Evaluation of a co-clustering of Facebook likes

A Radboud University's Bachelor's Thesis

June 7th 2014

H. M. Prins *prins_harmen@hotmail.com*
Student number: S4132297
Supervisors:
Prof. dr. T. Heskes *tomh@cs.ru.nl*
Dr. L. G. Vuurpijl *L.vuurpijl@ai.ru.nl*
T. Cruysen *tijs@media11.nl*

Contents

Abstract

Companies have gathered vast amounts of data about their customers. This paper examines the possibility of using this data to group customers to find novel patterns in the data, which the companies could use to their advantage. The data set to test this hypothesis comes from the social network Facebook, to be clustered using co-clustering which approximates the data using clusters. The algorithm did extract novel information, but it was not feasible to make this information explicit. Some ideas on how to improve the algorithm on this deficiency are given.

1. Introduction

Obtaining useful information from big sets of data has recently become increasingly popular for a number of reasons. Firstly, the data sets are getting bigger because the hardware gets increasingly better. Secondly, companies realize the use for data mining and store as much data as they can get. As a consequence many companies have a lot of data on servers, waiting to be analyzed.

What is more interesting about this data is that a lot of it is concerned with tracking users. This information can be used to predict what a user is going to do, and also alter the user's behavior by altering the presentation of a website, for example. As it is easy for companies to change websites based on who is watching it, marketing has entered the age of targeted marketing.

Social networks are websites that allow users to interact and share information with other users. This can be done in groups so that people with similar interests can share things they know other people will find interesting. Interaction is done by making posts, which can be done in groups or to their friends. In the case of Facebook, sharing information about oneself can be done by liking certain pages or posts. People tend to like a lot of different things to show their association with certain groups and maintain an individual image.

From a marketing perspective users lay bare their personalities on social networks, while also showing who they are similar to, and thus who they will behave like. Understanding these relations can give insight in how a user might respond to things. For example, if we know a certain user is part of a group, and that group is known to be responsive to a certain advertisement, chances are that user will also respond to that advertisement.

This paper will give an analysis of the possibility of grouping users based on what Facebook pages they like, and see if those groupings say something meaningful about the users and pages. If this is the case, clustering may well be a useful tool in selecting users for targeted marketing.

2. Previous research

2.1. Clustering

50 years since the first clustering algorithm, K-means, was made by Steinhaus, thousands of clustering algorithms have been made (Jain, 2010). K-means is still widely used, but it is not capable of solving every clustering problem. New clustering algorithms are made to solve the problems that older algorithms are not capable of solving, or to increase efficiency. This has led to the creation of thousands of clustering algorithms. Every one is suited for

a particular problem. For example, DBSCAN is very efficient in clustering density-based data.

Clustering algorithms are unsupervised learning algorithms that group data. They are called unsupervised because the groups and labels for those groups are not known beforehand. As a result these groups still have to be interpreted. In stead of grouping based on labels that are provided beforehand, clustering algorithms try to find patterns in the data and group it based on those.

Co-clustering methods cluster not just the objects in the data, but the features as well, to increase the performance, especially for sparse data sets. The two clusterings can improve each other, because they provide information about the data. Another advantage is the fact that the researcher can extract information from the feature clustering, as well as the correlations between object and feature clusters. The reason it is effective on sparse data, is because it implicitly reduces the dimensionality of the dataset (as will be explained later).

2.2. Collaborative Filtering

Collaborative filtering is a method of predicting unobserved features for an object based on other objects. First, the object is compared to other objects, to see how similar it is to every other object. Then for every unobserved feature the method tries to predict a value based on the observed features of the similar objects. For example, a user u_0 for a website that stores movie ratings has seen and rated movies m_0 through m_5. The user has not yet seen m_6, but users u_1, u_2 and u_3 have. To see how similar those users are to u_0, we compare the ratings. It turns out u_1 and u_2 have exactly the same ratings for the movies u_0 rated. To predict the rating u_0 will give for m_6, the ratings u_1 and u_2 gave are averaged.

The idea behind collaborative filtering is that the features are determined by some underlying pattern in the objects. In the movie example, u_0 might like action movies but dislike romantic movies, as do u_1 and u_2. Therefore they will generally like the same movies, and so, if m_6 is an action movie all three users will rate it similarly. With enough data, these underlying patterns can be revealed, and clusters can be based on them.

