TSSB-DW: Two-level clustering analysis

Yinqiao Yan

Last updated on 22/05/2021

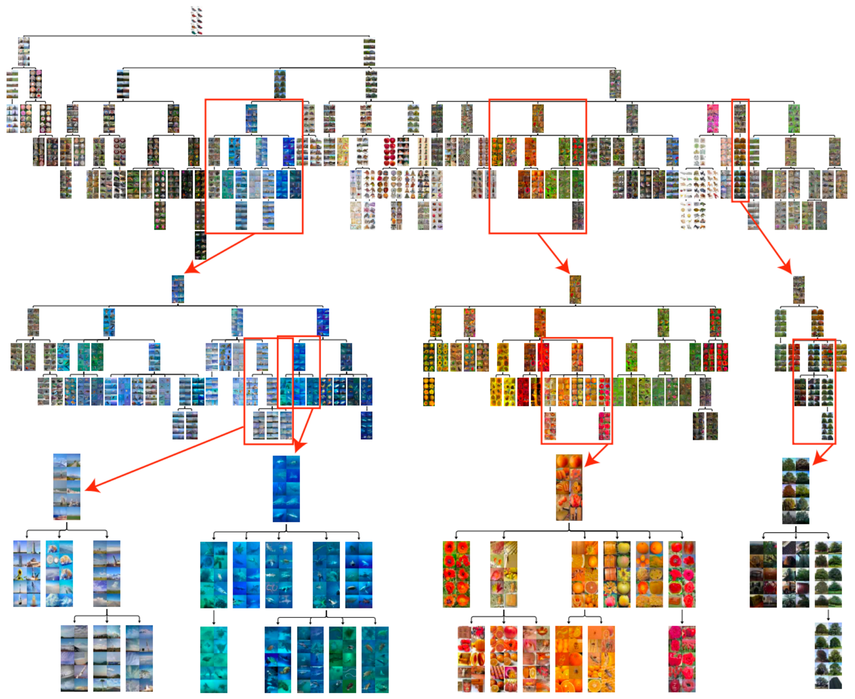
Contents

1 Introduction

Clustering is one of the most important unsupervised learning problem. For example, in single-cell analysis, cells are grouped into several cell types based on the cell profile data. This is really useful when we hope to analyze the potential various cell types existed in the original tissue, or explore some new types. In image clustering problems, images are clustered based on some features (extracted from PCA or deep neural networks) to find some underlying relationships or patterns behind the images in the same cluster.

Recently, tree-structured clustering methods are getting more attention in *relationship reconstruction problem among subclones*. We need to simultaneously cluster the data into several groups and reconstruct the underlying relationship among these groups. Adams et al. (2010) proposed a novel nonparametric Bayesian prior named tree-structured stick-breaking prior (TSSB). They extended the stick-breaking process of DP to a two-dimension case. This method has been implemented in tumor subclone phylogeny reconstruction problem (Deshwar et al., 2015; Yuan et al., 2015). TSSB has infinite width and depth, and each datum can locate in any internal node of the tree. In (Adams et al.2010), TSSB was applied to image hierarchical clustering problem (CIFAR-100). They firstly extracted the 256 binary features $x \in \{0, 1\}^{256}$ from a DNN, and then used the factored Bernoulli likelihood at each node. Note that the factored likelihood is based on the assumption that the elements of the data are independent. This setting can avoid the high dimensionality curse.

$$f\left(x_n \mid \theta_\epsilon\right) = \prod_{d=1}^{256}\left(1+\exp\left\{-\theta_\epsilon^{(d)}\right\}\right)^{-x_n^{(d)}}\left(1+\exp\left\{\theta_\epsilon^{(d)}\right\}\right)^{1-x_n^{(d)}}$$



2 Preliminaries

Adams et al. (2010) proposed a nonparametric Bayesian prior to model the underlying tree structure behind the different clusters, which is referred to as tree-structured stick-breaking process (TSSB). Recently, TSSB has been applied in some hierarchical clustering problem, such as image clustering (Adams et al. 2010) and tumor phylogeny reconstruction (Yuan et al. 2015, Deshwar et al. 2015). This prior has infinite depth and width and each datum can live at any internal node of the tree, not only the leaf nodes (different from the hierarchical clustering method).

2.1 DP and stick breaking process

Brief definition of Dirichlet process (DP).

- A DP is a **random probability measure** G defined on the probability space S (prior on the space of probability measure, $\mathcal{M}(S)$). Denoted by $G \sim DP(\alpha, H)$.
- For measurable **finite** partition $\{B_1, ..., B_k\}$ of S, the joint distribution of the vector $(G(B_1), \ldots, G(B_k))$ is the **Dirichlet distribution** with parameters $(\alpha H(B_1), \ldots, \alpha H(B_k))$.

Stick breaking construction of DP (equivalent definition).

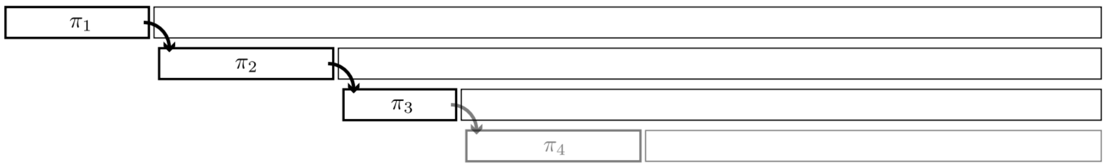
DP is almost surely discrete probability measure (natural property for clustering). It can be written as (Sethuraman 1994)

$$G(\cdot) = \sum_{k=1}^{\infty} \pi_k \delta_{Z_k}(\cdot)$$

- $\delta_{Z_k}(\cdot)$ denote a discrete measure concentrated at $Z_k \overset{\text{iid}}{\sim} H$.
- π_k are random weights, chosen to be independent of Z_k and satisfy $\sum_{k=1}^{N} \pi_k = 1$.

- $\pi_1 = V_1$ and $\pi_k = (1-V_1)(1-V_2)\cdots(1-V_{k-1})V_k, \quad k \geq 2.$
- $V_k \overset{\text{iid}}{\sim} \mathrm{Be}(1,\alpha).$

We can also denote $\;= (\pi_1, \pi_2, ...) \sim \mathrm{GEM}(\alpha)$, which is named after Griffiths, Engen, and McCloskey.



(a) Dirichlet process stick breaking

Truncation of DP.

