Incremental Clustering: Intrusion Detection by Visual Surveillance

Name - Amith Korada

Course - AI & ML
(Batch - 4)

Duration - 12 Months

Problem Statement - Building a Machine Learning model for intrusion detection

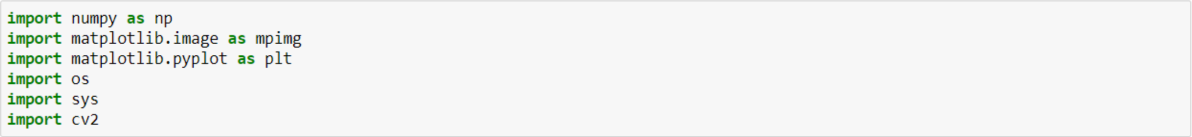
Prerequisites -

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

Python 3.6 This setup requires that your machine has the latest version of python. The following URL https://www.python.org/downloads/ can be referred to as download python.

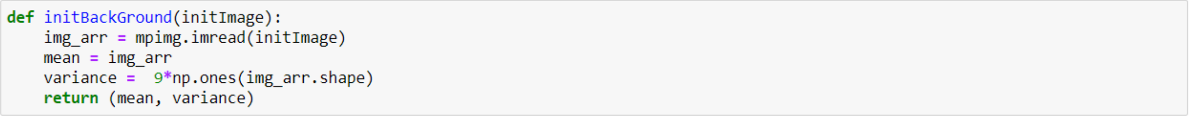
The second and easier option is to download anaconda and use its anaconda prompt to run the commands. To install anaconda check this URL https://www.anaconda.com/download/ You will also need to download and install the below 3 packages after you install either python or anaconda from the steps above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

1. Importing necessary libraries -



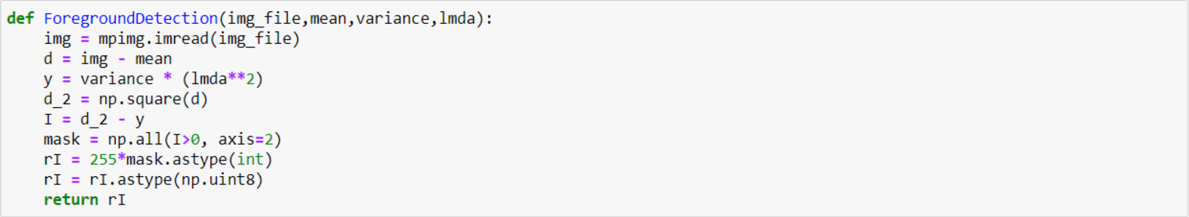
```python
import numpy as np
import matplotlib.image as mpimg
import matplotlib.pyplot as plt
import os
import sys
import cv2
```

2. Initialising mean and variance-



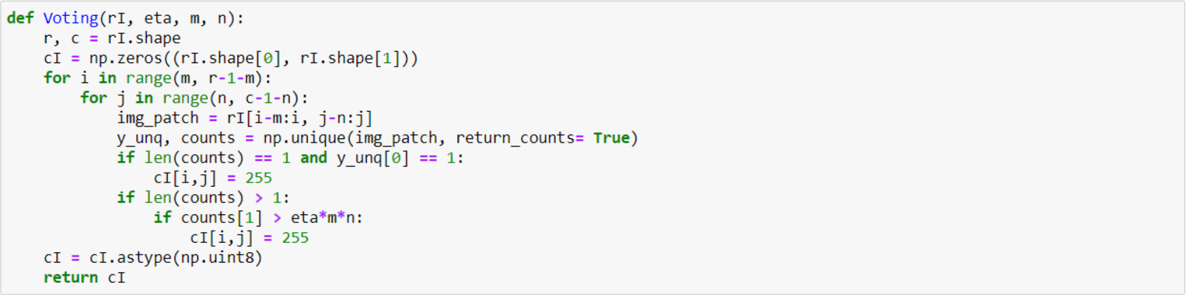
```python
def initBackGround(initImage):
    img_arr = mpimg.imread(initImage)
    mean = img_arr
    variance =  9*np.ones(img_arr.shape)
    return (mean, variance)
```