The technique has been used extensively in recommender systems (like the movie example above), which recommend new products to users who have already shown interest towards some products (or other information that can be used for prediction). These systems range from Facebook filtering the content you see to Amazon's "other users who bought this also bought..." section. Recommender systems have seen a rise in popularity since the rise of the internet, as the internet has provided access to the data needed to make effective predictions (Shaw and Welge, 2001). The systems efficiency is also thanks to website being able to automate the recommendation process.

Another use for collaborative filtering is finding good targets for directed marketing. Finding advertisements that are effective per individual increases response rate (Ling and Li, 1998), but may even leave the receiver of the advertisement appreciative rather than annoyed. A problem with the data, however, is that not many people respond to advertisements at all, and so there are only few features on which to base the predictions. And for users new to the system, there is nothing to work with at all! As a solution, instead of basing the similarity between users on what advertisements they reacted to, one could base

similarity on other features, shared interests or behaviors, so the model of the underlying pattern is based on second-order activities, rather than the activities we want to predict.

3. Data

To understand why the algorithm was chosen, some properties about the data set must be known. In this chapter, the different properties of the dataset will be highlighted. The dataset was provided by Media11 and concerns Facebook users' page likes.

Table 1: In the following sections the following notation will be used

$W = (w_{ij})_{n \times m}$	The dataset of n users and m pages
K	Number of user clusters
C	Number of page clusters
$P = \{P_1, P_2, \ldots, P_K\}$	Clustering of users
$Q = \{Q_1, Q_2, \ldots, Q_C\}$	Clustering of pages
$A = (a_{ik})_{n \times K}$	User cluster assignment
$B = (b_{jc})_{m \times C}$	Page cluster assignment
$X = (x_{kc})_{K \times C}$	Fraction of positive to all elements per cluster

The dataset is part of a larger dataset containing not just the information about what likes were recorded, but also answer to questionnaires, votes in contests and profile information. To be able to store so many relations efficiently and be able to add new tables of information to the dataset, the dataset is structured according to the many-to-many paradigm. This, combined with the fact that it was stored on a server, made it a little harder to obtain the data than just getting one table with the data. Some of the data seemed to be missing as well, as not all page IDs we found in the likes table were in the name table. This only became apparent after the algorithms had run because before that time the names were not relevant, so the proper name of every page could not always be established.

3.1. Format

Originally, the data is stored as a list of likes: every row contained one user ID and one page ID. For the purposes of clustering, this would have to be transformed to a sparse, binary matrix of user-page relations. These are binary relations because either someone likes something or he does not. Binary data can improve the speed of calculating the Mean Squared Error by dividing the dataset into two parts.

Since people tend to like very few things relative to the total number of things to like, (after processing, the average likes per user is 55, the total number of pages to like 97109), the data is called sparse, with a sparsity of 99.94%. This sparsity can be used to greatly increase the speed of the algorithm compared to similarly-sized data sets. (Jones et al., 2001–) implemented different kinds of sparse matrices. The drawback of these matrices is that element access has complexity $O(nd)$ where d is the number of elements per row. In return for that drawback, the matrix has an iterator that allows access to all nonzero elements, and since in our case $|W|_1 \ll nm$, this is a tremendous advantage.

3.2. Curse of Dimensionality

Another consequence of the large possible number of pages to like is the high dimensionality of the data. One of the problems with the nature of dimensionality that the concept of distance, the most common way of measuring similarity for clustering, becomes less meaningful. Therefore similarity measures like the cosine similarity should be used, or alternative ways of clustering should be used (Ungar and Foster, 1998). In this case, the choice for an alternative way of clustering, which implicitly reduces the dimensionality, was made.

3.3. Names

The dataset also contains the names of the pages, which means that some semantic value can be given to the clusterings, as some pages will turn out to be more similar, and it is possible to confirm or reject one's intuitions about such pages. For example, it should be possible to confirm if people who like fancy restaurants prefer wine over beer, whereas people who go to snack bars prefer beer.

3.4. Pre-processing

Out of the 291 501 pages, 194 392 had 1 like. These singularly liked pages do not contain any information, as it does not link any users. Out of the 14 975 users, 711 had three or fewer likes. Those pages and users were discarded. The pre-processing could have been more aggressive to decrease sparsity, but the choice was made to include as much data as possible to be able to get as much information about every aspect of the data as possible.

