l_1}(X,Y)}{2} \\
\Rightarrow \quad & D_I(X,Y) = \frac{D_{l_1}(X,Y)}{2}
\end{aligned}
$$

□

3.3.3 Worst Case Value

Lemma 3. *If x is the normalized count vector of a valid recommendation system, then $D_I(x) \in [0, 1 - \frac{N}{n}]$.*

Proof. Trivially, $D_I(x) \in [0,1]$, where the lower bound of 0 is attained when the normalized count vector is equal to the uniform distribution. Hence it is sufficient to prove that $D_I(x) \leq 1 - \frac{N}{n}$.

A valid recommendation system will not have the same item occur twice in the recommendation list of a user. So, in a recommendation list of length N, there must be exactly N distinct items. Hence, at least N items must get recommended. Consider the case where every user gets this same set of N items. This is the only way in which only N items can be recommended.

Since increasing coverage cannot increase the distance to the uniform distribution, if more than N items get recommended, the intersection distance cannot be greater than that obtained in the case of N items. Since coverage cannot be less than N, the maximum intersection distance is obtained in the case stated.

WLOG assume that the N items recommended to all users are $i = 1, 2 \ldots N$. Let $\mathcal{U}$ be the set of all users and $\mathcal{I}$ be the set of all items. Then the number of times item i gets recommended, c_i is given by,

$$
c_i = \begin{cases} |\mathcal{U}| & i \in \{1, 2 \ldots N\} \\ 0 & \text{otherwise} \end{cases}
$$

Hence, $\sum\limits_{i \in \mathcal{I}} c_i = N|\mathcal{U}|$. So,

$$
x_i = \begin{cases} |\frac{1}{N}| & i \in \{1, 2 \ldots N\} \\ 0 & \text{otherwise} \end{cases}
$$

Then,

$$\begin{aligned} D_I(x) &= 1 - \sum_{i \in \mathcal{I}} \min\left(\frac{1}{n}, x_i\right) = 1 - \left\{ N \min\left(\frac{1}{n}, \frac{1}{N}\right) \right. \\ &\quad \left. + (n - N) \min\left(\frac{1}{n}, 0\right) \right\} = 1 - \frac{N}{n} \end{aligned}$$

□

3.4 Summary

From lemma 1, we can see that increasing the coverage of a recommender system cannot increase the distance to the uniform distribution. This is desirable when considering intersection distance as a metric for aggregate diversity because we intuitively expect a system that recommends more items to be more diverse, a property that was not satisfied by the probability-based metrics.

Further, if many items are recommended to a large number of users, most components of the vector x will be roughly equal and close to $\frac{1}{n}$, resulting in a smaller intersection distance. If only a few items are recommended often, only a few components of x will have a value large enough to decrease the intersection distance noticeably.

An additional advantage of intersection distance is that it is bounded to $[0, 1]$ even in the general case, with a tighter bound for the case of valid recommendation lists, as shown in lemma 3. Further, it is easy to compute intersection distance, either using equation 3.3 or as half the l_1 distance. These properties make intersection distance a good choice of metric to evaluate aggregate diversity.

CHAPTER 4

Basic Heuristics to Improve Diversity

4.1 Pseudo Gradient Descent

The task of recommendation can often be separated into two phases - rating prediction and creation of recommendation lists. It is possible to optimize metrics associated with the final recommendation lists by altering the second phase alone, as is done in simple heuristics such as random re-ranking of item.

The second phase can be viewed as an incremental process where, at an instant, we have to decide what is the next best item to recommend to the user currently being served. During this process, ideally, we would like to use a technique like gradient descent to minimize the distance of the PDF induced from the counts of item recommendations to the uniform distribution.

We know that the the intersection distance is equal to half the L_1 distance between the normalized count vector and the uniform distribution. In gradient descent, we attempt to minimize a function by moving along the direction opposite to the gradient. Here, it is sufficient to look at the gradient of the L_1 distance as it is along the same direction as that of the intersection distance.

We have,

$$D_{L_1}(X) = \sum_i \mid x_i - \frac{1}{n} \mid$$

where n is the total number of items available. Hence,

$$\frac{\partial D_{l_1}}{\partial x_i} = \frac{\mid x_i - \frac{1}{n} \mid}{x_i - \frac{1}{n}}$$

Let c_i be the count of item i and x_i be the induced probability value of item i. Then, if $N = \sum_i c_i$,

$$x_i = \frac{c_i}{N}$$

Suppose at this instant, item k gets recommended. Let the counts histogram change to c' and the induced probabilities to x'. Then,

$$\begin{aligned}
c'_k &= c_k + 1 \\
c'_i &= c_i \quad \forall\, i \neq k \\
x'_k &= \frac{c'_k}{\sum_i c'_i} = \frac{c_k + 1}{N+1} = \frac{N}{N+1}\left(\frac{c_k}{N}\right) + \frac{1}{N+1} = \frac{N x_k}{N+1} + \frac{1}{N+1} \\
x'_i &= \frac{c'_i}{\sum_i c'_i} = \frac{c_i}{N+1} = \frac{N}{N+1}\left(\frac{c_i}{N}\right) = \frac{N c_i}{N+1} \quad (\forall\, i \neq k)
\end{aligned}$$

Let $\Delta x^{(k)}$ be the change in the vector x when the count of item k is incremented. Then,

$$\begin{aligned}
\Delta x^{(k)}_k &= \frac{1 - x_k}{N+1} \\
\Delta x^{(k)}_i &= \frac{-x_i}{N+1} \quad \forall\, i \neq k
\end{aligned}$$

The magnitude of Δx is fixed (albeit not constant) but its direction depends on the selected item k. Thus, an approximation to gradient descent would be to choose that item k which has the maximum component along $-\nabla D_{l_1}(x)$, that is, choose the item,

$$k^* = \arg\max_k \left(- < \Delta x^{(k)}, \nabla D_{l_1}(x) >\right) = \arg\min_k \left(< \Delta x^{(k)}, \nabla D_{l_1}(x) >\right) \tag{4.1}$$

This method will be referred to as *pseudo gradient descent* for future reference.

We can easily incorporate constraints such as minimum expected rating of an item or only items not seen (rated) by the user by considering only the set F of items that satisfy these constraints. That is,

$$k^* = \arg\max_{k \in F} \left(- < \Delta x^{(k)}, \nabla D_{l_1}(x) >\right) = \arg\min_{k \in F} \left(< \Delta x^{(k)}, \nabla D_{l_1}(x) >\right)$$

Note that this is a very simple technique that assumes that does not even assume that we can influence the ratings predicted by the recommender system. It is implemented by taking any standard recommender system as a base, selecting all items that have a predicted rating higher than a suitable threshold and recommending items according to the rule in 4.1.

Another possible baseline heuristic to improve diversity would be to obtain items with rating above a suitable threshold from a base recommender system and greedily recommend that item that would result in the maximum decrease in intersection distance.

4.2 Experiments

To test these methods, we needed implementations of standard recommender systems to provide base ratings. Experiments were conducted on the Apache Mahout platform [1] which provides parallel map-reduce based implementations of some basic recommender systems.

The following are the baseline recommendations systems used -

- ItemAverage - A simple recommender that always estimates the preference for an item to be the average of all known preference values for that item.
- ItemUserAverage - Like ItemAverage, except that estimated preferences are adjusted for the users' average preference value.
- UserBased - User based collaborative filtering.
- ItemBased - Item based collaborative filtering.
- ALSWR - Matrix factorization using alternating least squares with weighted-λ regularization [41].

Experiments were conducted on the MovieLens [2] and Netflix [3] datasets. The MovieLens dataset was collected by the GroupLens Research Project at the University of Minnesota. The statistics of the dataset are as follows -

No of users = 943

No of artists = 1682

Min ratings per user = 20

Min ratings per artist = 1

Max ratings per user = 737

Max ratings per artist = 583

Mean ratings per user = 106.044

Mean ratings per artist = 59.453

Variance in ratings per user = 10176.414

Variance in ratings per artist = 6457.721

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

[2] http://grouplens.org/datasets/movielens/

[3] http://www.netflixprize.com/community/viewtopic.php?id=68

Std dev in ratings per user = 100.878

Std dev in ratings per artist = 80.360

The Netflix dataset available for download has 480189 users and 17770 movies. This large size made it prohibitive for use in later experiments that involved transfer of the full rating matrix between programs. Hence, we used a representative sample created by selecting 10% of the movies at random and for each, retaining 33% of the ratings (stratified item sampling). From the resultant sample, we removed users with < 50 and movies with < 20 ratings. The resulting dataset had 3985 users and 1101 movies. Its statistics are as follows -

No of users = 3985

No of movies = 1101

Min ratings per user = 50

Min ratings per movie = 20

Max ratings per user = 357

Max ratings per movie = 1335

Mean ratings per user = 65.612

Mean ratings per movie = 237.477

Variance in ratings per user = 495.318

Variance in ratings per movie = 88178.809

Std dev in ratings per user = 22.256

Std dev in ratings per movie = 296.949

The following metrics were used to compare different methods -

- Intersection Distance between the normalized count vector and the corresponding uniform distribution (ID) is used to measure aggregate diversity. If item i has been recommended c_i times,

$$ID = 1 - \sum_i \left| \frac{c_i}{\sum_j c_j} - \frac{1}{n} \right|$$

  where n is the number of items.

- Item coverage (IC) - another coarser measure of aggregate diversity that is relatively easy to interpret. It is the number of items that occur in the top-N list of at least one user.

- Average Predicted Rating (APR) - This is indicative of how relevant a recommendation list is considered to be, by the recommender system. If $\mathcal{U}$ is the set of users, $L(u)$ is the recommendation list for user u and $\hat{r}(i, u)$ is the predicted rating of user u for item i, then,

$$APR = \frac{1}{N|\mathcal{U}|} \sum_{u \in \mathcal{U}} \sum_{i \in L(u)} \hat{R}_{ui}$$

Using each of the available recommender systems as a baseline to provide an initial list of relevant items, the following algorithms were tested -

- Pseudo Gradient Descent (PSG) - Iteratively recommends that item for which the component of the subsequent change in the induced PMF of the histogram of the number of times an item has been recommended along the negative of gradient of the required EMD (intersection distance) is maximum among the items relevant for the user.
- Unconstrained Pseudo Gradient Descent (UPSG) - Like PseudoGradientDescent but does not restrict itself to items relevant to the user. However, if two items have equal contribution, the item with a higher predicted rating for the user is chosen.
- Greedy - Iteratively recommends the item among the items relevant for the user, that results in maximum decrease in intersection distance.
- UnconstrainedGreedy - Like Greedy, but does not restrict itself to items relevant to the user.
- Random - Randomly selects N items from those relevant to the user.

As is the case with most gradient descent based methods, Pseudo Gradient Descent needs a good starting point. In the implementation, initially, the normalized count vector is initialized to something close to the uniform distribution to avoid loval optima problems.

These experiments were expected to verify the following hypotheses -

- Both methods result in a lower intersection distance than the baseline (any standard recommender system that they use to determine relevance of an item).
- Both methods do not substantially decrease the quality of the recommended list (in terms of item relevance).
- Randomly selecting items from the pool of relevant items should either result in lower diversity, relevance or both.
- If no relevance threshold is imposed, relevance of the list can degrade significantly, in either method.
- Diversity improves on increasing N (number of items in a user's recommendation list) but relevance degrades.

The results on the MovieLens dataset are presented in tables 4.1 - 4.5.

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.992	13.0	4.994	0.982	31.0	4.739
Random	0.951	82.0	4.443	0.901	166.0	4.262
Greedy	0.944	100.0	4.449	0.944	100.0	4.449
Unconstrained Greedy	0.189	1672.0	2.417	0.189	1672.0	2.417
Pseudo Gradient Descent	0.944	100.0	4.449	0.898	177.0	4.274
Unconstrained Pseudo Gradient Descent	0.124	1654.0	2.922	0.117	1662.0	2.936

Table 4.1: Results of basic heuristics on MovieLens : Baseline - ItemAverage

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.992	13.0	4.994	0.982	31.0	4.739
Random	0.951	82.0	4.466	0.906	158.0	4.273
Greedy	0.944	100.0	4.449	0.944	100.0	4.449
Unconstrained Greedy	0.189	1672.0	2.417	0.189	1672.0	2.417
Pseudo Gradient Descent	0.944	100.0	4.449	0.898	177.0	4.274
Unconstrained Pseudo Gradient Descent	0.122	1653.0	2.928	0.121	1655.0	2.927

Table 4.2: Results of basic heuristics on MovieLens : Baseline - ItemUserAverage

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.833	281.0	5.0	0.698	508.0	5.0
Random	0.712	485.0	4.995	0.565	732.0	4.911
Greedy	0.539	966.0	4.996	0.539	966.0	4.996
Unconstrained Greedy	0.157	1682.0	1.338	0.157	1682.0	1.338
Pseudo Gradient Descent	0.539	966.0	4.996	0.472	1218.0	4.914
Unconstrained Pseudo Gradient Descent	0.084	1682.0	2.344	0.086	1682.0	2.328

Table 4.3: Results of basic heuristics on MovieLens : Baseline - ItemBased

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.883	196.0	3.875	0.838	273.0	3.707
Random	0.862	232.0	3.638	0.825	294.0	3.584
Greedy	0.794	346.0	3.624	0.794	346.0	3.624
Unconstrained Greedy	0.227	1301.0	-0.997	0.227	1301.0	-0.997
Pseudo Gradient Descent	0.794	346.0	3.621	0.758	407.0	3.585
Unconstrained Pseudo Gradient Descent	0.427	964.0	-0.986	0.436	948.0	-0.983

Table 4.4: Results of basic heuristics on MovieLens : Baseline - UserBased

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.894	179.0	4.688	0.827	291.0	4.575
Random	0.844	262.0	4.415	0.761	402.0	4.261
Greedy	0.778	460.0	4.412	0.778	460.0	4.412
Unconstrained Greedy	0.166	1682.0	2.342	0.165	1682.0	2.322
Pseudo Gradient Descent	0.778	460.0	4.412	0.731	647.0	4.263
Unconstrained Pseudo Gradient Descent	0.082	1681.0	2.851	0.081	1681.0	2.847

Table 4.5: Results of basic heuristics on MovieLens : Baseline - ALSWR

The results on the Netflix dataset are presented in tables 4.6 - 4.10.

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.986	16.0	4.261	0.974	30.0	4.184
Random	0.941	67.0	4.032	0.892	124.0	3.913
Greedy	0.941	68.0	4.03	0.941	68.0	4.03
Unconstrained Greedy	0.662	969.0	2.218	0.662	969.0	2.218
Pseudo Gradient Descent	0.941	68.0	4.03	0.891	130.0	3.911
Unconstrained Pseudo Gradient Descent	0.66	976.0	2.418	0.66	981.0	2.418

Table 4.6: Results of basic heuristics on Netflix : Baseline - ItemAverage

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.986	16.0	4.261	0.974	30.0	4.184
Random	0.942	66.0	4.028	0.891	124.0	3.911
Greedy	0.941	68.0	4.03	0.941	68.0	4.03
Unconstrained Greedy	0.662	969.0	2.218	0.662	969.0	2.218
Pseudo Gradient Descent	0.941	68.0	4.03	0.891	130.0	3.911
Unconstrained Pseudo Gradient Descent	0.66	983.0	2.419	0.659	987.0	2.416

Table 4.7: Results of basic heuristics on Netflix : Baseline - ItemUserAverage

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.784	324.0	4.596	0.718	512.0	4.345
Random	0.61	603.0	4.051	0.469	895.0	3.856
Greedy	0.588	844.0	4.04	0.588	844.0	4.04
Unconstrained Greedy	0.015	1101.0	2.922	0.015	1101.0	2.922
Pseudo Gradient Descent	0.588	844.0	4.04	0.459	1050.0	3.855
Unconstrained Pseudo Gradient Descent	0.011	1101.0	3.1	0.011	1101.0	3.1

Table 4.8: Results of basic heuristics on Netflix : Baseline - ItemBased

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.671	491.0	3.914	0.638	599.0	3.498
Random	0.638	539.0	3.419	0.634	603.0	3.433
Greedy	0.635	610.0	3.43	0.635	610.0	3.43
Unconstrained Greedy	0.367	701.0	-1.0	0.367	701.0	-1.0
Pseudo Gradient Descent	0.635	610.0	3.43	0.635	610.0	3.43
Unconstrained Pseudo Gradient Descent	0.547	500.0	-1.0	0.541	507.0	-0.999

Table 4.9: Results of basic heuristics on Netflix : Baseline - UserBased

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.844	234.0	4.233	0.822	306.0	4.159
Random	0.778	317.0	4.036	0.729	430.0	3.921
Greedy	0.772	429.0	4.038	0.772	429.0	4.038
Unconstrained Greedy	0.259	1000.0	2.581	0.227	1016.0	2.623
Pseudo Gradient Descent	0.772	429.0	4.038	0.721	548.0	3.921
Unconstrained Pseudo Gradient Descent	0.244	1034.0	2.738	0.215	1053.0	2.787

Table 4.10: Results of basic heuristics on Netflix : Baseline - ALSWR

4.3 Observations and Discussion

We observe that both the Pseudo Gradient Descent and greedy methods are able to lower the intersection distance of the baseline recommenders. However, they do not perform significantly better than simple re-ranking. Further, there is little difference between the performance of the two methods, except that the performance of Pseudo Gradient Descent improves on increasing N.

On the other hand, we see that removing the constraint on rating threshold improves the intersection distance of both Pseudo Gradient Descent and greedy methods dramatically, indicating that there is considerable scope for improvement in this direction. However, this is also accompanied by a noticeable drop in the average predicted rating, as expected. In both respects, there is little difference between the two heuristics.

Unconstrained methods are also significantly more time consuming that constrained ones because to recommend each item for each user, it is necessary to iterate over the entire item set. In future experiments, to compare against these heuristics, the unconstrained versions of the pseudo gradient descent and greedy methods have been used.

CHAPTER 5

Auction Formulations

5.1 Item-as-bidder Model

Ideally, if we wish to optimize a recommender system for intersection distance, we would like to be able to compute the intersection distance, given the parameter values of the system and then modify those parameter values such that the intersection distance is improved. Since the intersection distance is calculated using recommendation lists, this requires us to be able to determine which items would be placed in a user's recommendation list. Many models of recommender systems have probabilistic parameters Θ and a function that computes the predicted rating of a user for an item given Θ. Then, to be able to determine the top-N items for a user, we need to be able to solve the following problem.

A number Θ is drawn according to some distribution and n functions $r_1(\Theta), r_2(\Theta) \ldots r_n(\Theta)$ are evaluated. If these numbers are sorted in descending order, what is the probability that $r_i(\Theta)$ will lie in the top-N?

In order to discover what assumptions need to be made on the functions $r_1, r_2, \ldots$ for the above problem to be solvable, we considered the following problem, which is similar to the stated problem.

Consider an auction in which each item in the recommender system is a bidder in the auction and each user is a good for sale. For each good (user), there is a separate auction with N units of the good, where N is the number of items required in a user's recommendation list. The valuation and bid of a bidder (item) i for a good (user) u is the predicted rating $\hat{R}_{ui}$. Bidders are assumed to bid truthfully so we do not require an incentive compatible mechanism for the auction. Then, this design will allocate the N units of good u to the N highest bidders, which means that the N items of highest predicted rating will be placed in the user's recommendation list. If we can calculate the probability that the a particular bidder (item) will be a winner in an auction (be placed in that user's recommendation list), then we have obtained the probability that the item has been recommended to that user.

This problem essentially requires us to determine the probability of a bidder winning a multi-unit auction, under suitable assumptions about the valuations of the bidders. We could not find or derive any results of this nature. Further, we found that this problem was related to the order statistics problem. The

k^{th} order statistic of n random variables is the value of the k^{th} smallest of the variables. Specifically, we need the $(n-N)^{th}$ order statistic $X_{(n-N)}$ of n independent, non-identically distributed normal random variables $\hat{R}_{u1}, \hat{R}_{u2} \ldots \hat{R}_{un}$. Then,the probability that item i gets recommended to user u is $Pr(\hat{R}_{ui} \geq X_{(n-N)})$. Predicted ratings $\hat{R}_{ui}$ are typically modelled as Gaussian random variables. To our knowledge, there are no results for the order statistics of Gaussian distributions. An approximate solution was found in [23] but this was cumbersome to use and we could not derive further results using this.

5.2 User-as-bidder Model

Another formulation we attempted made use of a k-duplicates combinatorial auction setup. Consider an single auction with each user in the recommender system as a bidder and each item as a good for sale. The valuation and bid of bidder u for good i is $\hat{R}_{ui}$. A bidder will bid for at most one unit of a good and will bid truthfully, thus removing the need for incentive compatibility. This is now a special case of the formulation used in [5] and hence, their algorithm can be applied to it. Unfortunately, this is only an approximate solution to the problem, although a fairly good one.

Each item can be recommended to at most k users in this setting, which is guaranteed to increase aggregate diversity. It can in fact be shown that the intersection distance can be lower bounded if each item is recommended at most k times. Additionally,

- The efficient allocation (optimal solution to the WDP) aims to maximize the sums of declared valuations of all bidders, which is equivalent to maximizing the total rating of all the lists. Thus, this method maximizes relevance for a certain level of diversity.
- Alternately a lower bound on relevance can be ensured as follows. To prevent items below a certain rating from being recommended to a user, set a reserve price for all items as the minimum rating desired.

However, this formulation poses some difficulties. This model ensures that for each item i, the number of times it gets recommended, $c_i \leq k$ where k is a tunable parameter. To calculate intersection distance, we need the a bound on the normalized count, that is, on $x_i = \frac{c_1}{\sum\limits_i c_i}$. But in this method, it is not even necessary that $\sum\limits_i c_i > 0$. For example, if we set a reserve price (rating threshold) as 6 when all ratings are in the range 1-5, no bidder has a positive surplus on any item, and hence, no item gets recommended.

Also, by setting a low enough reserve price, we can get $c_i = k \, \forall \, i$, which would result in zero intersection distance. But as there are no other constraints on this allocation, it is entirely possible that the same k

users win each auction, that is, all items get recommended to the same k users. This would mean that for most users, no items get recommended. A simple, albeit unlikely example is the case where one user has a rating of 5 for every item but all other users have a rating < 5 for every item. Then, if there is one unit of each item, an efficient auction would allocate all items to the user with all ratings as 5. We can, however, show that if user-mean centered ratings are used, that is, if the valuation $v_{ui} = \hat{R}_{ui} - \frac{1}{|\mathcal{I}|}\sum_i \hat{R}_{ui}$, then for any pair of users p, q, $\exists\ i \in \mathcal{I} : v_{pi} \leq v_{qi}$.

This is proved as follows.

$$v_{pi} - v_{qi} = \left(\hat{R}_{pi} - \hat{R}_{qi}\right) - \frac{1}{|\mathcal{I}|}\sum_{j \in \mathcal{I}}\left(\hat{R}_{pj} - \hat{R}_{qj}\right)$$

Let $X_i = \hat{R}_{pi} - \hat{R}_{qi}$. Then,

$$v_{pi} - v_{qi} = X_i - \frac{1}{|\mathcal{I}|}\sum_{j \in \mathcal{I}} X_j$$

Suppose $v_{pi} - v_{qi} > 0\ \forall\ i \in \mathcal{I}$

$$\begin{aligned}
\Rightarrow \quad & \sum_{i \in \mathcal{I}} v_{pi} - v_{qi} > 0 \\
\Rightarrow \quad & \sum_{i \in \mathcal{I}} \left(X_i - \frac{1}{|\mathcal{I}|}\sum_{j \in \mathcal{I}} X_j \right) > 0 \\
\Rightarrow \quad & \sum_{i \in \mathcal{I}} X_i - \sum_{j \in \mathcal{I}} X_j > 0 \\
\Rightarrow \quad & 0 > 0
\end{aligned}$$

which is a contradiction. Hence, $\exists\ i \in \mathcal{I} : v_{pi} \leq v_{qi}$.

This indicates that although theoretically it is possible for only a small number of users to get recommendations, the worst case situation can be avoided. However, this still does not ensure that there is an equitable distribution of recommended items among users. One possible method to ensure this would be to introduce budget constraints for bidders that could be used to limit the number of items recommended to a user. However, we were unable to find solutions to budget-constrained multi-unit combinatorial auctions. Later, we managed to formulate the problem as a min cost flow problem, which is much easier to solve optimally. Using this, we can place a threshold on the number of times an item is recommended, while also ensuring that items are recommended for all users. This is discussed in the next chapter.

CHAPTER 6

Min-Cost-Flow Based Formulations

In this chapter, we propose two min cost flow based algorithms that attempt to improve the intersection distance of a recommendation system in different ways. Both algorithms are two-phase approaches. The first phase involves obtaining the predicted rating matrix from a standard baseline recommender system. In the second phase, a minimum cost flow problem is used to identify recommendation lists of length N for users such that intersection distance is minimized, without loss in the relevance of recommendations for the user. Tha algorithms differ in their second phase. The first method attempts to threshold the number of times each item gets recommended, thereby bounding intersection distance. The second method designs a problem that jointly minimizes the total rating of the recommended items (a measure of relevance) and intersection distance. We use standard baseline algorithms for the first phase. The second phase of both algorithms is discussed in the following sections.

6.1 Min Cost Flow Problem

The minimum cost flow problem is a generalization of the maximum flow problem [19]. The problem is defined over a flow network $G = (V, E)$. Each edge $(u, v) \in E$ is associated with a capacity $l(u, v)$ and a cost $c(u, v)$ and each node $v \in V$ is associated with a balance - a demand or a supply, $b(v)$. By convention, $b(v) < 0$ is interpreted as a demand and $b(v) > 0$ as supply. A pseudoflow is a function $X : E \to \mathbb{R}$ satisfying the following capacity and antisymmetry constraints for each edge $(u, v) \in E$ -

$$X(u,v) \leq l(u,v)$$
$$X(u,v) = -X(v,u)$$

The excess of a pseudoflow X at node u, $e_X(u)$ is defined as

$$e_X(u) = b(u) - \sum_{(u,v)\in E} X(u,v)$$

A feasible flow is a pseudoflow X such that $e_X(v) = 0 \; \forall \; v \in V$. The cost of a pseudoflow X, $c(X)$ is given by

$$c(X) = \frac{1}{2} \sum_{(u,v)\in E} c(u,v)X(u,v)$$

The minimum cost flow problem is to find a flow of minimum cost. The integrality theorem for minimum cost flow states that if all capacities and balances are integers and there exists a feasible flow, there exists an integral minimum cost flow.

6.2 Bounding Number of Times an Item gets Recommended

6.2.1 Formulation

Suppose we are given the set of users $\mathcal{U}$ and the set of items $\mathcal{I}$. Let $\hat{R}_{ui}$ be the rating of user $u \in \mathcal{U}$ for item $i \in \mathcal{I}$, predicted by a standard recommendation system. Instead of just selecting the N items of highest rating to be recommended to a user, we would like to select items appropriately such that the intersection distance is decreased without compromising much on the relevance to the user. The following method assumes that we have access to a base recommendation system that provides us with reliable predicted ratings - a reasonable assumption, given the success of a number of rating prediction algorithms in competitions such as the Netflix prize.

Consider the following min cost flow problem. Let the flow network be $G = (V, E)$ with $V = \mathcal{U} \cup \mathcal{I} \cup \{u_d\}$, where u_d is a dummy user node. There is a directed edge from $i \in \mathcal{I}$ to $u \in \mathcal{U}$ iff $\hat{R}_{ui} \geq r_T$ (r_T is some suitable threshold) and $(i, u_d) \in E \; \forall \; i \in \mathcal{I}$.

The cost of edge (i, u), is $c_{iu} = M - \hat{R}_{ui}, i \in \mathcal{I}, u \in \mathcal{U}$ where M is a constant such that $M \geq \hat{R}_{ui} \; \forall \; u, i$. $c_{iu_d} = M \; \forall \; i \in \mathcal{I}$. Thus, an edge between a user and item is more costly is the user does not prefer the item much.

Each edge (i, u) has a capacity l_{iu} where

$$l_{iu} = \begin{cases} 1 & u \neq u_d \\ k_2 - k_1 & u = u_d \end{cases}$$

where k_1 and k_2 are parameters of the model.

The balances (supplies / demands) at nodes are given by

$$b_v = \begin{cases} -N & v = u \in \mathcal{U} \\ k_2 & v = i \in \mathcal{I} \\ -(k_2|\mathcal{I}| - N|\mathcal{U}|) & v = u_d \end{cases}$$

An integral min cost flow will only have a flow of 1 or 0 along edges $(i, u), u \neq u_d$. If the flow in edge (i, u) is 1, we interpret this as placing i in the recommendation list of u. The integrality theorem for min cost flow states that if the capacities and demands are integral and a feasible flow exists, then there is a minimum cost flow which is integral on each arc. When the flow is integral, the capacity constraint ensures that an item is recommended to a user exactly once and the balances ensure that

- Every user gets exactly N items.
- Every item is assigned to at most k_2 users.
- At least k_1 units of each item must go to non-dummy users.

Thus, each item gets recommended at least k_1 and at most k_2 times, that is, $k_1 \leq c_i \leq k_2 \ \forall \ i \in \mathcal{I}$. Now we can use bounds that will be proved in section 6.2.2 to guarantee an upper bound on the intersection distance of the resultant set of recommendation lists generated by this problem. In practice, it is convenient to use $k_1 = 0$ because it is difficult to ensure that all items get recommended at least once, and this is also not always desirable.

Note that as there are only incoming edges to the dummy user u_d, it is necessary for the balance at node u_d to be negative (u_d must be a demand node, not a supply node). Thus, we have feasible solutions only if $k_2|\mathcal{I}| \geq N|\mathcal{U}| \Rightarrow k_2 \geq \frac{N|\mathcal{U}|}{|\mathcal{I}|}$.