Ishwaran and James (2001) designed a block Gibbs sampler based on truncation version of stick-breaking process, denoted by

$$\sum_{k=1}^{N} \pi_k \delta_{Z_k}(\cdot), \quad N < \infty$$

where

- $\pi_1 = V_1, \quad \pi_k = V_k \prod_{j<k}(1-V_j), \quad k = 2, ..., N.$
- Set $V_N = 1$ to guarantee $\sum_{k=1}^{N} \pi_k = 1$ with probability 1.

Motivation of truncation.

1. Limitations of Polya urn Gibbs scheme (marginalizing) for infinite measure:
   - will lead to slowly mixing Markov chain (slowly converges to stationary distribution)
   - one-at-a-time updates
   - hard to solve non-conjugate cases
   - difficult to find a unified sampling framework (*Griffin (2016) pointed out the unavailability of a suitable Polya urn scheme for some priors.*)
2. Block Gibbs sampler:
   - simpler and more efficient
   - unified Gibbs framework
   - parallel computing (update a block of parameters each time)

Non-parametric mixture model. In (Ishwaran and James, 2001), the mixture model is given by

$$\begin{aligned}(X_i \mid Y_i) &\overset{\text{ind}}{\sim} \pi(X_i \mid Y_i), \quad i = 1, \ldots, n,\\ (Y_i \mid P) &\overset{\text{iid}}{\sim} P\\ P &\sim \mathcal{P}_N.\end{aligned}$$

~~To implement a Polya urn scheme sampler, integrate out the nonparametric prior.~~ (*Not recommended*)

To implement a block Gibbs sampler, introduce a index variable K. (*Data augmentation*)

$$\begin{aligned}(X_i \mid \mathbf{Z}, \mathbf{K}) &\overset{\text{ind}}{\sim} \pi\left(X_i \mid Z_{K_i}\right), \quad i = 1, \ldots, n\\ (K_i \mid \mathbf{p}) &\overset{\text{iid}}{\sim} \sum_{k=1}^{N} p_k \delta_k(\cdot), \quad N < \infty\\ (\mathbf{p}, \mathbf{Z}) &\sim \pi(\mathbf{p}) \times H^N(\mathbf{Z}),\end{aligned}$$

where

- $Y_i = Z_{K_i},\ Z_k \overset{\text{iid}}{\sim} H,\ k = 1, ..., N.$
- $\mathbf{p} = (p_1, ..., p_N) \sim \text{GEM}_N(\alpha)$ (belongs to generalized dirichlet distribution - *won't go into details here*)
- K_i are i.i.d and can be updated independently (parallel computing).

Block Gibbs sampler.

Update parameters **in blocks**.

$$(\mathbf{Z} \mid \mathbf{K}, \mathbf{X}),$$
$$(\mathbf{K} \mid \mathbf{Z}, \mathbf{p}, \mathbf{X}),$$
$$(\mathbf{p} \mid \mathbf{K}),$$

2.2 Tree-structured stick-breaking process

Extended to two breaking processes.

Adams et al. (2010) proposed this two-dimensional stick-breaking process, referred to as tree-structured stick-breaking process (TSSB).

- ν-breaks: decides the weight for staying at a node.

$$v_\epsilon \sim \text{Be}(1, \alpha(|\epsilon|))$$

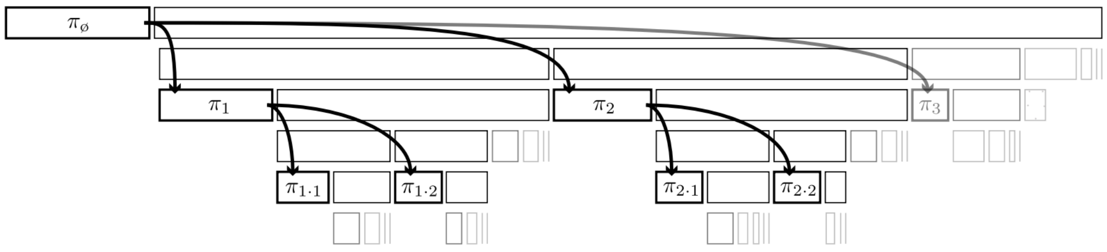
  (Adams et al. suggested $\alpha(j) = \lambda^j \alpha_0$.)

- ψ-breaks: decides the weight for selecting a child node to descend.

$$\psi_\epsilon \sim \text{Be}(1, \gamma)$$

The random weights are then given by

$$\pi_\epsilon = \nu_\epsilon \varphi_\epsilon \prod_{\epsilon' \prec \epsilon} \varphi_{\epsilon'} (1 - \nu_{\epsilon'}) \qquad \varphi_{\epsilon\epsilon_i} = \psi_{\epsilon\epsilon_i} \prod_{j=1}^{\epsilon_i - 1} (1 - \psi_{\epsilon j}) \qquad \pi_\emptyset = \nu_\emptyset,$$



(b) Tree-structured stick breaking

Hierarchical priors for node parameters.

- In DP, Z_k are i.i.d. from the base measure H.
- In TSSB, the i.i.d. assumption on the node parameters is inappropriate, since the random weights have a hierarchical structure. Hence we need a **transition distribution** for the node parameters.

$$T(\theta_\epsilon \leftarrow \theta_{pa(\epsilon)})$$

  where $\theta_{pa(\epsilon)}$ denote the parent node of θ_ϵ.

3 Our model

Motivated by the work of Ishwaran and James (2001), in this report we propose a truncation version of TSSB, referred to as TSSB-DW (**TSSB** with finite **D**epth and **W**idth). Factored normal likelihood is used to avoid the high-dimensionality problem, and the prior of node parameters θ_ε and σ^2_ε are the most commonly used conjugate prior Normal-invGamma distribution. The hyper-parameter λ (sometimes called the drift parameter in $T(\theta_\epsilon \leftarrow \theta_{pa(\epsilon)})$) has inverse Gamma distribution. The hyper-parameter $\eta_{\mathcal{N}}$ and η_Θ are fixed.

$$\begin{aligned}
(X_i \mid \theta, \Sigma, c_i = \varepsilon) &\overset{\text{ind}}{\sim} \prod_{\ell=1}^{L} N\left(X_i^\ell \mid \theta_\varepsilon^\ell, \eta_{\mathcal{N}}^{|\varepsilon|}\sigma_\varepsilon^{2\ell}\right), \quad i = 1, ..., n \\
c_i \mid \pi &\overset{\text{iid}}{\sim} \sum_\varepsilon \pi_\varepsilon \delta_\varepsilon \\
\pi &\sim \text{TSSB-DW}(\alpha_0, \rho, \gamma) \\
\theta_\emptyset^\ell &\sim N(\theta_\emptyset^\ell \mid \mu_0^\ell, \lambda^\ell), \quad \ell = 1, ..., L \\
\theta_\varepsilon^\ell \mid \theta_{pa(\varepsilon)}^\ell, \eta_\Theta &\overset{\text{iid}}{\sim} N(\theta_\varepsilon^\ell \mid \theta_{pa(\varepsilon)}, \eta_\Theta^{|\varepsilon|}\lambda^\ell) \\
\sigma_\varepsilon^{2\ell} &\overset{\text{iid}}{\sim} \text{InvGamma}(v_{sig}, s_{sig}) \\
\lambda^\ell &\overset{\text{iid}}{\sim} \text{InvGamma}(v_{dft}, s_{dft})
\end{aligned}$$

Another option of the prior of drift is uniform distribution.

$$\lambda^\ell \overset{\text{iid}}{\sim} \text{Unif}\,(\min, \max)$$

3.1 Posterior inference

The posterior inference includes the following steps.

