ROBUST LOW-RANK MATRIX FACTORIZATION WITH MISSING DATA BY MINIMIZING L1 LOSS APPLIED TO COLLABORATIVE FILTERING

by

Shama Mehnaz Huda

Bachelor of Science in Electrical Engineering, University of Arkansas, 2009

Submitted to the Graduate Faculty of

Swanson School of Engineering in partial fulfillment

of the requirements for the degree of

Master of Science

University of Pittsburgh

2011

UNIVERSITY OF PITTSBURGH

SWANSON SCHOOL OF ENGINEERING

This thesis was presented

by

Shama Mehnaz Huda

It was defended on

December 1, 2011

and approved by

Zhi-Hong Mao, Ph.D., Associate Professor, Department of Electrical and Computer Engineering & Department of Bioengineering

Ching-Chung Li, Ph.D., Professor, Department of Electrical and Computer Engineering & Department of Computer Science

Yiran Chen, Ph.D., Assistant Professor, Department of Electrical and Computer Engineering

Thesis Advisor: Zhi-Hong Mao, Ph.D., Associate Professor, Department of Electrical and Computer Engineering & Department of Bioengineering

ROBUST LOW-RANK MATRIX FACTORIZATION WITH MISSING DATA BY MINIMIZING L1 LOSS APPLIED TO COLLABORATIVE FILTERING

Shama Mehnaz Huda, M.S.

University of Pittsburgh, 2011

In this age of information overload and plethora of choices, people increasingly rely on automatic recommender systems to tell them what suits their needs. A very effective approach for creating recommender systems is collaborative filtering, which is the task of predicting the preference/rating that a user would assign to an item based on preference data of that user and preference data of other users. One way to conduct collaborative filtering is through dimensionality reduction. The underlying concept of the approach lies in the belief that there are only a few features (reduced dimensions) that influence the user's choice. In this paper we use low rank matrix factorization for dimensionality reduction. Singular Value Decomposition (SVD), which is minimizing the L_2 norm is the most popular technique to perform matrix factorization. However, in most recommendation system data sets, often the users only rate a small amount of items, which creates missing data. As a result SVD fails. In recent years L1 norm has gained much importance and popularity because it is robust to outliers and missing data. In this thesis we use alternate convex optimization to perform L1 norm minimization to solve the matrix factorization problem and apply it to collaborative filtering. We also review some of the major challenges that collaborative filtering faces today and some of the other techniques used. Additionally, this thesis discusses the importance and future of collaborative filtering in medical applications that concerns the database of patient history (prescriptions/symptoms) and how it can be used as a predictive task for the future of the patient.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my thesis advisor Dr. Zhi Hong Mao who agreed to advise me and generously provided invaluable advice during my research. His expertise on the topic helped me navigate through the hurdles efficiently. His independent and supportive attitude encouraged me to carefully search and find a topic that is of value and interest to me. I would also like to thank Dr. Ching-Chung Li and Dr. Yiran Chen for generously taking valuable time out of their schedules and sitting on the committee for my thesis. In addition, I want to thank Dr. Hong Koo Kim who had advised me during my first year research at the opto-electronics lab, under whom I learned a multitude of skills. I would like to thank the GroupLens Research Project at the University of Minnesota for the use of MovieLens dataset.

I would like to thank my manager at Philips Respironics, Greg Matthews who has allowed me to work on my thesis while also having a full time job. Without his support and understanding, the completion of this project would not be feasible. In addition, I would also like to thank my colleagues in the Advanced Algorithms Group at Philips for providing support and encouragement.

On a personal note, I would like to sincerely thank my parents Huda and Lisa who have taught me to work hard and have guided me to formulate solid goals in life. I would like to thank my sister Nazia, my Uncle Mustafa and my friend Meenal, all of whom have provided tremendous support, motivation and encouragement throughout my graduate school career.

1.0 INTRODUCTION

Information overload has become an increasingly concerning problem. Navigating through the enormous database of online shopping catalogues, movie choices, books, articles etc. is becoming laborious and convoluted. Intelligent search engines like Google and Bing have somewhat alleviated the issue by introducing customized searches. However, people are increasingly relying on automated recommender systems to tell them what to buy or what they need. This asks for accurate and reliable automated recommender systems that can deal with an enormous database. A very popular way to implement such a recommender system is by using collaborative filtering. The task of collaborative filtering is the task of predicting the preference a user assigns to items based on preference data of that user and preference data of other users.

In Chapter 2 of this thesis we first go through the formulation of collaborative filtering. We then review some of the main algorithms used to perform collaborative filtering including memory-based algorithms, model-based algorithms and hybrid algorithms. Of these methods, the model-based matrix factorization is of particular interest to us as we will be using to for our algorithm. In addition we review and discuss some of the main challenges faced by the collaborative filtering. The biggest problem faces is the sparse nature of the data. Our algorithm is designed to handle the sparsity of this data well. Some other challenges include synonymy, scalability, shilling attacks etc.

In Chapter 3 we study the details of matrix factorizations and finding the low rank approximation of different measures of discrepancies. The concept of maximum likelihood estimation is discussed with a Gaussian distribution (L2) noise model and a Laplacian

distribution (L1) noise model. The resulting loss functions and optimization problems are discussed. Finally, the recent popularity of L1 norm minimization because of its robustness to missing data and outliers is discussed along with its applications.

In Chapter 4 we explain the alternate convex optimization using the L1 loss function algorithm from [1]. We also explain the use of Principal Components Analysis as a tool for reducing dimensions and coming up with a rank. We explain the methods used to deal with missing data. The experimental protocols and error metrics used are also explained. Finally the results are presented and explained.

In chapter 5 we introduce the importance of collaborative filtering and automated recommender systems in the medical field. Enormous healthcare costs, difficulty in keeping track of patients with chronic disease are only some of the factors that concern the healthcare community today. The current healthcare system is very reactive in that it is employed after the patient shows symptoms and gets the disease. We discuss ways in which the healthcare system can end up being more proactive in that it will recognize the onset of disease and risk based on historical patient data. We review some of the automated recommendation work for the medical field that exists in literature today. Since I work in a medical device company, we also discuss some of the future work that can be done in respiratory medical devices. There is large therapy data available in these devices from patients, which we want to be able to use to predict prescription pressure for future patients.

2.0 COLLABORATIVE FILTERING

The task of collaborative filtering is the task of predicting the preference a user assigns to items based on preference data of that user and preference data of other users. Collaborative filtering is most often used for recommending systems for books, movies, webpages, articles etc. In the following sections we will go through the formulation and explain the different methods used and the challenges faced in the field.

2.1 FORMULATION

There are a large number of information filtering problems associated with collaborative filtering research. It is important to have a proper formulation of these problems that are structures well. Marlin [2] explained the formulation and we will review that below. This is what we will use in out thesis work.

Three independent characteristics are important for the space formulation, the type of preference indicators used, the inclusion of additional features and the treatment of preference dynamics. A different choice for each of these characteristics will yield a different kind of formulation.

The primary kinds of preference indicators used for collaborative filtering are numerical ratings triplets, numerical rating vectors, co-occurrence pairs, and count vectors. A rating triplet is of form (*u, i, r*) where *u* is a user index, *i* is an item index, and *r* is a rating value. It means that

user *u* gave the item *i* a rating of *r*, where the value of r can be be ordinal or continuous. A numerical rating vector is of the form $r_u = (r_1^u, \dots, r_N^u)$, where r_i^u is the rating assigned by user *u* to item *i* and it can be ordinal or continuous. Co-occurrence pairs have the form (*u*, *i*) where *u* is a user index and *i* is an item index. This is slightly different than the others. This implies that the user viewed, accessed or purchased item *i*. It could also mean that user *u* likes item *i*. A count vector can be used to see how many times user *u* has viewed the particular item.

Another important difference between preference vectors is whether they are explicitly provided with by the user or whether they were implicitly acquired while the user browsed and clicked on internet sites that interested them. A good comparison between explicit ratings and implicit rating are shown in Claypool [8]. A user needs to provide some added effort to provide an explicit rating whereas an implicit rating is collected. Claypool says the benefit of explicit ratings should outweigh the effort users put in to rate. This thesis uses MovelLens data provided by GroupLens which consists of explicit ratings.

