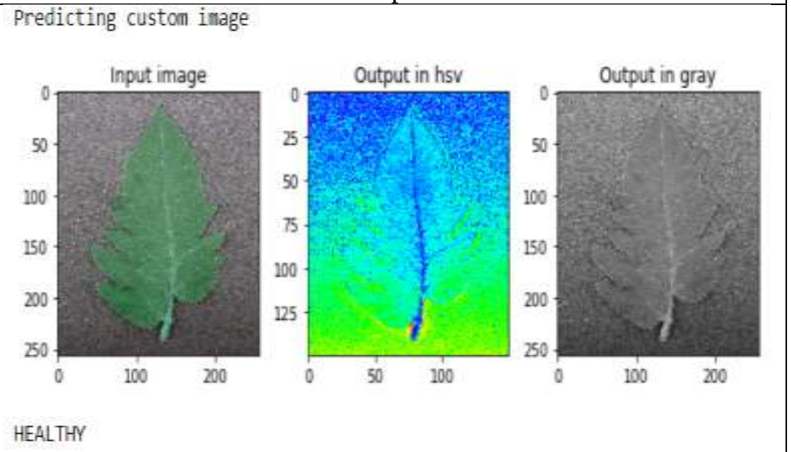
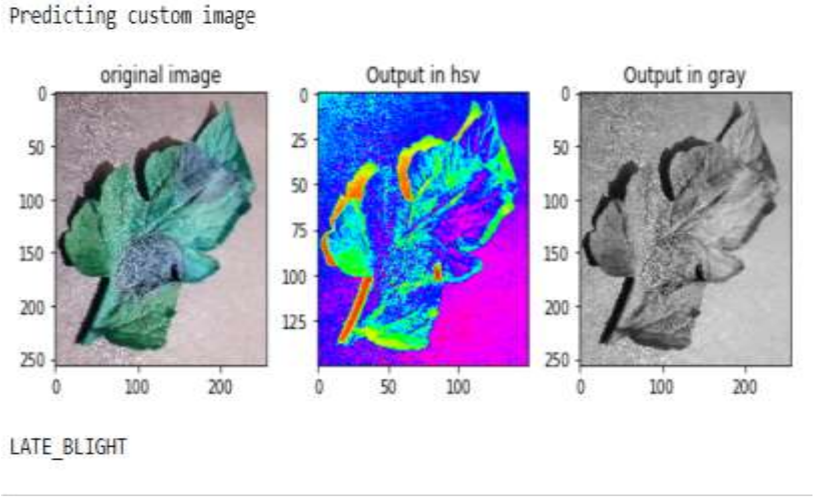
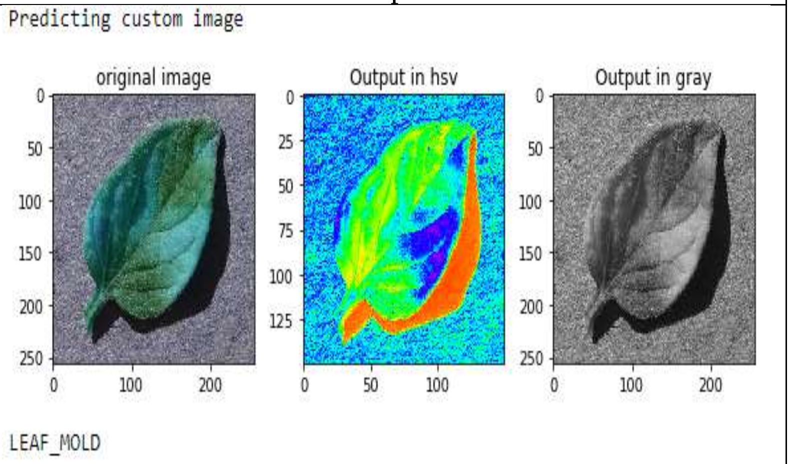
Input	Code	Output