6.2.2 Theoretical Guarantees

Review of notation - $\mathcal{I}$ - Set of all items

c_i - Number of times item i is recommended

n - Total number of items

$x_i = \frac{c_i}{\sum_{i \in \mathcal{I}} c_i}$

$D_I(x)$ - Intersection distance of the normalized count vector x from the uniform distribution

A Weak Bound

Lemma 4. *The intersection distance for a recommender system which satisfies $k_1 \leq c_i \leq k_2 \;\forall\; i \in \mathcal{I}$ is at most $\frac{k_2-k_1}{2k_1}$.*

Proof. Suppose $k_1 \leq c_i \leq k_2 \;\forall\; i \in \mathcal{I}$. Then,

$$\begin{aligned}
& c_i \leq k_2, \quad \sum_{i \in I} c_i \geq k_1 n \\
\Rightarrow \quad & x_i \leq \frac{k_2}{k_1 n} \\
\Rightarrow \quad & x_i - \frac{1}{n} \leq \frac{k_2 - k_1}{k_1 n} && (6.1) \\
& c_i \geq k_1, \quad \sum_{i \in I} c_i \leq k_2 n \\
\Rightarrow \quad & x_i \geq \frac{k_1}{k_2 n} \\
\Rightarrow \quad & x_i - \frac{1}{n} \geq \frac{-(k_2 - k_1)}{k_2 n} && (6.2)
\end{aligned}$$

From 6.1 and 6.2

$$\begin{aligned}
& \left| x_i - \frac{1}{n} \right| \leq \max\left(\left| \frac{k_2 - k_1}{k_1 n} \right|, \left| \frac{-(k_2 - k_1)}{k_2 n} \right| \right) \\
\Rightarrow \quad & \left| x_i - \frac{1}{n} \right| \leq \frac{k_2 - k_1}{k_1 n} \\
\Rightarrow \quad & \sum_{i \in \mathcal{I}} \left| x_i - \frac{1}{n} \right| \leq \frac{k_2 - k_1}{k_1} \\
\Rightarrow \quad & D_I(x) = \frac{1}{2} \sum_{i \in \mathcal{I}} \left| x_i - \frac{1}{n} \right| \leq \frac{k_2 - k_1}{2k_1}
\end{aligned}$$

□

This bound clearly weakens with increase in k_2. To study the effect of k_1, consider $f(k_1) = \frac{k_2-k_1}{2k_1} \Rightarrow f'(k_1) = \frac{2k_1(-1)-(k_2-k_1)(2)}{4k_1^2} = \frac{-k_2}{2k_1^2} < 0$. Thus $f(k_1)$ decreases with increase in k_1. Hence, increasing k_1 strengthens the bound. Note that this bound becomes useless when $k_1 = 0$. Since it requires k_1 and k_2 to be very close, it is not of much practical use. In particular, forcing $k_1 > 0$, that is, ensuring that each item gets recommended at least once is particularly difficult, both in cases when an item is unpopular (in which case we would not even want it to be recommended) and in cold start cases when we do not have enough ratings to determine the quality of an item.

A Stronger Bound

The previous bound assumes $c_i \geq k_1$. But this is a poor assumption in practice and, in the case of the min cost flow method, often results in no solution to the resulting min cost flow problem. We would like to derive a bound on intersection distance, only given that $c_i \leq k \ \forall \ i$. Let $\mathcal{U}$ be the set of users, $\mathcal{I}$ be the set of items and let N items be desired in each user's recommendation list. Also assume $|\mathcal{I}| = n$.

Lemma 5. *The intersection distance for a recommender system that satisfies $c_i \leq k \ \forall \ i \in \mathcal{I}$ is bounded as follows -*

$$D_I(x) \leq \begin{cases} 1 - \frac{1}{2}\left(\frac{N|\mathcal{U}|}{nk} - \frac{n}{N|\mathcal{U}|}\right) & , N|\mathcal{U}| \geq n \textit{ and } k < 1 + \frac{2N|\mathcal{U}|}{n} \\ 1 & , \textit{otherwise} \end{cases}$$

Proof. Since the sum of the number of times each item is recommended must equal the total number of recommendations required across all users, we have $\sum_{i=1}^{n} c_i = N|\mathcal{U}| \ \Rightarrow x_i \leq \frac{k}{N|\mathcal{U}|}$. Hence,

$$x_i - \frac{1}{n} \leq \frac{k}{N|\mathcal{U}|} - \frac{1}{n} \tag{6.3}$$

Let the maximum number of items i with $c_i = 0$ be m. Hence, for $(n - m)$ items, $c_i > 0$. Also, for all these, $c_i \leq k$. Hence, these items can account for at most $k(n - m)$ recommendations. But $\sum_{i=1}^{n} c_i = N|\mathcal{U}| \ \Rightarrow N|\mathcal{U}| \leq k(n - m) \Rightarrow m \leq n - \frac{N|\mathcal{U}|}{k}$.

Let $Z = \{i \in \mathcal{I} : c_i = 0\}$. Then $|Z| \leq m \leq n - \frac{N|\mathcal{U}|}{k}$.
Also, let $Y = \{i \in \mathcal{I} : c_i > 0\}$.

$\forall \, i \in Z, c_i = 0 \Rightarrow x_i = 0$.
$\forall \, i \in Z, c_i \geq 1 \Rightarrow x_i \geq \frac{1}{N|\mathcal{U}|}$.

$$\Rightarrow x_i - \frac{1}{n} \geq \frac{1}{N|\mathcal{U}|} - \frac{1}{n} \Rightarrow -\left(x_i - \frac{1}{n}\right) \leq -\left(\frac{1}{N|\mathcal{U}|} - \frac{1}{n}\right) \tag{6.4}$$

If $x \leq a$ and $-x \leq b$ then $|x| \leq max(|a|, |b|)$. Hence, from equations 6.3 and 6.4,

$$\left|x_i - \frac{1}{n}\right| \leq \max\left(\left|\frac{k}{N|\mathcal{U}|} - \frac{1}{n}\right|, \left|-\left(\frac{1}{N|\mathcal{U}|} - \frac{1}{n}\right)\right|\right) \tag{6.5}$$

We know that $k \geq \frac{N|\mathcal{U}|}{n}$ for a solution to exist for the min cost flow problem. Hence,

$$\frac{k}{N|\mathcal{U}|} - \frac{1}{n} \geq 0 \Rightarrow \left|\frac{k}{N|\mathcal{U}|} - \frac{1}{n}\right| = \frac{k}{N|\mathcal{U}|} - \frac{1}{n} \tag{6.6}$$

Suppose $N|\mathcal{U}| \leq n$

$$\Rightarrow \frac{1}{N|\mathcal{U}|} - \frac{1}{n} \geq 0 \Rightarrow \left|\frac{1}{N|\mathcal{U}|} - \frac{1}{n}\right| = \frac{1}{N|\mathcal{U}|} - \frac{1}{n}$$

Also,

$$\left|\frac{k}{N|\mathcal{U}|} - \frac{1}{n}\right| = \frac{k}{N|\mathcal{U}|} - \frac{1}{n} \geq \frac{1}{N|\mathcal{U}|} - \frac{1}{n} = \left|\frac{1}{N|\mathcal{U}|} - \frac{1}{n}\right| \tag{6.7}$$

Suppose $N|\mathcal{U}| > n$.

$$\Rightarrow \frac{1}{N|\mathcal{U}|} - \frac{1}{n} < 0 \Rightarrow \left|\frac{1}{N|\mathcal{U}|} - \frac{1}{n}\right| = \frac{1}{n} - \frac{1}{N|\mathcal{U}|}$$

Consider the condition for,

$$\frac{1}{n} - \frac{1}{N|\mathcal{U}|} > \frac{k}{N|\mathcal{U}|} - \frac{1}{n} \quad \Rightarrow \quad \frac{2}{n} > \frac{k-1}{N|\mathcal{U}|} \quad \Rightarrow \quad k < 1 + \frac{2N|\mathcal{U}|}{n} \tag{6.8}$$

From 6.5, 6.6, 6.7 and 6.8,

$$\left|x_i - \frac{1}{n}\right| \leq \begin{cases} \frac{1}{n} - \frac{1}{N|\mathcal{U}|} & , N|\mathcal{U}| > n \text{ and } k < 1 + \frac{2N|\mathcal{U}|}{n} \\ \frac{k}{N|\mathcal{U}|} - \frac{1}{n} & \text{, otherwise} \end{cases}$$

Now, intersection distance is given by,

$$\begin{aligned}
D_I(x) &= \frac{1}{2}\sum_{i=1}^{n}\left|x_i - \frac{1}{n}\right| = \frac{1}{2}\left(\sum_{i\in Z}\left|x_i - \frac{1}{n}\right| + \sum_{i\in Y}\left|x_i - \frac{1}{n}\right|\right) \\
&= \frac{1}{2}\left(\sum_{i\in Z}\frac{1}{n} + \sum_{i\in Y}\left|x_i - \frac{1}{n}\right|\right) = \frac{1}{2}\left(\frac{|Z|}{n} + \sum_{i\in Y}\left|x_i - \frac{1}{n}\right|\right) \\
&\leq \frac{1}{2}\left(\frac{1}{n}\left(n - \frac{N|\mathcal{U}|}{k}\right) + \sum_{i\in Y}\left|x_i - \frac{1}{n}\right|\right) \\
&\leq \frac{1}{2}\left(1 - \frac{N|\mathcal{U}|}{k} + \sum_{i\in Y} B\right) \qquad \left(\text{if } \left|x_i - \frac{1}{n}\right| \leq B \;\forall\, i \in \mathcal{I}\right) \\
&= \frac{1}{2}\left(1 - \frac{N|\mathcal{U}|}{k} + B|Y|\right) \leq \frac{1}{2}\left(1 - \frac{N|\mathcal{U}|}{k} + Bn\right) \qquad (\text{as } |Y| \leq n)
\end{aligned}$$

Substituting the bound B, we get,

$$D_I(x) \leq \begin{cases} 1 - \frac{1}{2}\left(\frac{N|\mathcal{U}|}{nk} - \frac{n}{N|\mathcal{U}|}\right) & , N|\mathcal{U}| > n \text{ and } k < 1 + \frac{2N|\mathcal{U}|}{n} \\ \frac{1}{2}\left(\frac{nk}{N|\mathcal{U}|} - \frac{N|\mathcal{U}|}{nk}\right) & , \text{otherwise} \end{cases}$$

The second bound can occasionally become too weak. It is desirable to remove the redundant cases when it exceeds 1.

$$\frac{1}{2}\left(\frac{nk}{N|\mathcal{U}|} - \frac{N|\mathcal{U}|}{nk}\right) > 1 \quad \Rightarrow \quad \frac{nk}{N|\mathcal{U}|} > 2 + \frac{N|\mathcal{U}|}{nk}$$

But for a feasible solution,

$$k > 1 + \frac{2N|\mathcal{U}|}{n} \Rightarrow \frac{nk}{N|\mathcal{U}|} > \frac{n}{N|\mathcal{U}|} + 2$$

Hence this bound will always be greater than 1. In this case, it is better to use the tighter bound of 1. Hence,

$$D_I(x) \leq \begin{cases} 1 - \frac{1}{2}\left(\frac{N|\mathcal{U}|}{nk} - \frac{n}{N|\mathcal{U}|}\right) & , N|\mathcal{U}| \geq n \text{ and } k < 1 + \frac{2N|\mathcal{U}|}{n} \\ 1 & , \text{otherwise} \end{cases}$$

□

Consider the bound $1 - \frac{1}{2}\left(\frac{N|\mathcal{U}|}{nk} - \frac{n}{N|\mathcal{U}|}\right)$. As k increases, $\frac{N|\mathcal{U}|}{nk}$ decreases so $1 - \frac{1}{2}\left(\frac{N|\mathcal{U}|}{nk} - \frac{n}{N|\mathcal{U}|}\right)$ increases. Thus, the bound worsens with increase in k.

It can be seen from experiments that the intersection distance for the min cost flow method does not fall below 0.5 for the examined range of parameter values. It can be shown that this bound never attains a value less than 0.5.

Lemma 6. *For all recommender systems,* $1 - \frac{1}{2}\left(\frac{N|\mathcal{U}|}{nk} - \frac{n}{N|\mathcal{U}|}\right) \geq 0.5$.

Proof. Suppose for some parameter value, the bound is lower than 0.5. Then,

$$1 - \frac{1}{2}\left(\frac{N|\mathcal{U}|}{nk} - \frac{n}{N|\mathcal{U}|}\right) < \frac{1}{2} \Rightarrow \frac{N|\mathcal{U}|}{nk} - \frac{n}{N|\mathcal{U}|} > 1$$

$$\Rightarrow \quad \frac{N|\mathcal{U}|}{nk} > 1 + \frac{n}{N|\mathcal{U}|} \Rightarrow k < \frac{N|\mathcal{U}|}{n\left(1 + \frac{n}{N|\mathcal{U}|}\right)} = \frac{(N|\mathcal{U}|)^2}{n\,(n + N|\mathcal{U}|)}$$

But for a feasible solution, $k \geq \frac{N|\mathcal{U}|}{n}$. Hence,

$$\frac{N|\mathcal{U}|}{n} < \frac{(N|\mathcal{U}|)^2}{n\,(n+N|\mathcal{U}|)} \quad \Rightarrow \quad 1 < \frac{N|\mathcal{U}|}{n+N|\mathcal{U}|} \quad \Rightarrow \quad n+N|\mathcal{U}| < N|\mathcal{U}| \quad \Rightarrow \quad n<0$$

which is not possible. This, this bound is not good enough to find a parameter value which is guaranteed to result in an intersection distance less than 0.5. □

6.2.3 Experiments

This method was implemented using a min cost flow solver from Google OR tools [1]. Since it also requires ratings from base recommendation systems, implementations from Apache Mahout [2] were used for the same. Experiments have been conducted on the Movielens and Netflix datasets. The method has been implemented in two ways -

- MinCostFlow(k_1, k_2) - This creates a flow problem in which the flow network has an edge from every item to every user. Edge costs used to ensure that items of higher rating are preferentially assigned to a user. The design of the problem ensures that N items are recommended to each user and each item gets recommended $\geq k_1$ and $\leq k_2$ times.
- MinCostFlowThreshold(k_1, k_2, r_0) - This is like MinCostFlow except that in the flow network, there is an arc from item i to user u only if $\hat{r}_{ui} \geq r_0$. This ensures that the average predicted rating of the system is at least r_0 (and will in practice be much higher than r_0).

We have already discussed that a necessary condition for the problem to have a feasible solution is that $k_2 \geq \frac{N|\mathcal{U}|}{|\mathcal{I}|}$. For the movielens dataset, as $|\mathcal{U}| = 943$ and $|\mathcal{I}| = 1682$, we get $k_2 \geq 6$ for $N = 10$ and $k \geq 12$ for $N = 20$. Also, as we have $N|\mathcal{U}| > n$ for $N > 1$, we can use the bound in 6.2.2 to estimate bounds on the results. For $N = 10$, $k \geq 6$ for a solution to exist. Thus, the tightest bound is for $k = 6$, which is 0.62. For $N = 20$, $k \geq 12$ for a solution to exist. Thus, the tightest bound is for $k = 12$, which is 0.577.

The existence of a solution for MinCostFlowThreshold also depends on the threshold chosen. As the threshold is increased, the number of edges decreases and, if other parameters are kept constant, the probability of the existence of a feasible solution decreases. For this range of values of k_1 and k_2, the problem is observed to have no solution for thresholds of 4 and 3.5. The results of the experiment are presented in Appendix B.The reported results are for a threshold of 3 and for those pairs (k_1, k_2), $k_1 \in \{0, 1\}$, $k_2 \in \{1, 2, \ldots 10\}$ which have solutions.

[1] http://code.google.com/p/or-tools

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

A sample of the variation of intersection distance and average predicted rating with the parameter k_2 are shown in Figure 6.1.

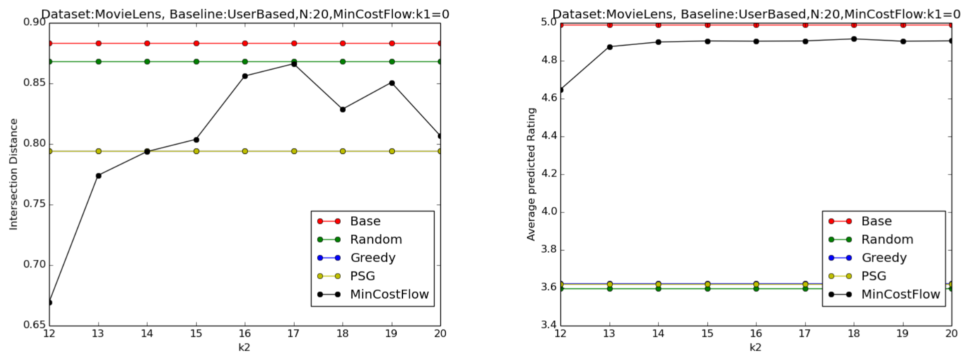

Figure 6.1: Dataset=MovieLens, Baseline=UserBased, N=20

The plots showing the behaviour in other settings can be found in Appendix D. In the following subsections, we present the best results obtained for both datasets.

Results on the MovieLens Dataset

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.992	13.0	4.994	0.992	13.0	4.994
Random	0.953	79.0	4.448	0.951	82.0	4.441
Pseudo Gradient Descent	0.944	100.0	4.449	0.898	177.0	4.274
Greedy	0.944	100.0	4.449	0.944	100.0	4.449
Min Cost Flow:0-6	0.621	638.0	4.002	-	-	-
Min Cost Flow:0-12	0.826	293.0	4.843	0.675	546.0	4.501
Min Cost Flow Threshold:0-7	0.703	500.0	4.416	-	-	-
Min Cost Flow Threshold:0-12	0.832	282.0	4.851	0.672	551.0	4.491

Table 6.1: Min cost flow bounding method on MovieLens : Baseline - ItemAverage

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.992	13.0	4.994	0.992	13.0	4.994
Random	0.952	80.0	4.456	0.952	80.0	4.439
Pseudo Gradient Descent	0.944	100.0	4.449	0.898	177.0	4.274
Greedy	0.944	100.0	4.449	0.944	100.0	4.449
Min Cost Flow:0-6	0.621	638.0	4.002	-	-	-
Min Cost Flow:0-12	0.826	293.0	4.843	0.675	546.0	4.501
Min Cost Flow Threshold:0-7	0.703	500.0	4.416	-	-	-
Min Cost Flow Threshold:0-12	0.832	282.0	4.851	0.672	551.0	4.491

Table 6.2: Min cost flow bounding method on MovieLens : Baseline - ItemUserAverage

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.833	281.0	5.0	0.833	281.0	5.0
Random	0.706	495.0	4.998	0.707	492.0	4.997
Pseudo Gradient Descent	0.539	966.0	4.996	0.472	1218.0	4.914
Greedy	0.539	966.0	4.996	0.539	966.0	4.996
Min Cost Flow:0-6	0.603	668.0	4.518	-	-	-
Min Cost Flow:0-12	0.704	498.0	4.544	0.685	529.0	4.51
Min Cost Flow Threshold:0-7	0.647	594.0	4.571	-	-	-
Min Cost Flow Threshold:0-12	0.716	477.0	4.544	0.702	502.0	4.583

Table 6.3: Min cost flow bounding method on MovieLens : Baseline - ItemBased

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.883	196.0	3.875	0.883	196.0	3.875
Random	0.874	212.0	3.61	0.869	221.0	3.619
Pseudo Gradient Descent	0.794	346.0	3.621	0.758	407.0	3.585
Greedy	0.794	346.0	3.624	0.794	346.0	3.624
Min Cost Flow:0-6	0.61	656.0	3.463	-	-	-
Min Cost Flow:0-12	0.846	259.0	4.907	0.669	556.0	4.648
Min Cost Flow Threshold:0-7	0.747	426.0	4.39	-	-	-
Min Cost Flow Threshold:0-12	0.844	263.0	4.896	0.671	554.0	4.645

Table 6.4: Min cost flow bounding method on MovieLens : Baseline - UserBased

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.894	179.0	4.678	0.894	179.0	4.661
Random	0.848	256.0	4.396	0.839	270.0	4.386
Pseudo Gradient Descent	0.778	460.0	4.412	0.731	647.0	4.263
Greedy	0.779	465.0	4.4	0.783	463.0	4.389
Min Cost Flow:0-6	0.605	665.0	3.711	-	-	-
Min Cost Flow:0-12	0.783	365.0	4.141	0.677	543.0	4.076

Table 6.5: Min cost flow bounding method on MovieLens : Baseline - ALSWR

Results on the Netflix Dataset

Due to the prohibitive run time on the Netflix dataset, only the smallest possible values of the parameter k_2 were tested as it was found that this results in the best possible value of intersection distance.

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.986	16.0	4.261	0.986	16.0	4.261
Random	0.942	65.0	4.029	0.942	66.0	4.029
Greedy	0.941	68.0	4.03	0.941	68.0	4.03
Pseudo Gradient Descent	0.941	68.0	4.03	0.891	130.0	3.911
Min Cost Flow:0-37	0.207	1070.0	3.917	-	-	-
Min Cost Flow:0-73	-	-	-	0.429	782.0	4.278

Table 6.6: Min cost flow bounding method on Netflix : Baseline - ItemAverage

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.986	16.0	4.261	0.986	16.0	4.261
Random	0.942	67.0	4.03	0.941	66.0	4.03
Greedy	0.941	68.0	4.03	0.941	68.0	4.03
Pseudo Gradient Descent	0.941	68.0	4.03	0.891	130.0	3.911
Min Cost Flow:0-37	0.207	1070.0	3.917	-	-	-
Min Cost Flow:0-73	-	-	-	0.429	782.0	4.278

Table 6.7: Min cost flow bounding method on Netflix : Baseline - ItemUserAverage

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.784	324.0	4.596	0.784	324.0	4.596
Random	0.605	602.0	4.044	0.609	590.0	4.041
Greedy	0.588	844.0	4.04	0.588	844.0	4.04
Pseudo Gradient Descent	0.588	844.0	4.04	0.459	1050.0	3.855
Min Cost Flow:0-37	0.199	1069.0	4.218	-	-	-
Min Cost Flow:0-73	-	-	-	0.401	837.0	4.359

Table 6.8: Min cost flow bounding method on Netflix : Baseline - ItemBased

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.671	491.0	3.914	0.671	491.0	3.914
Random	0.642	536.0	3.45	0.639	532.0	3.43
Greedy	0.635	610.0	3.43	0.635	610.0	3.43
Pseudo Gradient Descent	0.635	610.0	3.43	0.635	610.0	3.43
Min Cost Flow:0-37	0.194	1089.0	3.735	-	-	-
Min Cost Flow:0-73	-	-	-	0.403	842.0	4.504

Table 6.9: Min cost flow bounding method on Netflix : Baseline - UserBased

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.85	226.0	4.23	0.855	223.0	4.229
Random	0.78	314.0	4.035	0.781	312.0	4.034
Greedy	0.776	431.0	4.034	0.78	418.0	4.035
Pseudo Gradient Descent	0.772	429.0	4.038	0.721	548.0	3.921
Min Cost Flow:0-37	0.199	1070.0	3.543	-	-	-
Min Cost Flow:0-73	-	-	-	0.432	729.0	3.702

Table 6.10: Min cost flow bounding method on Netflix : Baseline - ALSWR

6.2.4 Observations and Discussion

In the MovieLens dataset, we observe that the improvement over the heuristics is not significant. Although the best results of this method do improve over the heuristics, it is by a very small margin. On some baseline recommenders, the APR is much better than the heuristics, improving almost by 20% but on others, the heristics perform better. An interesting point is that it is on the simpler recommenders (which primarily base their rating predcitions on averages) on which the performance is poor.

We observe that keeping k_1 constant, intersection distance seems to more or less increase with k_2. This is in line with the fact that the bound proved for intersection distance weakens with increase in k_2. The best results are obtained with the smallest possible parameter values. We also see that keeping k_2 constant, the performance for $k_1 = 1$ is worse than that of $k_1 = 0$. This is a little difficult to explain because it appears that the former would force an increase in coverage, but would also recommend a lot of items of poor relevance to attain this. For $k_1 > 1$, we cannot even obtain a feasible solution for small values for k_2.

A positive point is that the average predicted rating is always above 4, even when no threshold is used. The parameter values that result in the best intersection distance have a relatively low APR of about 4.2, which is beaten by pseudo gradient descent and greedy at 4.4, but other parameter values achieve an APR of as high as 4.7-4.8. This indicates that this method does not compromise too much on relevance. Also, the tuning of the parameters can be viewed as choosing the approprate relevance and diversity requirement.

We also observe that for the same parameter values, using a threshold does not necessarily improve the APR. This is possible because the systems are evaluated only on a sample of users. If the same optimum value to the min cost flow problem is obtained with and without the threshold, but with the sampled users having lower ratings, then it is possible that the performance of the method with a threshold is worse. Since the APR without the use of a threshold exceeds the set threshold of 3, and we are unable to find feasible solutions for larger thresholds, it appears that the threshold is not particularly useful. For ALSWR, even at a threshold of 3, there is no solution.

In the Netflix dataset, there is a dramatic improvement in intersection distance on using this method, with intersection distances falling as low as 0.1-0.2 without a significant drop in APR. Coverage is also significantly higher. On the downside, in many cases, the APR is lower than other methods almost by a magnitude of 0.5. The exception is when the baseline is ItemBased, where it actually performs better than all except the baseline.

We would expect that on increasing N, the intersection distance would be lowered, as there is more scope for improving diversity. However, in the case of the min cost flow problem, it actually increases the difficulty of finding a solution, hence it is actually possible for the intersection distance to worsen, or for there to be no solution to the problem. Both these cases are seen in the experimental results.

There are, however, some disadvantages to this method. Unlike the pseudo-gradient descent and greedy methods, this method precomputes recommendation lists for all users. That is, it is an offline system. It may be cumbersome to update recommendation lists frequently using new information. Further, since there is a need to transfer the complete predicted rating matrix from the baseline recommenders to the min cost flow solver, the system scales very poorly in the size of the matrix.

Also, the intersection distance obtained by randomly sampling a few users need not be the same as that obtained by considering all users. However, the value on a random sample is more useful to compare with the performance of other systems because in reality, not all users use a recommedation system with the same frequency.

6.3 Joint Optimization of Relevance and Diversity

Instead of performing a secondary minimization of intersection distance by bounding the number of times each item gets recommended, we would like to directly minimize intersection distance. The following formulation attempts to optimize a weighted combination of total rating of recommended items and the unnormalized intersection distance.

6.3.1 Formulation

Consider the minimum cost flow problem, shown in figure 6.2, on the flow network $G = (V, E)$ with $V = \mathcal{U} \cup \mathcal{I} \cup \{S, D\}$, where $\mathcal{U}$ is the set of users, $\mathcal{I}$ is the set of items, and S and D are two special nodes.

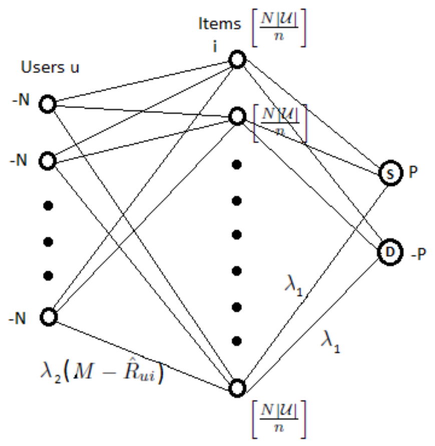
Items
$\left\lceil \frac{N|\mathcal{U}|}{n} \right\rceil$
i
Users u
-N
-N
-N
$\left\lceil \frac{N|\mathcal{U}|}{n} \right\rceil$
S
P
D
-P
λ_1
λ_1
$\lambda_2(M - \hat{R}_{ui})$
$\left\lceil \frac{N|\mathcal{U}|}{n} \right\rceil$

Figure 6.2: Min cost flow problem for minimizing a weighted combination of total rating and unnormalized intersection distance

The balances at the nodes are as follows

$$b(v) = \begin{cases} -N & , v \in \mathcal{U} \\ \left\lceil \frac{N|\mathcal{U}|}{n} \right\rceil & , v \in \mathcal{I} \\ P & , v = S \\ -P & , v = D \end{cases} \tag{6.9}$$

where P is a large constant.

There is an edge from every item $i \in \mathcal{I}$ to every user $u \in \mathcal{U}$. Also, there is an edge from S to every item i, from every item i to D and one from S to D. The costs of the edges are as follows -

$$c(x, y) = \begin{cases} \lambda_2(M - \hat{R}_{ui}) & , x \in \mathcal{I} \text{ and } y \in \mathcal{U} \\ \lambda_1 & , x = S \text{ and } y \in \mathcal{I} \\ \lambda_1 & , x \in \mathcal{I} \text{ and } y = D \\ 0 & , x = S \text{ and } y = D \end{cases} \tag{6.10}$$

where $\lambda_1 > 0$ and $\lambda_2 > 0$ are tunable parameters and M is a constant such that $M > \hat{R}_{ui} \, \forall \, u \in \mathcal{U}, \; i \in \mathcal{I}$.

The capacities of the edges are as follows -

$$l(x,y) = \begin{cases} 1 & , x \in \mathcal{I} \text{ and } y \in \mathcal{U} \\ \infty & , x = S \text{ and } y \in \mathcal{I} \\ \infty & , x \in \mathcal{I} \text{ and } y = D \\ \infty & , x = S \text{ and } y = D \end{cases} \tag{6.11}$$

In any feasible integral flow in this network, the flow $X(i, u)$ can take only values 0 and 1 on an edge from item i to user u. If $X(i, u) = 1$, then item i is recommended to user u and otherwise it is not. Since each user has a demand of N and no outgoing edges, exactly N items will be recommended to each user. Also, since each edge from an item to a user has a capacity of 1, an item can be recommended to a user only once. Thus, any solution to the problem is a valid set of recommendation lists.

Further, for each edge (i, u) with a non-zero flow, a cost of $\lambda_2(M - \hat{R}_{ui})$ is added to the objective. Summing this over all such edges, we get $\lambda_2 \left(MN|\mathcal{U}| - \sum\limits_{u} \sum\limits_{i \in L_N(u)} \hat{R}_{ui} \right)$. Minimizing this term is equivalent to maximizing the total predicted rating of the list.

