Principle Component Analysis

Sunday, 3 March 2024 10:50 AM

Very Important

** PCA analysis is a technique that can be used to simplify a dataset

** PCA can be used for reducing dimensionality by eliminating the later principal components

Given the data table, reduce the dimension from 2 to 1 using the principle component Analysis (PCA) algorithm

feature	Example 1	Example 2	Example 3	Example 4
X_1	4	8	13	7
X_2	11	4	5	14

Step 1 : calculate the mean

F	Ex 1	Ex 2	Ex 3	Ex 4
$\bar{X}_1$	4	8	13	7
$\bar{X}_2$	11	4	5	14

calculate the mean $\bar{X}_1 = \frac{1}{4}(4+8+13+7) = 8$

$$\bar{X}_2 = \frac{1}{4}(11+4+5+14) = 8.5$$

Step 2 : calculation of covariance matrix

$$S = \begin{bmatrix} cov(X_1, X_1) & cov(X_1, X_2) \\ cov(X_2, X_1) & cov(X_2, X_2) \end{bmatrix}$$

$N = 4$ (Because of 4 values)

$\bar{X}_1$ is mean

X_{1K} is value of X_1 at K index

$$conv(X_1, X_1) = \frac{1}{N-1} \sum_{K=1}^{N} (X_{1K} - \bar{X}_1)(X_{1K} - \bar{X}_1)$$

$$= \frac{1}{3}\left((4-8)^2 + (8-8)^2 + (13-8)^2 + (7-8)^2\right)$$

$$= 14$$

$$conv(X_1, X_2) = \frac{1}{N-1} \sum_{K=1}^{N} (X_{1K} - \bar{X}_1)(X_{2K} - \bar{X}_2)$$

$$= \frac{1}{3}\big((4-8)(11-8.5) + (8-8)(4-8.5)$$

$$+ (13-8)(5-8.5) + (7-8)(14-8.5)\big)$$

$$= -11$$

$$\mathrm{cov}(x_2, x_1) = \mathrm{cov}(x_1, x_2)$$
$$= -11$$

$$\mathrm{cov}(x_2, x_2) = \frac{1}{N-1} \sum_{K=1}^{N} (x_{2K} - \bar{x}_2)(x_{2K} - \bar{x}_2)$$

$$= \frac{1}{3} \left((11-8.5)^2 + (4-8.5)^2 + (5-8.5)^2 + (14-8.5)^2 \right)$$
$$= 23$$

$$S = \begin{bmatrix} \mathrm{cov}(x_1, x_1) & \mathrm{cov}(x_1, x_2) \\ \mathrm{cov}(x_2, x_1) & \mathrm{cov}(x_2, x_2) \end{bmatrix}$$

$$= \begin{bmatrix} 14 & -11 \\ -11 & 23 \end{bmatrix}$$

Step 3 : Calculate the Eigenvalues of the Covariance matrix

The characteristic equation of the covariance matrix is,

$$0 = \det(S - \lambda I)$$

$$= \begin{bmatrix} 14-\lambda & -11 \\ -11 & 23-\lambda \end{bmatrix}$$

$$= (14-\lambda)(23-\lambda) - (-11)(-11)$$

$$= \lambda^2 - 37\lambda + 201$$

$$\lambda = \frac{1}{2}(37 \pm \sqrt{565})$$

Roots of Quadratic Equation

$$\boxed{x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}}$$

$$\lambda_1 = 30.3849$$

$$\lambda_2 = 6.6151$$

Step 4: Computation of the Eigenvector

$$U = \begin{bmatrix} U_1 \\ U_2 \end{bmatrix}$$

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix} = (S - \lambda I)\, U$$

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 14-\lambda & -11 \\ -11 & 23-\lambda \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \end{bmatrix}$$

$$= \begin{bmatrix} (14-\lambda)U_1 - 11U_2 \\ -11U_1 + (23-\lambda)U_2 \end{bmatrix}$$

$$(14-\lambda)u_1 = 11 u_2$$

$$\boxed{\frac{u_1}{11} = \frac{u_2}{14-\lambda} = t}$$

$u_1 = 11t$, $u_2 = (14-\lambda)t$

$$U_1 = \begin{bmatrix} 11 \\ 14-\lambda_1 \end{bmatrix}$$

- To find a $\boxed{\text{unit eigenvector}}$, we compute the length of U_1 which is given by,

$$\|U_1\| = \sqrt{11^2 + (14-\lambda_1)^2}$$

$$= \sqrt{11^2 + (14-30.3849)^2}$$

$\lambda = 30.3849$

$$= 19.7348$$

0.83024
13.2490

$$e_1 = \begin{bmatrix} 11/\|U_1\| \\ (14-\lambda_1)/\|U_1\| \end{bmatrix}$$

$$= \begin{bmatrix} 11/19.7348 \\ (14-30.3849)/19.7348 \end{bmatrix}$$

$$= \begin{bmatrix} 0.5574 \\ -0.8303 \end{bmatrix}$$

$$e_2 = \begin{bmatrix} 0.8303 \\ 0.5574 \end{bmatrix}$$

Step 5
Computation of first principal components

$$= e_1^T \begin{bmatrix} X_{1k} - \bar{X}_1 \\ X_{2k} - \bar{X}_2 \end{bmatrix}$$

$$= [0.5574 \quad -0.8303] \begin{bmatrix} X_{11} - \bar{X}_1 \\ X_{21} - \bar{X}_2 \end{bmatrix}$$