Healthy
	print("Predicting custom image")
	labels = ["BACTERIAL_SPOT", "EARLY_BLIGHT", "
	HEALTHY", "LATE_BLIGHT", "LEAF_MOLD"]
	img = cv2.imread(r"C:/Users/aishwarya/Desktop/h1.JPG")
	img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
	img_pred = cv2.resize(img, (150, 150))
	#plt.imshow(img_pred,cmap="gray")
	img_pred2 = np.reshape(img_pred, (1, img_pred.shape[0
	]*img_pred.shape[1]))
	e=model.predict(img_pred2)
	#print(labels[(e[0])])
	NUM_ROWS = 1
	IMGs_IN_ROW = 3
	f, ax = plt.subplots(NUM_ROWS, IMGs_IN_ROW, figsize
	=(8,8))
	img1 = cv2.imread(r"C:/Users/aishwarya/Desktop/h1.JPG")
	ax[0].imshow(img1)
	ax[1].imshow(img_pred,cmap="hsv")
	img3 = cv2.imread(r"C:/Users/aishwarya/Desktop/h1.JPG")
	img3 = cv2.cvtColor(img3, cv2.COLOR_BGR2GRAY)
	ax[2].imshow(img3,cmap="gray")
	
	ax[0].set_title('Input image')
	ax[1].set_title('Output in hsv')
	ax[2].set_title('Output in gray')
	
	title = 'predicting custom image'
	plt.tight_layout()
	plt.show()
	print(labels[(e[0])])
		
		Predicting custom image
		 Input image
		Output in hsv
		Output in gray
		0
		50
		100
		150
		200
		250
		0
		100
		200
		0
		25
		50
		75
		100
		125
		0
		50
		100
		0
		50
		100
		150
		200
		250
		0
		100
		200
		HEALTHY

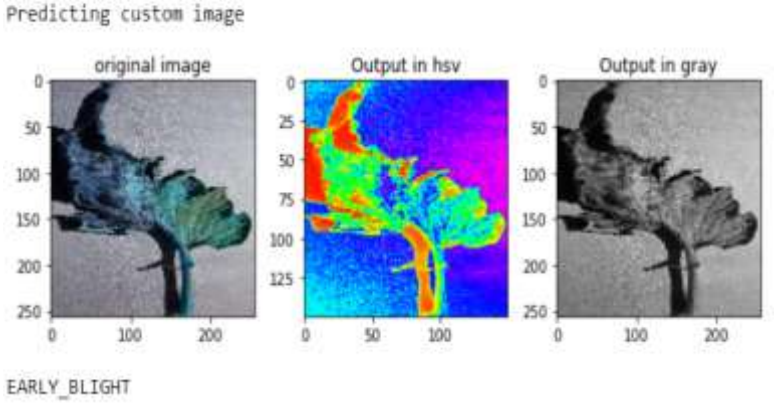
Input	Code	Output

Late Blight
	print("Predicting custom image")
	labels = ["BACTERIAL_SPOT", "EARLY_BLIGHT", "HEALTHY", "LATE_BLIGHT", "LEAF_MOLD"]
	img = cv2.imread(r"C:/Users/aishwarya/Desktop/lb1.JPG")
	img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
	img_pred = cv2.resize(img, (150, 150))
	img_pred2 = np.reshape(img_pred, (1, img_pred.shape[0]*img_pred.shape[1]))
	e=model.predict(img_pred2)
	NUM_ROWS = 1
	IMGs_IN_ROW = 3
	f, ax = plt.subplots(NUM_ROWS, IMGs_IN_ROW, figsize=(8,8))
	img1 = cv2.imread(r"C:/Users/aishwarya/Desktop/lb1.JPG")
	ax[0].imshow(img1)
	ax[1].imshow(img_pred,cmap="hsv")
	img3 = cv2.imread(r"C:/Users/aishwarya/Desktop/lb1.JPG")
	img3 = cv2.cvtColor(img3, cv2.COLOR_BGR2GRAY)
	ax[2].imshow(img3,cmap="gray")
	
	ax[0].set_title('original image')
	ax[1].set_title('Output in hsv')
	ax[2].set_title('Output in gray')
	
	title = 'predicting custom image'
	plt.tight_layout()
	plt.show()
	print(labels[(e[0])])
		
