# Project Report

## Pattern Discovery In Textures

Name:Keshav Anand

Course: AI and ML

(Batch-4)

Duration: 12 months

Problem Statement: Pattern discovery in textures using Spherical K-means.

### Prerequisites

What things you need to install the software and how to install them:

Python 3.6 This setup requires that your machine has latest version of python. The following url https://www.python.org/downloads/ can be referred to download python. Once you have python downloaded and installed, you will need to setup PATH variables (if you want to run python program directly, detail instructions are below in how to run software section). To do that check this: https://www.pythoncentral.io/add-python-to-path-python-is-not- recognized-as-an-internal-or-external-command/. Setting up PATH variable is optional as you can also run program without it and more instruction are given below on this topic.

Second and easier option is to download anaconda and use its anaconda prompt to run the commands. To install anaconda check this url https://www.anaconda.com/download/ You will also need to download and install below 3 packages after you install either python or anaconda from the steps above Sklearn (scikit-learn) numpy scipy if you have chosen to install python 3.6 then run below commands in command prompt/terminal to install these packages pip install -U scikit-learn pip install numpy pip install scipy if you have chosen to install anaconda then run below commands in anaconda prompt to install these packages conda install -c scikit-learn conda install -c anaconda numpy conda install -c anaconda scipy

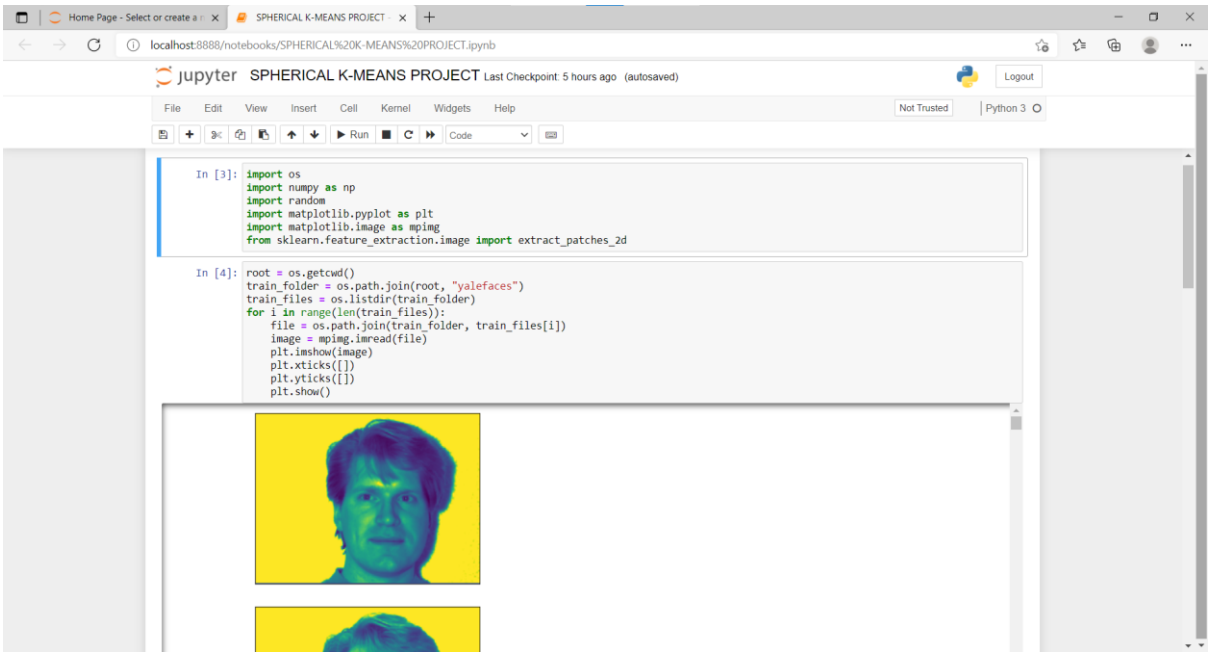
### Dataset used:

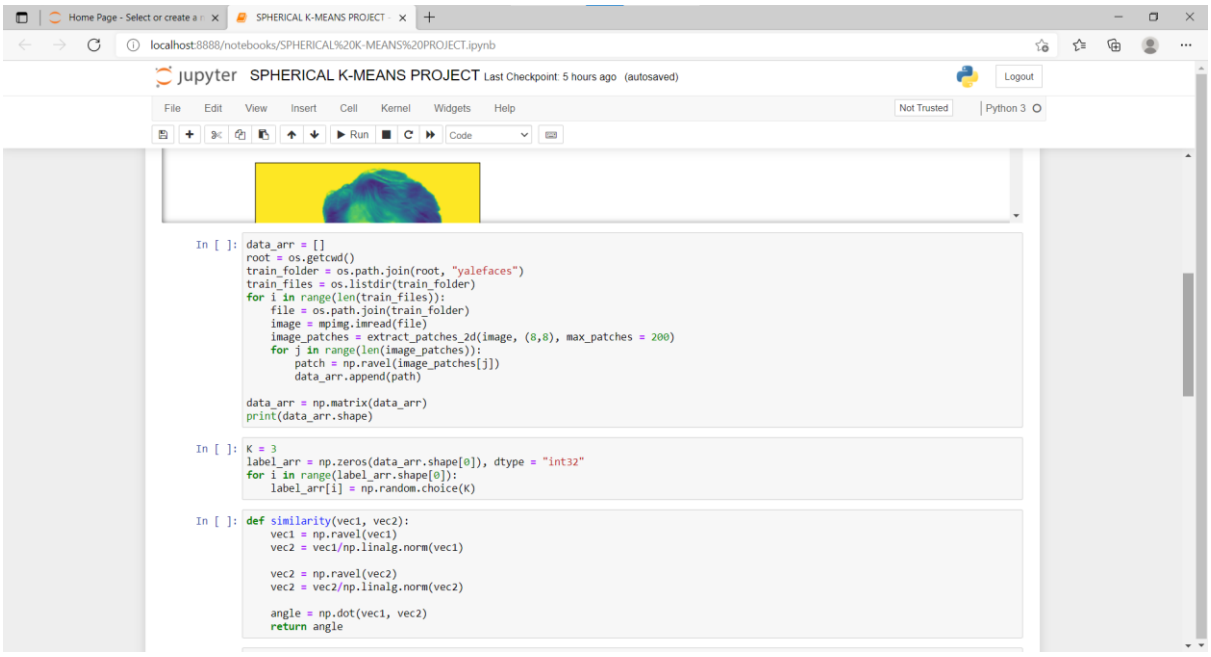
The dataset used is YALE Face dataset which is available on a website called Kaggle.com.

