Origins of Music: Looking at links between music around the globe using clustering

Shushruth Kallutla

Introduction

The Dataset I chose is titled Geographical Original of Music Data Set. It was sourced from the UCI Machine Learning Repository. The Dataset contains audio features and geographic data of music tracks from a private collection. The dataset contains 1059 tracks that originate from 33 different countries. 68 audio features are recorded along with longitude and latitude, thus totaling 70 variables in the dataset. The MARSYAS Framework was used to extract audio features. The dataset was originally used for linear regression and classification analysis to predict the geographic origin of music using audio features. This analysis is documented in Zhao et al. (2014). For this project, I will be using PCA along with Clustering to look for possible links between the music of different countries.

Methods

The Data was first simplified using PCA, Two principal components were preserved in the component space. The code for the PCA is as follows:

```
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt


#Load Dataset
geomusT = pd.read_csv("FinalProj.csv").values
geomus = geomusT.T[0: 68]
m, n = geomus.shape
```

```
print(geomus.shape)


#Find mean of Data
geomusMean = geomus.mean(axis = 1)
print(geomusMean.shape)


#Center Data around mean
geomusMeanTileT = np.tile(geomusMean, (1058, 1))
geomusMeanTile = geomusMeanTileT.T
Z = geomus - geomusMeanTile


#Caluculate Covariance Matrix
C = np.matmul(Z,Z.T)/(n-1)

#Check shape
print(C.shape)


#Find components
D,V = np.linalg.eig(C)
idx = D.argsort()[::-1]
Vs = V[:,idx]

#Create component space
Proj = np.matmul(Vs[:,0:2].T,Z)
Proj.shape


#Plot component space
plt.plot(Proj[0], Proj[1], 'x')
plt.title("Component Space")
plt.show()


#Define function for Clustering script
def set_Proj():
   return Proj.T
```

After PCA, The component space underwent clustering. The clusters were plotted on the component space as well as a world map using the python cartography module, Cartopy. The code for Clustering and plotting is as follows:

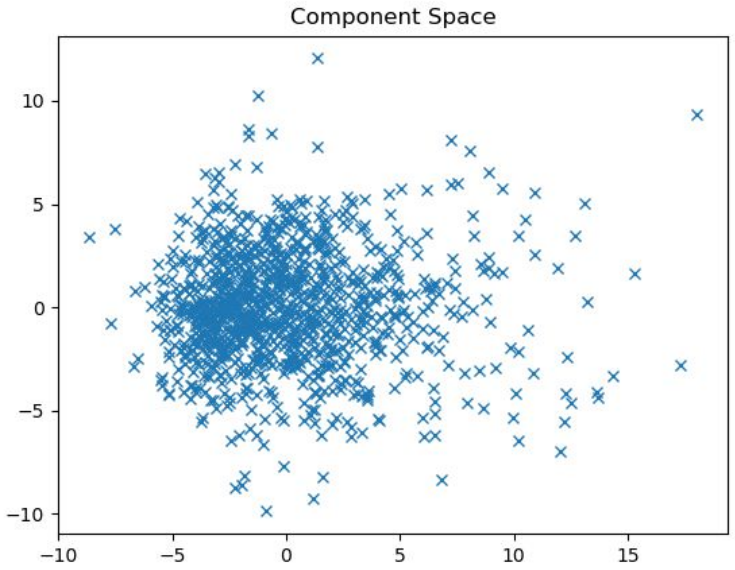
```
import PCA
import numpy as np
import pandas as pd
import cartopy.crs as ccrs

import matplotlib.pyplot as plt
import mpl_toolkits


#Load Dataset
geomusT = pd.read_csv("FinalProj.csv").values
geomus = geomusT.T
geomus.shape


#Load componenst space
clusData = PCA.set_Proj()
print(clusData.shape)


#Load color list
color_list = ["#803902", "#851000", "#008044", "#051485", "#850159",
"#008717",
          "#8a0500", "#6b1634", "#336915"]


#Set number of clusters
k = 2

# Assign k start points
C = clusData[np.random.randint(clusData.shape[0], size = k)]

for itr in range(200): # set number of k-mean iterations
   # initialize distance and cluster membership
   cluster_ind = np.zeros(len(clusData))
   distance = np.zeros((len(clusData), k))

   # find distance of every point to each centroid, and cluster membership
   i = 0
   centroid_calc = np.zeros((k, 3))
```

```
    for p in clusData:
        #find distance of points
        for d in range(k):
            distance[i][d] = (C[d][0] - p[0])**2 + (C[d][1] - p[1])**2



        #determine cluster membership
        for z in range(k):
            if (min(distance[i]) == distance[i][z]):
                cls_mem = z
                centroid_calc[z][0] += p[0]
                centroid_calc[z][1] += p[1]
                centroid_calc[z][2] += 1

        #update membership array
        cluster_ind[i] = cls_mem

        i += 1

    # update cluster centroids
    for i2 in range(k):
        C[i2][0] = centroid_calc[i2][0]/centroid_calc[i2][2]
        C[i2][1] = centroid_calc[i2][1]/centroid_calc[i2][2]


# Plot each iteration of k-means to show the first 4
i3 = 0
#Assign color to each cluster
for q in clusData:
    for n in range(k):
        if (cluster_ind[i3] == n):
            plt.plot(q[0],q[1], color= color_list[n] , marker='o')
    i3 += 1

#Plot Cluster
title = "Clusters with k = " + str(k)
plt.title(title)
plt.show()


#Load Location Data
spac_data = geomus[68:]
spac = spac_data.T
```

```
#Plot Component Space onto Map
ax = plt.axes(projection=ccrs.PlateCarree())
ax.stock_img()
ax.scatter(spac_data[0], spac_data[1], zorder=1, alpha= 0.2, c='b', s=10)
ax.set_title('Component space projected onto World Map')
plt.show()


#Plot CLusters on Maps
ax1 = plt.axes(projection=ccrs.PlateCarree())
ax1.stock_img()
i3 = 0

#Assign color to each cluster
for q in clusData:
   for n in range(k):
      if (cluster_ind[i3] == n):
         ax1.plot(spac[i3][0],spac[i3][1], color= color_list[n] , marker='o')
   i3 += 1

title2 = "Clusters projected onto World Map with k = " + str(k)
ax1.set_title(title2)
plt.show()
```