		Predicting custom image
		 original image
		Output in hsv
		Output in gray
		0
		50
		100
		150
		200
		250
		0
		100
		200
		0
		25
		50
		75
		100
		125
		0
		50
		100
		LATE_BLIGHT

Input	Code	Output

Leaf mold
	print("Predicting custom image")
	labels = ["BACTERIAL_SPOT", "EARLY_BLIGHT", "HEALTHY", "LATE_BLIGHT", "LEAF_MOLD"]
	img = cv2.imread(r"C:/Users/aishwarya/Desktop/lm1.JPG")
	img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
	img_pred = cv2.resize(img, (150, 150))
	img_pred2 = np.reshape(img_pred, (1, img_pred.shape[0]*img_pred.shape[1]))
	e=model.predict(img_pred2)
	
	NUM_ROWS = 1
	IMGs_IN_ROW = 3
	f, ax = plt.subplots(NUM_ROWS, IMGs_IN_ROW, figsize=(8,8))
	img1 = cv2.imread(r"C:/Users/aishwarya/Desktop/lm1.JPG")
	ax[0].imshow(img1)
	ax[1].imshow(img_pred,cmap="hsv")
	img3 = cv2.imread(r"C:/Users/aishwarya/Desktop/lm1.JPG")
	img3 = cv2.cvtColor(img3, cv2.COLOR_BGR2GRAY)
	ax[2].imshow(img3,cmap="gray")
	
	ax[0].set_title('original image')
	ax[1].set_title('Output in hsv')
	ax[2].set_title('Output in gray')
	
	title = 'predicting custom image'
	plt.tight_layout()
	plt.show()
	print(labels[(e[0])])
		
		
		Predicting custom image
		LEAF_MOLD

Input	Code	Output

Early blight
	print("Predicting custom image")
	labels = ["BACTERIAL_SPOT", "EARLY_BLIGHT", "HEALTHY", "LATE_BLIGHT", "LEAF_MOLD"]
	img = cv2.imread(r"C:/Users/aishwarya/Desktop/eb1.JPG")
	img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
	img_pred = cv2.resize(img, (150, 150))
	img_pred2 = np.reshape(img_pred, (1, img_pred.shape[0] *img_pred.shape[1]))
	e=model.predict(img_pred2)
	NUM_ROWS = 1
	IMGs_IN_ROW = 3
	f, ax = plt.subplots(NUM_ROWS, IMGs_IN_ROW, figsize =(8,8))
	img1 = cv2.imread(r"C:/Users/aishwarya/Desktop/eb1.JPG")
	ax[0].imshow(img1)
	ax[1].imshow(img_pred,cmap="hsv")
	img3 = cv2.imread(r"C:/Users/aishwarya/Desktop/eb1.JPG")
	img3 = cv2.cvtColor(img3, cv2.COLOR_BGR2GRAY)
	ax[2].imshow(img3,cmap="gray")
	
	ax[0].set_title('original image')
	ax[1].set_title('Output in hsv')
	ax[2].set_title('Output in gray')
	
	title = 'predicting custom image'
	plt.tight_layout()
	plt.show()
	print(labels[(e[0])])
		
		Predicting custom image
		 original image
		Output in hsv
		Output in gray
		0
		50
		100
		150
		200
		250
		0
		100
		200
		0
		25
		50
		75
		100
		125
		0
		50
		100
		EARLY_BLIGHT

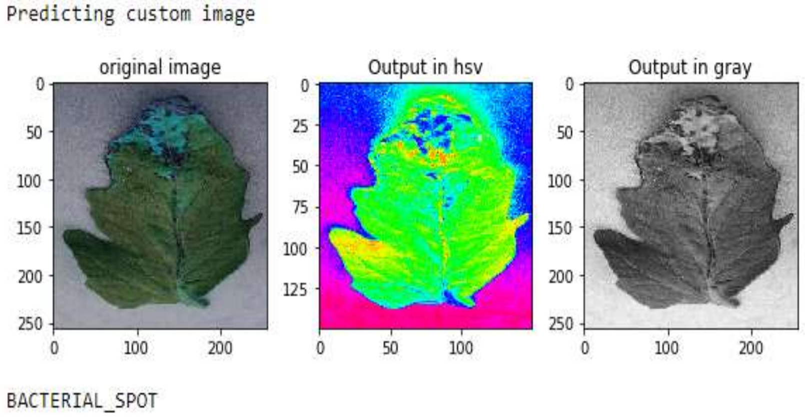
<table>
<tr><th>Input</th><th>Code</th><th>Output</th></tr>
<tr>
<td>