4. Algorithm

4.1. Co-clustering

Co-clustering is a method of approximating a matrix by dividing the rows and columns into clusters such that every row is in one vertical cluster and every column in one horizontal cluster. As a consequence, every data point is in one vertical and one horizontal cluster. Such a combination of clusters is called a co-cluster. The original matrix is approximated based on those co-clusters and a certain summary statistic about every co-cluster. For example, for real-valued data this summary statistic would be the average value of the co-cluster and the recreation-process would assign the average value to all data points in the co-cluster. For binary data the obvious summary statistic is whether most values are 1 or 0, but a more useful one might be the ratio of ones to zeros.

A visual example is given in Figure 1.

Because co-clustering uses two clusterings, one for every dimension, it combats the curse of dimensionality. The way this is done is by implicitly reducing the dimensionality of the dataset. As the formula for object cluster assignment only looks at feature clusters and not every feature, the dimensionality is reduced from the number of features to the number of feature clusters. The same goes for feature cluster assignment.

	Cars	Formula 1	Amsterdam	Ajax
Jane	y	y		
Peter	y	y		
Fred			y	y
Bart			y	

(*a*) An example of people liking pages

1	1	0	0
1	1	0	0
0	0	1	1
0	0	1	0

(*b*) Converted to a matrix

1	0
0	$\frac{3}{4}$

(*c*) A summary matrix that contains the average of every co-cluster

1	1	0	0
1	1	0	0
0	0	$\frac{3}{4}$	$\frac{3}{4}$
0	0	$\frac{3}{4}$	$\frac{3}{4}$

(*d*) Reconstructed matrix, based on the co-clusters and averages

$4 \times (1-1)^2 + 8 \times (0-0)^2 + (3 \times (1-\frac{3}{4})^2 + 1 \times (0-\frac{3}{4})^2) = 0.75$

(*e*) Calculating SSE

Figure 1: An example of the calculation of the sum of squared errors of a data set

Co-clustering for collaborative filtering uses the clusters to find which objects are similar and also which aspects are often grouped together. More importantly, if the ratio summary statistic is used for a binary dataset, the certainty of a relation (or lack of a relation) can be inferred from the summary statistic.

4.2. Description

The algorithm is based on the algorithm by (Li, 2005), and similar to the algorithm of (George and Merugu, 2005). Both these algorithms are co-clustering algorithms, clustering both objects and features. Each clustering can improve the other clustering. George's algorithm is designed for continuous data, whereas Li's algorithm is designed for binary data. The problem with George's algorithm in this case, is the fact that it tries to predict the value for all unknown relations based on the average of the known relations in a relation's (co)clusters. The average of a cluster with just ones and zeros is one, since the zeros are considered unknown by the algorithm. Therefore, one viable solution to any approximation is: all objects and features are in the same co-cluster, and all relations are one.
The complexity of George's clustering algorithm is linear in the number of non-zeros, the number of objects times the number of co-clusters, and the number of pages times the number of co-clusters, i.e.

$$O(|W|_1 + nKC + mKC) \tag{1}$$

Compare to the benchmarking results in Figure 2, which shows that the algorithm scales linearly for every variable displayed. To show that the complexity scales linearly with the products (i.e. nKC and mKC), we showed that for a constant product with different multiplicands the time does not change in Table 2.

Ignoring the structure of the data, George's algorithm is very similar to Li's, as they both create an approximation matrix based on a summary statistic and co-clusterings. Both approximation matrices try to minimize the sum of squared error.