As will be mentioned later in this chapter, there is something called content based and hybrid collaborative filtering which allows the use of additional features to perform the prediction task. In a non-content based pure approach, users are described by their preferences for items and items are described by user's preferences for them. This pure approach is what is used in this thesis. However, it is important to describe the other methods used. The additional features when used include demographical information about users, such as age, gender, occupation etc. Similarly for items there can be additional information such as artist and genre for music, genre, director and actors for music. The hybrid approach is good for two well know problems in the field of collaborative filtering, known as the cold start problem and the new user

problem. These will be explained in more detail in the following section. Pure formulations are simpler and more popularly used.

Another interesting angle to collaborative filtering is the sequence at which the preference indicators are collected. In most cases, preference indicators are viewed as static set of values. However when the datasets are collected over long period of times, the user preferences becomes highly dynamic and older preferences indicators can become irrelevant and inaccurate. This can end up causing inaccurate predictions as well. When implicit preference users are used, this is a serious problem because users cannot update their preference indicators. The pros to design algorithms to deal with dynamic preferences are that the predictions will adapt over time. However, they also make the models very complex. Sequential formulations are more complicated to design compared to the non-sequential formulations. A maximum entropy method was proposed in [9] for sequential formulations. They introduce a method based on mixtures of first order Markov chains for learning dynamic user profiles.

In this thesis the, collaborative filtering formulation is pure, non-sequential and only rating based. No additional features are included. Users and items are only described by preference indicators. This approach is attractive because it has been subject to a lot of research previously.

The primary task of collaborative filtering is recommendation. If you are given the rating vector r_u of m users and the rating vector of a particular active user x, r_x, then the task would be to recommend a set of items for active user x. The task of recommendation is basically the same as the task of prediction because recommendations are produced from a set of predictions. As a result, if there is a method for predicting ratings for items that have not been rated, a recommendation method can be built by first computing the predictions for the active user's

unrated items, then sorting them and recommending the top items. The research for recommendation methods are thus geared towards creating accurate rating prediction methods.

2.2 METHODS

The approaches to perform collaborative filtering can be divided into three sections – memory based techniques, model based techniques and hybrid techniques. These methods are comprehensively described in [4]. In the following sections we will provide a concise review of these methods and mention some of the work available in literature for each of the methods.

2.2.1 Memory-based

Memory based CF algorithms use the full or a subset of the database of users and items to make the prediction. It is assumed that each user is part of a group of people with similar interests. The key is to identify these similar users or neighbors of the active user and then based on those similar users make a prediction of new items for the active user. The memory based algorithm thus implements the neighborhood-based algorithm, which calculates the similarity or weight between two users or items. This represents distance, correlation or weight. Then the algorithm produces a prediction for the active user by taking the weighted average of all ratings of the user or item or item on a certain item or user. Or it can also use a simple weighted average. To

generate the top-N recommendation, the *k* most similar users or items need to be found after computing the similarities. Then the similar users and items need to be aggregated to get the top-N most frequent items as the recommendation.

The similarity computation step between users and items is very important for memory based CF algorithms. For a user-based CF algorithm, the similarity between two users who have rated the same items is calculated. Similarly for an item-based CF algorithm, the similarity computation between two items is to first work on the users who have rated both of these items and then to apply a similarity computation between the two co-rated items if the users [10]. There are many multiple ways to compute these similarity measures. In a correlation based similarity, the Pearson correlation is used, which measures how much two variables linearly relate to each other [11]. Another way to compute similarity is using the vector cosine-based similarity. In this case, two items or users can be treated as a vector of ratings.

The prediction and recommendation step is the most important step of the collaborative filtering algorithm. A subset of the nearest neighbors of the active user is chosen based on the similarity with him/her. A weighted aggregate of the ratings is then used to make predictions for the active user.

The next step is the top-N recommendation step. This technique studies the user-item matrix to find relations between different users or items and use them to make the recommendations.

2.2.2 Model-based

Model-based CF algorithms use machine learning and data mining concepts that allows the system to learn and identify complicated patterns based on training data. They can then use test data and real world data to make smart predictions for the CF tasks based on the learned models. If the ratings are categorical, classification algorithms are used and if the ratings are numerical then regression models and SVD methods are used. In this this thesis model based a model based regression algorithm is used for numerical ratings. Some machine learning concepts used for model based CF algorithms are Bayesian models, clustering models and dependency networks. These have been studied to point out the limitations of memory-based algorithms [3, 12]. We will describe a few of these models briefly.

Usually a Bayesian CF algorithm uses a naïve Bayes approach to make predictions. A simple overview of the algorithm is as follows. If the features are independent given the class, then the probability of a particular class given all the features can be computed. The predicted class is the class with the highest probability [13].

Another type of machine learning algorithm is clustering which we will glimpse at. A cluster is a collection of objects that are similar to each other when they are in the same cluster and different from objects in other clusters. Minkowski distance and Pearson correlation are some metrics that are used for measuring similarity. Furthermore, clustering methods can be classified into three groups, density based methods, partitioning methods and hierarchical methods. Clustering is usually an intermediate step and the resulting clusters are used for further analysis. There are different ways to use clustering algorithms to make predictions. One such technique used in [14], first partitions the data into clusters and the use memory-based CF algorithm to make predictions with each cluster.

Of importance and interest to us is the regression based CF algorithms. In a memory based algorithm, two rating vectors may be distant in terms of Euclidian distance but may have good similarity using vector cosine or Pearson correlation measures. This is where model-based regression algorithms can perform better than memory based algorithms. Regression based methods are also good at making prediction for numerical ratings which are common in real like recommender systems.

The regression model is what we use in this thesis. This will be explained in detail in the following chapter. Here we will take a brief glance at the existing algorithms. The basic regression model is $Y = UV^T + \varepsilon$, where Y is the measurement matrix containing rows of users and columns of items. UV^T is the factorized matrix that needs to be approximated and ε is the noise associated with real world measurements. Usually Y is a very sparse matrix which makes SVD a poor method to use. Canny [15] proposed a sparse factor analysis. Here the missing elements if the matrix is replaced with the average value of the non-missing elements. Then Canny uses the regression model as initialization of Expectation Maximization. Another regression approach proposed in [16] searches for similarities between items, creates a collection of simple linear models and combines them to rate predictions for an active user. The parameters of the linear regression function were estimated using ordinary least squares. In another approach proposed in [17], slope one algorithms were used to make CF predictions.

2.2.3 Hybrid

Collaborative filtering combines with other recommendation systems like content-based or demographic-based systems to make predictions or recommendations are known as hybrid collaborative filtering. Content-based recommender systems have been mentioned in the

previous chapter. They contain information in addition to preference indicators. They include information about users and items to be rated.

2.3 CHALLENGES FACED

There are various challenges that are faced during a collaborative filtering task. Online shopping and searching companies need to provide recommendations accurately and efficiently to be able to thrive in this competitive market. The companies that address these challenges the best end up satisfying their customer needs the most. Some of the most important problems in this field are explained in [4]. We will review these challenges in the following sections.

The most important problem is sparseness of the data. Usually users end up rating products or movies they like very much or moderately like. Sometimes they don't take the effort to rate a movie they did not like. In other cases, most users have not seen all the movies or have not used all the products. This creates a very sparse user-item matrix where most of the items have not been rated and is thus a very sparse matrix. The sparsity of the matrix creates problems for the CF task. One of the most important one is called the cold start problem or the new user/item problem [5]. This problem occurs when a new user or item is entered in the system and not enough information is available to find similar ones. Unless the user rates at least one movie it is difficult to recommend something for the user based only on his ratings. Similarly, it is difficult to rate items that no user has rated. Some methods to tackle this problem are explained in [6]. In content based CF algorithms, where external data other than just ratings are used to make predictions, are better at dealing with the cold start new item new user problem.

The very large data sets also create computational costs that are too high. One of the challenges is to be able to scale down the matrix. When there are millions of customers and millions of users, then depending on the algorithm the computational cost become O(n), which is too large. The computational resources required for such an algorithm is impractical.

The data sparsity problem can be reduced by getting rid of items that have been not been rated or have been sparsely rated. Dimensionality reduction methods like Singular Value Decomposition (SVD) are used to remove users and items with low significance. Principle Components Analysis (PCA) is also used has also been used [7] to reduce dimensions. The problem with this method is that the information that is disregarded might have been useful and thus can make the recommendation sub-par. Techniques like SVD can also handle scalability problems well but they have costly factorization steps. A method described in [18] uses existing users to compute the SVD. For a new set of ratings added, it then uses a folding in projection technique [19] to create the new system without recomputing using SVD, making it very scalable. Other model-based CF algorithms like clustering, make recommendations for users from smaller and highly similar clusters. This way they avoid using the whole database and makes the algorithm more scalable. However, there is often a tradeoff between prediction accuracy and scalability.

