Sparse Subspace Clustering for User Identification

Yuki Nishimura
M.S. in Data Science
Columbia University
New York, USA
yn2373@columbia.edu

***Abstract*—Standard recommendation methods rely on the assumption that each account is used by a single user. In reality though, multiple users may be sharing one account. In these cases, recommendations may lean toward more dominant users, or generalize to the combined weak preferences of multiple users. In addition, with shared accounts, services may be losing potential revenue. To solve these issues, the best way is to distinguish the users sharing the account. This is known to be a chicken and egg problem, because if we know the number of users sharing the account beforehand, it is relatively easy to allocate users to the items of their preferences, and if we know the pairings of users and items beforehand, it is trivial to discover the number of users using the account. Since this problem setting is analogous to the subspace clustering setting, where the objective is to find the number of subspaces, and the data points belonging to each subspace, we apply subspace clustering to this problem. Thus, in this project the objective is to apply subspace clustering to an account $\times$ item matrix, to distinguish the users sharing an account. Specifically, we apply Sparse Subspace Clustering to the MovieLens dataset, with major focus on discovering the pairings of users and movies in the multiple user setting. We show results of this experiment and discuss directions we could take for further research.**

***Index Terms*—sparse subspace clustering, recommendation systems, user identification, movielens**

I. Introduction

Recommendation systems have been used extensively by numerous online services to improve user experience. There have been many methods used for recommendation systems, including content based methods, collaborative filtering methods, and hybrid methods. Content based methods are methods that utilize feature information about items and a user preference profile to classify whether a specific user likes or dislikes an item. Collaborative filtering methods utilize only the rating data of users and items, and assumes that if different users had similar preferences in the past, then they would have similar preferences in the future as well. A popular collaborative filtering method is the Matrix Factorization method, where a user $\times$ item matrix is decomposed into a latent user matrix and latent item matrix, with the assumption that the user $\times$ item matrix is low rank due to the similarity in preferences of users and similarity in items. Hybrid methods combine the advantages of content based methods and collaborative filtering methods, by applying content based methods when the data size is still small, and applying collaborative filtering methods when there is less diversity in recommendations.

Although the above methods have achieved great results in recommendations, generally they all assume one user per account. In reality though, many accounts in online services may be shared by multiple users. The multiple user setting introduces 3 main problems that need attention. First, the quality of recommendations may deteriorate when there are multiple users, since the recommendations can focus on the more dominant user, or the recommendations may mistakenly decide that the combined weak preferences of multiple users is a strong preference of the account. Second, the sharing of accounts may reduce the revenue of companies that provide online services. Although some of these companies allow only one user per account, some users may be taking advantage by sharing accounts with other users for less payment, leading to a loss in potential revenue. Lastly, there are some privacy concerns that require deliberation, mainly due to the possibility of accurate identification of specific users in the multiple user setting. From these problems, the multiple user setting requires deep consideration.

In this paper, we focus on the multiple user setting with emphasis on identifying the different users sharing an account. For the identification of different users sharing an account, we first must identify the number of users of an account, then allocate the users to the items they each interacted with. This is a known chicken and egg problem, where the other step is dependent on the other step. Subspace clustering is thought to have potential to solve these types of problems, by identifying the number of subspaces and the subspaces the data points belong to. Therefore, here we will apply Sparse Subspace Clustering, the state-of-the-art subspace clustering method for the identification of users in shared accounts. But for simplicity, we will focus on the latter problem of identifying the items each user interacted with.

II. Previous Work

A. *Implicit User Identification*

Verstrepen et al. [2] proposed a method that solved the dominance and generality problem in the multiple user setting, by an approximation based on an assumption that the resulting recommendation score of an item will be the highest when the input items used to calculate the recommendation score are from the same user. As a result they were able to make recommendations that were less biased toward the dominant users, and were also able to make recommendations that were highly preferred by each user. Despite these results, since the users could not be distinguished within the accounts, both the loss of revenue problem and privacy concern problem could not be tackled.

B. Graph Based Method

A graph-based explicit user identification method proposed by Jiang et al. [4] represented sessions that occurred on an account with graph embeddings, which were later clustered using affinity propagation. Accurate separation of users and recommendations were achieved, only leaving behind the problem of metadata collection. Since metadata may not be obtainable in some cases, this approach is only applicable when enough metadata is provided.

