hw9

April 30, 2018

In [45]:
```
import numpy as np
from numpy import linalg as LA
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA, FastICA,KernelPCA,TruncatedSVD
```

In [46]:
```
XMat=np.array([(3,2,1),(2,4,5),(1,2,3),(0,2,5)])
average= np.mean(XMat,axis=0) #Find the sample mean.
average
```

Out[46]:
```
array([ 1.5,  2.5,  3.5])
```

In [47]:
```
m, n = np.shape(XMat)
data_adjust = []
avgs = np.tile(average, (m, 1))
data_adjust = XMat - avgs
covX = np.cov(data_adjust.T)   #Find the sample covariance matrix Q.
covX
```

Out[47]:
```
array([[ 1.66666667,  0.33333333, -1.66666667],
       [ 0.33333333,  1.        ,  1.        ],
       [-1.66666667,  1.        ,  3.66666667]])
```

In [48]:
```
#Find the eigenvalues and eigenvectors.
featValue, featVec=  np.linalg.eig(covX)
featValue, featVec
```

Out[48]:
```
(array([ 4.74888619,  1.56450706,  0.01994008]),
 array([[-0.45056922, -0.66677184, -0.59363515],
        [ 0.19247228, -0.72187235,  0.66472154],
        [ 0.87174641, -0.18524476, -0.45358856]]))
```

In [49]:
```
#Reconstruct the original samples from the PCA coefficients.
#Approximate the samples using principle components corresponding to the two largest
index = np.argsort(-featValue)
finalData = []
selectVec = np.matrix(featVec.T[index[:2]])
finalData = data_adjust * selectVec.T
reconData = (finalData * selectVec) + average
finalData,reconData
```

```
Out[49]: (matrix([[-2.95145599, -0.17610969],
                  [ 1.37104342, -1.69406159],
                  [-0.30682473,  0.78694448],
                  [ 1.8872373 ,  1.0832268 ]]),
          matrix([[ 2.94726021,  2.05905526,  0.95970224],
                  [ 2.0118026 ,  3.98678407,  5.0090182 ],
                  [ 1.11353336,  1.87287129,  3.0867493 ],
                  [-0.07259617,  2.08128939,  4.94453025]]))

In [50]: U,S,V = np.linalg.svd(XMat, full_matrices=False)
         lam = S**2
         PoV = np.cumsum(lam)/np.sum(lam)
         PoV

Out[50]: array([ 0.89793936,  0.99755596,  1.        ])
```

Introml HW9

Tianyunyang Wang tw1719 N14855366

2.

- (a)

The covariance matrix is computed as :
$Q_{kl} = \frac{1}{N}\sum_{i=1}^{N}(x_{ik} - \bar{x}_k)(x_{il} - \bar{x}_l)$
and
$Q_{lk} = \frac{1}{N}\sum_{i=1}^{N}(x_{il} - \bar{x}_l)(x_{ik} - \bar{x}_k)$
Since
$cov(l,k) = cov(k,l)$
Matrix is symmetrical about diagonal.
When it comes to the coefficient data, since the coefficients are orthogonal,
$cov(l,k) = cov(k,l) = 0$
therefore, the covariance matrix of the coefficient data is diagonal.

- (b)

According to
theorem 1.The eigenvectors corresponding to different eigenvalues of real symmetric matrices must be orthogonal
and
theorem 2. Let the eigenvector λ multiplicity be r, then there must be r linearly independent eigenvectors corresponding to λ, so the r eigenvector units can be orthogonalized.

From the above two, we can see that a real symmetric matrix with n rows and n columns can certainly find n units of orthogonal eigenvectors. Let these n eigenvectors be $e_1, e_2, \cdots, e_n$, and we compose them according to the columns:
$E = (e_1, e_2, \cdots, e_n)$
Then we have the following conclusions about the covariance matrix C:
$E^T CE = \Lambda =$

$$\begin{matrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \cdots & \\ & & & \lambda_n \end{matrix}$$

where λ_i is the eigenvalues and they equal to the variance according to (a).

3.

When facing a large bunch of data (as under most circumstances in reality), it is more beneficial if we keep a subset of coefficients. First of all, keep all the coefficients could lead to the overfitting problem. Sometimes the coefficients are also correlated and then cause us a lot more trouble. Thus we prefer using a subset of coefficients, When using PCA, we should keep the coefficients that leads to the least projection meanwhile, maintains the most information so we need to find the largest magnitude of coefficients.

4

If we directly use coefficients as features, too much information will be ignored. We have to reconstruct the original data from a few largest coefficients and then use these features to train the machine.