Conditional for $\theta_\varepsilon^\ell,\ \ell = 1, ..., L$

$$\begin{aligned}
p\left(\theta_\varepsilon^\ell \mid -\right) &\propto N\left(\theta_\varepsilon^\ell \mid \theta_{pa(\varepsilon)}^\ell, \eta_\Theta^{|\varepsilon|}\lambda^\ell\right) \prod_{ch(\varepsilon)} N\left(\theta_{ch(\varepsilon)}^\ell \mid \theta_\varepsilon^\ell, \eta_\Theta^{|\varepsilon|+1}\lambda^\ell\right) \prod_{\{i: c_i = \varepsilon\}} N\left(X_i^\ell \mid \theta_\varepsilon^\ell, \eta_{\mathcal{N}}^{|\varepsilon|}\sigma_\varepsilon^{2\ell}\right) \\
&\propto N(\mu_\varepsilon^\ell, \tau_\varepsilon^\ell)
\end{aligned}$$

where

$$\begin{aligned}
\mu_\varepsilon^\ell &= \left(\frac{N_\varepsilon \bar{X}_\varepsilon^\ell}{\eta_{\mathcal{N}}^{|\varepsilon|}\sigma_\varepsilon^{2\ell}} + \frac{W\bar{\theta}_{ch(\varepsilon)}^{\ell(t-1)} + \eta_\Theta \theta_{pa(\varepsilon)}^{\ell(t)}}{\eta_\Theta^{|\varepsilon|+1}\lambda^\ell}\right)\left(\frac{N_\varepsilon}{\eta_{\mathcal{N}}^{|\varepsilon|}\sigma_\varepsilon^{2\ell}} + \frac{W + \eta_\Theta}{\eta_\Theta^{|\varepsilon|+1}\lambda^\ell}\right)^{-1}, \\
\tau_\varepsilon^\ell &= \left(\frac{N_\varepsilon}{\eta_{\mathcal{N}}^{|\varepsilon|}\sigma_\varepsilon^{2\ell}} + \frac{W + \eta_\Theta}{\eta_\Theta^{|\varepsilon|+1}\lambda^\ell}\right)^{-1}.
\end{aligned}$$

and N_ε is the number of data stopping at node ε. If $N_\varepsilon = 0$, then

$$\mu_\varepsilon^\ell = \frac{W\bar{\theta}_{ch(\varepsilon)}^{\ell(t-1)} + \eta_\Theta \theta_{pa(\varepsilon)}^{\ell(t)}}{W + \eta_\Theta}, \quad \tau_\varepsilon^\ell = \frac{\eta_\Theta^{|\varepsilon|+1}\lambda^\ell}{W + \eta_\Theta}$$

Conditional for $\sigma_\varepsilon^{2\ell},\ \ell = 1, ..., L$

$$\begin{aligned}
p\left(\sigma_\varepsilon^{2\ell} \mid -\right) &\propto \prod_{\{i: c_i = \varepsilon\}} N\left(X_i^\ell \mid \theta_\varepsilon^\ell, \eta_{\mathcal{N}}^{|\varepsilon|}\sigma_\varepsilon^{2\ell}\right) \text{InvGamma}\left(v_{sig}, s_{sig}\right) \\
&\propto \text{InvGamma}\left(\tilde{v}_{sig}^\ell, \tilde{s}_{sig}^\ell\right)
\end{aligned}$$

where

$$\tilde{v}^{\ell}_{sig} = v_{sig} + N_{\varepsilon}/2$$
$$\tilde{s}^{\ell}_{sig} = s_{sig} + \frac{1}{2\eta_{\mathcal{N}}^{|\varepsilon|}} \sum_{\{i:c_i=\varepsilon\}} \left(X^{\ell}_i - \theta^{\ell}_{\varepsilon}\right)^2$$

Note that if there is no datum at the current node (i.e., $N_{\varepsilon} = 0$), then v, s do not change. The density of inverse Gamma distribution InvGamma(v, s) is

$$p(x) = \frac{s^v}{\Gamma(v)} x^{-(v+1)} \exp\left(-\frac{s}{x}\right)$$

where $v > 0$ and $s > 0$ are called shape and scale parameter respectively.

Conditional for c

$$p\left(c_i \mid \theta, \Sigma, \pi, X\right) \overset{\text{ind}}{\sim} \sum_{\varepsilon} \beta^{(i)}_{\varepsilon} \delta_{\varepsilon}$$

where

$$\beta^{(i)}_{\varepsilon} \propto \pi_{\varepsilon} \prod_{\ell=1}^{L} N\left(X^{\ell}_i \mid \theta^{\ell}_{\varepsilon}, \eta_{\mathcal{N}}^{|\varepsilon|} \sigma^{2\ell}_{\varepsilon}\right)$$

Conditional for π

The key is to update ν and ψ, and π_{ε} are given by

$$\pi_{\varepsilon} = \nu^*_{\varepsilon} \psi^*_{\varepsilon} \prod_{\varepsilon' \prec \varepsilon} \psi^*_{\varepsilon'} \left(1 - \nu^*_{\varepsilon'}\right)$$

where

$$\nu^*_{\varepsilon} \overset{\text{ind}}{\sim} \text{Be}\left(1 + N_{\varepsilon},\ \alpha(|\varepsilon|) + N_{\varepsilon \prec \cdot}\right)$$
$$\psi^*_{\varepsilon e_i} \overset{\text{ind}}{\sim} \text{Be}(1 + N_{\varepsilon e_i \preccurlyeq \cdot},\ \gamma + \sum_{j > e_i} N_{\varepsilon j \preccurlyeq \cdot})$$

$N_{\varepsilon} = \#\{i : c_i = \varepsilon\}$ is the number of data stopping at the node ε, $N_{\varepsilon \prec \cdot}$ is "the number of data that come down this path but does not stop at ε", and $N_{\varepsilon \preccurlyeq \cdot}$ is equal to $N_{\varepsilon} + N_{\varepsilon \prec \cdot}$.

