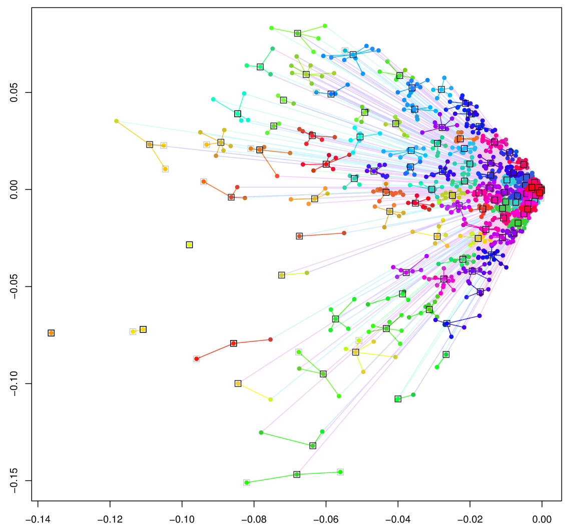
University of Zurich
Department of Informatics

Dynamic and Distributed Information Systems

Bachelor Thesis
January 9, 2012

Clustering High-dimensional Sparse Data



Manuel Gugger
of Gebenstorf AG, Switzerland

Student-ID: 07-927-759
s0792775@access.uzh.ch

Advisor: **Floarea Serban**

Prof. Abraham Bernstein, PhD
Department of Informatics
University of Zurich
http://www.ifi.uzh.ch/ddis

Acknowledgements

I would like to use this chance to thank all the people who helped me during this thesis for their support and their encouragement. First of all I would like to thank Mr. Bernstein for giving me the opportunity to develop this thesis at the Distributed Information Systems (DDIS) group although I had not much prior knowledge on the topic itself.

I am sincerely thankful to my supervisor Floarea Serban, Ph.D. student at the DDIS group, for the helpful hints and for giving much space on my investigations on the topic.

Last but not least I would like to thank my family and friends, especially Benjamin Häni, for their supporting words and for their understanding of sudden mood changes related to my ongoing work.

Manuel Gugger
Zurich, Switzerland

Abstract

This work is a practical approach on evaluating clustering algorithms on different datasets to examine their behaviour on high-dimensional and sparse datasets. High-Dimensionality and sparsity poses high demands on the algorithms due to missing values and computational requirements. It has already been proven that algorithms perform significantly worse under high-dimensional and sparse data. Here approaches to circumvent these difficulties are analysed. Distance matrices and recommender systems have been examined to either reduce the complexity or to impute missing data. A special focus is then put on the similarity between clustering solutions with the goal of finding a similar behaviour. The emphasis lies on getting flexible results instead of highly tweaking certain algorithms as the problem can not be solemnly reduced to the mathematical performance due to missing values. Generally good and flexible results have been achieved with a combination of content-based-filtering and hierarchical clustering methods or the affinity propagation algorithm. Kernel based clustering results differed much from other methods and were sensitive to changes on the input data.

Zusammenfassung

Diese Arbeit stellt eine praktische Herangehensweise an die Evaluation von Clustering-Algorithmen und deren Performance auf verschiedenen hoch-dimensionalen und kargen Datensätzen dar. Solche Datensätze stellen hohe Anforderungen an die Algorithmen in bezug auf den Rechenaufwand und Annahmen, welche getroffen werden müssen. Es werden Ansätze zur Lösung und Optimierung dieses mehrstufigen Problems diskutiert. Distanzmatrizen und Recommender-Systeme wurden eingesetzt, um die Komplexität des Problems zu reduzieren und um fehlende Werte zu berechnen. Der Hauptfokus lag im Vergleich der einzelnen Methoden in Bezug auf die Ähnlichkeit der Resultate mit dem Ziel, ähnliches Verhalten zu finden. Ein weiterer Schwerpunkt lag auf der flexibilität der Algorithmen bezüglich den Datensätzen, da auch die Kargheit der Daten sowie die Dimensionalität einen grossen Einfluss auf das Problem haben. Generell wurden mit einer Kombination von Recommender-Systemen, hierarchischen Methoden und Affinity Propagation gute Resultate erzielt. Kernel-basierte Algorithmen waren sensitiv bezüglich Änderungen im Ausgangsdatensatz.

Table of Contents

1 Introduction

Within the last few years, a public awareness has developed about the huge amount of data being collected. This itself is no new insight, but it evoked an even larger attention on data processing. Often data is collected without any particular use, in expectation that this pile of data might be useful at some point. Because data is collected as is without a prior purpose, the dimension of such collected data is often very high and dimensions are usually sparsely populated because a feature or dimension usually only refers to a small subset of all possible features.

These characteristics add two questions to the general uncertainty concerning data, namely why data is missing at all and if the sparsely populated feature is relevant for solving the problem of grouping items into similar sets (clustering).

In supervised learning with known classifications the problem might be less serious due to possible conclusions that may be drawn from a certain class to its assertive features (principal components) and performance can be measured directly upon the data. Yet in unsupervised learning (clustering) there does not exist a direct validation of results. Of course there is never a perfect solution, but in unsupervised learning there is no direct measurement for measuring the validity of a clustering result. Various measures and indexes exist and might give hints on the results but usually have limitations.

Along with the problems described above, with high dimensionality and sparsity a huge technical challenge arises. Computations are far more complex, mathematical approaches aren't always applicable and existing algorithms behave much differently in large feature spaces than in smaller feature spaces.

In order to circumvent these shortcomings, various approaches have been proposed. Within the following chapters, some of them will be analysed on their impact on clustering results and their applicability on various datasets.

The general idea behind this thesis is to evaluate different algorithms on different datasets and compare their similarity. If there is an actual structure in the data, different algorithms should more or less derive a similar structure from it. Further, the variation of results has been analysed with increasing sparsity. Although these results highly depend on the algorithms and the given data, there is still evidence that a threshold exists when certain approaches yield strongly different results.

An large amount of different approaches on the problem of clustering high-dimensional and sparse data have been proposed. This is by no means a representative overview about all approaches. Generally each algorithm and approach has their own target group of datasets and problems. But the sheer amount of different algorithms and approaches makes it a difficult task to find a suitable way to process the data. Therefore several commonly used approaches are tested on their findings.

1.1 Structure of the thesis

In the next chapter, the used datasets will be introduced with their particularities. Then the general approaches examined to compete the challenges on the datasets will be described. This includes singular value decomposition, distance matrices and collaborative filtering approaches. Then the tested algorithms will be outlined. A special focus has been put on hierarchical approaches and spectral clusterings. Partially recommender systems have also been examined. The results are presented in the last two chapters, starting with the evaluation of recommender systems and continuing with the evaluation of the algorithms.

2 Motivation

The high importance of clustering can be derived from its appliances in many different scientific topics. It has been successfully applied in bioinformatics clustering gene data [Dhaeseleer, 2005], in commerce building customer groups [Alzate et al., 2009] and in machine learning extracting concepts from data. Various clustering algorithms have been developed and new concepts as collaborative filtering have emerged. These algorithms are usually tightly coupled to a given problem range. Not much evaluation has been done on comparing different approaches on the applicability of their solutions, especially not for high-dimensional and sparse data. Whether algorithms can be compared at all is of course questionable due to the differences between the algorithms. Due to the importance of this huge and emerging market there should be put an emphasis on the comparability of these approaches. The difficulty arises because of the incompleteness of data. The movieLens rating dataset, which will be introduced in the next chapter, is a good example for incompleteness of data. There will always be many missing ratings due to the fact that that nobody will ever be able to watch all movies.

This leads directly to the actual problem of clustering high dimensional sparse data and the evaluation of the results. The problem is posed in three stages. With high dimensional sparse data it is uncertain whether the data reflects the actual distribution or not. There is a chance that the data is strongly biased, especially for user content. This is due to the fact that users tend to comment on items either if they disliked the content or highly liked them but not in between.

The next possible cause for errors is the choice of approaches or algorithms. It is difficult to find a suitable approach with only a small subset of the data. Whether a choice was good or bad is hard to determine.

Last but not least the results need to be verified. Whether two movies are similar may be judgeable, but for more complex data as proteins in bioinformatics it is way more difficult. Additionally high-dimensional data cannot be easily visualized.

These three stages combined make clustering high-dimensional and sparse data a highly difficult task.

Another important topic in unsupervised learning is the narrowing of individualities and particularities of data. This itself has no ethical implications. However the usage of results may have, especially when there is no measure of validity.

So in these terms it is important to find measures for quality and validity of clustering results and a general way on steps to find valid clusters or like in this thesis, an evaluation on which approaches might work better than others.

3 Datasets

In this chapter, a brief analysis of the examined datasets concerning their structure, their measures and their particularity is provided. Different approaches are usually aiming at different ideal datasets, so the first step in clustering usually is the analysis of the dataset.

3.1 MovieLens Movie Ratings

Movie ratings gained quite some attention, not only since the Netflix Prize in 2009 [1]. In this prize, a bounty of one million dollars has been placed for delivering movie rating predictions for user who haven't seen the movie. The predictions had to outrun the their current algorithm on rating predictions on at least 10%.

The ratings dataset used in this thesis is not the same as in the Netflix prize but similar. Grouplens Research has published their collection of MovieLens ratings [2] in different collection sizes. The smallest set of 100'000 ratings by 943 users on 1682 movies is used in this thesis. Each user has rated at least 20 movies. Additional demographic information about the user (age, gender, occupation and zip) is available. However this demographic information has not been used in this thesis but may be used to extend the model for predicting ratings of users described in the next chapter 4.2.4.

In Figure3.1 the structure of the dataset is represented. White space indicates missing ratings from users. The ratings are more or less well distributed as there are no visible clusters of certain users only rating a certain subset of movies.

3.2 Fibronectin Protein Expression Set

This dataset and investigations on it can be found in [Comi et al., 2003], the data is publicly available [3].

The set contains a total of 1131 proteins with 28 measured expression levels. This is not very high-dimensional, however not low dimensional either. Originally the dataset is not sparse but values have been randomly removed in steps of 10% to compare how results vary with increasing sparsity.

[1]http://www.netflixprize.com
[2]http://www.movielens.org
[3]http://pevsnerlab.kennedykrieger.org/microarraydata/

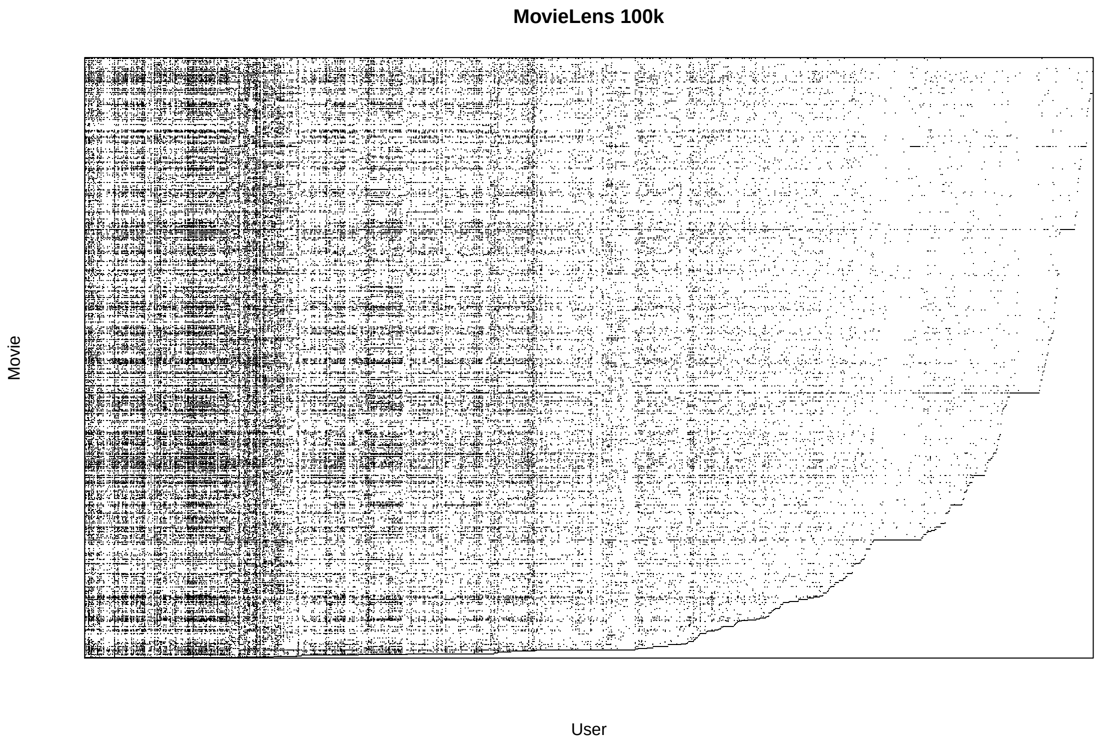


Figure 3.1: MovieLens Ratings Structure

3.3 Execution workflows

The last datasets investigated is a set containing values on how much a certain workflow suits a certain dataset. There is one dataset containing the root mean squared error for the predicted classification of categorical attributes and one for the regression of numerical values. The structure is different to the other datasets and highly emphasizes certain clusters already which can be seen in 3.2. It may be questionable whether such data should be clustered as is or if the data should already be divided as a first step. However there is still a point in examining the behaviour of approaches for such data.

