Implementation of Bayesian Hierachical Clustering

Lina Yang, Xiaodi Qin

May 1, 2017

1 Abstract

Hierarchical clustering is commonly used in unsupervised learning. There are several limitations to the traditional hierarchical clustering, including no guide to pre-specify number of clusters, difficult to choose suitable distance metric, and lack of probability based criterion for model evaluation. To overcome these limitations, Heller and Ghahramani (2005) developed a Bayesian hierarchical clustering based on Dirichlet process mixtures. It uses marginal likelihoods and Bayesian hypothesis testing as criteria for merging cluster.

Key words: Bayesian Methods, unsupervised learning, hierarchical model, clustering analysis

2 Background

3 Algorithm

This algorithm calculates the probability of merging for each two subtrees, and build the whole tree bottom-up, and finally returns a matrix in the structure of input matrix for ‘dendrogram‘ function in ‘Scipy‘ module. It can be used to cluster data with multivariate normal distribution or Bernoulli distribution, and is possible to be extended to other types of data.

For each pair of clusters, the hypotheses we need to test are H_1 vs. H_2, representing either merge or stay unmerged. And the probability of being merged will be calculated to make the decision. This probability, denoted by r, is given by Bayes’ rule,

$$r = \frac{\pi \mathrm{P}(D \mid H_1)}{\mathrm{P}(D \mid T)}$$

where $\mathrm{P}(D \mid H_1)$ is the probability of the data under H_1, and it can be computed by distribution function of the data and corresponding conjugate prior. π is the prior that all data in the two clusters belong to one cluster, and $\mathrm{P}(D \mid T)$ is a weighed sum of the probability of the data under each hypothesis, which can be expressed as

$$\mathrm{P}(D_k \mid T_k) = \pi_k \mathrm{P}(D_k \mid H_1^k) + (1 - \pi_k)\mathrm{P}(D_i \mid T_i)\mathrm{P}(D_j \mid T_j)$$

Based on the equation above, $\mathrm{P}(D_k \mid T_k)$ has to be calculated recursively, since it uses probabilities of data under its subtrees to determine the probability of data in current tree. The cutoff for not merging is 0.5 because of only two hypotheses considered.

The algorithm basicly consists of three parts:

1. Line 1 4 is the initialization part. $\mathrm{P}(D \mid T)$'s are calculated given distritbution of data assigned to all the leaf nodes, $\mathbb{S}$ is a set of indices containing all the data clusters, and every time two clusters are merged togther, their indices would be removed from $\mathbb{S}$; every time a new cluster is built up, the newly assigned index associated with it would be added to $\mathbb{S}$. Family is a function defined elsewhere with two options, "niw" for multivariate normal data and "bb" for Bernouli.

2. Line 5 12 describe the process of obtaining the probabilities of merging for pairwise leaf nodes using the equation above, and each of the potential clusters have an index stored in $\mathbb{P}$. Similar to $\mathbb{S}$, $\mathbb{P}$ keeps adding/deleting indices to maintain a set of combinations of all current available clusters during building the whole tree.

3. Line 13 28 repeats the process of computing $\mathrm{P}(D \mid T)$ for every two possible clusters, until $\mathbb{S}$ runs out of indices. In each run, the combination of the highest merging probability would be append to a list Z, which is a 4dimensional matrix, within each row first two numbers being the indices of two sub-clusters that have been merged, third number being weight evaluated as reciprocal of log odds, and the last being the number of nodes in the cluster.

The code was checked by the clustering results against the ones in Heller and Ghahramani (2005), using similar data. Unfortunately so far there is no other package implementing the same algorithm to compare with, the only one found on Bioconductor is for multinomial and time-series data, which was not included in our implementation.

The performance of this method comparing with other clustering methods was evaluated using a *purity score*, which will be elaborated in Section **??**.