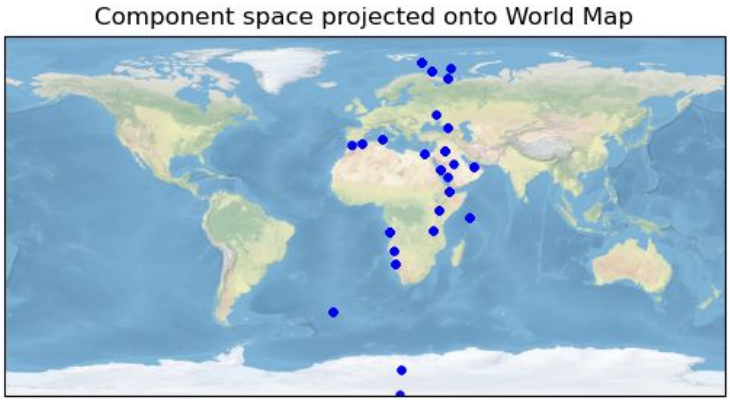
Results

The Component Space



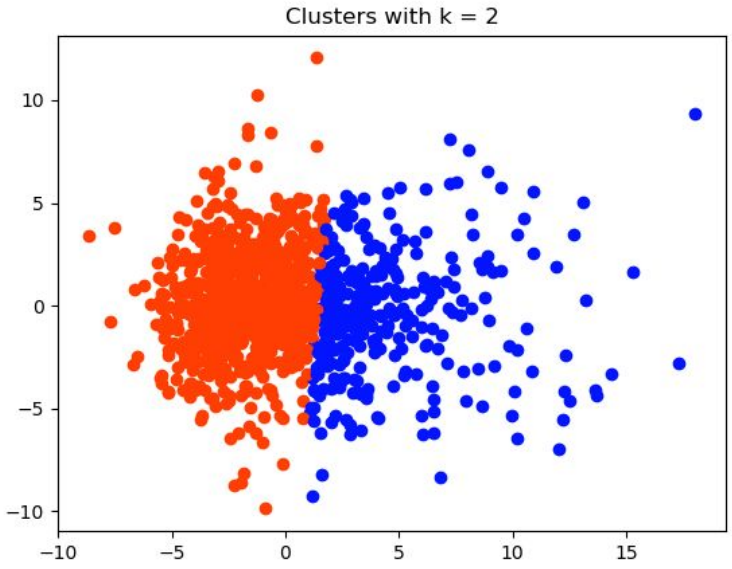
We can see that there seem to be no obvious clusters in the component space.

