Package ‘glmpca’

July 18, 2020

Title Dimension Reduction of Non-Normally Distributed Data

Version 0.2.0

Description Implements a generalized version of principal components analysis (GLM-PCA) for dimension reduction of non-normally distributed data such as counts or binary matrices.
Townes FW, Hicks SC, Aryee MJ, Irizarry RA (2019) <doi:10.1186/s13059-019-1861-6>.
Townes FW (2019) <arXiv:1907.02647>.

License LGPL (>= 3) | file LICENSE

Depends R (>= 3.5),

Imports MASS, methods, stats, utils

Suggests covr, ggplot2, knitr, logisticPCA, markdown, Matrix, testthat,

URL https://github.com/willtownes/glmpca

BugReports https://github.com/willtownes/glmpca/issues

Language en-US

VignetteBuilder knitr

LazyData false

RoxygenNote 7.1.1

Encoding UTF-8

NeedsCompilation no

Author F. William Townes [aut, cre, cph],
Kelly Street [aut],
Jake Yeung [ctb]

Maintainer F. William Townes <will.townes@gmail.com>

Repository CRAN

Date/Publication 2020-07-18 17:30:02 UTC

R topics documented:

`glmpca` *GLM-PCA*

Description

Generalized principal components analysis for dimension reduction of non-normally distributed data.

Usage

```
glmpca(
  Y,
  L,
  fam = c("poi", "nb", "nb2", "binom", "mult", "bern"),
  minibatch = c("none", "stochastic", "memoized"),
  optimizer = c("avagrad", "fisher"),
  ctl = list(),
  sz = NULL,
  nb_theta = NULL,
  X = NULL,
  Z = NULL,
  init = list(factors = NULL, loadings = NULL),
  ...
)
```

Arguments

`Y` matrix-like object of count or binary data with features as rows and observations as columns. Sparse matrices from the `Matrix` package are supported. Column-oriented sparsity is preferred.

`L` desired number of latent dimensions (positive integer).

`fam` string describing the likelihood to use for the data. Families include Poisson ('`poi`'), negative binomial with global overdispersion ('`nb`'), negative binomial with feature-specific overdispersion ('`nb2`'), or binomial ('`binom`'). Families '`mult`' and '`bern`' are deprecated as both are special cases of '`binom`' with `sz` set to NULL and 1, respectively. They are provided only for backward compatibility. Family '`nb2`' has not been thoroughly tested and is considered experimental.

`minibatch` string describing whether gradients should be computed with all observations ('`none`', the default) or a subset of observations, which is useful for larger datasets. Option '`stochastic`' computes a noisy estimate of the full gradient using a random sample of observations at each iteration. Option '`memoized`' computes the full data gradient under memory constraints by caching summary statistics across batches of observations.

`optimizer` string describing whether to use the fast AvaGrad method ('`avagrad`', the default) or the slower diagonal Fisher scoring method ('`fisher`') that was used in the original glmpca implementation.

`ctl` a list of control parameters. See 'Details'

`sz` numeric vector of size factors for each observation. If NULL (the default), colSums are used for family '`binom`', and colMeans are used for families '`poi`','`nb`', and '`nb2`'.

`nb_theta` initial value for negative binomial overdispersion parameter(s). Small values lead to more overdispersion. Default: 100. See `negative.binomial`. (`nb_theta`->∞ equivalent to Poisson).

`X` a matrix of column (observations) covariates. Any column with all same values (eg. 1 for intercept) will be removed. This is because we force a feature-specific intercept and want to avoid collinearity.

`Z` a matrix of row (feature) covariates, usually not needed.

`init` a list containing initial estimates for the factors (`U`) and loadings (`V`) matrices.

`...` additional named arguments. Provided only for backward compatibility.

Details

The basic model is $R = AX' + ZG' + VU'$, where $E[Y] = M = linkinv(R)$. Regression coefficients are `A` and `G`, latent factors are `U` and loadings are `V`. The objective is to minimize the deviance between `Y` and `M`. The deviance quantifies the goodness-of-fit of the GLM-PCA model to the data (smaller=better). Note that `glmpca` uses a random initialization, so for fully reproducible results one may use `set.seed`.

The `ctl` argument accepts any of the following optional components:

verbose Logical. Should detailed status messages be printed during the optimization run? Default: `FALSE`.

batch_size Positive integer. How many observations should be included in a minibatch? Larger values use more memory but lead to more accurate gradient estimation. Ignored if `minibatch='none'`. Default: 1000.

lr Positive scalar. The AvaGrad learning rate. Large values enable faster convergence but can lead to numerical instability. Default: 0.1. If a numerical divergence occurs, `glmpca` will restart the optimization `maxTry` times (see below) and reduce the learning rate by a factor of five each time.

penalty Non-negative scalar. The L2 penalty for the latent factors. Default: 1. Regression coefficients are not penalized. Only used by the Fisher scoring optimizer. Larger values improve numerical stability but bias the parameter estimates. If a numerical divergence occurs, `glmpca` will restart the optimization `maxTry` times (see below) and increase the penalty by a factor of five each time.