Bacterial spot

</td>
<td>

```
print("Predicting custom image")
labels = ["BACTERIAL_SPOT", "EARLY_BLIGHT", "HEALTHY", "LATE_BLIGHT", "LEAF_MOLD"]
img = cv2.imread(r"C:/Users/aishwarya/Desktop/bs1.JPG")
img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
img_pred = cv2.resize(img, (150, 150))
img_pred2 = np.reshape(img_pred, (1, img_pred.shape[0]*img_pred.shape[1]))
e=model.predict(img_pred2)

NUM_ROWS = 1
IMGs_IN_ROW = 3
f, ax = plt.subplots(NUM_ROWS, IMGs_IN_ROW, figsize=(8,8))
img1 = cv2.imread(r"C:/Users/aishwarya/Desktop/bs1.JPG")
ax[0].imshow(img1)
ax[1].imshow(img_pred,cmap="hsv")
img3 = cv2.imread(r"C:/Users/aishwarya/Desktop/bs1.JPG")
img3 = cv2.cvtColor(img3, cv2.COLOR_BGR2GRAY)
ax[2].imshow(img3,cmap="gray")

ax[0].set_title('original image')
ax[1].set_title('Output in hsv')
ax[2].set_title('Output in gray')

title = 'predicting custom image'
plt.tight_layout()
plt.show()
print(labels[(e[0])])
```

</td>
<td>

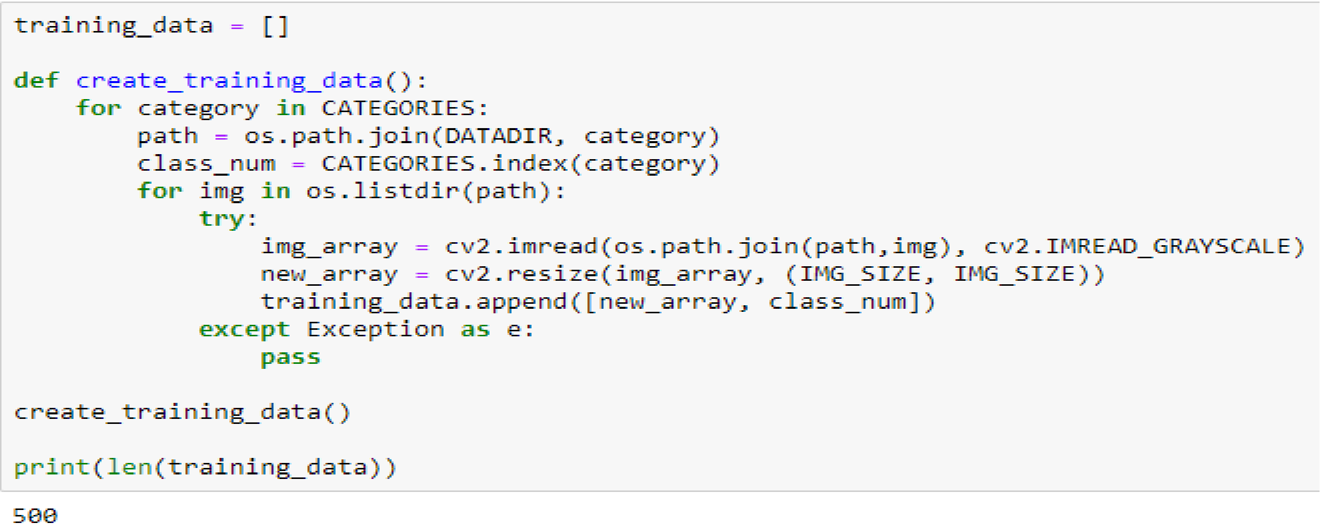


```
Predicting custom image
```

```
BACTERIAL_SPOT
```

