Line H. Clemmensen
Section of Statistics and Data Analysis
DTU Compute

Exercises 02582
Module 8
Spring 2019

March 25, 2019

Topics: Principal Component Analysis (PCA), Sparse PCA, Partial Least squares (PLS)

Exercises:

1 Apply Principal Component Analysis (PCA) to the face data set in `faces.mat`

(a) Load the face shape data in the file `faces.mat`

(b) Compute the mean shape and center the data. Plot the mean shape using

- Matlab: The function `drawshape.m`. The argument "`conlist`" is loaded along with the data.
- R: Use the function `drawshape(shape, conlist,firstplot,lty,col,lwd)` contained in the file `drawshape.R`, open the file to check how to specify the arguments. Use `source("drawshape.R")` before calling it.
- Python: Use the function `drawShape.py`.

(c) Compute a principal component analysis of the data. Try using both an eigen value decomposition (EVD) and a singular value decomposition (SVD). Remember that the EVD is computed on the correlation or covariance matrix and the SVD on the data matrix itself.

- Matlab: Use the functions `eig` and `svd`. Note that Matlab sorts the eigenvalues in descending order using svd, and ascending order using eig (by default). The command `svd(X, 'econ')` computes the so-called economy-size svd, which only contains singular vectors (and singular values) corresponding to non-zero singular values. This variant of svd is very handy when $p > n$.

- R: Use the functions `eigen` and `svd`. Eigenvalues and singular values are sorted decreasingly. Note that you need to specify the number of right and left singular vectors yourself to make the method computationally effective. Set the number of left singular vectors (nu) equal to zero.
- Python: Use `numpy.linalg.svd` and `numpy.linalg.eigh`. Set `full_matrices=FALSE` to get the economically sized singular decomposition. Note that the singular values are returned in descending order and the eigenvalues are returned in ascending order., or if you use `numpy.linalg.eig` the eigenvalues are returned unsorted.

(d) Plot the first mode of variation. This will provide a view of the most important variation in the data set. Let the mean face be the origin, we will use the mean face as a reference. Plot the mean face in black.

(e) Compute the face obtained by moving from the mean face along the first principal axis (first column of the loading matrix) out to a distance of +2.5 standard deviations ($\mu + 2.5\sigma_{l_1}$). The standard deviations can be obtained from the singular values (see the slides on how this is done). Plot the resulting face in red.

(f) Repeat this procedure to obtain a face at -2.5 standard deviations from the mean face. Plot this in blue.

(g) Explore the first few modes of variation using

- Matlab: The application `shape_inspector.m`. Start it using the call `shape_inspector(L(:,1:8),  d(1:8), X_mean, conlist, 37, 116, 8)` where L is the loading matrix, d is a vector of variances and X_{mean} is the mean shape.
- R: The function `shape_inspector(mu,V,sigma2,lty,col,lwd)` contained in the file `drawshape.R`, open the file to check how to specify the arguments. (Use `source("drawshape.R")` before calling it).
- Python: The `ShapeInspectorGUI.py` application.
  Read the file `usingShapeInspectorPY.txt` for instructions.

2 Extract Sparse Principal Components for the face data using three different methods:

(a) Compute a sparse PCA by thresholding all loadings from a regular PCA with absolute value less than 0.15. Now that the loading matrix has been changed, the scores matrix must be recomputed. How can you use this new scores matrix to compute the variance of the data along each principal axis? Use this result to plot the first mode of variation of the threshold SPCA for -2.5, 0 and +2.5 standard deviations. Investigate what has happened with the uncorrelatedness of the loadings and scores matrices of regular PCA. Are these properties fulfilled here?

(b) Use the Varimax criterion. Experiment with the number of columns from the loading matrix to rotate, and see how this affects sparsity. Again, what has happened with the uncorrelatedness of the scores and loading matrices? See the code listings for functions which perform Varimax rotation.

(c) Compute the first sparse principal loading vector using the Elastic net (remember to normalize to unit length). Start by using 10 non-zero loadings. Plot this solution and try different number of nonzero components. Can you put a meaningful anatomical label on this deformation? Would you be able to label the first mode of variation from regular PCA?

3 Apply Partial Least Squares regression to the sand data. *(Optional - only a Matlab solution provided)*

a Load data `sand.mat` and run a cross validation of partial least squares regression to decide the number of components that is adequate to model the sand data. Plot both the cross validation error and the percentage of explained variance in y to determine the number of components. See the code listings for useful PLS implementations.

b How would you plot the coefficients of the final PLS regression model (β)? Which variables are important for the prediction of y? (In terms of loadings, remember the pitfall of PCA concerning scaled vs non-scaled loadings - this also holds for PLS).

Line H. Clemmensen
Section of Statistics and Data Analysis
DTU Compute

Exercises 02582
Module 8
Spring 2019

March 25, 2019

Topics: Principal Component Analysis (PCA), Sparse PCA, Partial Least squares (PLS)

Resources for this exercise:

Listing 1: Resources in Matlab

```
svd(X, 'econ') % economical SVD
eig(XX) % eigen value decomposition
shape_inspector(L(:,1:8),  d(1:8), X_mean, conlist, 37, 116, 8);
        % inspect the modes of variation
L_varimax = rotatefactors(L); % Varimax rotation of PCs
v1_en = elasticnet(X, S(:,1), 1e6, -10, 0); % sparse PC using elastic net
v1_en = v1_en/sqrt(v1_en'*v1_en); % normalize to unit length
plsregress % PLS regression
```

Listing 2: Resources in R

```
source("drawshape.R")
library(manipulate)
library(R.matlab) # to be able to load .mat file in R
library(corrplot)
library(glmnet) # to perform elastic net

drawshape(mu,conlist,T,lwd=2) # plot face shape
svd(X) # svd
eigen(XX, TRUE) # eigen value decomposition of symmetric matrix
svd.scree(svdlist, ...) # make a scree plot
shape\_inspector(mu,V,sigma2,lty,col,lwd)
        # inspect the modes of variation
varimax(V[,1:12]) # varimax rotation
cv.glmnet(as.matrix(Xc), ...) # perform elastic net regression
dat <- readMat(file.path('sand.mat')) # sand data set (X: 59x2016)
```

Listing 3: Resources in Python

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn import linear_model
from sklearn import preprocessing
from ShapeInspectorGUI import runGUI, allDataC

drawShape(mu, conlist, ax) # plot a face shape
np.linalg.svd(X, full_matrices=False, compute_uv=True) # svd
np.linalg.eigh(XX) # eigen value decomposition of symmetric matrix
np.linalg.eig(XX) # eigen value decomposition of matrix
linear_model.ElasticNet(alpha = 0.0001, l1_ratio = 0.1, ...)
        # Ratio is 0 for only l2 penalty
```

End of exercise