Conditional for the hyperparameter drift $\lambda^{\ell},\ \ell = 1, ..., L$

$$p(\lambda^{\ell} \mid -) \propto \prod_{\varepsilon} N\left(\theta^{\ell}_{\varepsilon} \mid \theta^{\ell}_{pa(\varepsilon)}, \eta_{\Theta}^{|\varepsilon|} \lambda^{\ell}\right) \text{InvGamma}\left(v_{dft}, s_{dft}\right)$$
$$\propto \text{InvGamma}\left(\tilde{v}^{\ell}_{dft}, \tilde{s}^{\ell}_{dft}\right)$$

where

$$\tilde{v}^{\ell}_{dft} = v_{dft} + N_{nodes}/2, \quad N_{nodes} = 1 + \cdots + W^D = \frac{W^{D+1} - 1}{W - 1}$$
$$\tilde{s}^{\ell}_{dft} = s_{dft} + \sum_{\varepsilon} \frac{1}{2\eta_{\Theta}^{|\varepsilon|}} (\theta^{\ell}_{\varepsilon} - \theta^{\ell}_{pa(\varepsilon)})^2$$

Note that when $\varepsilon = \emptyset$, then $\theta_{pa(\varepsilon)}$ is equal to the initial mean μ_0. (since the prior of $\theta_{\emptyset}$ also includes the drift)

Conditional for the hyperparameters α_0, ρ, γ

In (Adams et al.2010), the authors used the slice sampler to update these hyperparameters. This is slice sampler-within-Gibbs framework.

$$
p\left(\alpha_0, \lambda \mid \{\nu_\epsilon\}\right) \propto \mathbb{I}\left(\alpha_0^{\min} < \alpha_0 < \alpha_0^{\max}\right) \mathbb{I}\left(\lambda^{\min} < \lambda < \lambda^{\max}\right) \prod_\epsilon \mathrm{Be}\left(\nu_\epsilon \mid 1, \lambda^{|\epsilon|}\alpha_0\right)
$$
$$
p\left(\gamma \mid \{\psi_\epsilon\}\right) \propto \mathbb{I}\left(\gamma^{\min} < \gamma < \gamma^{\max}\right) \prod_\epsilon \mathrm{Be}\left(\psi_\epsilon \mid 1, \gamma\right)
$$

Search for tree structure

In (Yuan et al. 2015), the authors added another swap-nodes step to propose a new tree structure. They switched the data Ids, node parameters (θ_ε and σ_ε^2) and ν-breaks ν_ε of the two chosen nodes. The proposal would be accepted if the new unnormalized posterior is large than the old one. Here we follow the idea in (Yuan et al. 2015).

4 Simulation study

4.1 Prepare data

Data: 7 classes normal data. Separable. Dims: 700 × 2 dims

Import the packages we will need.

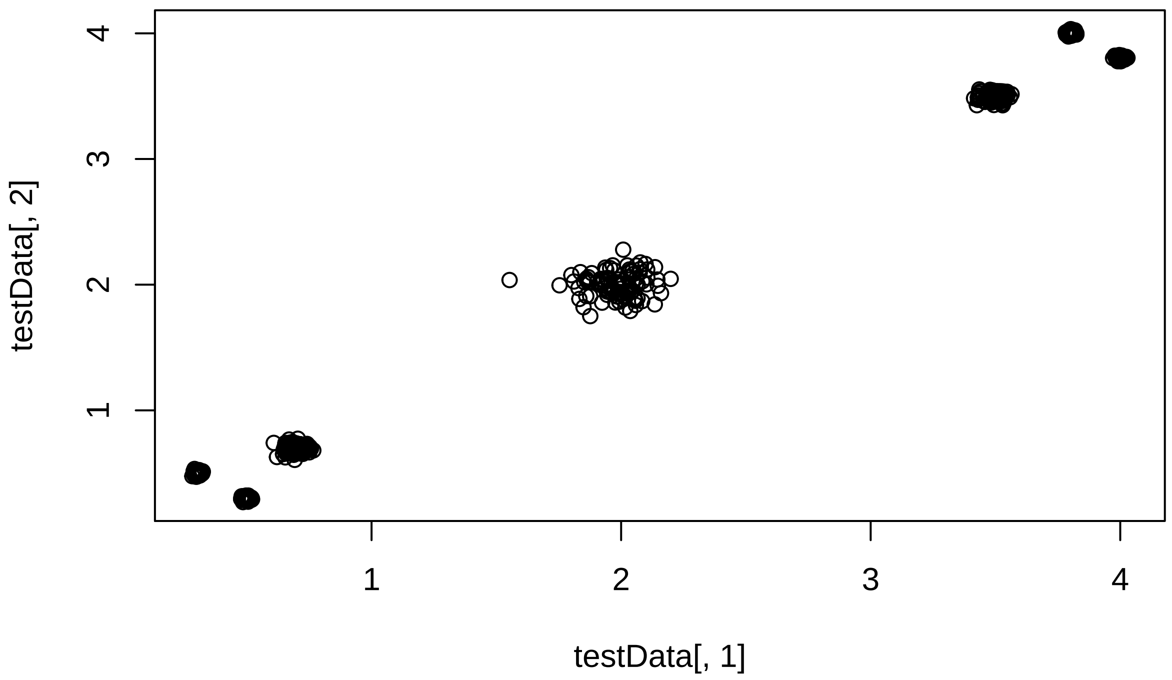
```r
library(igraph)
library(ggplot2)
library(networkD3)
```

In this study, the data matrix are 700 × 2. There are seven classes, each of which has 100 data.

```r
set.seed(24)

m <- 700
dims <- 2
testData <- rbind(rmvnorm(m/7, mean = rep(2.0, dims), sigma = diag(0.10^2, dims, dims)),
                  rmvnorm(m/7, mean = rep(3.5, dims), sigma = diag(0.03^2, dims, dims)),
                  rmvnorm(m/7, mean = c(3.8, 4.0), sigma = diag(0.01^2, dims, dims)),
                  rmvnorm(m/7, mean = c(4.0, 3.8), sigma = diag(0.01^2, dims, dims)),

                  rmvnorm(m/7, mean = rep(0.7, dims), sigma = diag(0.03^2, dims, dims)),
                  rmvnorm(m/7, mean = c(0.5, 0.3), sigma = diag(0.01^2, dims, dims)),
                  rmvnorm(m/7, mean = c(0.3, 0.5), sigma = diag(0.01^2, dims, dims))
)
numOfData = nrow(testData)
plot(testData[,1], testData[,2])
```



