Contextual Clustering

Géraud Le Falher

May 25, 2016

1 Problem definition

We observe users and their similarities along different known directions. We assume that these links arise from users' hidden profile and our goal is to recover these profiles to answer queries like: are users i and j similar along direction x? To better model real situations, we introduce a further bias: along all (*relevant?*) directions, there are only a small number of users clusters.

This actually defines the 3 outlined research questions:

1. *which situations are we modelling*
2. *answer* $(i, j, \boldsymbol{x})$ *similarity query (online)*
3. *recover the* $\boldsymbol{u_i}$*s*

Formally, there are n users $\boldsymbol{u_1}, \dots, \boldsymbol{u_n} \in \mathbb{R}^d$ (*and* $||\boldsymbol{u_i}|| = 1 \ \forall i$ *?*). Directions are a finite set of unit vector of $\mathbb{R}^d$ ($S = \{\boldsymbol{x_1}, \boldsymbol{x_2}, \dots, \boldsymbol{x_{|S|}}\}$). Thus, $\boldsymbol{u_i} \cdot \boldsymbol{x}$ literally measures the alignment of $\boldsymbol{u_i}$ with the direction $\boldsymbol{x}$. For instance, $\boldsymbol{x}$ may be the features of an item and $\boldsymbol{u_i} \cdot \boldsymbol{x}$ a rating. The similarity between $\boldsymbol{u_i}$ and $\boldsymbol{u_j}$ along $\boldsymbol{x}$ is $sim(\boldsymbol{u_i}, \boldsymbol{u_j}) = (|\boldsymbol{u_i} \cdot \boldsymbol{x} - \boldsymbol{u_j} \cdot \boldsymbol{x}| < t) \in \{0, 1\}$, where $t \in \mathbb{R}^+$ is a threshold ((or absolute value can be replaced by the square function).

As input, we are given a set of positive examples $\{(i_k, j_k, \boldsymbol{x^k} \in S)\}_{k=1\dots D}$ that we represent as a multigraph: nodes are users, edges are labelled by the direction along which they represent similarity.

2 Settings

In the following we consider batch setting, with a fully observed graph as test set. Yet for the sake of scalability, online setting may be more appropriate: either we see the whole graph but edge labels are given one by one or, more challenging, both edges and their labels are revealed at each time step.

3 Applications, use cases

As defined, the problem is rather general. Here we provide two concrete examples of application. The first is recommender systems. Say we observe a ratings matrix, where rows are users and columns items. Items are characterised by a known vector of d features. Users react to these features according to their hidden profile. We build the train set from the observed entries of the rating matrix and later fill missing entries by answering similarity queries (although if ratings from both users are missing, then we can only say whether they are equal or not, but not their value)

The second is social networks. Here users also have features (some of them are known, like demographics) and they like contents. *(and then what?)*

4 Related work

- bandits [1, 2]
- Online matrix completion [3]
- label propagation [4]
- multilabel classification [5]
- Recovering d different metric spaces whose intersection form the observed social multiplex in almost linear time with bounded distortion [6]. Assume there are K social categories modelled by Euclidean spaces $\mathcal{D}_i$. Users in there are near uniformly distributed and categories have small local correlation:

  $$\forall i \neq i',\ \forall r, r' = O(polylog(n)),\ \forall u, u',\ |\mathcal{B}_i(u, r) \cap \mathcal{B}_{i'}(u', r')| \leq O(\log n)$$

  These spaces give rise to small world networks $\mathcal{G}_i$ with edge probability $\propto \mathcal{D}_i(u, v)^{-d}$ and we observe the real network $\mathcal{G} = \bigcup_i \mathcal{G}_i$. From $\mathcal{G}$, the proposed algorithm recovers in $n\text{polylog}n$ time a bounded approximation $\mathcal{D}'_i$ of all $\mathcal{D}_i$

  $$\sigma\mathcal{D}_i(u, v) \leq \mathcal{D}'_i(u, v) \leq \delta\mathcal{D}_i(u, v) + \Delta$$

- online similarity/dissimilarity prediction on graph [7]
- coclustering [8] (except there's no notion of user/item features there)

5 Approaches

5.1 Baselines

A natural way to recover the $\boldsymbol{u}$ vectors is to solve the following optimization problem:

$$\min \sum_{k=1}^{D} \left((\boldsymbol{u_{i_k}} - \boldsymbol{u_{j_k}}) \cdot \boldsymbol{x^k} \right)^2 \tag{1}$$

which can also be expressed as a quadratic program, denoting by $\tilde{\boldsymbol{u}}$ the concatenation of all users vectors:

$$\tilde{\boldsymbol{u}}^{\boldsymbol{T}} Q \tilde{\boldsymbol{u}} + \boldsymbol{c}^T \tilde{\boldsymbol{u}}$$

A caveat of this method is lack of scalability. Furthermore, we need to impose constraints to avoid the trivial solution $\tilde{\boldsymbol{u}} = 0$. If we constrain $\boldsymbol{u}$ to be unit norm, the problem is not convex. An alternative is to fix some values: either we already know some user profiles or some dimension.