Another problem faced by the collaborative filtering community is synonymy. This occurs when similar or same items have different names and entries. A lot of the recommendation systems are unable to recognize the similarity between the items. An example of synonymy would be ‘adventure film’ and ‘adventure movie’. In such a situation, memory-based CF systems would not be able identify a match between them to compute the similarity. These drawbacks can degrade the performance of a CF system. SVD and Latent Semantic Indexing techniques can deal with the synonymy problem fairly well [20].

Shilling attacks is known as the problem where users rate their own products positively multiple times and rate their competitor products negatively. It is desirable for the collaborative filtering community to take precautions to this problem.

Another problem faced is known as the gray sheep and black sheep. Gray sheep is referred to as the problem when certain users don't agree or disagree constantly with a certain group or people. This means that they cannot really benefit from collaborative filtering. Black sheep are those users who have very distinctive taste and recommending for this group of people is rather difficult. A hybrid content-based approach was suggested in [21]

3.0 MATRIX FACTORIZATION

As mentioned in the previous chapter, the model-based regression algorithm is used in this thesis. Most of the successful realizations of latent factor models are based on matrix factorization. Recently, they have become popular because they offer good scalability and predictive accuracy. They also offer room for modeling real world situations. Recommender systems often have many different types of input data. In this thesis the input data are movie ratings that are explicit. In the following sections we focus on factorizations where the matrix is represented as a product of two simpler matrices. This low norm concept is described below.

3.1 LOW NORM

Suppose there is a dataset that is organized as an observed matrix, $Y \in \mathbb{R}^{m \times n}$, then a product of two matrices U and V where $U \in \mathbb{R}^{m \times k}$ and $V \in \mathbb{R}^{n \times k}$ can approximate the observed matrix. If we consider the rows of Y as data vectors Y_i, then this data vector can be approximated by a linear combination $U_i V^T$ of the rows of V^T. The rows of V^T can be seen as factors and the entries of U as coefficients of the linear combinations. The way the approximation is done is by minimizing some measure of discrepancy between the observed matrix Y and model UV^T. A detailed study and explanation of low norm matrix factorization is found in [31].

3.1.1 Dimensionality Reduction

As mentioned in chapter 2, a popular way to conduct collaborative filtering is through dimensionality reduction, which also takes care of the scalability challenge. In the case of movie ratings, the underlying concept of the approach lies in the belief that there are only a few features (e.g. clarity, comedic influence, actors etc.) that influence the user's choice. Each movie has certain amounts of these features dominant in them. For movie ratings, the observed matrix will be $Y \in \mathbb{R}^{m \times n}$ where *m* is the number of users and *n* is the number of movies. Since a lot of users do not rate all the movies, this creates a very sparse matrix. The goal is to fit the target matrix *Y* with a rank *k* matrix $X = UV'$ where $U \in \mathbb{R}^{m \times k}$ and $V \in \mathbb{R}^{n \times k}$. The rank k are the features that need to be learned. To illustrate this, let's consider trying to rate MOVIE-1 for USER-1. Let's say there are *k=4* features – technological wizardry, comedic element, background score and thriller element. USER-1's preference for these features combined with the influence of these features on the MOVIE-1 will determine the rating.

Features(k)	*Tech wizardry*	*Comedic element*	*Background score*	*Thriller element*
USER 1	u_{11}	u_{12}	u_{13}	u_{14}

×

Features(k)	MOVIE 1
Tech wizardry	v'_{11}
Comedic element	v'_{21}
Background score	v'_{31}
Thriller element	v'_{41}

=

PREDICTED RATING
y_{11}

$$y_{11} = u_{11}v'_{11} + u_{12}v'_{21} + u_{13}v'_{31} + u_{14}v'_{41} \qquad (1)$$

The equation above shows that the rating y_{11} is a linear combination of *U* and *V'*. The underlying assumption that allows us to do this is that the prediction tasks – columns of *Y* are related. The same features are used to predict all of them, although in different ways. Each row of *U* thus becomes a feature vector and each row of *V* is the linear predictor.

3.2 MAXIMUM LIKELIHOOD ESTIMATION

To find the discrepancy between the observed matrix *Y* and model UV^T, one can model the error between them as noise. Minimizing this noise is how you would approximate the matrix. Depending on what data distribution this noise is coming from, the results can be very different. What distribution we choose to have the noise from is critical to this thesis. Here we use the concept of maximum likelihood estimation (MLE). Maximum likelihood estimation is the method of estimating the parameters of a statistical model. We can show that maximizing the likelihood is equivalent to minimizing a cost function. The format of the cost function is determined by the distribution of the noise in the data.

If the observed datum in the real world is an *m* dimensional column vector $\mathbf{y_i}$, then it is always accompanied with an additional value of noise. If there are *n* such observed datum, then

$$\boldsymbol{y_i} = \boldsymbol{\mu_i} + \varepsilon_i \qquad i = 1, \dots n \qquad (2)$$

where $\boldsymbol{\mu_i}$ is the unobservable unknown true value and ε_i is the additive noise that is to be minimized. $\boldsymbol{\mu_i}$ lies in a *k* dimensional linear subspace such that

$$\boldsymbol{\mu_i} = \mathrm{U}\boldsymbol{v_i} \qquad (3)$$

$\boldsymbol{v_i}$ is the projection of $\boldsymbol{y_i}$ on the subspace defined by the columns of *U*.

With the assumption that the measurements are independent and identically distributed, the log likelihood of the measurement is

$$l(\boldsymbol{\mu}; \boldsymbol{y}) = \log p\,(\boldsymbol{y_1}, \dots, \boldsymbol{y_n}\,|\boldsymbol{\mu_1}, \dots, \boldsymbol{\mu_n}) = \sum_{i=1}^{n} \log(\boldsymbol{y_i}|\boldsymbol{\mu_i}) \qquad (4)$$

The value of $\boldsymbol{\mu_i}$ that maximizes the likelihood of the measurement $l(\boldsymbol{\mu}; \boldsymbol{y})$ is what is desired. subject to the condition that these $\boldsymbol{\mu_i}$ s reside in a low dimensional subspace defined by U. We will minimize the error between the observed data and the predicted, $= \boldsymbol{y_i} - \mathrm{U}\boldsymbol{v_i}$, data using the above. Below are two detailed derivations and explanations of two different noise models.

3.2.1 Gaussian Noise Model

The derivation of the Least squares (L2 norm) minimization from theNormal error noise model is shown below

$$v_{MLE} = argmax_v log \prod_{i=1}^{n} P(Y_i|U_i; v) \qquad (5)$$

$$= argmax_v \sum_{i=1}^{n} logP(Y_i|U_i; v)$$

$$= argmax_v \sum_{i=1}^{n} log\left(\frac{1}{\sqrt{2\pi\sigma^2}} exp\left(-\frac{(Y_i - U_i^T v)^2}{2\sigma^2}\right)\right)$$

$$= argmax_v \sum_{i=1}^{n} log\frac{1}{\sqrt{2\pi\sigma^2}} + log\left(exp\left(-\frac{(Y_i - U_i^T v)^2}{2\sigma^2}\right)\right)$$

$$= argmax_v \sum_{i=1}^{n} \left(-\frac{(Y_i - U_i^T v)^2}{2\sigma^2}\right)$$

$$= argmin_v \sum_{i=1}^{n} \left(\frac{(Y_i - U_i^T v)^2}{2\sigma^2}\right)$$

$$= \frac{1}{2\sigma^2} argmin_v \sum_{i=1}^{n} (Y_i - U_i^T v)^2$$

$$v_{MLE} = argmin_v \sum_{i=1}^{n} (Y_i - U_i^T v)^2 \qquad (6)$$

$$= (Uv - Y)^T (Uv - Y) \text{ Closed form solution}$$

Performing least squares linear regression makes us make certain unrealistic assumptions about the error vectors. In particular, cases where the error distribution is heavier tailed than the Normal distribution, which means it has more probability in the tails than the Normal, the L2 norm loss is very sensitive to outliers and does not perform well. This requires a more robust regression method. When the noise is large, or when there is a large existence of outliers, least squares weights each observation equally in getting parameter estimates. Robust methods are able to weight the observations unequally, thereby giving lower weights to outliers. That is, the observations that produce large residuals are down weighted. A robust method is to use the Laplacian distribution instead of the Gaussian distribution for error measurements. We will discuss this in the following section. The L2 norm also has a closed form solution and is able to find a global minimum.

