Package: iClusterVB (via r-universe)

September 20, 2024

Type Package

Title Fast Integrative Clustering and Feature Selection for High Dimensional Data

Version 0.1.2

Author Abdalkarim Alnajjar [aut, cre, cph] (<https://orcid.org/0009-0008-4439-1214>), Zihang Lu [aut]

Maintainer Abdalkarim Alnajjar <abdalkarim.alnajjar@queensu.ca>

Description A variational Bayesian approach for fast integrative clustering and feature selection, facilitating the analysis of multi-view, mixed type, high-dimensional datasets with applications in fields like cancer research, genomics, and more.

License MIT + file LICENSE

URL https://github.com/AbdalkarimA/iClusterVB

BugReports https://github.com/AbdalkarimA/iClusterVB/issues

Depends R (>= 4.0.0)

Imports cluster, clustMixType, cowplot, ggplot2, graphics, grDevices, mclust, MCMCpack, mvtnorm, pheatmap, poLCA, R.utils, Rcpp (>= 1.0.12), stats, utils, VarSelLCM

Suggests knitr, rmarkdown, survival, survminer

LinkingTo Rcpp, RcppArmadillo

Encoding UTF-8

LazyData true

LazyDataCompression xz

RoxygenNote 7.3.2

VignetteBuilder knitr

NeedsCompilation yes

Repository CRAN

Date/Publication 2024-08-20 19:10:02 UTC

Contents

---

chmap *Generates a heat map based on an iClusterVB object*

Description

Generates a heat map based on an iClusterVB object

Usage

```
chmap(fit, rho = 0.5, cols = NULL, title = NULL, ...)
```

Arguments

`fit`	A fitted iClusterVB object.
`rho`	The minimum probability of inclusion for features shown on the heatmap. Default is 0.5. 0 would show all features. Only useful for VS_method = 1.
`cols`	A vector of colors to use for the clusters. The default is a random selection of colors.
`title`	A character vector or a single value. Title of the heat map. The default is "View 1 - Distribution 1", ..., "View R - Distribution R".
`...`	Additional arguments to be passed down to `pheatmap`

Value

Returns a heat map for each data view.

Examples

```
# Setting up the data
dat1 <- list(
  gauss_1 = sim_data$continuous1_data[c(1:20, 61:80, 121:140, 181:200), 1:75],
  gauss_2 = sim_data$continuous2_data[c(1:20, 61:80, 121:140, 181:200), 1:75],
  poisson_1 = sim_data$count_data[c(1:20, 61:80, 121:140, 181:200), 1:75],
  multinomial_1 = sim_data$binary_data[c(1:20, 61:80, 121:140, 181:200), 1:75]
)
```

```
# Recoding `0`s to `2`s
dat1$multinomial_1[dat1$multinomial_1 == 0] <- 2

dist <- c(
  "gaussian", "gaussian",
  "poisson", "multinomial"
)

fit_iClusterVB <- iClusterVB(
  mydata = dat1,
  dist = dist,
  K = 4,
  initial_method = "VarSelLCM",
  VS_method = 1,
  max_iter = 25
)


# We can set the colors, turn off scaling and set titles

chmap(fit_iClusterVB,
  cols = c("red", "blue", "green", "purple"),
  title = c("Gene Expression", "DNA Methylation", "Copy Number", "Mutation Status"),
  scale = "none"
)
```

---

iClusterVB *Fast Integrative Clustering for High-Dimensional Multi-View Data Using Variational Bayesian Inference*

Description

iClusterVB offers a novel, fast, and integrative approach to clustering high-dimensional, mixed-type, and multi-view data. By employing variational Bayesian inference, iClusterVB facilitates effective feature selection and identification of disease subtypes, enhancing clinical decision-making.

Usage

```
iClusterVB(
  mydata,
  dist,
  K = 10,
  initial_method = "VarSelLCM",
  VS_method = 0,
  initial_cluster = NULL,
  initial_vs_prob = NULL,
  initial_fit = NULL,
  initial_omega = NULL,
  input_hyper_parameters = NULL,
```

```
  max_iter = 200,
  early_stop = 1,
  per = 10,
  convergence_threshold = 1e-04
)
```