Another immediate idea is to solve d separate problems, one across each dimension, and somehow combine the results. Yet this is also not very satisfactory, as it looses all cross dimension information.

5.2 Approximation

To simplify the problem, we could transform it from vector to scalar by using random projections. Indeed, let $\boldsymbol{a} \in \{-1, 1\}^d$ be a vector drawn uniformly at random. Then $\mathbf{E}_{\boldsymbol{a} \in \{-1,1\}^d} \left[(\boldsymbol{u} \cdot \boldsymbol{a}) \times (\boldsymbol{x} \cdot \boldsymbol{a}) \right] = \boldsymbol{u} \cdot \boldsymbol{x}$ and we get a weighted version of (1):

$$\min \sum_{k=1}^{D} (\boldsymbol{x_k} \cdot \boldsymbol{a})^2 \left((\boldsymbol{u_{i_k}} - \boldsymbol{u_{j_k}}) \cdot \boldsymbol{a} \right)^2$$

Then we would like to take advantage of the graph structure as well (which is implicit in (1), as each edge contributes one term to the sum). For instance by sparsifying the graph with relevant trees. Yet one should keep in mind that the input is a multigraph, whose edges labels are vectors and not mere weights. One idea could be to cluster the set of all directions of the training set using k-means and work on these k induced graphs.

Instead of random $\boldsymbol{a}$, we could also use directions in S. Indeed, let's define the loss incurred by our estimated profiles by projecting over all S:

$$\mathcal{L} = \sum_{i,j \in E} \sum_{\boldsymbol{x_l} \in S} (\boldsymbol{x_{ij}} \cdot \boldsymbol{x_l}) \left((\boldsymbol{u_i} - \boldsymbol{u_j}) \cdot \boldsymbol{x_{ij}} \right)^2$$

where we drop the square as we assume that all directions in S are in the positive orthant of $\mathbb{R}^d$.

By inverting the summation signs and considering that E is union of the edge sets E_m induced by each of the directions in S, we have

$$\begin{aligned}\mathcal{L} &= \sum_{\boldsymbol{x_m} \in S} \sum_{\boldsymbol{x_l} \in S} \boldsymbol{x_l} \cdot \boldsymbol{x_m} \sum_{i,j \in E_m} ((\boldsymbol{u_i} - \boldsymbol{u_j}) \cdot \boldsymbol{x_m})^2 \\ &= \sum_{m=1}^{|S|} V_m^T L_m V_m \sum_{\boldsymbol{x_l} \in S} \boldsymbol{x_l} \cdot \boldsymbol{x_m}\end{aligned}$$

where:

$$\begin{aligned}L_m &\text{ is the laplacian of } E_m \\ V_m &= U^T X_m \\ U &= [\boldsymbol{u_1} | \boldsymbol{u_2} | \dots | \boldsymbol{u_n}]\end{aligned}$$

6 First practical steps

This part is quite messy...

6.1 datasets

6.1.1 real

textbfMovieLens 1M[1] contains demographics information about 6040 users, namely age bracket, job (converted to median income), gender and ZIP code (converted to population density), whereas **hetrec2011-movielens-2k**[2] contains the following information on a subset of movies: release year, nationality (converted to 0: US,UK,CA,AUS; 1: EU; 2: others), number of critics ratings on rottentomatoes[3], average critics ratings, number of audience ratings, average ratings. In addition, for some of the pair (user, movie), we have a integer rating between 1 and 5.

To match the dimension between users and movies features, I later ditched demographics attributes and instead fit to each user ratings a linear model[4], which add a dimension to store the intercept parameter.

R6A - Yahoo! Front Page Today Module User Click Log Dataset[5]. While it was originally very rich, we only observe the result of a heavy pre processing described in [9]. Users and articles are described by 6 features and there is also binary feedback: article cliked or not.

Unfortunately, being real data, they don't fit well our assumptions. Specifically, computing dot products between all possible user/items pairs, there is no threshold that discriminate between similar ratings or not. Thus we turn to synthetic data.