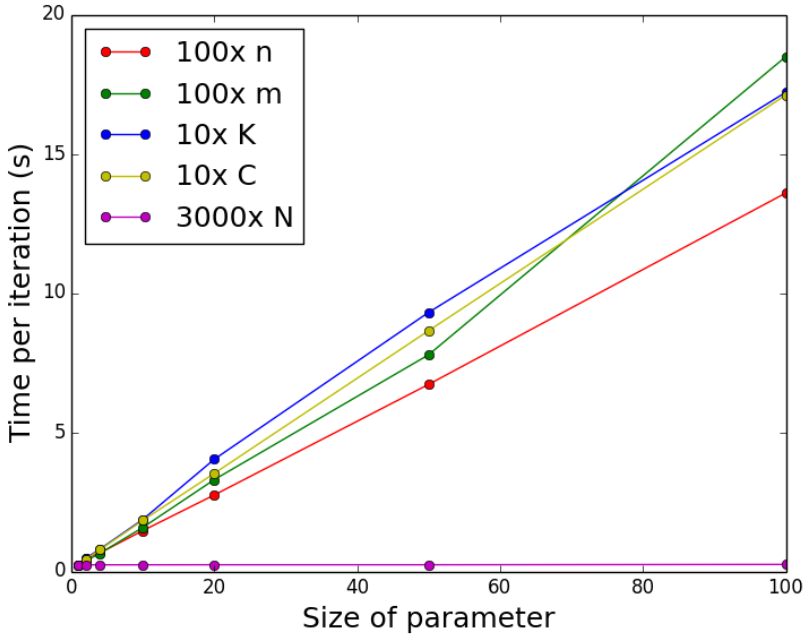

Figure 2: Experimental confirmation of the algorithm's linear time complexity for number of objects (n), features (m), object clusters (K), feature clusters (C) and likes (N)

Table 2: Given that the product of the three variables is constant, the time also does not change much

n	K	C	time	m	K	C	time
5000	200	10	102.67	5000	200	10	130.46
2500	400	10	103.52	2500	400	10	130.79
1250	800	10	110.44	1250	800	10	130.69
2500	200	20	111.99	2500	200	20	106.65
1000	100	100	117.41	1000	100	100	119.39
1250	10	800	111.28	1250	10	800	104.36
2500	10	400	114.01	2500	10	400	106.54
5000	10	200	105.82	5000	10	200	136.82

Li's algorithm also shares some similarities with Ungar's algorithm, which is also designed for binary data. It used fuzzy K-means clustering to assign objects to clusters, then do the same for features, and then repeat based on those clusters. It did not perform well on real data, Ungar et al. instead continue to work with another algorithm. Fuzzy clustering may be applied to Li's algorithm as well, possibly improving performance.

Ungar's algorithm is based on Expectation Maximization (EM), as are George's and Li's algorithms. EM repeats two steps: it calculates an approximation to the original data set, and then checks if changing the model on which the approximation is based improves this approximation. The way this is done in George's and Li's algorithm is by changing the cluster assignment of every object and feature iteratively.

The goal of Li's algorithm is to minimize the Sum of Squared Errors

$$SSE = \sum_{i,j} (W_{ij} - \hat{W}_{ij})^2 \tag{2}$$

with

$$\hat{W} = AXB^T \tag{3}$$

A and B are designed so that they 'select' elements from X corresponding the the relevant co-cluster. For example, if the data point at $(0, 0)$ belongs to user cluster 5 and page cluster 1, A selects row 5 and B selects column 1 from X. This way we can reconstruct the matrix by filling in the average value of each co-cluster for all it's data points.

Li's algorithm iteratively calculates a W such that the Sum of Squared Errors reaches a local minimum, by following the steps as described in Algorithm 1

Algorithm 1: Li's algorithm

1. Initialize A, B randomly
2. Calculate X based on Equation (9)
3. Repeat until convergence:
   (a) Calculate A based on Equation (4)
   (b) Calculate B based on Equation (5)
   (c) Calculate X based on Equation (6)

To calculate A, every element is calculated according to

$$\hat{a}_{ik} = \begin{cases} 1 & \text{if } \sum_{c=1}^{C} \sum_{j \in Q_c} (w_{ij} - x_{kc})^2 < \sum_{c=1}^{C} \sum_{j \in Q_c} (w_{ij} - x_{lc})^2 \\ & \text{for } l = 1 \dots K, l \neq k \\ 0 & \text{Otherwise} \end{cases} . \tag{4}$$

To calculate B, every element is calculated according to

$$\hat{b}_{jc} = \begin{cases} 1 & \text{if } \sum_{k=1}^{K} \sum_{i \in P_k} (w_{ij} - x_{kc})^2 < \sum_{k=1}^{K} \sum_{i \in P_k} (w_{ij} - x_{kl})^2 \\ & \text{for } l = 1 \dots C, l \neq c \\ 0 & \text{Otherwise} \end{cases} . \tag{5}$$

X can be seen as three different things: a centroid matrix, where every row is the centroid for an object cluster and every column for a feature cluster, an association matrix, where every data point is the association between an object and feature cluster, and a summary

matrix, which summarizes the values in each co-cluster. The first perception shows the similarity between K-means and this algorithm, as Li describes in their paper. The second perception can be used as a way to interpret the clusters, as objects and feature in highly associated clusters are likely associated as well. The third perception is actually the way X is calculated:

$$X_{k,c} = \frac{1}{\text{size}(P_k)\text{size}(Q_c)} \sum_{i \in P_k} \sum_{j \in Q_c} W_{ij} \tag{6}$$

where size(cluster) gives the number of objects and features in an object cluster and feature cluster respectively.