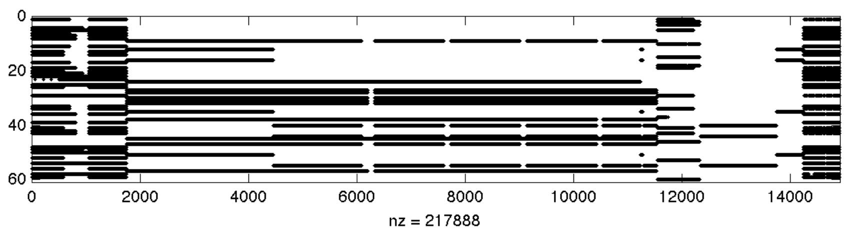


Figure 3.2: Classification Workflows Structure

4 General Approaches

There is a wide range of possible clustering approaches. Some are more widely used, some less. Some emphasize certain particularities and assume a certain distribution on the data. Some tend to smooth outliers better. So taking the first steps in clustering given a certain problem is difficult. The sheer possibilities are frightening. And it is even harder to know whether an approach performs better or worse for high-dimensional and sparse data as much more assumptions have to be made than in common datasets.

In this chapter, an analysis of of the problem and and descriptions of possible approaches is given. This is of course no complete set of all approaches but the ones highly discussed in recent scientific literature.

4.1 Implications of High-Dimensionality and Sparsity

Bellman's "Curse of dimensionality" [Bellman, 1961] describes the general problem of having much information without the possibility to evaluate all possible combinations since they are exponentially growing with increasing dimensions. Additionally it is much more difficult to visualize the data and to verify solutions visually.

Sparsity leads to two implications when clustering. It results in difficulties to decide about the proportions of the dimensions and their possible bias. Additionally when dimensions lack information, the problem emerges that it is impossible to decide about the relevancy of this dimension. This implies that the resulting distance measure may have a certain range of valid actual distances.

4.2 Missing Values

The first thing to think about is why data is missing. Missing data can be classified as "missing completely at random", "missing at random" and "non-ignorable missing values" [Acock, 2005]. Values that are missing completely at random have no ulterior cause for their absence. Although it may be challenging to find such reasons, for user based content as ratings it is often the case that missing values are completely random. Randomly missing values have a ulterior cause lying within other attributes or features of the dataset, so there is a chance to find the structure of missing values in the data. Non-ignorable missing values are missing due to a certain constraint in the possibility of responding. Such values cannot be ignored as this would strongly lead to biased data. In the datasets used in this thesis the analysis is relatively straightforward.

Some users simply haven't seen a certain movie. These values are missing completely at random. But there is a possibility that the user forgot to rate movies he has seen already. It is unknown whether someone actively refused to watch a movie or if a user simply did not hear about it. When actively refusing to watch a genre as it might be the case for certain genres, the missing value then would be missing at random and not completely at random. Hence to predict a rating for a user then might destroy information about the users preferences. Excluding professionals who rate more than thousand movies, there is a chance that individuals only rate movies when they either liked the movie especially well or the opposite. In such cases the missing values might be regarded as non-ignorable because using statistical approaches as mean or average would lead to strongly biased data. So in this dataset there is a chance for the occurrence of all types of missing values.

In the fibronectin dataset sparsity is created artificially. Of course it is arguable that it depends on the data whether missing values create a larger or smaller loss of information on the problem. However in this artificial case data is missing completely at random.

The work flow dataset is special in terms of that missing values are caused by work flows which simply aren't suited to the problem and can never be executed. Hence the missing data is not actually missing but containing information that cannot be expressed within the numerical boundaries of the values.

The lack of ability to define whether two instances are close in a subspace due to missing values usually leads to ignoring this dimension when measuring the distance. With high-dimensional sparse data (implying many missing values!) the problem that arises is that possible subspaces lack important dimensions due to missing values. Now for pairwise comparisons between instances, each comparison takes place in a completely different subspaces. This leads to different distances measuring other dimensions. Each distance is then is relatively indifferent to the other. This makes comparison and clustering difficult as there is no common space to identify clusters. The same problem occurs in graph-based clustering where different measures may lead to completely random clusters. Four approaches on filling missing values are described below.

4.2.1 Deletion of Missing Values

The standard procedure found in practically all clustering environments is that missing values are deleted, either pairwise or listwise.

Pairwise Deletion

With pairwise deletion, attributes with missing values are excluded from the analysis and only values that can be compared are analysed. This has the advantage that not all dimensions with missing values are excluded from the distance calculation. This leads to different dimension sample sizes during the bivariate correlation calculation and hence calculations aren't always on the same dimensions. As previously described, this may lead to infeasible graphs where edges derive from different subspaces and therefore aren't comparable any longer. Hence the validity of this approach decreases with increasing sparsity and dimensionality.

Listwise Deletion

Listwise deletion deletes all cases with missing values. This has two implications. For highly sparse datasets as the movie ratings set there would be no rows left as each row contains missing values. But even if some cases remain, the data set will be highly reduced. But distances can be calculated on same dimensions. But with a smaller subset of cases, the chance of having biased data increases. Using this approach assures that meaningful comparisons can be made but its feasibility depends on the data and whether the problem allows removal of cases.

4.2.2 Substitution

Another common approach is to substitute missing values with the mean value assuming a normal distribution in the data. However this approach is not suitable for highly sparse data. Although even if the data is normally distributed and the mean may not be a bad guess, this leads to many cases having the exact same value. The consequence is a variance of zero meaning that for most cases the importance of the feature or attribute is neglectible. For example taking the mean movie ratings for each user for unrated movies, the importance of this user will decrease during clustering as there unrated movies to not differ from each other and will probably be clustered together. This problem becomes more apparent with distance matrices (which will be described later) as cases with missing values filled with the mean of the attribute will have a high similarity. This will lead to larger clusters and there is a tendency that no clusters can be found as there will be high similarities between cases throughout the dataset.

One possibility is to build subgroups and then only take the mean of the subgroup. However depending on the data it may be unfeasible to determine appropriate subgroups. For the movie ratings the genre may be taken. For the FibroPlast set proteins with a similar chemical structure could be taken. However this entails the chance of building completely wrong subgroups and the actual structure of the data may not be detected.

4.2.3 Imputation

Imputation of missing values is a more sophisticated approach to avoid the shortcomings of using mean values. There are several different approaches, the most common is using EM (expectation maximization).

Closely related to imputation is the term ”Collaborative Filtering”. However its mostly related to predicting user preferences for a certain item. So it’s actually suited to the movie rating set, however it has also been applied to the protein set in this thesis. The general approach is similar to other statistical models and various different models have been proposed. Some of them are mentioned in the next chapter.

Single Imputation

Single imputation takes the coherences between attributes to estimate the maximum likelihood value. A description of the algorithm using EM can be found in [Dempster et al., 1977]. When only little data is missing this approach might yield good results. However when much data is missing the resulting values lack again in variance and filled values are usually treated as normal values during the next iteration. This may lead to unexpected behaviour and misleading filling of values. An advantage is that calculation costs are relatively low, but still this approach isn't suited for highly sparse data while having many dimensions may be an advantage since more attributes can be analysed.

Multiple Imputation

Multiple imputation takes several sets and then calculates then estimates a suitable model. With confidence intervals the coherence between the imputed datasets can be analysed. This leads to a more uniform distribution. Further information can be found in [Wayman, 2003]. The drawback is that the calculation is expensive, especially for high-dimensional data. And depending on the missing values and the problem, results may still be bad.

4.2.4 Imputation based on Meta Information

During the Netflix prize, Simon Funk has posted an interesting approach on his blog [1]. It is often referred as SVD (which is explained in the next section) but only in terms of taking information about the user into account. The dimensions like age, zip, Profession, Genre and Actors are reduced into a single rating prediction for each user and movie.

General Idea

The general idea is to build a learner which learns from a certain feature set. In this case actor and genre has been used. However the model could easily be extended with additional information as gender and profession. Then the existing ratings are analysed based on the selected feature set. For each rating its features are compared. So if the user always rates (and likes) action movies, the importance of this feature for the user increases. In the end each user contains a feature set containing the importance of features for him. A linear predictor based on the feature set is built which multiplies the average movie rating with the features, adapting it to the users preferences. The quality of the prediction then depends on the accuracy of the users ratings and the accuracy of the model (e.g. if the problem can be reduced to the selected features and if they are independent).

The Math

$$UserRating_{Movie} = (\prod_{k=1}^{n} UserFeature_k \times MovieFeature_k) \times (Value_{Movie} - AvgUserOffset) \tag{4.1}$$

The general idea behind the linear prediction is found in 4.1 where the $UserFeature$ is the importance of the feature to a user and the $MovieFeature$ is the grade how much the movie reflects this feature (and zero if the movie is not related to this feature). There is only information whether a movie has a certain genre or not in the movie ratings set, same goes for actors. So the $MovieFeature$ is either zero or one. The $Value_{Movie}$ is the average of all ratings (even the predicted ones).

[1]http://sifter.org/ simon/journal/20061211.html

When training user features, as a first step the error 4.2 is calculated between the actual rating and the predicted one. The learning rate [2] is also taken into account as the higher the error the more quickly the model learns. To note is that the predicted rating already includes all $UserFeatures$ and $MovieFeature$.

$$Error_{Prediction} = (LearningRate \times ActualRating) - PredictedRating_{Movie,User} \tag{4.2}$$

The actual learning process is found in 4.3 and 4.4.

$$UserFeature_k = Error_{Prediction} \times UserFeature_k \tag{4.3}$$

$$Value_{Movie} = Error_{Prediction} \times Value_{Movie} \tag{4.4}$$

The importance of the user feature for initial values is always one while the movie value may be set to the mean of all its ratings at the beginning. However there the problem with sparsity appears again. If a movie only contains a few bad ratings, there is a chance that this does not reflect the distribution. So assuming a normal distribution, Simon Funk proposes a blending approach between the mean of movie ratings and the mean over all ratings of all movies, taking the variance between ratings into account. 4.5 denotes the blended initial mean rating where k is the ratio between the variance of all movie ratings and and the variance of the movie specific ratings. This yields the advantage that a movie's value tends more in direction of the average mean of all movies when only few ratings have been made.

$$MeanRating_{Movie} = \frac{AverageRating_{AllMovies} \times k + (\sum_{k=1}^{n} Ratings_k)}{k + n} \tag{4.5}$$

Depending on the number of ratings a user has made, there is a possibility that a certain feature for a user has a huge impact on the predicted rating. With a large training set this problem may be diminished, however the smaller the training set the more likely it may occur. So a way of penalizing the magnitude of feature vectors is necessary [Tikhonov, 1943]. The approach is analogue to the mean rating, the 4.3 and 4.4 can be adapted so that k (the variance ratio denoted above) is also taken into account.