C. Subspace Clustering

A method that does not require metadata and attempts to explicitly identify users is the subspace clustering method proposed by Zhang et al. [3]. Generalized Principal Component Analysis (GPCA) [6], a subspace clustering method, was applied to find the number of subspaces (users), and data belonging to each subspace (items of each user). Although the problem setting of identifying users is analogous to the subspace clustering, the GPCA worked poorly in identifying the items each user interacted with, underperforming other baseline clustering methods such as K-means and Spectral Clustering. The reason mentioned was that standard clustering methods were able to group together similar movies without focusing too much on the ratings that were given to those movies. This may be related to the fact that people tend to rate movies that they like, instead of movies that they dislike. In addition, this was not mentioned but another hypothesis that could be given is that GPCA was not able to perform well due to the presence of noise in the data.

III. Approach

Previous work exhibited different possible directions of improvement in the multiple user setting. Here, we decide to focus on subspace clustering, since there may be more cases where metadata is not available, and achieving explicit user identification could help resolve the 3 main problems that were mentioned earlier.

One of the possible reasons of the failure in subspace clustering was due to the characteristic of GPCA, where differentiating polynomials were needed to obtain subspaces, which was highly sensitive to noise and outliers. As an improvement of this method, Sparse Subspace Clustering (SSC) was proposed by Elhamifar et al. [5], which took a completely different approach. This approach was based on the fact that each point in a union of subspaces had a sparse representation with respect to a dictionary formed by all the other data points in the same subspace. By solving a series of basis pursuit problems and applying spectral clustering, this method was able to effectively cluster data points from a union of subspaces. In addition, Lasso optimization could be incorporated in each of the basis pursuit problems, allowing the model to become more robust towards outliers and noise. Thus, in this paper we will apply SSC to address the user identification problem.

The assumption in our problem setting is that each data point, which consists of an item representation and the preference of that item, will lie on the same subspace if they are from the same user. This is very intuitive, because each user will have similar preferences towards similar items, which will make data points generated by a user lie on the same or close subspace. Following this assumption, we would need to acquire an item representation for each item prior to applying subspace clustering. Including this step needed for item representation retrieval, our approach follows the approach taken by Zhang et al. [3], which could be divided into the 3 steps which are explained next.

A. Item Representation Retrieval with Matrix Factorization

To obtain item representations for each item, Matrix Factorization (MF) [1] will be used. Generally, MF is used to fill in missing values of a user $\times$ item matrix for collaborative filtering, but here we make use of the latent item matrix obtained after the MF training. The MF training is done according to the following loss function L,

$$L = min \sum_{u,i} (r_{ui} - \mathbf{x}_u^T \mathbf{y}_i - b_u - b_i)^2 + \lambda_x \sum_u ||\mathbf{x}_u||^2 + \lambda_y \sum_i ||\mathbf{y}_i||^2 \tag{1}$$

where r_{ui} is the preference of item i given by user u, x_u is the latent user vector of user u, y_i is the latent item vector of item i, b_u is the bias term for user u, b_i is the bias term for item i, λ_x is the regularization term for latent user vectors, and λ_y is the regularization term for latent item vectors. The dimension K of x_u and y_i is a hyperparameter that represents the rank of the target matrix R, where each element of R is r_{ui}. Taking the derivative of the terms in the minimization with respect to both $\mathbf{x}_u$ and $\mathbf{y}_i$ then setting them to 0, L can be minimized by alternating the minimization between $\mathbf{x}_u$ and $\mathbf{y}_i$ in the following manner,

$$\tilde{\mathbf{x}}_u^T = \mathbf{r}_u^{\mathbf{b_i}} \tilde{Y} (\tilde{Y}^T \tilde{Y} + \lambda_x I)^{-1} \tag{2}$$

$$\tilde{\mathbf{y}}_i^T = \mathbf{r}_i^{\mathbf{b_u}} \tilde{X} (\tilde{X}^T \tilde{X} + \lambda_y I)^{-1} \tag{3}$$

where $\tilde{\mathbf{x}}_u^T = (\mathbf{x}_u^T, b_u)$, $\tilde{\mathbf{y}}_i^T = (\mathbf{y}_i^T, 1)$, $\mathbf{r}_u^{\mathbf{b_i}} = \mathbf{r}_u - \mathbf{b_i}$ is set before the first part of the update, and $\tilde{\mathbf{x}}_u^T = (\mathbf{x}_u^T, 1)$, $\tilde{\mathbf{y}}_i^T = (\mathbf{y}_i^T, b_i)$, $\mathbf{r}_i^{\mathbf{b_u}} = \mathbf{r}_i - \mathbf{b_u}$ is set before the second part of the update. After iterating until convergence, we can obtain the $\mathbf{y_i}$ vector to use as the item representation for item i. We obtain this vector for all of the items.

