pls

index
/home/mpastor/soft/eTAM/src/pls.py

```
# -*- coding: utf-8 -*-
#
#    Description    PLS toolkit using NIPALS algorithm
#
#
#    Authors:       Manuel Pastor (manuel.pastor@upf.edu)
#
#    (c) PhI 2013
```

Modules

numpy sys

Classes

pls

class **pls**

Methods defined here:

__init__(self)

build(self, X, Y, targetA=0, targetSSX=0.0)

```
Build a new PLS model with the X and Y numpy matrice provided using NIPALS algorithm

The dimensionality of the model can be defined either providing
1. directly the number of LV to extract (targetA)
2. the fraction of SSX that the model will explain (targetSSX)

The X and Y matrices are centered but no other scaling transform is applied

Does not return anything, but updates internals vectors and variables
```

center(self, X)

```
Centers the numpy matrix (X) provided as argument
```

computeSS(self, X, Y)

```
Computes the Sum-Of-Squares for provided X and Y matrices

Returns
SSX:    sum of squates of the X matrix
SSY:    sum of squares of the Y matrix
d:      vector with the SSX for every object
```

deflateLV(self, X, Y, t, p, c)

```
Deflates both the X and Y matrices, using the provided t, p and c vectors

Returns deflated X and Y
```

extractLV(self, X, Y)

```
Extracts a single LV from the provided X and Y matrices using NIPALS algorithm

This method assumes that both X and Y are centered. No deflation is applied

Returns
t:    vector of scores
p:    vector of loadings
w:    vector of weights
c:    inner relationship
```

getLOO(self, X, Y, x, A)

```
Builds a model of A dimension with the provided X and Y matrices, yielding a prediction y for the query object x.
Typically used as inner loop in LOO CV method.

Notice that both X and Y must be centered, while x must have been centered with the model averages

Returns the predicted y value for the query object x
```

loadModel(self, filename)

```
Loads the whole model from a binary file in numpy .npy format
```

project(self, x, A)

```
projects query object x into current model using A LV

Returns
y:    vector of predicted Y values using growing number of LV
```

```
t:    vector of scores
d:    SSX for every dimension
```

saveModel(self, filename)

```
Saves the whole model to a binary file in numpy .npy format
```

validateLOO(self, A)

```
Validates A dimensions of an already built PLS model, using Leave-One-Out cross-validation

Returns nothing. The results of the cv (SSY, SDEP and Q2) are stored internally
```

Functions

readData(filename)

```
Reads numpy X and Y matrices from a file in GOLPE .dat format, asuming a single Y value at the end

Returns X and Y as a numpy matrices
```