Possible Improvements

As proposed in [Piotte and Chabbert, 2009] additionally to the above model time variables may be regarded. The assumption there is that ratings made directly after seeing the movie and which are more recent are more guiding than others. So to distinguish them the timestamp can be built into the model. While above model is linear, it is sometimes useful to peform a nonlinear transformation (a sigmoid function is proposed) on the output to get a more natural result. However these possibilities are concerning the estimation of predictions, hence they have not been investigated more. The model built and used in thesis does not use timestamps.

[2]Simon Funk proposed a learning rate value of 0.001

Similarities

The approach above is similar to the backpropagation algorithm [Rumelhart et al., 1986] with a linear output neuron where the prediction error defines the adaption of neurons. Each feature vector would be a neuron in an artificial neural network.

4.3 Singular Value Decomposition

This approach is closely related to missing values, however not entirely and therefore placed in its own section. For clustering, singular value decomposition brings two advantages. The first advantage is the resulting dimensionality reduction, minimizing the impact of the "curse of dimensionality". A second advantage is that missing values do not need to be filled explicitly as the dimension reduced matrix derived by svd approximates may fill the original matrix without missing values (approximating the whole matrix through the dimensionality reduced matrix). So either singular value decomposition can be used as a standalone approach or in combination with filling missing values in finding a dense representation for the sparse matrix.

4.3.1 Mathematical Meaning

Singular value decomposition is a factorization of the original matrix into two matrices containing each the left singular vectors and the right singular vectors. While eigenvectors and eigenvalues only exist for quadratic matrices, singular values exist for non quadratic matrices too. Similar to eigenvalues, singular values describe the characteristics of a matrix but usually in a space with fewer dimensions.

In data mining often the term "Principal component analysis" (PCA) is used. PCA is usually done with the help of singular value decomposition to find the most important dimensions by finding an orthogonal subspace.

4.3.2 Challenges with High-Dimensional Data

Traditional approaches on singular value decomposition do not work with missing values. Additionally the calculation gets expensive with increasing dimensions, necessitating extensive resources. However various approaches have been developed that perform a partial svd of a sparse matrix. For further information on SVD [Halko et al., 2011] is recommended. A good summary on the topic concerning the netflix data is provided in [Kurucz et al., 2007] where the Lanczos method is recommended.

4.3.3 Implications of SVD

While in Simon Funk's approach a model is built, SVD approximates a given matrix through one with lower dimensions. This approaches may may result in an information loss. But the partial SVD result depends on the representativity of the values in the original matrix. Further implications on svd and high-dimensional data can be found in [Becavin et al., 2011] where it is stated that, depending on the method, the magnitudes of the singular values with increasing dimensionality do not change, hence SVD may return results with a certain randomness. Usually experiments have to be done on how many dimensions should be used.

4.4 Distance matrices

Clustering usually involves two steps. Evaluating the similarity of two instances and then decide whether they belong in the same cluster or not. Many clustering algorithms take a distance matrix as input. This has the advantage of leaving the option to the user what kind of distance measure is used but has the disadvantage that the algorithm has to adapt on the scaling of the input. A distance matrix can be seen as a graph, each cell contains the distance (or dissimilarity) between two instances. Distances may be measured in an euclidean space. Depending on the problem, other spaces should be considered too. For some data the space might be unknown. The problem with high-dimensional data is that it is very likely that some dimensions differ from others in terms of subspaces or scaling.

As previously stated, calculating distances is problematic with missing values due to the fact that distances are calculated in different subspaces and are therefore no longer representative. Usually as with imputation of data calculating a distance can be done either with pairwise or listwise deletion. But again only pairwise deletion is applicable for sparse datasets, but with the chance of getting infeasible results.

There are various distance measures as the manhattan distance, gower distance, euclidean distance and jaccard distance. The Jaccard distance (using the jaccard index described below) measure is often mentioned with sparse matrices.

4.4.1 Jaccard Index

The Jaccard index is a form of distance introduced by Paul Jaccard [Jaccard, 1901]. The reason for its relation to sparse data comes from the fact that it only measures the occurrences of observations and not the absence. Assuming a questionnaire containing questions with binary answers, two responses are only similar when both answers were yes but not when both answers were no. There is an advantage in terms of applicability on sparse datasets and filling missing values with zero as they would not affect the final result while other distance measures would take the 0 into account. This may be useful for actors and genres where both movies contain the same actor a similarity can be deduced. But the opposite case is no sign for similarity. This has the advantage that more data can be used but the problem with different distance subspaces remains. In case of the ratings, this would omit the information about individual user preferences assuming that all users preferences are the same yielding a global measure on movies with higher ratings. However this still is prone to returning irrelevant results.

4.4.2 Pearson Similarity Coefficient

A common distance metric is the Pearson Similarity Coefficient (PCC) [Eisen et al., 1998]. It is often used in bioinformatics. It is based upon the covariance of two features taking the standard deviation into account. This is especially useful to denote dependencies between variables as it denotes the correlation between them. Another view on the PCC is that it behaves similar to an euclidean distance, however as it takes the standard deviation into account it is not as prone to slightly different scaling between features. So this might be useful for movie ratings where a similar rating does not imply a similar satisfaction among users.

4.4.3 Visualizing Distance Matrices

Distance matrices are a way to simplify high dimensional data as relationships between instances are reduced to a single measure. In figure 4.1 and figure 4.2 the voronoi decompositions of the distance matrices are presented. A voronoi decomposition of a distance matrix is a visual representation of a distance matrix. In the figures each movie corresponds to a node. It is interesting to see that figure 4.1 seems to have a better distribution than the distance matrix based on imputed data in figure 4.2. Yet for clustering using the imputed datasets seemed to give better results as explained in the results chapter. These figures should serve as an example to the question whether a distance matrix can represent high dimensional data adequately.

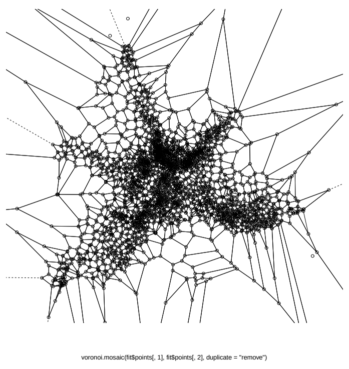


Figure 4.1: Voronoi Decomposition of Jaccard Distance Matrix using the Original Dataset

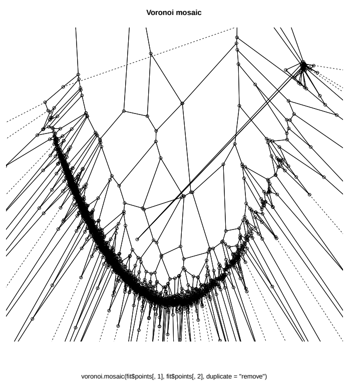


Figure 4.2: Voronoi Decomposition of Jaccard Distance Matrix using the Imputed Ratings with Simon Funk's Model

5
Methods

There is a wide range of different algorithms for clustering and indexes for evaluation. Most of them are tied to specific problems. The goal of of this thesis was not to examine which algorithms perform marginally better than others under certain circumstances but whether some are more flexible and robust than others (under different environments). The following selection is by no means a representative one of existing algorithms, however the goal was to select algorithms of different kinds.

5.1 Partitioning Methods

Partitioning methods are either centroid or medoid based, depending whether they cluster instances around the gravity centre or build the cluster upon an average dissimilarity in the cluster. Generally these types of clustering algorithms are relatively intuitive compared to others. The drawback is that with increasing dimensions and number of instances they become computationally expensive. This is because the distance for each point to its cluster center must be calculated and compared to the other distances. These algorithms usually are also prone to noise.

5.1.1 K-Means

K-Means is an often used algorithm for various types of problems. However opinions on its scalability vary, yet it is commonly agreed that it performs well on large datasets but not when feature dimensions are large too [Whitworth, 2010].

Canopy Clustering

Although not a very recent approach, canopy based clustering is related to clustering high-dimensional datasets [McCallum et al., 2000]. It is mainly an approach on reducing calculation costs but also decreasing the error of wrong instances in wrong clusters. Their approach is to find a cheap distance measure and build subgroups (canopies). Clustering is then done on these canopies, hence computational costs are reduced since they are only performed on the canopies. To note is that canopies can overlap.

Although this approach has not been examined further in this thesis, there are still possibilities for further investigations in this direction in terms of splitting the dataset and perform clustering on these subsets reducing the impact of the ”Curse of Dimensionality”. Especially in terms of the movie ratings set there is the possibility to divide the set into good (with a high average rating), mediocre and bad movies and then cluster them or group into genres. But since an item cannot change its canopy after it is categorized, this method may yield a strong bias and information may get lost. But depending on the canopy, there is a possibility that certain unrelated features may be omitted.

5.2 Hierarchical Clustering

Unlike K-Means where the number of clusters has to be defined before running the algorithm, hierarchical clustering seeks a hierarchic representation of clusters. Generally the complexity of hierarchical approaches are high depending on the type. Hierarchical clustering methods usually take a distance matrix as input. This has the advantage that the observations themselves are not needed as in k-means but limiting the possible amount of information per instance.

5.2.1 Agglomerative Hierarchical Clustering

This type of hierarchical clustering is a bottom up approach where each instance starts in its own cluster. Then clusters are iteratively merged together until the number of desired clusters is reached.

Ward's Method

Given a distance matrix (or a graph) there still needs a criterion to be defined when a cluster is formed. Ward's method [Ward, 1963] defines this as the minimal intra-cluster variance. Starting with clusters containing one instance (agglomerative), in each step the two points giving a minimal intra-cluster variance are merged. Ward's method has been used in this thesis, hence when a hierarchical method is mentioned it refers to Ward's method.

5.2.2 Divisive Hierarchical Clustering

In divisive clustering one initial big cluster is divided into sub-clusters until the desired number of clusters is reached. Conceptually this approach is more complex but has the advantage that not a complete tree needs to be calculated. The tree is only built until the desired number of clusters is reached. There is a chance that this approach might yield more accurate results than agglomerative methods as the problem as a whole is considered while a bottom-up approach can only find local optimizations when points are considered separately. Divisive approaches have not been examined further as initial tests yielded similar results to Ward's method.

5.2.3 Related Work

Related work on hierarchical clustering can be found in [Abbas, 2008]. It has been found that while K-Means tends to work generally better especially on huge (but not high-dimensional) datasets, hierarchical approaches work better on random datasets. Assuming that high sparsity, many dimensions and imputation of values may lead to a certain amount of randomness in the data, hierarchical approaches may have advantages for the problem analysed here. It is also stated that hierarchical approaches are less prone to noise. With many dimensions it is more difficult to talk about noise and outliers since complex cluster shapes may occur anyway. This is a second argument that supports the applicability of hierarchical approaches for high-dimensional sparse data.

5.3 Density Based Clustering

In density based clustering algorithms, clusters are defined as regions in a space with high density. Hence clusters can take arbitrary shapes while a minimum variance criteria may lead to a certain cluster shape. Another advantage is that such algorithms are less prone to noise they lie in areas with a low density. Further information on density based clustering can be found in [Kriegel et al., 2011b].

5.3.1 DBSCAN

DBSCAN takes two parameters, ϵ and the minimum number of points that may form a cluster. ϵ denotes the distance measure whether a point is considered a neighbour. For each point its neighbourhood is examined. Two points are density reachable when their distance (including other points) does not exceed ϵ. If enough density reachable points defined as the minimum number of points are found, a cluster is formed.

Due to the density based approach, DBSCAN tends to find clusters that are of arbitrary shapes while for example K-Means may have difficulties when data is not linearly separable (because a linear distance measure is taken from the possible center). Additionally the number of clusters does not need to be defined in advance and DBSCAN is not as prone to noise. Disadvantages are that the quality of DBSCAN results also depend on the distance matrix given as input. Due to the given maximal density reachability there is also a chance that if the data contains many different regions with different densities it cannot be clustered adequately with DBSCAN. This may especially be the case for large datasets (and when missing data is imputed).

5.3.2 Related Work

DBSCAN is an often used and cited algorithm. However as described it has its shortcomings with high-dimensional data. Therefore various approaches have been proposed to avoid the problems with many dimension. With many dimensions, different clusters may exist in different subspaces. PreDeCon [Boehm et al., 2004] extends DBSCAN with the concept that cluster points must have a similar subspace preference (having a strong magnitude in certain features) and may therefore perform better on high-dimensional data. More recent work to also handle dynamic data can be found in [Kriegel et al., 2011a]. However probably the problem of different densities still persists. These approaches have not been examined further in this thesis.

