

# An Introduction to lr2cluster

Kurnia Susvitasari

2022-10-26

This document is to provide an introductory tutorial to `lr2cluster` package. The package implements tools to help identify cluster assignments using multiple data sources when some direct methods such as contact tracing and genomic sequencing are only available for some data. There are also complimentary functions, such as to estimate a cluster's size increment and to choose which new cases to be sequenced next. This tutorial provides an overview of `lr2cluster`'s basic functionalities.

## Installing Package

To install the package, you need to install package remotes first:

```
install.packages("remotes")
```

Once remotes is successfully installed, run the following to install and load `lr2cluster`:

```
remotes::install_github("ksusvita92/lr2cluster")  
library(lr2cluster)
```

```
library(lr2cluster)
```

## Valencia Data

The data of TB cases in Valencia is available at <https://doi.org/10.7554/eLife.76605>. The following is a short script to prepare the data. In this tutorial, we do not provide the real patient's geographical information.

```
library(readxl)  
library(dplyr)
```

```
##  
## Attaching package: 'dplyr'  
  
## The following objects are masked from 'package:stats':  
##  
##   filter, lag  
  
## The following objects are masked from 'package:base':  
##  
##   intersect, setdiff, setequal, union  
  
url <- paste("https://elifesciences.org/download/aHR0cHM6Ly9jZG4uZWxpZmVzY2llbmNlcy5vcmc",  
             "vYXJ0aWNsZXNvNzY2MDUvZWxpZmVzY2MDUtc3VwcDEtdjEueGxzeA--/elife-76605-suppl-v1.xlsx?_ha",  
             "sh=pzQwKD1DzDLre7kKrWI%2Fhd%2BjY2FGgpekrPI4vXrlWNo%3D", sep = "")  
destfile <- "rawdt.xlsx"
```

```

curl::curl_download(url, destfile)
rawdt <- read_excel(destfile, range = "A3:AB778", na = "NA")

## New names:
## * `` -> `...1`

# subset the necessary column and cases
tb_valencia <- rawdt %>%
  transmute(ID = ...1, Cluster = `Genomic
Cluster ID`,
  Gender = Gender,
  Foreign = ifelse(`Country of birth`=="SPAIN", "No", "YES"),
  Diabetes = Diabetes,
  HIV = `HIV infected`) %>%
  filter(!is.na(Gender), !is.na(Foreign), !is.na(Diabetes), !is.na(HIV))

# subset clusters having >2 members
nm <- as.data.frame(table(tb_valencia$Cluster)) %>% filter(Freq > 2) %>% pull(Var1)
tb_valencia <- tb_valencia %>% filter(Cluster %in% nm)

# create toy example for patient's location
set.seed(123)
tb_valencia <- tb_valencia %>%
  mutate(Longitude = rnorm(nrow(tb_valencia)), Latitude = rnorm(nrow(tb_valencia)))

```

## Transform to pairwise-level data

Pairwise-level data is a data used in the pairwise logistic regression model. This data represents the similarity (if the variable is categorical) and distance (if the variable is numeric) between a pair of cases. For example, in the Valencia data, latitude and longitude are the columns that represent the geographical location of an individual. Then the pairwise-level of these variables is the spatial distance between two cases.

To transform Valencia data into pairwise-level data, run:

```

dt <- tb_valencia[tb_valencia$Cluster != "unique",] #exclude unclustered cases
pairdt <- zCovariate(cluster = dt$Cluster,
  X = dt[,3:6],
  location = dt[,7:8],
  id = dt$ID)
head(pairdt)

```

```

##   case to.case y  Spatial Gender Foreign Diabetes HIV
## 1 G401m  G368m 1 132.6483  DIFF    DIFF    DIFF  no
## 2 G553m  G368m 1 251.3182  DIFF    DIFF    DIFF  no
## 3 G566m  G368m 1 190.1091  male    YES    DIFF  no
## 4 G1163  G368m 1 294.5370  male    YES    DIFF  no
## 5 G1411  G368m 1 348.3176  DIFF    DIFF    DIFF  no
## 6 G201   G368m 0 271.9317  male    YES    DIFF  no

```

Notice that there are 4 new columns in the data frame above: `case`, `to.case`, `y`, and `Spatial`.

- `case`, `to.case`: vector of case id; together represent a pair.
- `y`: a binary variable; `y = 1` means if a pair is in the same cluster, `y = 0` means otherwise.
- `Spatial`: numeric vector that represents spatial distance between a pair.

Each categorical variable will have one additional level, called **DIFF** which indicates if a pair has different values. To see more details about the function above, run `?zCovariate`.

## Pairwise Logistic Regression

There are two logistic regression models introduced in this package: the multinomial logistic regression (MLR) and the pairwise logistic regression (PLR). The difference between the two models lies on the data type used to train the model, and the response variable. For instance, MLR uses individual-level data and the response variable is the index of the cases' clusters, whereas PLR uses pairwise-level data and binary response variable indicating whether two cases are in the same cluster.