Arguments

`mydata`	A list of length R, where R is the number of datasets, containing the input data. • Note: For **categorical** data, `0`'s must be re-coded to another, non-`0` value.
`dist`	A vector of length R specifying the type of data or distribution. Options include: 'gaussian' (for continuous data), 'multinomial' (for binary or categorical data), and 'poisson' (for count data).
`K`	The maximum number of clusters, with a default value of 10. The algorithm will converge to a model with dominant clusters, removing redundant clusters and automating the determination of the number of clusters.
`initial_method`	The initialization method for cluster allocation. Options include: "VarSelLCM" (default), "random", "kproto" (k-prototypes), "kmeans" (continuous data only), "mclust" (continuous data only), or "lca" (poLCA, categorical data only).
`VS_method`	The variable/feature selection method. Options are 0 for clustering without variable/feature selection (default) and 1 for clustering with variable/feature selection.
`initial_cluster`	The initial cluster membership. The default is NULL, which uses initial_method for initial cluster allocation. If not NULL, it will override the initial values setting for this parameter.
`initial_vs_prob`	The initial variable/feature selection probability, a scalar. The default is NULL, which assigns a value of 0.5.
`initial_fit`	Initial values based on a previously fitted iClusterVB model (an iClusterVB object). The default is NULL.
`initial_omega`	Customized initial values for feature inclusion probabilities. The default is NULL. If not NULL, it will override the initial values setting for this parameter. If VS_method = 1, initial_omega is a list of length R, with each element being an array with dimensions {dim=c(N, p[[r]])}. Here, N is the sample size and p[[r]] is the number of features for dataset r, where r = 1, ..., R.
`input_hyper_parameters`	A list of the initial hyper-parameters of the prior distributions for the model. The default is NULL, which assigns alpha_00 = 0.001, mu_00 = 0, s2_00 = 100, a_00 = 1, b_00 = 1,kappa_00 = 1, u_00 = 1, v_00 = 1.
`max_iter`	The maximum number of iterations for the VB algorithm. The default is 200.
`early_stop`	Whether to stop the algorithm upon convergence or to continue until `max_iter` is reached. Options are 1 (default) to stop when the algorithm converges, and 0 to stop only when `max_iter` is reached.
`per`	Print information every "per" iterations. The default is 10.
`convergence_threshold`	The convergence threshold for the change in ELBO. The default is 0.0001.

Value

The `iClusterVB` function creates an object (list) of class `iClusterVB`. Relevant outputs include:

`elbo:` The evidence lower bound for each iteration.

`cluster:` The cluster assigned to each individual.

`initial_values:`
A list of the initial values.

`hyper_parameters:`
A list of the hyper-parameters.

`model_parameters:`
A list of the model parameters after the algorithm is run.

- Of particular interest is `rho`, a list of the posterior inclusion probabilities for the features in each of the data views. This is the probability of including a certain predictor in the model, given the observations. This is only available if `VS_method = 1`.

Note

If any of the data views are "gaussian", please include them **first**, both in the input data `mydata` and correspondingly in the distribution vector `dist`. For example, `dist <- c("gaussian","gaussian", "poisson", "multinomial")`, and **not** `dist <- c("poisson", "gaussian","gaussian", "multinomial")` or `dist <- c("gaussian", "poisson", "gaussian", "multinomial")`

Examples

```
# sim_data comes with the iClusterVB package.
dat1 <- list(
  gauss_1 = sim_data$continuous1_data[c(1:20, 61:80, 121:140, 181:200), 1:75],
  gauss_2 = sim_data$continuous2_data[c(1:20, 61:80, 121:140, 181:200), 1:75],
  poisson_1 = sim_data$count_data[c(1:20, 61:80, 121:140, 181:200), 1:75],
  multinomial_1 = sim_data$binary_data[c(1:20, 61:80, 121:140, 181:200), 1:75]
)

# We re-code `0`s to `2`s

dat1$multinomial_1[dat1$multinomial_1 == 0] <- 2

dist <- c(
  "gaussian", "gaussian",
  "poisson", "multinomial"
)

# Note: `max_iter` is a time-intensive step.
# For the purpose of testing the code, use a small value (e.g. 10).
# For more accurate results, use a larger value (e.g. 200).

fit_iClusterVB <- iClusterVB(
  mydata = dat1,
  dist = dist,
  K = 4,
```

```
  initial_method = "VarSelLCM",
  VS_method = 1,
  max_iter = 50
)

# We can obtain a summary using the summary() function
summary(fit_iClusterVB)
```

`laml` *LAML (Acute Myeloid Leukemia) Data*

Description

This is a subset of the LAML (Acute Myeloid Leukemia) data (TCGA, 2013). The Acute Myeloid Leukemia (laml_tcga) datasets were download using the cBioPortal for Cancer Genomics tool (Cerami et al., 2012; Gao et al., 2013). The 170 samples with gene expression data and mutation data were included. Only a subset of the genes was selected, as desribed below. To access the data containing all the genes, please visit: https://github.com/AbdalkarimA/iClusterVB

Usage

```
data(laml)
```

Value

Within the data file, there is:

`laml.cli:`	A dataframe of clinical information for the 170 samples.
`laml.exp:`	A matrix of 170 samples and the gene expression values of the 500 genes chosen by Zainul Abidin and Westhead (2016) based on having the highest ranked-based coefficients of variation and standard deviation across the samples. Some names may have been updated or corrected from the supplementary material.
`laml.mut:`	A matrix of 170 samples and the mutation status of 156 genes that had >=2 mutations. 1 indicates the presence of mutation, and 0 indicates the absence of mutation.

