Capstone Project Report

Name: Sumit Patel

Course: AI and ML (Batch – AUG 2020)

Duration: 10 months

Incremental Clustering: Intrusion Detection by Visual Surveillance

Problem Statement:

Factor analysis is a useful technique to find latent factors that can potentially describe multiple attributes, which is sometimes very useful for dimensionality reduction. Use the Airline Passenger Satisfaction dataset to perform factor analysis. (Use only the columns that represent the ratings given by the passengers, only 14 columns). Choose the best features possible that helps in dimensionality reduction, without much loss in information.

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-externalcommand/. 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 numpy

pip install matplotlib

If you have chosen to install anaconda then run below commands in anaconda prompt to install these packages:

conda install -c anaconda numpy

conda install -c anaconda matplotlib

Dataset used:

Any bright coloured image

Importing the libraries and loading dataset.

```
# Importing libraries
import numpy as np
import matplotlib.image as mpimg
import matplotlib.pyplot as plt
import os
import sys
import cv2
%matplotlib inline
```

```
# Function to initialize the mean and vector images
def initBackground(initImage):
    img_arr = mpimg.imread(initImage)
    mean = img_arr
    var = 9*np.ones(img_arr.shape)
    return (mean, var)
```

```
# Classify images into foreground and background pixels using Chebyshevs inequality based classifier
def ForegroundDetection(img_file, mean, variance, lmda):
    img = mpimg.imread(img_file)
    d = img - mean
    y = var*(lmda**2)
    d_2 = np.square(d)
    I = d_2 - y
    mask = np.all(I>0, axis = 2)
    r1 = 255*mask.astype(int)
    r1 = r1.astype(np.uint8)
    return r1
```

```
# Reduce image noise using voting scheme
def Voting(rI, eta, m, n):
    r,c = rI.shape
    cI = np.zeros((rI.shape[0], rI.shape[1]))
    for i in range(m,r-1-m):
        for j in range(n,c-1-n):
            img_patch = rI[i-m:i, j-n:j]
            y_unq, counts = np.unique(img_patch, return_counts = True)
            if len(counts) == 1 and y_unq[0] == 1:
                cI[i,j] = 255
            if len(counts) > 1:
                if counts[1] > eta*m*n:
                    cI[i,j] = 255
    cI = cI.astype(np.uint8)
    return cI
```

```
# Update the mean and variance images using a weighted average scheme
def meanvarUpdate(cI, img_path, M, V, alpha):
    img = mpimg.imread(img_path)
    mean_upd = np.zeros(img.shape)
    var_upd = np.zeros(img.shape)
    d = img - M
    d_2 = np.square(d)
    for i in range(cI.shape[0]):
        for j in range(cI.shape[1]):
            if cI[i,j] == 0:
                mean_upd[i,j,:] = (1-alpha)*M[i,j,:] + alpha*img[i,j,:]
                var_upd[i,j,:] = (1-alpha)*M[i,j,:] + alpha*d_2[i,j,:]
                var_upd[i,j,:] = np.clip(var_upd[i,j,:], a_min = 9, a_max = None)
    return(mean_upd, var_upd)
```

```
def Background_Subtraction(img_dir, lmda, eta, m, n, alpha):
    img_file_name = os.listdir(img_dir)
    initImage = os.path.join(img_dir, img_file_name[0])
    mean, var = initBackground(initImage)

    for i in range(1, len(img_file_name)):
        img_path = os.path.join(img_dir, img_file_name[i])
        fig, ax = plt.subplots(1,3,figsize = (10,10))
        rI = ForegroundDetection(img_path, mean, var, lmda)
        ax[0].imshow(rI, cmap = 'gray')

        cI = Voting(rI, eta, m, n)
        mean, var = meanvarUpdate(cI, img_path, mean, var, alpha)
        ax[1].imshow(cI, cmap = 'gray')

        img = mpimg.imread(img_path)
        ax[2].imshow(img, cmap = 'gray')

        plt.show()
```

```
Background_Subtraction("./Images", 0.8, 0.7, 8, 8, 0.8)
```