The next section will describe how the algorithm was implemented.

4.3. Implementation

Li did not give an implementation, so in this section the implementation will be provided.

In this section, the implementation of the algorithm is discussed. A few improvements to lower the time complexity of the algorithm are explained, and the complexity of George's algorithm is reached.

As explained earlier, A and B 'select' elements from X by multiplying matrix multiplication with vectors with one 1 and 0's otherwise. This can be done much more efficiently than using matrix multiplication, namely by actually storing which row and column A and B select, respectively.

$$X_A = (X_{a_1}, X_{a_2} \dots X_{a_n}) \text{ with } A = \{a_1, a_2 \dots a_n\} \tag{7}$$

This redefinition of A changes the calculation of its elements to:

$$a_i = \arg\max_x \sum_{c=1}^{C} \sum_{j \in Q_c} (w_{ij} - x_{kj})^2 \tag{8}$$

The following optimizations are possible because of the sparse and binary nature of the dataset. X contains the ratio of positive elements to all elements in every co-cluster:

$$X_{kc} = \frac{\sum_{ij} I(A_i = k) I(B_j = c) W_{ij}}{\sum_{ij} I(A_i = k) I(B_j = c)}, \tag{9}$$

where $I(\text{statement})$ returns 1 if the statement is true and 0 otherwise. This equation takes the number of positive elements in W from co-cluster k, c and divides it by the size of k, c. Because the data is saved in a sparse format, counting the positive elements in all clusters has complexity $O(|W|_1)$. Because the size of each co-cluster is the product of the sizes of its two clusters, the time complexity of calculating co-cluster sizes from cluster sizes is $O(KC)$. The complexity of calculating the size of each cluster is $O(n + m)$.

To decide which cluster every row (and column) should be assigned to, the algorithm calculates the sum of squared errors for every possible assignment, according to

$$SSE_{ik} = \sum_{c=1}^{C} \sum_{j \in Q_c} (w_{ij} - x_{kj})^2. \tag{10}$$

Because the dataset is binary, it can be divided into two parts: one part containing all positive data points (easily accessible through the sparse data format) and all zeros. This division leads to a rewriting of the calculation to

$$SSE_{ik} = \sum_{c=1}^{C} \sum_{j \in Q_c} \begin{cases} (1 - x_{kc})^2 & \text{if } w_{ij} = 1 \\ (0 - x_{kc})^2 & \text{if } w_{ij} = 0 \end{cases} \tag{11}$$

which can be simplified to

$$SSE_{ik} = \sum_{c=1}^{C} |Q_{ci}|_1 \times (1 - x_{kc})^2 + |Q_{ci}|_0 \times x_{kc}^2 \tag{12}$$

where Q_{ci} is the vector of all data points from cluster c and row i, $|Q_{ci}|_1$ the number of positive elements in that row and cluster, and $|Q_{ci}|_0$ being the number of zeros. $|Q_{ci}|_1$ and $|Q_{ci}|_0$ can be pre-calculated for all i's with complexity $O(|W|_1)$. This brings the total complexity of calculating the next iteration of the object clusters to $O(|W|_1) + O(nKC)$. When adding the complexity of calculating the next feature clusters, the complexity becomes $O(|W|_1 + nKC + mKC)$, matching 1 exactly.

5. Evaluation

5.1. Synthetic data

The algorithm can easily be tested by using synthetic data: one simply assigns certain labels to objects and features and then assigns object-feature-relations (1's in the matrix) according to those labels and certain probabilities. For example, it may create 2 row labels and 2 column labels, and say that for every data point in each co-cluster the probability is

$$\begin{matrix} .1 & .7 \\ .6 & .2 \end{matrix}.$$

The algorithm will then cluster the data, after which the given clusters can be compared to the labels that generated the data. If the clusters match the original labels, the algorithm is successful. However, this may not extend to real data, as the original data probably is not labeled (we would use supervised learning if that was the case). Even if the data were labeled, there may be different patterns beside those labels, which cannot be evaluated externally.