3. Classifying images into the foreground and background pixels-



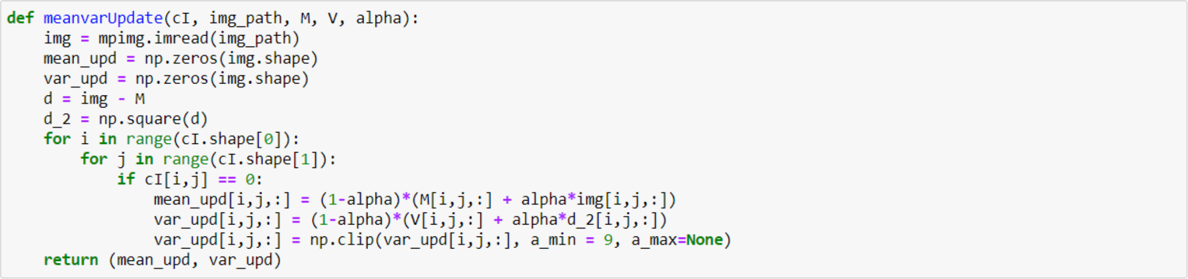
```python
def ForegroundDetection(img_file,mean,variance,lmda):
    img = mpimg.imread(img_file)
    d = img - mean
    y = variance * (lmda**2)
    d_2 = np.square(d)
    I = d_2 - y
    mask = np.all(I>0, axis=2)
    rI = 255*mask.astype(int)
    rI = rI.astype(np.uint8)
    return rI
```

4. Removing the noise-



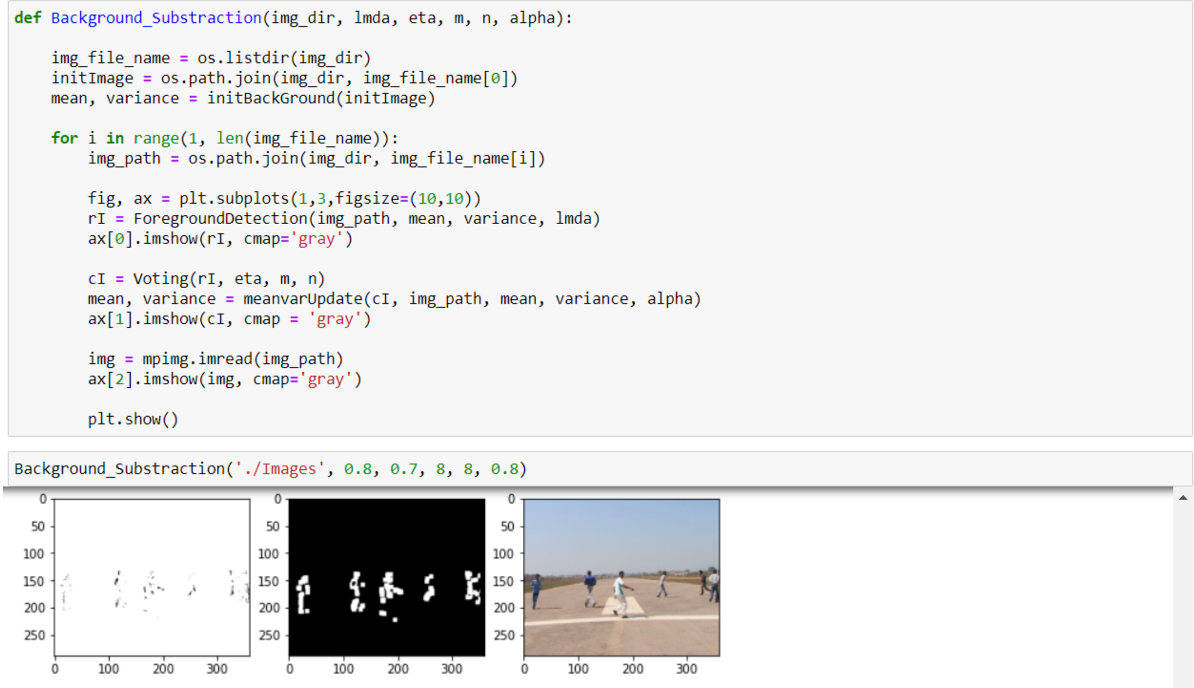
```python
def Voting(rI, eta, m, n):
    r, c = rI.shape
    cI = np.zeros((rI.shape[0], rI.shape[1]))
    for i in range(m, r-1-m):
        for j in range(n, c-1-n):
            img_patch = rI[i-m:i, j-n:j]
            y_unq, counts = np.unique(img_patch, return_counts= True)
            if len(counts) == 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
```

5. Updating mean and variance-



```python
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)*(V[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)
```

6. Background Subtraction-



```
def Background_Substraction(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, variance = 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, variance, lmda)
        ax[0].imshow(rI, cmap='gray')

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

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

        plt.show()
```

```
Background_Substraction('./Images', 0.8, 0.7, 8, 8, 0.8)
```