Also, consider any item i for which $c_i > \left[\frac{N|\mathcal{U}|}{n}\right]$. Since, the net outgoing flow from i is c_i, and its supply is $\left[\frac{N|\mathcal{U}|}{n}\right]$, the flow $X(S, i) = c_i - \left[\frac{N|\mathcal{U}|}{n}\right]$, so the edge (S, i) contributes a cost of $\lambda_1 \left(c_i - \left[\frac{N|\mathcal{U}|}{n}\right]\right)$. Similarly, if $c_i < \left[\frac{N|\mathcal{U}|}{n}\right]$, for the supply at i to be fully used up, a flow of $\left[\frac{N|\mathcal{U}|}{n}\right] - c_i$ must be present along edge (i, D). Thus, this contributes a cost of $\lambda_1 \left(\left[\frac{N|\mathcal{U}|}{n}\right] - c_i\right)$. There will also be a flow along (S, D) for a feasible solution but it can be ignored as this edge has zero cost and infinite capacity. Thus, each item contributes an additional $\lambda_1 \left| c_i - \left[\frac{N|\mathcal{U}|}{n}\right] \right|$ to the total cost. Minimizing this is approximately equivalent to minimizing the intersection distance because

$$\begin{aligned} \lambda_1 \left| c_i - \frac{N|\mathcal{U}|}{n} \right| &= \frac{\lambda_1}{N|\mathcal{U}|} \left| \frac{c_i}{N|\mathcal{U}|} - \frac{1}{n} \right| \\ \Rightarrow \sum_i \lambda_1 \left| c_i - \frac{N|\mathcal{U}|}{n} \right| &= \frac{\lambda_1}{N|\mathcal{U}|} \sum_i \left| \frac{c_i}{N|\mathcal{U}|} - \frac{1}{n} \right| \\ &= \frac{\lambda_1}{N|\mathcal{U}|} \sum_i \left| x_i - \frac{1}{n} \right| \\ &= \frac{2\lambda_1}{N|\mathcal{U}|} D_I(x) \propto D_I(x) \quad (\text{as } \lambda > 0) \end{aligned}$$

It has been assumed that either there is flow on the edge (S, i) or on (i, D) but not both. Since there is an edge (S, D) of cost 0, in any optimal solution, the above holds. But for a general flow, there may be redundant flow from S to i that completely flows back to D, in addition to the flow already accounted for. Then, for each item i, this redundant flow, say w_i, contributes to a cost of $2\lambda_1 w_i$.

Hence, the complete objective being minimized is

$$\begin{aligned} Z &= \lambda_2 \left(MN|\mathcal{U}| - \sum_u \sum_{i \in L_N(u)} \hat{R}_{ui} \right) \\ &\quad + \lambda_1 \sum_i \left| c_i - \left\lceil \frac{N|\mathcal{U}|}{n} \right\rceil \right| + 2\lambda_1 \sum_i w_i \end{aligned}$$

Since in an optimal solution, $w_i = 0 \; \forall \; i$, minimizing this objective is approximately equivalent to minimizing a weighted combination of the total rating of the lists (relevance) and the intersection distance, with the parameters λ_1 and λ_2 used to determine the relative importance of each.

6.3.2 Experiments

This method was also tested using the min cost flow solver from Google OR tools and the base recommendation systems available in Apache Mahout. The following are the results on the MovieLens dataset. Double Obj:l_1-l_2 refers to an experiment with $\lambda_1 = l_1$ and $\lambda_2 = l_2$. The best results from earlier methods have been included for the sake of comparison. A graphical representation of the same is available in Appendix E.

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.993	12.2	4.996	0.983	29.2	4.741
Random	0.952	81.0	4.461	0.905	161.6	4.278
Pseudo Gradient Descent	0.944	100.0	4.449	0.898	177.0	4.274
Greedy	0.945	101.6	4.451	0.9	179.0	4.275
Min Cost Flow:0-6	0.621	638.0	4.002	-	-	-
Min Cost Flow:0-12	0.826	293.0	4.843	0.675	546.0	4.501
Double Obj:1-1	0.549	759.0	3.062	0.357	1160.0	3.113
Double Obj:1-2	0.605	664.2	4.358	0.396	1092.8	3.966
Double Obj:1-3	0.646	595.8	4.469	0.646	644.0	4.502
Double Obj:1-4	0.733	449.4	4.627	0.686	571.0	4.558
Double Obj:1-5	0.757	409.2	4.518	0.695	559.2	4.552
Double Obj:1-6	0.753	415.0	4.557	0.7	551.2	4.714
Double Obj:1-7	0.743	432.2	4.614	0.684	572.4	4.648
Double Obj:1-8	0.746	427.4	4.574	0.689	569.2	4.59
Double Obj:1-9	0.729	456.4	4.626	0.687	571.6	4.555
Double Obj:1-10	0.742	434.6	4.609	0.681	578.2	4.662

Table 6.11: Min cost flow double objective method on MovieLens : Baseline - ItemAverage

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.993	12.2	4.996	0.983	29.2	4.741
Random	0.95	83.4	4.452	0.904	164.0	4.275
Pseudo Gradient Descent	0.944	100.0	4.449	0.898	177.0	4.274
Greedy	0.945	101.6	4.451	0.9	179.0	4.275
Min Cost Flow:0-6	0.621	638.0	4.002	-	-	-
Min Cost Flow:0-12	0.826	293.0	4.843	0.675	546.0	4.501
Double Obj:1-1	0.549	759.0	3.062	0.357	1160.0	3.113
Double Obj:1-2	0.605	664.2	4.358	0.396	1092.8	3.966
Double Obj:1-3	0.646	595.8	4.469	0.646	644.0	4.502
Double Obj:1-4	0.733	449.4	4.627	0.686	571.0	4.558
Double Obj:1-5	0.757	409.2	4.518	0.695	559.2	4.552
Double Obj:1-6	0.753	415.0	4.557	0.7	551.2	4.714
Double Obj:1-7	0.743	432.2	4.614	0.684	572.4	4.648
Double Obj:1-8	0.746	427.4	4.574	0.689	569.2	4.59
Double Obj:1-9	0.729	456.4	4.626	0.687	571.6	4.555
Double Obj:1-10	0.742	434.6	4.609	0.681	578.2	4.662

Table 6.12: Min cost flow double objective method on MovieLens : Baseline - ItemUserAverage

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.817	308.2	5.0	0.696	536.6	5.0
Random	0.679	539.6	4.994	0.543	811.6	4.914
Pseudo Gradient Descent	0.539	966.0	4.996	0.472	1218.0	4.914
Greedy	0.527	1009.6	4.993	0.465	1229.4	4.915
Min Cost Flow:0-6	0.603	668.0	4.518	-	-	-
Min Cost Flow:0-12	0.704	498.0	4.544	0.685	529.0	4.51
Double Obj:1-1	0.549	759.0	3.153	0.357	1160.0	3.197
Double Obj:1-2	0.561	738.0	4.552	0.377	1125.4	4.556
Double Obj:1-3	0.549	759.4	4.549	0.376	1128.2	4.551
Double Obj:1-4	0.554	750.4	4.545	0.374	1129.8	4.526
Double Obj:1-5	0.552	753.4	4.553	0.381	1115.8	4.532
Double Obj:1-6	0.557	744.6	4.554	0.376	1126.0	4.526
Double Obj:1-7	0.557	745.4	4.544	0.372	1130.6	4.554
Double Obj:1-8	0.553	752.6	4.554	0.378	1124.6	4.542
Double Obj:1-9	0.556	746.8	4.534	0.379	1123.4	4.543
Double Obj:1-10	0.559	741.8	4.544	0.377	1127.0	4.542

Table 6.13: Min cost flow double objective method on MovieLens : Baseline - ItemBased

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.863	229.6	3.948	0.819	305.0	3.771
Random	0.845	261.4	3.664	0.799	338.0	3.621
Pseudo Gradient Descent	0.794	346.0	3.621	0.758	407.0	3.585
Greedy	0.771	395.4	3.664	0.745	457.4	3.623
Min Cost Flow:0-6	0.61	656.0	3.463	-	-	-
Min Cost Flow:0-12	0.846	259.0	4.907	0.669	556.0	4.648
Double Obj:1-1	0.549	759.0	0.239	0.357	1160.0	0.297
Double Obj:1-2	0.629	624.2	4.629	0.512	881.8	4.439
Double Obj:1-3	0.656	578.0	4.725	0.563	789.8	4.597
Double Obj:1-4	0.7	505.2	4.845	0.641	649.4	4.71
Double Obj:1-5	0.693	515.6	4.839	0.642	648.6	4.709
Double Obj:1-6	0.709	489.0	4.842	0.637	656.4	4.71
Double Obj:1-7	0.7	505.2	4.843	0.63	665.8	4.707
Double Obj:1-8	0.696	511.6	4.842	0.632	662.0	4.709
Double Obj:1-9	0.701	503.6	4.844	0.641	649.2	4.707
Double Obj:1-10	0.708	491.6	4.845	0.638	653.4	4.709

Table 6.14: Min cost flow double objective method on MovieLens : Baseline - UserBased

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.885	194.2	4.698	0.834	294.8	4.586
Random	0.833	280.4	4.427	0.758	426.4	4.286
Pseudo Gradient Descent	0.778	460.0	4.412	0.731	647.0	4.263
Greedy	0.777	480.4	4.424	0.733	648.6	4.286
Min Cost Flow:0-6	0.605	665.0	3.711	-	-	-
Min Cost Flow:0-12	0.783	365.0	4.141	0.677	543.0	4.076
Double Obj:1-1	0.549	759.0	2.953	0.357	1160.0	3.004
Double Obj:1-2	0.551	754.8	3.77	0.361	1155.4	3.705
Double Obj:1-3	0.599	674.6	3.956	0.427	1036.6	3.89
Double Obj:1-4	0.642	602.4	4.094	0.525	854.8	4.071
Double Obj:1-5	0.643	599.8	4.104	0.518	869.6	4.085
Double Obj:1-6	0.646	595.6	4.094	0.516	871.6	4.07
Double Obj:1-7	0.643	600.8	4.087	0.513	877.4	4.06
Double Obj:1-8	0.646	594.6	4.093	0.526	855.2	4.071
Double Obj:1-9	0.644	598.2	4.102	0.518	868.8	4.078
Double Obj:1-10	0.646	595.8	4.1	0.524	860.4	4.072

Table 6.15: Min cost flow double objective method on MovieLens : Baseline - ALSWR

The results on the Netflix dataset are as follows -

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.985	17.25	4.26	0.973	31.6	4.183
Random	0.941	67.25	4.03	0.891	126.8	3.909
Pseudo Gradient Descent	0.941	68.0	4.03	0.891	130.0	3.911
Greedy	0.94	71.5	4.028	0.89	136.0	3.909
Min Cost Flow:0-37	0.207	1070.0	3.917	-	-	-
Min Cost Flow:0-73	-	-	-	0.429	782.0	4.278
Double Obj:1-1	0.207	1052.5	3.135	0.14	1093.6	3.128
Double Obj:1-2	0.353	949.0	4.3	0.194	1062.8	3.837
Double Obj:1-3	0.38	903.25	4.305	0.374	940.4	4.171
Double Obj:1-4	0.613	645.25	4.606	0.7	656.6	4.485
Double Obj:1-5	0.612	644.0	4.587	0.697	659.0	4.508
Double Obj:1-6	0.618	639.5	4.612	0.701	657.0	4.463
Double Obj:1-7	0.619	637.25	4.578	0.698	655.6	4.466
Double Obj:1-8	0.621	643.25	4.538	0.696	656.6	4.497
Double Obj:1-9	0.612	643.5	4.61	0.703	659.0	4.467
Double Obj:1-10	0.622	640.75	4.556	0.699	667.0	4.472

Table 6.16: Min cost flow double objective method on Netflix : Baseline - ItemAverage

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.985	17.4	4.26	0.973	31.6	4.183
Random	0.941	67.4	4.027	0.891	124.8	3.911
Pseudo Gradient Descent	0.941	68.0	4.03	0.891	130.0	3.911
Greedy	0.941	71.0	4.028	0.89	136.0	3.909
Min Cost Flow:0-37	0.207	1070.0	3.917	-	-	-
Min Cost Flow:0-73	-	-	-	0.429	782.0	4.278
Double Obj:1-1	0.21	1055.2	3.135	0.14	1093.6	3.128
Double Obj:1-2	0.353	945.0	4.3	0.194	1062.8	3.837
Double Obj:1-3	0.384	897.0	4.311	0.374	940.4	4.171
Double Obj:1-4	0.614	643.0	4.605	0.7	656.6	4.485
Double Obj:1-5	0.612	641.8	4.587	0.697	659.0	4.508
Double Obj:1-6	0.617	637.6	4.611	0.701	657.0	4.463
Double Obj:1-7	0.618	635.0	4.578	0.698	655.6	4.466
Double Obj:1-8	0.62	641.0	4.541	0.696	656.6	4.497
Double Obj:1-9	0.611	640.8	4.609	0.703	659.0	4.467
Double Obj:1-10	0.621	638.4	4.557	0.699	667.0	4.472

Table 6.17: Min cost flow double objective method on Netflix : Baseline - ItemUserAverage

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.788	316.5	4.611	0.721	490.0	4.36
Random	0.606	595.25	4.055	0.473	889.25	3.866
Pseudo Gradient Descent	0.588	844.0	4.04	0.459	1050.0	3.855
Greedy	0.586	835.0	4.052	0.457	1053.5	3.865
Min Cost Flow:0-37	0.199	1069.0	4.218	-	-	-
Min Cost Flow:0-73	-	-	-	0.401	837.0	4.359
Double Obj:1-1	0.207	1052.5	3.363	0.137	1093.25	3.36
Double Obj:1-2	0.203	1063.0	4.243	0.144	1098.5	4.068
Double Obj:1-3	0.208	1064.0	4.251	0.14	1096.75	4.065
Double Obj:1-4	0.261	1039.75	4.425	0.236	1047.5	4.264
Double Obj:1-5	0.265	1036.5	4.427	0.25	1019.5	4.265
Double Obj:1-6	0.271	1025.5	4.429	0.222	1078.0	4.265
Double Obj:1-7	0.268	1019.0	4.426	0.226	1088.5	4.265
Double Obj:1-8	0.267	1030.25	4.427	0.249	1025.0	4.263
Double Obj:1-9	0.286	994.25	4.427	0.251	1022.5	4.265
Double Obj:1-10	0.269	1026.25	4.427	0.253	1016.0	4.266

Table 6.18: Min cost flow double objective method on Netflix : Baseline - ItemBased

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.673	487.667	3.913	0.641	597.333	3.498
Random	0.645	522.667	3.428	0.638	602.667	3.434
Pseudo Gradient Descent	0.588	844.0	4.04	0.459	1050.0	3.855
Greedy	0.638	612.0	3.434	0.638	612.0	3.434
Min Cost Flow:0-37	0.194	1089.0	3.735	-	-	-
Min Cost Flow:0-73	-	-	-	0.403	842.0	4.504
Double Obj:1-1	0.208	1053.333	0.264	0.139	1093.667	0.258
Double Obj:1-2	0.293	1010.667	4.346	0.363	1028.667	4.299
Double Obj:1-3	0.393	895.0	4.575	0.387	999.667	4.348
Double Obj:1-4	0.507	760.667	4.71	0.504	872.0	4.476
Double Obj:1-5	0.508	760.333	4.712	0.503	875.667	4.476
Double Obj:1-6	0.504	761.333	4.71	0.498	881.0	4.476
Double Obj:1-7	0.501	761.667	4.71	0.497	877.333	4.476
Double Obj:1-8	0.506	762.667	4.711	0.504	872.667	4.476
Double Obj:1-9	0.507	760.333	4.71	0.503	874.667	4.476
Double Obj:1-10	0.51	759.667	4.71	0.505	871.0	4.477

Table 6.19: Min cost flow double objective method on Netflix : Baseline - UserBased

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.852	223.6	4.249	0.823	298.2	4.172
Random	0.78	312.2	4.049	0.724	443.0	3.933
Pseudo Gradient Descent	0.772	429.0	4.038	0.721	548.0	3.921
Greedy	0.774	432.6	4.048	0.722	562.2	3.933
Min Cost Flow:0-37	0.199	1070.0	3.543	-	-	-
Min Cost Flow:0-73	-	-	-	0.432	729.0	3.702
Double Obj:1	0.21	1055.2	3.157	0.14	1093.6	3.148
Double Obj:2	0.209	1061.0	3.559	0.142	1096.6	3.495
Double Obj:3	0.248	1020.8	3.634	0.189	1068.8	3.581
Double Obj:4	0.343	908.8	3.773	0.289	974.2	3.721
Double Obj:5	0.34	908.0	3.773	0.29	970.6	3.718
Double Obj:6	0.335	910.6	3.772	0.288	967.2	3.718
Double Obj:7	0.338	909.6	3.773	0.288	970.6	3.721
Double Obj:8	0.341	909.4	3.774	0.29	968.8	3.719
Double Obj:9	0.338	915.0	3.772	0.288	969.2	3.718
Double Obj:10	0.345	908.4	3.775	0.292	972.4	3.722

Table 6.20: Min cost flow double objective method on Netflix : Baseline - ALSWR

6.3.3 Observations and Discussion

In the MovieLens dataset, we observe that the best values of intersection distance obtained outperform those of earlier methods by a noticeable margin. This is expected because we are directly minimizing the intersection distance, not a maintaining a surrogate condition that would keep intersection distance low. On the Netflix dataset however, the performance of the earlier min cost flow based method was found to be better.

On keeping λ_1 constant, if we increase λ_2, we observe that the intersection distance generally increases, but so does the average predicted rating. Thsi is because, for low values of λ_2, the intersection distance term dominates the cost of the min cost flow problem, but as it increases, the importance given to the total rating of all recommended items increases. As we can see, $\lambda_2 = 1$ sometimes results in a very poor average predicted rating, but even increasing it to 1 or 2 is sufficient to make this comparable to other methods. Also, at these values, the intersection distance is still lower than that obtained by other methods.

It is observed that at $\lambda_1 = 1$, the nature of the min cost flow solution is such that the only supply taken from the supply node is to satisfy the deficit that arises because the total demand over the user nodes is less than the total supply at the item nodes. Thus increasing λ_1 will only result in the algorithm returning either the same solution, or another optimum solution with the same value. Also, since it is the ratio $\frac{\lambda_1}{\lambda_2}$ that actually determines the relative weights, decreasing λ_1 is also not of much use. Since λ_1 and λ_2 are constrained to be integers for the min cost flow problem to have an integral optimum solution, the examined values show the best possible intersection distance attainable in this formulation.

A disadvantage of this method is that it is not possible to prove bounds on the resultant intersection distance. For instance, if the ratings are skewed in such a manner that N items have a significantly higher rating than all other items, for some values of λ_1 and λ_2, it may be optimal to recommend these N items to all users, drawing supply from the supply node. However, if this is really the nature of the dataset in question, optimizing intersection distance may not be the best objective in any case.

CHAPTER 7

Gradient Descent Methods to Optimize Intersection Distance

So far, all the methods examined take ratings predicted by a standard recommendation system and re-rank items in suitable ways so that the resultant set of recommendation lists has an improved intersection distance without much loss in relevance. It would be more desirable to modify the system so that the scores it predicts are directly indicative of both relevance and diversity. To do this, we have to choose a suitable parameterization of the system that enables us to compute a measure of both relevance and diversity and learn parameters that optimize both.

A common parameterization used in literature that appears to extend to different objectives is formulating the predicted rating as $\hat{R}_{ui} \sim \mathcal{N}(U_u^T V_i, \sigma^2)$. That is, the predicted rating of user u for item i is sampled from the normal distribution centered at the dot product of a user latent factor U_u and an item latent factor V_i, with standard deviation σ.

7.1 A Permutation-based Idea

Suppose we are generating predicted ratings using $\hat{R}_{ui} \sim \mathcal{N}(U_u^T V_i, \sigma^2)$ and to each user u, we would like to recommend the top-N items according to this rating. In this setting, we need to calculate the intersection distance as a function of U_u and V_i. Since ratings are probabilistic, ideally we should compute a distribution over the possible values of intersection distance. A simpler, but perhaps not as accurate method, would be to compute and maximize the expected value of the intersection distance, or an upper bound of the same. The following is an attempt at computing an upper bound on the expected intersection distance.

For each user u, the predicted ratings induce an ordering over items. This ordering can be viewed as some permutation of the list of all items in the system. Consider a permutation $\rho = (i_1, i_2, i_3, \ldots i_n)$. Let $Pr(\rho \mid u)$ be the probability that the permutation ρ indicates the preference ordering of items for user u. $\rho = (i_1, i_2, i_3, \ldots i_n)$ will be the preference ordering for user u if, regardless of the value of the rating for item i_n, the rating of item i_{n-1} is higher than that, the rating of item i_{n-2} is higher than that of item i_{n-1} and so on until item i_1 has the highest rating. If we assume ratings of two items for the same user are independent given the latent factor matrices U and V, we get

$$Pr(\rho \mid u) = \int\limits_{r_n} Pr(\hat{R}_{ui_n} = r_n) \left(\int\limits_{r_{n-1} \geq r_n} Pr(\hat{R}_{ui_{n-1}} = r_{n-1}) \left(\dots \int\limits_{r_2 \geq r_3} Pr(\hat{R}_{ui_2} = r_2) \left(\int\limits_{r_1 \geq r_2} Pr(\hat{R}_{ui_1} = r_1) dr_1 \right) dr_2 \dots \right) dr_{n-1} \right) dr_n$$

Consider the term

$$\begin{aligned}
&\int\limits_{r_2 \geq r_3} Pr(\hat{R}_{ui_2} = r_2) \left(\int\limits_{r_1 \geq r_2} Pr(\hat{R}_{ui_1} = r_1) dr_1 \right) dr_2 \\
&= \int\limits_{r_2 \geq r_3} \frac{1}{\sigma} \phi \left(\frac{r_2 - U_u^T V_{i_2}}{\sigma} \right) Q \left(\frac{r_2 - U_u^T V_{i_1}}{\sigma} \right) dr_2 \\
&\quad \text{(where } \phi(x) \text{ and } Q(x) \text{ are respectively the PDF of the standard} \\
&\quad \text{normal distribution and the Gaussian Q function, that is, } Q(x) = 1 - \Phi(x)) \\
&\leq \int\limits_{r_2 = r_3}^{\infty} \frac{1}{\sigma} \phi \left(\frac{r_2 - U_u^T V_{i_2}}{\sigma} \right) e^{-\frac{1}{2} \left(\frac{r_2 - U_u^T V_{i_1}}{\sigma} \right)^2} dr_2 \\
&\quad \text{(as } Q(x) \leq e^{\frac{-x^2}{2}} \text{ using Chernoff bounds)} \\
&= \int\limits_{r_2 = r_3}^{\infty} \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-1}{2\sigma^2} \left((r_2 - U_u^T V_{i_2})^2 + (r_2 - U_u^T V_{i_1})^2 \right)} dr_2
\end{aligned} \tag{7.1}$$

Using standard results on the integrals of Gaussian functions, we can prove the following lemma. The proof can be found in Appendix A.

Lemma 7.

$$I = \int\limits_{t}^{\infty} \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-1}{2\sigma^2} \left((x-\mu_1)^2 + (x-\mu_2)^2 \right)} dx \leq \sqrt{\pi}\, \phi \left(\frac{\mu}{\sqrt{2}} \right) \tag{7.2}$$

If we substitute this bound in 7.1, assuming n is odd, item pairs, i_1 and i_2, i_3 and i_4 and so on, get converted to a constant in sequence and hence get taken out of the integral. Thus, we get,

$$\begin{aligned}
Pr(\rho \mid u) &\leq \left(\sqrt{\pi}\right)^{\frac{n-1}{2}} \phi\left(\frac{\mu_{12}}{\sqrt{2}}\right) \phi\left(\frac{\mu_{34}}{\sqrt{2}}\right) \dots \phi\left(\frac{\mu_{n-2,n-1}}{\sqrt{2}}\right) \int_{r_n} Pr(\hat{R}_{ui_n} = r_n) dr_n \\
&= \left(\sqrt{\pi}\right)^{\frac{n-1}{2}} \phi\left(\frac{\mu_{12}}{\sqrt{2}}\right) \phi\left(\frac{\mu_{34}}{\sqrt{2}}\right) \dots \phi\left(\frac{\mu_{n-2,n-1}}{\sqrt{2}}\right) \\
&= \left(\sqrt{\pi}\right)^{\frac{n-1}{2}} \prod_{j=1}^{\frac{n-1}{2}} \phi\left(\frac{\mu_{2j-1,2j}}{\sqrt{2}}\right) \\
&\text{where } \mu_{2j-1,2j} = \frac{\mu_{2j-1} - \mu_{2j}}{\sigma} = \frac{1}{\sigma} U_u^T (V_{i_{2j-1}} - V_{i_{2j}})
\end{aligned}$$

Note that the permutation ρ determines which item is in which position. In future, we refer to $V_{i_j}^{(\rho)}$ as the item factor corresponding to the j^{th} item according to permutation ρ and $\mu_{2j-1,2j}^{(\rho)} = \frac{\mu_{2j-1}^{(\rho)} - \mu_{2j}^{(\rho)}}{\sigma} = \frac{1}{\sigma} U_u^T (V_{i_{2j-1}}^{(\rho)} - V_{i_{2j}}^{(\rho)})$

Let $T_k(\rho, i)$ be the event that i is in the top k items in the permutation ρ. Then,

$$\begin{aligned}
E(c_i) &= \sum_u \sum_\rho T_N(\rho, i) Pr(\rho \mid u) \\
\Rightarrow \qquad E(x_i) &= \frac{1}{N|\mathcal{U}|} \sum_u \sum_\rho T_N(\rho, i) Pr(\rho \mid u) \qquad (7.3)
\end{aligned}$$

Then, the following lemma, proved in Appendix A, can be used to bound the intersection distance

Lemma 8. *For any random variable $x \in [0, 1]$,*

$$E(|x - a|) \leq E[x] \frac{(1 - a - a^2)}{1 - a} + \frac{2a^2 - a}{1 - a} \qquad (7.4)$$

This is useful if $1 - a - a^2 \geq 0 \Rightarrow a^2 + a - 1 \leq 0 \Rightarrow \frac{-1-\sqrt{5}}{2} \leq a \leq \frac{-1+\sqrt{5}}{2}$. Consider $a = \frac{1}{n}$. Then since $a \geq 0$, $a \geq \frac{-1-\sqrt{5}}{2}$. Also, $a \leq \frac{-1+\sqrt{5}}{2} \Rightarrow n \geq \frac{2}{-1+\sqrt{5}} = 1.618$. Thus, for $n \geq 2$, the following result is valid.

