Homework 3

Assignment Info

Homework #: HW3
Description: PCA
Course: EN.553.636 Introduction to Data Science
Semester: Spring 2023, Homewood Campus
Instructor: Tamas Budavari
TA: Matthew Tivnan
Date: February 25, 2023

Student Info

Name: Amir Hossein Daraie
JHED-ID: adaraie1
Email: [adaraie1@jhu.edu (mailto:adaraie1@jhu.edu)](mailto:adaraie1@jhu.edu)

We load the `music_scaled.csv` dataset ([source (https://archive.ics.uci.edu/ml/datasets/Geographical+Original+of+Music#)](https://archive.ics.uci.edu/ml/datasets/Geographical+Original+of+Music#)). The dataset contains a sample of traditional songs from different cultures. Features F1 to F68 are quantitative summaries of the songs from audio analysis software. These features have been subject to standard scaling. They are stored as predictors in `X`. The latitudes of the countries from which the songs originate are stored as a target variable `y`.

In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
import numpy as np

data = pd.read_csv("music_scaled.csv")
X = data.iloc[:,:68]
y = data["Latitude"]
display(X)
```

	F1	F2	F3	F4	F5	F6	F7	F8	
0	-1.094000	-1.280592	2.806926	-0.097576	-0.791472	2.440896	0.003710	-0.864715	0.7
1	-1.285544	-0.940198	-0.721321	-0.172044	-2.127893	2.549762	1.365750	0.489953	1.6
2	0.503962	0.497136	-0.319168	0.330719	-0.398783	-0.749429	-2.380589	0.951098	-0.4
3	-1.119978	0.696697	0.612882	-0.983295	1.333148	1.557607	-0.999593	-1.067051	0.8
4	1.256214	1.066239	0.984965	-0.312513	-2.111077	0.009980	0.509741	0.837961	0.4
...	...	...	...	...	...	...	...	...	
1054	0.089066	0.045324	-0.527161	0.274475	0.113220	-1.011250	-1.071873	-1.533993	-3.4
1055	0.558342	0.274368	-0.356357	-0.410656	0.710800	-0.240603	-0.370162	0.870655	-1.4
1056	0.880180	0.668551	0.115084	-0.373637	0.848817	-0.859218	-0.823580	0.051974	-1.1
1057	-0.990084	-1.094018	3.645894	-0.474362	-1.129270	0.188591	0.092048	0.252823	-0.3
1058	-0.133272	-0.074305	-0.554388	-0.582548	0.307454	-0.958630	-1.523819	-1.589983	-3.6

1059 rows × 68 columns

Question 1 (2 pts)

a) Perform linear regression of **y** on **X** using `sklearn.linear_model.LinearRegression` when the regressors consist of:

- F1 only;
- F1 and F2 only;
- F1, F2, and F3 only;
- F1, F2, F3, ..., and F68.

b) In each of the above 4 cases, print the estimated coefficient for **F1**.

In [3]:

```
model = LinearRegression()
for i in range(X.shape[1]):
    coeffs_i = model.fit(X.iloc[:,:i+1],y)
    print(f'Coeff F1 for LinearRegression model trained on X[:,0:{i+1:2}] is {coeffs
```

