COURSE PROJECT

An Efficient Hierarchical Clustering Algorithm via Non-parametric Bayesian Approach

Beining G. Wu[1]

[1]*Department of Statistics and Finance*, *University of Science and Technology of China, Hefei, China.*

ABSTRACT
Clustering is an important type of unsupervised learning problem. In this course project, we present a modified hierarchical clustering algorithm [Heller and Ghahramani, 2005], which involves the the probabilistic model an Bayesian method. It differs from the traditional clustering algorithm as it involves a Bayesian hypothesis testing procedure to merge data. And it performs an approximate inference for the Dirichlet Process mixture model, which provides a lower bound for the marginal likelihood. We describe the procedure of this algorithm and use it to analyze a real world data.

KEYWORDS
Nonparametric Bayesian Analysis, Clustering Analysis, Dirichlet Process, Hierarchical Clustering.

1. Introduction

The clustering problem is a basic unsupervised learning problem, and hierarchical method is probably one of the most frequently used method solving this problem. Basically, this kind of method organize the data in tree structures, and one would hope that this hierarchical structure would provide some information on the data clustering.

Traditional hierarchical clustering algorithm [Cooke et al., 2011] is a basic bottom-up agglomerative procedure, which merges "the closest" pairs at each stage, and then iteratively organize the data into a tree structure.

However, both the traditional hierarchical algorithm and other methods (K-means, spectral clustering) are too restrictive. In this article, we follow an efficient modified hierarchical clustering algorithm [Heller and Ghahramani, 2005], which involves Bayesian method. This method can be proved to have some advantages.

In this project, we begin with a basic review of traditional clustering algorithm in section 2. Then we describe the main algorithm in section 3. Theoretical supplements and of the algorithm is provided in section 4. Finally, we compare our Bayesian hierarchical method with the traditional clustering algorithm in section 5.

This is the course project for the Bayesian Analysis, Fall 2021, which is lectured by Professor Weiping Zhang.
Email to: andrewwu@mail.ustc.edu.cn

2. Review of Clustering Algorithm

2.1. K-means

The K-means algorithm [Hartigan and Wong, 1979] is one of the most classical iterative descent method which is applicable when all the variables are quantitative. Let $D = \{x_1, x_2, \ldots, x_N\}$ be the data sets, where all the $x_i \in \mathbb{R}^d$ are unlabelled numeric data and K is the user-specified number of clusters. The algorithm iteratively cluster the data into K clusters through minimizing the following object function:

$$V = \sum_{i=1}^{K} \sum_{x_j \in C_i} d(x_j, \mu_i),$$

where C_i is each cluster and μ_i is the arithmetic mean of the data, or the center in each cluster. To begin with, the algorithm randomly specify the center of these K clusters and assign each data point to the closest cluster. Then the algorithm iteratively calculate the center after assignment, and then re-assign the data to the adjusted clusters until convergence. This cost function has several local minima and therefore several re-runs are required to obtain a cluster.

2.2. Spectral Clustering

The spectral clustering algorithm [Shi and Malik, 2000] and [Ng et al., 2002] is an alternative clustering method which involves the spectral method. Here, a spectral decomposition is performed to the similarity matrix whose entries are the distance or other similarity metric. Then we can choose the top few eigenvectors and project the data to the linear spans of these eigenvectors. Once the projection is done, one performs a traditional clustering (K-means or other methods mentioned below) to cluster these data.

2.3. Hierarchical Clustering

The hierarchical clustering differs greatly from the aforementioned methods of clustering. The goal here is not to find a single partitioning of the data, instead the algorithm finds the hierarchy of the data, which is usually represented as a tree. This hierarchical structure provides us with more interesting structures and we can perform partition procedures in multiple levels.

This hierarchical algorithm has two types. The first one is agglomerative, which merges separated data at each iteration. And the other one is divisive, where the data is in one group at the beginning and are partitioned finer in each iteration. Here we introduce this traditional algorithm which organize the data in a hierarchical way.

Both these clustering algorithms possess drawbacks. The K-means algorithm has one fatal disadvantage that one has to specify the number of clusters, which itself is a hypothesis testing problem. For the hierarchical clustering algorithm, one fatal limitation is that we don't know at what level should we prune the tree to obtain a practical clustering. This can be improved with involving probabilistic models, which provides criterion whether the data is well organized. Moreover, probabilistic models make it possible to make predictions based on the observed data. We can use the observed data to train our model parameters via optimization and use this model to predict future data.

Algorithm 1: Bottom-up Hierarchical Clustering

Input: Data set $D = \{x_1, x_2, \ldots, x_N\}$ and metric d
Initialize: Trivial subtree $D_i = \{x_i\}$ and number of iteration c;
while $c > 1$ **do**
  Find the minimal distance pair of sub-trees, i.e.

$$(i_0, j_0) = \arg\min d(D_i, D_j)$$

  Merge $D_k = D_{i_0} \cup D_{j_0}$, Delete D_{i_0}, D_{j_0} and $c \leftarrow c - 1$.
end
Output: A hierarchical tree structure.

3. Bayesian Hierarchical Clustering: Algorithm

In this section, we introduce an improved hierarchical method, which involves the Bayesian method. It's also a bottom-up agglomerative method of merging data, like the traditional hierarchical clustering. However, this modified algorithm involves Bayesian probabilistic model. Briefly speaking, this algorithm would perform a Bayesian hypothesis testing in considering a potential merge. The algorithm iteratively calculate the posterior merging probability and perform a merge on the data pair with the highest merging probability.