5.4 Spectral Clustering

Spectral clustering is not entirely distinguishable from other clustering approaches and related to SVD and kernel principal component analysis (KPCA). While SVD calculates singular values, spectral clustering algorithms use Eigenvectors and Eigenvalues hence requiring a square matrix (usually a distance matrix). The similarity to KPCA refers to the used kernel which transforms the data with a given kernel method (linear or non-linear), allowing further analysis and hopefully easier distinction between clusters. The work on this topic is immense covering many different research fields. A good introduction can be found in [Luxburg, 2007]. In this thesis, kernel based clustering algorithm proposed by [Ng et al., 2001] is used.

5.4.1 Related Work

It has been found that in terms of graph partitioning, kernel based k-means (meaning transforming the data in advance) performs just as effective as spectral methods [Dhillon et al., 2004]. It has been examined that K-Means with transformed input data behaves similarly [Welling, 2009] to normal k-means.

5.5 Other Clustering Methods

The following algorithms are not directly related to the above families of algorithms but are also examined in this thesis. They have been selected as they are often cited or mentioned and using different approaches than above listed algorithms.

5.5.1 MCL - Markov Cluster Algorithm

MCL is graph clustering algorithm based on markov matrices [van Dongen, 2000] and is often used in bioinformatics. The algorithm performs random walks through a graph. The paradigm is that within the graph, strong flow (meaning high connectedness) is promoted and weak flow is demoted. The flow through the graph is modelled by a markov matrix. The basic operations are expansion where unconnected nodes are being connected, and inflation, where the flow is either weakened or strengthened. To note is the fact that mcl does not use complex procedures as joins or splits, making the algorithm computational highly feasible.

Yet there are limitations. As stated by the inventor of the algorithm, it may have problems with tree based graphs. There is a chance that especially a sparse high-dimensional matrix may lead to tree like structures when there is an ulterior cause for missing values (especially random missing values). Also the difficulty with distances measures referring to different subspaces may lead to tree like structures. As a practical example ratings can be regarded. Assuming a user likes and rates a certain movie and then watches and rates similar movies with a lower rating, this may lead to a problematic hierarchical structure.

This is also a main difference between MCL and other graph based clustering algorithms as DB-SCAN or hierarchical approaches that there is no horizontal but rather an as is representation of the graph.

5.5.2 Affinity Propagation

Affinity propagation (AP) is a recent algorithm that has been developed at the university of Toronto [Frey and Dueck, 2007]. AP also takes a distance matrix as input. Unlike K-Means and similar to hierarchical agglomerative approaches, AP considers all points as possible cluster centers (cluster reference point called exemplar). Then messages are exchanged between all points on the existing edges. Each point has an availability and a responsibility for all other points. The responsibility reflects how suited another point is to serve as an exemplar and the availability denotes the appropriateness for a certain point to be exemplar. This considers the appropriateness for other related points also. Iteratively these values are optimized, reducing the availability when a point is assigned to an exemplar and updating the responsibility as the maximum similarity minus the maximal responsibility and similarity of one related point.

AP has been proposed for various kinds of datasets including gene microarray data, clustering face images and clustering of sentences in a document. The argue that the good performance of affinity propagation lies in the fact that it does not take a random samples for cluster centers but considering all points as possible exemplars [1]. This is a clear advantage for large datasets with few or an unknown number of clusters. Additionally the message passing between points has a certain similarity to spectral clustering methods, however in spectral clustering no exemplars for clusters are chosen. Hierarchical agglomerative methods are similar that each point initially has its own cluster, however during the merge poor decisions cannot be undone in later steps while with Affinity Propagation, no such decisions are made during clustering and exemplars may change during the clustering process.

5.5.3 Biclustering

Biclustering (also called two-mode clustering) is a different approach on clustering where rows and columns are clustered altogether. The goal is to find sub matrices that are similar in terms of their pattern. Biclustering therefore differs from many other approaches in not taking a distance or similarity matrix but the raw data. The important tuning parameters are not the number of close neighbours and a certain similarity threshold but the minimum number of columns and row a cluster must contain and optionally the desired number of clusters among other parameters depending on the algorithm.

There are many different algorithms that use a biclustering approach. An extensive survey on different algorithms is found in [Prelić et al., 2006] and Bimax has been proposed. In this thesis, biclustering will refer to the Bimax algorithm.

5.5.4 Related Work

A comparison on MCL and Affinity Propagation can be found in [Vlasblom and Wodak, 2009] where it has been stated that MCL performed much better in concerns of tolerance and robustness, especially with noisy data. For above difficulties with tree based structures Akama et al. proposed a branching of the graph [Akama et al., 2007]. It has also been found that MCL performs poorly on sparse graphs [Mishra et al., 2011].

[1] http://www.psi.toronto.edu/affinitypropagation/faq.html

5.6 Recommender Systems

Recommender systems are a very recent development in comparison to clustering algorithms. Recommender systems aim at creating user profiles to recommend similar items given an item. A thorough introduction to recommender systems can be found in [Ricci et al., 2011]. Such systems have emerged due to the data which is usually very high-dimensional. The topic is indirectly related to clustering as usually only a certain amount of items are recommended. But the challenges are similar, namely finding a cluster of similar items of given size for a certain user.

5.6.1 Methods

There are various algorithms available (e.g. k-nearest neighbourhood and Pearson Correlation as described in 4.4.2 which are widely used). Yet two main approaches are distinguishable from each other.

Collaborative Filtering

Collaborative filtering approaches aim at finding similarities between users through information on what they like (e.g. user x rated the same movie as user y). A often mentioned family of algorithms is Slope One [Lemire and Maclachlan, 2005] which is examined in this thesis and is described in 7.2.3. Their approach is computationally very feasible using a linear regression between ratings. Using this regression for each movie and user an estimate can be given how much the movie is recommended for the user.

Content Based Filtering

Content based filtering uses meta-information on rated items to estimate their similarity. Similar to Simon Funk's 4.2.4 approach on predicting ratings which can be regarded as content based filtering, meta information is used to build a user profile finding the users preferences.

6
Evaluation

There are many different measures for evaluating and validating clustering solutions. A good summary on this topic can be found in [Manning et al., 2008]. In the following pages selected measures and indexes are presented. They are by no means representative but have been used and examined in this thesis.

6.1 External Criterions

A natural approach is to decide whether a clustering solution makes sense in an intuitive way. This assumes that a possible model already exists on the given problem. This is applicable for the movie dataset but may be not be applicable in more conceptional problems. Another problem that arises is that such verification involving people's opinion includes expensive inquiries. Especially with large datasets chances are small that an individual evaluation covers a significant amount of the solution. This is directly related to the problem of sparse data. Clustering is made on a lot of assumptions when data is missing. During the evaluation data is missing again as questionnaires can hardly contain all clusters. Of course randomized questionnaires may still give statistically significant results [Acock, 2005]. So there is need for mathematical ways to evaluate cluster solutions.

6.2 Internal Criterions

An often used measurement is the intra-cluster variance and the inter-cluster variance. This function is used in practically all clustering algorithms. The goal is to minimize the variance of all points in a cluster. In other terms, goal is to find maximal similarities within a cluster. This makes intuitively sense as points which are close in the space should be placed in the same cluster. The inter-cluster variance is not as significant as the intra-cluster variance but still often analysed. The goal is to maximize the inter-cluster variance, assuming that clusters are better with decreasing similarity.

Although it is a measure how mathematically accurate results are, it is not a measure how well the actual clustering is. Especially with high-dimensions as previously discussed distances may not have enough information. Additionally there is evidence that points that are close in the feature space may not belong in the same cluster as they are placed in. This is due the fact that they may be close in the entire feature space but are related in totally different subspaces. Additionally when much data is imputed, there is a tendency that along one feature vector variance is small.

This ultimately leads to a cluster along this feature vector as points are close and hence a good intra-cluster variance is reached, but based on made-up data.

6.2.1 Dunn Index

The dunn index [Dunn, 1973] is a direct measure for the quality of a cluster algorithm. The index takes both the intra-cluster distance and the inter-cluster distance into account, seeking to maximize the ratio of the smallest distance between clusters and the maximal distance of points within a cluster.

Yet the index has a special characteristic. Due to the mathematical formulation of the index there is a chance that unequally balanced clusters (in terms of intra-cluster and inter-cluster distances) cause the index to be very low. This has two implications, namely that the index is prone to noise and that in a high-dimensional feature space distances tend to vary much more than in a low feature space. Large cluster diameters including outliers maximize the intra-cluster distance and therefore cause a low index. A high-dimensional feature space does not imply a low index but sparsity may lead to points located in various subspaces which can lead both to either maximal distances between clusters but also maximal distance between cluster points. Hence there is a chance that with high-dimensional data the index is low.

6.2.2 Entropy

The entropy referred to in this thesis is the entropy of the distribution of points among clusters [Meilă, 2007] hence the uncertainty associated with a cluster assignment to a point considering other possible clusters. The entropy measures the number of possible clusters a certain point could be in. With three clusters, the entropy would be $\frac{1}{3}$ or 3 in terms of Bits. The entropy is usually denoted in bits.

6.3 Natural Clusters

Clustering helps in finding concept groups. Instances are grouped together, each cluster then may act as a concept space. Especially in terms of recommendation systems meaningful clusters are searched as the underlying model is human judgeable. The concept should be comprehensible and capture the natural structure of the data [Steinbach et al., 2003]. But judging the quality of clusters concerning their meaningfulness implies a clear goal on how the clustering results should be handled. Different clustering results may have different meaningfulnesses for various problems. So in terms of movie recommendation it is straightforward to judge whether a movie was a good recommendation or not but this is not easily doable for other datasets. An initial reason why a recommendation was good might differ from the real reason the clustering algorithm found to recommend a certain movie, hence an algorithm may fail for other steps.

Often clustering algorithms are used to assign labels to instances which are then later used for machine learners to predict a classification. Yet the problem arises that when the clustering algorithm performs bad, the learner may not be able to deduce a concept from it. The chance that completely different underlying concepts may be deduced cannot be neglected. So whether a cluster is meaningful and natural depends on the actual problem and the judgement of the user and cannot be calculated.

6.3.1 Blending Results

If several tests with a different training and testing set on an algorithm yield comparable results it could be assumed that a general concept has been found. This works well especially for smaller datasets with few cluster centers. With high dimensional data the amount of possible concepts increases drastically simply through the fact that there are far more attributes. The chances are therefore that with different training and test sets the amount of possible concepts is large so that a general concept cannot be deduced. However often it is recommended to blend results. This means a combination of different approaches that contribute to the final solution. The BellKor Team that won the Netflix Prize did blend various models to improve their prediction of the rating and used different matrix factorizations (4.3) for dimensionality reduction to reduce the impact of non-suitable factorizations [Koren, 2009]. In other terms they did look for natural concepts that are backed by several different approaches.

The assumption in this thesis is that a similar blending may be done on cluster solutions and additionally, a possible judgement on the quality of clustering may be received when different algorithms yield more similar results than others. This also has a certain relevance for the imputation of data. If results are entirely different for one algorithm depending on the input data (e.g. the original data matrix and the decomposed matrix containing the singular values), then results should be judged with caution as a certain randomness is implied.

Although blending of results has not been examined further, the similarity of cluster results has. Assuming that two clustering algorithms return similar solutions these solutions must have a certain significance. For this approach two indexes have been used.

6.3.2 Rand Index

The rand index [Rand, 1971] is a measure of similarity for two set partitions and hence directly applicable on clustering results. The range goes from 0 (no similarity) to 1 (identical partitions). An intuitive definition of the rand index is given in 6.1 where I and J are the partitions of a set.

$$R_{I,J} = \frac{Agreements_{I,J}}{Agreements_{I,J} + Disagreements_{I,J}} \tag{6.1}$$

Further information on the rand index, the mathematical background and an estimation of the validity, especially on multi-class problems, can be found in [Santos and Embrechts, 2009]. They also proposed the rand index for feature selection and have found that using twice as many features (attributes) as classes the quality of results increases. This also leaves options for other feature selection approaches. One could randomly choose features and then run various clustering algorithms on these features until clustering algorithms yield more similar results. The implication of this would then be that these features are significant for the problem or at least not misleading as all algorithms deduced similar solutions from them. However this approach has not been examined further in this thesis.