Therefore, an internal validation measure is required. There are a number of options to choose from. A bad score with these measures does not imply a bad clustering, as the measure assumes a certain structure in the data. The validation measures also need a dissimilarity measure which suffer from the curse of dimensionality.

5.2. Squared errors

As reducing the sum of squared errors between the approximated and original matrix is the goal of the algorithm, measuring it is a straightforward way of evaluating the performance of the algorithm. However, this sum does not mean anything without any context, so a standard to compare the numbers to should be devised as well. To get this standard we performed a permutation test. In a permutation test, new matrices are created by randomly

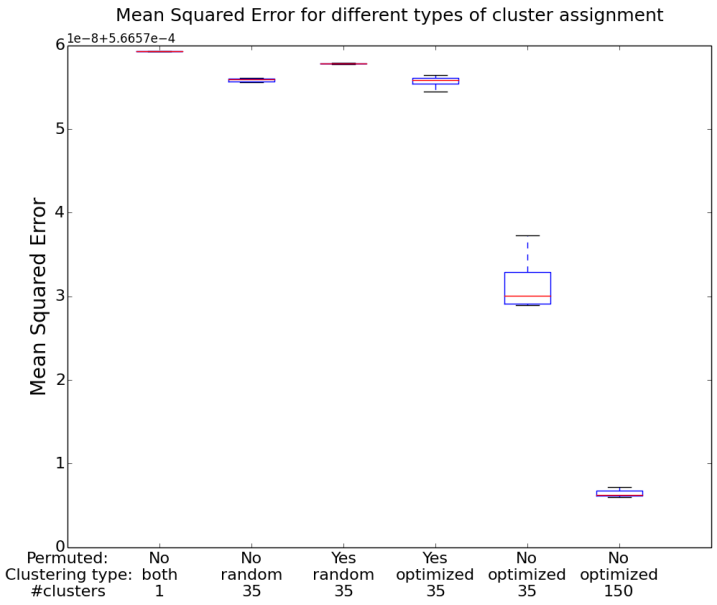

Figure 3: Clustering decreases the mean squared error of the original dataset significantly more than that of a permuted one.

permuting the original data. Those matrices are clustered, and the sum of squared errors is calculated. Besides that we also clustered the algorithm with $K = C = 1$ (no clusters), random cluster assignments and $K = C = 150$, as opposed to $K = C = 35$.

The mean squared error for the unclustered dataset is worse than the random cluster assignment. This makes sense as the approximation matrix with one co-cluster is just the average of the whole matrix, whereas an approximation matrix based on random assignment has slight fluctuations in the averages of the co-clusters, which approximate the original matrix slightly more. The unclustered, permuted matrix also profits from these fluctuations, but slightly less as it has less structure.

The more interesting observations can be seen in the right half of the figure. The clustering of the permuted matrix does improve the approximation slightly, because even randomness contains some patterns. However, the original data set provides significantly more information the algorithm can use than the permuted does, as can be seen in its effect on the mean squared error. As expected, more clusters means a closer approximation, and thus a lower mean squared error.

5.3. Visual comparison

Another way to assess the clusterings is by comparing the clusters using variables associated with the objects and features that were not known to the algorithm. For example, the age of the users in the different clusters can be compared, or the page category. Some results are in Figure 4. The values are normalized to compensate for differing cluster sizes.