We now describe this algorithm in a detailed manner. Assume that the numeric data set $D = \{x_1, x_2, \ldots, x_N\} \subset \mathbb{R}^d$ is given. We initialize the algorithm with trivial sub-trees $\{T_i\}_{i=1}^N$, where each tree contains exactly one point $\mathcal{D}_i = \{x_i\}$. At each stage, the algorithm traverse through all the pairs of existing trees (not including the subtrees and choose one pair to perform the merge. At each merge process, the algorithm assigns new tree $T_k = (T_i, T_j)$, merging data $\mathcal{D}_k = \mathcal{D}_i \cup \mathcal{D}_j$, and delete the $\mathcal{D}_i$, $\mathcal{D}_j$.

Here we go deeper into the selection process. For each pair of existing groups of data D_i and D_j, we consider the potential merge $\mathcal{D}_k = \mathcal{D}_i \cup \mathcal{D}_j$ and denote $\mathcal{H}_1^k$ as the hypothesis that the all the data from group $\mathcal{D}_k$ are from one parametric distribution density $p(x|\theta)$, as opposed to generated from a mixed density $\sum \pi_i p(x|\theta_i)$. Under Bayesian scheme we place a prior $p(\theta|\beta)$ to the parameters. Therefore we can then calculate the probability of the data group $\mathcal{D}_k$ under this hypothesis as

$$p(\mathcal{D}_k|\mathcal{H}_1^k) = \int \prod_{x_i \in \mathcal{D}_k} p(x_i|\theta) p(\theta|\beta) d\theta.$$

The alternative hypothesis $\mathcal{H}_2^k$ is that the data in $\mathcal{D}_k$ are from different clusters. Instead of summing up over the exponentially many partitions which is often intractable, the algorithm only calculates probability of tree-consistent partitions as an approximate way. Precisely, we have

$$p(\mathcal{D}_k|\mathcal{H}_2^k) = p(\mathcal{D}_i|T_i) p(\mathcal{D}_j|T_j),$$

where $p(\mathcal{D}_i|T_i)$ is the probability of the dataset under a tree. We make further interpretation of this term below. If the data in $\mathcal{D}_i$ are from the same cluster, which means the hypothesis $\mathcal{H}_1^i$ is true, then it's naturally tree-consistent. Otherwise it suffices to let the left and right subtrees

to be tree consistent. So we define this term recursively as follows

$$p(\mathcal{D}_k|T_k) = \pi_k p(\mathcal{D}_k|\mathcal{H}_1^k) + (1-\pi_k)p(\mathcal{D}_i|T_i)p(\mathcal{D}_j|T_j),$$

where $\pi_k = p(\mathcal{H}_1^k)$.

With all these in hand, we can now consider the posterior probability that $\mathcal{H}_1^k$ is true as simple application of Bayesian rule:

$$p(\mathcal{H}_k^1|\mathcal{D}_k) = \frac{\pi_k p(\mathcal{D}_k|\mathcal{H}_1^k)}{\pi_k p(\mathcal{D}_k|\mathcal{H}_1^k) + (1-\pi_k)p(\mathcal{D}_i|T_i)p(\mathcal{D}_j|T_j)}.$$

We denote this quantity as $r_k = p(\mathcal{H}_k^1|\mathcal{D}_k$, and choose the largest one to perform the merge.

We present our final algorithm here, which may looks quite simple

Algorithm 2: Bayesian Hierarchical Clustering

Input: Data $\mathcal{D} = \{x_1, x_2, \ldots, x_N\}$, model $p(x|\theta)$, prior $p(\theta|\beta)$

Initialize: number of clusters $c = n$, and $\mathcal{D}_i = \{x_i\}$;

while *c>1* **do**

For each pair $\mathcal{D}_i$ and $\mathcal{D})_j$ compute the posterior merging probability

$$r_k = \frac{\pi_k p(\mathcal{D}_k|\mathcal{H}_1^k)}{p(\mathcal{D}_k|T_k}.$$

And find the pair $\mathcal{D}_i$ and $\mathcal{D}_j$ with maximal value.;

Merge $T_k \leftarrow (T_i, T_j)$ and $\mathcal{D}_k \leftarrow \mathcal{D}_i \cup \mathcal{D}_j$.

end

Output: A Bayesian mixture model

4. Theory of Bayesian Hierarchical Clustering

We still have some problems that remain. Note that in the computation of posterior merging probability r_k we actually use π_k, the prior merging probability. But we're only given the prior on the models' parameter, which is insufficient to calculate the prior merging probability.

Here is where the Dirichlet process mixture model comes in.

4.1. Dirichlet Process Mixture Models

To begin with, we briefly review the Dirichlet process mixture model, which is our generative probability model in the Bayesian hierarchical clustering model. Consider a finite mixture model with K components

$$p(x|\phi) = \sum_{i=1}^{K} p(s=i|\mathbf{p})p(x|\theta_j).$$

Here x is the data, s is the label of this data and $\phi = (\mathbf{p}, \theta)$ is the model parameters. Often we would require θ to have conjugate prior, for example, if θ is the Gaussian location parameter

then we would require the prior on θ to be Gaussian. The prior on the multinomial parameter $\mathbf{p}$ is the conjugate Dirichlet distribution, namely

$$p(\mathbf{p}|\alpha) = \frac{\Gamma(\alpha)}{\Gamma(\alpha/K)^K} \prod_{i=1}^{K} p_i^{\alpha/K-1}.$$

Therefore, the marginal probability of the data can be formulated as

$$p(\mathcal{D}|\alpha, \beta) = \sum_{\mathbf{s}} p(\mathbf{s}|\alpha) p(\mathcal{D}|\beta, \mathbf{s}).$$

Here the sum is over all possible labels. All these quantity can be well defined through a limit process, in which $K \to \infty$. Unrigorously speaking, the Dirichlet process is exactly limiting K to obtain a prior, which the number of components K is random and can be arbitrarily large. This extension removed the drawbacks in the K-means, where the number of model components should primarily specified by the user.