</td>
</tr>
</table>

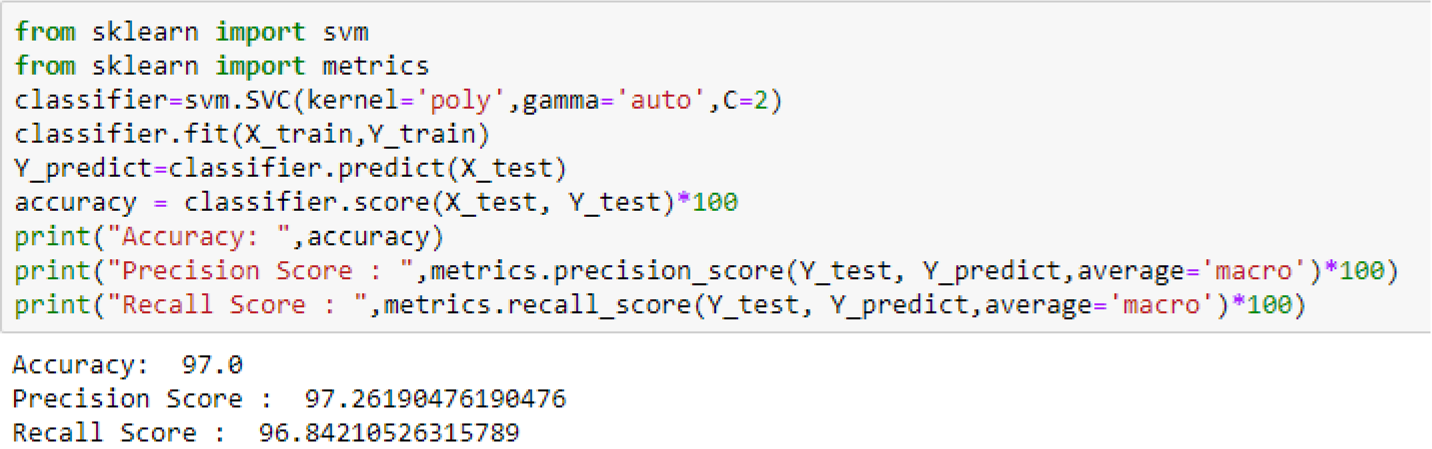
- **Number of images trained**



```
training_data = []

def create_training_data():
    for category in CATEGORIES:
        path = os.path.join(DATADIR, category)
        class_num = CATEGORIES.index(category)
        for img in os.listdir(path):
            try:
                img_array = cv2.imread(os.path.join(path,img), cv2.IMREAD_GRAYSCALE)
                new_array = cv2.resize(img_array, (IMG_SIZE, IMG_SIZE))
                training_data.append([new_array, class_num])
            except Exception as e:
                pass

create_training_data()

print(len(training_data))
```

```
500
```