4.2 Initialization and training

The settings of implementing the model are:

- $\eta_{\mathcal{N}} = 1$ and $\eta_{\Theta} = 0.5$
- Update order: i) Node parameters, ii) Data assignments, iii) Swap nodes
- burnIn = 100, Iter = 1000
- maxDepth = 3, maxWidth = 3
- "OnlyTree"

The drift λ is assigned the **inverse Gamma** prior. Then we **initialize** the Root Node and the corresponding Tssb.

```
empCov <- cov(t(testData))
empCov2 <- cov(testData)
priorSigmaScale = mean(diag(empCov))
priorDriftScale = min(diag(empCov2))

minDepth  <- 0
maxDepth  <- 3
maxWidth  <- 3
etaNormal <- 1
etaTheta  <- 0.3
Flag.onlyTree <- T

q0 <- Normal_DW_Factored_eta_ST$new(priorSigmaScale = priorSigmaScale,
                               priorDriftScale = priorDriftScale,
                               etaNormal = etaNormal, etaTheta = etaTheta,
                               dataDims = ncol(testData))

tssbMCMC <- TssbMCMC_DW_Factored_eta_ST$new(q0, data = testData,
                               dpAlpha = 1, dpGamma = 1, dpLambda = 1,
                               maxDepth = maxDepth, minDepth = minDepth,
                                maxWidth = maxWidth,
                                Flag.onlyTree = Flag.onlyTree)
#> Initialization: D = 3 and W = 3
#> No. 1 child of root is initializing...
#> No. 2 child of root is initializing...
#> No. 3 child of root is initializing...
#> Initialization Is Over!
```

After initialization, we update all the parameters using Gibbs sampler.

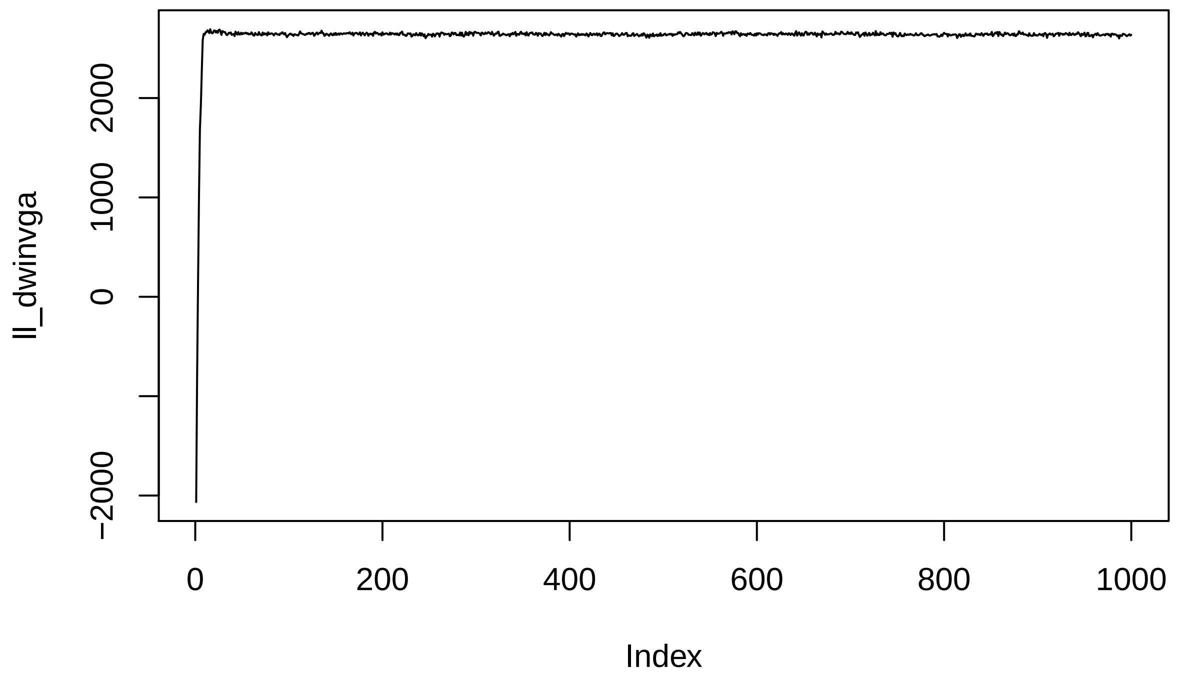
```
#> ==== iter:  0
#> Update node params:  0.01616788
#> Update drift:  0.01362896
#> Update sticks:  0.02428007
#> Update 3 hypers:  0.08781195
#> Update assignments:  0.785902
#> Swap-nodes move:  0.03925395
#> Update Keep tree:  0.01178002
#> ==== iter:  500
#> Update node params:  0.002918005
#> Update drift:  0.0006561279
#> Update sticks:  0.000510931
#> Update 3 hypers:  0.0009548664
#> Update assignments:  0.5217822
#> Swap-nodes move:  0.02447009
#> Update Keep tree:  0.0008749962
#> ==== iter:  1000
#> Update node params:  0.002606869
#> Update drift:  0.0006458759
#> Update sticks:  0.0004270077
#> Update 3 hypers:  0.000772953
#> Update assignments:  0.5405581
#> Swap-nodes move:  0.02488208
#> Update Keep tree:  0.0008718967
```

The total execution time is

```
#> Time difference of 8.715345 mins
```

4.3 Results

we plot the log-likelihood curve to demonstrate the convergence of the Markov chain.