With this prior, we are able to compute the prior merging probability, namely π_k. Let $\mathcal{D}_k$ be the potential dataset (which is merged from other), and $n_k = |\mathcal{D}_k|$. Then the prior merging probability, which is also the prior probability for $\mathcal{H}_1^k$ is relative probability of assigning all the data in $\mathcal{D}_k$ to one components versus the probability of all other tree consistent partition. In principle we have the following

Algorithm 3: Algorithm for computing prior merging probability

Initialize: For each leaf nodes i, set $d_i = \alpha$, $\pi_i = 1.$;
foreach *internal node* k **do**
  compute $d_k = \alpha\Gamma(n_k) + d_{\text{left}} d_{\text{right}}$;
  $\pi_k = \alpha\Gamma(n_k)/d_k$
end

4.2. Properties of Approximate Inference

We've mentioned before that our algorithm computes the posterior probability of the alternate hypothesis in an approximate way. As a matter of fact, we can prove that this approximation is valid to some extent.

We first calculate the exact inference quantity of Dirichlet process mixture model.

Lemma 4.1. *The marginal likelihood of data in the DPM is*

$$p(\mathcal{D}_k) = \sum_{v \in \mathcal{V}} \frac{\alpha^{m_v} \prod_{l=1}^{m_v} \Gamma(n_{l,v})}{\Gamma(n_k + \alpha)/\Gamma(\alpha)} \prod_{l=1}^{m_v} p(\mathcal{D}_l^v).$$

where $\mathcal{V}$ is the collection of all possible partitions of $\mathcal{D}_k$ and m_v is the number of the clusters in partitioning v, $b_{l,b}$ is the number of points in cluster l of partitioning v, abd $\mathcal{D}_l^v$ are the data points in cluster l of partitioning v.

This lemma can be easily shown in the following manner

$$p(\mathcal{D}_k) = \sum_{v \in \mathcal{V}} p(v) p(\mathcal{D}^v),$$

Where $p(v)$ corresponds to the prior on the partitioning v and $p(\mathcal{D}_l^v)$ corresponds to the likelihood of the partitioning under the data. Precisely, we can formulate them as follows,

$$p(v) = \frac{\alpha^{m_v} \prod_{l=1}^{m_v} \Gamma(n_{l,v})}{\Gamma(n_k + \alpha)/\Gamma(\alpha)}.$$

and

$$p(\mathcal{D}^v) = \prod_{l=1}^{m_v} p(\mathcal{D}_l^v).$$

Now we turn to the quantity $p(\mathcal{D}_k|T_k)$ in our approximation algorithm.

Theorem 4.2. *The quantity $p(\mathcal{D}_k|T_k)$ computed in our Bayesian Hierarchical Clustering algorithm is*

$$p(\mathcal{D}_k|T_k) = \sum_{v \in \mathcal{V}_{T_k}} \frac{\alpha^{m_v} \prod_{l=1}^{m_v} \Gamma(n_{l,v})}{d_k} \prod_{l=1}^{m_v} p(\mathcal{D}_l^v), \tag{1}$$

where $\mathcal{V}_{T_k}$ is the collection of tree-consistent partitions of $\mathcal{D}_k$ under T_k.

Proof. The proof of this result is through a inductive reasoning. By the computation in Bayesian hierarchical clustering, we have

$$p(\mathcal{D}_k|T_k) = p(\mathcal{D}_k|\mathcal{H}_1^k) \frac{\alpha \Gamma(n_k)}{d_k} + p(\mathcal{D}_i|T_i) p(\mathcal{D}_j|T_j) \frac{d_i d_j}{d_k}.$$

In this equation, the required quantity is computed recursively. So we naturally consider using an induction. First we consider the leaf nodes, which don't have subtrees, and $n_k = 1, d_k = \alpha$. This implies that

$$p(\mathcal{D}_k|T_k) = p(\mathcal{D}_k|\mathcal{H}_1^k).$$

By the inductive hypothesis, we have equation 1 true for the sub-trees T_i and T_j. And ■

Combining last two results to get the following

Corollary 4.3. *For any binary tree T_k with the data points in $\mathcal{D}_k$ as its leaves, we have following lower bound for the marginal likelihood of a Dirichlet process mixture model:*

$$\frac{d_k \Gamma(\alpha)}{\Gamma(n_k + \alpha)} p(\mathcal{D}_k|T_k) \leqslant p(\mathcal{D}_k).$$

4.3. Computational Complexity

For now we have asserted that our approximation algorithm is practical. Moreover, the following proposition asserts that our algorithm is efficient.