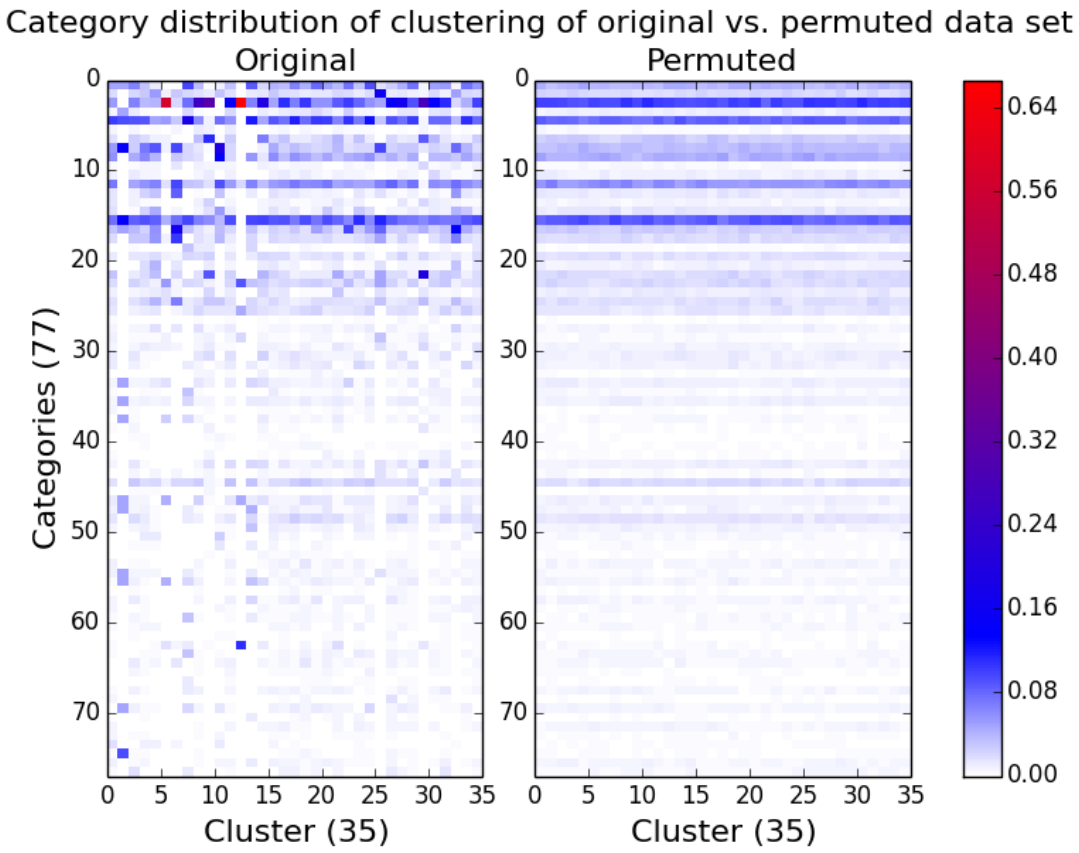

Figure 4: Correlation matrices for clusters and variables unknown to the algorithm. The values are normalized per cluster, so the values represent the fraction of the whole cluster

Clustering has an effect on the distribution of the categories over the clusters, in the original data set. It barely has an effect in a permuted data set. The effect is not massive, but that is to be expected, as the categories do not say anything inherently about the similarity between two pages. For example, classical music and metal music both fall in the same category, but there is no reason to assume they are similar. Conversely, Bach is considered an artist, a different category from classical music, but is obviously related to it.

Sometimes, however, most items in a category do seem related, and two categories seem very close together, as can be seen in cluster 30. In it, 90% of the likes are represented by two categories.

6. Discussion

Squared Error evaluation showed that the dataset does contain information. There are patterns in the data that the clustering algorithm can use to improve the clustering. How-

ever, extracting explicit information that could be used to make understandable predictions (people who like beer like snack bars) is another challenge.

Possible ways to improve the algorithm include fuzzy clustering, summing multiple clusterings and better pre-processing. Fuzzy clustering assigns an object to each cluster with a certain membership level. This might improve the prediction as every data point does not depend on merely one co-cluster. However, it may also overgeneralize. Summing multiple clusterings runs the algorithm multiple times and counts the number of co-occurences of every pair of objects (and features). The more often two objects (or features) co-occur, the more similar they are. During pre-processing a lot of noisy features were removed, but still many features only had only a couple of positive data points. The average number of likes per user was 55 out of 97 000 possible likes, which gives a sparsity of 99.94%. Stripping all users with fewer than 20 likes and similarly with pages could increase the density. It would also increase the speed of the algorithm, but since the dataset is smaller it would likely have a harder time predicting new data.

6.1. Uses

Can the algorithm, or any clustering algorithm, be useful in collaborative filtering? Yes, predictions can be made. However, it is hard to confirm if those predictions are correct. Also, dimensionality reduction and increasing the amount of information (gender, age for users, category for pages) should improve the results. Combining clustering with classification (semi-supervised learning) could also improve predictions, as the supervised portion of the data set could guide the unsupervised depending on their closeness.