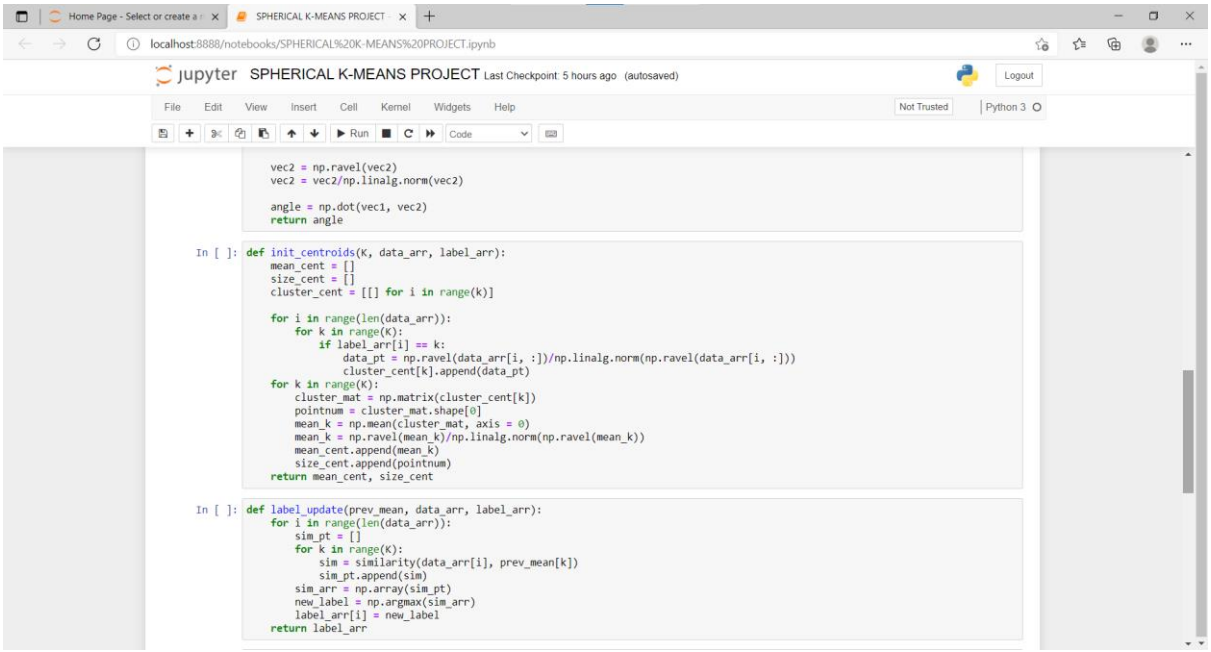
## Method used for Process:

