Principal Component Analysis

Nisha Akole, G V V Sharma*

Contents

***Abstract*—This manual provides a brief description on how to implement Principal Component Analysis from scratch and use it for dimensionality reduction of data.**

1. Objective

Our objective is to implement PCA and reduce dimensionality of data which gives better understanding of data visually.

2. Load Dataset

The dataset used for PCA is available at the following link. Download all the data file in the folder where you want to write code for PCA.

https://github.com/prabhatrai111/Commensal−Radar

```
import numpy as np
import scipy.io as sio
from sklearn.utils.extmath import randomized_svd

mat_contents = sio.loadmat('data_all.mat')
X_data = mat_contents['data_all']
```

3. About PCA

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables (entitles each of which takes on various numerical values) into a set of values of linearly uncorrelated variables called principal components.

There are four main operations in the PCA:

1) Pre-processing
2) Co-variance Matrix
3) Eigen Vectors
4) Feature Vector and Plot

*The authors are with the Department of Electrical Engineering, Indian Institute of Technology, Hyderabad 502285 India e-mail: gadepall@iith.ac.in.

4. Pre-processing

Take the whole dataset ignoring the labels. Feature scaling is an important task in PCA because various scales of feature will affect and we may not get an optimized output. The functions StandardScaler or MinMaxScaler will standardize the features by making mean = 0 and variance = 1. These functions are imported from sklearn.preprocessing. StandardScaler uses formula such as

$$z = (X - \mu)/\sigma$$

where, μ =Mean of data X
σ =Standard Deviation of data X

X = np.matrix(X_data)
$\mu = X.mean(0)$
$\sigma = X.std(0)$
X_std= $(X - \mu)/\sigma$

X.mean(0) and X.std(0) will give columnwise mean and standard deviation of our data matrix.

5. Compute Covariance Matrix

Computing the covariance matrix will help us to understand the relationship between the variables. If variables are highly correlated, they contain redundant information. Covariance matrix will be symmetric with diagonal values as a variance of the corresponding element. If variables are increasing or decreasing together, then its positive covariance.If one variable is increasing and other is decreasing, sign of covariance will be negative.

```
X_cov = np.cov(X_std)
```

6. Compute Eigenvectors

The diagonal values of covariance matrix are the eigenvalues of a covariance matrix where large eigenvalues correspond to large variance. Eigenvector of covariance matrix is the axes along with data has maximum variation. Eigenvalues and