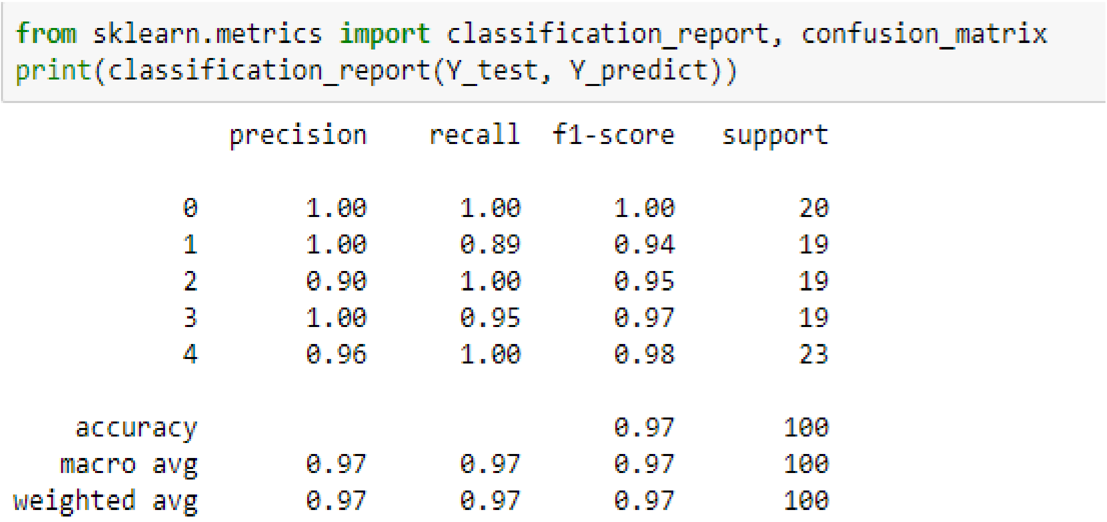
- **Accuracy**



```
from sklearn import svm
from sklearn import metrics
classifier=svm.SVC(kernel='poly',gamma='auto',C=2)
classifier.fit(X_train,Y_train)
Y_predict=classifier.predict(X_test)
accuracy = classifier.score(X_test, Y_test)*100
print("Accuracy: ",accuracy)
print("Precision Score : ",metrics.precision_score(Y_test, Y_predict,average='macro')*100)
print("Recall Score : ",metrics.recall_score(Y_test, Y_predict,average='macro')*100)
```

```
Accuracy:  97.0
Precision Score :  97.26190476190476
Recall Score :  96.84210526315789
```

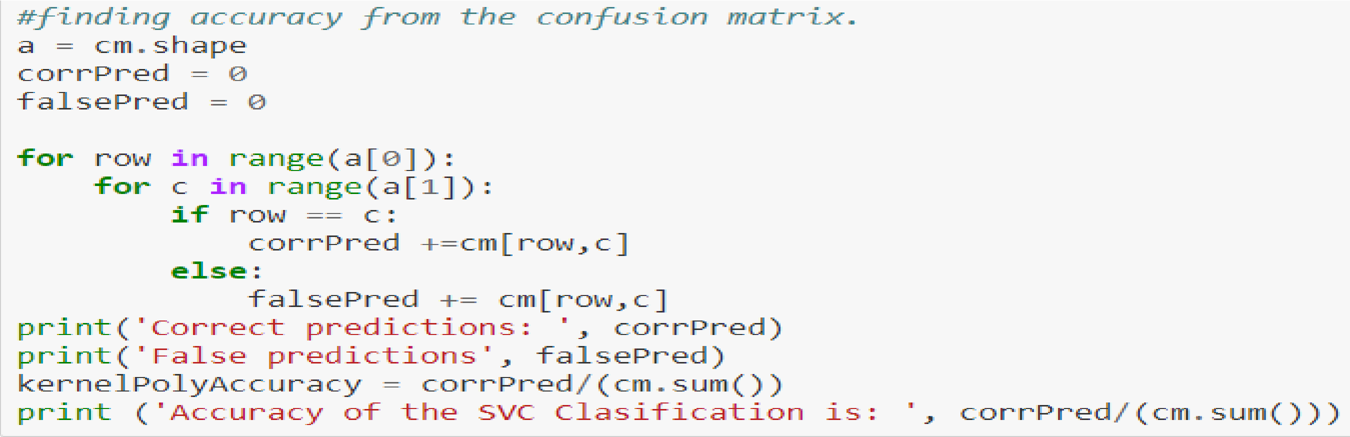
- **Precision classification Report**



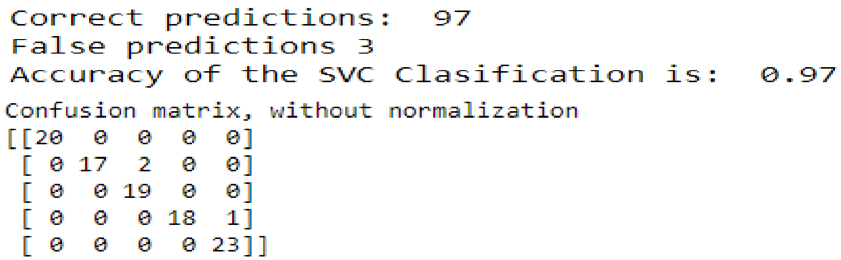
```python
from sklearn.metrics import classification_report, confusion_matrix
print(classification_report(Y_test, Y_predict))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        20
           1       1.00      0.89      0.94        19
           2       0.90      1.00      0.95        19
           3       1.00      0.95      0.97        19
           4       0.96      1.00      0.98        23

    accuracy                           0.97       100
   macro avg       0.97      0.97      0.97       100
weighted avg       0.97      0.97      0.97       100
```