```
Coeff F1 for LinearRegression model trained on X[:,0: 1] is 0.659734
Coeff F1 for LinearRegression model trained on X[:,0: 2] is 1.669923
Coeff F1 for LinearRegression model trained on X[:,0: 3] is 2.222158
Coeff F1 for LinearRegression model trained on X[:,0: 4] is 2.324427
Coeff F1 for LinearRegression model trained on X[:,0: 5] is 2.268298
Coeff F1 for LinearRegression model trained on X[:,0: 6] is 2.845235
Coeff F1 for LinearRegression model trained on X[:,0: 7] is 3.332087
Coeff F1 for LinearRegression model trained on X[:,0: 8] is 3.356679
Coeff F1 for LinearRegression model trained on X[:,0: 9] is 3.269857
Coeff F1 for LinearRegression model trained on X[:,0:10] is 3.303232
Coeff F1 for LinearRegression model trained on X[:,0:11] is 3.292077
Coeff F1 for LinearRegression model trained on X[:,0:12] is 3.320890
Coeff F1 for LinearRegression model trained on X[:,0:13] is 3.347535
Coeff F1 for LinearRegression model trained on X[:,0:14] is 3.326816
Coeff F1 for LinearRegression model trained on X[:,0:15] is 3.285898
Coeff F1 for LinearRegression model trained on X[:,0:16] is 2.973623
Coeff F1 for LinearRegression model trained on X[:,0:17] is 2.933209
Coeff F1 for LinearRegression model trained on X[:,0:18] is 5.019515
Coeff F1 for LinearRegression model trained on X[:,0:19] is 6.364016
Coeff F1 for LinearRegression model trained on X[:,0:20] is 4.998355
Coeff F1 for LinearRegression model trained on X[:,0:21] is 6.364778
Coeff F1 for LinearRegression model trained on X[:,0:22] is 4.133695
Coeff F1 for LinearRegression model trained on X[:,0:23] is 4.730007
Coeff F1 for LinearRegression model trained on X[:,0:24] is 4.981489
Coeff F1 for LinearRegression model trained on X[:,0:25] is 4.930128
Coeff F1 for LinearRegression model trained on X[:,0:26] is 5.093094
Coeff F1 for LinearRegression model trained on X[:,0:27] is 5.198480
Coeff F1 for LinearRegression model trained on X[:,0:28] is 5.163329
Coeff F1 for LinearRegression model trained on X[:,0:29] is 5.157667
Coeff F1 for LinearRegression model trained on X[:,0:30] is 5.312941
Coeff F1 for LinearRegression model trained on X[:,0:31] is 5.275585
Coeff F1 for LinearRegression model trained on X[:,0:32] is 5.282178
Coeff F1 for LinearRegression model trained on X[:,0:33] is 5.303313
Coeff F1 for LinearRegression model trained on X[:,0:34] is 4.938900
Coeff F1 for LinearRegression model trained on X[:,0:35] is 4.758449
Coeff F1 for LinearRegression model trained on X[:,0:36] is 3.920741
Coeff F1 for LinearRegression model trained on X[:,0:37] is 3.979084
Coeff F1 for LinearRegression model trained on X[:,0:38] is 3.655118
Coeff F1 for LinearRegression model trained on X[:,0:39] is 2.546008
Coeff F1 for LinearRegression model trained on X[:,0:40] is 2.426859
Coeff F1 for LinearRegression model trained on X[:,0:41] is 2.217109
Coeff F1 for LinearRegression model trained on X[:,0:42] is 2.310249
Coeff F1 for LinearRegression model trained on X[:,0:43] is 2.528894
Coeff F1 for LinearRegression model trained on X[:,0:44] is 2.577281
Coeff F1 for LinearRegression model trained on X[:,0:45] is 2.579688
Coeff F1 for LinearRegression model trained on X[:,0:46] is 2.777911
Coeff F1 for LinearRegression model trained on X[:,0:47] is 2.778351
Coeff F1 for LinearRegression model trained on X[:,0:48] is 2.750939
Coeff F1 for LinearRegression model trained on X[:,0:49] is 2.624263
Coeff F1 for LinearRegression model trained on X[:,0:50] is 2.613820
Coeff F1 for LinearRegression model trained on X[:,0:51] is 2.802340
Coeff F1 for LinearRegression model trained on X[:,0:52] is 2.730715
Coeff F1 for LinearRegression model trained on X[:,0:53] is 5.263762
Coeff F1 for LinearRegression model trained on X[:,0:54] is 6.424380
Coeff F1 for LinearRegression model trained on X[:,0:55] is 8.821812
Coeff F1 for LinearRegression model trained on X[:,0:56] is 9.162812
Coeff F1 for LinearRegression model trained on X[:,0:57] is 8.983055
Coeff F1 for LinearRegression model trained on X[:,0:58] is 8.984070
Coeff F1 for LinearRegression model trained on X[:,0:59] is 8.619820
Coeff F1 for LinearRegression model trained on X[:,0:60] is 8.961607
Coeff F1 for LinearRegression model trained on X[:,0:61] is 8.991983
```