Proposition 4.4. *The number of the tree-consistent partitions is exponential in the number of the data points for the balanced binary tree.*

Proof. Assume that our balanced binary tree is of depth l, then the number of the data is of $O(2^l)$. On the other hand, assume that T_i has C_i tree-consistent partitions of $\mathcal{D}_i$ and T_j has C_j, then the number of the tree-consistent partitions for $T_k = (T_i, T_j)$ of data $\mathcal{D}_k$ is $C_i C_j + 1$. For each leaf nodes $C_i = 1$, these implies that the tree-consistent partitions of the whole tree grows in $O(2^{2^l}) \approx O(2^n)$. ■

5. Application to Real Data

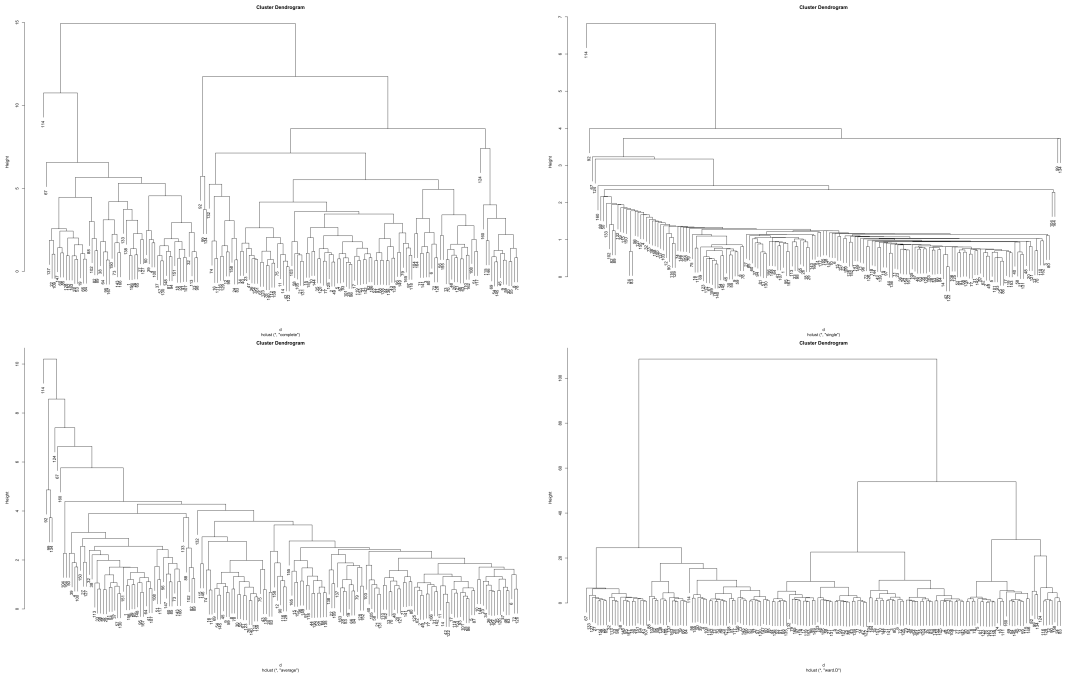
In this section we use the aforementioned clustering method to study a real world data. Here the data we use is country socio-economic data from Kaggle. This dataset stores the social-economic index and health factors of each country. Our object is to categorize these countries for the NGO to make practical aids.

The whole dataset contains 167 countries and 9 features and we list them as follows

- child_mort: Death of children under 5 years of age per 1000 live births.
- exports: Exports of goods and services per capita. Given as %age of the GDP per capita.
- health: Total health spending per capita. Given as %age of GDP per capita.
- imports: Imports of goods and services per capita. Given as %age of the GDP per capita.
- income: Net income per person.
- inflation: The measurement of the annual growth rate of the Total GDP.
- life_expec: The average number of years a new born child would live if the current mortality patterns are to remain the same.
- total_fer: The number of children that would be born to each woman if the current age-fertility rates remain the same.
- gdpp: The GDP per capita. Calculated as the Total GDP divided by the total population.

This clustering problem doesn't have a real version of labels, and the number of the clusters are unspecified. We begin with the traditional hierarchical clustering.

Figure 1.: Traditional Hierarchical Clustering Result



These four figures are the dendrogram output of the tradition hierarchical clustering algorithm, with different metric function.

