# Package: PrivateLR (via r-universe)

September 16, 2024

**Type** Package

**Title** Differentially Private Regularized Logistic Regression

**Version** 1.2-22

**Date** 2018-03-19

**Author** Staal A. Vinterbo <Staal.Vinterbo@ntnu.no>

**Maintainer** Staal A. Vinterbo <Staal.Vinterbo@ntnu.no>

**Description** Implements two differentially private algorithms for estimating L2-regularized logistic regression coefficients. A randomized algorithm F is epsilon-differentially private (C. Dwork, Differential Privacy, ICALP 2006 <DOI:10.1007/11681878_14>), if |log(P(F(D) in S)) - log(P(F(D') in S))| <= epsilon for any pair D, D' of datasets that differ in exactly one record, any measurable set S, and the randomness is taken over the choices F makes.

**License** GPL (>= 2)

**NeedsCompilation** no

**Date/Publication** 2018-03-20 13:48:35 UTC

**Repository** https://laats.r-universe.dev

**RemoteUrl** https://github.com/cran/PrivateLR

**RemoteRef** HEAD

**RemoteSha** 186f6945da348ef3cd4c5dca483aaa7212ec4336

## Contents

# PrivateLR *Differentially Private Logistic Regression*

## Description

PrivateLR implements two randomized algorithms for estimating $L_2$-regularized logistic regression coefficients that allow specifying the maximal effect a single point change in the training data are allowed to have. Specifically, the algorithms take as parameter the maximum allowed change in log-likelihood of producing particular coefficients resulting from any single training data point substitution.

## Usage

```
dplr(object, ...)

## S3 method for class 'formula'
dplr(object, data, lambda=NA, eps=1, verbose=0,
     rp.dim = 0, threshold='fixed', do.scale=FALSE, ...)
## S3 method for class 'numeric'
dplr(object, x, ...)
## S3 method for class 'logical'
dplr(object, x, ...)
## S3 method for class 'factor'
dplr(object, x, ...)
## S3 method for class 'data.frame'
dplr(object, target=ncol(object),...)
## S3 method for class 'matrix'
dplr(object, target=ncol(object),...)
## S3 method for class 'dplr'
predict(object, data, type = "probabilities", ...)
## S3 method for class 'dplr'
summary(object, ...)
## S3 method for class 'dplr'
print.summary(x, ...)
## S3 method for class 'dplr'
print(x, ...)

scaled(fml, data)
```

## Arguments

`object` can be given as an object of `formula`, `data.frame`, `matrix`, or `factor`, `logical`, `numeric` vector.

If a `data.frame`, `matrix` is given, this object contains both the dependent variable indexed by `target` as well as the independent variables, of which all are used. If the dependent variable is a factor, the first level is encoded as 0 and all others as 1.

In `dplr.formula` `object` is an object of class `formula` or an object that can be coerced into one.

If given as a vector, `object` contains the values of the dependent variable. The vector `object` can be of class `numeric`, in which case it must only contain values 0 and 1, `logical` in which case it is coerced into numeric by `as.numeric(object)`, or be of class `factor`, in which case it is coerced into numeric by encoding the first factor level as 0 and all the other levels as 1.

`data` a data frame or matrix containing the variables in the model described by `formula`.

`lambda` the regularization parameter. If `NA` (default), the smallest regularizer lambda such that `2 * log(1 + 1/(4* n * lambda)) == eps/10` is used. If `eps` is 0, then `lambda` is set to 0.001.

`eps` the privacy level. The coefficients of the model are computed by a method that guarantees `eps`-differential privacy. If `eps` is 0, then non-private regularized logistic regression is performed.

`verbose` regulates how much information is printed, 0 nothing, 1 a little, 2 more.

`rp.dim` if `rp.dim` is non-zero, random projection is performed on the data before estimating the model parameters. If `rp.dim` is positive, the projection will be onto `rp.dim` dimensions. If `rp.dim` is negative, `rp.dim` is set to 1/2 * (1/2)^(-2) * log(n). If `rp.dim` is larger than the dimensions of the data, a warning is given and no projection is performed.

`threshold` `dplr` can non-privately estimate the optimal probability threshold for classification by one of two methods: `'youden'`, or `'topleft'`. The method ’youden’ computes the threshold that maximizes the Youden J, while ’topleft’ computes the threshold corresponding to the point on the ROC curve that is closest to (0,1). Any other value (default) will result in a threshold of 0.5.

`do.scale` The privacy guarantees are for data where the covariate vectors lie within the unit ball. If `do.scale` is `TRUE`, input data will be scaled such that the covariate vectors all lie within the unit ball.

`type` `predict` can yield two types of results. If `type` is `"probabilities"`, then probabilities are returned, otherwise predictions of class values are returned using the threshold given by the `p.tr` element of `object`.

`x` In the `print` and `print.summary`, `x` is an object of class `"dplr"` or `summary.dplr`, typically returned by `dplr` or `summary`. Otherwise, the parameter `x` can either be a numeric matrix containing the covariates or dependent variables (one per column) corresponding to the dependent variable `object`, or a data frame containing a mix of numeric and factor columns. Any factor is internally recoded as contrasts.

`target` the index of the column in `data` that contains the target values. Default is the last column of `data`.

`fml` A formula that describes the dimensions of the data that should be scaled into the unit ball.