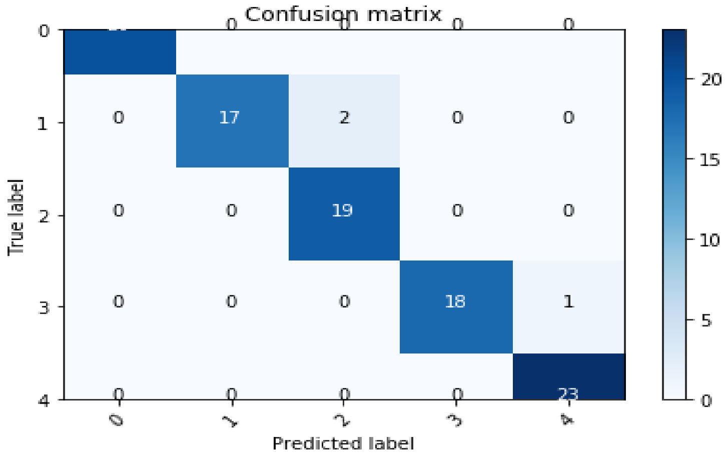
- **Confusion matrix (graph)**



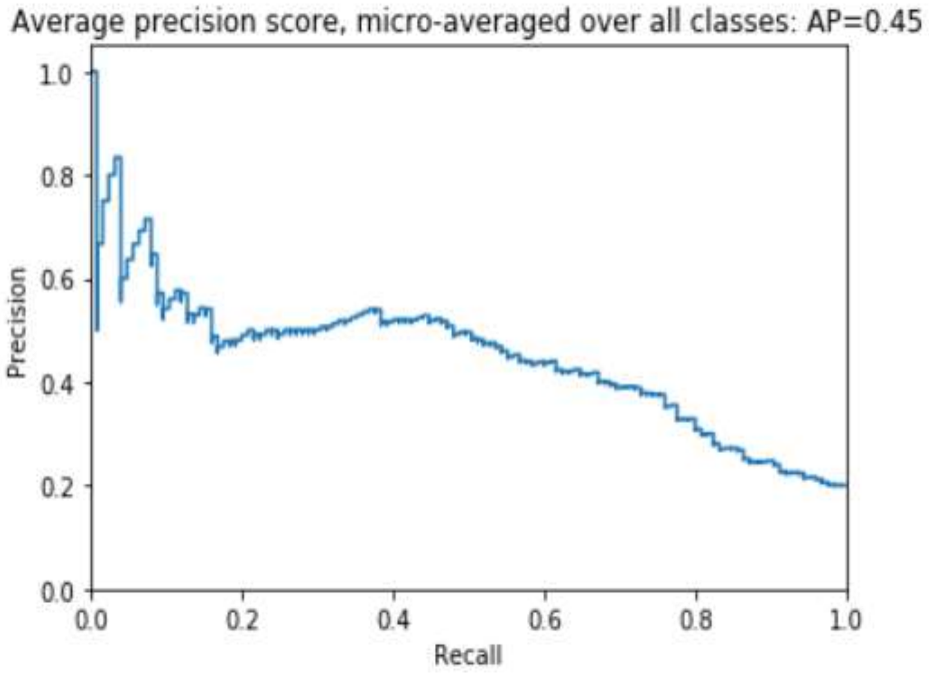
```
#finding accuracy from the confusion matrix.
a = cm.shape
corrPred = 0
falsePred = 0

for row in range(a[0]):
    for c in range(a[1]):
        if row == c:
            corrPred +=cm[row,c]
        else:
            falsePred += cm[row,c]
print('Correct predictions: ', corrPred)
print('False predictions', falsePred)
kernelPolyAccuracy = corrPred/(cm.sum())
print ('Accuracy of the SVC Clasification is: ', corrPred/(cm.sum()))
```



