import sys

import os

import cv2

import numpy as np

Get and store the temperature parameters(low/medium/high/unknown) in DB

Get and store the mositure parameters(low/medium/high/unknown) in DB

Get and store the plan type in DB

Get and store the leaf type(Younger/Middle\_aged/Older/Unknown) in DB

Get the image files

Square Differences Based Image Matching Algorithm

Formula used: 

# Read the images from the file

templ\_image = cv2.imread('pic1.jpg')

main\_image = cv2.imread('pic2.jpg')

While not end of image

#Each pixel is represented by a two dimensional Array

(X1,Y1) <----- Next Pixel dimensions of main\_image

While not end of Template

SUM <----- 0.0

(X,Y) <----- Next Pixel dimensions of template\_image

While(not end of arraysX[],y[]

Read next value of arrays X[], X1[]

Read next value of arrays Y[],Y1[]

Calculate the difference between the X, Y vector values

Calculate the sum of the square differences as per the above formula

End while

Add the square difference to the sum

End while

Sotre the square difference in the result vector

end while

# We want the minimum squared difference

mnLoc = cv2.minMaxLoc(result)

# Draw the rectangle:

# Extract the coordinates of our best match

MPx,MPy = mnLoc

# Step 2: Get the size of the template. This is the same size as the match.

trows,tcols = templ\_image.shape[:2]

# Step 3: Draw the rectangle on large\_image

cv2.rectangle(large\_image, (MPx,MPy),(MPx+tcols,MPy+trows))

# Display the original image with the rectangle around the match.

cv2.imshow('output',large\_image)

# The image is only displayed if we call this

cv2.waitKey(7000)

Temparature,Rainfall && Evaporation based Regression Algorithm:

import pandas as pd

from pandas import DataFrame

import matplotlib.pyplot as plt

from sklearn import linear\_model

import statsmodels.api as sm

wthr\_condns = pd.read\_csv(r'weather.csv')

df = DataFrame(wthr\_condns,columns=['Week','MinTemp','MaxTemp','Rainfall','Evaporation','Humidity','Pressure','Blight'])

X = df[['MinTemp','MaxTemp','Rainfall','Evaporation']]

Y = df['Blight']

# with Linear Regression

regr = linear\_model.LinearRegression()

regr.fit(X, Y)

print('Intercept: \n', regr.intercept\_)

print('Coefficients: \n', regr.coef\_)

# prediction with sklearn

lweek\_MinTemp = 12.4

lweek\_MaxTemp = 32.0

lweek\_Rainfall = 2.4

lweek\_Evaporation = 8.2

print ('Predicted Blight probability with sklearn: ', regr.predict([[lweek\_MinTemp,lweek\_MaxTemp,lweek\_Rainfall,lweek\_Evaporation]]))

print ('Predicted Blight probability with linear regression:',regr.intercept\_ + (regr.coef\_[0]\*lweek\_MinTemp) + (regr.coef\_[1]\*lweek\_MaxTemp)+ (regr.coef\_[2]\*lweek\_Rainfall)+(regr.coef\_[3]\*lweek\_Evaporation))