**//MATLAB CODE for Disease Detection //**

>> multisvm  
Not enough input arguments.

Error in multisvm (line 5)  
itrind=size(test,1);

Error using ClassificationSVM (line 249)  
Use fitcsvm to train an SVM model.

Error in multisvm (line 29)  
classes = ClassificationSVM(svmStruct,tst);

Error in DetectDisease\_GUI>pushbutton6\_Callback (line 381)  
result = multisvm(Train\_Feat,Train\_Label,test);

Error in gui\_mainfcn (line 95)  
feval(varargin{:});

Error in DetectDisease\_GUI (line 46)  
gui\_mainfcn(gui\_State, varargin{:});

Error in  
matlab.graphics.internal.figfile.FigFile/read>@(hObject,eventdata)DetectDisease\_GUI('pushbutton6\_Callback',hObject,eventdata,guidata(hObject))  
Error while evaluating UIControl Callback

Not enough input arguments.

Error in multisvm (line 5)  
itrind=size(test,1);  
  
Error using ClassificationSVM (line 249)  
Use fitcsvm to train an SVM model.

Error in multisvm (line 29)  
classes = ClassificationSVM(svmStruct,tst);

Error in DetectDisease\_GUI>pushbutton6\_Callback (line 381)  
result = multisvm(Train\_Feat,Train\_Label,test);

Error in gui\_mainfcn (line 95)  
feval(varargin{:});

Error in DetectDisease\_GUI (line 46)  
gui\_mainfcn(gui\_State, varargin{:});

Error in  
matlab.graphics.internal.figfile.FigFile/read>@(hObject,eventdata)DetectDisease\_GUI('pushbutton6\_Callback',hObject,eventdata,guidata(hObject))  
Error while evaluating UIControl Callback.

% Project Title: Disease Detection of COVID 19  
clc  
close all  
clear all  
[filename, pathname] = uigetfile({'\*.\*';'\*.bmp';'\*.jpg';'\*.gif'}, 'Pick a Leaf Image File');  
I = imread(pathname,filename);  
I = imresize(I,[256,256]);  
%figure, imshow(I); title('Query Leaf Image');  
% Enhance Contrast  
I = imadjust(I,stretchlim(I));  
figure, imshow(I);title('Contrast Enhanced');  
% Otsu Segmentation  
I\_Otsu = im2bw(I,graythresh(I));  
% Conversion to HIS  
I\_HIS = rgb2hsi(I);  
%% Extract Features  
% Function call to evaluate features  
%[feat\_disease seg\_img] = EvaluateFeatures(I)  
% Color Image Segmentation  
% Use of K Means clustering for segmentation  
% Convert Image from RGB Color Space to L\*a\*b\* Color Space  
% The L\*a\*b\* space consists of a luminosity layer 'L\*', chromaticity-layer 'a\*' and 'b\*'.  
% All of the color information is in the 'a\*' and 'b\*' layers.  
cform = makecform('srgb2lab');  
% Apply the colorform  
lab\_he = applycform(I,cform);  
% Classify the colors in a\*b\* colorspace using K means clustering.  
% Since the image has 3 colors create 3 clusters.  
% Measure the distance using Euclidean Distance Metric.  
ab = double(lab\_he(:,:,2:3));  
nrows = size(ab,1);  
ncols = size(ab,2);  
ab = reshape(ab,nrows\*ncols,2);  
nColors = 3;  
[cluster\_idx cluster\_center] = kmeans(ab,nColors,'distance','sqEuclidean', ...  
'Replicates',3);  
%[cluster\_idx cluster\_center] = kmeans(ab,nColors,'distance','sqEuclidean','Replicates',3);  
% Label every pixel in tha image using results from K means  
pixel\_labels = reshape(cluster\_idx,nrows,ncols);  
%figure,imshow(pixel\_labels,[]), title('Image Labeled by Cluster Index');  
% Create a blank cell array to store the results of clustering  
segmented\_images = cell(1,3);  
% Create RGB label using pixel\_labels  
rgb\_label = repmat(pixel\_labels,[1,1,3]);  
for k = 1:nColors  
colors = I;  
colors(rgb\_label ~= k) = 0;  
segmented\_images{k} = colors;  
end  
figure, subplot(3,1,1);imshow(segmented\_images{1});title('Cluster 1'); subplot(3,1,2);imshow(segmented\_images{2});title('Cluster 2');  
subplot(3,1,3);imshow(segmented\_images{3});title('Cluster 3');  
set(gcf, 'Position', get(0,'Screensize'));  
% Feature Extraction  
x = inputdlg('Enter the cluster no. containing the ROI only:');  
i = str2double(x);  
% Extract the features from the segmented image  
seg\_img = segmented\_images{i};  
% Convert to grayscale if image is RGB  
if ndims(seg\_img) == 3  
img = rgb2gray(seg\_img);  
end  
%figure, imshow(img); title('Gray Scale Image');  
% Evaluate the disease affected area  
black = im2bw(seg\_img,graythresh(seg\_img));  
%figure, imshow(black);title('Black & White Image');  
m = size(seg\_img,1);  
n = size(seg\_img,2);  
zero\_image = zeros(m,n);  
%G = imoverlay(zero\_image,seg\_img,[1 0 0]);  
cc = bwconncomp(seg\_img,6);  
diseasedata = regionprops(cc,'basic');  
A1 = diseasedata.Area;  
sprintf('Area of the disease affected region is : %g%',A1);  
I\_black = im2bw(I,graythresh(I));  
kk = bwconncomp(I,6);  
leafdata = regionprops(kk,'basic');  
A2 = leafdata.Area;  
sprintf(' Total leaf area is : %g%',A2);  
%Affected\_Area = 1-(A1/A2);  
Affected\_Area = (A1/A2);  
if Affected\_Area < 0.1  
Affected\_Area = Affected\_Area+0.15;  
end  
sprintf('Affected Area is: %g%%',(Affected\_Area\*100))  
% Create the Gray Level Cooccurance Matrices (GLCMs)  
glcms = graycomatrix(img);  
% Derive Statistics from GLCM  
stats = graycoprops(glcms,'Contrast Correlation Energy Homogeneity');  
Contrast = stats.Contrast;  
Correlation = stats.Correlation;  
Energy = stats.Energy;  
Homogeneity = stats.Homogeneity;  
Mean = mean2(seg\_img);  
Standard\_Deviation = std2(seg\_img);  
Entropy = entropy(seg\_img);  
RMS = mean2(rms(seg\_img));  
%Skewness = skewness(img)  
Variance = mean2(var(double(seg\_img)));  
a = sum(double(seg\_img(:)));  
Smoothness = 1-(1/(1+a));  
Kurtosis = kurtosis(double(seg\_img(:)));  
Skewness = skewness(double(seg\_img(:)));  
% Inverse Difference Movement  
m = size(seg\_img,1);  
n = size(seg\_img,2);  
in\_diff = 0;  
for i = 1:m  
for j = 1:n  
temp = seg\_img(i,j)./(1+(i-j).^2);  
in\_diff = in\_diff+temp;  
end  
end  
IDM = double(in\_diff);  
  