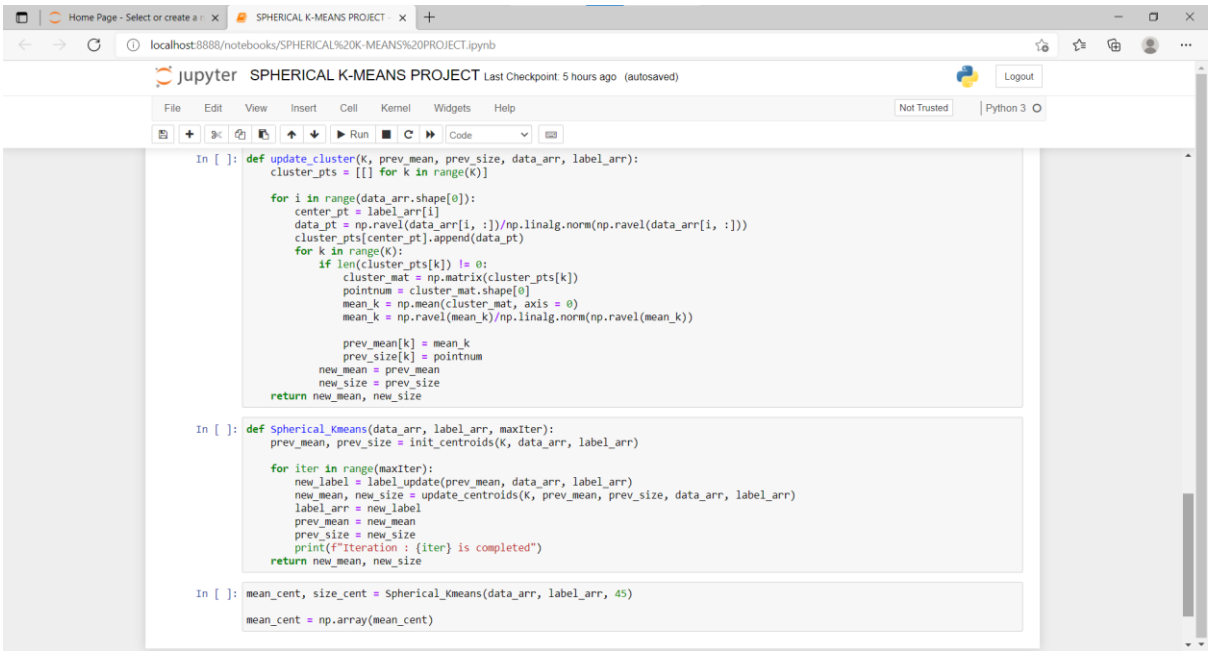
Spherical K-MEANS

## Screenshots of Source Code and Output:




Home Page - Select or create a
SPHERICAL K-MEANS PROJECT
localhost:8888/notebooks/SPHERICAL%20K-MEANS%20PROJECT.ipynb
jupyter SPHERICAL K-MEANS PROJECT Last Checkpoint: 5 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
vec2 = np.ravel(vec2)
vec2 = vec2/np.linalg.norm(vec2)
angle = np.dot(vec1, vec2)
return angle
In [ ]:
def init_centroids(K, data_arr, label_arr):
mean_cent = []
size_cent = []
cluster_cent = [[] for i in range(k)]
for i in range(len(data_arr)):
for k in range(K):
if label_arr[i] == k:
data_pt = np.ravel(data_arr[i, :])/np.linalg.norm(np.ravel(data_arr[i, :]))
cluster_cent[k].append(data_pt)
for k in range(K):
cluster_mat = np.matrix(cluster_cent[k])
pointnum = cluster_mat.shape[0]
mean_k = np.mean(cluster_mat, axis = 0)
mean_k = np.ravel(mean_k)/np.linalg.norm(np.ravel(mean_k))
mean_cent.append(mean_k)
size_cent.append(pointnum)
return mean_cent, size_cent
In [ ]:
def label_update(prev_mean, data_arr, label_arr):
for i in range(len(data_arr)):
sim_pt = []
for k in range(K):
sim = similarity(data_arr[i], prev_mean[k])
sim_pt.append(sim)
sim_arr = np.array(sim_pt)
new_label = np.argmax(sim_arr)
label_arr[i] = new_label
return label_arr


Home Page - Select or create a
SPHERICAL K-MEANS PROJECT
localhost:8888/notebooks/SPHERICAL%20K-MEANS%20PROJECT.ipynb
jupyter SPHERICAL K-MEANS PROJECT Last Checkpoint: 5 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
In [ ]:
def update_cluster(K, prev_mean, prev_size, data_arr, label_arr):
cluster_pts = [[] for k in range(K)]
for i in range(data_arr.shape[0]):
center_pt = label_arr[i]
data_pt = np.ravel(data_arr[i, :])/np.linalg.norm(np.ravel(data_arr[i, :]))
cluster_pts[center_pt].append(data_pt)
for k in range(K):
if len(cluster_pts[k]) != 0:
cluster_mat = np.matrix(cluster_pts[k])
pointnum = cluster_mat.shape[0]
mean_k = np.mean(cluster_mat, axis = 0)
mean_k = np.ravel(mean_k)/np.linalg.norm(np.ravel(mean_k))
prev_mean[k] = mean_k
prev_size[k] = pointnum
new_mean = prev_mean
new_size = prev_size
return new_mean, new_size
In [ ]:
def Spherical_Kmeans(data_arr, label_arr, maxIter):
prev_mean, prev_size = init_centroids(K, data_arr, label_arr)
for iter in range(maxIter):
new_label = label_update(prev_mean, data_arr, label_arr)
new_mean, new_size = update_centroids(K, prev_mean, prev_size, data_arr, label_arr)
label_arr = new_label
prev_mean = new_mean
prev_size = new_size
print(f"Iteration : {iter} is completed")
return new_mean, new_size
In [ ]:
mean_cent, size_cent = Spherical_Kmeans(data_arr, label_arr, 45)
mean_cent = np.array(mean_cent)