```
Coeff F1 for LinearRegression model trained on X[:,0:62] is 8.923926
Coeff F1 for LinearRegression model trained on X[:,0:63] is 8.864724
Coeff F1 for LinearRegression model trained on X[:,0:64] is 8.697397
Coeff F1 for LinearRegression model trained on X[:,0:65] is 8.636580
Coeff F1 for LinearRegression model trained on X[:,0:66] is 8.896444
Coeff F1 for LinearRegression model trained on X[:,0:67] is 8.453820
Coeff F1 for LinearRegression model trained on X[:,0:68] is 8.303671
```

Question 2 (3 pts)

a) Perform a manual implementation to compute a full set of principal components of **X** using the following steps:

- Compute the sample covariance matrix
- Perform the eigen decomposition of the covariance matrix
- Project the data onto the principal components (eigenvectors of the covariance matrix)

b) Perform linear regression of **y** on the PCs using `sklearn.linear_model.LinearRegression` when the regressors consist of

- PC1 only;
- PC1 and PC2 only;
- PC1, PC2, and PC3 only;
- and PC1, PC2, PC3, ..., and PC68.

c) In each of the above 4 cases, print the estimated coefficient for **PC1**.

In [4]:

```
X = X.to_numpy().T
```

In [5]:

```
X_mean =  X.mean(axis=1)[np.newaxis]
X_norm = (X - X_mean.T)
C = (X_norm @ X_norm.T) / (X.shape[0]-1)
```

In [18]:

```
L,E = np.linalg.eigh(C)

L = L[::-1]
E = E[:,::-1]

print(E)
print(L)
```

```
[[-4.53163805e-02 -5.34917047e-02  1.24463335e-01 ...  1.24187040e-01
  -5.39367002e-01  4.05276290e-01]
 [-6.56485245e-02 -6.39472605e-02  1.95394851e-01 ...  8.90617762e-03
   4.79952222e-01 -4.51821923e-01]
 [ 2.64016518e-02 -1.74354510e-01  1.46017715e-01 ... -1.03322170e-01
  -5.31592446e-02 -2.03133315e-02]
 ...
 [-2.19513585e-01  4.70861118e-02  1.97728427e-02 ... -2.31882852e-01
   3.51629833e-04 -1.71679372e-02]
 [-2.07908901e-01  5.45498221e-02  2.31983793e-02 ...  4.08714128e-01
   1.34858715e-03  1.11392139e-02]
 [-1.84100793e-01  3.78032967e-02  3.08133404e-02 ... -2.58634779e-01
   2.10911740e-03  1.13587698e-02]]
[2.14881064e+02 1.27394978e+02 8.28128332e+01 7.52252605e+01
 6.01153840e+01 3.56450275e+01 3.29628534e+01 2.95488651e+01
 2.70883283e+01 2.33705421e+01 2.20670446e+01 2.05901873e+01
 1.98397696e+01 1.75940962e+01 1.58917496e+01 1.53888493e+01
 1.49037698e+01 1.46028604e+01 1.33415399e+01 1.28066744e+01
 1.25648283e+01 1.10105340e+01 1.00304329e+01 9.75370925e+00
 9.04484796e+00 8.48678659e+00 7.82412888e+00 7.62835773e+00
 7.39242856e+00 7.12918059e+00 6.96642477e+00 6.62051506e+00
 6.37047002e+00 6.23531289e+00 5.66438793e+00 5.30256457e+00
 4.90429335e+00 4.50817017e+00 4.45081042e+00 4.27317832e+00
 3.80632046e+00 3.46345504e+00 3.26709653e+00 3.17792786e+00
 3.03475943e+00 2.91508764e+00 2.78791724e+00 2.73749539e+00
 2.55627052e+00 2.36102325e+00 2.22497900e+00 2.19391580e+00
 1.96675949e+00 1.78034028e+00 1.67542710e+00 1.60152871e+00
 1.38972242e+00 1.30957048e+00 1.23733821e+00 1.17853179e+00
 1.09863142e+00 1.02484100e+00 1.00244234e+00 8.99845010e-01
 8.04932403e-01 6.90808823e-01 2.92059226e-01 9.39042503e-02]
```

In [19]:

```
A = E.T @ X # Project data onto principle components
```

In [20]:

```
model = LinearRegression()
model.fit(A[:1,:].T,y)
model.coef_
```

Out[20]:

```
array([0.08259023])
```

In [21]:

```
model = LinearRegression()
A = A.T
for i in range(X.shape[0]):
    coeffs_i = model.fit(A[:,:i+1],y)
    print(f'Coeff F1 for LinearRegression model trained on X[:,:{i+1:2}] is {coeffs_
```