6.3.3 Variation of Information

Another measure of similarities between cluster solutions is the variation of information [Meilă, 2007]. It is defined in 6.2 where H_x is the entropy of a cluster solution and $I_{x,y}$ is the shared information between cluster X and cluster Y.

$$VI_{x,y} = H_x + H_y - 2 \times I_{x,y} \tag{6.2}$$

6.4 Random Clusters

Another possibility to evaluate a cluster solution is to compare it to clusters that were assigned completely random and their corresponding indexes. This gives hints on indexes whether the given value has some significance or if it is random. This is especially useful in high-dimensional data where it may occur that distances are large. But it is also a measure about the impact of outliers. Assuming that in large datasets there is a significant amount of outliers which are still clustered, these outliers will have a large impact on the indexes. When clusters are randomly assigned, the distance to outliers may not be much larger than with a normal clustering.

6.5 RMSE

The root mean squared error (RMSE) is an often used measure for recommender systems when predictions are made how a user would rate a given item. It takes the squared deviation between the actual rating and the predicted one. So it is a measure for the performance of the predictor and hence for the quality of the recommendation. In 7 the RMSE of different recommender system algorithms has been evaluated.

6.6 Mantel Test

The mantel test determines the correlation between two matrices [Mantel, 1967]. Especially with distance matrices, changing one point in the feature space can lead to drastic changes in the distance matrix. Mantel proposed a randomization approach to avoid this problem by assessing the correlation many times by randomly permuting rows and columns. This has the consequence that the measure is adequate for distance matrices where different distances are measured. In this thesis this measure has been mainly used to determine the similarity between the input distance matrices.

6.7 Visualizing Results

Usually when clustering results are evaluated, a graphic representation on a 2D grid is given. As described before, it is difficult to find an adequate representation of high-dimensional data in a low dimensional space with far lesser dimensions. For the following figures, the distance matrix resulting from the original set has been non-metrically scaled [Venables and Ripley, 2002] as has been made for the voronoi decompositions (4.4.3). As nice as the solution may look in figure 6.1 and 6.2, they may not be representative for the actual problem. This especially stands out when the distance matrix is not based on the non-metrically scaled data but the original distance matrix. Results based on the original dataset differ much as shown in figure 6.3. So it is not distinguishable whether clustering results are bad or data received by scaling is not representative.

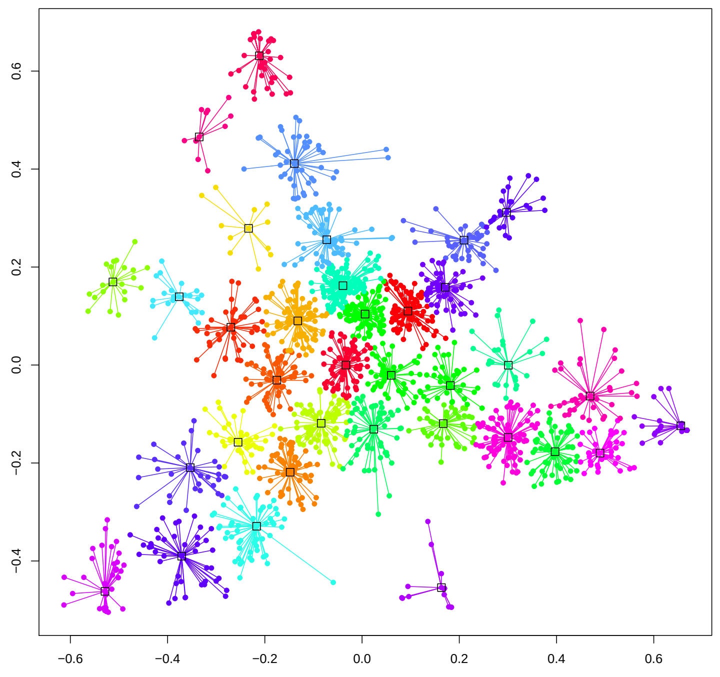


Figure 6.1**:** Affinity propagation results on the the non-metrically scaled distance matrix

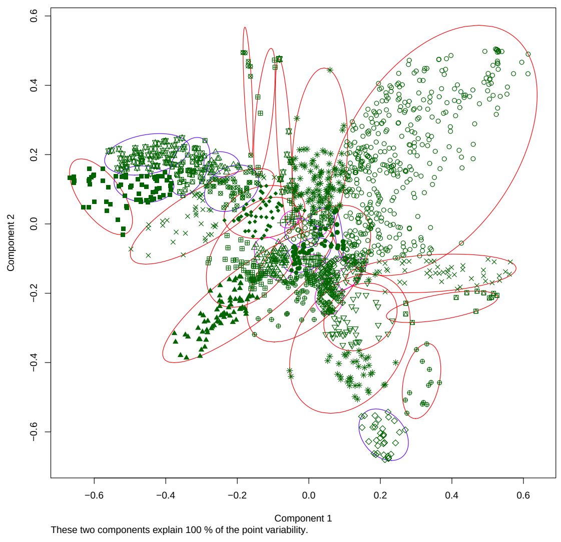


Figure 6.2**:** Hierarchical clustering results on the non metrically scaled distance matrix

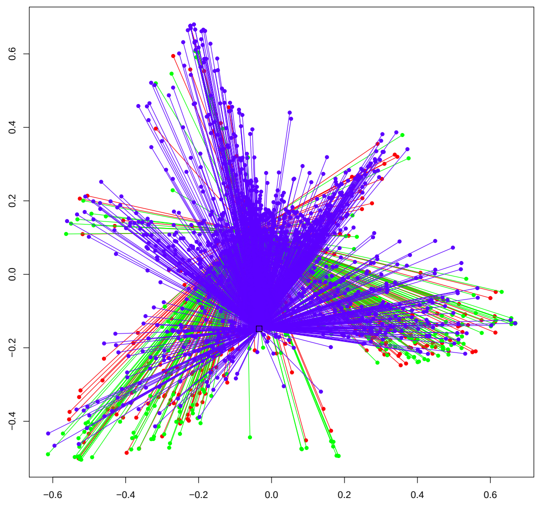


Figure 6.3**:** Affinity propagation results on the original dataset visualized in the non-metrically scaled space

7 RMSE and Recommender Systems

In this chapter commonly used algorithms of recommender systems have been examined. As described previously, recommender systems usually predict missing values and then recommend similar items.

7.1 Relation to Clustering

The approach was to find a way on how data can be transformed for later processing and to circumvent the missing value problem. As by now, partial singular value decomposition has been described to avoid problems with missing values (4.3). Additionally a content based predictor (4.2.4) may be used to fill missing values. Completely ignoring missing values is usually not a good option.

The last approach is that recommender systems may be used to output how much each item would be recommended for a certain user. Usually this is just used for recommending a handful of items. This is just another representation of predicted ratings with same underlying assumption.

It would be expectable that again results in the end should hopefully have a certain similarity to other methods as SVD or Simon Funk's predictor in terms of natural clusters or graphs. These recommendations will later be used for clustering, but as a preliminary step the root mean squared error (6.5) has been evaluated on common predictors to decide on their suitability to the problem.

There is room for argumentation that recommender systems are not suited for other datasets as movie ratings like the fibronectin set as the underlying distribution may be different. In terms of the RMSE it does not actually matter as only existing values are evaluated and hence only the quality of the model is measured. However it still might be of interest how recommenders perform on other datasets and how predictions and their corresponding cluster solutions compare to other approaches. The imputations for missing values on the fibronectin set have been analysed in 8.6. This underlies the assumption that for natural clusters (6.3) algorithms should more or less preserve characteristics of the given data and even though different approaches are used the output should not differ strongly between solutions.

7.2 Recommender Algorithms

With recommender systems there are usually two main approaches, item based and user based ones. Item based algorithms take similarities between items as input and then look for similar items to recommend. User based algorithms look for similarities between users and then recommend items based on similar users. In the following tests, three approaches have been examined.

7.2.1 N-Nearest Neighbour

The user based algorithm used in this thesis is the n-nearest neighbour algorithm using the Pearson Correlation Coefficient 4.4.2 for similarity. This algorithm is commonly used although it has been found that it does not perform well in high dimensional spaces [Weber et al., 1998]. Especially when dimensions increase the computation time drastically grows.

7.2.2 Generic Item Based

The item based algorithm uses a very simple approach just looking for the most similar items without searching for similar users. Here the loglikelihood similarity [Dunning, 1993] has been used as item similarity. Item based approaches are often used in commercial applications as they are quite fast and similarities between items are usually static while user based similarities are more dynamic and pre-computation to save computation time is more difficult. Ratings are usually predicted through linear regression.

7.2.3 Slope One

Slope One is an item-based approach, but it is yet even more simplified than the generic item based algorithm 7.2.2. To reduce overfitting of the linear regression, slope one just takes the average difference between two movie ratings as a linear factor [Lemire and Maclachlan, 2005]. Although the algorithm is relatively simple, it has been proven that slope one can compare with more sophisticated algorithms [Cacheda et al., 2011].

7.3 Results

Results on each dataset reflect the average value of ten executions per algorithm. The sparsity is artificially generated by limiting the training set to the given sparsity and then evaluate the results on the remaining data. The movie ratings set is already highly sparse (about 6%) but even more sparse when not all ratings are used for training. The Fibronectin set intially is a dense matrix but with a smaller training set, the impact of missing values can be observed.

7.3.1 Movie Ratings

In 7.1 the results of the predictors on the movie rating set can be found. Interestingly all collaborative filtering approaches behave similarly while the generic item based one seemed the most stable. Yet some randomness has been observed in lower sparsity levels as it depends whether a good training set is received or not. A sparsity level of 0.1 already responds to training set of 10'000 ratings and only 0.6% of values of the whole matrix. For lower sparsity levels not enough ratings were available for successful predicting. While slope one worked well with fewer ratings, n-nearest neighbour did not yield results anymore. This probably is due to the fact that best rmse results were received with 100 neighbours taken into account. For reference, with Simon's content based filtering approach (4.2.4) the received rmse on a 60%/40% training/testing set split was 1.06. It must be stated various sources had much better RMSEs reaching almost 0.9. So there is space left on tweaking Simon Funk's model.

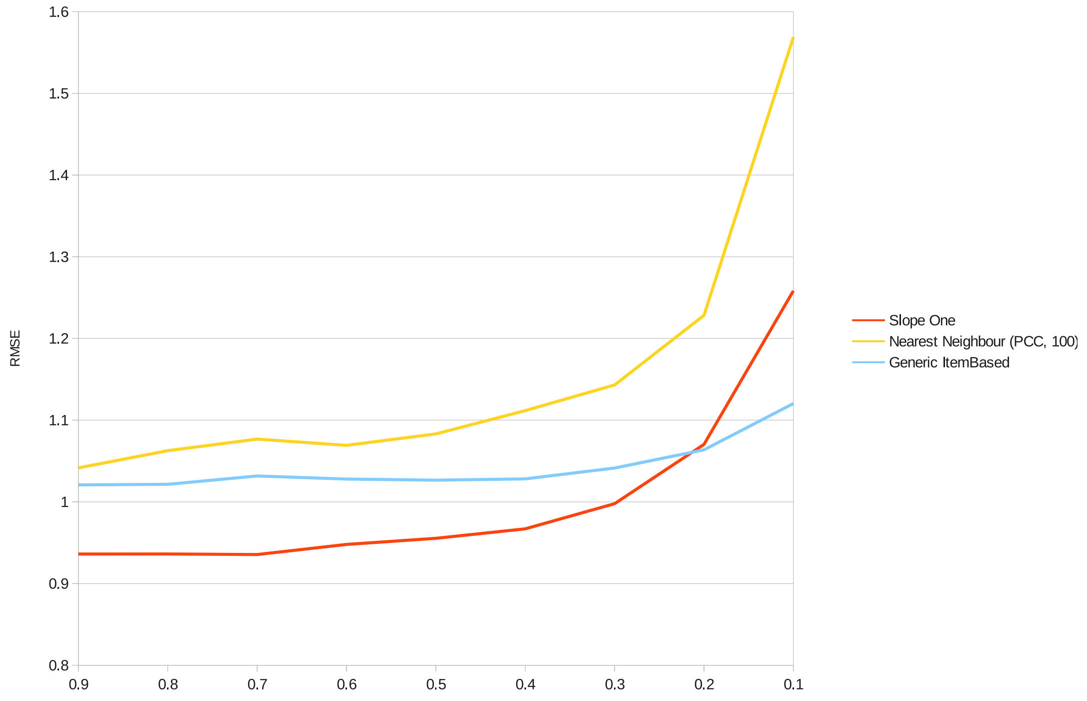


Figure 7.1: Movie Ratings RMSE

7.3.2 Fibronectin

In 7.2 the results on the Fibronectin set can be found. Interestingly results where much more flaky than for movie ratings. This may be caused by a more complex underlying model or simply the existence of certain outliers which affect the prediction. While the n-nearest neighbour approach with the pearson similarity coefficient still did not give comparable results, the item-based approach returned similar results to slope one. This is a sign that n-nearest neighbour with pcc seems to overfit the data. It should be kept in mind that this a user based method. In other terms, it may be argued that sophisticated definition of user preferences lead to bad results.