Substituting $a = \frac{1}{n}$ in 7.4, we get

$$\begin{aligned}
E\left[x_i - \frac{1}{n}\right] &\leq \frac{\left(1 - \frac{1}{n} - \frac{1}{n^2}\right)}{1 - \frac{1}{n}} + \frac{\frac{2}{n^2} - \frac{1}{n}}{1 - \frac{1}{n}} \\
&= \frac{n^2 - n - 1}{n(n-1)} E[x_i] - \frac{n^2 - 2}{n(n-1)} \qquad (7.5)
\end{aligned}$$

Now, intersection distance,

$$\begin{aligned}
D_I(x) &= \frac{1}{2}\sum_{i=1}^{n}\left|x_i - \frac{1}{n}\right| \\
\Rightarrow E[D_I(x)] &\leq \frac{1}{2}\sum_{i=1}^{n}\frac{n^2-n-1}{n(n-1)}E[x_i] - \frac{1}{2}(n)\frac{n^2-2}{n(n-1)} \\
&= \frac{n^2-n-1}{2n(n-1)}\sum_{i=1}^{n}E[x_i] - \frac{n^2-2}{2(n-1)} \qquad (7.6)
\end{aligned}$$

Consider equation 7.3. Any permutation ρ adds $(\sqrt{\pi})^{\frac{n-1}{2}}\prod_{j=1}^{\frac{n-1}{2}}\phi\left(\frac{\mu^{(p)}_{2j-1,2j}}{\sqrt{2}}\right)$ to the numerator of $E[x_i]$ for $i \in \{i_1, i_2 \ldots i_N\}$. Thus, it adds the term N times to $\sum_{i=1}^{n} E[x_i]$. Thus,

$$\begin{aligned}
\sum_{i=1}^{n}E[x_i] &\leq \frac{1}{N|\mathcal{U}|}\sum_{u}\sum_{\rho}N\,(\pi)^{\frac{n-1}{4}}\prod_{j=1}^{\frac{n-1}{2}}\phi\left(\frac{\mu^{(\rho)}_{2j-1,2j}}{\sqrt{2}}\right) \\
&= \frac{(\pi)^{\frac{n-1}{4}}}{|\mathcal{U}|}\sum_{u}\sum_{\rho}\prod_{j=1}^{\frac{n-1}{2}}\phi\left(\frac{\mu^{(\rho)}_{2j-1,2j}}{\sqrt{2}}\right) \\
&= \frac{(\pi)^{\frac{n-1}{4}}}{|\mathcal{U}|}\sum_{u}\sum_{\rho}\prod_{j=1}^{\frac{n-1}{2}}\frac{1}{\sqrt{2\pi}}e^{-\frac{1}{2}\left(\frac{\mu^{(p)}_{2j-1,2j}}{\sqrt{2}}\right)^2} \\
&= \frac{(\pi)^{\frac{n-1}{4}}}{|\mathcal{U}|}\frac{1}{(2)^{\frac{n-1}{4}}(\pi)^{\frac{n-1}{4}}}\sum_{u}\sum_{\rho}e^{-\frac{1}{4\sigma^2}\sum_{j=1}^{\frac{n-1}{2}}\left(U_u^T\left(V^{(\rho)}_{i_{2j-1}}-V^{(\rho)}_{i_{2j}}\right)\left(\right)^2\right.} \\
&= \frac{1}{2^{\frac{n-1}{4}}|\mathcal{U}|}\sum_{u}\sum_{\rho}e^{-\frac{1}{4\sigma^2}\sum_{j=1}^{\frac{n-1}{2}}\left(U_u^T\left(V^{(\rho)}_{i_{2j-1}}-V^{(\rho)}_{i_{2j}}\right)\right)^2} \qquad (7.7)
\end{aligned}$$

Let B and F respectively be the bounds on $\sum_{i=1}^{n} E[x_i]$ and $E\left[D_I(x)\right]$ from equations 7.7 and 7.6 respectively. Then,

$$B = \frac{(n^2-n-1)}{2n(n-1)}F - \frac{n^2-2}{2(n-1)} \qquad (7.8)$$

To learn parameters U_u and V_i by gradient descent, we need $\frac{\partial B}{\partial U_u}$ and $\frac{\partial B}{\partial V_i}$. Now,

$$\frac{\partial B}{\partial U_u} = \frac{(n^2-n-1)}{2n(n-1)}\frac{\partial F}{\partial U_u} \qquad (7.9)$$

From the summation over all users, only the term associated with user u will have a non-zero component in the derivative with respect to U_u. To simplify 7.9, we need $\frac{\partial f}{\partial x}$ for $f(x) = \frac{1}{c}e^{-(x^T a)^2+b}$. Suppose $x = (x_1\ x_2\ \ldots\ x_n)^T$. Now,

$$\begin{aligned} \frac{\partial f}{\partial x_i} &= \frac{1}{c}e^{-(x^T a)^2+b}\,(-2a_i) \\ \Rightarrow \qquad \frac{\partial f}{\partial x} &= \frac{-2}{c}\left(e^{-(x^T a)^2+b}\right)a^T \end{aligned} \tag{7.10}$$

Using 7.10 to obtain $\frac{\partial F}{\partial U_u}$ and substituting in 7.9,

$$\begin{aligned} \frac{\partial B}{\partial U_u} &= \frac{(n^2-n-1)}{2n(n-1)}\frac{1}{2^{\frac{n-1}{4}}|\mathcal{U}|}\sum_p \left(e^{-\frac{1}{4\sigma^2}\sum_{j=1}^{\frac{n-1}{2}}\left(U_u^T\left(V_{i_{2j-1}}^{(p)}-V_{i_{2j}}^{(p)}\right)\right)^2}\right)\left(\frac{-2}{\sigma^2}\right)\left(\sum_{j=1}^{\frac{n-1}{2}} V_{2j-1}^{(p)T} - V_{2j}^{(p)T}\right) \\ &= -\frac{(n^2-n-1)}{n(n-1)\sigma^2}\frac{1}{2^{\frac{n-1}{4}}|\mathcal{U}|}\sum_p G(p)\left(\sum_{j=1}^{\frac{n-1}{2}} V_{2j-1}^{(p)T} - V_{2j}^{(p)T}\right) \end{aligned} \tag{7.11}$$

where $G(p) = e^{-\frac{1}{4\sigma^2}\sum_{j=1}^{\frac{n-1}{2}}\left(U_u^T\left(V_{i_{2j-1}}^{(p)}-V_{i_{2j}}^{(p)}\right)\right)^2}$. Note that if we obtain permutation p' by swapping items i_{2j} and i_{2j-1} for any $j = 1, 2, \ldots \frac{n-1}{2}$, $G(p) = G(p')$. Consider all such permutations obtained by swapping items in this manner. For any fixed $j_0 \in \left(1, 2, \ldots \frac{n-2}{2}\right)$, in exactly half of these permutations, items i_{2j_0-1} (in p) and i_{2j_0} are at positions $2j_0 - 1$ and $2j_0$ respectively and in the other half, they are respectively at positions $2j_0$ and $2j_0 - 1$. Hence, half of them will contibute $V_{i_{2j_0-1}}^{(p)T} - V_{i_{2j_0}}^{(p)T}$ and the other half will contribute $V_{i_{2j_0}}^{(p)T} - V_{i_{2j_0-1}}^{(p)T}$. Thus, the sum $\sum_p G(p)\left(\sum_{j=1}^{\frac{n-1}{2}} V_{2j-1}^{(p)T} - V_{2j}^{(p)T}\right) = 0$ for all such permutations. Since we can partition the set of all permutations into subsets, each of which satisfies this property, the total sum $\sum_p G(p)\left(\sum_{j=1}^{\frac{n-1}{2}} V_{2j-1}^{(p)T} - V_{2j}^{(p)T}\right)$ is also equal to 0.

So we get $\frac{\partial B}{\partial U_u} = 0$, which cannot be used for gradient descent.

We did attempt to see if any of the upper bounds used in the derivation could be tightened further but they did not lead to tractable expansions. Hence, we decided to work on a practically useful relaxation of the problem, which is discussed in the following section.

7.2 Using Rating Thresholds

Consider the following relaxation to the problem at hand. Instead of having to recommend exactly N items to each user, we recommend an item to a user u if the predicted rating of the user for the item is greater than some threshold β. In this setting, we derive an error function that captures a combination of rating accuracy and intersection distance. Parameters of the recommendation system are then learned to minimize this error using gradient descent.

7.2.1 Formulation

We assume that the predicted ratings are distributed as $\hat{R}_{ui} \sim \mathcal{N}(U_u^T V_i, \sigma^2)$, where U and V are the matrices of latent factors corresponding to users and items respectively. Squared loss or log-likelihood of the known ratings can be used as a measure of rating accuracy. The measure for diversity is intersection distance. Let R be the matrix of known ratings and $K = \{(u,i) \,:\, u \in \mathcal{U},\ i \in \mathcal{I},\ R_{ui} \neq \phi\}$.

Let $E(\hat{R})$ be the error term we are interested in for rating accuracy. We want to use an error term (to be minimized) because the diversity objective, intersection distance is to be minimized and we would like to use a weighted linear combination of the two. Since we want to assume a probabilistic model for the ratings, log-likelihood seems to be a better choice of accuracy measure than squared loss. Since we want an objective for minimization, we use the negative of the log likelihood. That is,

$$E(\hat{R}) = -\log(Pr(R \mid U, V))$$

A common assumption used to make the above error function more tractable is that ratings of different user-item pairs are independent given the latent factors. Thus, we have,

$$\begin{aligned}
E(\hat{R}) &= -\log\left(\prod_{(u,i)\in K} Pr(R_{ui} \mid U, V)\right) \\
&= -\sum_{(u,i)\in K} \log\left(Pr(R_{ui} \mid U, V)\right) \\
&= -\sum_{(u,i)\in K} \log\left(\frac{1}{\sigma}\,\phi\left(\frac{R_{ui} - U_u^T V_i}{\sigma}\right)\right) \\
&= -\sum_{(u,i)\in K} \left(-\log\sigma - \frac{1}{2}\left(\frac{R_{ui} - U_u^T V_i}{\sigma}\right)^2 - \frac{1}{2}\log 2\pi\right)
\end{aligned}$$

Ignoring the constants, it is sufficient to consider

$$E(\hat{R}) = \frac{1}{2} \sum_{(u,i)\in K} \left(\frac{R_{ui} - U_u^T V_i}{\sigma} \right)^2$$

Taking partial derivative with respect to each parameter to be learned, we get,

$$\begin{aligned}
\frac{\partial E(\hat{R})}{\partial U_u} &= \sum_i \left(\frac{1}{2}\right)(2)\left(\frac{R_{ui} - U_u^T V_i}{\sigma}\right)\left(\frac{-1}{\sigma}\right) V_i^T \\
&= -\sum_i \left(\frac{R_{ui} - U_u^T V_i}{\sigma}\right) V_i^T = \sum_i \left(\frac{U_u^T V_i - R_{ui}}{\sigma}\right) V_i^T \\
\frac{\partial E(\hat{R})}{\partial V_i} &= \sum_u \left(\frac{1}{2}\right)(2)\left(\frac{R_{ui} - U_u^T V_i}{\sigma}\right)\left(\frac{-1}{\sigma}\right) U_u^T \\
&= -\sum_u \left(\frac{R_{ui} - U_u^T V_i}{\sigma}\right) U_u^T = \sum_u \left(\frac{U_u^T V_i - R_{ui}}{\sigma}\right) U_u^T
\end{aligned}$$

Calculating the gradient of intersection distance is done as follows. Let p_{ui} be the probability that item i is recommended to user u. Then,

$$p_{ui} = Pr(\hat{R}_{ui} \geq \beta) = 1 - \Phi\left(\frac{\beta - U_u^T V_i}{\sigma}\right)$$

Let c_i be the expected number of times item i is recommended and let $\overline{c}$ be the expected number of items recommended overall. Then,

$$c_i = \sum_u p_{ui}, \quad \overline{c} = \sum_i c_i$$

We have proved in Appendix A that,

$$\begin{aligned}
\frac{\partial D_I}{\partial U_u} &= \sum_i \left|\frac{c_i}{\overline{c}} - \frac{1}{n}\right| \left(\frac{n}{(nc_i - \overline{c})\overline{c}}\right) \left(\frac{\overline{c}}{\sigma^2}\,\phi\left(\frac{\beta - U_u^T V_i}{\sigma}\right) V_i^T - \frac{c_i}{\sigma^2} \sum_{i'} \phi\left(\frac{\beta - U_u^T V_{i'}}{\sigma}\right) V_{i'}^T\right) \\
\frac{\partial D_I}{\partial V_i} &= \frac{n}{\overline{c}\,\sigma^2} \left(\sum_u \phi\left(\frac{\beta - U_u^T V_i}{\sigma}\right) U_u^T\right) \left(\left(\frac{\overline{c}}{nc_i - \overline{c}}\right)\left|\frac{c_i}{\overline{c}} - \frac{1}{n}\right| - \sum_{i'} \left|\frac{c_{i'}}{\overline{c}} - \frac{1}{n}\right| \left(\frac{c_{i'}}{nc_{i'} - \overline{c}}\right)\right)
\end{aligned}$$

Net error to minimize, $E = \alpha E(\hat{R}) + (1-\alpha) D_I$ where α is a parameter that decides how much importance is given to each objective. Then, we have,

$$\begin{aligned}
\frac{\partial E}{\partial U_u} &= \alpha \frac{\partial E(\hat{R})}{\partial U_u} + (1-\alpha)\frac{\partial D_I}{\partial U_u} \\
&= \frac{\alpha}{\sigma^2}\sum_i (U_u^T V_i - R_{ui}) V_i^T + \\
& \frac{1-\alpha}{\sigma^2}\sum_i \left|\frac{c_i}{\bar{c}} - \frac{1}{n}\right| \left(\frac{n}{(nc_i - \bar{c})\bar{c}}\right) \left(\bar{c}\,\phi\left(\frac{\beta - U_u^T V_i}{\sigma}\right) V_i^T - c_i \sum_{i'} \phi\left(\frac{\beta - U_u^T V_{i'}}{\sigma}\right) V_{i'}^T\right)
\end{aligned}$$

Also,

$$\begin{aligned}
\frac{\partial E}{\partial V_i} &= \alpha \frac{\partial E(\hat{R})}{\partial V_i} + (1-\alpha)\frac{\partial D_I}{\partial V_i} \\
&= \frac{\alpha}{\sigma^2}\sum_u (U_u^T V_i - R_{ui}) U_u^T + \\
& \frac{1-\alpha}{\sigma^2}\left(\frac{n}{\bar{c}}\right)\left(\sum_u \phi\left(\frac{\beta - U_u^T V_i}{\sigma}\right) U_u^T\right)\left(\left(\frac{\bar{c}}{nc_i - \bar{c}}\right)\left|\frac{c_i}{\bar{c}} - \frac{1}{n}\right| - \sum_{i'}\left|\frac{c_{i'}}{\bar{c}} - \frac{1}{n}\right|\left(\frac{c_{i'}}{nc_{i'} - \bar{c}}\right)\right)
\end{aligned}$$

We can learn the parameters of the system using gradient descent as follows -

$$U_u \leftarrow U_u - \eta \frac{\partial E}{\partial U_u}$$
$$V_i \leftarrow V_i - \eta \frac{\partial E}{\partial V_i}$$

The system has a number of parameters to be tuned manually -

- η - Step size for gradient descent
- α - Weight given to rating accuracy
- σ - Variance of distribution used for predicting ratings
- β - Rating threshold above which items are recommended
- K - Number of latent dimensions

Note that the ”ratings” predicted by this system need not necessarily be equal to the users’ actual ratings, since we explicitly offset the error function using intersection distance. The purpose of predicting such ”ratings” is that it avoids having to explicitly store a recommendation list for each user, as would be required by methods like the min cost flow method.

7.2.2 Experiments

Since this system has a very large number of parameters, the following grid-search based method was used. First, for a large number of possible parameter settings - 50, 100 and 150 latent features, rating thresholds of 3, 3.5 and 4 and $\sigma,\ \alpha,\ \eta \in \{0.1, 0.2 \ldots 0.9\}$ were run for ten iterations each of batch gradient descent. Following this, for each combination of K and β, the best 5 values in terms of intersection distance and the best 5 in terms of average predicted rating were taken and these were run for 100 iterations. The results of these parameter values are shown in tables Appendix C.

7.2.3 Observations and Discussion

We observe that there are some parameter settings for which we attain an intersection distance below 0.4, sometimes as low as 0.1-0.2, which is significantly better than those from any other method. The average predicted rating shows a wider trend. For a few settings, it reaches as high as 4.5-4.6 but it also attains low values of 3.8-3.9. Some earlier methods have resulted in a higher APR.

We also see that most settings in the set of values selected for a dataset after 10 iterations have the same or similar values of σ. This possibly suggests that there is a particular value of σ at which the distribution $\hat{R}_{ui} \sim \mathcal{N}(U_u^T V_i, \sigma^2)$ models the true ratings well. We also note that for the MovieLens dataset, most settings have a low value of α as well. This is unsurprising since a low value of α increases the weightage given by the objective function to intersection distance. For the Netflix dataset, we find that most settings have a low value of η. It is possible that the starting point in this case is close a minimum and hence a low learning rate is required to prevent oscillations. It is also possible that since most of these parameter settings have a high variance (σ), not much importance can be given to individual observations, which also calls for a low learning rate. This could also explain why higher values of α are required. Since ratings can vary significantly from their mean values, a high importance to accuracy of known ratings would be required to stabilize mean values appropriately.

Overall, the method looks promising. It requires initial tuning of parameters for the dataset, but is significantly more scalable as it eliminates the need to compute and transfer a complete rating matrix.

CHAPTER 8

Conclusions and Future Work

In this project, we have identified the deficiencies of intuitive metrics such as coverage and entropy for the evaluation of the aggregate diversity of a recommender system. We have proved that the intersection distance of the normalized recommended item count vector to the uniform distribution overcomes these deficiencies, while being bounded and easy to compute. We have also proposed a number of techniques for the joint optimization of recommendation list relevance and diversity in terms of intersection distance and demonstrated their efficacy through experiments on real world datasets.

Some directions for future research would be to examine appropriate non-convex optimization techniques that directly optimize the parameters of a recommender system such that the recommendation lists it produces have a high intersection distance, which overcome the lack of convergence guarantees for gradient descent. Another alternative would be to improve the scalability of methods such as the min cost flow formulation which does directly optimize intersection distance.

To the best of our knowledge, this is one of the few works which optimize aggregate diversity directly and the first to optimize intersection distance as a metric. A portion of this work is currently under review at ACM RecSys 2015.

APPENDIX A

Derivations - Gradient Descent Method

The following are the proofs of the lemmas used in the gradient descent methods described in Chapter 7.

A.1 Proof of Lemma 7

Lemma 7 states that

$$I = \int_t^\infty \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-1}{2\sigma^2}\left((x-\mu_1)^2+(x-\mu_2)^2\right)} dx \leq \sqrt{\pi}\,\phi\left(\frac{\mu}{\sqrt{2}}\right) \tag{A.1}$$

Consider the integral -

$$I = \int_t^\infty \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-1}{2\sigma^2}\left((x-\mu_1)^2+(x-\mu_2)^2\right)} dx \tag{A.2}$$

Let $\frac{x-\mu_1}{\sigma} = y \Rightarrow dx = \sigma dy$. Then

$$\frac{(x-\mu_1)^2}{\sigma^2} = y^2, \quad \frac{(x-\mu_2)^2}{\sigma^2} = \frac{(\sigma y + (\mu_1-\mu_2))^2}{\sigma^2}$$

Let $\mu = \frac{\mu_1-\mu_2}{\sigma}$. Then

$$\frac{(x-\mu_2)^2}{\sigma^2} = (y+\mu)^2$$

Also, when $x = t$, $y = \frac{t-\mu_1}{\sigma} = t'$. Substituting these in A.2, we get

$$\begin{aligned} I &= \int_{t'}^\infty \frac{1}{\sqrt{2\pi}} e^{-\frac{y^2}{2}} e^{-\frac{(y+\mu)^2}{2}} dy = \sqrt{2\pi}\int_{t'}^\infty \left(\frac{1}{\sqrt{2\pi}}e^{-\frac{y^2}{2}}\right)\left(\frac{1}{\sqrt{2\pi}}e^{-\frac{(y+\mu)^2}{2}}\right) dy \\ &= \sqrt{2\pi}\int_{t'}^\infty \phi(y)\phi(y+\mu)dy \end{aligned}$$

Using, $\int \phi(x)\phi(a+bx)dx = \frac{1}{t}\phi\left(\frac{a}{t}\right)\Phi\left(tx+\frac{ab}{t}\right)+c$ where $t=\sqrt{1+b^2}$, we get

$$\begin{aligned}
I &= \sqrt{2\pi}\left(\frac{1}{\sqrt{2}}\phi\left(\frac{\mu}{\sqrt{2}}\right)\Phi\left(\sqrt{2}y+\frac{\mu}{\sqrt{2}}\right)\right)\Big|_{t'}^{\infty} \\
&= \sqrt{\pi}\,\phi\left(\frac{\mu}{\sqrt{2}}\right)\left(\Phi(\infty)-\Phi\left(\sqrt{2}\left(\frac{t-\mu_1}{\sigma}\right)+\frac{\mu}{\sqrt{2}}\right)\right) \\
&= \sqrt{\pi}\,\phi\left(\frac{\mu}{\sqrt{2}}\right)\left(1-\Phi\left(\sqrt{2}\left(\frac{t-\mu_1}{\sigma}\right)+\frac{\mu}{\sqrt{2}}\right)\right) \\
&\le \sqrt{\pi}\,\phi\left(\frac{\mu}{\sqrt{2}}\right)
\end{aligned}$$

A.2 Proof of Lemma 8

Lemma 8 states that for any random variable $x \in [0,1]$,

$$E(|x-a|) \le E[x]\frac{(1-a-a^2)}{1-a}+\frac{2a^2-a}{1-a} \tag{A.3}$$

Suppose we have a random variable $x \in [0,1]$, then, we can bound $E[|x-a|]$ as follows

$$\begin{aligned}
E(|x-a|) &= \int_0^1 |x-a|Pr(x)dx \\
&= \int_a^1 (x-a)Pr(x)dx + \int_0^a (a-x)Pr(x)dx \\
&= \int_0^1 (x-a)Pr(x)dx - \int_0^a (x-a)Pr(x)dx + \int_0^a (a-x)Pr(x)dx \\
&= \int_0^1 xPr(x)dx - a\int_0^1 Pr(x)dx + 2\int_0^a (a-x)Pr(x)dx \\
&= E[x]-a+2\int_0^a (a-x)Pr(x)dx \qquad \text{(A.4)}
\end{aligned}$$

$x \in [0,1] \Rightarrow 1-x \in [0,1]$. Then,

$$\begin{aligned}
& Pr(x \le a) = Pr(1-x \ge 1-a) \le \frac{1-E[x]}{1-a} \quad \text{(Markov's inequality)} \\
\Rightarrow \quad & Pr(x) \le \frac{1-E[x]}{1-a}\ \forall\, x \le a \qquad \text{(A.5)}
\end{aligned}$$

Substituting from A.5 in A.4, we get,

$$\begin{aligned}
E(|x-a|) &\leq E[x] - a + 2\int_0^a (a-x)\frac{1-E[x]}{1-a} Pr(x)dx \\
&= E[x] - a + 2\frac{1-E[x]}{1-a}\int_0^a (a-x)dx \\
&= E[x] - a + 2\frac{1-E[x]}{1-a}\left(\frac{a^2}{2}\right) \\
&= E[x]\left(1 - \frac{a^2}{1-a}\right) + \left(-a + \frac{a^2}{1-a}\right) \\
&= E[x]\frac{(1-a-a^2)}{1-a} + \frac{2a^2-a}{1-a} \qquad \text{(A.6)}
\end{aligned}$$

A.3 Gradient of Intersection Distance

Let p_{ui} be the probability that item i is recommended to user u. Then,

$$p_{ui} = Pr(\hat{R}_{ui} \geq \beta) = 1 - \Phi\left(\frac{\beta - U_u^T V_i}{\sigma}\right)$$

Let c_i be the expected number of times item i is recommended and let $\overline{c}$ be the expected number of items recommended overall. Then,

$$c_i = \sum_u p_{ui}, \quad \overline{c} = \sum_i c_i$$

Then,

$$\begin{aligned}
\frac{\partial p_{ui}}{\partial U_u} &= -\phi\left(\frac{\beta - U_u^T V_i}{\sigma^2}\right)\left(\frac{-1}{\sigma^2}\right) V_i^T = \frac{1}{\sigma^2}\,\phi\left(\frac{\beta - U_u^T V_i}{\sigma^2}\right) V_i^T \\
\frac{\partial p_{ui}}{\partial V_i} &= -\phi\left(\frac{\beta - U_u^T V_i}{\sigma^2}\right)\left(\frac{-1}{\sigma^2}\right) U_u^T = \frac{1}{\sigma^2}\,\phi\left(\frac{\beta - U_u^T V_i}{\sigma^2}\right) U_u^T \\
\frac{\partial p_{ui}}{\partial U_{u'}} &= 0 \text{ if } u' \neq u \\
\frac{\partial p_{ui}}{\partial V_{i'}} &= 0 \text{ if } i' \neq i
\end{aligned}$$

Using this, we get,

$$\begin{aligned}
\frac{\partial c_i}{\partial U_u} &= \sum_{u'} \frac{\partial p_{u'i}}{\partial U_u} \\
&= \frac{\partial p_{ui}}{\partial U_u} + \sum_{u' \neq u} \frac{\partial p_{u'i}}{\partial U_u} \\
&= \frac{\partial p_{ui}}{\partial U_u} \\
&= \frac{1}{\sigma^2}\, \phi\left(\frac{\beta - U_u^T V_i}{\sigma^2}\right) V_i^T \\
\frac{\partial c_i}{\partial V_{i'}} &= 0 \text{ if } i' \neq i \\
\frac{\partial c_i}{\partial V_i} &= \sum_u \frac{\partial p_{ui}}{\partial V_i} \\
&= \frac{1}{\sigma^2} \sum_u \phi\left(\frac{\beta - U_u^T V_i}{\sigma^2}\right) U_u^T
\end{aligned}$$

Also, using $\overline{c} = \sum_i c_i$, we get,

$$\begin{aligned}
\frac{\partial \overline{c}}{\partial U_u} &= \sum_i \frac{\partial c_i}{\partial U_u} = \frac{1}{\sigma^2} \sum_i \phi\left(\frac{\beta - U_u^T V_i}{\sigma^2}\right) V_i^T \\
\frac{\partial \overline{c}}{\partial V_i} &= \sum_{i'} \frac{\partial c_{i'}}{\partial V_i} = \frac{\partial c_i}{\partial V_i} + \sum_{i' \neq i} \frac{\partial c_{i'}}{\partial V_i} \\
&= \frac{\partial c_i}{\partial V_i} \\
&= \frac{1}{\sigma^2} \sum_u \phi\left(\frac{\beta - U_u^T V_i}{\sigma^2}\right) U_u^T
\end{aligned}$$

Intersection distance D_I is given by,

$$D_I = \sum_i \left| \frac{c_i}{\overline{c}} - \frac{1}{n} \right|$$

Taking partial derivativatives with respect to the parameters,

$$\begin{aligned}
\frac{\partial D_I}{\partial U_u} &= \sum_i \frac{\left|\frac{c_i}{\overline{c}} - \frac{1}{n}\right|}{\left(\frac{c_i}{\overline{c}} - \frac{1}{n}\right)} \frac{\partial}{\partial U_u}\left(\frac{c_i}{\overline{c}}\right) \\
&= \sum_i \frac{\left|\frac{c_i}{\overline{c}} - \frac{1}{n}\right|}{\left(\frac{c_i}{\overline{c}} - \frac{1}{n}\right)} \frac{\overline{c}\frac{\partial c_i}{\partial U_u} - c_i \frac{\partial \overline{c}}{\partial U_u}}{(\overline{c})^2} \\
&= \sum_i \left|\frac{c_i}{\overline{c}} - \frac{1}{n}\right| \left(\frac{n\overline{c}}{(nc_i - \overline{c})(\overline{c})^2}\right) \left(\frac{\overline{c}}{\sigma^2}\,\phi\left(\frac{\beta - U_u^T V_i}{\sigma}\right) V_i^T - \frac{c_i}{\sigma^2}\sum_{i'} \phi\left(\frac{\beta - U_u^T V_{i'}}{\sigma}\right) V_{i'}^T\right) \\
&= \sum_i \left|\frac{c_i}{\overline{c}} - \frac{1}{n}\right| \left(\frac{n}{(nc_i - \overline{c})\overline{c}}\right) \left(\frac{\overline{c}}{\sigma^2}\,\phi\left(\frac{\beta - U_u^T V_i}{\sigma}\right) V_i^T - \frac{c_i}{\sigma^2}\sum_{i'} \phi\left(\frac{\beta - U_u^T V_{i'}}{\sigma}\right) V_{i'}^T\right)
\end{aligned}$$

Also,

$$\begin{aligned}
\frac{\partial D_I}{\partial V_i} &= \sum_{i'} \frac{\left|\frac{c_{i'}}{\overline{c}} - \frac{1}{n}\right|}{\left(\frac{c_{i'}}{\overline{c}} - \frac{1}{n}\right)} \frac{\partial}{\partial V_i}\left(\frac{c_{i'}}{\overline{c}}\right) \\
&= \sum_{i'} \left|\frac{c_{i'}}{\overline{c}} - \frac{1}{n}\right| \left(\frac{n}{(nc_{i'} - \overline{c})\overline{c}}\right) \left(\overline{c}\frac{\partial c_{i'}}{\partial V_i} - c_{i'}\frac{\partial \overline{c}}{\partial V_i}\right) \\
&= \left|\frac{c_i}{\overline{c}} - \frac{1}{n}\right| \left(\frac{n}{(nc_i - \overline{c})\overline{c}}\right) \left(\overline{c}\frac{\partial c_i}{\partial V_i} - c_i\frac{\partial \overline{c}}{\partial V_i}\right) - \sum_{i' \neq i} \left|\frac{c_{i'}}{\overline{c}} - \frac{1}{n}\right| \left(\frac{nc_{i'}}{(nc_{i'} - \overline{c})\overline{c}}\right) \frac{\partial \overline{c}}{\partial V_i} \\
&= \left|\frac{c_i}{\overline{c}} - \frac{1}{n}\right| \left(\frac{n(\overline{c} - c_i)}{(nc_i - \overline{c})\overline{c}}\right) \frac{\partial c_i}{\partial V_i} - \sum_{i' \neq i} \left|\frac{c_{i'}}{\overline{c}} - \frac{1}{n}\right| \left(\frac{nc_{i'}}{(nc_{i'} - \overline{c})\overline{c}}\right) \frac{\partial c_i}{\partial V_i} \\
&= \frac{n}{\overline{c}}\frac{\partial c_i}{\partial V_i}\left(\left|\frac{c_i}{\overline{c}} - \frac{1}{n}\right| \left(\frac{\overline{c} - c_i}{nc_i - \overline{c}}\right) - \sum_{i' \neq i} \left|\frac{c_{i'}}{\overline{c}} - \frac{1}{n}\right| \left(\frac{c_{i'}}{nc_{i'} - \overline{c}}\right)\right) \\
&= \frac{n}{\overline{c}}\frac{\partial c_i}{\partial V_i}\left(\left(\frac{\overline{c}}{nc_i - \overline{c}}\right)\left|\frac{c_i}{\overline{c}} - \frac{1}{n}\right| - \sum_{i'} \left|\frac{c_{i'}}{\overline{c}} - \frac{1}{n}\right| \left(\frac{c_{i'}}{nc_{i'} - \overline{c}}\right)\right) \\
&= \frac{n}{\overline{c}\,\sigma^2}\left(\sum_u \phi\left(\frac{\beta - U_u^T V_i}{\sigma}\right) U_u^T\right)\left(\left(\frac{\overline{c}}{nc_i - \overline{c}}\right)\left|\frac{c_i}{\overline{c}} - \frac{1}{n}\right| - \sum_{i'} \left|\frac{c_{i'}}{\overline{c}} - \frac{1}{n}\right| \left(\frac{c_{i'}}{nc_{i'} - \overline{c}}\right)\right)
\end{aligned}$$

APPENDIX B

Tables - Basic Min Cost Flow Method

The following tables show the variation of intersection distance and average predicted rating with the parameter k_2 of the basic min cost flow method described in Section 6.2 on the MovieLens dataset for different settings.