References

Cancer Genome Atlas Research Network, Ley, T. J., Miller, C., Ding, L., Raphael, B. J., Mungall, A. J., Robertson, A., Hoadley, K., Triche, T. J., Jr, Laird, P. W., Baty, J. D., Fulton, L. L., Fulton, R., Heath, S. E., Kalicki-Veizer, J., Kandoth, C., Klco, J. M., Koboldt, D. C., Kanchi, K. L., Kulkarni, S., ... Eley, G. (2013). Genomic and epigenomic landscapes of adult de novo acute myeloid leukemia. The New England journal of medicine, 368(22), 2059–2074. https://doi.org/10.1056/NEJMoa1301689

Cerami, E., Gao, J., Dogrusoz, U., Gross, B. E., Sumer, S. O., Aksoy, B. A., Jacobsen, A., Byrne, C. J., Heuer, M. L., Larsson, E., Antipin, Y., Reva, B., Goldberg, A. P., Sander, C., & Schultz, N. (2012). The cBio cancer genomics portal: an open platform for exploring multidimensional cancer genomics data. Cancer discovery, 2(5), 401–404. https://doi.org/10.1158/2159-8290.CD-12-0095

Gao, J., Aksoy, B. A., Dogrusoz, U., Dresdner, G., Gross, B., Sumer, S. O., Sun, Y., Jacobsen, A., Sinha, R., Larsson, E., Cerami, E., Sander, C., & Schultz, N. (2013). Integrative analysis of complex cancer genomics and clinical profiles using the cBioPortal. Science signaling, 6(269), pl1. https://doi.org/10.1126/scisignal.2004088

Zainul Abidin, F. N., & Westhead, D. R. (2017). Flexible model-based clustering of mixed binary and continuous data: application to genetic regulation and cancer. Nucleic acids research, 45(7), e53. https://doi.org/10.1093/nar/gkw1270

`piplot` *Generates a probability inclusion plot based on an iClusterVB object*

Description

Generates a probability inclusion plot based on an iClusterVB object

Usage

```
piplot(
  fit,
  plot_grid = TRUE,
  ylab = "Probability of Inclusion",
  title = NULL,
  ...
)
```

Arguments

`fit`	A fitted iClusterVB object.
`plot_grid`	LOGICAL. Whether to use the `plot_grid` function from the **cowplot** package. The default is TRUE.
`ylab`	The y-axis label. The default is "Probability of Inclusion".
`title`	The title of the plots. It can be a character vector or a single value. The default output is "View 1 - Distribution 1", ..., "View R - Distribution R".
`...`	Additional arguments to add to the `plot_grid` function.

Value

Returns a probability inclusion plot or plots.

Examples

```
# Setting up the data
dat1 <- list(
  gauss_1 = sim_data$continuous1_data[c(1:20, 61:80, 121:140, 181:200), 1:75],
  gauss_2 = sim_data$continuous2_data[c(1:20, 61:80, 121:140, 181:200), 1:75],
  poisson_1 = sim_data$count_data[c(1:20, 61:80, 121:140, 181:200), 1:75],
  multinomial_1 = sim_data$binary_data[c(1:20, 61:80, 121:140, 181:200), 1:75]
```

```
)

# Recoding `0`s to `2`s
dat1$multinomial_1[dat1$multinomial_1 == 0] <- 2

dist <- c(
  "gaussian", "gaussian",
  "poisson", "multinomial"
)

fit_iClusterVB <- iClusterVB(
  mydata = dat1,
  dist = dist,
  K = 4,
  initial_cluster= c(rep(1, 20), rep(2, 20), rep(3, 20), rep(4, 20)),
  VS_method = 1,
  max_iter = 20
)

piplot(fit_iClusterVB, plot_grid = FALSE)
```

`plot.iClusterVB` *Generic plot method for 'iClusterVB' objects*

Description

Generic plot method for 'iClusterVB' objects

Usage

```
## S3 method for class 'iClusterVB'
plot(x, ...)
```

Arguments

`x`	A fitted iClusterVB object.
`...`	Potential further arguments (unused)

Value

Returns an evidence lower bound (ELBO) plot and a barplot of cluster percentages.