```
Coeff F1 for LinearRegression model trained on X[:,: 1] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,: 2] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,: 3] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,: 4] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,: 5] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,: 6] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,: 7] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,: 8] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,: 9] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:10] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:11] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:12] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:13] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:14] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:15] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:16] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:17] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:18] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:19] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:20] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:21] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:22] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:23] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:24] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:25] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:26] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:27] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:28] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:29] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:30] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:31] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:32] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:33] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:34] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:35] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:36] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:37] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:38] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:39] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:40] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:41] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:42] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:43] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:44] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:45] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:46] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:47] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:48] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:49] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:50] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:51] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:52] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:53] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:54] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:55] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:56] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:57] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:58] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:59] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:60] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:61] is 0.082590
```

```
Coeff F1 for LinearRegression model trained on X[:,:62] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:63] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:64] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:65] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:66] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:67] is 0.082590
Coeff F1 for LinearRegression model trained on X[:,:68] is 0.082590
```

Question 3 (3 pts)

a) What do you observe in terms of estimated coefficients of **F1** when the number of regressors are increased compared to the estimated coefficients of **PC1** when the number of regressors are increased?

b) Explain the reason behind this observation.

Answer: For the estimated coefficients of **F1**, we observe that as the number of regressors increase, it change and vary. This shows that The slope in the first dimension changes as the number of regressors change.

However, we observed almost no change in the estimated coefficients of **PC1** when we project the data points into the PC space. This is what I was expecting at the beginning as well. The reason is that when we project data into the PC space, the basis vectors are linearly independent, henceforth, the coefficients should be the same, regardless of the number of PCi axis that we include.

So PCi is orthogonal to PCj for $0 <= i, j <= 68$

Question 4 (2 pts)

a) Print the correlation matrix of all the PC's.

b) Which matrix does it closely resemble to?

c) Which feature of the principal components is depicted through this matrix?

In [22]:

```
print(E @ E.T)
```

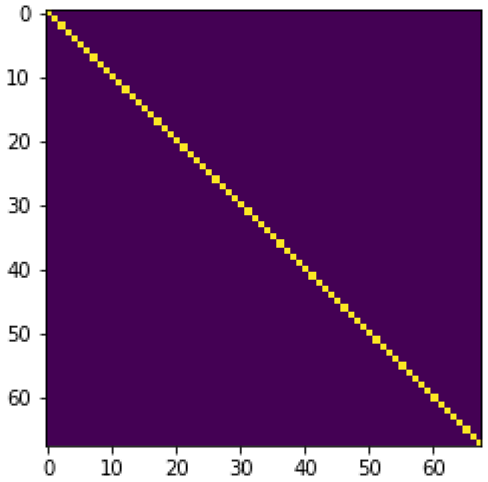
```
[[ 1.00000000e+00 -4.99600361e-16 -1.94289029e-16 ...  4.33680869e-18
  -1.64798730e-17 -8.50014503e-17]
 [-4.99600361e-16  1.00000000e+00  9.54097912e-17 ... -8.67361738e-19
  -2.77555756e-17 -9.54097912e-17]
 [-1.94289029e-16  9.54097912e-17  1.00000000e+00 ... -2.65412692e-16
   9.70903045e-17  1.77863366e-16]
 ...
 [ 4.33680869e-18 -8.67361738e-19 -2.65412692e-16 ...  1.00000000e+00
  -3.00134266e-16  1.39780765e-16]
 [-1.64798730e-17 -2.77555756e-17  9.70903045e-17 ... -3.00134266e-16
   1.00000000e+00 -9.15337684e-17]
 [-8.50014503e-17 -9.54097912e-17  1.77863366e-16 ...  1.39780765e-16
  -9.15337684e-17  1.00000000e+00]]
```

In [23]:

```
plt.imshow(E @ E.T)
```

Out[23]:

```
<matplotlib.image.AxesImage at 0x1568553d0>
```



It closely resembles the Identity matrix.

In [24]:

```
np.isclose(E@E.T, np.eye(68))
```

Out[24]:

```
array([[ True,  True,  True, ...,  True,  True,  True],
       [ True,  True,  True, ...,  True,  True,  True],
       [ True,  True,  True, ...,  True,  True,  True],
       ...,
       [ True,  True,  True, ...,  True,  True,  True],
       [ True,  True,  True, ...,  True,  True,  True],
       [ True,  True,  True, ...,  True,  True,  True]])
```

It shows that princple components are orthogonal to each other.