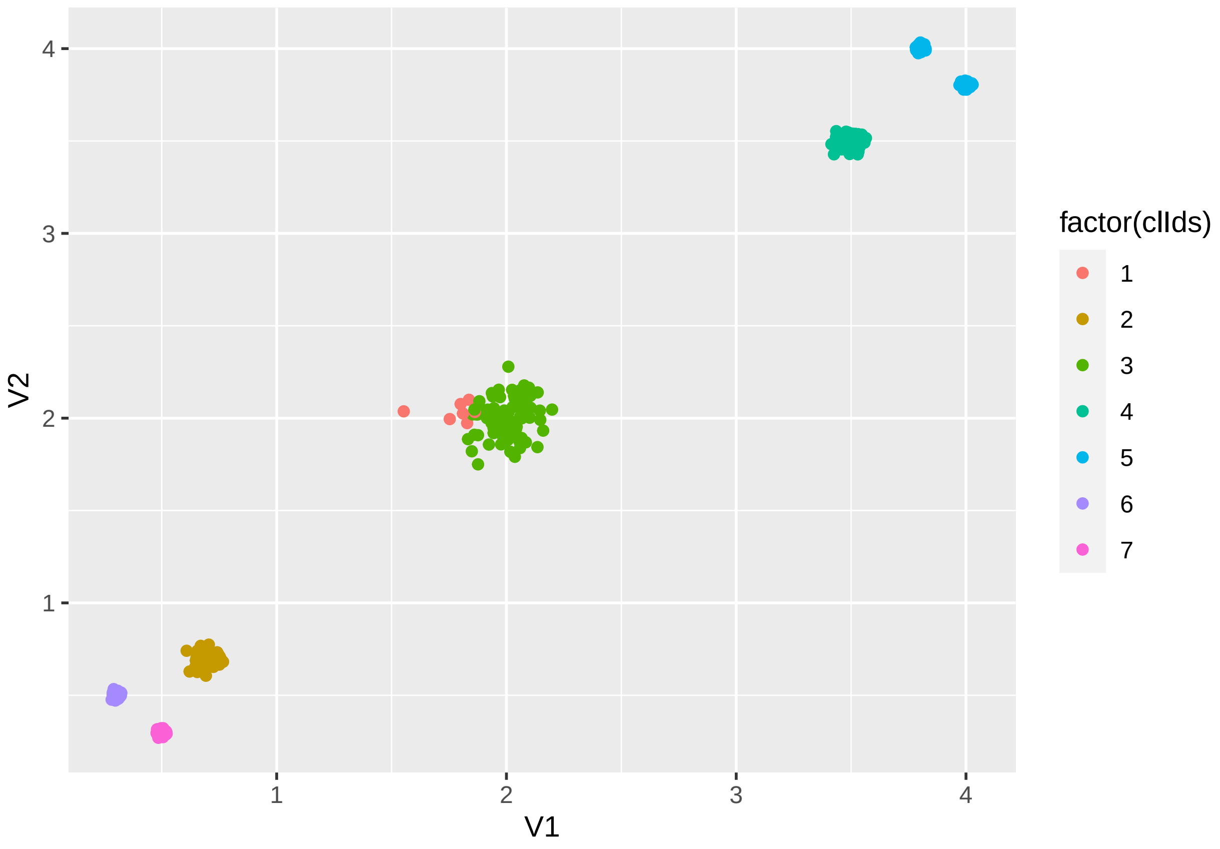


Finally, we plot the tree structure result using `forceNetwork` function in `networkD3` package.

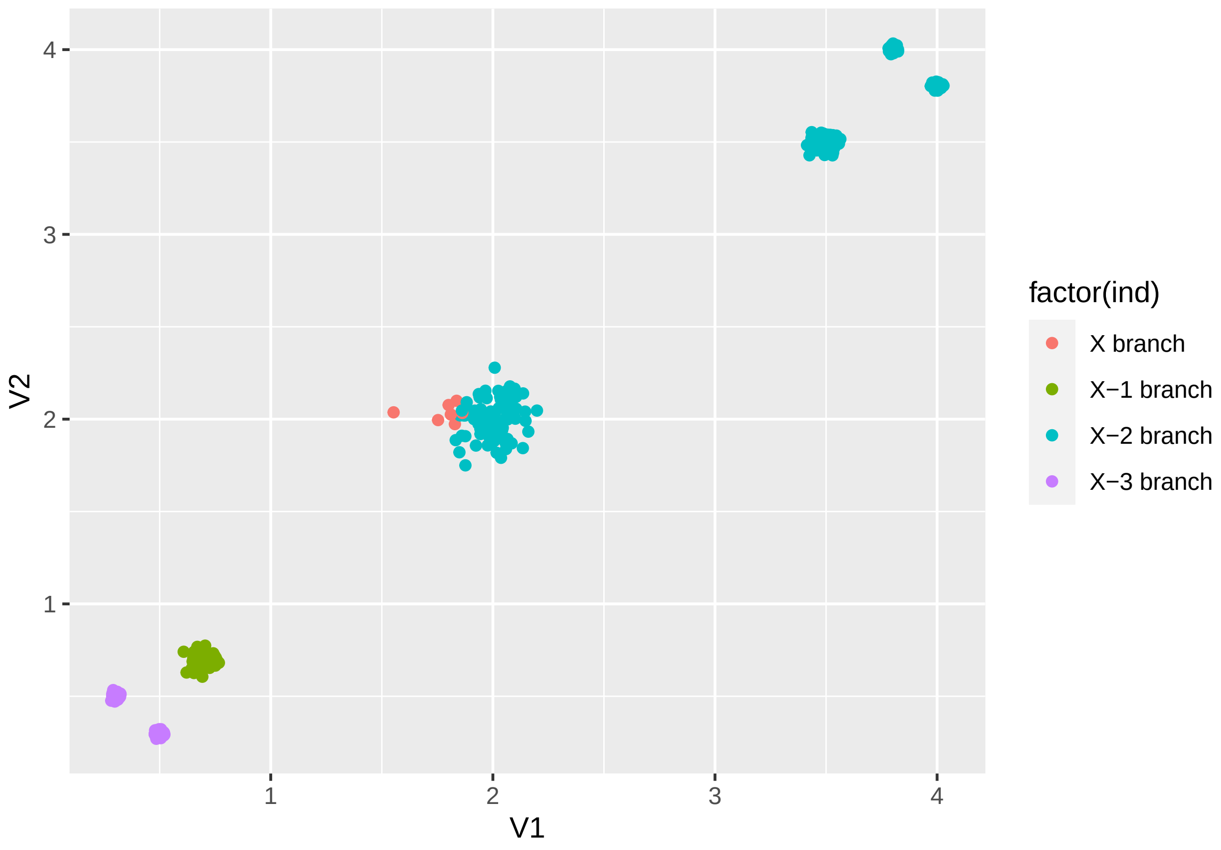
Select a single tree

To determine the final tree structure, (Yuan et al. 2015) provided an idea. They referred to the number of nodes whose weight is larger than 0.01 as the **big node number (BNN)**.

1. Group all MCMC samples into unique BNN categories.
2. Find out the most frequently occurring unique BNN group.
3. Choose the tree structure in this group which has the maximum marginal likelihood.