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.992	13.0	4.994	0.992	13.0	4.994
Random	0.953	79.0	4.448	0.951	82.0	4.441
Pseudo Gradient Descent	0.944	100.0	4.449	0.898	177.0	4.274
Greedy	0.944	100.0	4.449	0.944	100.0	4.449
Min Cost Flow:0-6	0.621	638.0	4.002	-	-	-
Min Cost Flow:0-7	0.711	486.0	4.422	-	-	-
Min Cost Flow:0-8	0.76	403.0	4.649	-	-	-
Min Cost Flow:0-9	0.798	340.0	4.731	-	-	-
Min Cost Flow:0-10	0.82	302.0	4.786	-	-	-
Min Cost Flow:0-11	0.839	271.0	4.873	-	-	-
Min Cost Flow:0-12	0.826	293.0	4.843	0.675	546.0	4.501
Min Cost Flow:0-13	0.81	320.0	4.83	0.761	402.0	4.738
Min Cost Flow:0-14	0.813	315.0	4.829	0.79	354.0	4.854
Min Cost Flow:0-15	0.814	313.0	4.815	0.806	327.0	4.857
Min Cost Flow:0-16	0.833	281.0	4.816	0.817	307.0	4.827
Min Cost Flow:0-17	0.823	297.0	4.811	0.823	297.0	4.802
Min Cost Flow:0-18	0.834	280.0	4.771	0.822	299.0	4.85
Min Cost Flow:0-19	0.824	296.0	4.785	0.794	347.0	4.788
Min Cost Flow:0-20	0.837	275.0	4.781	0.876	208.0	4.924
Min Cost Flow:1-6	0.781	368.0	4.292	-	-	-
Min Cost Flow:1-7	0.813	314.0	4.624	-	-	-
Min Cost Flow:1-8	0.813	315.0	4.661	-	-	-

Table B.1: Min cost flow bounding method on MovieLens : Baseline - ItemAverage

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Min Cost Flow:1-9	0.818	306.0	4.749	-	-	-
Min Cost Flow:1-10	0.878	206.0	4.885	-	-	-
Min Cost Flow:1-11	0.859	237.0	4.841	-	-	-
Min Cost Flow:1-12	0.854	246.0	4.857	0.853	247.0	4.866
Min Cost Flow:1-13	0.836	276.0	4.848	0.836	276.0	4.834
Min Cost Flow:1-14	0.817	308.0	4.84	0.834	280.0	4.804
Min Cost Flow:1-15	0.809	321.0	4.804	0.872	216.0	4.887
Min Cost Flow:1-16	0.857	240.0	4.839	0.843	264.0	4.852
Min Cost Flow:1-17	0.869	220.0	4.807	0.851	251.0	4.865
Min Cost Flow:1-18	0.826	292.0	4.799	0.834	279.0	4.764
Min Cost Flow:1-19	0.83	286.0	4.839	0.849	254.0	4.803
Min Cost Flow:1-20	0.867	223.0	4.798	0.855	244.0	4.82
Min Cost Flow Threshold:0-7	0.703	500.0	4.416	-	-	-
Min Cost Flow Threshold:0-8	0.751	418.0	4.578	-	-	-
Min Cost Flow Threshold:0-9	0.784	364.0	4.695	-	-	-
Min Cost Flow Threshold:0-10	0.822	300.0	4.771	-	-	-
Min Cost Flow Threshold:0-11	0.836	276.0	4.886	-	-	-
Min Cost Flow Threshold:0-12	0.832	282.0	4.851	0.672	551.0	4.491
Min Cost Flow Threshold:0-13	0.833	281.0	4.855	0.746	428.0	4.726
Min Cost Flow Threshold:0-14	0.828	290.0	4.836	0.779	372.0	4.83
Min Cost Flow Threshold:0-15	0.829	288.0	4.829	0.797	342.0	4.823
Min Cost Flow Threshold:0-16	0.827	291.0	4.835	0.82	302.0	4.813
Min Cost Flow Threshold:0-17	0.849	254.0	4.826	0.825	295.0	4.864
Min Cost Flow Threshold:0-18	0.838	272.0	4.813	0.83	286.0	4.841
Min Cost Flow Threshold:0-19	0.842	266.0	4.806	0.813	314.0	4.793
Min Cost Flow Threshold:0-20	0.82	303.0	4.815	0.823	297.0	4.784
Min Cost Flow Threshold:1-7	0.812	316.0	4.603	-	-	-
Min Cost Flow Threshold:1-8	0.825	294.0	4.684	-	-	-
Min Cost Flow Threshold:1-9	0.839	270.0	4.765	-	-	-
Min Cost Flow Threshold:1-10	0.881	200.0	4.869	-	-	-
Min Cost Flow Threshold:1-11	0.84	269.0	4.858	-	-	-
Min Cost Flow Threshold:1-12	0.853	247.0	4.857	0.849	254.0	4.839
Min Cost Flow Threshold:1-13	0.834	279.0	4.829	0.856	243.0	4.845
Min Cost Flow Threshold:1-14	0.841	268.0	4.805	0.85	253.0	4.845
Min Cost Flow Threshold:1-15	0.819	305.0	4.844	0.8	337.0	4.811
Min Cost Flow Threshold:1-16	0.852	249.0	4.868	0.832	283.0	4.837
Min Cost Flow Threshold:1-17	0.859	237.0	4.841	0.831	284.0	4.763
Min Cost Flow Threshold:1-18	0.856	243.0	4.819	0.841	268.0	4.839
Min Cost Flow Threshold:1-19	0.832	283.0	4.767	0.838	272.0	4.784
Min Cost Flow Threshold:1-20	0.847	257.0	4.781	0.834	279.0	4.805

Table B.2: Min cost flow bounding method on MovieLens : Baseline - ItemAverage (contd.)

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.992	13.0	4.994	0.992	13.0	4.994
Random	0.952	80.0	4.456	0.952	80.0	4.439
Pseudo Gradient Descent	0.944	100.0	4.449	0.898	177.0	4.274
Greedy	0.944	100.0	4.449	0.944	100.0	4.449
Min Cost Flow:0-6	0.621	638.0	4.002	-	-	-
Min Cost Flow:0-7	0.711	486.0	4.422	-	-	-
Min Cost Flow:0-8	0.76	403.0	4.649	-	-	-
Min Cost Flow:0-9	0.798	340.0	4.731	-	-	-
Min Cost Flow:0-10	0.82	302.0	4.786	-	-	-
Min Cost Flow:0-11	0.839	271.0	4.873	-	-	-
Min Cost Flow:0-12	0.826	293.0	4.843	0.675	546.0	4.501
Min Cost Flow:0-13	0.81	320.0	4.83	0.761	402.0	4.738
Min Cost Flow:0-14	0.813	315.0	4.829	0.79	354.0	4.854
Min Cost Flow:0-15	0.814	313.0	4.815	0.806	327.0	4.857
Min Cost Flow:0-16	0.833	281.0	4.816	0.817	307.0	4.827
Min Cost Flow:0-17	0.823	297.0	4.811	0.823	297.0	4.802
Min Cost Flow:0-18	0.834	280.0	4.771	0.822	299.0	4.85
Min Cost Flow:0-19	0.824	296.0	4.785	0.794	347.0	4.788
Min Cost Flow:0-20	0.837	275.0	4.781	0.876	208.0	4.924
Min Cost Flow:1-6	0.781	368.0	4.292	-	-	-
Min Cost Flow:1-7	0.813	314.0	4.624	-	-	-
Min Cost Flow:1-8	0.813	315.0	4.661	-	-	-
Min Cost Flow:1-9	0.818	306.0	4.749	-	-	-
Min Cost Flow:1-10	0.878	206.0	4.885	-	-	-
Min Cost Flow:1-11	0.859	237.0	4.841	-	-	-
Min Cost Flow:1-12	0.854	246.0	4.857	0.853	247.0	4.866
Min Cost Flow:1-13	0.836	276.0	4.848	0.836	276.0	4.834
Min Cost Flow:1-14	0.817	308.0	4.84	0.834	280.0	4.804
Min Cost Flow:1-15	0.809	321.0	4.804	0.872	216.0	4.887

Table B.3: Min cost flow bounding method on MovieLens : Baseline - ItemUserAverage

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Min Cost Flow:1-16	0.857	240.0	4.839	0.843	264.0	4.852
Min Cost Flow:1-17	0.869	220.0	4.807	0.851	251.0	4.865
Min Cost Flow:1-18	0.826	292.0	4.799	0.834	279.0	4.764
Min Cost Flow:1-19	0.83	286.0	4.839	0.849	254.0	4.803
Min Cost Flow:1-20	0.867	223.0	4.798	0.855	244.0	4.82
Min Cost Flow Threshold:0-7	0.703	500.0	4.416	-	-	-
Min Cost Flow Threshold:0-8	0.751	418.0	4.578	-	-	-
Min Cost Flow Threshold:0-9	0.784	364.0	4.695	-	-	-
Min Cost Flow Threshold:0-10	0.822	300.0	4.771	-	-	-
Min Cost Flow Threshold:0-11	0.836	276.0	4.886	-	-	-
Min Cost Flow Threshold:0-12	0.832	282.0	4.851	0.672	551.0	4.491
Min Cost Flow Threshold:0-13	0.833	281.0	4.855	0.746	428.0	4.726
Min Cost Flow Threshold:0-14	0.828	290.0	4.836	0.779	372.0	4.83
Min Cost Flow Threshold:0-15	0.829	288.0	4.829	0.797	342.0	4.823
Min Cost Flow Threshold:0-16	0.827	291.0	4.835	0.82	302.0	4.813
Min Cost Flow Threshold:0-17	0.849	254.0	4.826	0.825	295.0	4.864
Min Cost Flow Threshold:0-18	0.838	272.0	4.813	0.83	286.0	4.841
Min Cost Flow Threshold:0-19	0.842	266.0	4.806	0.813	314.0	4.793
Min Cost Flow Threshold:0-20	0.82	303.0	4.815	0.823	297.0	4.784
Min Cost Flow Threshold:1-7	0.812	316.0	4.603	-	-	-
Min Cost Flow Threshold:1-8	0.825	294.0	4.684	-	-	-
Min Cost Flow Threshold:1-9	0.839	270.0	4.765	-	-	-
Min Cost Flow Threshold:1-10	0.881	200.0	4.869	-	-	-
Min Cost Flow Threshold:1-11	0.84	269.0	4.858	-	-	-
Min Cost Flow Threshold:1-12	0.853	247.0	4.857	0.849	254.0	4.839
Min Cost Flow Threshold:1-13	0.834	279.0	4.829	0.856	243.0	4.845
Min Cost Flow Threshold:1-14	0.841	268.0	4.805	0.85	253.0	4.845
Min Cost Flow Threshold:1-15	0.819	305.0	4.844	0.8	337.0	4.811
Min Cost Flow Threshold:1-16	0.852	249.0	4.868	0.832	283.0	4.837
Min Cost Flow Threshold:1-17	0.859	237.0	4.841	0.831	284.0	4.763
Min Cost Flow Threshold:1-18	0.856	243.0	4.819	0.841	268.0	4.839
Min Cost Flow Threshold:1-19	0.832	283.0	4.767	0.838	272.0	4.784
Min Cost Flow Threshold:1-20	0.847	257.0	4.781	0.834	279.0	4.805

Table B.4: Min cost flow bounding method on MovieLens : Baseline - ItemUserAverage (contd.)

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.833	281.0	5.0	0.833	281.0	5.0
Random	0.706	495.0	4.998	0.707	492.0	4.997
Pseudo Gradient Descent	0.539	966.0	4.996	0.472	1218.0	4.914
Greedy	0.539	966.0	4.996	0.539	966.0	4.996
Min Cost Flow:0-6	0.603	668.0	4.518	-	-	-
Min Cost Flow:0-7	0.634	616.0	4.555	-	-	-
Min Cost Flow:0-8	0.644	599.0	4.564	-	-	-
Min Cost Flow:0-9	0.657	577.0	4.556	-	-	-
Min Cost Flow:0-10	0.673	550.0	4.554	-	-	-
Min Cost Flow:0-11	0.699	507.0	4.53	-	-	-
Min Cost Flow:0-12	0.704	498.0	4.544	0.685	529.0	4.51
Min Cost Flow:0-13	0.709	489.0	4.567	0.707	493.0	4.537
Min Cost Flow:0-14	0.725	463.0	4.593	0.722	467.0	4.533
Min Cost Flow:0-15	0.738	441.0	4.577	0.732	451.0	4.586
Min Cost Flow:0-16	0.758	407.0	4.51	0.741	435.0	4.562
Min Cost Flow:0-17	0.72	471.0	4.508	0.747	426.0	4.571
Min Cost Flow:0-18	0.776	377.0	4.631	0.713	482.0	4.541
Min Cost Flow:0-19	0.76	403.0	4.547	0.762	400.0	4.561
Min Cost Flow:0-20	0.74	438.0	4.548	0.769	388.0	4.572
Min Cost Flow:1-6	0.65	588.0	4.525	-	-	-
Min Cost Flow:1-7	0.64	606.0	4.52	-	-	-
Min Cost Flow:1-8	0.645	597.0	4.544	-	-	-
Min Cost Flow:1-9	0.645	597.0	4.546	-	-	-
Min Cost Flow:1-10	0.7	504.0	4.512	-	-	-
Min Cost Flow:1-11	0.683	534.0	4.552	-	-	-
Min Cost Flow:1-12	0.69	522.0	4.597	0.685	530.0	4.58
Min Cost Flow:1-13	0.731	452.0	4.585	0.698	508.0	4.587
Min Cost Flow:1-14	0.737	442.0	4.557	0.711	486.0	4.548
Min Cost Flow:1-15	0.756	411.0	4.503	0.749	423.0	4.537

Table B.5: Min cost flow bounding method on MovieLens : Baseline - ItemBased

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Min Cost Flow:1-16	0.756	410.0	4.573	0.754	413.0	4.536
Min Cost Flow:1-17	0.71	487.0	4.559	0.748	424.0	4.551
Min Cost Flow:1-18	0.78	370.0	4.635	0.767	392.0	4.541
Min Cost Flow:1-19	0.731	452.0	4.54	0.782	366.0	4.598
Min Cost Flow:1-20	0.784	364.0	4.516	0.775	378.0	4.552
Min Cost Flow Threshold:0-7	0.647	594.0	4.571	-	-	-
Min Cost Flow Threshold:0-8	0.639	607.0	4.561	-	-	-
Min Cost Flow Threshold:0-9	0.671	553.0	4.519	-	-	-
Min Cost Flow Threshold:0-10	0.679	540.0	4.561	-	-	-
Min Cost Flow Threshold:0-11	0.675	546.0	4.603	-	-	-
Min Cost Flow Threshold:0-12	0.716	477.0	4.544	0.702	502.0	4.583
Min Cost Flow Threshold:0-13	0.702	502.0	4.563	0.702	502.0	4.539
Min Cost Flow Threshold:0-14	0.727	459.0	4.623	0.714	481.0	4.587
Min Cost Flow Threshold:0-15	0.734	448.0	4.563	0.729	456.0	4.55
Min Cost Flow Threshold:0-16	0.752	417.0	4.566	0.756	411.0	4.551
Min Cost Flow Threshold:0-17	0.747	425.0	4.583	0.757	408.0	4.552
Min Cost Flow Threshold:0-18	0.758	407.0	4.563	0.75	420.0	4.581
Min Cost Flow Threshold:0-19	0.769	389.0	4.522	0.769	389.0	4.565
Min Cost Flow Threshold:0-20	0.776	376.0	4.559	0.762	401.0	4.549
Min Cost Flow Threshold:1-6	0.637	611.0	4.577	-	-	-
Min Cost Flow Threshold:1-7	0.637	610.0	4.542	-	-	-
Min Cost Flow Threshold:1-8	0.65	588.0	4.541	-	-	-
Min Cost Flow Threshold:1-9	0.685	529.0	4.544	-	-	-
Min Cost Flow Threshold:1-10	0.664	565.0	4.535	-	-	-
Min Cost Flow Threshold:1-11	0.702	502.0	4.527	-	-	-
Min Cost Flow Threshold:1-12	0.732	451.0	4.549	0.702	501.0	4.59
Min Cost Flow Threshold:1-13	0.724	465.0	4.516	0.725	462.0	4.584
Min Cost Flow Threshold:1-14	0.741	435.0	4.565	0.746	428.0	4.576
Min Cost Flow Threshold:1-15	0.748	424.0	4.576	0.735	446.0	4.542
Min Cost Flow Threshold:1-16	0.709	490.0	4.55	0.746	427.0	4.547
Min Cost Flow Threshold:1-17	0.749	423.0	4.576	0.748	424.0	4.556
Min Cost Flow Threshold:1-18	0.772	383.0	4.601	0.745	429.0	4.568
Min Cost Flow Threshold:1-19	0.772	383.0	4.575	0.746	428.0	4.551
Min Cost Flow Threshold:1-20	0.797	341.0	4.564	0.742	434.0	4.578

Table B.6: Min cost flow bounding method on MovieLens : Baseline - ItemBased (contd.)

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.883	196.0	3.875	0.883	196.0	3.875
Random	0.874	212.0	3.61	0.869	221.0	3.619
Pseudo Gradient Descent	0.794	346.0	3.621	0.758	407.0	3.585
Greedy	0.794	346.0	3.624	0.794	346.0	3.624
Min Cost Flow:0-6	0.61	656.0	3.463	-	-	-
Min Cost Flow:0-7	0.743	432.0	4.399	-	-	-
Min Cost Flow:0-8	0.795	345.0	4.733	-	-	-
Min Cost Flow:0-9	0.817	307.0	4.837	-	-	-
Min Cost Flow:0-10	0.847	258.0	4.888	-	-	-
Min Cost Flow:0-11	0.853	247.0	4.902	-	-	-
Min Cost Flow:0-12	0.846	259.0	4.907	0.669	556.0	4.648
Min Cost Flow:0-13	0.851	251.0	4.928	0.774	380.0	4.874
Min Cost Flow:0-14	0.879	203.0	4.919	0.794	347.0	4.899
Min Cost Flow:0-15	0.862	232.0	4.902	0.804	330.0	4.905
Min Cost Flow:0-16	0.856	243.0	4.905	0.856	242.0	4.903
Min Cost Flow:0-17	0.844	262.0	4.904	0.866	225.0	4.904
Min Cost Flow:0-18	0.874	212.0	4.914	0.829	288.0	4.916
Min Cost Flow:0-19	0.867	224.0	4.919	0.851	251.0	4.903
Min Cost Flow:0-20	0.844	262.0	4.91	0.807	325.0	4.905
Min Cost Flow:1-6	0.844	262.0	4.227	-	-	-
Min Cost Flow:1-7	0.867	224.0	4.698	-	-	-
Min Cost Flow:1-8	0.866	225.0	4.798	-	-	-
Min Cost Flow:1-9	0.876	209.0	4.892	-	-	-
Min Cost Flow:1-10	0.89	185.0	4.91	-	-	-
Min Cost Flow:1-11	0.894	178.0	4.917	-	-	-
Min Cost Flow:1-12	0.897	174.0	4.931	0.861	234.0	4.923
Min Cost Flow:1-13	0.861	233.0	4.904	0.832	283.0	4.895
Min Cost Flow:1-14	0.851	250.0	4.906	0.857	240.0	4.883
Min Cost Flow:1-15	0.914	145.0	4.921	0.863	231.0	4.904

Table B.7: Min cost flow bounding method on MovieLens : Baseline - UserBased

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Min Cost Flow:1-16	0.861	234.0	4.901	0.866	225.0	4.925
Min Cost Flow:1-17	0.81	320.0	4.923	0.85	252.0	4.914
Min Cost Flow:1-18	0.863	231.0	4.916	0.866	225.0	4.902
Min Cost Flow:1-19	0.86	235.0	4.899	0.857	241.0	4.906
Min Cost Flow:1-20	0.861	233.0	4.904	0.841	267.0	4.896
Min Cost Flow Threshold:0-7	0.747	426.0	4.39	-	-	-
Min Cost Flow Threshold:0-8	0.797	341.0	4.738	-	-	-
Min Cost Flow Threshold:0-9	0.826	293.0	4.821	-	-	-
Min Cost Flow Threshold:0-10	0.844	262.0	4.908	-	-	-
Min Cost Flow Threshold:0-11	0.85	253.0	4.912	-	-	-
Min Cost Flow Threshold:0-12	0.844	263.0	4.896	0.671	554.0	4.645
Min Cost Flow Threshold:0-13	0.839	271.0	4.908	0.773	382.0	4.874
Min Cost Flow Threshold:0-14	0.842	265.0	4.896	0.801	334.0	4.912
Min Cost Flow Threshold:0-15	0.867	223.0	4.905	0.801	334.0	4.908
Min Cost Flow Threshold:0-16	0.861	234.0	4.895	0.817	307.0	4.912
Min Cost Flow Threshold:0-17	0.866	226.0	4.909	0.872	215.0	4.927
Min Cost Flow Threshold:0-18	0.837	275.0	4.894	0.822	300.0	4.892
Min Cost Flow Threshold:0-19	0.857	241.0	4.912	0.809	321.0	4.921
Min Cost Flow Threshold:0-20	0.858	239.0	4.909	0.849	254.0	4.922
Min Cost Flow Threshold:1-7	0.86	235.0	4.709	-	-	-
Min Cost Flow Threshold:1-8	0.863	231.0	4.818	-	-	-
Min Cost Flow Threshold:1-9	0.87	218.0	4.891	-	-	-
Min Cost Flow Threshold:1-10	0.897	174.0	4.926	-	-	-
Min Cost Flow Threshold:1-11	0.876	208.0	4.928	-	-	-
Min Cost Flow Threshold:1-12	0.876	208.0	4.921	0.89	185.0	4.906
Min Cost Flow Threshold:1-13	0.868	222.0	4.902	0.865	227.0	4.91
Min Cost Flow Threshold:1-14	0.867	224.0	4.895	0.864	228.0	4.904
Min Cost Flow Threshold:1-15	0.854	246.0	4.927	0.85	252.0	4.906
Min Cost Flow Threshold:1-16	0.873	213.0	4.908	0.857	240.0	4.925
Min Cost Flow Threshold:1-17	0.847	257.0	4.912	0.846	259.0	4.917
Min Cost Flow Threshold:1-18	0.86	235.0	4.904	0.84	269.0	4.929
Min Cost Flow Threshold:1-19	0.874	212.0	4.913	0.834	280.0	4.913
Min Cost Flow Threshold:1-20	0.875	211.0	4.891	0.836	276.0	4.907

Table B.8: Min cost flow bounding method on MovieLens : Baseline - UserBased (contd.)

Recommender	ID@10	IC@10	APR@10	ID@20	IC@20	APR@20
Base	0.894	179.0	4.678	0.894	179.0	4.661
Random	0.848	256.0	4.396	0.839	270.0	4.386
Pseudo Gradient Descent	0.778	460.0	4.412	0.731	647.0	4.263
Greedy	0.779	465.0	4.4	0.783	463.0	4.389
Min Cost Flow:0-6	0.605	665.0	3.711	-	-	-
Min Cost Flow:0-7	0.699	507.0	3.901	-	-	-
Min Cost Flow:0-8	0.743	433.0	4.02	-	-	-
Min Cost Flow:0-9	0.776	377.0	4.104	-	-	-
Min Cost Flow:0-10	0.782	367.0	4.123	-	-	-
Min Cost Flow:0-11	0.768	390.0	4.126	-	-	-
Min Cost Flow:0-12	0.783	365.0	4.141	0.677	543.0	4.076
Min Cost Flow:0-13	0.772	384.0	4.18	0.737	443.0	4.127
Min Cost Flow:0-14	0.779	372.0	4.197	0.773	382.0	4.112
Min Cost Flow:0-15	0.809	321.0	4.202	0.751	418.0	4.135
Min Cost Flow:0-16	0.81	319.0	4.189	0.784	363.0	4.137
Min Cost Flow:0-17	0.816	309.0	4.199	0.781	369.0	4.134
Min Cost Flow:0-18	0.833	281.0	4.189	0.804	329.0	4.152
Min Cost Flow:0-19	0.826	293.0	4.233	0.804	329.0	4.19
Min Cost Flow:0-20	0.851	251.0	4.2	0.791	352.0	4.167
Min Cost Flow:1-6	0.784	363.0	3.79	-	-	-
Min Cost Flow:1-7	0.797	341.0	3.937	-	-	-
Min Cost Flow:1-8	0.854	245.0	4.108	-	-	-
Min Cost Flow:1-9	0.829	288.0	4.121	-	-	-
Min Cost Flow:1-10	0.85	253.0	4.193	-	-	-
Min Cost Flow:1-11	0.801	335.0	4.166	-	-	-
Min Cost Flow:1-12	0.802	333.0	4.162	0.818	306.0	4.162
Min Cost Flow:1-13	0.82	303.0	4.206	0.787	358.0	4.142
Min Cost Flow:1-14	0.797	341.0	4.191	0.8	337.0	4.153
Min Cost Flow:1-15	0.808	323.0	4.173	0.81	320.0	4.154
Min Cost Flow:1-16	0.826	292.0	4.22	0.8	337.0	4.146
Min Cost Flow:1-17	0.837	274.0	4.199	0.809	321.0	4.164
Min Cost Flow:1-18	0.833	281.0	4.236	0.791	352.0	4.173
Min Cost Flow:1-19	0.828	289.0	4.216	0.825	294.0	4.234
Min Cost Flow:1-20	0.804	329.0	4.197	0.816	309.0	4.188

Table B.9: Min cost flow bounding method on MovieLens : Baseline - ALSWR

APPENDIX C

Tables - Gradient Descent Method

The following are the results on selected parameter settings of the gradient descent method outlined in Section 7.2.

K	Beta	Sigma	Alpha	Eta	ID	IC	APR
50	3.0	0.5	0.7	0.1	0.2447	1581.0	3.7797
50	3.0	0.5	0.6	0.9	0.2563	1581.0	3.7793
50	3.0	0.1	0.1	0.2	0.3342	1266.0	3.6547
50	3.0	0.1	0.1	0.1	0.3688	1279.0	3.6541
50	3.0	0.5	0.7	0.2	0.3905	1589.0	3.7806
50	3.0	0.1	0.1	0.3	0.4341	1257.0	3.6532
50	3.0	0.5	0.7	0.3	0.4286	1593.0	3.7768
50	3.0	0.1	0.1	0.4	0.4309	1270.0	3.6543
50	3.0	0.1	0.1	0.5	0.4378	1275.0	3.654
50	3.0	0.5	0.7	0.4	0.4519	1582.0	3.778
50	3.5	0.9	0.5	0.6	0.4673	1633.0	4.3085
50	3.5	0.9	0.5	0.5	0.4724	1630.0	4.3062
50	3.5	0.4	0.9	0.1	0.4683	1394.0	4.0205
50	3.5	0.4	0.9	0.2	0.4771	1393.0	4.021
50	3.5	0.9	0.5	0.7	0.4736	1634.0	4.3087
50	3.5	0.4	0.9	0.3	0.4783	1380.0	4.022
50	3.5	0.9	0.5	0.8	0.4796	1629.0	4.3065
50	3.5	0.4	0.9	0.4	0.4824	1388.0	4.0225
50	3.5	0.9	0.5	0.9	0.4799	1633.0	4.3122
50	3.5	0.4	0.8	0.9	0.4811	1389.0	4.0222
50	4.0	0.8	0.7	0.9	0.4881	1548.0	4.5974
50	4.0	0.8	0.8	0.1	0.4872	1544.0	4.6018
50	4.0	0.8	0.7	0.8	0.4888	1527.0	4.5948
50	4.0	0.8	0.8	0.2	0.4884	1547.0	4.6003
50	4.0	0.4	0.9	0.3	0.4927	1085.0	4.3728
50	4.0	0.4	0.9	0.2	0.4929	1080.0	4.3682
50	4.0	0.8	0.8	0.3	0.4899	1532.0	4.5989

Table C.1: Gradient Descent Method on MovieLens Dataset

K	Beta	Sigma	Alpha	Eta	ID	IC	APR
50	4.0	0.4	0.9	0.1	0.4916	1103.0	4.3707
50	4.0	0.8	0.8	0.4	0.4919	1525.0	4.6021
50	4.0	0.4	0.9	0.4	0.4934	1068.0	4.3711
100	3.0	0.1	0.2	0.5	0.2738	1341.0	3.7276
100	3.0	0.1	0.3	0.1	0.2288	1342.0	3.7286
100	3.0	0.1	0.1	0.3	0.3254	1342.0	3.7255
100	3.0	0.1	0.1	0.7	0.3826	1342.0	3.7272
100	3.0	0.1	0.1	0.6	0.4131	1332.0	3.7263
100	3.0	0.1	0.1	0.8	0.4231	1337.0	3.7287
100	3.0	0.1	0.2	0.9	0.4276	1339.0	3.7267
100	3.0	0.1	0.2	0.2	0.4323	1334.0	3.7274
100	3.0	0.1	0.1	0.5	0.449	1333.0	3.7264
100	3.0	0.1	0.2	0.7	0.4425	1340.0	3.7253
100	3.5	0.1	0.1	0.2	0.5016	994.0	3.9843
100	3.5	0.1	0.2	0.4	0.2959	998.0	3.9882
100	3.5	0.1	0.3	0.3	0.3193	1002.0	3.9888
100	3.5	0.1	0.2	0.3	0.3667	996.0	3.9876
100	3.5	0.1	0.1	0.4	0.3947	1005.0	3.9882
100	3.5	0.1	0.1	0.3	0.4365	990.0	3.9887
100	3.5	0.1	0.1	0.6	0.4346	986.0	3.9869
100	3.5	0.1	0.1	0.8	0.4322	1004.0	3.9878
100	3.5	0.1	0.1	0.5	0.4452	978.0	3.9869
100	3.5	0.1	0.1	0.9	0.4492	1002.0	3.9867
100	4.0	0.1	0.2	0.1	0.7308	557.0	4.2866
100	4.0	0.1	0.1	0.2	0.4818	542.0	4.287
100	4.0	0.1	0.1	0.3	0.4372	551.0	4.286
100	4.0	0.1	0.1	0.5	0.4158	557.0	4.2867
100	4.0	0.1	0.1	0.1	0.3991	554.0	4.2863
100	4.0	0.1	0.4	0.3	0.423	548.0	4.2853
100	4.0	0.1	0.2	0.7	0.4361	549.0	4.2866
100	4.0	0.1	0.2	0.6	0.4435	548.0	4.2844
100	4.0	0.1	0.1	0.8	0.4542	540.0	4.2843
100	4.0	0.1	0.1	0.9	0.4418	552.0	4.2858

Table C.2: Gradient Descent Method on MovieLens Dataset (contd.)