For marketing purposes, clustering could be used, but a supervised method would be able to give more explicit results, since the data can be interpreted using the given labels.In (Kosinski and Graepel, 2013) the certainty of predictions could be established. If the data would be combined with marketing information (responsiveness to different marketing strategies, affinity towards certain products or brands), new marketing information could be predicted from the combined data.

6.2. Ethics

Besides marketing information, more sensitive and private information can be predicted from facebook data. For example, relationship status, sexual tendencies, race and religion can be predicted from Facebook likes (Kosinski and Graepel, 2013). It is important that companies treat their data carefully, but also for people to understand the risk of publicly showing information that may reveal personal information.

Acknowledgement

We would like to thank Media11 for providing the data set and T. Heskes and L. Vuurpijl for their guidance.

References

T. George and S. Merugu. A scalable collaborative filtering framework based on co-clustering. *Fifth IEEE International Conference on Data Mining (ICDM05)*, 5:625628, 2005. doi: 10.1109/ICDM.2005.14.

A. K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31(8): 651666, 2010. doi: 10.1016/j.patrec.2009.09.011.

Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. URL http://www.scipy.org/.

Stillwell D. Kosinski, M. and T. Graepel. Private traits and attributes are predictable from digital records of human behavior. *Proceedings of the National Academy of Sciences of the United States of America*, 110(15), 2013. doi: doi:10.1073/pnas.1218772110.

T. Li. A general model for clustering binary data. *Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, 11:188, 2005. doi: 10.1145/1081870.1081894.

C. X. Ling and C. Li. Data mining for direct marketing : Problems and solutions, 1998.

Subramaniam C. Tan G. W. Shaw, M. J. and M. E. Welge. Knowledge management and data mining for marketing. *Decision Support Systems*, 31(1), 2001.

L. H. Ungar and D. P. Foster. Clustering methods for collaborative filtering, 1998.

Glossary

algorithm A program that a computer can use to process data

benchmarking The act of measuring performance of a system

boxplot A graphical representation of a list of data, it shows the quartiles and outliers

classification A method for dividing a set of objects into groups such that objects in a group belong together, based on the object features

cluster A subset of a set of items, usually objects, and in this case features as well. Clusters are generated by unsupervised grouping algorithms

clustering Verb: The act of forming clusters, or grouping objects in data without known labels. Noun: A division of a dataset into clusters such that each objects belongs to (at least, but in this case exactly) one cluster

co-cluster A co-cluster is a subset of data points such that all data points in the co-cluster belong to the same object cluster and feature cluster

collaborative filtering The act of selecting (filtering) a number of products for a user based on users that are similar to that user (collaboration)

curse of dimensionality When objects in a dataset have a large number of features, distance becomes uniform and thus less useful

data point A data point is a single value in a matrix identified by it's location on all axes of the matrix. In the case of a Facebook like-matrix, every such point is a like

data set The data from which information should be extracted, can take the form of a set of tables with attributes or a matrix with data points

expectation maximization An iterative approach to estimate parameters in a statistical model

Facebook A social medium people use to communicate to their friends what they find interesting

Facebook page A page containing information about a certain topic or person. If a person associates with a page, he can like it to show this association to his friends

feature The objects in the dataset contain features that are used to cluster the objects. In co-clustering the features themselves are also clustered. Features are usually words, pixel-values, or in this case, Facebook pages. In a matrix representation of a dataset these are usually represented by the columns

K-means The oldest and most well-known clustering algorithm. It assigns every object to one of K clusters based on their distance to that clusters mean, recalculates the mean and repeats

like On Facebook liking something is an action of approval. In terms of data it is an association between a user and a page

object In clustering the goal is to group the objects in a dataset. These are usually books, products, web pages, or, in this case, users. In a matrix representation of a dataset these are usually represented by the rows

recommender system A system that recommends products to users. Often employs Collaborative Filtering

squared error A statistical measure of closeness of two objects, by squaring the error between every point of the two objects. These errors can be summed or averaged to get the final measure

synthetic data Data that is generated rather than observed. Can be used to test a system before applying it on real-world data

user A user is a person with a Facebook account which has certain values, the gender, age and which pages the liked on Facebook, for example

unsupervised classification A classification method where the groups are not known beforehand and objects are grouped solely based on their features