### Fit PLR model

PLR perform regression analysis which maps pairwise predictors to a binary response variable. Suppose we want to fit predictors `sex`, `foreign`, `dx_data`, and `latitude` and `longitude`.

```
fit_plr <- plr(formula = Cluster ~ Gender+Foreign+Diabetes+HIV+Latitude+Longitude,
               data = dt)
summary(fit_plr)
```

```
##
## Call:
## glm(formula = mod, family = binomial(), data = traindata)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -0.3175  -0.2366  -0.2242  -0.1775   2.9739
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -4.1446254  0.2696505 -15.370 < 2e-16 ***
## Genderfemale -0.0215090  0.2333326  -0.092  0.92655
## Gendermale   0.1057894  0.1327907   0.797  0.42565
## ForeignNo    0.5722562  0.1479516   3.868  0.00011 ***
## ForeignYES   1.0130621  0.2335062   4.338 1.43e-05 ***
## Diabetesno   -0.0137715  0.1525509  -0.090  0.92807
## Diabetesyes  -0.1957849  0.5991499  -0.327  0.74384
## HIVno        -0.1180810  0.1836775  -0.643  0.52031
## HIVyes       -0.1802961  1.0268152  -0.176  0.86062
## Spatial      0.0002080  0.0006323   0.329  0.74222
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 2499.0  on 11324  degrees of freedom
## Residual deviance: 2473.2  on 11315  degrees of freedom
## AIC: 2493.2
##
## Number of Fisher Scoring iterations: 7
```

`summary()` function provides the estimated regression coefficients, together with the Wald test. An important note for the users is if ones want to fit spatial distance between two cases, ones must name the geographical

location as latitude (or lat, Lat, Latitude), and longitude (or long, Long, Longitude).

## Predict if two cases are in the same cluster

Suppose there are a collection of new cases which we have no information on their true clusters, and we want to predict the probability if a pair belongs to the same cluster. To illustrate this, let us split the `tb_valencia` data into two, one is to train the PLR model and the other is to test the model.

To make sure we split the data such that there are at least one case per cluster in each data split, use `createDataPartition()` from `caret` package. In this example, we set 60% proportion of the data to be in training set.

```
library(caret)

## Loading required package: ggplot2
## Loading required package: lattice

set.seed(123)
id <- createDataPartition(dt$Cluster, p = .6, list = F)
traindt <- dt[id,]
testdt <- dt[-id,]
```

Use `plr()` function to fit PLR model on `traindt`, and run the following to get the estimated probability of the response variable:

```
fit_plr <- plr(formula = Cluster ~ Gender+Foreign+Diabetes+HIV+Latitude+Longitude,
               data = traindt)
pred_plr <- predict(obj = fit_plr,
                   newdata = testdt,
                   case.id = testdt$ID) #case.id can be NULL
```

The function `predict.plr()` (or `predict()`) returns a data frame which contains a vector of the predicted probability if a pair of cases are in the same cluster, and its standard error.

## Finding the optimum threshold

As in any binary classification problem where a threshold must be selected, `lr2cluster` provides a function to find an optimum threshold which depends on how the user values the cost of false positive errors (saying two cases are in the same cluster when they are not) versus false negative errors (saying two cases are not in the same cluster when they are).

The function `optThreshold()` requires a vector of the true response variable and its prediction. The later is obtained from `pred_plr$y`, and to obtain the former, use `zCovariate()` function on `testdt`.

```
# get the true response variable of testdt data
tr_resp <- zCovariate(cluster = testdt$Cluster)$y
opt_threshold <- optThreshold(response = tr_resp,
                             prediction = pred_plr$y,
                             cost.ratio = 50) #default is 1
```

```
## Setting levels: control = 0, case = 1
```

```
opt_threshold
```

```
##
## -- The optimum threshold using generalized Youden's index --
## =====
```

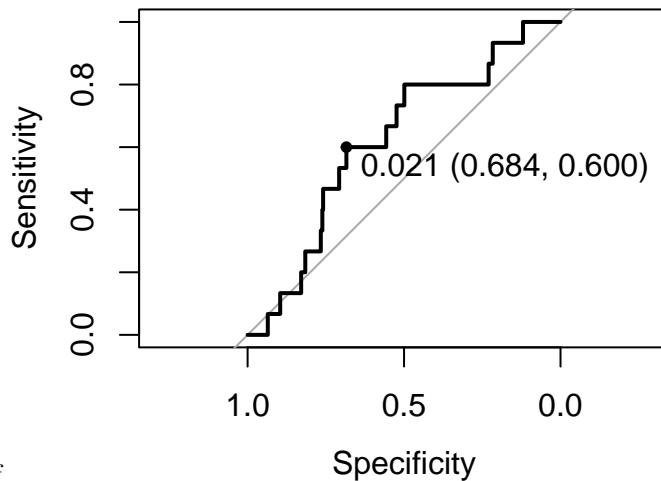
```
##
## threshold specificity sensitivity accuracy auc
## 0.0206544 0.6842975 0.6000000 0.6832653 0.6200000
```

`optThreshold()` returns a list of values, such as:

- **threshold**: the optimum threshold obtained, given `cost.ratio`.
- **specificity**, **sensitivity**: the true negative rate and the true positive rate, evaluated at the optimum threshold.
- **accuracy**: accuracy of the prediction.
- **'roc'**: an object of class "roc".