feat\_disease = [Contrast,Correlation,Energy,Homogeneity, Mean, Standard\_Deviation, Entropy, RMS, Variance, Smoothness, Kurtosis, Skewness, IDM];  
%%  
% Load All The Features  
load('Training\_Data.mat')  
% Put the test features into variable 'test'  
test = feat\_disease;  
result = multisvm(Train\_Feat,Train\_Label,test);  
%disp(result);  
% Visualize Results  
if result == 0  
helpdlg(' Alternaria Alternata ');  
disp(' Alternaria Alternata ');  
elseif result == 1  
helpdlg(' Anthracnose ');  
disp('Anthracnose');  
elseif result == 2  
helpdlg(' Bacterial Blight ');  
disp(' Bacterial Blight ');  
elseif result == 3  
helpdlg(' Cercospora Leaf Spot ');  
disp('Cercospora Leaf Spot');  
elseif result == 4  
helpdlg(' Healthy Leaf ');  
disp('Healthy Leaf ');  
end  
%% Evaluate Accuracy  
load('Accuracy\_Data.mat')  
Accuracy\_Percent= zeros(200,1);  
for i = 1:500  
data = Train\_Feat;  
%groups = ismember(Train\_Label,1);  
groups = ismember(Train\_Label,0);  
[train,test] = crossvalind('HoldOut',groups);  
cp = classperf(groups);  
svmStruct = svmtrain(data(train,:),groups(train),'showplot',false,'kernel\_function','linear');  
classes = svmclassify(svmStruct,data(test,:),'showplot',false);  
classperf(cp,classes,test);  
Accuracy = cp.CorrectRate;  
Accuracy\_Percent(i) = Accuracy.\*100;  
end  
Max\_Accuracy = max(Accuracy\_Percent);  
sprintf('Accuracy of Linear Kernel with 500 iterations is: %g%%',Max\_Accuracy)