3.2.2 Laplacian Noise Model

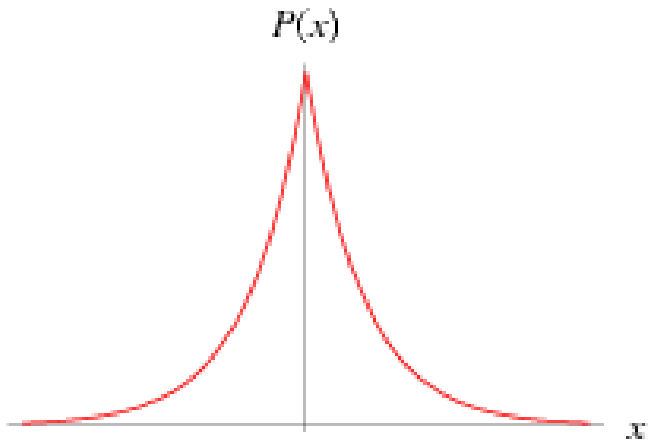

Figure 1. The Laplace distribution

The Laplacian distribution is shown in Figure 1. The Gaussian model achieves the least square error and the Laplace model tried to achieve the least absolute value error. It is obvious that the effect of outliers in the estimation of the Gaussian model where the error is squared is much larger than the Laplace method, where the error is not squared. For an outlier with a large deviation, the Laplace model explains it away by accommodating a large error. This is easier with the Laplace model as the Laplace distribution has a heavier tail than the Gaussian distribution. Consequently, the Laplace model is more robust by de-emphasizing the impact of data with large deviation. The derivation of the L1 minimization from the Laplace error noise model is shown below.

$$v_{MLE} = argmax_v log \prod_{i=1}^{n} P(Y_i|U_i; v) \qquad (7)$$

$$= argmax_v \sum_{i=1}^{n} logP(Y_i|U_i; v)$$

$$= argmax_v \sum_{i=1}^{n} log\left(\frac{1}{2b} exp\left(-\frac{|Y_i - U_i^T v|}{b}\right)\right)$$

$$= argmax_v \sum_{i=1}^{n} log\frac{1}{2b} + log\left(\frac{1}{2b} exp\left(-\frac{|Y_i - U_i^T v|}{b}\right)\right)$$

$$= argmax_v \sum_{i=1}^{n} -\frac{|Y_i - U_i^T v|}{b}$$

$$= argmax_v \sum_{i=1}^{n} -\frac{|Y_i - U_i^T v|}{b}$$

$$= argmin_v \sum_{i=1}^{n} \frac{|Y_i - U_i^T v|}{b}$$

$$= \frac{1}{b} argmin_v \sum_{i=1}^{n} |Y_i - U_i^T v|$$

$$v_{MLE} = argmin_v \sum_{i=1}^{n} |Y_i - U_i^T v| \qquad (8)$$

In summary, the maximum likelihood (ML) solution to the matrix factorization (subspace computation) depends on the noise distribution assumed. When the noise follows independent and identical Gaussian distribution, the ML solution is obtained by minimizing a L2 norm cost function. When the noise follows independent and identical Laplacian distribution, the ML solution is achieved by minimizing a L1 norm cost function. The case of L1 norm can deal with outliers.

3.3 LINEAR PROGRAMMING

A linear program (LP) is an optimization problem, which has a linear objective functions along with constraints that consist of linear equalities and inequalities.

$$min_{x,t} = \mathbf{1}^{\intercal}\boldsymbol{x}$$

$$s.t. -\boldsymbol{t} \leq U^{(t-1)}\boldsymbol{x} - \boldsymbol{y_i} \leq \boldsymbol{t} \qquad (9)$$

This is a solvable linear program. In this thesis we use the L1 Magic package [30] in our algorithm to solve it.

3.4 L1 APPLICATIONS

L1 norm has gained importance and popularity in the last decade. Scientists have found important and practical use for them. Real world data is almost always corrupted with outliers, noise and missing data. A robust method to approximate this data had become necessary. Many papers have been published recently that use the L1 norm for interesting applications.

One such application is image retrieval. Extraction of discriminative features and a feasible similarity metric for recovering images that are similar in content with the search image are important steps in the image retrieval system. This paper proposes a sparsity promoting technique using L1 norm minimization that finds the sparsest solution of an under-determined system of linear equations. They use the L1 norm as a similarity metric. Their results show that the L1 minimization provide promising alternatives to existing techniques.

In another application, the L1 norm penalty function was used to compute epicardial potentials from multi-electrode body surface ECG measurements. Previously the Tikhonov regularization was used, which employed the L2 norm penalty functions or their derivatives. However, the L2 caused a considerable smoothing of the solution. Using the L1 norm produced better results and even detected two distinct areas of early activation that indicated the presence of two left-sided pathways which were not distinguished by L2 regularization.

Other applications include face recognition, sparse signal recovery etc.

4.0 ALGORITHM

The algorithm suggested by Qifa Ke and Takeo Kanade [1] introduces and implements a scheme that alternately optimizes a cost function using L1 norm minimization. Ke and Kanade [1] apply this alternative optimization to structure from motion, which deals with a sparse matrix that contains outliers and missing data and achieves good results. In this thesis we have applied their suggested algorithm with slight modifications to collaborative filtering. Our data set is also a large sparse matrix with many missing data.

We want to factorize a sparse matrix Y into its subspaces U and V via robust efficient L1 minimization. The cost function is shown below.

$$minimize \ \|Y - UV^{\intercal}\|_1 \qquad (10)$$

where $Y \in \mathbb{R}^{m \times n}$, $U \in \mathbb{R}^{m \times k}$ and $V \in \mathbb{R}^{n \times k}$.

The matrices *U* and *V* will be learned by the algorithm. Since *U* and *V* are both unknown, the problem as stated now is non-convex. However, if one of the unknowns *U* or *V* was known, then the cost function w.r.t to the other unknown becomes a convex function. The global minimum of this cost function can be obtained. The cost function can thus be minimized by alternatively minimizing over *U* and *V*.

4.1 ALTERNATE CONVEX OPTIMIZATION

The alternate optimization problem can be shown as

$$V^{(t)} = arg\ min_V \left\| Y - U^{(t-1)}V^{\intercal} \right\|_1 \qquad (11)$$

$$U^{(t)} = arg\ min_U \left\| Y^{\intercal} - V^{(t-1)}U^{\intercal} \right\|_1 \qquad (12)$$

where *t* is the iteration number. The two equations above can be formulated into a convex linear program.

$$\left\| Y - U^{(t-1)}V^{\intercal} \right\|_1 = \sum_{j=1}^{n} \left\| \boldsymbol{y_j} - U^{(t-1)}\boldsymbol{v_j} \right\|_1 \qquad (13)$$

$$\boldsymbol{v_j} = arg\ min_x \left\| \boldsymbol{y_j} - U^{(t-1)}\boldsymbol{x} \right\|_1 \qquad (14)$$

where $U^{(t-1)}$ is the subspace matrix from the previous iteration, $\boldsymbol{y_j}$ *j*th column of matrix *Y*, $\boldsymbol{v_j}$ is the *j*th column of matrix $V^{\intercal}$.

This problem then becomes the linear programming problem described in section 4.

$$\left\| Y^{\intercal} - V^{(t-1)}U^{\intercal} \right\|_1 = \sum_{i=1}^{m} \left\| \boldsymbol{y_i} - V^{(t-1)}\boldsymbol{u_i} \right\|_1 \qquad (15)$$

$$\boldsymbol{u_i} = argmin_x \left\| \boldsymbol{y_i} - V^{(t-1)}\boldsymbol{x} \right\|_1 \qquad (16)$$

where $V^{(t-1)}$ is the subspace matrix from the previous iteration, $\boldsymbol{y_i}$ *i*th row of matrix *Y*, $\boldsymbol{u_i}$ is the *i*th row of matrix *U*. For the experiment we use the L1 Magic optimization package [30] was used for the minimization.