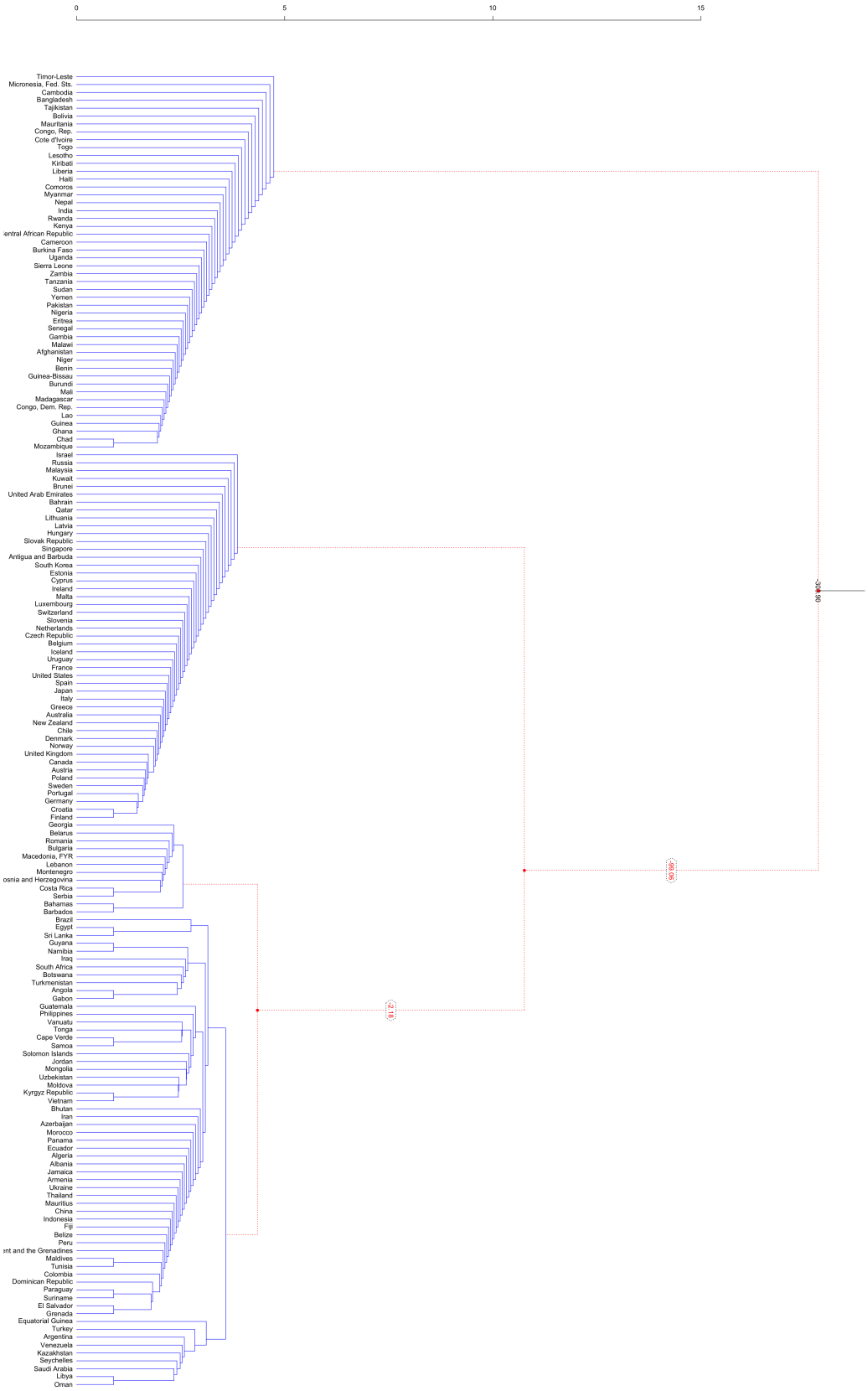
These dendrogram implies that the structures of these trees are very disordered. Different metric would actually result in very different tree structures. And this disordered tree structure makes the clusters unidentifiable. Indeed, different levels of cut would also cause the clustering results to be different, this leads to the intractability. Intuitively, the Ward's method gives the best, or at least most organized dendrogram. The agglomerative coefficients are listed below.

Table 1.: Agglomerative Coefficients

	Single Linkage	Average Linkage	Complete Linkage	Ward's Distance
AC	0.843	0.875	0.913	0.952

Now let's consider the Bayesian hierarchical clustering to be used in this data. From the dendrogram plotted above and below, we see that the BHC algorithm creates a more organized tree structure. This advantage is more significant in the higher level.

Figure 2.: Bayesian Hierarchical Clustering Result



6. Conclusion

In this course project article, we have presented a novel hierarchical clustering algorithm using Bayesian hypothesis testing and non-parametric Bayesian models. We stated that it can approximately perform inference on the Dirichlet process mixture meanwhile has polynomial time complexity, which outperforms the exact inference that summing up over exponentially many partitions. Finally, we use an example to show that the BHC algorithm could provide a more organized, hence tractable hierarchical structures.

References

[Cooke et al., 2011] Cooke, E. J., Savage, R. S., Kirk, P. D., Darkins, R., and Wild, D. L. (2011). Bayesian hierarchical clustering for microarray time series data with replicates and outlier measurements. *BMC Bioinformatics*, 12(1):399.

[Hartigan and Wong, 1979] Hartigan, J. A. and Wong, M. A. (1979). Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108.

[Heller and Ghahramani, 2005] Heller, K. A. and Ghahramani, Z. (2005). Bayesian hierarchical clustering. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML '05, page 297–304, New York, NY, USA. Association for Computing Machinery.

[Ng et al., 2002] Ng, A., Jordan, M., and Weiss, Y. (2002). On spectral clustering: Analysis and an algorithm. In Dietterich, T., Becker, S., and Ghahramani, Z., editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press.

[Shi and Malik, 2000] Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905.