To plot the ROC, you can run:

```
plot(opt_threshold)
```



roc-1.pdf

which also shows the position of the optimum threshold given `cost.ratio`.

## Cluster Assignment

The previous section is to predict if some new cases belong to the same cluster. However, ones may wonder if those cases may belong to one of the known clusters in the training set. This section is addressed to find the most probable clusters a new case can be assigned to based on some scores called *cluster scores*. These scores, which are on a scale between 0 to 1, represent how likely a new case belong to some clusters.

### Finding probable clusters a case most likely to belong to

The cluster assignment can be done by setting a threshold or choosing the best  $K$  clusters with higher ranks. If a threshold is provided, any clusters whose scores are less than the threshold will be omitted. Otherwise, the method will assign a case to the best  $K$  clusters with higher ranks.

To do this task, we will use `traindt` and `testdt` again, so that we can test the accuracy of our assignments. We will use the same variables as the previous section as well.

```
assgn_plr <- clusterPLR(formula = Cluster ~ Gender+Foreign+Diabetes+HIV+Latitude+Longitude,
                        data = traindt,
                        newdata = testdt,
                        threshold = NULL,
```

```

                                nbest = 3)
assgn_plr

##
## -- Cluster assignment using pairwise logistic regression --
## =====
##
## Choosing the best 3 clusters.
## Showing only the first six cases.
##
##   rank_1 rank_2 rank_3
## 1  CL015  CL021  CL009
## 2  CL009  CL021  CL022
## 3  CL004  CL063  CL003
## 4  CL004  CL003  CL063
## 5  CL003  CL063  CL004
## 6  CL021  CL009  CL072

```

The code above is to find the best  $K = 3$  clusters a case can be assigned to using PLR model. If ones want to compare the method using random assignment or MLR model, run `clusterRandom()` or `clusterMLR()`.

To obtain the best probable clusters for all new cases, run:

```

getbest(assgn_plr) #or
assgn_plr$best_cluster

```

To obtain the cluster scores, run:

```

getclusterScore(assgn_plr) #or
assgn_plr$cluster_score

```

See `?clusterPLR`, `?clusterRandom`, and `?clusterMLR` for more details.

## Compute the accuracy for correctly assigning cases to their clusters

The accuracy in this context is the fraction of new cases whose true clusters is in the best  $K$  clusters predicted. For example, suppose that a case's true cluster is "Cluster A". If "Cluster A" is predicted as one of the  $K$  clusters, then the method will consider the assignment correct.

To compute the accuracy for predicting the new cases assignment, run

```

acc(obj = assgn_plr, true.cluster = testdt$Cluster)

```

## Other Application

### Cases to be sequenced next

Instead of predicting clusters a case can be assigned to, we can turn this relationship around, and ask, for a given cluster, which unassigned cases are most likely to belong to it. This could be used, for example, to decide which of several unsequenced cases should be sequenced next, if we are interested in identifying all the cases that belong to a particular cluster.

To do this task, run

```
next_case <- case2sequence(obj = assgn_plr,
                           case.id = testdt$ID,
                           nbest = 3)
```

```
next_case
```

```
##
## -- Predict cases to be sequenced next given clusters --
## =====
##
## Choosing the best 3 cases to be sequenced next.
## Showing only few cases.
##
##   cluster priority_1 priority_2 priority_3
## 1   CL001      G1939      G1058  G788FE29
## 2   CL002      G1188      G1303    G1910
## 3   CL003       G152      G1603    G1621
## 4   CL004      G1910      G1303    G1188
## 5   CL007       G108      G562m    G1342
## 6   CL008      G1188      G1303    G1910
```

The above code returns the suggested 3 cases to be chosen given a cluster based on their cluster score's rank. If ones want to find  $K$  best cases, change  $nbest = K$  or, ones can also provide a threshold which serves as a cut-off to any cases with lower score.

To obtain the best cases for all clusters, run

```
getbest(next_case) #or
next_case$best_cases
```

To obtain the accuracy of this task, run

```
acc(obj = next_case, true.cluster = testdt$Cluster)
```

## Estimate a cluster's size increment

Suppose that we have a cluster of interest  $C$  with some cases in it, and a collection of unassigned new cases. We can estimate the total number of new cases that would get assigned to  $C$ , and therefore estimate the increment of cluster  $C$ 's size.

```
clusterSize(obj = assgn_plr, rho = 0)
```

The argument  $\rho$  in `clusterSize()` represents the probability that a case does not belong to any of any given clusters.

## Closing Remarks

`lr2cluster` is a tool to assign newly identified cases of an infectious disease to existing transmission clusters using several data streams. The application is extended to also for example, predict which new cases to be sequenced next, given a cluster, and estimate a cluster's size increment.

For some general questions and bug report, please send to [ksusvita@gmail.com](mailto:ksusvita@gmail.com).