4.2 PCA TO FIND CORRECT LOW RANK

Principal component analysis (PCA) is a powerful, quantitative tool derived from linear algebra that is used to reduce a complex set of data to a lower dimension. It works under the notion that there is some redundancy of variables in the set of data. Redundancy here means that the variables are correlated to each other, possibly because they are representing the same construct. This redundancy makes it possible to reduce the observed variables into smaller number of principal components that will account for most of the variance in the observed variables. Intuitively speaking, PCA's role can be thought of as revealing a simplified internal structure of a complicated set of data which best explains the variance in the data. The method involves an eigenvalue decomposition of the covariance matrix of the high dimensional or multivariate data set, which has been mean centered for each variable. This basically means a covariance matrix is created from the set of data from which the eigenvectors are found. These eigenvectors with their corresponding eigenvalues help to form the principal component of the data set. The principal component can be defined as a linear combination of optimally-weighted observed variables. The number of components extracted is equal to the number of observed variables being analyzed.

However, in most analyses, only the first few components account for significant amounts of variance, so only these first few components are retained. The elimination of the other components result in the reduction in dimension of the data set that usually has minimal error when compared to the original set of data. Hence a low dimensional data set can be obtained for analysis using PCA, without significant loss of information. This is the method we use to find out what rank to choose for the matrix factors U and V.

4.3 INITIALIZATION

The algorithm requires U to be initialized. Ke and Kanade [1] found out that a random initialization was just as good as filling the missing data with column mean and performing SVD on it to find *U*. The algorithm was not sensitive to the initialization.

4.4 MISSING DATA

4.4.1 Method 1

The matrix we are concerned with is sparse and has a lot of missing data. Ke and Kanade [1] proposed a simple way to handle the missing data. The constraint for each missing datum when solving the equations in the section above is dropped. The missing data can be recovered once subspaces are computed.

$$\sum_{i=1}^{m}\sum_{j=1}^{n}\left|y_{ij}-\boldsymbol{u_i^{\intercal}v_j}\right| \qquad (17)$$

Dropping such an item in convex programming is equivalent to dropping a constraint in equation in the linear program mentioned in section 4. This is different from other traditional methods where the missing data is explicitly discovered. This method however does create some problems since our data set is very sparse with entire columns of *Y* with missing data. These columns correspond to movies that no user rated. For these particular columns, all the constraints are removed. Each user was required to rate at least 20 movies so the rows of *Y* will always at least that many non-missing values.

4.4.2 Method 2

In addition to the above, we also used another method to deal with the missing data and compared results. Instead of dropping the constraint, we filled in the missing data in the columns with their corresponding column mean.

For both methods, the movie items that have been rated less than the rank chosen for that algorithm will be removed from the dataset. It does not make sense otherwise.

4.5 CONVERGENCE

The algorithm decreases the cost function at each alternating minimization step. This cost function is an absolute value and is lower bounded, $E(U,V) \geq 0$ so it will converge. The algorithm converges when the difference between adjacent iterations is small enough. If $\theta(a,b)$ is the angle between two vectors **a** and **b** then the algorithm converges when

$$\theta(u_i^t, u_i^{t+1}) < \alpha \qquad (18)$$

u_i is the *i*th column of U or V and α is a small positive number. α was set to 4 degrees for the collaborative filtering experiment.

4.6 ALGORITHM SUMMARY

Below is the summary of the algorithm

Initialize U randomly

For t=1 till convergence

If method 1

Remove missing data

Else if method 2

Fill missing data with column mean

Perform alternate convex minimization

$$V^{(t)} = arg\ min_V \left\| Y - U^{(t-1)} V^{\intercal} \right\|_1$$

$$U^{(t)} = arg\ min_U \left\| Y^{\intercal} - V^{(t-1)} U^{\intercal} \right\|_1$$

$Converge\ if \quad \theta(u_i^t, u_i^{t+1}) < \alpha$

4.7 EXPERIMENTATION

In this section we provide details of the data set used. We also explain the experimental protocols and error metrics in evaluating the performance of the algorithm.

4.7.1 Data Set

It is very important for empirical research on rating prediction algorithms to have the availability of large data sets. For this thesis the MovieLens (ML) data set was used. The data in MovieLens was collected through the on-going MovieLens project, and is distributed by GroupLens Research at the University of Minnesota. The data consists of 6040 users, 3952 movies, and 1000209 ratings collected from users who joined the MovieLens recommendation service in 2000. Ratings are on a scale from 1 to 5 and the base data set is 95:8% sparse. The *Y* matrix mentioned in section 5.1.1 will contain all these ratings. There will be 6040 rows representing the users and 3952 columns representing movies.

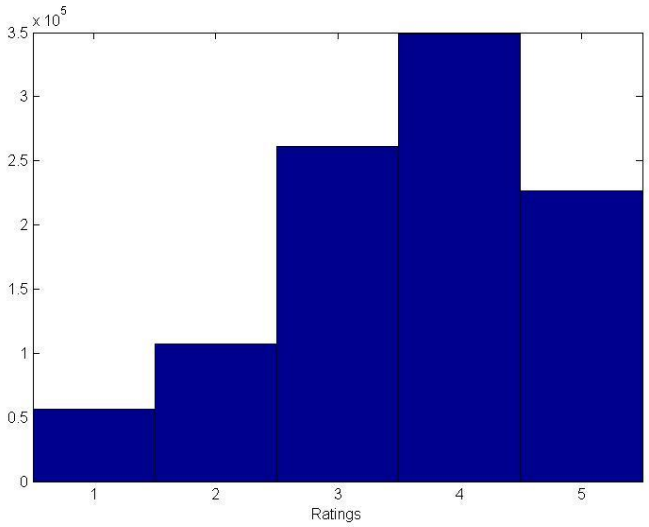

Figure 2. Data distribution of the MovieLens ratings

.

4.7.2 Weak generalization

One of the first prediction experiments was performed by Breese, Heckerman, and Kadie [3]. The experimental protocol they used has been popularly followed in literature. This is the same protocol we will use. This section will provide a brief overview of the one of the protocols. The ratings in the experiment will be split into an observed set, and a held out set. The observed set is used for training the algorithm and the held out set is for testing the performance of the method. For a validation set, the training set can further be split. We will not be using a validation set for our experiment. This method is called weak generalization because it can only predict ratings for items of the same users who have been used in the training set. In the end, this just becomes a matrix completion problem and can be implemented by directly using the algorithm in 5.1.1.

4.7.3 Strong generalization

Another type of generalization that is more useful is called strong generalization. This generalization is used to predict ratings for novel users. The set of users is first divided into two sets consisting of training users and test users. The learning part of the algorithm is performed with all the available ratings from the training users. A validation set can be created from the training set but we will not use it in our experiment. To test this method, the ratings of each user are split into an observed set, which the algorithm can access, and a held out set which the algorithm will predict.

For the alternate minimization algorithm, the training set of users is first used for learning. This means U and $V^{\intercal}$ will be learned from the training set. For the testing set, U will be randomly initialized again as mention in section 5.1.1 but learned $V^{\intercal}$ will be used from the training. The algorithm will optimize U over a fixed $V^{\intercal}$ till the difference between adjacent iterations of columns of U is less than a small number as shown in section 5.1.3.

As mentioned above, in both weak and strong generalizations each user's ratings are partitioned in a set of observed items and a set of held out items. There are several ways to do this. We used the all-but-1 method in our algorithm were all of the user's ratings are observed except for one which we will test. For the experiments we will use 5000 users for weak generalization and 1040 for strong generalization.

4.7.4 Error metrics

The error measures we use are those that are described in Marlin [2]. It is the mean absolute error. The equation is shown below.

$$MAE = \frac{1}{M}\sum_{i=1}^{M}\left|r_{heldout_i} - r_{predicted_i}\right| \qquad (19)$$

Where M is the number of users, $r_{heldout_i}$ is the actual rating and $r_{predicted_i}$ is the predicted rating. Marlin [2] uses a normalization value of $E[MAE]$.

$$NMAE = \frac{MAE}{E[MAE]} \qquad (20)$$

where E[MAE] is the expected value of the MAE assuming uniformly distributed observed and predicted rating values. An NMAE error of less than one means a method is doing better than randomly predicting ratings, while an NMAE value of greater than one means the method is performing worse than random. Marlin's value of E[MAE]=1.6 is what we used for our experiment.