`...` `verbose`, `lambda`, and `eps` parameters. Not used in `summary`, `print`, and `predict` functions. In addition, a Boolean argument `op` can be given to `dplr` to select between *objective perturbation* (`op = TRUE`, the default) and *output perturbation* (`op = FALSE`).

## Details

The function `dplr` implements logistic regression using the differentially private methods by Chaudhuri, Monteleoni, and Sarwate.

The interface is similar but not identical to that of `lm`, with the addition of the possibility of supplying a data matrix or data.frame together with a target column index (defaults to `ncol(data)`).

The returned model instance has a convenience function `model$pred` that takes a data matrix or data frame to be classified as input.

The `print` function currently prints the summary.

The `scaled` function scales data such that covariate vectors lie within the unit ball. Note that the response variable is put as the last column in the data frame `data` returned. Also, the response column name might have changed, depending on the left side of the formula given.

**Methods details:**

A randomized algorithm $A$, taking a dataset as input, is said to be $\epsilon$-differentially private if it holds that

$$|\log(P(A(D) \in S)) - \log(P(A(D') \in S))| \leq \epsilon$$

for any pair of datasets $D, D'$ that differ in exactly one element, and any set $S$. We now turn to the algorithms implemented by `dplr`.

Let $\|v\|$ denote the L2 norm of a vector $v$, and let

$$J(w, \lambda) = ALL(w) + \lambda/2\|w\|^2$$

where $ALL(w)$ is the average logistic loss over the training data of size $n$ and dimension $d$ with labels $y$ and covariates $x$. L2-regularized logistic regression computes

$$w^* = \arg\min_w J(w, \lambda)$$

for a given $\lambda$.

The function `dplr` implements two approaches to $\epsilon$-differential private L2 regularized logistic regression (see the ... argument `op` above). The first is *output perturbation*, where we compute

$$w' = w^* + 2/(n\lambda\epsilon)b,$$

where $b$ is a $d$-dimensional real vector sampled with probability proportional to $\exp(-\|b\|)$.

The second is *objective perturbation*. Let

$$F(w, \lambda, \epsilon) = J(w, \lambda) + 2/(\epsilon n)b^T w$$

where $n$ and $b$ are as above. Let $c = 0.25$ and let $z = 2\log(1 + c/(\lambda n))$, then if

$$\epsilon - z > 0,$$

we compute

$$w' = \arg\min_w F(w, \lambda, \epsilon - z)$$

otherwise we compute an *adjusted lambda* version

$$w' = \arg\min_w F(w, c/(n(exp(\epsilon/4) - 1)), \epsilon/2).$$

The logistic regression model coefficients $w'$ are then $\epsilon$-differentially private.

## Value

The `dplr` function returns a class `"dplr"` list object comprised of elements including:

`par` the coefficients of the logistic model.

`coefficients` same as `par`

`value`, `counts`, `convergence`, `message`
these are as returned by the `optim` method.

`CIndex` the area under the ROC curve (aka., C-Index) of the model on its training data.

`eps` the supplied privacy level.

`lambda` the regularization parameter used

`n` the number of data points

`d` the dimensionality of the data points

`pred` a convenience function such that `predict(model, data, ...)` is equivalent to `model$pred(data,...)`.

`p.tr` this is the classification probability threshold.

`did.rp` TRUE if random projection was performed.

`rp.dim` if random projection was performed this contains the number of dimensions projected onto. Only present if random projection was performed.

`rp.p` the projection matrix used for random projection. Only present if random projection was performed.

`scaled` TRUE if data was scaled by providing `do.scale = TRUE`.

`status` a text string indicating the status of the computations. `'ok'` means all is well, `'adjusted lambda'` means that the regularizer was too small and had to be adjusted, and `'unique.outcomes'` means that the response had only one value, resulting in fixed coefficients returned.

The `scaled` function returns a list of the following:

`data` the scaled data frame

`scale` the scaling factor used.

## Warning

The privacy level is only guaranteed for the coefficients of the model, not for all the other returned values, and also only in the case when input data points (potentially after expansion of factors) are of L2-norm <= 1. In particular using prediction thresholds estimated using data (methods `'youden'` and `'topleft'`), as well as built in scaling of data is not guaranteed. Both of these are turned off by default.

## Note

This implementation was in part supported by NIH NLM grant 7R01LM007273-07 and NIH Roadmap for Medical Research grant U54 HL108460.

### Author(s)

Staal A. Vinterbo <sav@ucsd.edu>

### References

Chaudhuri K., Monteleoni C., and Sarwate, A. Differentially Private Empirical Risk Minimization. *JMLR*, 2011, 12, 1069-1109

### See Also

glm and predict

### Examples

```
data(iris)

# the following two are equivalent
# and predict Species being any
# but the first factor level.
model <- dplr(iris)
model <- dplr(Species ~ ., iris)

# pick a particular factor level and privacy level 2
model <- dplr(I(Species != 'setosa') ~ ., iris, eps=2)

# The following is again equivalent to the two first
# examples. Note that we need to remove 'Species' from the
# covariate matrix/data frame, and
# that the class reported by summary will now
# not be 'Species' but 'dplr.class'.
model <- dplr(iris$Species, iris[,-5])

# two equivalent methods to get at the predicted
# probabilities
p <- model$pred(iris)
p <- predict(model, iris)

# print a summary of the model. Note that
# only the coefficients are guaranteed
# to be generated in an eps-differentially
# private manner.
summary(model)
```

# Index