Appendix A. R Codes for Experiments

```
library(cluster)

# Read data and preprocessing
data <- read.csv(file="Country-data.csv",header=TRUE)
pure_data <- data[,-1]
labels <- data[,1]

pure_data <- scale(pure_data)
d <- dist(pure_data)

# Traditional Hierarchical Clusteing Methods
# Use four different subset metric
hc_complete <- hclust(d, method = "complete")
hc_single <- hclust(d, method = "single")
hc_average <- hclust(d, method = "average")
hc_ward <- hclust(d, method = "ward.D")

# Computing the agglomerative coefficients.
ac <- c(
```

```
    agnes(pure_data, method = "complete")$ac,
    agnes(pure_data, method = "single")$ac,
    agnes(pure_data, method = "average")$ac,
    agnes(pure_data, method = "ward")$ac)

    ac
    par(mfrow=c(2,2))
    plot(hc_complete)
    plot(hc_single)
    plot(hc_average)
    plot(hc_ward)
    # Bayesian Hierarchical Clustering
    percentiles <- FindOptimalBinning(pure_data, itemLabels, transposeData=TRUE, verbos

    ##
    ## DATA DISCRETISATION
    ## -------------------
    ## Percentiles: 0.1 0.8 0.1
    ## We have the following parameters for the data array:
    ## nGenes:       167
    ## nExperiments: 9
    ## ***Please check that these are the right way round! (it affects the discretisati
    ##
    ## Discretisation logEvidence: -386.981758529484
    ## (Need to add this to the model logEvidence)
    ## -------------------
    ## [1] Running Bayesian Hierarchical Clustering....
    ## [1] "DataType: multinomial"
    ## [1] Optimising global hyperparameter...
    ## [1]   782.5754 -1073.2366
    ## [1]  1265.925 -1072.931
    ## [1]  1564.651 -1072.837
    ## [1]  1806.560 -1072.783
    ## [1]  1714.159 -1072.802
    ## [1]  1898.782 -1072.766
    ## [1]  1955.778 -1072.757
    ## [1]  1991.004 -1072.751
    ## [1]  2012.774 -1072.748
    ## [1]  2026.229 -1072.745
    ## [1]  2034.545 -1072.744
    ## [1]  2039.684 -1072.743
    ## [1]  2042.861 -1072.743
    ## [1]  2044.824 -1072.743
    ## [1]  2046.037 -1072.742
    ## [1]  2046.787 -1072.742
    ## [1]  2047.250 -1072.742
    ## [1]  2047.584 -1072.742
    ## [1]  2047.584 -1072.742
    ## [1]  2047.584 -1072.742
```

```
## [1] Hyperparameter: 2047.58354290636
## [1] Lower bound on overall LogEvidence: -1.0727e+03
## [1] ********************
##
## DATA DISCRETISATION
## --------------------
## Percentiles: 0.15 0.7 0.15
## We have the following parameters for the data array:
## nGenes:       167
## nExperiments: 9
## ***Please check that these are the right way round! (it affects the discretisati
##
## Discretisation logEvidence: -386.981758529484
## (Need to add this to the model logEvidence)
## --------------------
## [1] Running Bayesian Hierarchical Clustering....
## [1] "DataType: multinomial"
## [1] Optimising global hyperparameter...
## [1]   782.5754 -1073.2366
## [1]  1265.925 -1072.931
## [1]  1564.651 -1072.837
## [1]  1806.560 -1072.783
## [1]  1714.159 -1072.802
## [1]  1898.782 -1072.766
## [1]  1955.778 -1072.757
## [1]  1991.004 -1072.751
## [1]  2012.774 -1072.748
## [1]  2026.229 -1072.745
## [1]  2034.545 -1072.744
## [1]  2039.684 -1072.743
## [1]  2042.861 -1072.743
## [1]  2044.824 -1072.743
## [1]  2046.037 -1072.742
## [1]  2046.787 -1072.742
## [1]  2047.250 -1072.742
## [1]  2047.584 -1072.742
## [1]  2047.584 -1072.742
## [1]  2047.584 -1072.742
## [1] Hyperparameter: 2047.58354290636
## [1] Lower bound on overall LogEvidence: -1.0727e+03
## [1] ********************
##
## DATA DISCRETISATION
## --------------------
## Percentiles: 0.2 0.6 0.2
## We have the following parameters for the data array:
## nGenes:       167
## nExperiments: 9
## ***Please check that these are the right way round! (it affects the discretisati
##
```

```
## Discretisation logEvidence: 347.054111859041
## (Need to add this to the model logEvidence)
## -------------------
## [1] Running Bayesian Hierarchical Clustering....
## [1] "DataType: multinomial"
## [1] Optimising global hyperparameter...
## [1]     3.364745 -1513.910937
## [1]     5.135255 -1514.610152
## [1]     2.27051 -1514.01470
## [1]     3.031412 -1513.919341
## [1]     3.698078 -1513.904230
## [1]     4.247031 -1514.006035
## [1]     3.698078 -1513.904230
## [1]     3.698078 -1513.904230
## [1] Hyperparameter: 3.69807846784773
## [1] Lower bound on overall LogEvidence: -1.5139e+03
## [1] ********************
##
## DATA DISCRETISATION
## -------------------
## Percentiles: 0.25 0.5 0.25
## We have the following parameters for the data array:
## nGenes:       167
## nExperiments: 9
## ***Please check that these are the right way round! (it affects the discretisati
##
## Discretisation logEvidence: 347.054111859041
## (Need to add this to the model logEvidence)
## -------------------
## [1] Running Bayesian Hierarchical Clustering....
## [1] "DataType: multinomial"
## [1] Optimising global hyperparameter...
## [1]     3.364745 -1513.910937
## [1]     5.135255 -1514.610152
## [1]     2.27051 -1514.01470
## [1]     3.031412 -1513.919341
## [1]     3.698078 -1513.904230
## [1]     4.247031 -1514.006035
## [1]     3.698078 -1513.904230
## [1]     3.698078 -1513.904230
## [1] Hyperparameter: 3.69807846784773
## [1] Lower bound on overall LogEvidence: -1.5139e+03
## [1] ********************
##
## DATA DISCRETISATION
## -------------------
## Percentiles: 0.3 0.4 0.3
## We have the following parameters for the data array:
## nGenes:       167
## nExperiments: 9
```