4.7.5 Example Problem

Here we will show an example problem. The following is an 8×6 matrix rank 2 matrix from from [1] The highlighted data points are the ones that will be omitted to emulate missing data:

Y

9.4700	-7.3000	-2.4300	8.1300	7.8700	7.5600
8.4200	-0.1300	-2.0300	6.9900	5.8300	1.5000
-12.4900	-5.7100	2.8800	-10.1500	-7.5500	2.6200
1.0300	-4.5600	-0.3400	1.0200	1.5500	3.9200
1.6900	11.2600	-0.1700	0.9900	-0.9000	-8.9700
3.8300	9.4800	-0.7200	2.8300	0.8900	-7.1600
1.8400	5.8300	-0.3200	1.3100	0.1900	-4.4800
8.0800	8.9700	-1.7500	6.3700	3.9100	-6.0300

$Y_{missing}$ shows the same matrix as Y except the highlighted points have been replaced with zeros indicating missing values.

$Y_{missing}$

9.4700	-7.3000	0	8.1300	7.8700	7.5600
0	-0.1300	-2.0300	0	5.8300	1.5000
-12.4900	-5.7100	2.8800	0	-7.5500	2.6200
1.0300	0	0	1.0200	0	0
1.6900	11.2600	-0.1700	0.9900	-0.9000	-8.9700
3.8300	9.4800	-0.7200	2.8300	0.8900	0
1.8400	0	0	1.3100	0.1900	-4.4800
8.0800	8.9700	-1.7500	6.3700	0	-6.0300

$Y_{missing\_weak\_generalization\_set}$ is a 6×6 matrix randomly selected subset from $Y_{missing}$ to generate a weak generalization matrix. This will be used as training data and the missing values are used as test data.

$Y_{missing\_weak\_generalization\_set}$

9.4700	-7.3000	0	8.1300	7.8700	7.5600
-12.4900	-5.7100	2.8800	0	-7.5500	2.6200
1.0300	0	0	1.0200	0	0
1.6900	11.2600	-0.1700	0.9900	-0.9000	-8.9700
1.8400	0	0	1.3100	0.1900	-4.4800
8.0800	8.9700	-1.7500	6.3700	0	-6.0300

The remainder 2×6 matrix taken from $Y_{missing}$ was chosen as the $Y_{missing\_strong\_generalization\_set}$. As explained before this is only used as test data.

$Y_{missing\_strong\_generalization\_set}$

0	-0.1300	-2.0300	0	5.8300	1.5000
3.8300	9.4800	-0.7200	2.8300	0.8900	0

The following two matrices were generated from the algorithm. The missing data was taken care of by removing them (Method 1). If we compare the matrices below to Y, then we can see that the predicted values are very close to the original values.

$Y_{weak\_generalization\_result}$

9.4784	-7.2999	-2.4310	8.1299	7.8698	7.5621
-12.4898	-5.7097	2.8797	-10.1473	-7.5513	2.6233
1.0300	-4.4941	-0.3423	1.0200	1.5358	3.8608
1.6947	11.2601	-0.1694	0.9900	-0.9036	-8.9665
1.8390	5.8260	-0.3188	1.3101	0.1950	-4.4801
8.0800	8.9700	-1.7516	6.3698	3.9148	-6.0300

$Y_{strong\_generalization\_result}$

8.4311	-0.1294	-2.0280	6.9968	5.8300	1.5005
3.8301	9.4800	-0.7199	2.8266	0.8943	-7.1516

4.8 RESULTS AND DISCUSSION

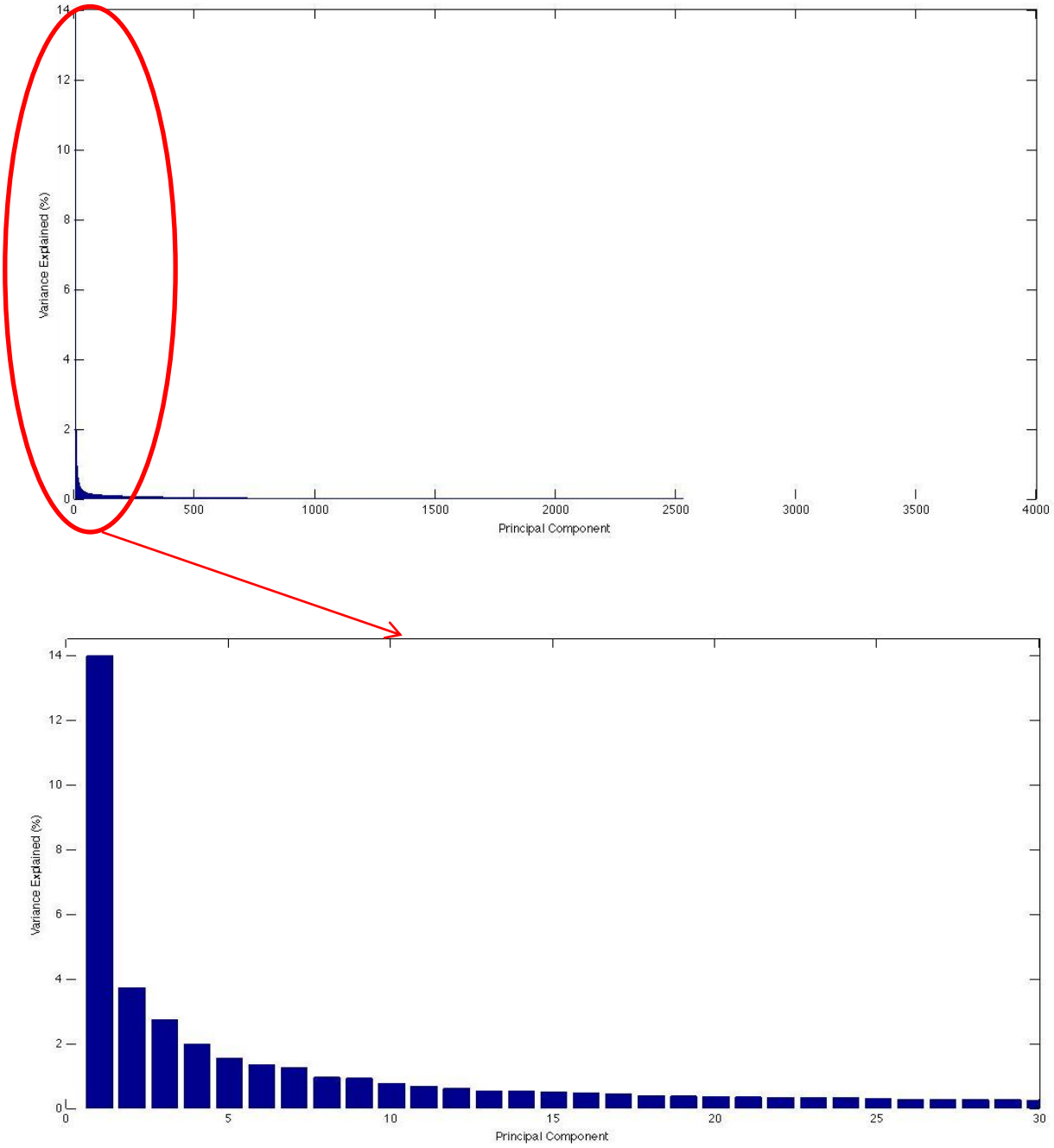

Figure 3. Variance of data explained by the Principal Components

The top figure shows all the principal components. The bottom figure is zoomed in to the first 30 principal components.

Principal Components Analysis was performed on the measurement matrix. The top portion of Figure 3 shows the shows all of the principal components and how much of the data is explained by each of them. It is clear that only the first 15-20 principal components explain the variance of the data best. This means that there are about 15-20 features that are common to all the users and movies. This result prompted me to choose a rank of 10, 12, 15 and 20 for matrices U and V while performing the experiment.

Table 1. Weak NMAE and Strong NMAE results for the algorithm where missing data was removed.

Rank	**10**	**12**	**15**	**20**
Weak NMAE	0.5314	0.5617	0.6513	0.9647
Strong NMAE	0.4952	0.5695	0.6006	0.7336

Table 2. Weak NMAE and Strong NMAE results for the algorithm where missing data was replaced with the column mean

RANK	**10**	**12**	**15**	**20**
Weak NMAE	0.4887	0.4931	0.4825	0.4958
Strong NMAE	0.4827	0.5057	0.4834	0.5022

Table 1 and 2 show the results, weak NMAE and strong NMAE, from the collaborative filtering algorithms. The ranks of the matrices chosen were chosen based on the results from PCA.