maxTry Positive integer. In case of numerical divergence, how many times should optimization be restarted with a more stable penalty or learning rate? Default: 10.

minIter Positive integer. Minimum number of iterations (full passes through the dataset) before checking for numerical convergence. Default: 30.

maxIter Positive integer. Maximum number of iterations. If numerical convergence is not achieved by this point, the results may not be reliable and a warning is issued. Default: 1000.

tol Positive scalar. Relative tolerance for assessing convergence. Convergence is determined by comparing the deviance at the previous iteration to the current iteration. Default: 1e-4.

epsilon Positive scalar. AvaGrad hyperparameter. See Savarese et al (2020). Default: 0.1.

betas Numeric vector of length two. AvaGrad hyperparameters. See Savarese et al (2020). Default: `c(0.9,0.999)`.

minDev Scalar. Minimum deviance threshold at which optimization is terminated. Useful for comparing different algorithms as it avoids the need to determine numerical convergence. Default: NULL

Value

An S3 object of class `glmpca` with copies of input components `optimizer`, `minibatch`, `ctl`,`X`, and `Z`, along with the following additional fitted components:

factors a matrix `U` whose rows match the columns (observations) of `Y`. It is analogous to the principal components in PCA. Each column of the factors matrix is a different latent dimension.

loadings a matrix `V` whose rows match the rows (features/dimensions) of `Y`. It is analogous to loadings in PCA. Each column of the loadings matrix is a different latent dimension.

coefX a matrix `A` of coefficients for the observation-specific covariates matrix `X`. Each row of coefX corresponds to a row of `Y` and each column corresponds to a column of `X`. The first column of coefX contains feature-specific intercepts which are included by default.

coefZ a matrix `G` of coefficients for the feature-specific covariates matrix `Z`. Each row of coefZ corresponds to a column of `Y` and each column corresponds to a column of `Z`. By default no such covariates are included and this is returned as NULL.

dev a vector of deviance values. The length of the vector is the number of iterations it took for GLM-PCA's optimizer to converge. The deviance should generally decrease over time. If it fluctuates wildly, this often indicates numerical instability, which can be improved by decreasing the learning rate or increasing the penalty, see `ctl`.

dev_smooth a locally smoothed version of `dev` that may be easier to visualize when `minibatch='stochastic'`.

glmpca_family an S3 object of class glmpca_family. This is a minor extension to the family or negative.binomial object used by functions like glm and glm.nb. It is basically a list with various internal functions and parameters needed to optimize the GLM-PCA objective function. For the negative binomial case, it also contains the final estimated value of the overdispersion parameter (`nb_theta`).

offsets For Poisson and negative binomial families, the offsets are the logarithmically transformed size factors. These are needed to compute the predicted mean values.

References

Savarese P, McAllester D, Babu S, and Maire M (2020). Domain-independent Dominance of Adaptive Methods. *arXiv* `https://arxiv.org/abs/1912.01823`

Townes FW (2019). Generalized Principal Component Analysis. *arXiv* `https://arxiv.org/abs/1907.02647`

Townes FW, Hicks SC, Aryee MJ, and Irizarry RA (2019). Feature Selection and Dimension Reduction for Single Cell RNA-Seq based on a Multinomial Model. *Genome Biology* https://doi.org/10.1186/s13059-019-1861-6

See Also

predict.glmpca, prcomp, glm, logisticSVD, scry::devianceFeatureSelection, scry::nullResiduals

Examples

```
#create a simple dataset with two clusters
mu<-rep(c(.5,3),each=10)
mu<-matrix(exp(rnorm(100*20)),nrow=100)
mu[,1:10]<-mu[,1:10]*exp(rnorm(100))
clust<-rep(c(”red”,”black”),each=10)
Y<-matrix(rpois(prod(dim(mu)),mu),nrow=nrow(mu))
#visualize the latent structure
res<-glmpca(Y, 2)
factors<-res$factors
plot(factors[,1],factors[,2],col=clust,pch=19)
```

predict.glmpca *Predict Method for GLM-PCA Fits*

Description

Predict the mean matrix from a fitted generalized principal component analysis model object.

Usage

```
## S3 method for class 'glmpca'
predict(object, ...)
```

Arguments

`object`	a fitted object of class inheriting from `glmpca`.
`...`	additional named arguments. Currently ignored.

Details

Let `Y` be the data matrix originally used to estimate the parameters in `fit`. The GLM-PCA model regards each element of `Y` as a random sample from an exponential family distribution such as a Poisson, negative binomial, or binomial likelihood. The components of a GLM-PCA fit are combined to produce the predicted mean of this distribution at each entry of `Y`. This matrix may be regarded as a ’denoised’ version of the original data.

Value

a dense `matrix` of predicted mean values.

Warning

The predicted mean matrix returned by this function will have the same dimensions as the original data matrix and it will be dense even if the original data were sparse. This can exhaust available memory for large datasets, so use with caution.

See Also

`glmpca`, `predict.glm` with `type='response'`

Index