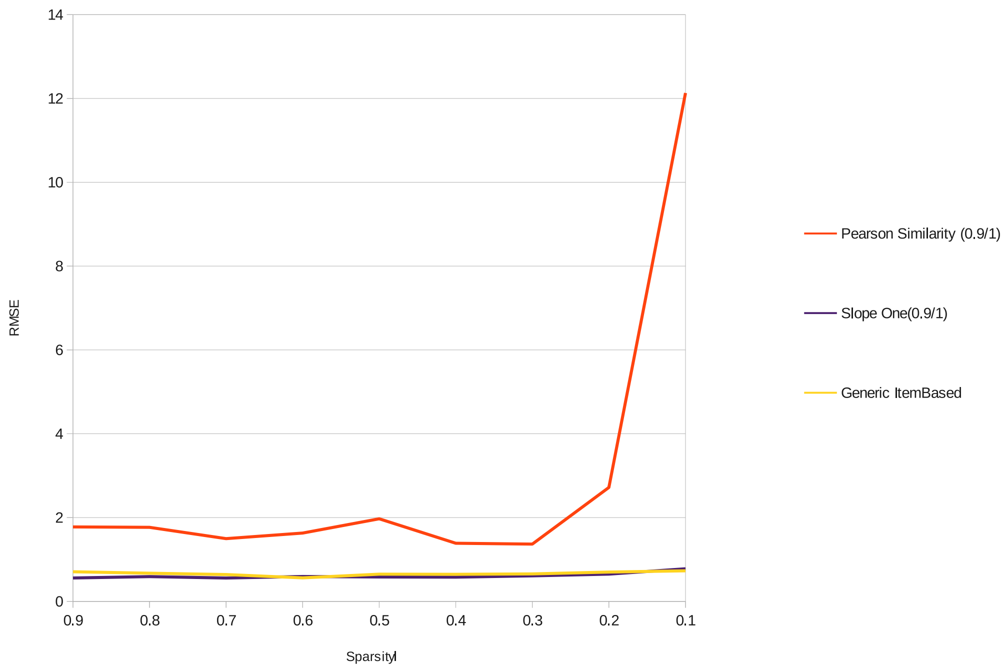


Figure 7.2: Fibronectin RMSE

7.3.3 A Note on Overfitting

Overfitting is a often used term in machine learning denoting the problem that outliers or certain instances may get overrated leading to bad performance of predictors. It is also related to clustering when too many clusters are found.

The simpler a model is and the more linearly it fits the given data, the smaller the distances between data points will be as there is a tendency that items become very similar to each other. This may yield good estimates on the RMSE, but it makes clustering afterwards much more difficult as similar instances are much more difficult to distinguish from each other.

In these terms the RMSE might not be such a good indicator on the actual validity of results. This problem gains also importance as especially for movie ratings where users have to rate in whole numbers. Predictions are calculated in decimals and there will always be a natural bias caused by rating constraints.

8
Clustering Results

In this last chapter, results are presented. In the previous chapters, various method and approaches have been discussed. For each dataset, data has been preprocessed by building graphs, using svd and collaborative filtering. This data always derives from the same initial data. It has been processed using the described algorithms. As previously stated the goal was to find natural clusters (6.3), similar behaviour among clustering algorithms and the impact of preprocessing methods on the data.

Unfortunately not all algorithms could be evaluated against each other as the used implementations did not always allow this and there is a general question about the representativeness of cluster solution comparisons due to many unknown ascendancies and assumptions.

Some might wonder why unlabeled data has been used when evaluating on labeled data would have given a more direct measure on the performance of each algorithm. One reason is the difficulty to find a high-dimensional sparse dataset. Artificially created sparsity is unfortunately not a real world problem. Clustering on a synthetic dataset would have been a possible option but also one that might not reflect the actual difficulty of the problem. Also finding a high-dimensional synthetic dataset with an exact solution is not an easy task. So instead random assigned clusters have been used to compare the performance of the algorithms.

Another problem with high-dimensional data sets is the difficulty to visualize the data or results. With many points there simply is no way data can be presented in a manageable manner.

8.1 Movie Ratings

8.1.1 Input Data

In 8.1 the similarities between the distance matrices computed from different input data sets can be found. As previously stated with sparse data sets, the general problem is to find a graph representation of the problem as many algorithms take a distance matrix as input. Again depending on the preprocessing of the data, the assumption is that similarities between distance matrices can be taken as evidence that the distance matrix is representative for the actual problem.

Noticeable points are:

- Although different approaches, distance matrices based on imputed Data with Simon Funk's approach and slope one recommendations yielded the most similar distance matrices with a correlation over 50% while their correlation to the distance matrix based on the original sparse data is much lower

Input Data Sets	Jaccard	Negative Square
Original Data Set & Simon SVD	0.25	0.32
Original Data Set & Mahout Recommendations	0.36	0.40
Simon SVD & Mahout Recommendations	0.54	0.57
Original Data Set & SVD	0.015	0.21
Simon SVD & SVD	0.01	0.12
Mahout Recommendations & SVD	0.01	0.08
Original Data Set & Full SVD	0.48	0.25
Full SVD & Simon SVD	0.16	0.12
Full SVD & Mahout Recommendations	0.33	0.06

Table 8.1: Mantel test of movie rating distance matrices

- The distance matrix based on slope one recommendations is more similar to the one based on the original data set than the one using Simon Funk's model
- When imputing data with svd, the resulting matrix is almost 50% similar to the one based on the original set, yet it is further away from the collaborative and content-based filtered ones
- Using the dimensionality reduced data set yields completely different results with almost zero correlation to all other distance matrices. It is assumable that results seem to differ, but the patterns in the distance matrix seem to go in another direction than the collaborative and content filtering approaches.

For the above svd calculations, 120 singular vectors have been used. It may be possible that using more singular vectors may improve results. It is to note the impact of dimensionality reduction was huge on the resulting distance matrix.
For reference it should be noted that the mantel test found only an average similarity of 0.18 between the negative squared distance matrix and the jaccard distance matrix on the same data.

8.1.2 Number of Clusters

The main evaluation is found in 8.1.3. There were significant differences between the algorithms in terms of runtime and number of clusters found. The runtime has not been examined in detail, but the number of clusters found is worth for a short analysis.

Depending on the input data and the algorithm, the number of clusters found did vary. In 8.2 only algorithms are listed that do not need the desired number of clusters as an input. Interestingly only Affinity Propagation (5.5.2) did find an interesting number of clusters. Of course this is no good quality measure, but for such a large dataset lower numbers of found clusters aren't very feasible as well.

Hence in further evaluation, algorithms that did not give many clusters have been excluded. Also for sake of comparability, cluster sizes have been equalled. So it is possible that in some cases indexes and clustering may be optimized by selecting a different number of clusters.

Algorithm	Min #	Max #	Notes
Affinity Propagation	2	141	Using the slope one recommendations yielded most clusters, using the negative squared distance matrix gave slightly better intra-cluster similarities than using jaccard.
DBSCAN	3	15	Using the jaccard distance matrix yielded best results, yet far away from enough clusters. To note is also that the clusters were unequally distributed in terms of their size.
Biclustering	1	3	In all testing cases, Biclustering did not result in many clusters. Most clusters were again found with the slope one recommendations
MCL	1	8	MCL found most clusters when using imputed data with Simon Funk's model and did find less when transforming the data in a distance matrix first. To note is that the found clusters were unequally distributed in terms of their size.

Table 8.2: Movie rating cluster numbers

Index	Random	Spectral	HClust	KMeans
Pearson Gamma	0.0006	0.045	0.135	0.066
Entropy	4.905	4.766	4.633	4.766
Intra-cluster distance	0.51	0.346	0.110	0.268
Inter-cluster distance	0.512	0.514	0.518	0.515
Intra-cluster sum of squares	302.21	110.48	51.40	86.67

Table 8.3: Clustering results on original dataset

8.1.3 Evaluation

The possibilities of comparing clustering solutions is never ending. There are two main approaches that have been used in this thesis. First comparisons of the algorithms comparing them to randomly assigned clusters has been made. The random cluster statistics serve as a base measure and can be found in 8.3, as input data the jaccard distance matrix on the original data set has been used. It can be seen that while the entropy and the inter-cluster distance do not significantly change, hierarchical clustering clearly is the most distinguished from random clusters on the given data.

Yet it is not clear whether the input data is meaningful at all. It is possible that slight changes on the input data have severe consequences on the data. So similar to finding similarities between the input matrices, the goal was to find similarities between clustering results with different input data and different algorithms. Ideally the algorithms should provide again similar results as strongly differing results may imply a certain kind of randomness as the input data is not entirely different.

- Of all clustering algorithms, spectral clustering (using a radial basis function kernel) seemed to be most affected by varying the input distance matrices. The average cluster similarity (rand index 6.1) with the different distance matrices was only 0.053. Of course this also depends on the chosen kernel. Compared to hierarchical clustering these results are signif-

Index	Random	Spectral	HClust	KMeans
Pearson Gamma	0.003	0.099	0.147	0.12
Entropy	4.905	4.88526	4.555	4.82
Intra-cluster distance	0.264	0.092	0.067	0.063
Inter-cluster distance	0.264	0.265	0.260	0.266
Intra-cluster sum of squares	71.305	9.011	5.613	6.202

Table 8.4: Clustering results on imputed dataset using Simon Funk's model

icantly worse. Additionally the clustering results between different input data as imputed data by collaborative and content based filtering did have little similarities to the original data with a rand index of 0.057.

- The average similarity between clusters with hierarchical clustering was 0.28 using the same input data. So in these terms it is a little bit more stable and reflects the differences between the distance matrices found in 8.1 more.
- Imputed data using SVD and dimensionality reduced input data with 120 dimensions as examined above led to completely different results in all cases with similarities ranging from 0.03 to 0.13.
- An interesting point was found when comparing the kernel based results with the hierarchical clustering based results. A similarity between the results of 0.31 was found when using Simon Funk's predicted values and a jaccard distance matrix. In all other cases results were less similar. A possible interpretation might be that as with Simon Funk's approach a clear model is imputed leaving less things uncertain while other approaches leave more space for interpretation leading to more differences between results.

The improvements on clustering with content based filtering filled data are described in 8.4.

So it is to say that the hierarchical clustering using Ward's method did give the best results. Additionally, just by using the original dataset without imputing missing values, clustering results weren't completely random but there was a certain randomness in it (e.g. when looking at the entropy).

The problem is that all these indexes and values depend on the input data set. They do show how well an algorithm solved the mathematical problem. They cannot show whether the input data was representative for the problem or not. The rand index between the hierarchical clustering result based on the original set and the one based on imputed data varies around 0.025. Interestingly clustering results based on the collaborative and content-based filtering data led to a rand index of 0.113. So in these terms there is a slight evidence data imputation leads to better results in terms of the similarity.