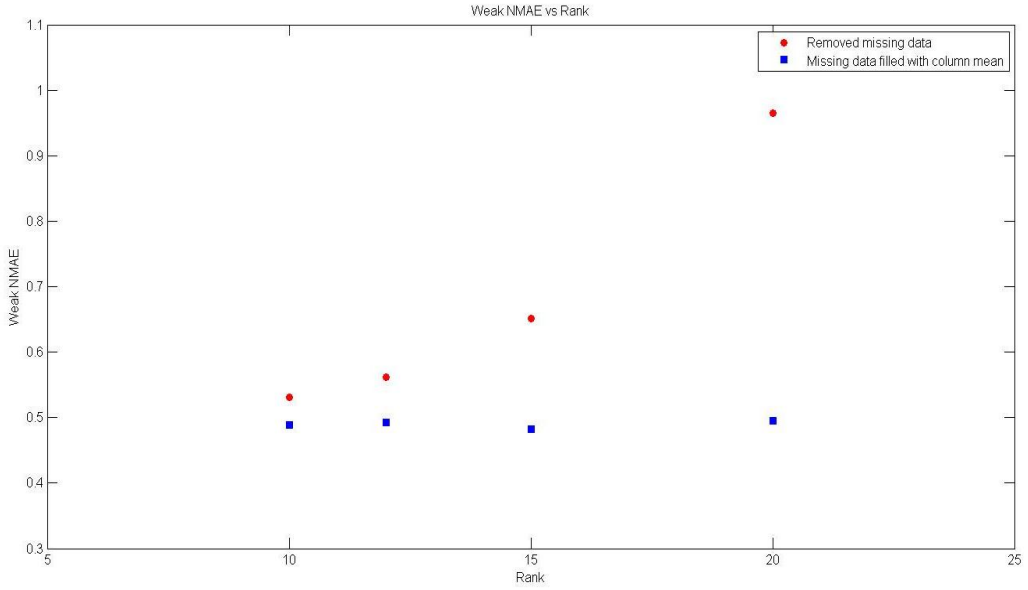

Figure 4. A plot showing Weak NMAE vs Rank.

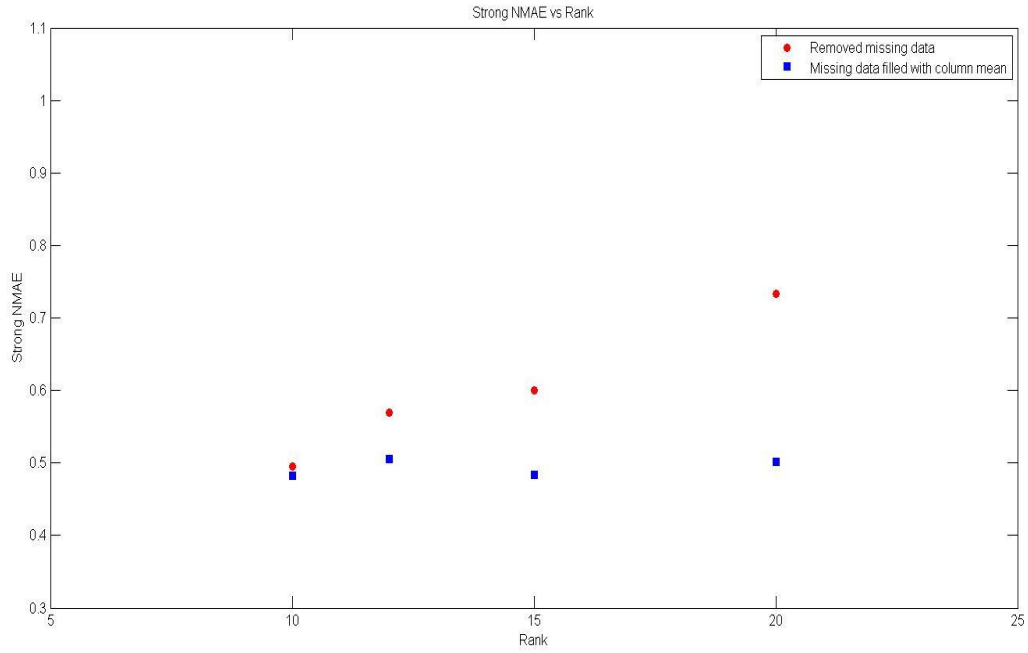

Figure 5. A plot showing Strong NMAE vs Rank.

For both figures, the red dots are results from the algorithm where the missing data was removed. The blue dots are from the algorithm where the missing data was replaced with the column mean.

Figure 4 shows the weak NMAE from both the algorithms, one where the missing data was removed, and the other where the missing data was filled with the column mean. The results are very interesting. It shows that the second method performed much better than the first one. As the rank was increased from 10 to 20, the results for the first algorithm deteriorated. We think this happened because removing the missing data reduced the information used to predict the ratings, thus generating inaccurate predictions. In addition, as the rank was increased the number of variables to be learned increased as well, which caused even more inaccuracy and increased the error. We think this method was more successful in Ke and Kanade [1] because the data set they used was not as sparse as ours, rather it had more outliers. On the other hand, the second method that we tried showed consistently low error even as the rank increased, which confirmed the lowest rank tested by PCA was sufficient. The results from the second method are comparable to the results out in literature [32, 33], which verifies this is a successful method to perform collaborative filtering. Figure 5 shows the strong NMAE for both algorithms and the results are similar to the weak NMAE.

Our algorithm addressed the challenges mentioned in Chapter 2. The sparseness of the dataset was tackled using two method described above. The matrix factorization and dimensionality reduction helped make the problem more scalable and addressed the synonymy challenge. We expect synonyms to fall in the same lower dimension. In addition, gray sheep/black sheep can be considered outliers and the L1 noise model made the problem robust to outliers.

5.0 IMORTANCE OF COLLABORATIVE FILTERING IN THE HEALTHCARE INDUSTRY

The one area of science than the human race's well –being relies most heavily on is healthcare. Every human is prone to get diseases, either hereditary or otherwise. They are also subject to the natural aging process which comes with numerous physical ailments and disabilities.

As an employee of a global healthcare company myself, I was motivated to work on a thesis topic that could be practically applied to the healthcare industry. The huge costs of healthcare, difficulty in keeping track of patients with chronic disease are only some of the factors that concern the healthcare community today. This kind of crisis has led to the need of preventive care, which entails recognizing disease risk and taking early action to prevent them. However, to manually take on such a task is highly impractical in terms of both cost efficiency and time efficiency, which is why the aim should be to automate such a system with a reliable recommendation algorithm.

The healthcare in the US is currently a recurring topic amongst policy makers. This is because these costs are only expected to rise in the future. The system is overburdened with the health concerns of both the aging generation and the younger generation. A study shows that since 1992, the average age of patients visiting hospitals increased to 45 years, and the visit rate

for persons 45 years of age and over increased by 17% from 407.3 to 478.2 visits per 100 persons [22]. The healthcare industry thus needs to focus its shift from reactive treatment to proactive treatment.

It is very challenging for a single healthcare professional to fully understand the complex issues surrounding the different disease generating factors and entire medical history of a patient. In a current traditional medical environment, physicians use a physical examination, aided with family background to assess the situation of the patient. To further assist the investigation they order laboratory tests and depend on their results. This method is very focused on a limited number of diseases and is dependent on the physicians experience and competence.

As a result, the current medical care jumps in when the symptoms of the disease emerges, making is reactive. What we want is a proactive system where the treatment would eliminate the disease as the earliest signs. The genome revolution has brought about important progress in preventive healthcare. The current technologies have provided a comprehensive list of disease-gene associations giving us thorough information on the possibility of developing special diseases [23]. The goal of this kind of research is that once all the disease related mutations are catalogued, we will be able to predict each individual's predisposition to future diseases. However, these genome based innovations are still limited [24]. This leads to opening alternative ways to pursue preventive healthcare practices.

Computer-aided medical prediction systems have been the interest of related research recently. One such popular system is the Apache III [25], a scoring system that predicts inpatient mortality. Apache uses a combination of acute physiological measurements, age, and chronic health status to make these predictions. There are also a number of systems developed for

predicting risk of individual diseases, such as specific heart conditions [26], hepatitis [27], Alzheimer's disease [28], etc.