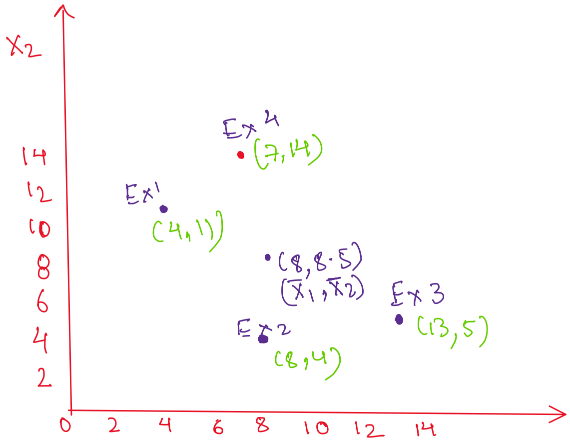
$$= 0.5574(X_{11} - \bar{X}_1) - 0.8303(X_{21} - \bar{X}_2)$$

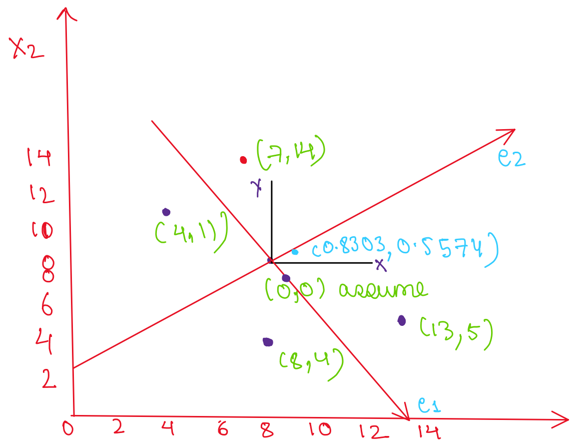
$$= 0.5574(4-8) - 0.8303(11-8.5)$$

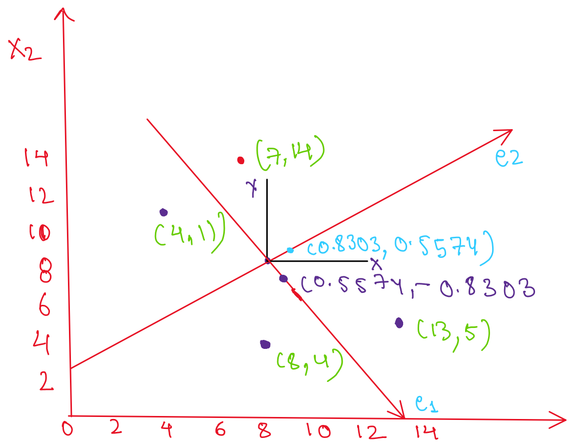
$$= -4.30535$$

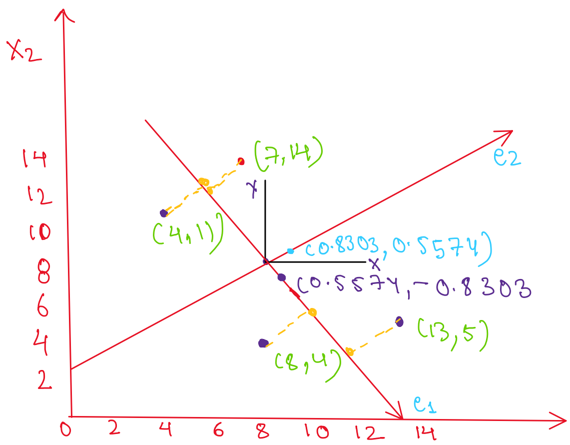
Feature	Ex 1	Ex 2	Ex 3	Ex 4
X_1	4	8	13	7
X_2	11	4	5	14
First Principle component	-4.3052	3.7361	5.6928	-5.1238

Step 6: Geometrical meaning of first Principle components.









Normal PCA

```
from sklearn.decomposition import PCA
import pandas as pd
import numpy as np

df = df.read_csv('pca.csv')

print (df)

pca = PCA(n_components=2)

pca_model = pca.fit_transform(df)

print (pca_model)
```

Randomized PCA

```
from sklearn.decomposition import PCA
import pandas as pd
import numpy as np

df = df.read_csv('pca.csv')

print (df)

                                          # SVD is si
random = PCA(n_components=2, svd_solver=':

pcaf model = random.fit_transform(df)

print (pcaf model)
```

Randomized PCA performs better on more r
of dimensions.

Data Preprocessing using Sklearn

The preprocessing module from scikit-learn offers several functionalities like encoding the data to different formats, splitting the data into training and test sets, and many more

① # LabelBinarizer

for Example Some list is given like

[1, 2, 6, 4, 2]

here there can be five labels
Their are total 5 Elements) and 4 are distinct
For Ex 1 is least value & 6 is the highest value
So,
1 can be encoded as [1, 0, 0, 0]

6 can be encoded as [0, 0, 0, 1]

2 can be Encoded as [0, 1, 0 0]

4 can be Encoded as [0, 0, 1, 0]

Python_Code

```
from Sklearn.preprocessing import LabelBinarizer

# Creating object that represents label Binarizer
lb = LabelBinarizer()
label = lb.fit_transform([1, 2, 6, 4, 3])
print(label)
```

output

```
[[1 0 0 0 0]
 [0 1 0 0 0]
 [0 0 0 0 1]
 [0 0 0 1 0]
 [0 0 1 0 0]]
```

② creating labels for categorical data (data that is non-numeric)

```
from sklearn import preprocessing

data = ['Apple', 'Mango', 'Banana', 'Chikoo', 'Jackfruit']

labels = ['T', 'F', 'F', 'T', 'T']

# Creating object that represents the label encoder

encode = preprocessing.LabelEncoder()

data_e = encode.fit_transform(data)
labels_e = encode.fit_transform(labels

print(data_e)
```

Output
[0 4 1 2 3] ← (Values as in Alpha

print (labels_e)

[1 0 0 1]