K	Beta	Sigma	Alpha	Eta	ID	IC	APR
150	3.0	0.1	0.1	0.7	0.2334	1412.0	3.8016
150	3.0	0.1	0.1	0.1	0.2624	1407.0	3.8009
150	3.0	0.1	0.1	0.4	0.3553	1411.0	3.8042
150	3.0	0.1	0.2	0.4	0.387	1408.0	3.8037
150	3.0	0.1	0.1	0.5	0.4207	1408.0	3.8019
150	3.0	0.1	0.3	0.2	0.4172	1410.0	3.8036
150	3.0	0.1	0.2	0.1	0.4278	1407.0	3.8004
150	3.0	0.1	0.2	0.6	0.4391	1398.0	3.8038
150	3.0	0.1	0.1	0.6	0.4419	1407.0	3.8011
150	3.0	0.1	0.4	0.4	0.4523	1407.0	3.8064
150	3.5	0.1	0.2	0.2	0.4388	1104.0	4.0419
150	3.5	0.1	0.1	0.4	0.2741	1104.0	4.042
150	3.5	0.1	0.3	0.1	0.3217	1107.0	4.043
150	3.5	0.1	0.2	0.1	0.3647	1113.0	4.0415
150	3.5	0.1	0.1	0.1	0.4051	1108.0	4.0416
150	3.5	0.1	0.2	0.7	0.4172	1107.0	4.0391
150	3.5	0.1	0.1	0.3	0.4254	1112.0	4.0392
150	3.5	0.1	0.2	0.3	0.4282	1116.0	4.0438
150	3.5	0.1	0.1	0.5	0.4536	1103.0	4.0418
150	3.5	0.1	0.3	0.2	0.4462	1114.0	4.0418
150	4.0	0.1	0.1	0.9	0.678	654.0	4.3294
150	4.0	0.1	0.2	0.3	0.455	666.0	4.3316
150	4.0	0.1	0.1	0.5	0.3864	671.0	4.33
150	4.0	0.1	0.1	0.6	0.3792	669.0	4.3332
150	4.0	0.1	0.1	0.3	0.4208	653.0	4.3313
150	4.0	0.1	0.4	0.3	0.41	664.0	4.3322
150	4.0	0.1	0.2	0.1	0.4284	659.0	4.3331
150	4.0	0.1	0.1	0.2	0.4314	660.0	4.3311
150	4.0	0.1	0.1	0.7	0.4432	662.0	4.3307
150	4.0	0.1	0.3	0.5	0.4487	664.0	4.3301

Table C.3: Gradient Descent Method on MovieLens Dataset (contd.)

K	Beta	Sigma	Alpha	Eta	ID	IC	APR
50	3.5	0.9	0.5	0.1	0.1914	1100.0	4.2243
50	3.5	0.9	0.9	0.1	0.2826	1101.0	4.2175
50	3.5	0.9	0.3	0.1	0.3471	1101.0	4.2251
50	3.5	0.9	0.1	0.1	0.3792	1101.0	4.2252
50	3.5	0.9	0.7	0.1	0.401	1101.0	4.2269
50	3.5	0.8	0.3	0.1	0.4189	1100.0	4.1554
50	3.5	0.8	0.7	0.1	0.4275	1101.0	4.1547
50	3.5	0.8	0.5	0.1	0.4515	1099.0	4.1583
50	3.5	0.8	0.1	0.1	0.4434	1101.0	4.155
50	3.5	0.8	0.9	0.1	0.4477	1100.0	4.1598
50	3.0	0.9	0.5	0.1	0.1281	1101.0	3.9067
50	3.0	0.9	0.1	0.1	0.2244	1101.0	3.9043
50	3.0	0.9	0.7	0.1	0.3552	1101.0	3.9042
50	3.0	0.9	0.9	0.1	0.3676	1101.0	3.9101
50	3.0	0.9	0.3	0.1	0.3863	1101.0	3.91
50	3.0	0.8	0.3	0.1	0.4163	1101.0	3.8391
50	3.0	0.8	0.7	0.1	0.4317	1101.0	3.8385
50	3.0	0.8	0.9	0.1	0.4483	1101.0	3.839
50	3.0	0.8	0.5	0.1	0.4423	1101.0	3.8407
50	3.0	0.8	0.1	0.1	0.4549	1101.0	3.8379
50	4.0	0.9	0.5	0.1	0.2638	1098.0	4.5875
50	4.0	0.9	0.3	0.2	0.2698	1089.0	4.5868
50	4.0	0.9	0.1	0.1	0.3426	1092.0	4.5845
50	4.0	0.9	0.9	0.1	0.3558	1098.0	4.5885
50	4.0	0.9	0.3	0.1	0.3826	1096.0	4.5872
50	4.0	0.9	0.5	0.2	0.4135	1097.0	4.5864
50	4.0	0.9	0.7	0.1	0.4398	1092.0	4.5905
50	4.0	0.9	0.1	0.2	0.4379	1095.0	4.5881
50	4.0	0.9	0.9	0.2	0.4478	1094.0	4.5835
50	4.0	0.9	0.7	0.2	0.45	1095.0	4.5873

Table C.4: Gradient Descent Method on Netflix Dataset

K	Beta	Sigma	Alpha	Eta	ID	IC	APR
100	3.0	0.9	0.3	0.1	0.1151	1101.0	3.9614
100	3.0	0.9	0.9	0.1	0.2273	1101.0	3.9633
100	3.0	0.9	0.1	0.1	0.3342	1101.0	3.9586
100	3.0	0.9	0.7	0.1	0.365	1101.0	3.9613
100	3.0	0.9	0.5	0.1	0.4027	1101.0	3.9601
100	3.0	0.8	0.5	0.1	0.4206	1101.0	3.8975
100	3.0	0.8	0.1	0.1	0.4272	1101.0	3.8954
100	3.0	0.8	0.7	0.1	0.4354	1101.0	3.8976
100	3.0	0.8	0.3	0.1	0.4396	1101.0	3.8927
100	3.0	0.8	0.9	0.1	0.4543	1101.0	3.8945
100	3.5	0.9	0.7	0.1	0.1723	1101.0	4.2692
100	3.5	0.9	0.3	0.1	0.2396	1101.0	4.2696
100	3.5	0.9	0.9	0.1	0.3375	1101.0	4.271
100	3.5	0.9	0.1	0.1	0.3782	1101.0	4.2625
100	3.5	0.9	0.5	0.1	0.3898	1101.0	4.2648
100	3.5	0.8	0.3	0.1	0.425	1101.0	4.1956
100	3.5	0.8	0.7	0.1	0.4269	1101.0	4.1944
100	3.5	0.8	0.9	0.1	0.4402	1101.0	4.1975
100	3.5	0.8	0.1	0.1	0.4551	1099.0	4.194
100	3.5	0.8	0.5	0.1	0.4429	1101.0	4.1962
100	4.0	0.9	0.9	0.1	0.2413	1097.0	4.6156
100	4.0	0.9	0.7	0.1	0.2828	1100.0	4.6136
100	4.0	0.9	0.1	0.1	0.3582	1095.0	4.6195
100	4.0	0.9	0.5	0.1	0.3962	1097.0	4.6202
100	4.0	0.9	0.3	0.1	0.4007	1098.0	4.621
100	4.0	0.9	0.1	0.2	0.4163	1097.0	4.6185
100	4.0	0.9	0.7	0.2	0.4225	1098.0	4.6166
100	4.0	0.9	0.5	0.2	0.4331	1099.0	4.6212
100	4.0	0.8	0.5	0.2	0.4615	1085.0	4.5513
100	4.0	0.8	0.9	0.2	0.4512	1093.0	4.5529

Table C.5: Gradient Descent Method on Netflix Dataset (contd.)

K	Beta	Sigma	Alpha	Eta	ID	IC	APR
150	3.0	0.9	0.5	0.1	0.1023	1101.0	4.0188
150	3.0	0.9	0.9	0.1	0.2256	1101.0	4.019
150	3.0	0.9	0.1	0.1	0.3413	1101.0	4.0196
150	3.0	0.9	0.3	0.1	0.3708	1101.0	4.0206
150	3.0	0.9	0.7	0.1	0.4011	1101.0	4.024
150	3.0	0.8	0.5	0.1	0.4026	1101.0	3.9551
150	3.0	0.8	0.3	0.1	0.4214	1101.0	3.9545
150	3.0	0.8	0.7	0.1	0.4342	1101.0	3.9582
150	3.0	0.8	0.1	0.1	0.4497	1101.0	3.9585
150	3.0	0.8	0.9	0.1	0.4507	1101.0	3.9565
150	3.5	0.9	0.9	0.1	0.1573	1101.0	4.3101
150	3.5	0.9	0.3	0.1	0.2599	1101.0	4.3087
150	3.5	0.9	0.5	0.1	0.3302	1101.0	4.3089
150	3.5	0.9	0.1	0.1	0.3777	1101.0	4.3069
150	3.5	0.9	0.7	0.1	0.3812	1101.0	4.3089
150	3.5	0.8	0.9	0.1	0.421	1101.0	4.2385
150	3.5	0.8	0.5	0.1	0.4299	1101.0	4.2354
150	3.5	0.8	0.7	0.1	0.4307	1100.0	4.2439
150	3.5	0.8	0.3	0.1	0.4545	1100.0	4.2392
150	3.5	0.8	0.1	0.1	0.4535	1101.0	4.2395
150	4.0	0.9	0.3	0.1	0.2239	1100.0	4.6531
150	4.0	0.9	0.7	0.1	0.2631	1100.0	4.6493
150	4.0	0.9	0.5	0.1	0.3265	1101.0	4.6509
150	4.0	0.9	0.1	0.1	0.3889	1100.0	4.654
150	4.0	0.8	0.7	0.2	0.4001	1097.0	4.5819
150	4.0	0.9	0.1	0.2	0.4145	1099.0	4.6501
150	4.0	0.8	0.1	0.2	0.4291	1101.0	4.5806
150	4.0	0.8	0.3	0.1	0.4375	1096.0	4.5836
150	4.0	0.8	0.9	0.1	0.4413	1098.0	4.5849
150	4.0	0.8	0.5	0.1	0.456	1098.0	4.5854

Table C.6: Gradient Descent Method on Netflix Dataset (contd.)

APPENDIX D

Graphs - Basic Min Cost Flow Method

The following plots show the variation of the performance of the min cost flow algorithm described in section 6.2 with different parameters.

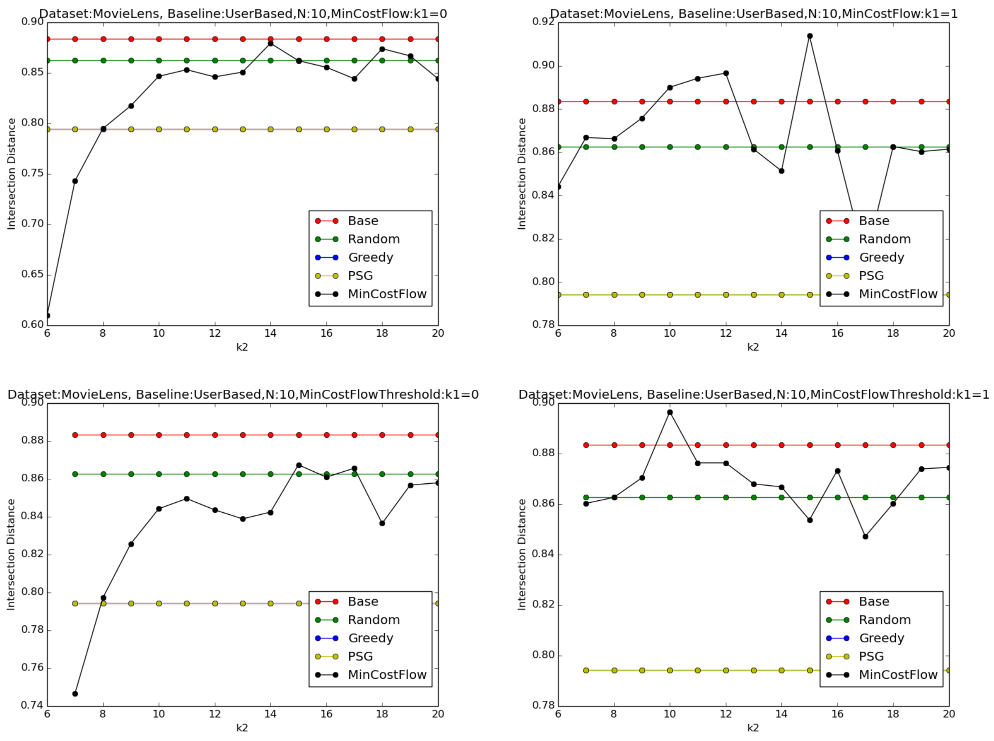

Figure D.1: Intersection Distance: Dataset=MovieLens, Baseline=UserBased, N=10

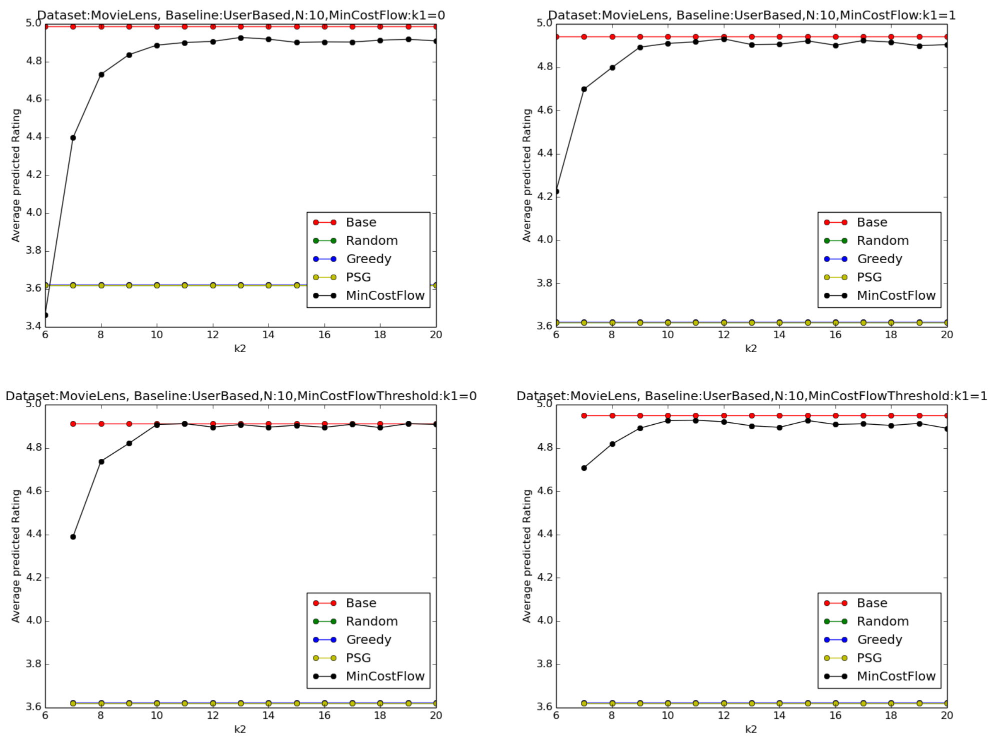

Figure D.2: Average Predicted Rating: Dataset=MovieLens, Baseline=UserBased, N=10

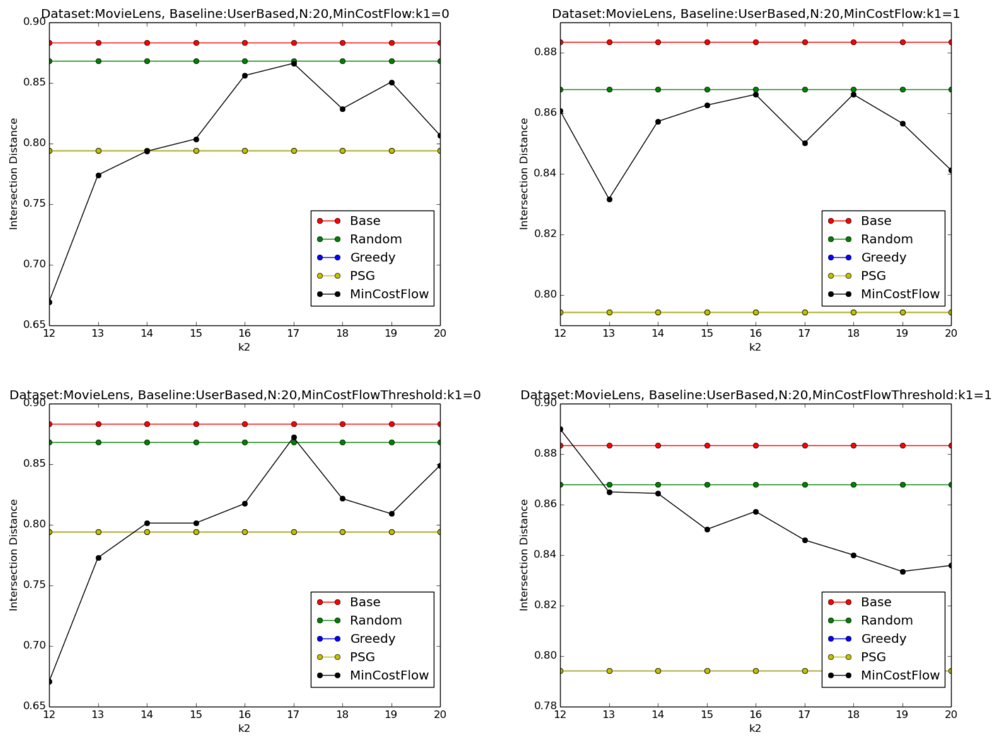

Figure D.3: Intersection Distance: Dataset=MovieLens, Baseline=UserBased, N=20

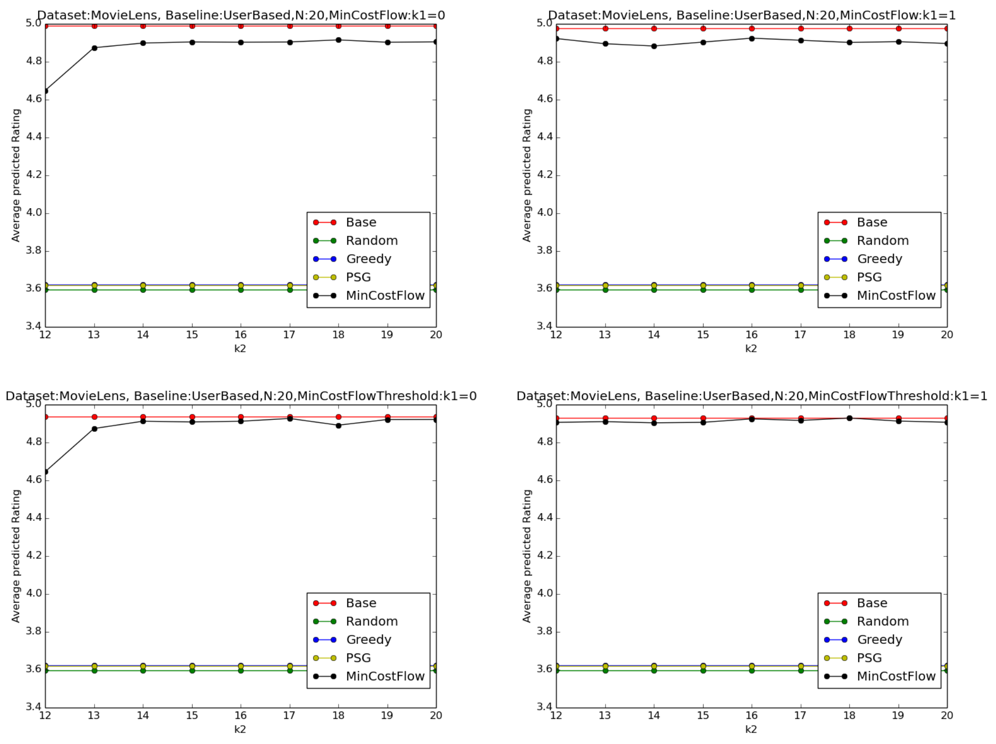

Figure D.4: Average Predicted Rating: Dataset=MovieLens, Baseline=UserBased, N=20

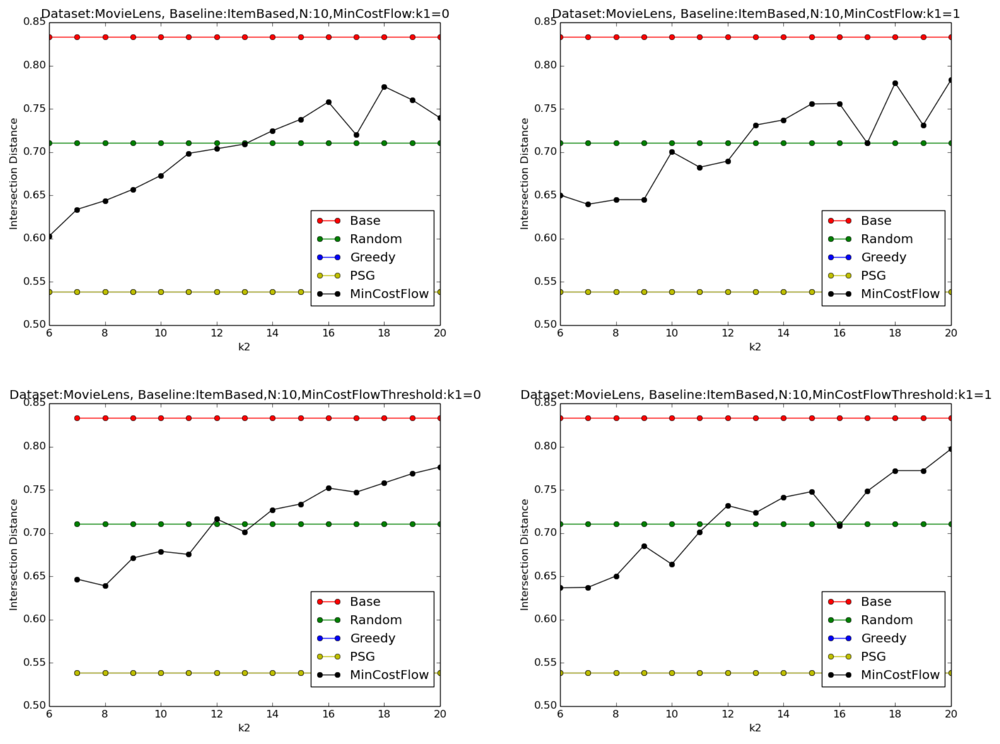

Figure D.5: Intersection Distance: Dataset=MovieLens, Baseline=ItemBased, N=10

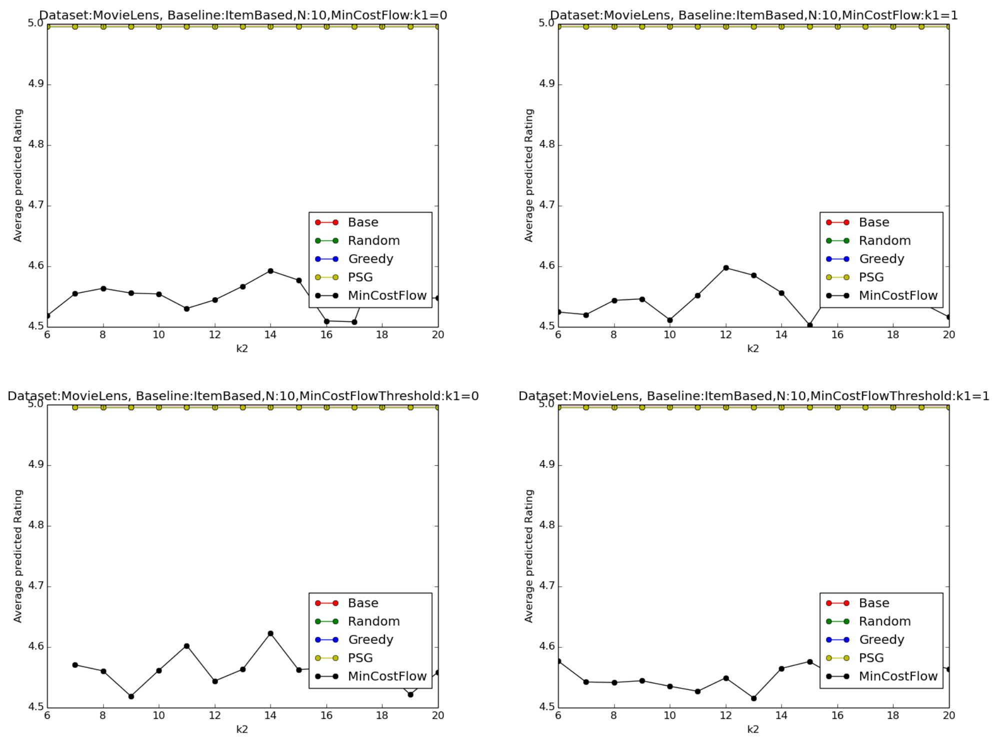

Figure D.6: Average Predicted Rating: Dataset=MovieLens, Baseline=ItemBased, N=10

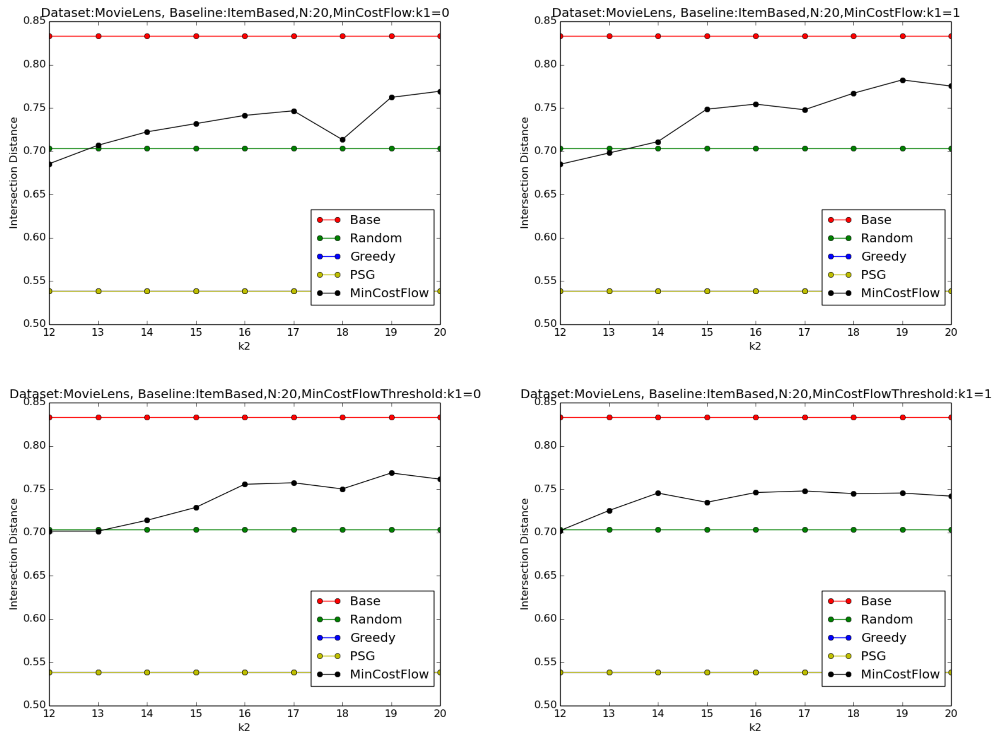

Figure D.7: Intersection Distance: Dataset=MovieLens, Baseline=ItemBased, N=20

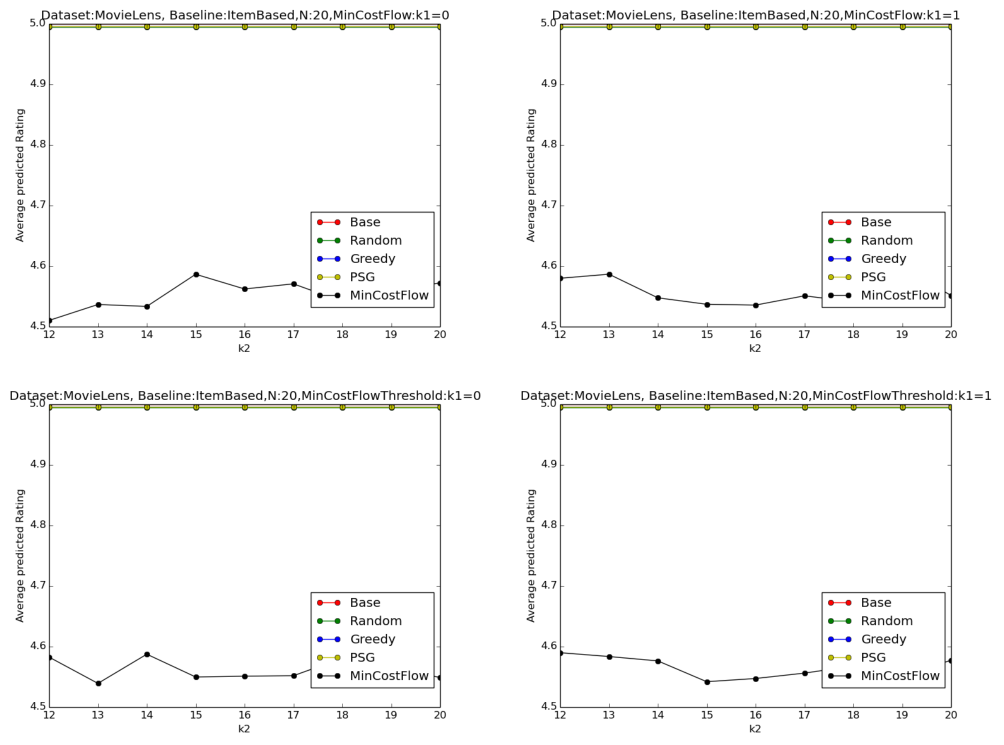

Figure D.8: Average Predicted Rating: Dataset=MovieLens, Baseline=ItemBased, N=20