B. Construction of Artificial 2-user Account

Now that we have obtained the item representations, we will construct an artificial 2-user account matrix. Specifically, 2 random users are sampled without replacement from the user space and the item representations of items that they interacted with are combined to define 1 2-user account matrix. At the end of each item representation vector of the 2-user account matrix, the corresponding preference score (usually the rating) that the user gave to the item is appended, to create the 2-user account matrix with preference scores V_a. This is repeated until no users are left, assuming that there existed an even number of users. In the case where there were an odd number of users, then the last single user left could be discarded.

C. Sparse Subspace Clustering on each Account

After the previous step, each 2-user account a will have a corresponding V_a matrix. SSC will be performed on each V_a matrix, identifying which users interacted with which item. Specifically, each item representation preference vector $\mathbf{v}_{ai}$ in the V_a matrix will be given a label of either one of the two users after SSC. SSC will be performed as follows.

First, for an account a, the following objective function will be minimized for each i:

$$min||\mathbf{c}_i||_1 \tag{4}$$
$$\text{s.t. } \mathbf{v}_{ai} = V_{a\hat{i}}^T \mathbf{c}_i.$$

Here, $\mathbf{c}_i \in \mathbb{R}^{N_a-1}$ is the target sparse vector, and $V_{a\hat{i}} \in \mathbb{R}^{(N_a-1)\times(K+1)}$ is a matrix in which $\mathbf{v}_{ai}$ was removed from V_a. Note that N_a is the number of item representation preference vectors in V_a. Intuitively, the objective function finds the sparsest combination of the item representation preference vectors from V_a that could form the vector $\mathbf{v}_{ai}$ without using $\mathbf{v}_{ai}$ from V_a. Since the above objective function solves for equality, it may not be compatible for data with noise. Therefore, we relax the objective function to the following Lasso optimization:

$$min\frac{1}{2}||\mathbf{v}_{ai} - V_{a\hat{i}}^T \mathbf{c}_i||_2^2 + \lambda||\mathbf{c}_i||_1 \tag{5}$$

This is precisely the basis pursuit denoising problem that is common in sparse recovery. Thus, we could use accelerated proximal gradient (APG) to minimize the objective function efficiently. The APG algorithm [13] is as follows:

```
Initialize
$\mathbf{c}_{i0} \in \mathbb{R}^{N_a-1}$ with random initialization
$\mathbf{p_1} = \mathbf{v}_{ai1} \leftarrow \mathbf{v}_{ai0}$
$t_1 \leftarrow 1$
$L \geq \lambda_{max}(V_{a\hat{i}}^T V_{a\hat{i}})$
while $\mathbf{c}_{ik}$ has not converged (where $k$ is the iteration number) do
    $t_{k+1} \leftarrow \frac{1+\sqrt{1+4t_k^2}}{2}$
    $\beta_{k+1} \leftarrow \frac{t_k - 1}{t_k+1}$
    $\mathbf{p}_{k+1} \leftarrow \mathbf{c}_{ik} + \beta_{k+1}(\mathbf{c}_{ik} - \mathbf{c}_{i(k-1)})$
    $\mathbf{w}_{k+1} \leftarrow \mathbf{p}_{k+1} - \frac{1}{L} V_{a\hat{i}}^T (V_{a\hat{i}} \mathbf{p}_{k+1} - \mathbf{v}_{ai})$
    $\mathbf{c}_{i(k+1)} \leftarrow soft(\mathbf{w}_{k+1}, \lambda/L)$
end while
Output $\mathbf{c}_{i(k+1)}$
```

where $soft$ is the soft-thresholding operator.

After obtaining $\mathbf{c}_{i(k+1)}$, a zero is inserted to the ith position of $\mathbf{c}_{i(k+1)}$. This process is repeated for all i in account a to obtain the matrix $C = [\hat{\mathbf{c}}_1, \ldots, \hat{\mathbf{c}}_{N_a}] \in \mathbb{R}^{N_a \times N_a}$. Then $\tilde{C}$ is formed by $\tilde{C_{ij}} = |C_{ij}| + |C_{ji}|$, which is the adjacency matrix used as input for spectral clustering. Since each of the rows of $\tilde{C}$ correspond to the item representation preference vectors in the form of nodes of a graph structure, spectral clustering will cluster the nodes, resulting in the clustering of item representation preference vectors. The hope is that as a result, the clusters will represent each user of an account, and contain the item representation preference vectors that correspond to each user.