Component Space on World Map

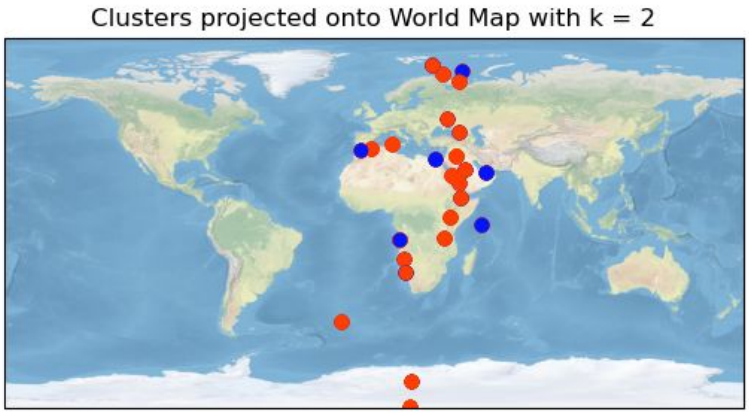


We can see the 33 countries that the music is sourced from on the world map. The location data of the music is limited to the country of origin and the location maps the capital of the country of origin.

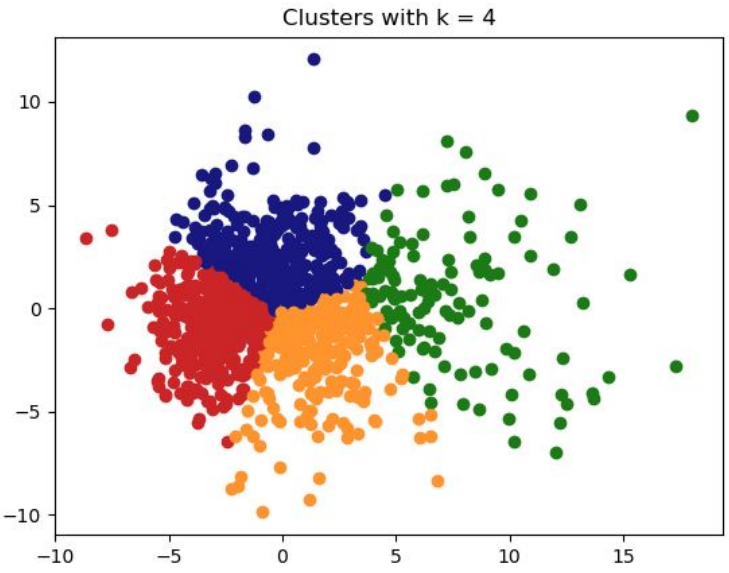
Clusters with k = 2



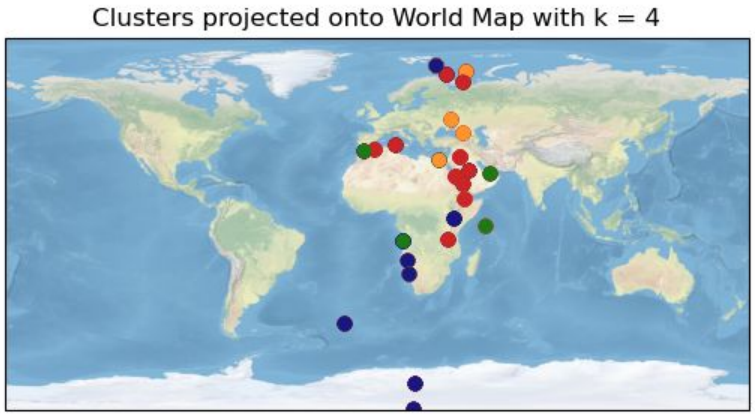
Clusters with k = 2 projected onto world map



