Package 'BiocSklearn'

October 16, 2019

Title interface to python sklearn via Rstudio reticulate

Description This package provides interfaces to selected sklearn elements, and demonstrates fault tolerant use of python modules requiring extensive iteration.

Version 1.6.0

Author Vince Carey

Suggests testthat, restfulSE, HDF5Array, BiocStyle

Depends R (>= 3.5.0), reticulate, methods, SummarizedExperiment, knitr

Imports BBmisc

Maintainer VJ Carey `<stvjc@channing.harvard.edu>`

License Artistic-2.0

LazyLoad yes

biocViews StatisticalMethod, DimensionReduction, Infrastructure

RoxygenNote 6.1.1

VignetteBuilder knitr

SystemRequirements python (>= 2.7), sklearn, numpy, pandas, h5py

git_url https://git.bioconductor.org/packages/BiocSklearn

git_branch RELEASE_3_9

git_last_commit 6682693

git_last_commit_date 2019-05-02

Date/Publication 2019-10-15

R topics documented:

h5mat 2
H5matref 2
SkDecomp-class 3
skIncrPCA 4
skIncrPPCA 4
skKMeans 5
SklearnEls 6
skPartialPCA_step 7
skPCA 7

Index **9**

`h5mat` *create a file connection to HDF5 matrix*

Description

create a file connection to HDF5 matrix

Usage

```
h5mat(file, dsname = "assay001")
```

Arguments

`file`	a pathname to an HDF5 file
`dsname`	internal name of HDF5 matrix to use

Value

instance of (S3) h5py._hl.files.File

Examples

```
fn = system.file("ban_6_17/assays.h5", package="BiocSklearn")
h5mat(fn)
```

`H5matref` *obtain an HDF5 dataset reference suitable for handling as numpy matrix*

Description

obtain an HDF5 dataset reference suitable for handling as numpy matrix

Usage

```
H5matref(filename, dsname = "assay001")
```

Arguments

`filename`	a pathname to an HDF5 file
`dsname`	internal name of HDF5 matrix to use, defaults to 'assay001'

Value

instance of (S3) "h5py._hl.dataset.Dataset"

Examples

```
fn = system.file("ban_6_17/assays.h5", package="BiocSklearn")
ban = H5matref(fn)
ban
np = import("numpy", convert=FALSE) # ensure
ban$shape
np$take(ban, 0:3, 0L)
fullpca = skPCA(ban)
dim(getTransformed(fullpca))
ta = np$take
# project samples
## Not run:   # on celaya2 this code throws errors, and
#  I have seen
# .../lib/python2.7/site-packages/sklearn/decomposition/incremental_pca.py:271: RuntimeWarning: Mean of empt
#   explained_variance[self.n_components_:].mean()
# .../lib/python2.7/site-packages/numpy/core/_methods.py:85: RuntimeWarning: invalid value encountered in dou
#   ret = ret.dtype.type(ret / rcount)
ta(ban, 0:20, 0L)$shape
st = skPartialPCA_step(ta(ban, 0:20, 0L))
st = skPartialPCA_step(ta(ban, 21:40, 0L), obj=st)
st = skPartialPCA_step(ta(ban, 41:63, 0L), obj=st)
oo = st$transform(ban)
dim(oo)
cor(oo[,1:4], getTransformed(fullpca)[,1:4])

## End(Not run) # so blocking this part of example for now
```

SkDecomp-class *container for sklearn objects and transforms*

Description

container for sklearn objects and transforms

Usage

```
## S4 method for signature 'SkDecomp'
getTransformed(x)

## S4 method for signature 'SkDecomp'
pyobj(x)
```

Arguments

x	instance of SkDecomp

Value

the getTransformed method returns a matrix

Slots

`transform` stored as R matrix

`method` string identifying method

`object` reference to the python object with decomposition components

---

`skIncrPCA` *use sklearn IncrementalPCA procedure*

Description

use sklearn IncrementalPCA procedure

Usage

```
skIncrPCA(mat, n_components, batch_size)
```

Arguments

`mat`	a matrix – can be R matrix or numpy.ndarray
`n_components`	number of PCA to retrieve
`batch_size`	number of records to use at each iteration

Value

matrix with rotation

Examples

```
irloc = system.file("csv/iris.csv", package="BiocSklearn")
irismat = SklearnEls()$np$genfromtxt(irloc, delimiter=',')
ski = skIncrPCA(irismat)
ski25 = skIncrPCA(irismat, batch_size=25L) # non-default
getTransformed(ski)[1:3,]
getTransformed(ski25)[1:3,]
```

`skIncrPPCA` *optionally fault tolerant incremental partial PCA for projection of samples from SummarizedExperiment*

Description

optionally fault tolerant incremental partial PCA for projection of samples from SummarizedExperiment

Usage

```
skIncrPPCA(se, chunksize, n_components, assayind = 1,
  picklePath = "./skIdump.pkl", matTx = force, ...)
```

Arguments

`se`	instance of SummarizedExperiment
`chunksize`	integer number of samples per step
`n_components`	integer number of PCs to compute
`assayind`	not used, assumed set to 1
`picklePath`	if non-null, incremental results saved here via sklearn.externals.joblib.dump, for each chunk. If NULL, no saving of incremental results.
`matTx`	a function defaulting to force() that accepts a matrix and returns a matrix with identical dimensions, e.g., `function(x) log(x+1)`
`...`	not used

Value

python instance of `sklearn.decomposition.incremental_pca.IncrementalPCA`

Note

Will treat samples as records and all features (rows) as attributes, projecting. to an `n_components`-dimensional space. Method will acquire chunk of assay data and transpose before computing PCA contributions. In case of crash, restore from `picklePath` using `SklearnEls()$joblib$load` after loading reticulate. You can use the `n_samples_seen_` component of the restored python reference to determine where to restart. You can manage resumption using `skPartialPCA_step`.