IV. Experiment

We applied our proposed approach on the MovieLens Latest Small Dataset [14]. This dataset consists of 610 unique users and 9724 movies, with ratings beginning from 0.5 to 5 increasing by 0.5. Note that for this dataset, items are movies, and preferences are ratings. From this dataset, we filtered out users that rated less than 100 movies or rated more than 999 movies to balance the number of movies each user rated to the order of 100s. In addition we filtered out movies that were rated less than 2 times, resulting in a dataset consisting of 236 unique users and 4860 movies. For MF, the hyperparameters were chosen as $K = 10, \lambda_x = 10, \lambda_y = 10$, and were run for 30 epochs until convergence.

After constructing the artificial 2-user accounts from the obtained movie representations, we applied SSC on each account matrix. For comparison, we also applied K-means and spectral clustering on each account matrix. For all of these clustering methods, the number of subspaces (clusters) were fixed to 2 following the experimental set-up of Zhang et al. [3]. Note that in a real-life setting though, we should apply BIC or use the number of zero eigenvalues of the Laplacian matrix obtained during spectral clustering to select the number of subspaces.

Evaluation of user identification quality was based on the similarity metric also proposed by Zhang et al. [3]. Given a mapping $I : V_a^s \rightarrow \{0, 1\}$, where V_a^s is the set of movie representation rating vectors for account a, the true mapping I^*, and $\Pi(\{0, 1\})$, which is the set of permutations of $\{0, 1\}$, the similarity metric could be expressed as the following.

$$S(I, I^*) = \max_{\pi \in \Pi(\{0,1\})} \frac{1}{N_a} \sum_{\mathbf{v}_{ai} \in V_a^s} 1\{\pi(I(\mathbf{v}_{ai})) = I^*(\mathbf{v}_{ai})\} \tag{6}$$

Since there is no prior knowledge of which user corresponds to cluster 0 and which user corresponds to cluster 1, the permutations of {0,1} that maximizes the similarity metric will be taken. Thus, this similarity metric will always be greater than or equal to 0.5. For each account matrix, we calculate the similarity of the true labels and cluster labels, then organize them as a cumulative distribution in Figure 1.

The results show that spectral clustering performed the best, and SSC performed the worst in terms of similarity. Viewing the actual labels of the clustering though, spectral clustering had a strong tendency to give labels to item representation preference vectors that were mostly 0's or 1's. Therefore, for spectral clustering the similarity shown is not actually showing the quality of the clustering, but it is showing the unbalance in the number of movies rated by users in each account. On the other hand, K-means and SSC both had cluster labels

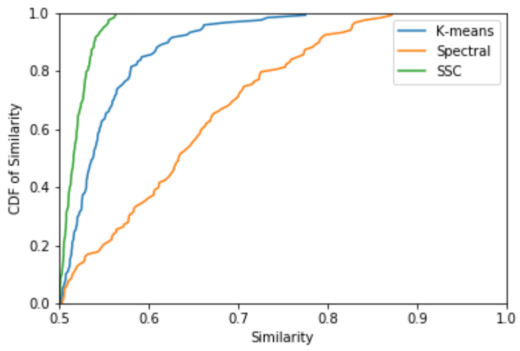

Fig. 1. Similarities of Different Clustering Methods

TABLE I
GENRE RATIO DIFFERENCES OF CLUSTERS OF REPRESENTING ACCOUNT

	K-Means	Spectral	SSC
Difference of Genre Ratio	0.293	1.123	0.201

that were more balanced, providing a better assessment of user identification quality. K-means may have performed better than SSC because K-means does not give focus on a single feature such as ratings, and decided on clusters depending on the similarities in item representations. SSC may have given too much focus on ratings, which prevented the method from performing well with this dataset.

Another reason for why the SSC did not perform well for user identification may be due to the way the problem was set up. The assumption was that each data point, which consists of a movie representation and the rating of that movie, will lie on the same subspace if they are from the same user. Since users may rate movies randomly regardless of the content or genre of the movie, the assumption may not have held true. Thus, it may be that SSC did not actually perform horribly in the clustering task. To evaluate this, we calculate the genre ratios in each of the two clusters obtained, and sum up the absolute value of the difference in genre ratio between clusters. If there are large differences in genre ratios between clusters, it would mean that the different types of items belong in each cluster, meaning that the clustering worked better.