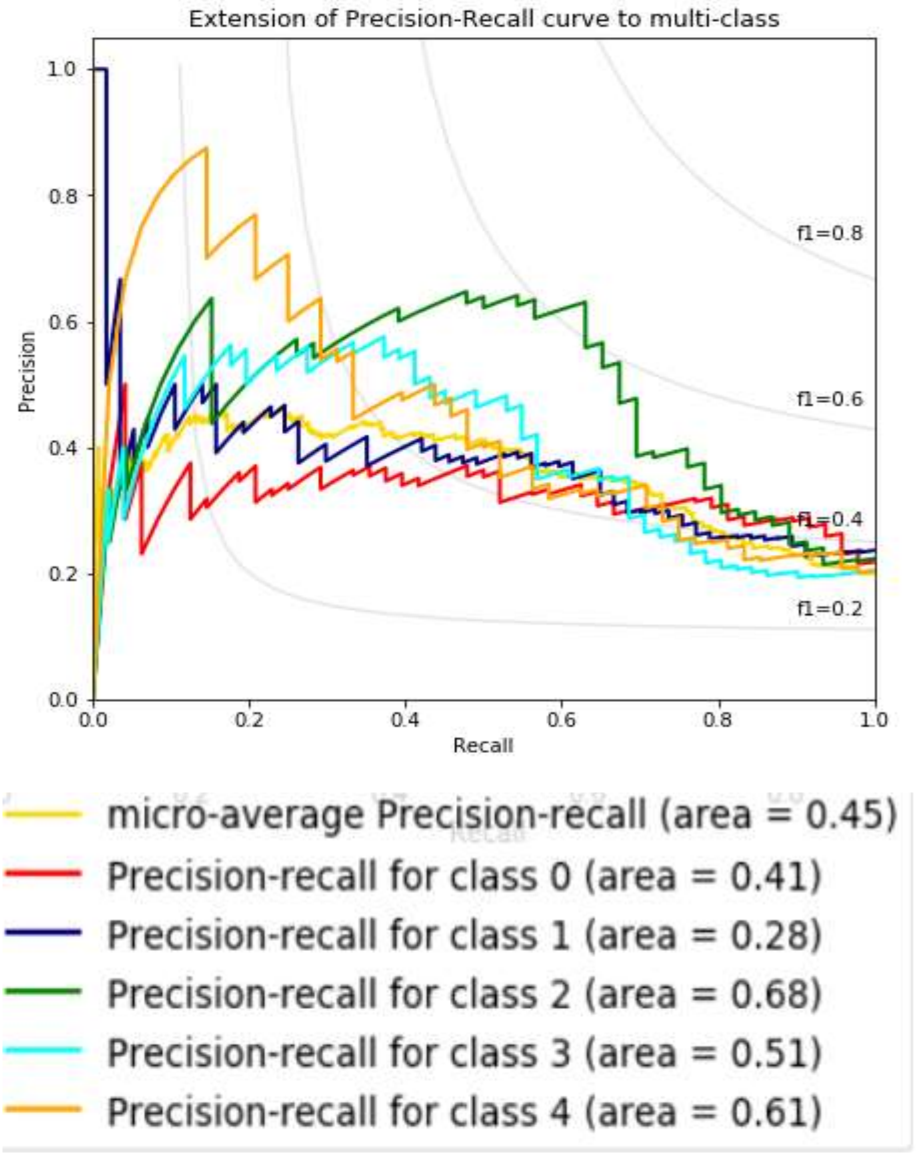
```
Correct predictions:  97
False predictions 3
Accuracy of the SVC Clasification is:  0.97
Confusion matrix, without normalization
[[20  0  0  0  0]
 [ 0 17  2  0  0]
 [ 0  0 19  0  0]
 [ 0  0  0 18  1]
 [ 0  0  0  0 23]]
```



- **Graph (precision-recall curve):**



- **Precision recall curve for each class**



Class 0-Bacterial_spot
Class 1-Early blight
Class 2-Healthy
Class 3- Late blight
Class 4-Leaf mold