Examples

```
# Setting up the data
dat1 <- list(
  gauss_1 = sim_data$continuous1_data[c(1:20, 61:80, 121:140, 181:200), 1:75],
  gauss_2 = sim_data$continuous2_data[c(1:20, 61:80, 121:140, 181:200), 1:75],
```

```
  poisson_1 = sim_data$count_data[c(1:20, 61:80, 121:140, 181:200), 1:75],
  multinomial_1 = sim_data$binary_data[c(1:20, 61:80, 121:140, 181:200), 1:75]
)

# Recoding `0`s to `2`s
dat1$multinomial_1[dat1$multinomial_1 == 0] <- 2

dist <- c(
  "gaussian", "gaussian",
  "poisson", "multinomial"
)

fit_iClusterVB <- iClusterVB(
  mydata = dat1,
  dist = dist,
  K = 4,
  initial_method = "VarSelLCM",
  VS_method = 1,
  max_iter = 25
)

plot(fit_iClusterVB)
```

`sim_data` *Simulated Dataset*

Description

The dataset consists of $N = 240$ individuals and $R = 4$ data views with different data types. Two of the data views are continuous, one is count, and one is binary. The *true* number of clusters was set to $K = 4$, and the cluster proportions were set at $\pi_1 = 0.25, \pi_2 = 0.25, \pi_3 = 0.25, \pi_4 = 0.25$, such that we have balanced cluster proportions. Each of the data views had $p_r = 500$ features, $r = 1, \ldots, 4$, but only 50, or 10%, were relevant features that contributed to the clustering, and the rest were noise features that did not contribute to the clustering. In total, there were $p = \sum_{r=1}^{4} = 2000$ features.

For data view 1 (continuous), relevant features were generated from the following normal distributions: $\text{N}(10, 1)$ for Cluster 1, $\text{N}(5, 1)$ for Cluster 2, $\text{N}(-5, 1)$ for Cluster 3, and $\text{N}(-10, 1)$ for Cluster 4, while noise features were generated from $\text{N}(0, 1)$. For data view 2 (continuous), relevant features were generated from the following normal distributions: $\text{N}(-10, 1)$ for Cluster 1, $\text{N}(-5, 1)$ for Cluster 2, $\text{N}(5, 1)$ for Cluster 3, and $\text{N}(10, 1)$ for Cluster 4, while noise features were generated from $\text{N}(0, 1)$. For data view 3 (binary), relevant features were generated from the following Bernoulli distributions: $\text{Bernoulli}(0.05)$ for Cluster 1, $\text{Bernoulli}(0.2)$ for Cluster 2, $\text{Bernoulli}(0.4)$ for Cluster 3, and $\text{Bernoulli}(0.6)$ for Cluster 4, while noise features were generated from $\text{Bernoulli}(0.1)$. For data view 4 (count), relevant features were generated from the following Poisson distributions: $\text{Poisson}(50)$ for Cluster 1, $\text{Poisson}(35)$ for Cluster 2, $\text{Poisson}(20)$ for Cluster 3, and $\text{Poisson}(10)$ for Cluster 4, while noise features were generated from $\text{Poisson}(2)$.

Usage

```
data(sim_data)
```

Format

A list containing four datasets, and other elements of interest.

`summary.iClusterVB` *Generic summary method for 'iClusterVB' objects*

Description

Generic summary method for 'iClusterVB' objects

Usage

```
## S3 method for class 'iClusterVB'
summary(object, rho = 0.5, ...)
```

Arguments

`object`	A fitted iClusterVB object.
`rho`	The minimum posterior inclusion probability of interest to count the number of features that are >= `rho`. Default is 0.5. Only works for VS_method = 1.
`...`	Potential further arguments

Value

Returns a summary list for an 'agnes' object.

Examples

```
# Setting up the data
dat1 <- list(
  gauss_1 = sim_data$continuous1_data[c(1:20, 61:80, 121:140, 181:200), 1:75],
  gauss_2 = sim_data$continuous2_data[c(1:20, 61:80, 121:140, 181:200), 1:75],
  poisson_1 = sim_data$count_data[c(1:20, 61:80, 121:140, 181:200), 1:75],
  multinomial_1 = sim_data$binary_data[c(1:20, 61:80, 121:140, 181:200), 1:75]
)

# Recoding `0`s to `2`s
dat1$multinomial_1[dat1$multinomial_1 == 0] <- 2

dist <- c(
  "gaussian", "gaussian",
  "poisson", "multinomial"
)
```

```
fit_iClusterVB <- iClusterVB(
  mydata = dat1,
  dist = dist,
  K = 4,
  initial_method = "VarSelLCM",
  VS_method = 1,
  max_iter = 25
)

## S3 method for class 'iClusterVB'
summary(fit_iClusterVB, rho = 0.75)
```

Index