```
## ***Please check that these are the right way round! (it affects the discretisati
##
## Discretisation logEvidence: 614.612889295938
## (Need to add this to the model logEvidence)
## -------------------
## [1] Running Bayesian Hierarchical Clustering....
## [1] "DataType: multinomial"
## [1] Optimising global hyperparameter...
## [1]     1.836881 -1624.231927
## [1]     2.663119 -1625.795809
## [1]     1.326238 -1628.742610
## [1]     2.170214 -1624.384703
## [1]     1.836881 -1624.231927
## [1]     1.836881 -1624.231927
## [1] Hyperparameter: 1.83688103937537
## [1] Lower bound on overall LogEvidence: -1.6242e+03
## [1] ********************
##
## DATA DISCRETISATION
## -------------------
## Percentiles: 0.35 0.3 0.35
## We have the following parameters for the data array:
## nGenes:       167
## nExperiments: 9
## ***Please check that these are the right way round! (it affects the discretisati
##
## Discretisation logEvidence: 614.612889295938
## (Need to add this to the model logEvidence)
## -------------------
## [1] Running Bayesian Hierarchical Clustering....
## [1] "DataType: multinomial"
## [1] Optimising global hyperparameter...
## [1]     1.836881 -1624.231927
## [1]     2.663119 -1625.795809
## [1]     1.326238 -1628.742610
## [1]     2.170214 -1624.384703
## [1]     1.836881 -1624.231927
## [1]     1.836881 -1624.231927
## [1] Hyperparameter: 1.83688103937537
## [1] Lower bound on overall LogEvidence: -1.6242e+03
## [1] ********************
##
## DATA DISCRETISATION
## -------------------
## Percentiles: 0.31 0.38 0.31
## We have the following parameters for the data array:
## nGenes:       167
## nExperiments: 9
## ***Please check that these are the right way round! (it affects the discretisati
##
```

```
## Discretisation logEvidence: 614.612889295938
## (Need to add this to the model logEvidence)
## -------------------
## [1] Running Bayesian Hierarchical Clustering....
## [1] "DataType: multinomial"
## [1] Optimising global hyperparameter...
## [1]     1.836881 -1624.231927
## [1]     2.663119 -1625.795809
## [1]     1.326238 -1628.742610
## [1]     2.170214 -1624.384703
## [1]     1.836881 -1624.231927
## [1]     1.836881 -1624.231927
## [1] Hyperparameter: 1.83688103937537
## [1] Lower bound on overall LogEvidence: -1.6242e+03
## [1] ********************
##
## DATA DISCRETISATION
## -------------------
## Percentiles: 0.32 0.36 0.32
## We have the following parameters for the data array:
## nGenes:       167
## nExperiments: 9
## ***Please check that these are the right way round! (it affects the discretisati
##
## Discretisation logEvidence: 614.612889295938
## (Need to add this to the model logEvidence)
## -------------------
## [1] Running Bayesian Hierarchical Clustering....
## [1] "DataType: multinomial"
## [1] Optimising global hyperparameter...
## [1]     1.836881 -1624.231927
## [1]     2.663119 -1625.795809
## [1]     1.326238 -1628.742610
## [1]     2.170214 -1624.384703
## [1]     1.836881 -1624.231927
## [1]     1.836881 -1624.231927
## [1] Hyperparameter: 1.83688103937537
## [1] Lower bound on overall LogEvidence: -1.6242e+03
## [1] ********************
##
## DATA DISCRETISATION
## -------------------
## Percentiles: 0.33 0.34 0.33
## We have the following parameters for the data array:
## nGenes:       167
## nExperiments: 9
## ***Please check that these are the right way round! (it affects the discretisati
##
## Discretisation logEvidence: 614.612889295938
## (Need to add this to the model logEvidence)
```

```
## -------------------
## [1] Running Bayesian Hierarchical Clustering....
## [1] "DataType: multinomial"
## [1] Optimising global hyperparameter...
## [1]     1.836881 -1624.231927
## [1]     2.663119 -1625.795809
## [1]     1.326238 -1628.742610
## [1]     2.170214 -1624.384703
## [1]     1.836881 -1624.231927
## [1]     1.836881 -1624.231927
## [1] Hyperparameter: 1.83688103937537
## [1] Lower bound on overall LogEvidence: -1.6242e+03
## [1] ********************
##
## DATA DISCRETISATION
## -------------------
## Percentiles: 0.34 0.32 0.34
## We have the following parameters for the data array:
## nGenes:       167
## nExperiments: 9
## ***Please check that these are the right way round! (it affects the discretisati
##
## Discretisation logEvidence: 614.612889295938
## (Need to add this to the model logEvidence)
## -------------------
## [1] Running Bayesian Hierarchical Clustering....
## [1] "DataType: multinomial"
## [1] Optimising global hyperparameter...
## [1]     1.836881 -1624.231927
## [1]     2.663119 -1625.795809
## [1]     1.326238 -1628.742610
## [1]     2.170214 -1624.384703
## [1]     1.836881 -1624.231927
## [1]     1.836881 -1624.231927
## [1] Hyperparameter: 1.83688103937537
## [1] Lower bound on overall LogEvidence: -1.6242e+03
## [1] ********************
##
## DATA DISCRETISATION
## -------------------
## Percentiles: 0.35 0.3 0.35
## We have the following parameters for the data array:
## nGenes:       167
## nExperiments: 9
## ***Please check that these are the right way round! (it affects the discretisati
##
## Discretisation logEvidence: 614.612889295938
## (Need to add this to the model logEvidence)
## -------------------
## [1] Running Bayesian Hierarchical Clustering....
```