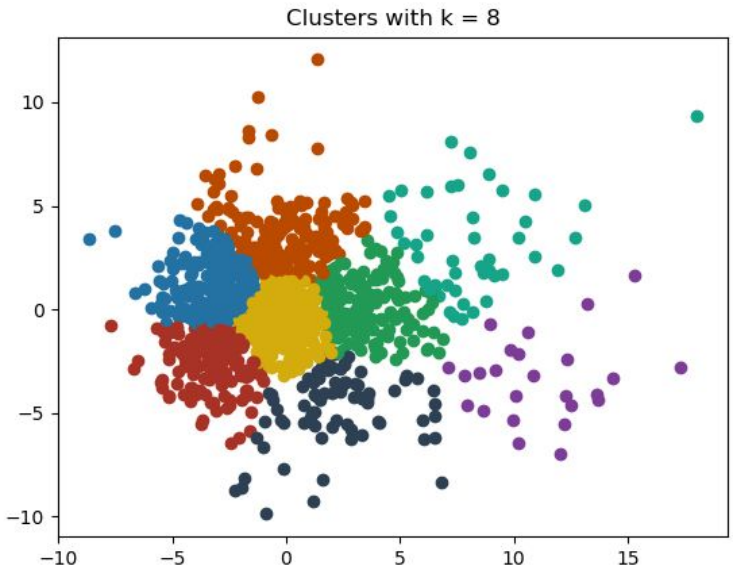
Clusters with k = 4



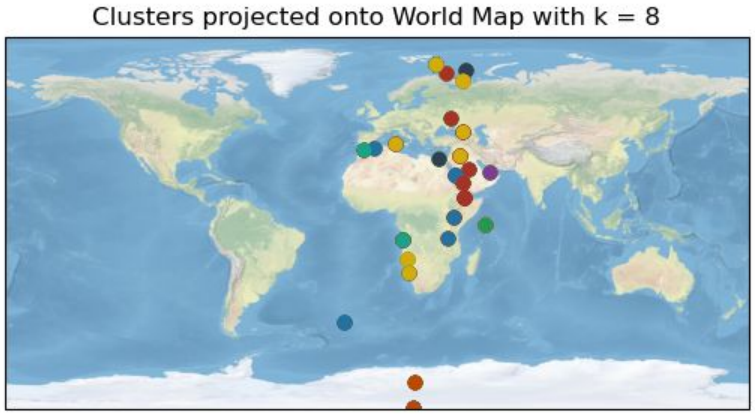
Clusters with k = 4 projected onto world map



Clusters with k = 8



Clusters with k = 8 projected onto world map



Discussion

We can see that the component space does not have any apparent clusters and hence does not have an apparent k value that would best fit the data. When we plot the space on a map we can see that the music in the dataset comes primarily from African and Middle Eastern countries. We must note that as the data is sourced from only a few countries and the dataset only records the location of the country's capital and not the exact location of the music source. This may lead to points overlapping. Hence the cluster maps we see may not be a perfect representation of the geographic distribution of the clusters. Further analysis could find a better way of representing the overlapping data.

In the cluster map with two clusters, the map does not seem to have an obvious trend. One group seems to be spread all over the map while the other seems to cover Northern Africa. This could imply that North African Music that makes up the cluster share some musical features. This could further imply a common origin and a history of cultural contact between these countries.

In the cluster map with four clusters, we see that some clusters seem to be localized to certain regions. Such as the yellow cluster seems to be localized around Europe and the green cluster seems to be localized around Africa. The Blue cluster is generally localized towards the south with one exception. The red cluster seems to be localized in the north. These clusters suggesting relationships in the music of those regions mapped by each cluster. This map seems to have the most apparent trends amongst the maps we've plotted and hence k = 4 clusters could be a candidate for best fit.

In the cluster map with 8 clusters, there doesn't seem to be any identifiable trend. Some clusters seem to be localized such as brown in the south. But others seem more spread out, such as yellow.

These clusters can be a starting point for researchers studying the evolution of music in these regions.

References

Fang Zhou, Claire Q and Ross. D. King

Predicting the Geographical Origin of Music, ICDM, 2014

https://archive.ics.uci.edu/ml/datasets/Geographical+Original+of+Music

Cartopy.

v0.18. 10-Dec-2020. Met Office. UK.

https://github.com/SciTools/cartopy/archive/v0.11.2.tar.gz