Exercise Principal Component Analysis

June 29, 2019

1 Principal Component Analysis of Height and Weight Data

Weight and height are strongly correlated.

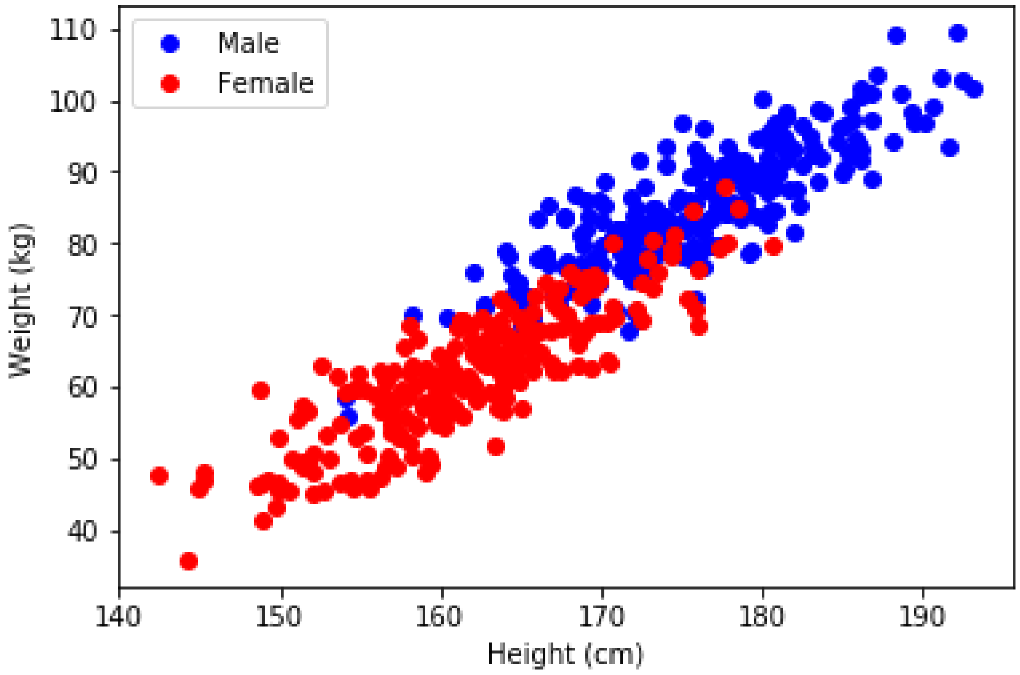
But men and women have different weights heights.

In this exercise you will investigate what information PCA extracts.

```
In [5]: import matplotlib.pyplot as pl
        import pandas as pd
        %matplotlib inline

        df = pd.read_csv('weights-heights-metric.csv')

        pl.plot(df.loc[df['Gender']=='Male', 'Height'], df.loc[df['Gender']=='Male', 'Weight']
                df.loc[df['Gender']=='Female', 'Height'], df.loc[df['Gender']=='Female', 'Weig
        pl.xlabel('Height (cm)')
        pl.ylabel('Weight (kg)')
        pl.legend(['Male','Female']);
```



2 Assignment 1:

Compute PCA on a data set of weight and height of men and women.
Plot the principal components of the data.
Which variance is captured by each principal component?

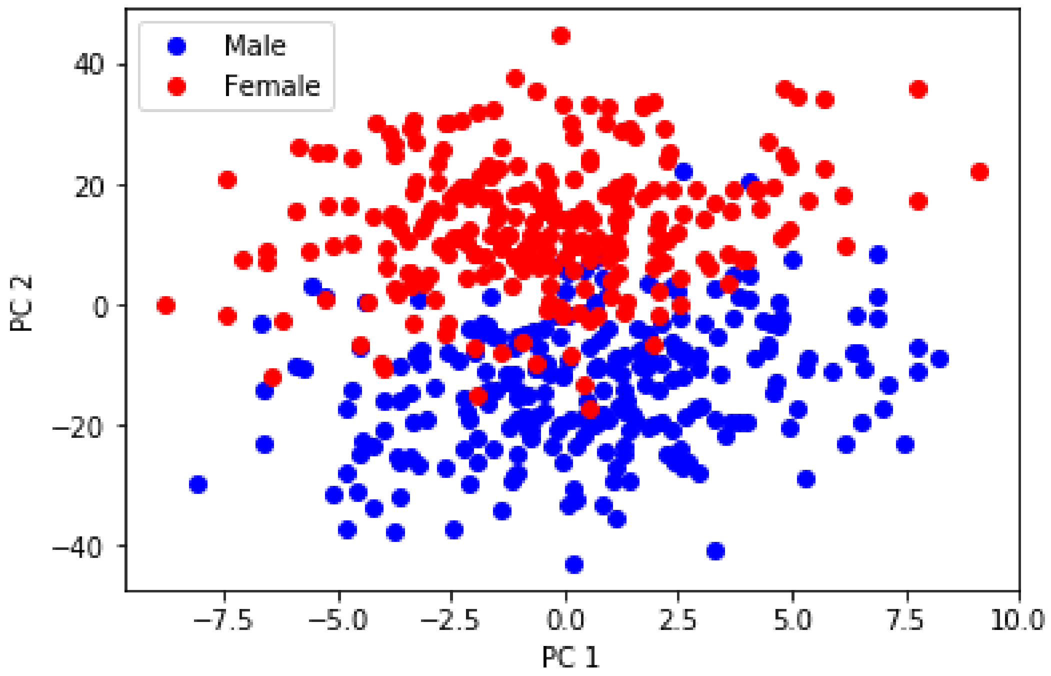
```
In [19]: import numpy as np
         import scipy as sp

         def pca(X, ncomp=2):
             # subtract the mean
             mu = X.mean(axis=0)
             Xcentered = X - mu
             # compute covariance matrix
             C = Xcentered.T @ Xcentered * (1 / (X.shape[0]-1))
             # compute eigenvectors of covariance matrix
             V, U = np.linalg.eig(C)
             return (Xcentered @ U)
```

```
In [23]: # perform PCA
         X = df[['Height','Weight']].values
         Xpca = pca(X)

         # plot data in PCA space
         pl.plot(Xpca[df['Gender']=='Male',0],Xpca[df['Gender']=='Male',1],'bo',
                Xpca[df['Gender']=='Female',0],Xpca[df['Gender']=='Female',1],'ro')
         pl.xlabel('PC 1')
         pl.ylabel('PC 2')
         pl.legend(['Male','Female'])
```

```
Out[23]: <matplotlib.legend.Legend at 0x22748df41d0>
```



3 Assignment 2:

Compute the covariance matrix of the original data.
Compute the covariance matrix of the data in the PCA space.
Print the covariance of the first and second dimension of the original and the PCA space.

```
In [32]: # Reference for testing
         C = np.cov(X.T)
         Cpca = np.cov(Xpca.T)
         print(f"Numpy:\nCov width/height: {C[0,1]}\nCov first PC/second PC {Cpca[0,1]}")

         # Manual computation
         def cov(X):
             Xcen = X - X.mean(axis=0)
             C = Xcen.T @ Xcen * (1/(X.shape[0]-1))
             return C
         C = cov(X)
         Cpca = cov(Xpca)
         print(f"Manual:\nCov width/height: {C[0,1]}\nCov first PC/second PC {Cpca[0,1]}")
```

```
Numpy:
Cov width/height: 133.0227029932385
Cov first PC/second PC 1.708720005835912e-14
Manual:
```

```
Cov width/height: 133.0227029932385
Cov first PC/second PC 1.708720005835912e-14
```