All the data are separated correctly based on their class.



5 MNIST dataset

MNIST is a famous handwritten digits data set.

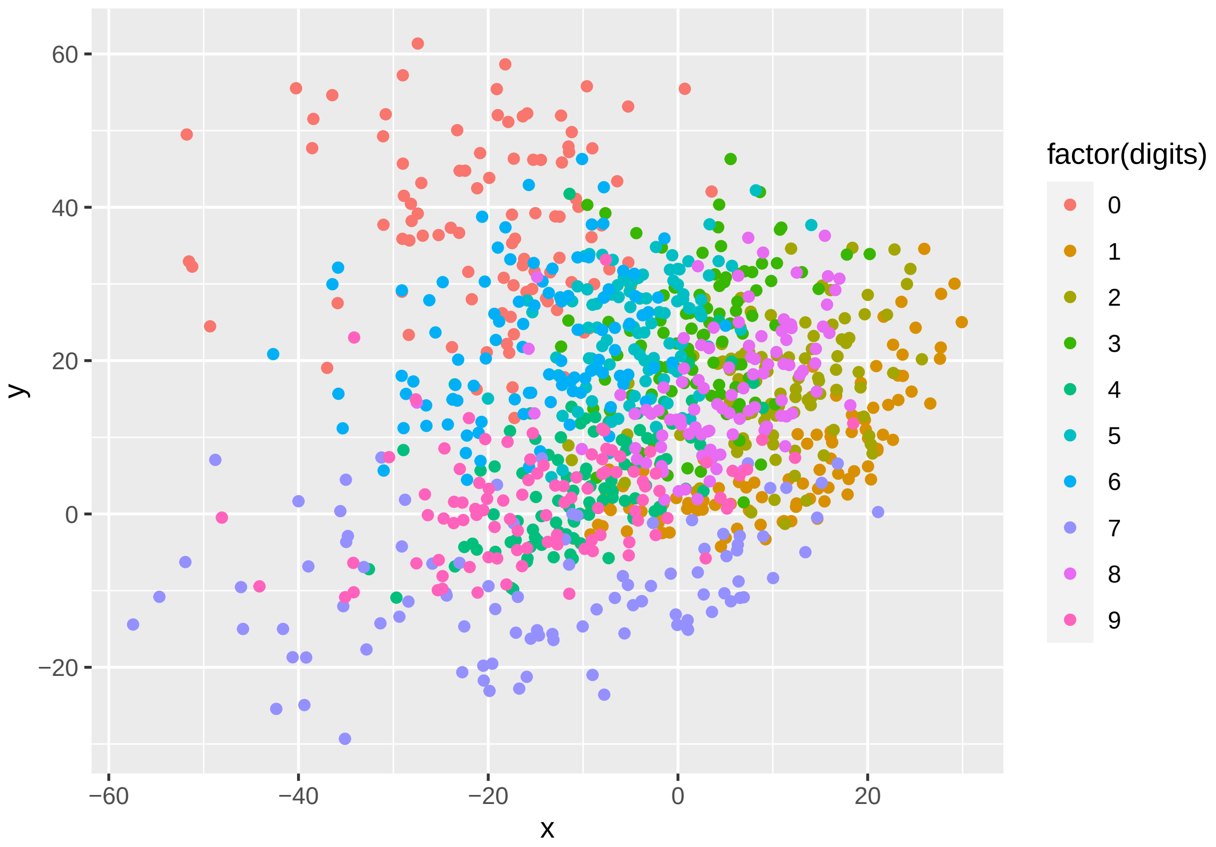
- Training: 60,000 images, Test: 10,000 images.
- Each image has 28x28 pixels.
- Each pixel of the image is in [0,255].

5.1 Prepare the MNIST-mini data

We extracted **5-dim features** for all training data from an *AutoEncoder* pre-trained using the whole training set. Then we sampled a subset of the training set where **each digit has 100 samples**. Therefore, the dimensions of data matrix we finally used is 1000×5. (*The features extracted from AutoEncoder are partly separated, which is helpful for subsequent clustering.*)

The settings are the same as in the simulation study, except that

- burnIn = 500, Iter = 3,000.



5.2 Initialization and training

```
#> Initialization: D = 3 and W = 3
#> No. 1 child of root is initializing...
#> No. 2 child of root is initializing...
#> No. 3 child of root is initializing...
#> Initialization Is Over!
```