```
## [1] "DataType: multinomial"
## [1] Optimising global hyperparameter...
## [1]     1.836881 -1624.231927
## [1]     2.663119 -1625.795809
## [1]     1.326238 -1628.742610
## [1]     2.170214 -1624.384703
## [1]     1.836881 -1624.231927
## [1]     1.836881 -1624.231927
## [1] Hyperparameter: 1.83688103937537
## [1] Lower bound on overall LogEvidence: -1.6242e+03
## [1] ********************
##
## DATA DISCRETISATION
## --------------------
## Percentiles: 0.36 0.28 0.36
## We have the following parameters for the data array:
## nGenes:       167
## nExperiments: 9
## ***Please check that these are the right way round! (it affects the discretisati
##
## Discretisation logEvidence: 614.612889295938
## (Need to add this to the model logEvidence)
## --------------------
## [1] Running Bayesian Hierarchical Clustering....
## [1] "DataType: multinomial"
## [1] Optimising global hyperparameter...
## [1]     1.836881 -1624.231927
## [1]     2.663119 -1625.795809
## [1]     1.326238 -1628.742610
## [1]     2.170214 -1624.384703
## [1]     1.836881 -1624.231927
## [1]     1.836881 -1624.231927
## [1] Hyperparameter: 1.83688103937537
## [1] Lower bound on overall LogEvidence: -1.6242e+03
## [1] ********************
##
## DATA DISCRETISATION
## --------------------
## Percentiles: 0.37 0.26 0.37
## We have the following parameters for the data array:
## nGenes:       167
## nExperiments: 9
## ***Please check that these are the right way round! (it affects the discretisati
##
## Discretisation logEvidence: 614.612889295938
## (Need to add this to the model logEvidence)
## --------------------
## [1] Running Bayesian Hierarchical Clustering....
## [1] "DataType: multinomial"
## [1] Optimising global hyperparameter...
```

```
## [1]     1.836881 -1624.231927
## [1]     2.663119 -1625.795809
## [1]     1.326238 -1628.742610
## [1]     2.170214 -1624.384703
## [1]     1.836881 -1624.231927
## [1]     1.836881 -1624.231927
## [1] Hyperparameter: 1.83688103937537
## [1] Lower bound on overall LogEvidence: -1.6242e+03
## [1] ********************
##
## DATA DISCRETISATION
## --------------------
## Percentiles: 0.38 0.24 0.38
## We have the following parameters for the data array:
## nGenes:       167
## nExperiments: 9
## ***Please check that these are the right way round! (it affects the discretisati
##
## Discretisation logEvidence: 614.612889295938
## (Need to add this to the model logEvidence)
## --------------------
## [1] Running Bayesian Hierarchical Clustering....
## [1] "DataType: multinomial"
## [1] Optimising global hyperparameter...
## [1]     1.836881 -1624.231927
## [1]     2.663119 -1625.795809
## [1]     1.326238 -1628.742610
## [1]     2.170214 -1624.384703
## [1]     1.836881 -1624.231927
## [1]     1.836881 -1624.231927
## [1] Hyperparameter: 1.83688103937537
## [1] Lower bound on overall LogEvidence: -1.6242e+03
## [1] ********************
##
## DATA DISCRETISATION
## --------------------
## Percentiles: 0.39 0.22 0.39
## We have the following parameters for the data array:
## nGenes:       167
## nExperiments: 9
## ***Please check that these are the right way round! (it affects the discretisati
##
## Discretisation logEvidence: 353.181055796486
## (Need to add this to the model logEvidence)
## --------------------
## [1] Running Bayesian Hierarchical Clustering....
## [1] "DataType: multinomial"
## [1] Optimising global hyperparameter...
## [1]     1.836881 -1426.400132
## [1]     2.663119 -1429.847196
```

```
## [1]     1.326238 -1426.343835
## [1]     0.9929046 -1430.4346794
## [1]     1.326238 -1426.343835
## [1]     1.326238 -1426.343835
## [1] Hyperparameter: 1.32623792124926
## [1] Lower bound on overall LogEvidence: -1.4263e+03
## [1] ********************
##
## OPTIMISED DISCRETISATION
## ------------------------
## Percentiles: 0.35 0.3 0.35
## LogEvidence: -1009.619
```

```
discreteData <- DiscretiseData(t(pure_data), percentiles=percentiles)
```

```
##
## DATA DISCRETISATION
## -------------------
## Percentiles: 0.35 0.3 0.35
## We have the following parameters for the data array:
## nGenes:       9
## nExperiments: 167
## ***Please check that these are the right way round! (it affects the discretisati
##
## Discretisation logEvidence: -293.679718723953
## (Need to add this to the model logEvidence)
## -------------------
```

```
discreteData <- t(discreteData)
hc3 <- bhc(discreteData, labels, verbose=TRUE)
```

```
## [1] Running Bayesian Hierarchical Clustering....
## [1] "DataType: multinomial"
## [1] Optimising global hyperparameter...
## [1]     1.836881 -1396.013621
## [1]     2.663119 -1398.999740
## [1]     1.326238 -1395.072094
## [1]     0.9929046 -1407.3113011
## [1]     1.326238 -1395.072094
## [1]     1.326238 -1395.072094
## [1] Hyperparameter: 1.32623792124926
## [1] Lower bound on overall LogEvidence: -1.3951e+03
## [1] ********************
```

```
plot(hc3)
```