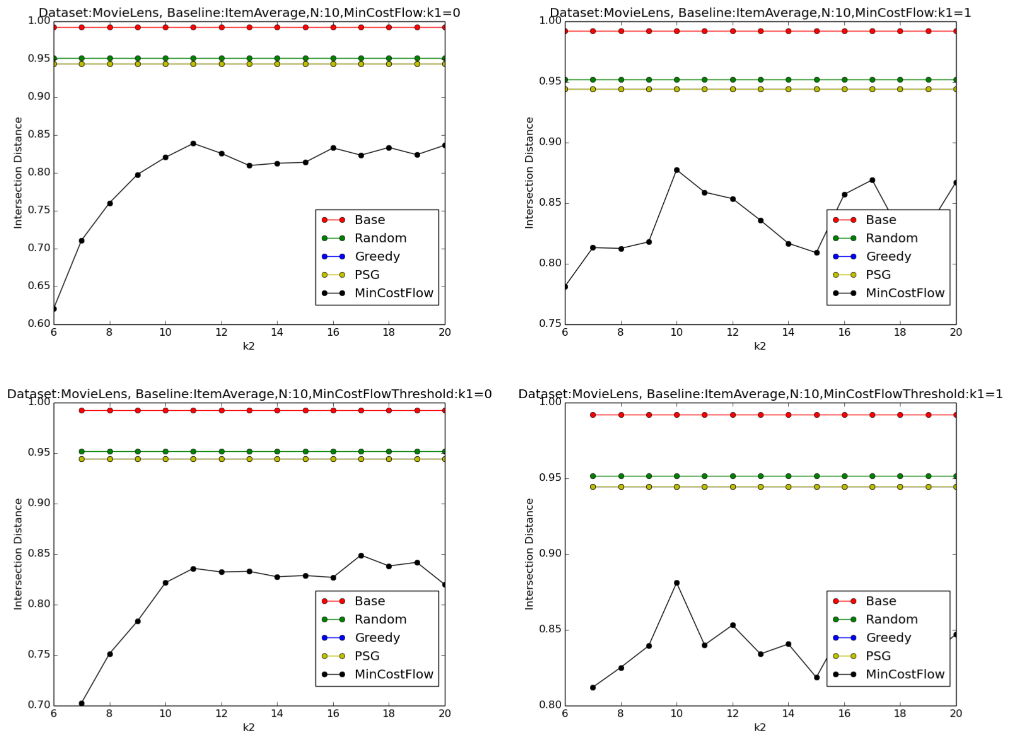

Figure D.9: Intersection Distance: Dataset=MovieLens, Baseline=ItemAverage, N=10

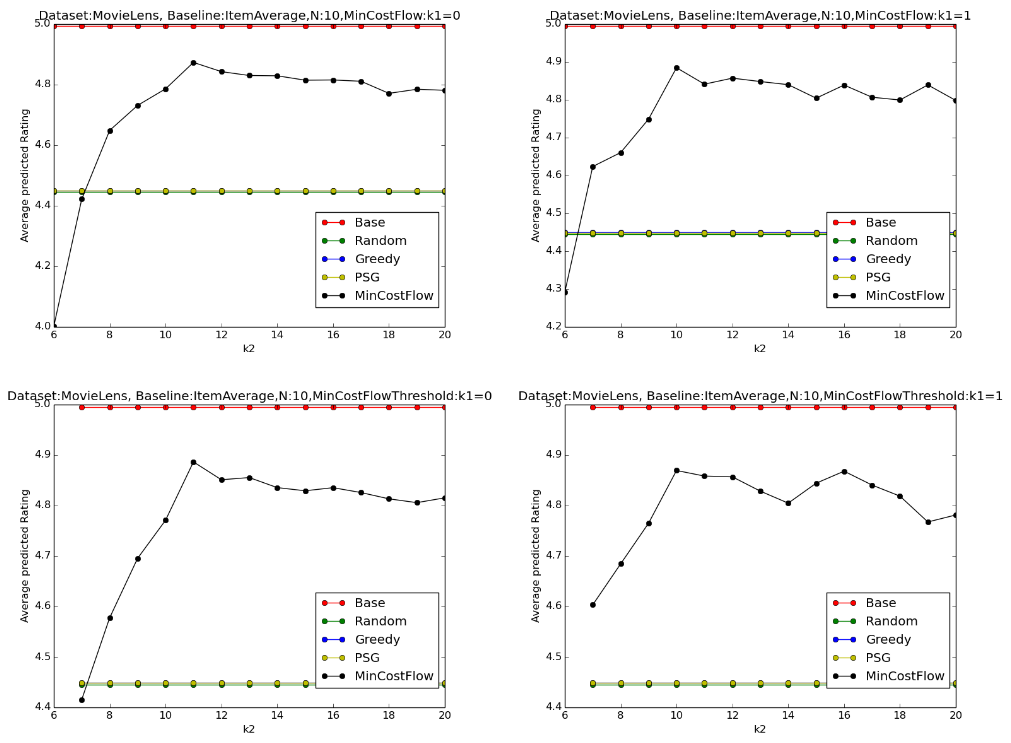

Figure D.10: Average Predicted Rating: Dataset=MovieLens, Baseline=ItemAverage, N=10

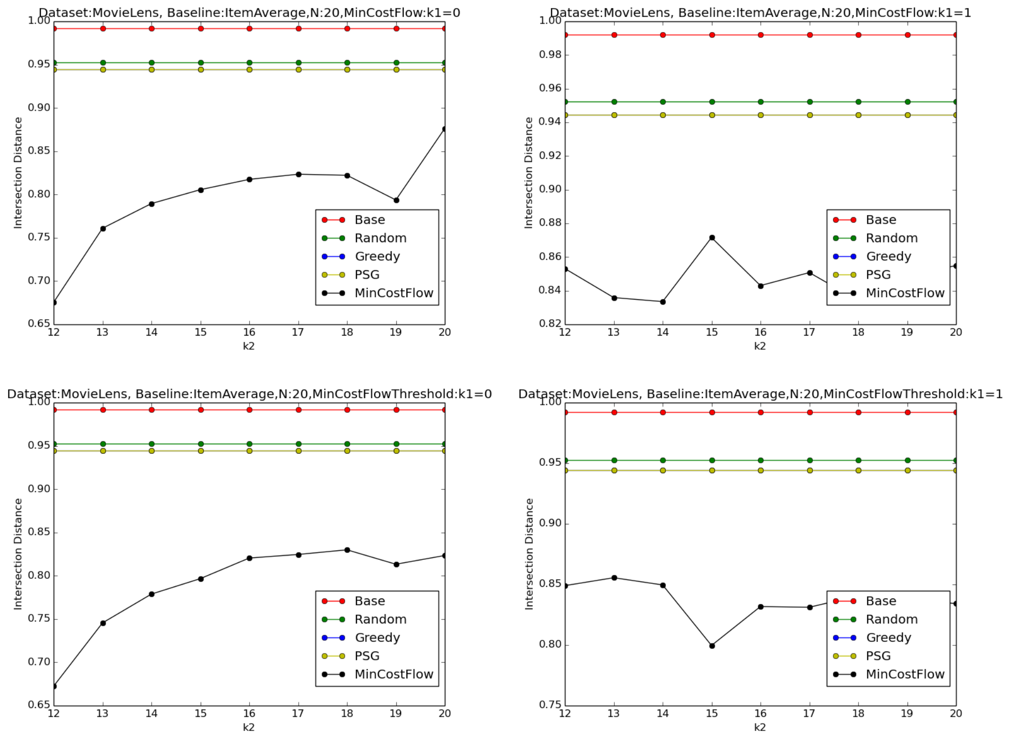

Figure D.11: Intersection Distance: Dataset=MovieLens, Baseline=ItemAverage, N=20

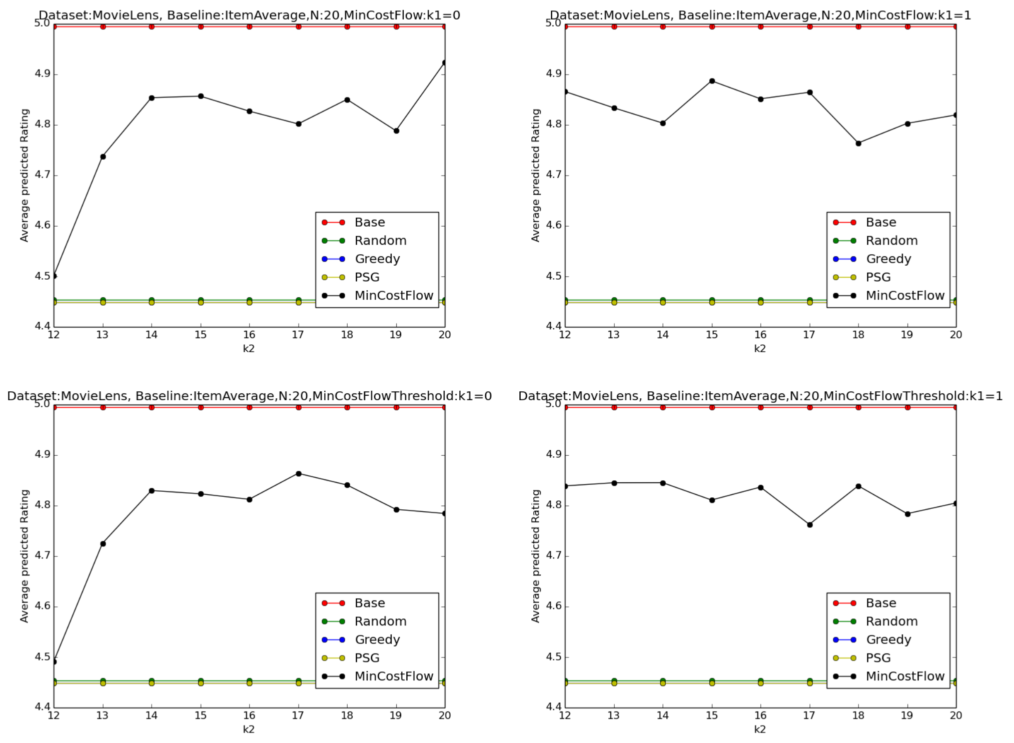

Figure D.12: Average Predicted Rating: Dataset=MovieLens, Baseline=ItemAverage, N=20

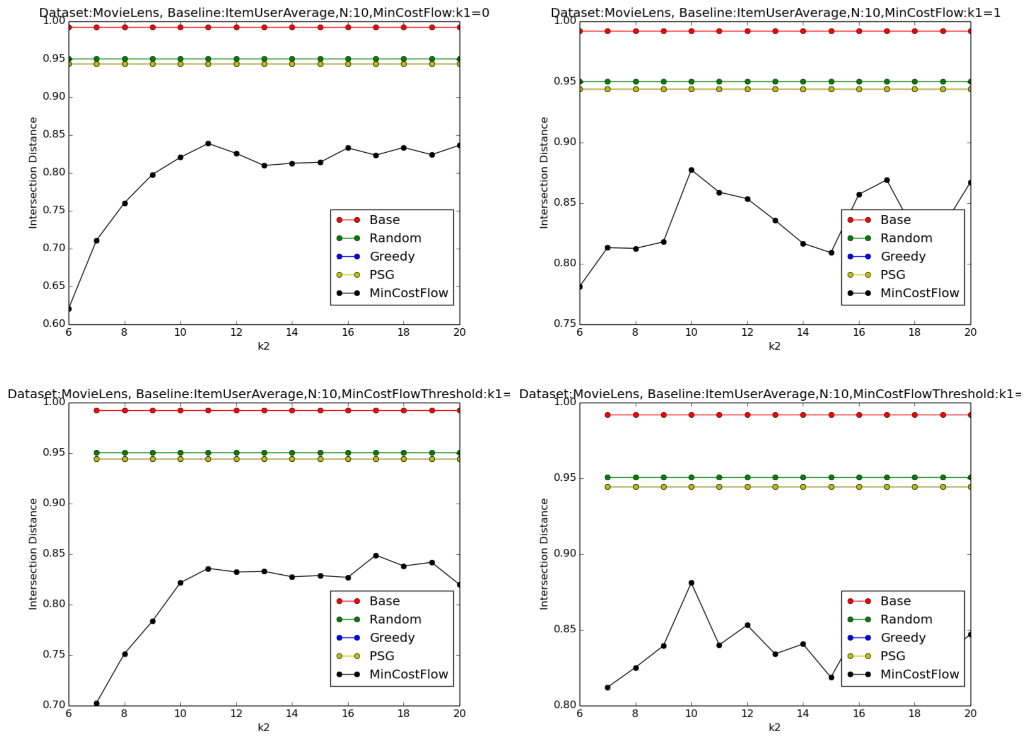

Figure D.13: Intersection Distance: Dataset=MovieLens, Baseline=ItemUserAverage, N=10

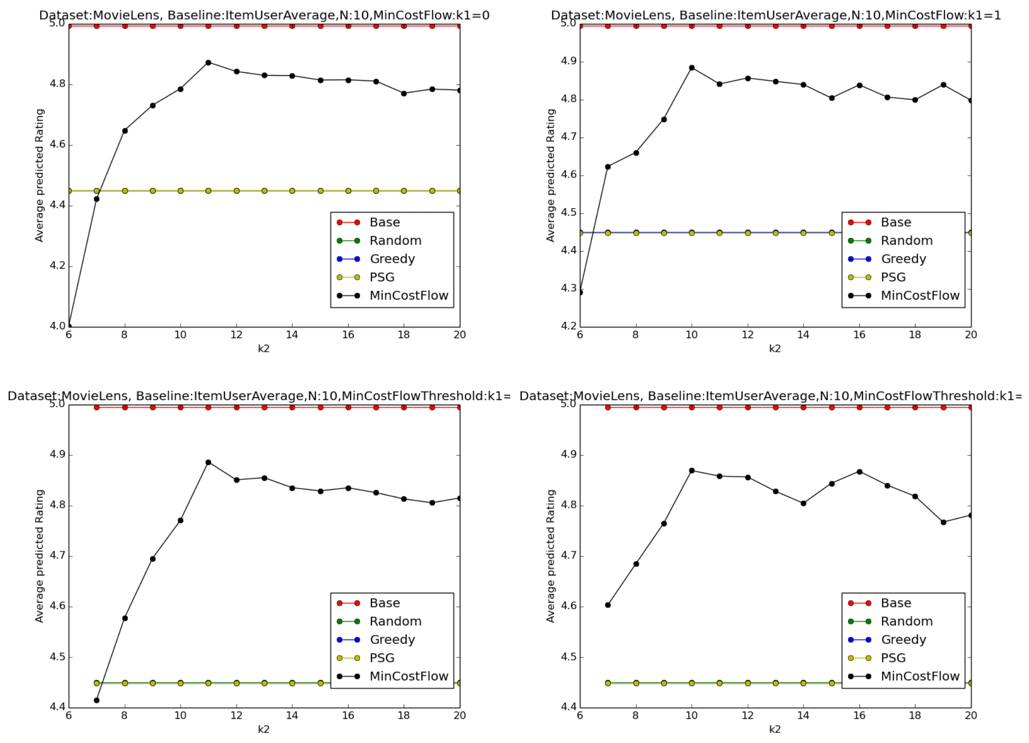

Figure D.14: Average Predicted Rating: Dataset=MovieLens, Baseline=ItemUserAverage, N=10

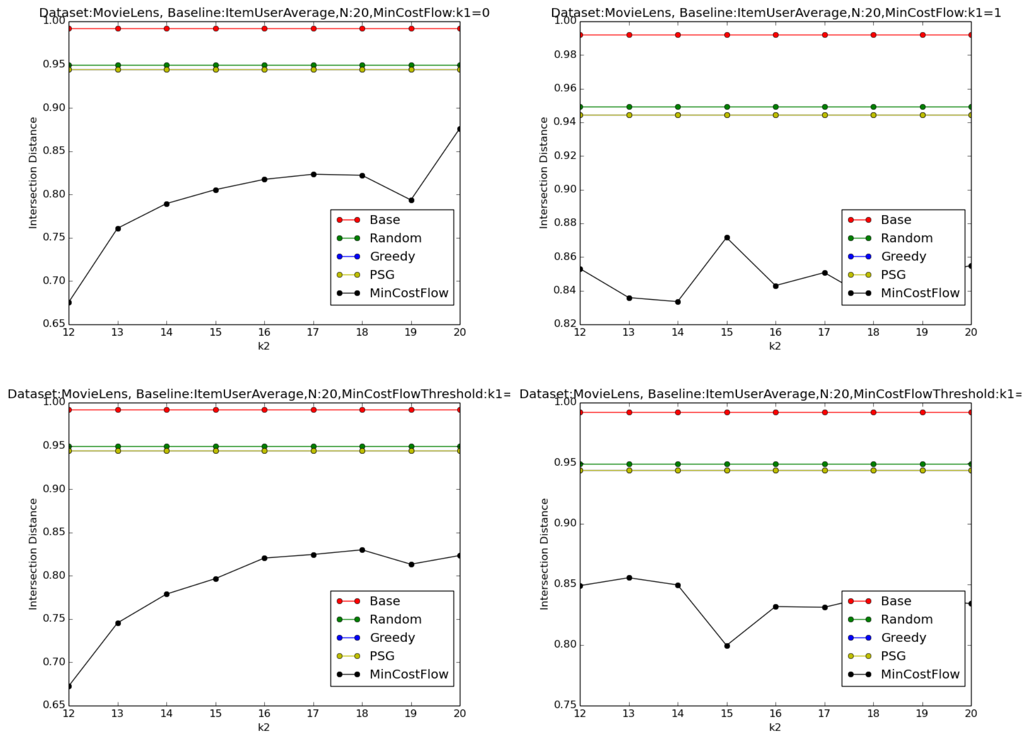

Figure D.15: Intersection Distance: Dataset=MovieLens, Baseline=ItemUserAverage, N=20

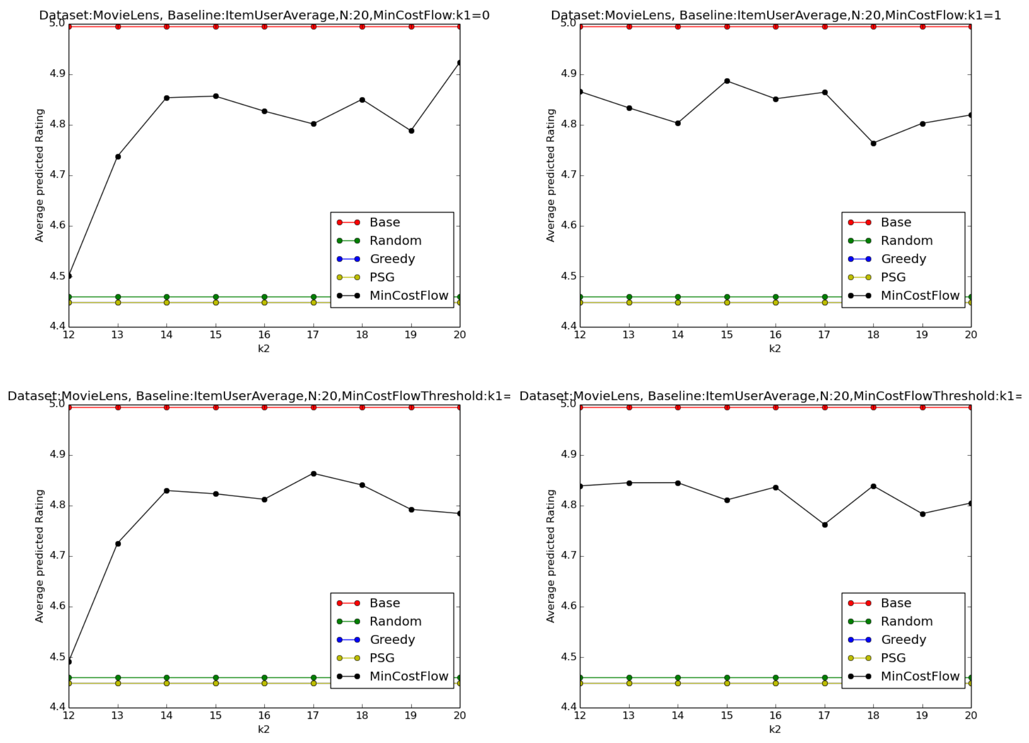

Figure D.16: Average Predicted Rating: Dataset=MovieLens, Baseline=ItemUserAverage, N=20

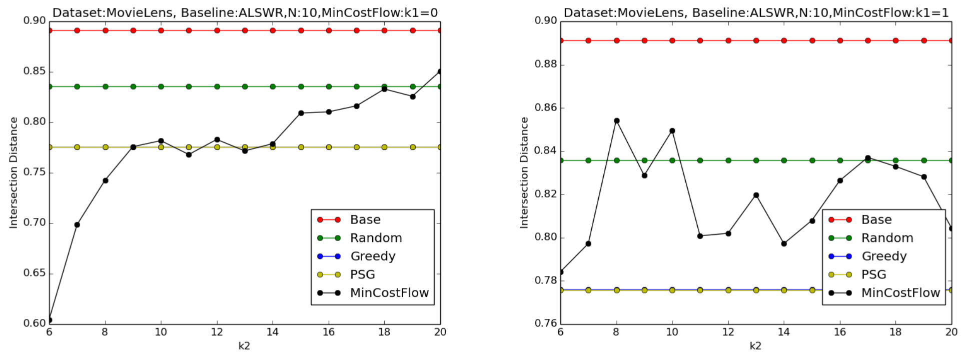

Figure D.17: Intersection Distance: Dataset=MovieLens, Baseline=ALSWR, N=10

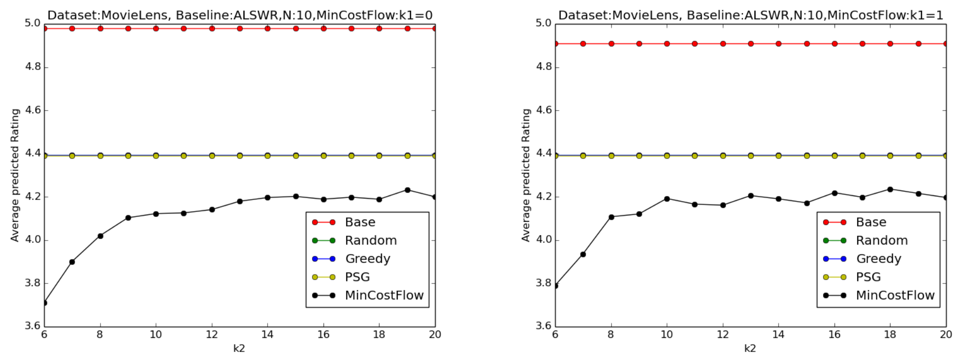

Figure D.18: Average Predicted Rating: Dataset=MovieLens, Baseline=ALSWR, N=10

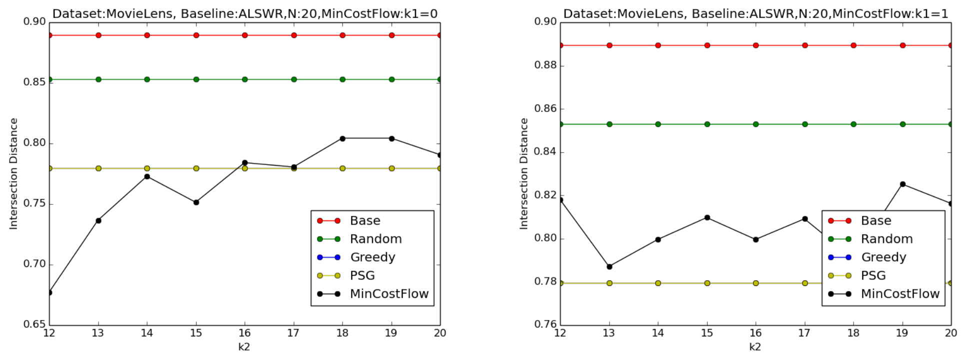

Figure D.19: Intersection Distance: Dataset=MovieLens, Baseline=ALSWR, N=20

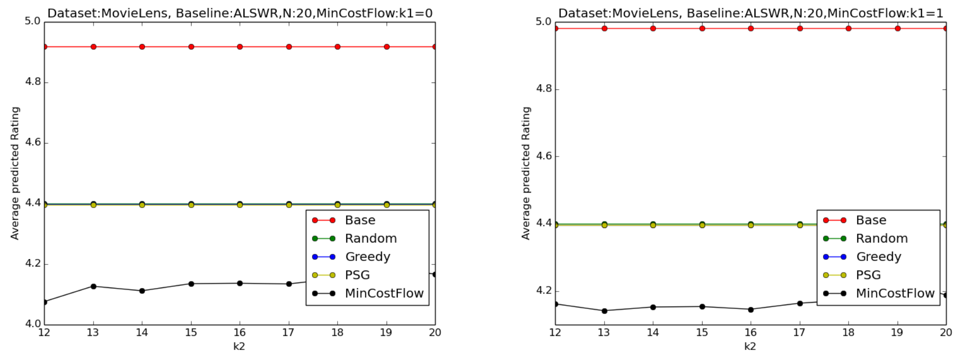

Figure D.20: Average Predicted Rating: Dataset=MovieLens, Baseline=ALSWR, N=20

APPENDIX E

Graphs - Dual Objective Min Cost Flow Method

The following plots show the variation of the performance of the dual objective optimizing min cost flow algorithm described in section 6.3 with the parameter λ_2 for different datasets, base recommenders and number of items per recommendation list, N.