[1] http://files.grouplens.org/datasets/movielens/ml-1m-README.txt
[2] http://ir.ii.uam.es/hetrec2011/datasets/movielens/readme.txt
[3] rottentomatoes.com
[4] `sklearn.linear_model.Ridge`
[5] http://webscope.sandbox.yahoo.com/catalog.php

6.1.2 synthetic

We generate n users and m items as uniform random unit vectors. Then we discretize all $\boldsymbol{u_i} \cdot \boldsymbol{x_j}$ to get ratings between 1 and 5 and finally we set a proportion z of ratings to 0 (unobserved). *This doesn't take the clustering assumption into account.*

We then sample 50% of the non zero entries. Items hit by this process are in the train set while the others will be used for test. Users hit will later be divided into train and test set, while the others are excluded (since we didn't observe anything about them, we can't learn anything about them either). Then we infer user profiles from available data (*this time I use a noisy version of the original* $\boldsymbol{u}$ *vectors, which is admittedly an easier setting*).

Now we can build the input multigraph, where there is an edge between two users along an item if they gave it the same rating. As there are many possible such edges and we initially want to iterate fast, I decided to discard some of them. Basically I consider only edges between users that are close, as measured by the distance between their common ratings vector. *A more principled way would be to consider all edges and discard those whose dot product difference is above a given threshold.*

Finally we solve (1) and get the results shown in Figure 1.

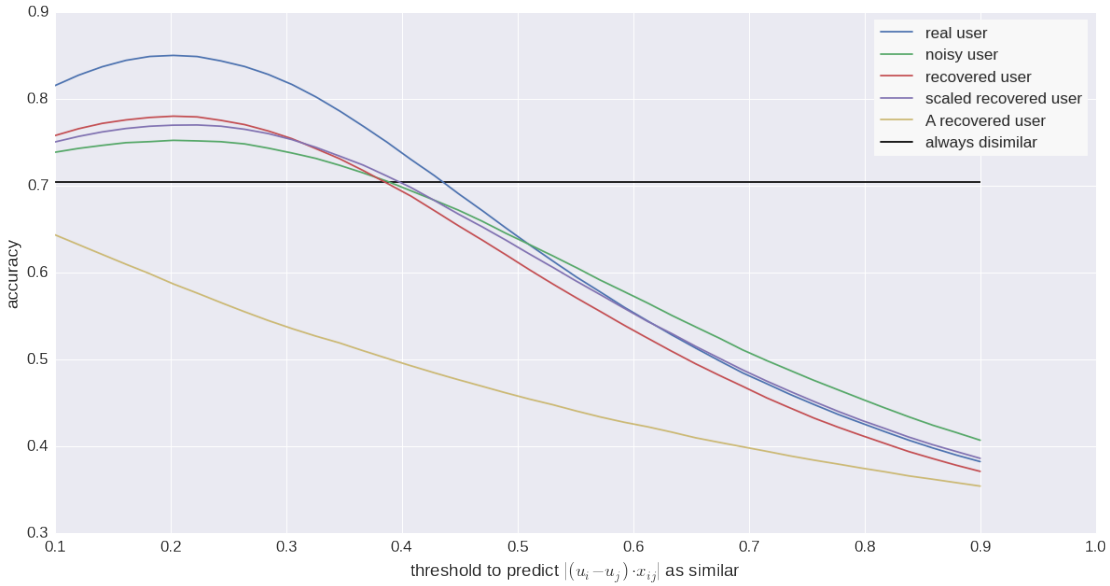

Figure 1: .

References

[1] C. Gentile *et al.*, "Online clustering of bandits", in *ICML 2014*, 2014.

[2] S. Li *et al.*, "Data-dependent clustering in exploration-exploitation algorithms", *ArXiv e-prints*, p. 8, 2015.

[3] B. Lois *et al.*, "Online matrix completion and online robust pca", *ArXiv e-prints*, 2015.

[4] X. Zhu *et al.*, "Semi-supervised learning using gaussian fields and harmonic functions", *Int. Conf. Mach. Learn. - ICML 2003*, vol. 20, no. 2, p. 912, 2003.

[5] G. Madjarov *et al.*, "An extensive experimental comparison of methods for multi-label learning", *Pattern Recognit.*, vol. 45, no. 9, pp. 3084–3104, 2012.

[6] I. Abraham *et al.*, "Low-distortion inference of latent similarities from a multiplex social network", p. 51, 2012.

[7] C. Gentile *et al.*, "Online similarity prediction of networked data from known and unknown graphs", *JMLR Workshop Conf. Proc.*, vol. 30, pp. 1–34, 2013.

[8] I. S. Dhillon, "Co-clustering documents and words using bipartite spectral graph partitioning", in *Proc. seventh ACM SIGKDD Int. Conf. Knowl. Discov. data Min. - KDD '01*, 2001, pp. 269–274.

[9] W. Chu *et al.*, "A case study of behavior-driven conjoint analysis on yahoo!", in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discov. data Min. - KDD '09*, 2009, p. 1097.