Table I shows that spectral clustering had the biggest difference in genre ratio between clusters, which was because there were only a few data points in one cluster over another. Focusing on K-means and SSC, it seems that K-means provided a larger difference in genre ratio. Thus, in terms of better quality clusters regarding the difference in genres of clusters, K-means performed better than SSC. But these results can lead us to the next hypothesis that SSC also took into account the ratings of movies given by each user, which diluted the differences in genre. This could definitely be another direction of research.

V. DISCUSSION

The low performance of SSC, the state-of-the-art algorithm for subspace clustering, was not what was expected, which hints that there may have been an issue with the problem setting. It may be that identifying users with only movie representations and movie ratings was not possible with subspace clustering in the first place, or that there needs to be additional metadata to support the decision making of subspace clustering. In addition, the lack of data could have caused SSC to be unable to discover a subspace representing each user, or representing a user with one subspace was not modeling real-life accurately. With many different possible hypotheses for why SSC was low performing, some future directions that could be taken to untangle this mystery is to change the dataset, increase the data size, and use different item representations. There would be great excitement if subspace clustering becomes compatible with these multiple user setting problems, since they seem very similar to start with.

REFERENCES

[1] Y. Koren, R. Bell and C. Volinsky, ”Matrix Factorization Techniques for Recommender Systems,” in Computer, vol. 42, no. 8, pp. 30-37, Aug. 2009, doi: 10.1109/MC.2009.263.
[2] K. Verstrepen, B. Goethals, “Top-N Recommendation for Shared Accounts,” RecSys ’15: Proceedings of the 9th ACM Conference on Recommender Systems. 2015.
[3] A. Zhang, N. Fawaz, S. Ioannidis, and A. Montanari, “Guess Who Rated This Movie: Identifying Users Through Subspace Clustering,” 28th Conference on Uncertainty in Artificial Intelligence.2012.
[4] J. Jiang, C. Li, Y. Chen, and W. Wang, “Identifying Users behind Shared Accounts in Online Streaming Services,” SIGIR’18, Ann Arbor, MI, USA. July 8-12, 2018.
[5] E. Elhamifar and R. Vidal, ”Sparse subspace clustering,” 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, 2009, pp. 2790-2797, doi: 10.1109/CVPR.2009.5206547.
[6] R. Vidal, Y. Ma, and S. Sastry, “Generalized principal component analysis (gpca),” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol 27, no 12, 2005.
[7] J. Wright, Y. Peng, Y. Ma, A. Ganesh, S. Rao, “Robust Principal Component Analysis: Exact Recovery of Corrupted Low-Rank Matrices by Convex Optimization,” Advances in Neural Information Processing Systems 22. 2009.
[8] F. Ong and M. Lustig, ”Beyond Low Rank + Sparse: Multiscale Low Rank Matrix Decomposition,” in IEEE Journal of Selected Topics in Signal Processing, vol. 10, no. 4, pp. 672-687, June 2016, doi: 10.1109/JSTSP.2016.2545518.
[9] T. Bouwmans, A. Sobral, S. Javed, K.J. Soon, and E. Zahzah, “Decomposition into Low-rank plus Additive Matrices for Background/Foreground Separation: A Review for a Comparative Evaluation with a Large-Scale Dataset,” Computer Science Review. February 2017.
[10] R.P. Prima, D. Nurjanah, and R. Rismala, “Top-N Recommendation for Shared Account on Book Recommender System,” International Conference on Information Technology Systems and Innovation (ICITSI). 2018.
[11] J. Pu, Y. Panagakis, and M. Pantic, “Learning Low Rank and Sparse Models via Robust Autoencoders,” IEEE International Conference on Acoustics, Speech and Signal Processing. 2019.
[12] C. Zhou, and R.C. Paffenroth, “Anomaly Detection with Robust Deep Autoencoders,” KDD’17, August 13–17, 2017, Halifax, NS, Canada.
[13] Y. Ma and J. Wright, “Sparse and Low-Dimensional Models for High-Dimensional Data: Theory, Algorithmsand Applications,” Cambridge. 2020.
[14] F.M. Harper and J.A. Konstan, “The MovieLens Datasets: History and Context,” ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4: 19:1–19:19. ¡https://doi.org/10.1145/2827872¿. 2015.