8.2 Fibronectin

8.2.1 Sparsity

As for Fibronectin, the sparsity level has also been taken into account. The reference number of clusters were again taken from the results found by affinity propagation as again MCL, Bi-

Density Level	Number of Clusters found	Net Similarity
1	66	558
0.9	66	559
0.8	64	531
0.7	61	473
0.6	56	420
0.5	50	404
0.4	50	403
0.3	30	168
0.2	36	133
0.1	62	250

Table 8.5: Impact of sparsity on affinity propagation results

cluster and DBSCAN did not find many clusters. In 8.5 the behaviour of affinity propagation with increasing sparsity of the input can be found. The net similarity is the function that affinity propagation tries to maximize. The net similarity decreases constantly with increasing sparsity. Increasing the sparsity can be imagined as reducing the number of edges between points. So with increasing sparsity (randomly removed values), the length of certain edges tends to infinity and as a consequence the net similarity decreases. Interesting is that between a density of 0.4 and 0.3 the net similarity drops significantly. A similar effect has been observed using other distance matrices. There is a tendency that from density levels from around 0.4 and below matrices start to differ much from each other, possibly due the increasing effect of assumptions made by the specific distance measure. This effect was drastic when comparing euclidean distances with jaccard distances. While nearly being on par until a density of 0.4 with a similarity of above 0.9, it dropped immediately to 0.7 and 0.5 with missing values above 0.5.

Filling Missing Values

With increasing sparsity, the need for imputing missing values also increases. Building a model for imputing data would be a difficult if not impossible task. The problem with collaborative filtering is that it usually needs many users (or dimensions) to be able to make predictions. So this is a shortcoming for this dataset as it only contains 28 attributes. There is however the option to invert the matrix so that similar proteins are searched to predict the expected expression for each attribute. For clustering, the resulting set may be inverted again. In 8.6 the similarities between the distance matrices using imputed values with slope one can be found. To ensure comparability all values have been scaled and the original values are kept with only imputing the missing values. Again with an increasing number of missing values the performance of slope one usually decreases. But it seems that slope one is a possible approach on imputing missing values even for protein data or when content based filtering is not a feasible. Interestingly just using 30% and imputing the rest yielded results that were closer to the original data set than using 60% of the data.

Evaluation

In 8.7 the impact of sparsity on hierarchical clustering can be seen. Again hierarchical clustering was flexible in terms of numbers of clusters found. The jaccard distance measure has been

Density	Imputed, 0.9	Original dataset
0.9	1	0.964
0.6	0.991	0.957
0.3	0.974	0.970
0.3	0.950	0.908

Table 8.6: Mantel test on jaccard distance using imputed values with slope one

Density Level	Inter-cluster distance	Intra-cluster	Intra-cluster sum of squares	Entropy	Rand index to result on dense data
1	0.429	0.139	6.193	4.0185	1
0.9	0.429	0.139	6.193	4.018	0.479
0.8	0.43	0.141	6.531	3.9752	0.411
0.7	0.429	0.146	6.957	4.036	0.358
0.6	0.429	0.151	7.471	4.013	0.286
0.5	0.429	0.159	8.365	4.073	0.23
0.4	0.429	0.174	10.951	4.044	0.161
0.3	0.427	0.204	15.152	4.125	0.087
0.2	0.427	0.253	21.346	4.133	0.0407
0.1	0.426	0.287	29.06	4.095	0.023

Table 8.7: Impact of sparsity on hierarchical clustering

used and DBSCAN, MCL and Biclustering did not give good results (e.g. only one cluster found). Among all others, hierarchical clustering did provide the most stable results while spectral clustering gave relatively random results where even a small density decrease led to entirely different clustering results. Unfortunately affinity propagation and hierarchical could not be evaluated directly. However when examining the exemplars selected by affinity propagation, hierarchical clustering had them also always placed in different clusters which is a good sign.

Especially interesting is that again the entropy and the inter-cluster distance do not change much. To note is that all measurements are taken from the full matrix (the distance matrix from the complete set). From a density level of 0.4, clustering results do not have much in common with the probably better results based on more information. Above the similarity is more significant.

8.3 Execution Workflows

The execution workflows were especially interesting from the point of view that the assumption was that there should be around 7 clusters as the data is highly structured which can be seen in 3.2. A negative squared distance matrix has been used and impossible distances between workflows have been set to negative infinity. Additionally a sparse graph also has been used.

Interestingly MCL found exactly 7 clusters for both the sparse and the dense graph, and this extremely fast. BiClustering only found 1 cluster. Affinity propagation did find 11 clusters, when reducing dimensions it found 10 clusters but with a higher net similarity.

9 Limitations

The list of limitations concerning the results presented in this thesis is endless. First of all the usual limitations apply. Calculations are often heuristic, many assumptions and decisions concerning input parameters on the algorithms have been made. Evaluation is difficult on unlabeled data and indexes may tell statistics on how well an algorithm solved the mathematical problem but not whether the result is meaningful at all.

It is to say that the approach in this thesis may be naive. With clustering it one usually has to carefully examine the initial problem and depending on the problem should choose the suitable methods. From this point of view the approach was very naive as many different methods have been mixed together and compared against each other. Hence in a certain way it could have been obvious that results vary greatly.

There is also a chance that this thesis underestimates the power of the algorithms as they were designed for other problems, different kind of input data or that they have not been tweaked to the maximum. So there is a certain amount of randomness in the results caused by imputation of data, transformation of data, clustering the data and evaluating the data on different aspects.

Hence the results found in this thesis should be regarded with caution.

10
Future Work

This work only contains a small subset of possible approaches and algorithms on the problem. There are several different ways to further explore the problem. Clustering high-dimensional data is actually a tripartite problem.

The first problem is to examine why data is missing at all. While it is obvious for user based content as movie ratings, it may not be as clear in other cases. It is often recommended to remove instances with many missing values and to remove outliers. Yet with sparse data it is difficult to say whether an instance is an outlier or not. Removing outliers may make the task of clustering easier, but also greatly reduces information on the data which may lead to a too strong generalization. The same applies to SVD. So there is much space on examining different approaches on how input data should be handled.

Several more algorithms could be analysed to compare their results against each other. The algorithms tested here were not specifically suited to high-dimensional data as for example OptiGrid [Hinneburg and Keim, 1999] or PreDeCon [Boehm et al., 2004]. It would be interesting to compare their findings to more traditional clustering approaches. Alternatively as often proposed is to divide the data in advance in small subsets and then cluster these subsets and to evaluate whether such results differ much from others.

Last but not least, especially in terms of movies it is difficult whether found clusters are meaningful or not. Data where cluster associations of instances are known in advance may be analysed to get a more direct validation measure on cluster results as visualizing high-dimensional data is difficult.

11 Conclusions

Clustering High-Dimensional sparse data is a difficult task. Given the data it is impossible to envisage the structure or visualize it. Often clustering algorithms are evaluated in a two-dimensional space where the solution can be evaluated graphically. Yet this is not feasible for high-dimensional data as it would not reflect the actual problem adequate. The next problem occurs when handling the sparsity. There are algorithms that might handle sparse data but results then depend on how much the sparse data reflects the actual problem behind the data. Last but not least an algorithm has to be selected to suit the data. Algorithms usually have a preference concerning input data and might work better on one or another distance metric. Especially with spectral clustering the used kernel has to be selected carefully.

Preprocessing the Input Data

As sparse matrices may lead to sparse graphs that do not represent the actual problem, filling or imputing the missing values is an important task. Especially due to the fact that from density levels of the matrix of 40% and below clustering results became entirely different as with dense input data. Of course removing rows with missing values is an option if this is a possible option. Then for imputing data building a predictor probably gives the most distinguished predictions as a model is implied. With movie ratings predicted by Simon Funk's model all algorithms (even the ones that did not perform well) did find significantly more clusters than on the original data. Using a recommender system to predict the values gave similar results yet not as distinguished as with Simon Funk's model. Using SVD to reduce dimensionality or to impute missing values led to results that were entirely different than the other approaches. So this approach might be better suited with low sparsity levels or when the data is highly representative for the whole matrix.

As probably assumed filling missing values with row or column averages or zeroes does not give useful data. However when using the jaccard distance measure zeroes do at least not distort the resulting graph.

Cluster Algorithms

Often algorithms are evaluated on synthetic data and may perform slightly better on the given data than the other algorithm. This advantage does not have a big impact on sparse data as for movie ratings the data has a natural bias as a user can only choose between whole numbers and a bias from imputing the values, namely how much the the predicted values differ from the rating the user might have given. So each instance is not actually an exact point but rather has a range in each dimension where the actual value might be. When trying different algorithms it seemed that some either made hard constraints and therefore had to find clusters as the number of cluster is given as an input parameter or they had to try finding clusters without any further input or a minimal distance or minimal cluster size. As high-dimensional data tends to be diffuse and usually high distances may occur quickly, algorithms that did not take the cluster number as an input usually did not find many cluster. This is probably given due to the fact that clusters have different densities and are therefore more difficult to locate in high-dimensional data.

An exception was affinity propagation which performed well and did find similar results than other algorithms. It would be interesting to see how it performs on labeled data compared to hierarchical clustering. Hierarchical clustering seemed also proved very versatile on all datasets and performed better than k-means and spectral clustering.

Although each algorithm probably has its advantages depending on the data and tweaking the parameters, affinity propagation and hierarchical clustering seemed to be versatile.

Evaluation

Evaluation is a difficult task, especially given the non visualizable high-dimensionality. Usually indices as the intra-cluster distance can only reflect how well the algorithm has solved the mathematical problem, but even then the entropy seemed not to be a good measure as it did not vary much compared to random clusters. Hence the approach was to find similarities between different cluster solutions using the rand index. A similar approach has been used in the netflix prize, namely blending the results using various different approaches. This may be used for clustering algorithms too as long as results are not entirely different.

A
Appendix

A.1 R and its Packages

Most work has been done using R [Team, 2011] and its many packages aimed at clustering. For further reference, the following packages have been used:

- Affinity Propagation of the ”apcluster” package [Bodenhofer et al., 2011]
- Hierarchical clustering and k-means are part of R [Team, 2011]
- A variety of spectral clustering algorithms and kernels including the one used in this thesis can be found in the ”kernlab” package [Karatzoglou et al., 2004]
- Distance matrices have been calculated using the ”vegan” package [Oksanen et al., 2011]
- DBScan, cluster statistics and comparisons have been done using the ”fpc” package [Hennig, 2010]
- The mantel test can be found in the ”ade4” package [Dray and Dufour, 2007]
- Partial singular value decomposition with lanczos bidiagonalization has been calculated using the ”irlba” package [Baglama and Reichel, 2011]
- The Biclustering package can be found in the ”biclust” package [Kaiser et al., 2011]
- Non-metric multidimensional scaling has been done using the ”MASS” package [Venables and Ripley, 2002]

A.2 Mahout - Recommender Systems

Apache Mahout [1] is is machine learning library. It consists of various algorithms. In this thesis the slope one, the k-nearest-neighbour and the pearson similarity implementations have been used. An extensive introduction for Mahout can be found in [Owen and Anil, 2010].

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

A.3 Other Implementations

An implementation of MCL can be found on the inventor's [van Dongen, 2000] website [2]. The content-based filtering model is an own implementation based on Simon Funk's approach.