Algorithm 1: Bayesian Hierachical Clustering

```
Input: Data $X = (X_0, X_1, ..., X_N)$, $family \in \{niw\}$, hyperparameter $\alpha$, scaling factor on
       the prior precision of the mean $r$.
Output: A linkage matrix $Z$
Set $\mathbb{S} = \emptyset$
For $l \in \{0, 1, \ldots, N-1\}$:
    $n_l^0 = 1$, $d_l^0 = \alpha$, $X_l^0 = X_l$, $ml_l = \textsc{family}(X_l^0)$
    $\mathbb{S} = \mathbb{S} \cup \{l\}$
Set $t = 0$, $\mathbb{P} = \emptyset$
For $i \in \{0, 1, \ldots, N-2\}$:
    For $j \in \{i+1, \ldots, N-1\}$:
        $c_{1,t} = i$, $c_{2,t} = j$, $n_t = n_i^0 + n_j^0$
        $X_t = (X_i^0, X_j^0)^T$, $d_t = \alpha\Gamma(n_t) + d_i^0 d_j^0$
        $P_{1,t} = \textsc{family}(X_t)\alpha\Gamma(n_t)/d_t$, $P_{2,t} = ml_i ml_j (d_i^0 d_j^0 / d_t)$
        $logodds_t = \log P_{1,t} - \log P_{2,t}$
        $\mathbb{P} = \mathbb{P} \cup \{t\}$, $t = t + 1$
Set $p = 0$, $Z = []$
While $1$:
    $idx = \arg\max_{idx \in \mathbb{P}} logodds$
    $Z.\textsc{append}([c_{1,idx}, c_{2,idx}, logodds_{idx}, n_{idx}])$
    $n_{N+p}^0 = n_{idx}$, $X_{N+p}^0 = X_{idx}$, $d_{N+p}^0 = d_{idx}$, $ml_{N+p} = P_{1,idx} + P_{2,idx}$
    $rm = \{c_{1,idx}, c_{2,idx}\}$, $\mathbb{S} = \mathbb{S} \setminus rm$
    If $\mathbb{S} = \emptyset$:
        break
    For $q \in \mathbb{S}$:
        $c_{1,t} = N + p$, $c_{2,t} = q$, $n_t = n_{N+p}^0 + n_q^0$
        $X_t = (X_{N+p}^0, X_q^0)^T$, $d_t = \alpha\Gamma(n_t) + d_{N+p}^0 d_q^0$
        $P_{1,t} = \textsc{family}(X_t)\alpha\Gamma(n_t)/d_t$, $P_{2,t} = ml_{N+p} ml_q (d_{N+p}^0 d_q^0 / d_t)$
        $logodds_t = \log P_{1,t} - \log P_{2,t}$
        $\mathbb{P} = \mathbb{P} \cup \{t\}$, $t = t + 1$
    $\mathbb{P} = \mathbb{P} \setminus \{r : c_{1,r} \in rm \vee c_{2,r} \in rm\}$
    $\mathbb{S} = \mathbb{S} \cup \{N + p\}$, $p = p + 1$
return $Z$
```

4 Optimization

5 Application

5.1 Simulated data sets

5.2 Real data sets

aggregationGionis et al. (2007) spiral Chang and Yeung (2004)

6 Comparative analysis

7 Discussion

We wrote a function to calculate the purity score which is the evaluation method used in Heller's BHC paper. The purity score calculates as follows: Pick two leaves i, j uniformly at random with restriction that i, j are from same class, and then find the smallest subtree containing i and j. Find all leaves that locate in this subtree and calculate the proportion of leaves that are from the same class as i (j) in the subtree. This proportion is the purity score we need. We compared this purity score calculated from BHC to 3 kinds of traditional agglomerative clustering using average, single, and complete linkage. We tested their performance on synthetic data, as well as real binary data from CEDAR dataset and real multivariate datasets obtained from other publications.

	SINGLE	COMPLETE	AVERAGE	BHC
SYNTHETIC_multivariate	0.689	0.6	0.689	1.0
Aggregation	0.85	1.0	1.0	0.788
Spiral	1.0	0.332	0.334	0.35
SYNTHETIC_binary	0.641	0.561	0.608	0.681
CEDAR	0.572	0.73	0.863	0.978

8 Code

The repository can be found at `https://github.com/qxxxd/bhc`.

9 References

Chang, H. and D.-Y. Yeung (2004). Robust path-based clustering for the unsupervised and semi-supervised learning settings.

Gionis, A., H. Mannila, and P. Tsaparas (2007, March). Clustering Aggregation. *ACM Trans. Knowl. Discov. Data 1*(1).

Heller, K. A. and Z. Ghahramani (2005). Bayesian Hierarchical Clustering. In *Proceedings of the 22Nd International Conference on Machine Learning*, ICML '05, New York, NY, USA, pp. 297–304. ACM.