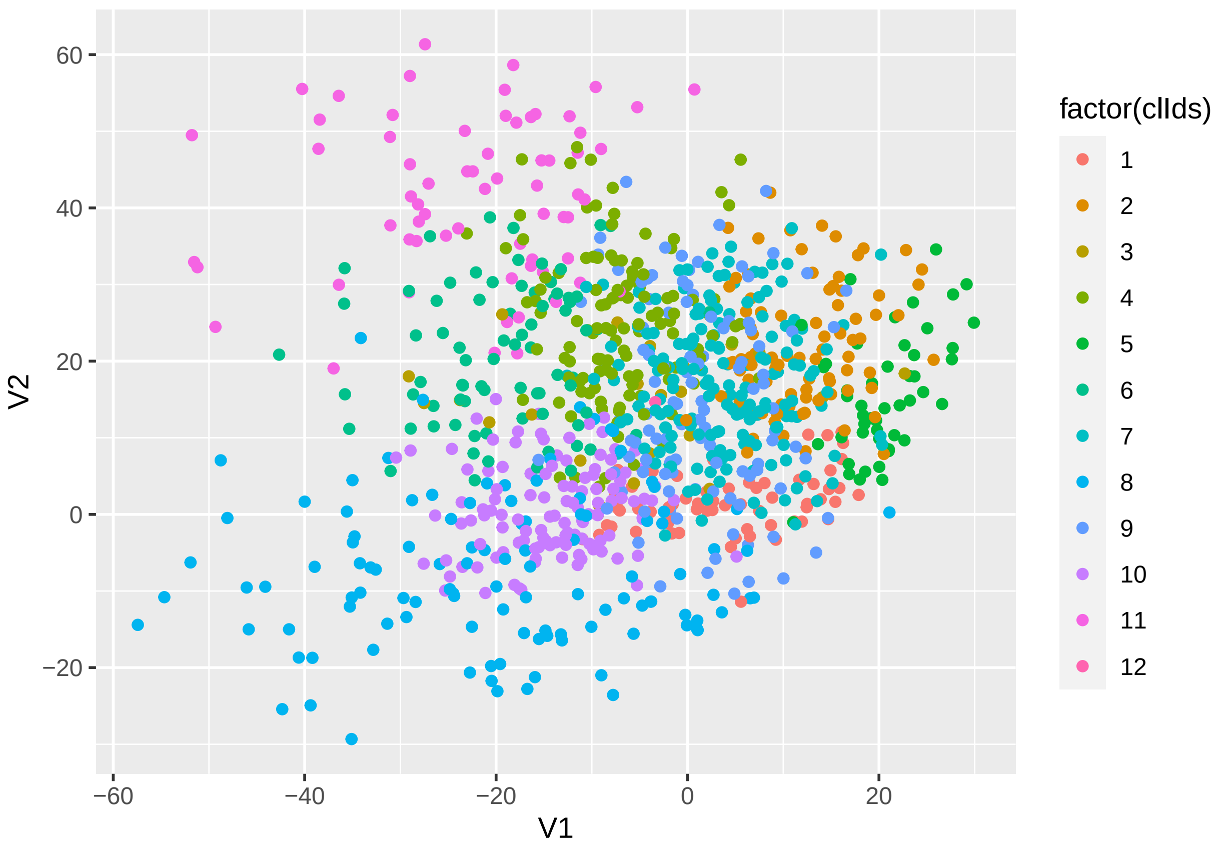
After initialization, we update all the parameters using Gibbs sampler.

```
#> ==== iter:  0
#> Update node params:  0.01296782
#> Update drift:  0.0008890629
#> Update sticks:  0.002919912
#> Update 3 hypers:  0.004132032
#> Update assignments:  1.019284
#> Swap-nodes move:  0.0325501
#> Update Keep tree:  0.001177073
#> ==== iter:  1000
#> Update node params:  0.005414963
#> Update drift:  0.000649929
#> Update sticks:  0.0007550716
#> Update 3 hypers:  0.001730919
#> Update assignments:  0.834553
#> Swap-nodes move:  0.02189994
#> Update Keep tree:  0.00100112
#> ==== iter:  2000
#> Update node params:  0.00564599
#> Update drift:  0.0006928444
#> Update sticks:  0.0009150505
#> Update 3 hypers:  0.001341105
#> Update assignments:  0.8016839
#> Swap-nodes move:  0.02917409
#> Update Keep tree:  0.0009260178
#> ==== iter:  3000
#> Update node params:  0.005882025
#> Update drift:  0.0006608963
#> Update sticks:  0.0007648468
#> Update 3 hypers:  0.001832962
#> Update assignments:  0.7969749
#> Swap-nodes move:  0.02704
#> Update Keep tree:  0.001327038

#> Total execution time:
#> Time difference of 34.62513 mins
```

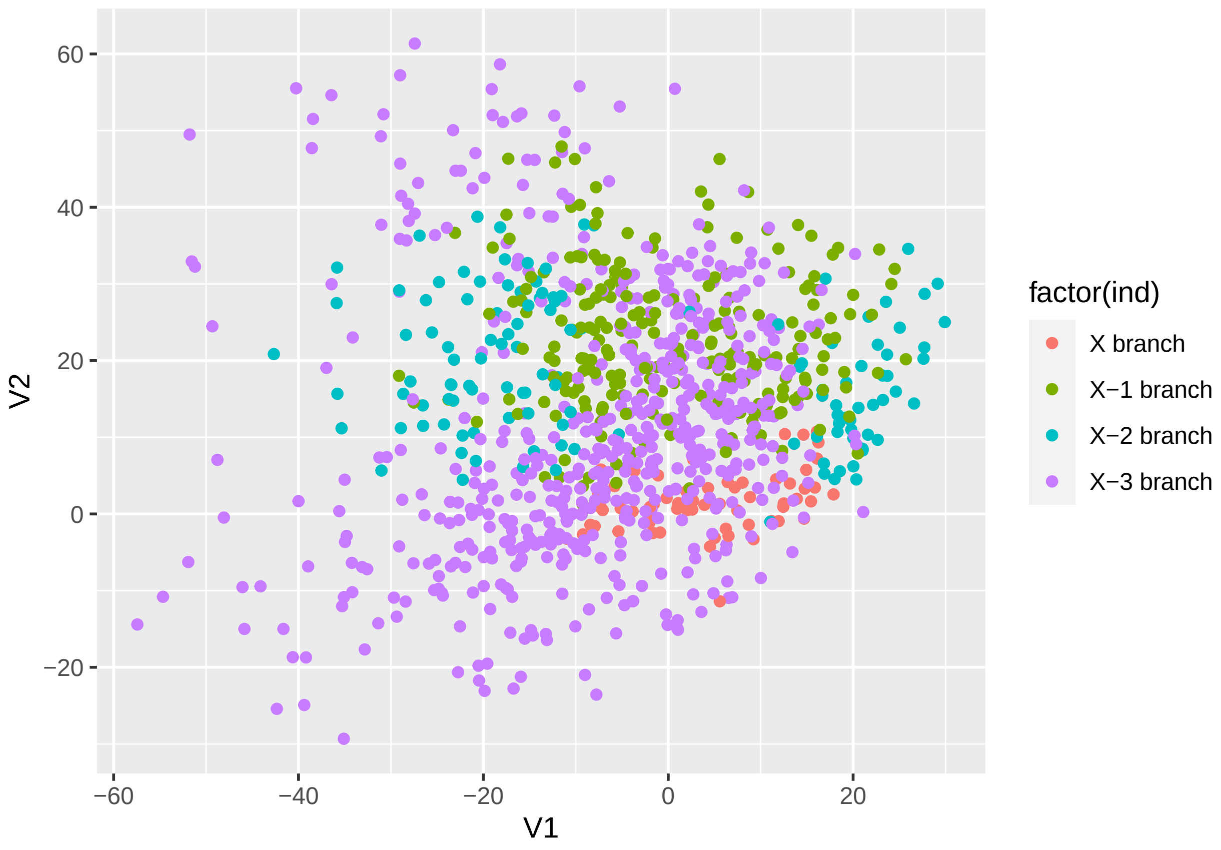
5.3 Results

The clustering result is shown in the following figure.



Result 1. Data are clustered almost based on their true classes, although there is a large overlap area. This demonstrates the ability of our model to cluster data with overlapping.

Next, we aggregate clusters on the same branch together to show the “similarity” (closely-located) within each branch.



Result 2. Data in the same branches are closer than others. This shows our model is able to discover some unknown relationship behind clusters.

6 Future work

1. Compare the performances of clustering among our model and the baseline methods (hierarchical clustering and K-centroid), using some statistics such as **v-measure**.
2. Apply our model to some general real data.