One paper [29] proposes CARE, a Collaborative Assessment and Recommendation Engine, which relies only on a patient's medical history using ICD- 9-CM ICD-9-CM (International Classification of Diseases, 9th revision, Clinical Modification) codes in order to predict diseases risks in the future. This is a more general predictive system and does not focus on specific diseases like the above. CARE uses collaborative filtering to predict each patient's greatest disease risks based on their own medical history and that of similar patients. The also propose an Iterative version, ICARE, which incorporates ensemble concepts for improved performance. These systems require no specialized information and provide predictions for medical conditions of all kinds in a single run.

I currently work in a medical device company that specializes in Continuous Positive Airway Pressure (CPAP) devices, which is a device that produces a mode of respiratory ventilation and is primarily used to treat patients with sleep apnea. Sleep apnea is a sleep disorder that is characterized by unnatural pauses in breathing or occurrences of low breathing during sleep. A pause in breathing which can last from a few seconds to a minute is called an apnea. An abnormally low breathing is called a hypopnea. Such breathing pauses can causes disruption in sleep. Cases have been reported where patients with obstructive sleep apnea have fallen asleep during driving that can cause fatal crashes. Positive pressure through the airway path can prevent these breathing pauses to take place and provide a patient with an uninterrupted night of sleep. Usually the patients using the devices that are specific to treating sleep apnea do not need to be hospitalized.

Different variants of more sophisticated PAP devices are used in more serious patients that are hospitalized and critically ill with respiratory failure. Occasionally neuromuscular diseases are also treated using a variety of CPAP devices. In a hospital, the PAP ventilation is most commonly used for congestive heart failure and acute exacerbation of obstructive airway disease. As one can imagine there is a huge amount of device therapy data that can be collected from both hospitalized and non-hospitalized data. Device therapy data consists of Currently there is a serious lack of retrospective analysis of the device therapy data that could provide valuable information for future CPAP and respiratory devices.

I am currently involved in a data mining project that could benefit tremendously from the concepts of collaborative filtering. The existence of so many devices in such disparate locations and conditions create a very diverse and large dataset. In addition, there are many factors that cause the patient condition to worsen or get better. The idea of the project is to collect historical therapy data from patients using PAP devices and use that data to first of all find the most important and relevant factors that influence their condition. Eventually we would want to be able to take these factors and perform some kind of collaborative prediction to be able to predict prescriptions for future patients and better characterize patient needs. This can prove to be very effective and reduce physician time and involved which also reduces cost. It can also help the engineers get better insight of how the devices are behaving to try and improve the algorithms.

BIBLIOGRAPHY

[1] Q. Ke and T. Kanade. Robust `1-norm factorization in the presence of outliers and missing data. In Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, 2005.

[2] Marlin, B. (2004. Collaborative filtering: A machine learning perspective. Master's thesis, University of Toronto, Computer Science Department.

[3] John S. Breese, David Heckerman, and Carl Kadie. Empirical Analysis of Predictive for Collaborative Filtering. In Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence, pages 43-52, July 1998.

[4] X. Su and T. M. Khoshgoftaar, A Survey of Collaborative Filtering Techniques, Advances in Artificial Intelligence (2009).

[5] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: a survey of the state-of-theart and possible extensions," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 6, pp. 734–749, 2005.

[6] A.I. Schein, A. Popescul, L.H. Ungar, and D.M. Pennock, "Methods and Metrics for Cold-Start Recommendations," Proc. 25th Ann. Int'l ACM SIGIR Conf., 2002.

[7] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, "Eigentaste: a constant time collaborative filtering algorithm," *Information Retrieval*, vol. 4, no. 2, pp. 133–151, 2001.

[8] Mark Claypool, Phong Le, Makoto Wased, and David Brown. Implicit interest indicators. In Intelligent User Interfaces, pages 33-40, 2001.

[9] Dmitry Y. Pavlov and David M. Pennock. A maximum entropy approach to collaborative filtering in dynamic, sparse, high dimensional domains. In Proceedings of the Sixteenth Annual Conference on Neural Information Processing Systems (NIPS 2002), 2002.

[10] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl, "Itembased collaborative filtering recommendation algorithms," in *Proceedings of the 10th International Conference on World Wide Web (WWW '01)*, pp. 285–295, May 2001.

[11] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: an open architecture for collaborative filtering of netnews," in *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, pp. 175–186, New York, NY, USA, 1994.

[12] C. Basu, H. Hirsh, and W. Cohen, "Recommendation as classification: using social and content-based information in recommendation," in *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI '98)*, pp. 714–720, Madison,Wis, USA, July 1998.

[13] K. Miyahara and M. J. Pazzani, "Improvement of collaborative filtering with the simple Bayesian classifier," *Information Processing Society of Japan*, vol. 43, no. 11, 2002.

[14] M. O'Connor and J. Herlocker, "Clustering items for collaborative filtering," in *Proceedings of the ACM SIGIR Workshop on Recommender Systems (SIGIR '99)*, 1999.

[15] J. Canny, "Collaborative filtering with privacy via factor analysis," in *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 238–245, Tampere, Finland, August 2002.

[16] S. Vucetic and Z. Obradovic, "Collaborative filtering using a regression-based approach," *Knowledge and Information Systems*, vol. 7, no. 1, pp. 1–22, 2005.

[17] D. Lemire and A. Maclachlan, "Slope one predictors for online rating-based collaborative filtering," in *Proceedings of the SIAM Data Mining Conference (SDM '05)*, 2005.

[18] B. M. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Incremental SVD-based algorithms for highly scaleable recommender systems," in *Proceedings of the 5th International Conference on Computer and Information Technology (ICCIT '02)*, 2002.

[19] M. W. Berry, S. T. Dumais, and G. W. O'Brien, "Using linear algebra for intelligent information retrieval," *SIAM Review*, vol. 37, no. 4, pp. 573–595, 1995.

[20] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society for Information Science*, vol. 41, no. 6, pp. 391–407, 1990.

[21] M. Claypool, A. Gokhale, T. Miranda, et al., "Combining content-based and collaborative filters in an online newspaper," in *Proceedings of the SIGIR Workshop on Recommender Systems: Algorithms and Evaluation*, Berkeley, Calif, USA, 1999.

[22] D. K. Cherry, C. W. Burt, and D. Woodwell. A national ambulatory medical care survey: 2001 summary. Advance Data, 337:1-16, 2001.

[23] W. T. C. Consortium. A national ambulatory medical care survey: 2001 summary. Nature, 447:661-678, 2007.

[24] J. Loscalzo. Association studies in an era of too much information - clinical analysis of new biomarker and genetic data. Circulation, 116(17):1866-1870, 2007.

[25] D. T. Wong and W. A. Knaus. Predicting outcome in critical care: the current status of the apache prognostic scoring system. Canadian Journal of Anesthesia, 38:374-383, 1991.

[26] O. Cord_on, F. Herrera, J. de la Monta~na, A. S_anchez, and P. Villar. A prediction system for cardiovascularity diseases using genetic fuzzy rule-based systems. In Proceedings of the 8th Ibero-American Conference on AI, pages 381-391. Springer Berlin, 2002.

[27] F. Piscaglia, A. Cucchetti, A. Orlandini, E. Sagrini, A. Gianstefani, C. Crespi, G. Pelosi, M. Valli, L. Sacchelli, C. Ferrari, and L. Bolondi. Prediction of significant fibrosis in chronic hepatitis c patients by artificial neural network analysis of clinical factors. volume 39, March 2007.

[28] Y. Liu, L. Teverovskiy, O. Lopez, H. Aizenstein, C. Meltzer, and J. Becker. Discovery of biomarkers for alzheimer's disease prediction from structural mr images. In 2007 IEEE International Symposium on Biomedical Imaging, April 2007.

[29] Davis D, Chawla NV, Blumm N, Christakis N, Barabasi A-L (2008b) Predicting individual disease risk based on medical history. In: Proceedings of the ACM conference on information and knowledge management

[30] l1-magic. [Online]. Available: http://www.l1-magic.org

[31] Srebro, N.: Learning with matrix factorizations. Ph.D. thesis, Massachusetts Institute of Technology (2004)

[32] J. D. M. Rennie and N. Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proc. of ICML*, 2005.

[33] D. DeCoste, "Collaborative prediction using ensembles of maximum margin matrix factorizations," in *Proceedings of the 23rd International Conference on Machine Learning (ICML '06)*, pp. 249–256, Pittsburgh, Pa, USA, June 2006.