Examples

```
# demo SE made with TENxGenomics:
# mm = matrixSummarizedExperiment(h5path, 1:27998, 1:750)
# saveHDF5SummarizedExperiment(mm, "tenx_750")
#
if (requireNamespace("HDF5Array")) {
  se750 = HDF5Array::loadHDF5SummarizedExperiment(
     system.file("hdf5/tenx_750", package="BiocSklearn"))
  lit = skIncrPPCA(se750[, 1:50], chunksize=5, n_components=4)
  round(cor(pypc <- lit$transform(dat <- t(as.matrix(assay(se750[,1:50]))))),3)
  rpc = prcomp(dat)
  round(cor(rpc$x[,1:4], pypc), 3)
}
```

`skKMeans` *interface to sklearn.cluster.KMeans with attention to direct work with HDF5*

Description

interface to sklearn.cluster.KMeans with attention to direct work with HDF5

Usage

```
skKMeans(mat, ...)
```

Arguments

`mat`	a matrix-like datum or reference to such
`...`	arguments to sklearn.cluster.KMeans

Note

You can use ‘py_help(SklearnEls()$skcl$KMeans)‘ to get python documentation on parameters and return structure.

Examples

```
# start with numpy array reference as data
irloc = system.file("csv/iris.csv", package="BiocSklearn")
skels = SklearnEls()
irismat = skels$np$genfromtxt(irloc, delimiter=',')
ans = skKMeans(irismat, n_clusters=2L)
names(ans) # names of available result components
table(iris$Species, ans$labels_)
# now use an HDF5 reference
irh5 = system.file("hdf5/irmat.h5", package="BiocSklearn")
fref = skels$h5py$File(irh5)
ds = fref$`__getitem__`("quants") # thanks Samuela Pollack!
ans2 = skKMeans(skels$np$array(ds)$T, n_clusters=2L) # HDF5 matrix is transposed relative to python array layou
table(ans$labels_, ans2$labels_)
ans3 = skKMeans(skels$np$array(ds)$T,
   n_clusters=8L, max_iter=200L,
   algorithm="full", random_state=20L)
```

`SklearnEls` *mediate access to python modules from sklearn.decomposition*

Description

mediate access to python modules from sklearn.decomposition

Usage

```
SklearnEls()
```

Value

list of (S3) "python.builtin.module"

Note

Returns a list with elements np (numpy), pd (pandas), h5py (h5py), skd (sklearn.decomposition), joblib (sklearn.externals.joblib), each referring to python modules.

Examples

```
els = SklearnEls()
names(els$skd) # slow at first
# try py_help(els$skd$PCA) # etc.
```

`skPartialPCA_step` *take a step in sklearn IncrementalPCA partial fit procedure*

Description

take a step in sklearn IncrementalPCA partial fit procedure

Usage

```
skPartialPCA_step(mat, n_components, obj)
```

Arguments

`mat`	a matrix – can be R matrix or numpy.ndarray
`n_components`	number of PCA to retrieve
`obj`	sklearn.decomposition.IncrementalPCA instance

Value

trained IncrementalPCA reference, to which ’transform’ method can be applied to obtain projection for any compliant input

Note

if obj is missing, the process is initialized with the matrix provided

Examples

```
irloc = system.file("csv/iris.csv", package="BiocSklearn")
irismat = SklearnEls()$np$genfromtxt(irloc, delimiter=',')
ta = SklearnEls()$np$take
ipc = skPartialPCA_step(ta(irismat,0:49,0L))
ipc = skPartialPCA_step(ta(irismat,50:99,0L), obj=ipc)
ipc = skPartialPCA_step(ta(irismat,100:149,0L), obj=ipc)
head(names(ipc))
ipc$transform(ta(irismat,0:5,0L))
fullproj = ipc$transform(irismat)
fullpc = prcomp(data.matrix(iris[,1:4]))$x
round(cor(fullpc,fullproj),3)
```

`skPCA` *use sklearn PCA procedure*

Description

use sklearn PCA procedure

Usage

```
skPCA(mat, ...)
```

Arguments

`mat`	a matrix – can be R matrix or numpy.ndarray
`...`	additional parameters passed to sklearn.decomposition.PCA, for additional information use `py_help(SklearnEls()$sk$PCA)`

Value

matrix with rotation

Note

If no additional arguments are passed, all defaults are used.

Examples

```
irloc = system.file("csv/iris.csv", package="BiocSklearn")
irismat = SklearnEls()$np$genfromtxt(irloc, delimiter=',')
skpi = skPCA(irismat)
getTransformed(skpi)[1:5,]
```

Index

getTransformed *(SkDecomp-class)*, 3
getTransformed,SkDecomp-method *(SkDecomp-class)*, 3

h5mat, 2
H5matref, 2

pyobj *(SkDecomp-class)*, 3
pyobj,SkDecomp-method *(SkDecomp-class)*, 3

SkDecomp-class, 3
skIncrPCA, 4
skIncrPPCA, 4
skIncrPPCA,SummarizedExperiment-method *(skIncrPPCA)*, 4
skKMeans, 5
SklearnEls, 6
skPartialPCA_step, 7
skPCA, 7