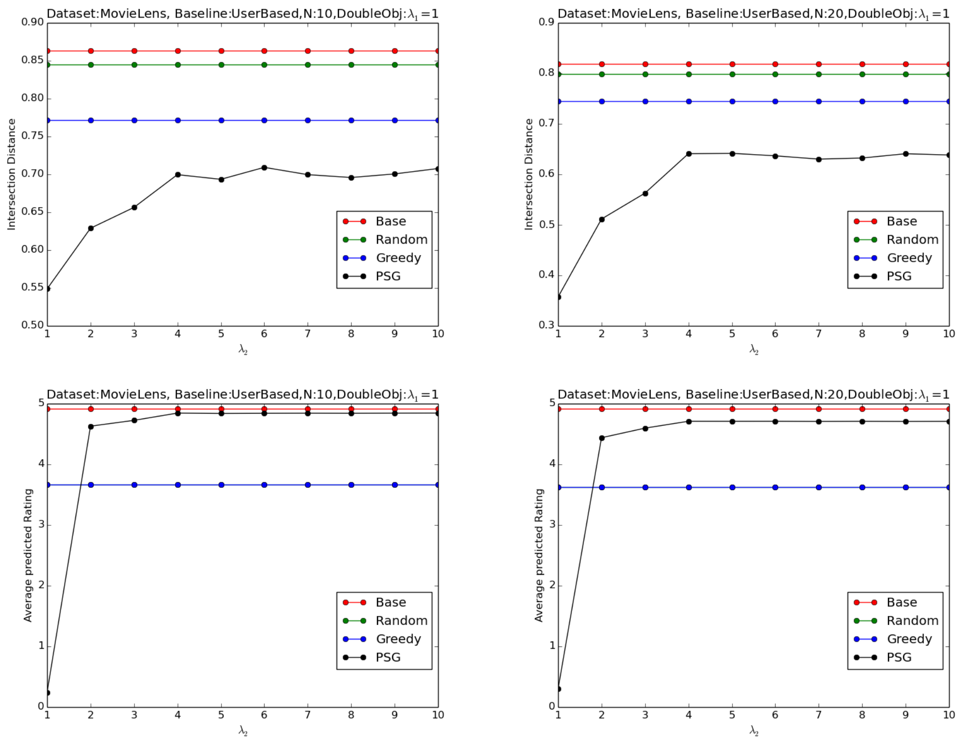

Figure E.1: Dataset=MovieLens, Baseline=UserBased

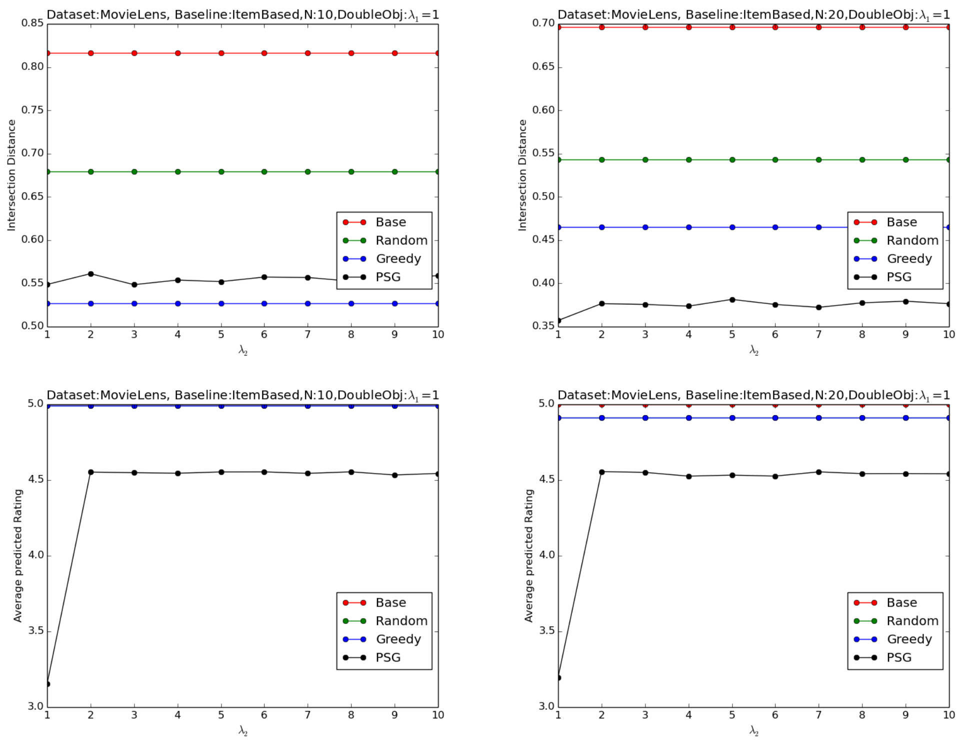

Figure E.2: Dataset=MovieLens, Baseline=ItemBased

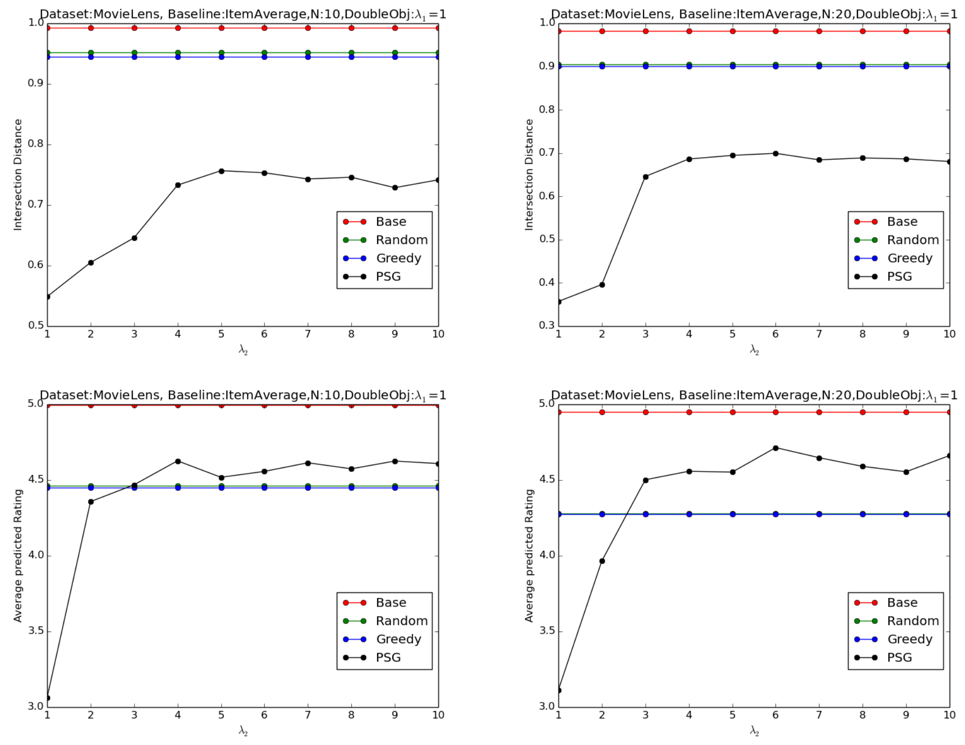

Figure E.3: Dataset=MovieLens, Baseline=ItemAverage

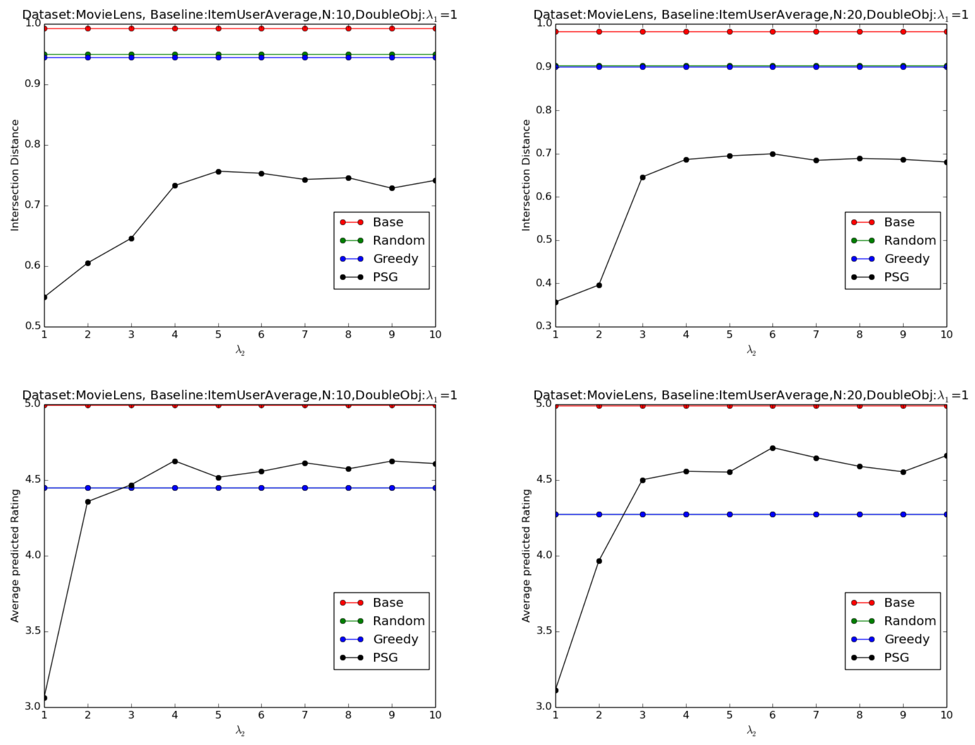

Figure E.4: Dataset=MovieLens, Baseline=ItemUserAverage

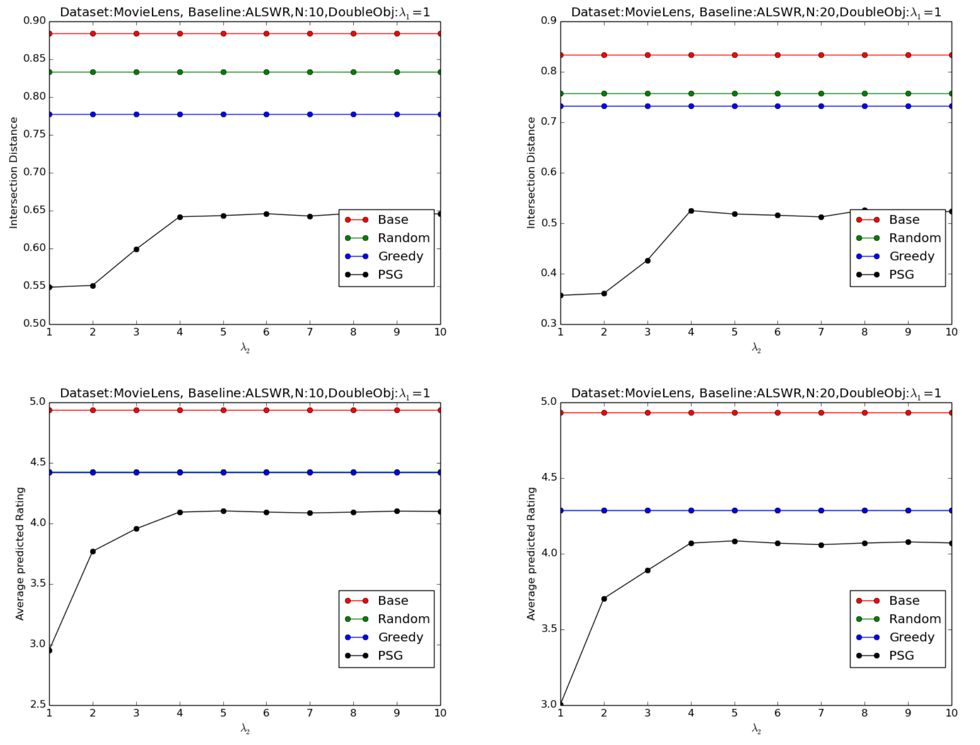

Figure E.5: Dataset=MovieLens, Baseline=ALSWR

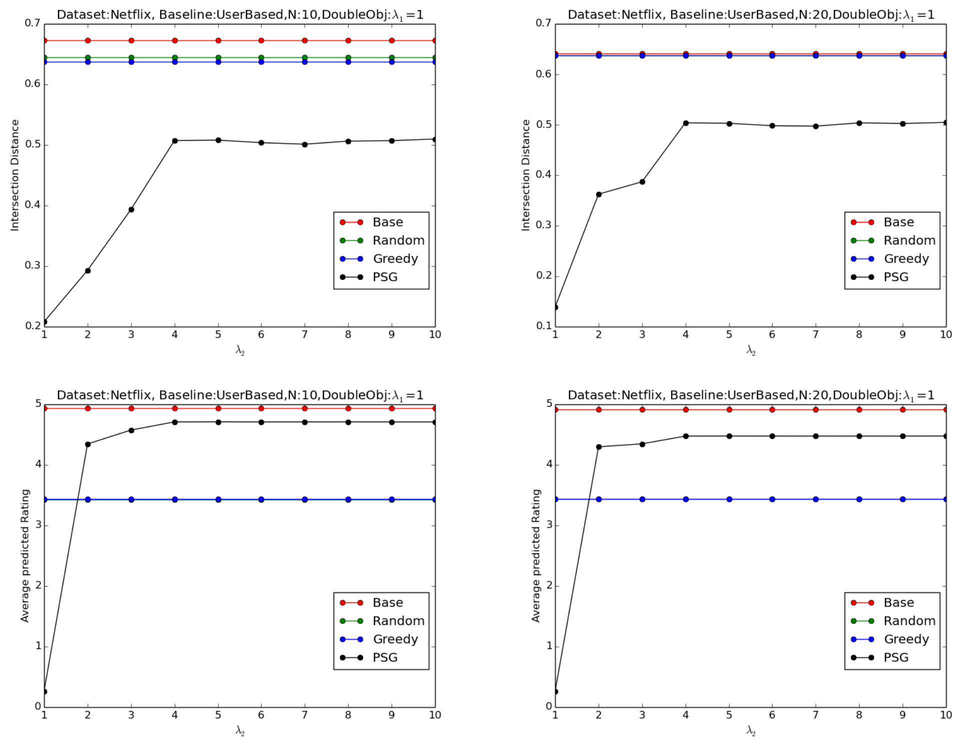

Figure E.6: Dataset=Netflix, Baseline=UserBased

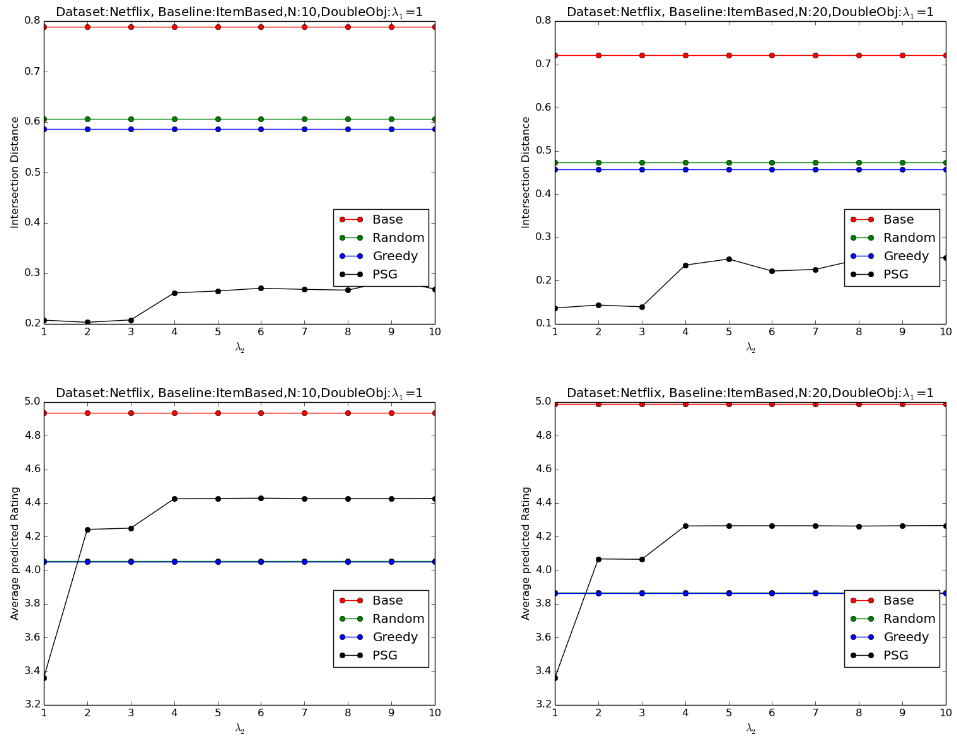

Figure E.7: Dataset=Netflix, Baseline=ItemBased

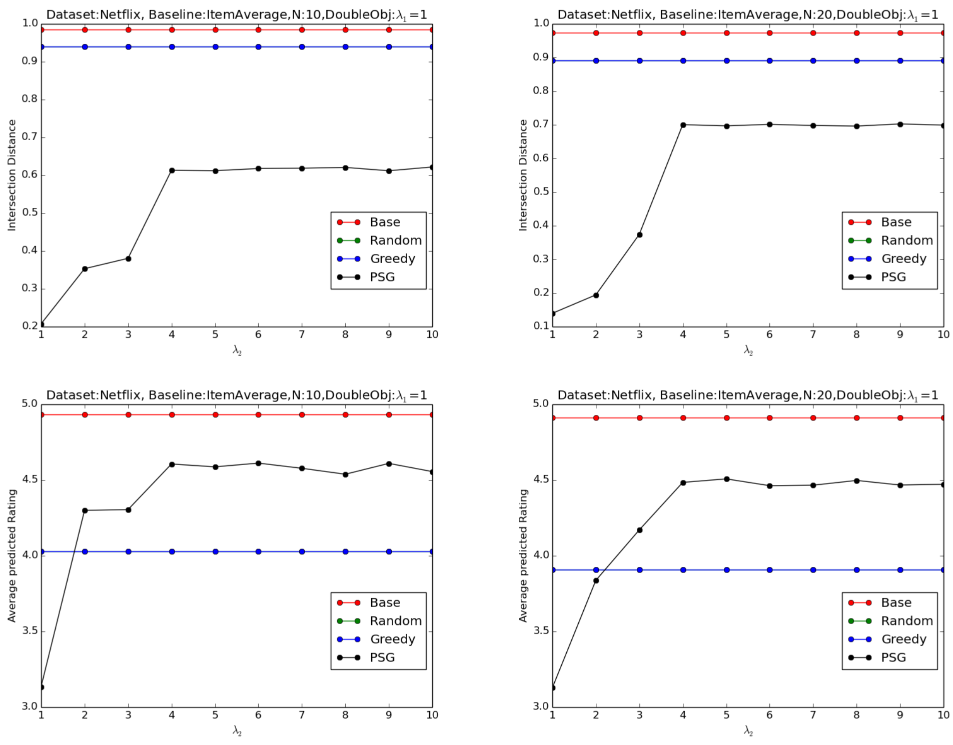

Figure E.8: Dataset=Netflix, Baseline=ItemAverage

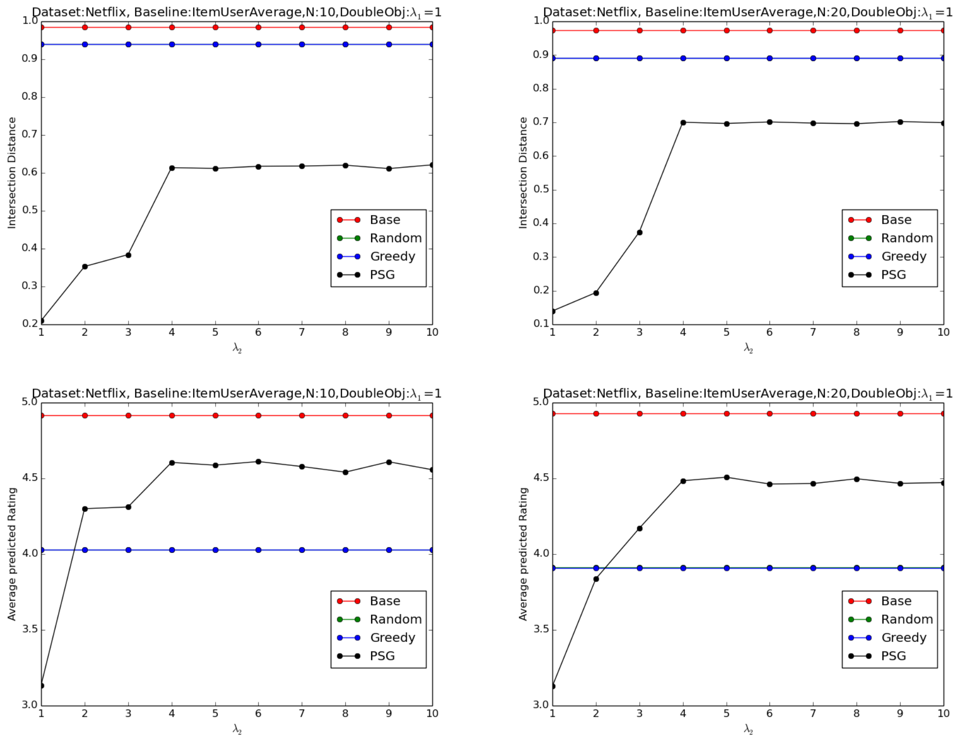

Figure E.9: Dataset=Netflix, Baseline=ItemUserAverage

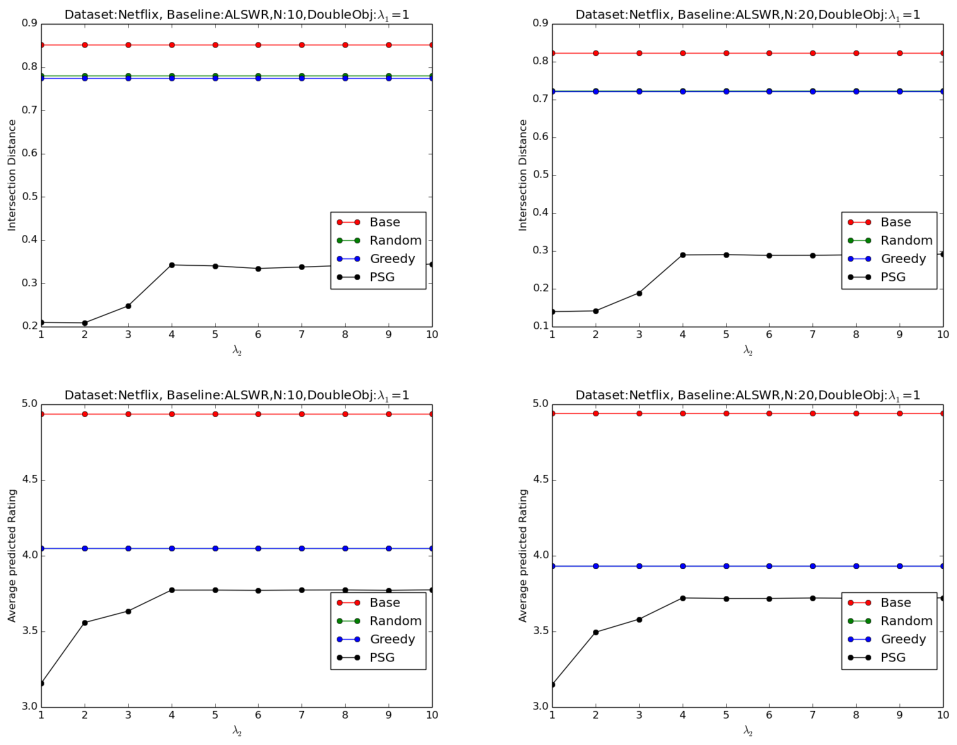

Figure E.10: Dataset=Netflix, Baseline=ALSWR

REFERENCES

[1] **Adomavicius, G.** and **Y. Kwon**, Maximizing aggregate recommendation diversity:a graph-theoretic approach. *In Workshop on Novelty and Diversity in Recommender Systems - DiveRS 2011, Proceedings of the Fifth ACM Conference on Recommender Systems*, RecSys '11. 2011.

[2] **Adomavicius, G.** and **Y. Kwon** (2012). Improving aggregate recommendation diversity using ranking-based techniques. *IEEE Trans. on Knowl. and Data Eng.*, **24**(5), 896–911. ISSN 1041-4347. URL `http://dx.doi.org/10.1109/TKDE.2011.15`.

[3] **Agrawal, R.**, **S. Gollapudi**, **A. Halverson**, and **S. Leong**, Diversifying search results. *In WSDM '09: Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, WSDM '09. 2009.

[4] **Balakrishnan, N.**, **S. Bendre**, and **H. Malik** (1992). General relations and identities for order statistics from non-independent non-identical variables. *Annals of the Institute of Statistical Mathematics*, **44**(1), 177–183. ISSN 0020-3157. URL `http://dx.doi.org/10.1007/BF00048680`.

[5] **Bartal, Y.**, **R. Gonen**, and **N. Nisan**, Incentive compatible multi unit combinatorial auctions. *In Proceedings of the 9th Conference on Theoretical Aspects of Rationality and Knowledge*, TARK '03. ACM, New York, NY, USA, 2003. ISBN 1-58113-731-1. URL `http://doi.acm.org/10.1145/846241.846250`.

[6] **Billsus, D.** and **M. Pazzani**, Learning collaborative information filters. *In Proceedings of the 15th International Conference on Machine Learning*, ICML '98. Morgan Kaufmann, 1998.

[7] **Bromiley, P.** (2014). Products and convolutions of gaussian probability density functions. Internal Report 2003-003, Imaging Sciences Research Group, Institute of Population Health, School of Medicine, University of Manchester.

[8] **Brynjolfsson, E.**, **Y. J. Hu**, and **M. D. Smith** (2010). Research commentary—long tails vs. superstars: The effect of information technology on product variety and sales concentration patterns. *Info. Sys. Research*, **21**(4), 736–747. ISSN 1526-5536. URL `http://dx.doi.org/10.1287/isre.1100.0325`.

[9] **Cao, G.** and **M. West** (1997). Computing distributions of order statistics. *Comunications in Statistics*, **26**, 755–764. URL `ftp://ftp.stat.duke.edu/pub/WorkingPapers/92-A02.ps`.

[10] **Carbonell, J.** and **J. Goldstein**, The use of mmr, diversity-based reranking for reordering documents and producing summaries. *In Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '98. 1998.

[11] **Castells, P.** and **S. Vargas**, Novelty and diversity metrics for recommender systems: Choice, discovery and relevance. *In Proceedings of International Workshop on Diversity in Document Retrieval (DDR 2011) at the 33rd European Conference on Information Retrieval (ECIR 2011)*. 2011.

[12] **Cha, S.-H.**, Fast image template and dictionary matching algorithms. *In Proceedings of ACCV 98 Lecture notes in Computer ScienceComputer Vision*, volume 1351 of *ACCV 98*. Springer, 1998.

[13] **Cha, S.-H.** and **S. N. Srihari** (2002). On measuring the distance between histograms. *The Journal of the Pattern Recognition Society*, **35**, 1355–1370.

[14] **Chandar, P.** and **B. Barterette**, Analysis of various evaluation measures for diversity. *In Proceedings of the 1st International Workshop on Diversity in Document Retrieval at European Conference on Information Retrieval*, ECIR '11. 2011.

[15] **Cohen, S.** and **L. Guibas**, The earth mover's distance under transformation sets. *In Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2*, ICCV '99. IEEE Computer Society, Washington, DC, USA, 1999. ISBN 0-7695-0164-8. URL `http://dl.acm.org/citation.cfm?id=850924.851526`.

[16] **Demange, G.**, **D. Gale**, and **M. Sotomayor** (1986). Multi-Item Auctions. *Journal of Political Economy*, **94**(4), 863–72. URL `http://ideas.repec.org/a/ucp/jpolec/v94y1986i4p863-72.html`.

[17] **Fiat, A.**, **S. Leonardi**, **J. Saia**, and **P. Sankowski**, Single valued combinatorial auctions with budgets. *In Proceedings of the 12th ACM Conference on Electronic Commerce*, EC '11. ACM, New York, NY, USA, 2011. ISBN 978-1-4503-0261-6. URL `http://doi.acm.org/10.1145/1993574.1993609`.

[18] **Fleder, D.** and **K. Hosanagar** (2009). Blockbuster culture's next rise or fall: The impact of recommender systems on sales diversity. *Manage. Sci.*, **55**(5), 697–712. ISSN 0025-1909. URL `http://dx.doi.org/10.1287/mnsc.1080.0974`.

[19] **Goldberg, A. V.** (1997). An efficient implementation of a scaling minimum-cost flow algorithm. *Journal of Algorithms*, **22**(1), 1 – 29.

[20] **Hale, T.** (). The theoretical basics of popular inequality measures. Technical report, University of Texas Inequality Project.

[21] **Hastie, T.**, **R. Tibshirani**, and **J. Friedman**, *The Elements of Statistical Learning*, chapter Additive Models, Trees, and Related Methods. Springer New York Inc., 2008.

[22] **Hurley, N.** and **M. Zhang** (2011). Novelty and diversity in top-n recommendation – analysis and evaluation. *ACM Trans. Internet Technol.*, **10**(4), 14:1–14:30. ISSN 1533-5399. URL `http://doi.acm.org/10.1145/1944339.1944341`.

[23] **Janjoom, A.** and **Z. Al-Saiyari** (2013). Moments of non-identical order statistics from burr xii distribution with gamma and normal outliers. *Journal of Mathematics and Statistics*, **9**(1), 51–61. ISSN 1549-3644.

[24] **Kawamae, N.**, Serendipitous recommendations via innovators. *In Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '10. ACM, New York, NY, USA, 2010. ISBN 978-1-4503-0153-4. URL `http://doi.acm.org/10.1145/1835449.1835487`.

[25] **Lathia, N.**, **S. Hailes**, **L. Capra**, and **X. Amatriain**, Temporal diversity in recommender systems. *In Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '10. ACM, New York, NY, USA, 2010. ISBN 978-1-4503-0153-4. URL `http://doi.acm.org/10.1145/1835449.1835486`.

[26] **Leyton-Brown, K.**, **Y. Shoham**, and **M. Tennenholtz**, An algorithm for multi-unit combinatorial auctions. *In In Proceedings of the National Conference on Artificial Intelligence (AAAI*. 2000.

[27] **McNee, S. M.**, **J. Riedl**, and **J. A. Konstan**, Being accurate is not enough: How accuracy metrics have hurt recommender systems. *In CHI '06 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '06. ACM, New York, NY, USA, 2006. ISBN 1-59593-298-4. URL `http://doi.acm.org/10.1145/1125451.1125659`.

[28] **Mei, Q.**, **J. Guo**, and **D. Radev**, Divrank: The interplay of prestige and diversity in information networks. *In Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10. ACM, New York, NY, USA, 2010. ISBN 978-1-4503-0055-1. URL `http://doi.acm.org/10.1145/1835804.1835931`.

[29] **Nadarajah, S.** and **S. Kotz** (2008). Exact distribution of the max/min of two gaussian random variables. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, **16**(2), 210–212. ISSN 1063-8210.

[30] **Nguyen, T. T.**, **P.-M. Hui**, **F. M. Harper**, **L. Terveen**, and **J. A. Konstan**, Exploring the filter bubble: The effect of using recommender systems on content diversity. *In Proceedings of the 23rd International Conference on World Wide Web*, WWW '14. ACM, New York, NY, USA, 2014. ISBN 978-1-4503-2744-2. URL `http://doi.acm.org/10.1145/2566486.2568012`.

[31] **Rajaraman, A.** and **J. D. Ullman**, *Mining of Massive Datasets*, chapter 9: Recommendation Systems. Cambridge University Press, New York, NY, USA, 2011. ISBN 1107015359, 9781107015357.

[32] **Rendle, S.**, **C. Freudenthaler**, **Z. Gantner**, and **L. Schmidt-Thieme** (2012). Bpr: Bayesian personalized ranking from implicit feedback. *CoRR*, **abs/1205.2618**.

[33] **Rubner, Y.**, **C. Tomasi**, and **L. J. Guibas**, A metric for distributions with applications to image databases. *In Proceedings of the Sixth International Conference on Computer Vision*, ICCV '98. IEEE Computer Society, Washington, DC, USA, 1998. ISBN 81-7319-221-9. URL `http://dl.acm.org/citation.cfm?id=938978.939133`.

[34] **Salakhutdinov, R.** and **A. Mnih**, Probabilistic matrix factorization. *In Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007*. 2007. URL `http://papers.nips.cc/paper/3208-probabilistic-matrix-factorization`.

[35] **Salakhutdinov, R.** and **A. Mnih**, Bayesian probabilistic matrix factorization using markov chain monte carlo. *In Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*. 2008. URL `http://doi.acm.org/10.1145/1390156.1390267`.

[36] **Sandoval, S. V.** (2012). *Novelty and Diversity Enhancement and Evaluation in Recommender Systems*. Ph.D. thesis, Universidad Autónoma de Madrid.

[37] **Shoham, Y.** and **K. Leyton-Brown**, *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*, chapter Protocols for Multiagent Resource Allocation: Auctions. Cambridge University Press, New York, NY, USA, 2008. ISBN 0521899435.

[38] **Vargas, S.** and **P. Castells**, Rank and relevance in novelty and diversity metrics for recommender systems. *In Proceedings of the Fifth ACM Conference on Recommender Systems*, RecSys '11. ACM, New York, NY, USA, 2011. ISBN 978-1-4503-0683-6. URL `http://doi.acm.org/10.1145/2043932.2043955`.

[39] **Zhang, M.** and **N. Hurley**, Statistical modeling of diversity in top-n recommender systems. *In Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Volume 01*, WI-IAT '09. IEEE Computer Society, Washington, DC, USA, 2009. ISBN 978-0-7695-3801-3. URL `http://dx.doi.org/10.1109/WI-IAT.2009.83`.

[40] **Zhou, T.**, **Z. Kuscsik**, **J.-G. Liu**, **M. Medo**, **J. R. Wakeling**, and **Y.-C. Zhang**, Solving the apparent diversity-accuracy dilemma of recommender systems. *In Proceedings of the National Academy of Sciences of the United States of America*. 2010.

[41] **Zhou, Y.**, **D. Wilkinson**, **R. Schreiber**, and **R. Pan**, Large-scale parallel collaborative filtering for the netflix prize. *In Proceedings of the 4th International Conference on Algorithmic Aspects in Information and Management*, AAIM '08. Springer-Verlag, Berlin, Heidelberg, 2008. ISBN 978-3-540-68865-5. URL `http://dx.doi.org/10.1007/978-3-540-68880-8_32`.

[42] **Zhu, X.**, **A. B. Goldberg**, **J. Van**, and **G. D. Andrzejewski**, Improving diversity in ranking using absorbing random walks. *In Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL-HLT' 07. 2007.

[43] **Ziegler, C.-N.**, **S. M. McNee**, **J. A. Konstan**, and **G. Lausen**, Improving recommendation lists through topic diversification. *In Proceedings of the 14th International Conference on World Wide Web*, WWW '05. ACM, New York, NY, USA, 2005. ISBN 1-59593-046-9. URL `http://doi.acm.org/10.1145/1060745.1060754`.