A.4 Sample R Code

For reference, below a list of calls for algorithms and approaches used in this thesis based on the movie ratings set.

```
# load dataset
Ratings <-
  read.csv("/pathToCSVFile/movie_ratings.csv",
   header=TRUE, sep="\t", dec=".", na.strings="Na", fill = TRUE)

# load sparse matrix
library(Matrix)
sparseMatrix <- scan("/pathToSparseMatrixFile/sparseMatrixRatings.txt",
what=list(integer(), integer(), integer()))
sparseMatrix <- sparseMatrix(i=classHelper[[1]], j=classHelper[[2]],
x=classHelper[[3]])

# SVD and full representation
library(irlba)
svd <- irlba(sparseMatrix, nu=120, nv=120)
plot(svded$d, main = "Largest Singular values of Execution plans")
svdFullMatrixRepresentation <- u%*%d%*%t(v)

# create distance matrix
library(vegan)
library(apcluster)
JaccardDistance <- vegdist(Ratings, method="jaccard",
binary=FALSE, diag=FALSE, upper=FALSE, na.rm = TRUE)

NegativeSquaredDistance <- negDistMat(Ratings, r = 2)

# mantel test for distance matrices
library(ade4)
scaledJaccardDistance <- scale(JaccardDistance)
scaledNegativeSquaredDistance <- scale(NegativeSquaredDistance)
mt <- mantel.randtest(scaledJaccardDistance,
dist(scaledNegativeSquaredDistance), nrepet=1000)

# clustering
library(apcluster)
library(kernlab)
```

[2] http://micans.org/mcl/

```r
library(stats)
library(fpc)
library(biclust)

spectral <- specc(NegativeSquaredDistance, centers=141, kernel = "
   rbfdot")
hierarchical <- hclust(JaccardDistance, method="ward")
hierarchical <- cutree(JaccardDistance, k= 141)
DBSCAN <- dbscan(JaccardDistance, 10, showplot = 0)
kmeans <- kmeans(JaccardDistance, 141)
affinityPropagation <- apcluster(NegativeSquaredDistance)
biclust <- biclust(as.matrix(Dataset), method=BCBimax(), alpha=0.05,
   number=50)

#evaluation
library(fpc)
statistics <- cluster.stats(JaccardDistance, spectral@.Data,
   kmeans@clusters, compareonly = FALSE)
```

List of Figures

List of Tables

Bibliography

[Abbas, 2008] Abbas, O. A. (2008). Comparisons between data clustering algorithms. *Int. Arab J. Inf. Technol.*, 5(3):320–325.

[Acock, 2005] Acock, A. C. (2005). Working with missing values. *Journal of Marriage and Family*, 67(4):1012–1028.

[Akama et al., 2007] Akama, H., Miyake, M., and Jung, J. (2007). How to take advantage of the limitations with markov clustering?

[Alzate et al., 2009] Alzate, C., Espinoza, M., Moor, B., and Suykens, J. A. (2009). Identifying customer profiles in power load time series using spectral clustering. In *Proceedings of the 19th International Conference on Artificial Neural Networks: Part II*, ICANN '09, pages 315–324, Berlin, Heidelberg. Springer-Verlag.

[Baglama and Reichel, 2011] Baglama, J. and Reichel, L. (2011). *irlba: Fast partial SVD by implicitly-restarted Lanczos bidiagonalization*. R package version 1.0.1.

[Becavin et al., 2011] Becavin, C., Tchitchek, N., Mintsa-Eya, C., Lesne, A., and Benecke, A. (2011). Improving the efficiency of multidimensional scaling in the analysis of high-dimensional data using singular value decomposition. *Bioinformatics*, 27(10):1413–1421.

[Bellman, 1961] Bellman, R. E. (1961). *Adaptive control processes - A guided tour*. Princeton University Press, Princeton, New Jersey, U.S.A.

[Bodenhofer et al., 2011] Bodenhofer, U., Kothmeier, A., and Hochreiter, S. (2011). Apcluster: an r package for affinity propagation clustering. *Bioinformatics*, 27:2463–2464.

[Boehm et al., 2004] Boehm, C., Kailing, K., Kriegel, H.-P., and Kroeger, P. (2004). Density connected clustering with local subspace preferences. *Data Mining, IEEE International Conference on*, 0:27–34.

[Cacheda et al., 2011] Cacheda, F., Carneiro, V., Fernández, D., and Formoso, V. (2011). Comparison of collaborative filtering algorithms: Limitations of current techniques and proposals for scalable, high-performance recommender systems. *ACM Trans. Web*, 5:2:1–2:33.

[Comi et al., 2003] Comi, A. M., Hunt, P., Vawter, M. P., Pardo, C. A., Becker, K. G., and Pevsner, J. (2003). Increased fibronectin expression in sturge-weber syndrome fibroblasts and brain tissue. *Pediatric Research*, 53(5):762–769.

[Dempster et al., 1977] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, B*, 39.

[Dhillon et al., 2004] Dhillon, I., Guan, Y., and Kulis, B. (2004). A unified view of kernel k-means, spectral clustering and graph cuts. Technical report.

[Dray and Dufour, 2007] Dray, S. and Dufour, A. (2007). The ade4 package: implementing the duality diagram for ecologists. *Journal of Statistical Software*, 22(4):1–20.

[Dunn, 1973] Dunn, J. C. (1973). A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, 3(3):32–57.

[Dunning, 1993] Dunning, T. (1993). Accurate methods for the statistics of surprise and coincidence. *COMPUTATIONAL LINGUISTICS*, 19(1):61–74.

[Dhaeseleer, 2005] Dhaeseleer, P. (2005). How does gene expression clustering work? *Nature Biotechnology*, 23(12):1499–1501.

[Eisen et al., 1998] Eisen, M. B., Spellman, P. T., Brown, P. O., and Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. *Proc Natl Acad Sci U S A*, 95(25):14863–14868.

[Frey and Dueck, 2007] Frey, B. J. and Dueck, D. (2007). Clustering by passing messages between data points. *Science*, 315:972–976.

[Halko et al., 2011] Halko, N., Martinsson, P. G., and Tropp, J. A. (2011). Finding structure with randomness: stochastic algorithms for constructing approximate matrix decompositions. *Techniques*, 53(2):1–81.

[Hennig, 2010] Hennig, C. (2010). *fpc: Flexible procedures for clustering*. R package version 2.0-3.

[Hinneburg and Keim, 1999] Hinneburg, A. and Keim, D. A. (1999). Optimal grid-clustering: Towards breaking the curse of dimensionality in high-dimensional clustering. In Atkinson, M. P., Orlowska, M. E., Valduriez, P., Zdonik, S. B., and Brodie, M. L., editors, *VLDB'99, Proceedings of 25th International Conference on Very Large Data Bases, September 7-10, 1999, Edinburgh, Scotland, UK*, pages 506–517. Morgan Kaufmann.

[Jaccard, 1901] Jaccard, P. (1901). Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin del la Société Vaudoise des Sciences Naturelles*, 37:547–579.

[Kaiser et al., 2011] Kaiser, S., Santamaria, R., Tatsiana, Khamiakova, Sill, M., Theron, R., Quintales, L., and Leisch., F. (2011). *biclust: BiCluster Algorithms*. R package version 1.0.1.

[Karatzoglou et al., 2004] Karatzoglou, A., Smola, A., Hornik, K., and Zeileis, A. (2004). kernlab – an S4 package for kernel methods in R. *Journal of Statistical Software*, 11(9):1–20.

[Koren, 2009] Koren, Y. (2009). The bellkor solution to the netflix grand prize. *Baseline*, (August):1–10.

[Kriegel et al., 2011a] Kriegel, H.-P., Kroeger, P., Ntoutsi, I., and Zimek, A. (2011a). Density based subspace clustering over dynamic data. In Cushing, J. B., French, J. C., and Bowers, S., editors, *SSDBM*, volume 6809 of *Lecture Notes in Computer Science*, pages 387–404. Springer.

[Kriegel et al., 2011b] Kriegel, H.-P., Kröger, P., Sander, J., and Zimek, A. (2011b). Density-based clustering. *Wiley Interdisc. Rew.: Data Mining and Knowledge Discovery*, 1(3):231–240.

[Kurucz et al., 2007] Kurucz, M., Benczúr, A. A., and Torma, B. (2007). Methods for large scale svd with missing values. In *KDDCup 2007*.

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

[Luxburg, 2007] Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and Computing*, 17:395–416.

[Manning et al., 2008] Manning, C. D., Raghavan, P., and Schtze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.

[Mantel, 1967] Mantel, N. (1967). The detection of disease clustering and a generalized regression approach. *Cancer Research*, 27(2 Part 1):209–220.

[McCallum et al., 2000] McCallum, A., Nigam, K., and Ungar, L. (2000). Efficient clustering of high-dimensional data sets with application to reference matching. In *KNOWLEDGE DISCOVERY AND DATA MINING*, pages 169–178.

[Meilă, 2007] Meilă, M. (2007). Comparing clusterings—an information based distance. *J. Multivar. Anal.*, 98:873–895.

[Mishra et al., 2011] Mishra, R., Shukla, S., Arora, D. D., and Kumar, M. (2011). An effective comparison of graph clustering algorithms via random graphs. *International Journal of Computer Applications*, 22(1):22–27. Published by Foundation of Computer Science.

[Ng et al., 2001] Ng, A. Y., Jordan, M. I., and Weiss, Y. (2001). On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*, pages 849–856. MIT Press.

[Oksanen et al., 2011] Oksanen, J., Blanchet, F. G., Kindt, R., Legendre, P., Minchin, P. R., O'Hara, R. B., Simpson, G. L., Solymos, P., Stevens, M. H. H., and Wagner, H. (2011). *vegan: Community Ecology Package*. R package version 2.0-1.

[Owen and Anil, 2010] Owen, S. and Anil, R. (2010). *Mahout in action (MEAP)*. Manning.

[Piotte and Chabbert, 2009] Piotte, M. and Chabbert, M. (2009). The pragmatic theory solution to the netflix grand prize, in: Netflix prize documentation.

[Prelić et al., 2006] Prelić, A., Bleuler, S., Zimmermann, P., Wille, A., Bühlmann, P., Gruissem, W., Hennig, L., Thiele, L., and Zitzler, E. (2006). Comparison of biclustering methods: A systematic comparison and evaluation of biclustering methods for gene expression data. TIK Report 227, Computer Engineering and Networks Laboratory (TIK), ETH Zurich.

[Rand, 1971] Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850.

[Ricci et al., 2011] Ricci, F., Rokach, L., and Shapira, B. (2011). Introduction to recommender systems handbook. In Ricci, F., Rokach, L., Shapira, B., and Kantor, P. B., editors, *Recommender Systems Handbook*, pages 1–35. Springer US.

[Rumelhart et al., 1986] Rumelhart, D., Hintont, G., and Williams, R. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.

[Santos and Embrechts, 2009] Santos, J. M. and Embrechts, M. (2009). On the use of the adjusted rand index as a metric for evaluating supervised classification. In *Proceedings of the 19th International Conference on Artificial Neural Networks: Part II*, ICANN '09, pages 175–184, Berlin, Heidelberg. Springer-Verlag.

[Steinbach et al., 2003] Steinbach, M., Ertz, L., and Kumar, V. (2003). The challenges of clustering high-dimensional data. In *In New Vistas in Statistical Physics: Applications in Econophysics, Bioinformatics, and Pattern Recognition*. Springer-Verlag.

[Team, 2011] Team, R. D. C. (2011). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

[Tikhonov, 1943] Tikhonov, A. (1943). On the stability of inverse problems. *Doklady Akademii nauk SSSR*, 39(5):195–198.

[van Dongen, 2000] van Dongen, S. (2000). *Graph Clustering by Flow Simulation*. PhD thesis, University of Utrecht.

[Venables and Ripley, 2002] Venables, W. N. and Ripley, B. D. (2002). *Modern Applied Statistics with S*. Springer, New York, fourth edition. ISBN 0-387-95457-0.

[Vlasblom and Wodak, 2009] Vlasblom, J. and Wodak, S. J. (2009). Markov clustering versus affinity propagation for the partitioning of protein interaction graphs. *BMC Bioinformatics*, 10(1):99.

[Ward, 1963] Ward, J. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58:236–244.

[Wayman, 2003] Wayman, J. C. (2003). Multiple imputation for missing data: What is it and how can i use it? Paper presented at the 2003 Annual Meeting of the American Educational Research Association, Chicago, IL.

[Weber et al., 1998] Weber, R., Schek, H.-J., and Blott, S. (1998). A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *Proceedings of the 24rd International Conference on Very Large Data Bases*, VLDB '98, pages 194–205, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Welling, 2009] Welling, M. (2009). Kernel k-means and spectral clustering.

[Whitworth, 2010] Whitworth, G. B. (2010). An introduction to microarray data analysis and visualization. *Methods in Enzymology*, 470(x):19–50.