In [1]:

```
from pathlib import Path
import matplotlib.pyplot as plt
import numpy as np
%matplotlib notebook
from sklearn import svm, metrics, datasets
from sklearn.utils import Bunch
from sklearn.model_selection import GridSearchCV, train_test_split

from skimage.io import imread
from skimage.transform import resize
import skimage
```

In [2]:

```
def load_image_files(container_path, dimension=(104, 104)):

    image_dir = Path(container_path)
    folders = [directory for directory in image_dir.iterdir() if directory.is_dir()]
    categories = [fo.name for fo in folders]
    descr = "A Rice Disease detection using SVM"
    images = []
    flat_data = []
    target = []
    for i, direc in enumerate(folders):
        for file in direc.iterdir():
            img = skimage.io.imread(file)
            img_resized = resize(img, dimension, anti_aliasing=True, mode='reflect')
            flat_data.append(img_resized.flatten())
            images.append(img_resized)
            target.append(i)

    flat_data = np.array(flat_data)
    target = np.array(target)
    images = np.array(images)
    return Bunch(data=flat_data,
                 target=target,
                 target_names=categories,
                 images=images,
                 DESCR=descr)
```

In [3]:

```
image_dataset = load_image_files("C:/Users/kusha/PycharmProjects/Rice_Disease_prediction_
GUI/rice_leaf_diseases/") #Load here dataset
print(image_dataset.target_names)
```

```
['Bacterial leaf blight', 'Brown spot', 'Leaf smut']
```

In [4]:

```
print(image_dataset.target_names)
```

```
['Bacterial leaf blight', 'Brown spot', 'Leaf smut']
```

In [5]:

```
X_train, X_test, y_train, y_test = train_test_split(image_dataset.data, image_dataset.ta
rget, test_size=0.3,random_state=109)
param_grid = [
  {'C': [1, 10, 100, 1000], 'kernel': ['linear']},
  {'C': [1, 10, 100, 1000], 'gamma': [0.001, 0.0001], 'kernel': ['rbf']},
 ]
svc = svm.SVC()
clf = GridSearchCV(svc, param_grid)
clf.fit(X_train, y_train)
```

```
C:\Users\kusha\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:1978: Future
```

```
Warning: The default value of cv will change from 3 to 5 in version 0.22. Specify it expl
icitly to silence this warning.
  warnings.warn(CV_WARNING, FutureWarning)
C:\Users\kusha\Anaconda3\lib\site-packages\sklearn\model_selection\_search.py:814: Deprec
ationWarning: The default of the `iid` parameter will change from True to False in versio
n 0.22 and will be removed in 0.24. This will change numeric results when test-set sizes
are unequal.
  DeprecationWarning)
```

Out[5]:

```
GridSearchCV(cv='warn', error_score='raise-deprecating',
             estimator=SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
                           decision_function_shape='ovr', degree=3,
                           gamma='auto_deprecated', kernel='rbf', max_iter=-1,
                           probability=False, random_state=None, shrinking=True,
                           tol=0.001, verbose=False),
             iid='warn', n_jobs=None,
             param_grid=[{'C': [1, 10, 100, 1000], 'kernel': ['linear']},
                         {'C': [1, 10, 100, 1000], 'gamma': [0.001, 0.0001],
                          'kernel': ['rbf']}],
             pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
             scoring=None, verbose=0)
```

In [6]:

```
y_pred = clf.predict(X_test)
```

In [7]:

```
print(clf.score(X_test,y_test))
```

```
0.8055555555555556
```

In [8]:

```
print("Classification report for - \n{}:\n{}\n".format(
    clf, metrics.classification_report(y_test, y_pred)))
```

```
Classification report for -
GridSearchCV(cv='warn', error_score='raise-deprecating',
             estimator=SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
                           decision_function_shape='ovr', degree=3,
                           gamma='auto_deprecated', kernel='rbf', max_iter=-1,
                           probability=False, random_state=None, shrinking=True,
                           tol=0.001, verbose=False),
             iid='warn', n_jobs=None,
             param_grid=[{'C': [1, 10, 100, 1000], 'kernel': ['linear']},
                         {'C': [1, 10, 100, 1000], 'gamma': [0.001, 0.0001],
                          'kernel': ['rbf']}],
             pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
             scoring=None, verbose=0):
              precision    recall  f1-score   support

           0       0.77      0.83      0.80        12
           1       0.78      0.88      0.82         8
           2       0.86      0.75      0.80        16

    accuracy                           0.81        36
   macro avg       0.80      0.82      0.81        36
weighted avg       0.81      0.81      0.81        36
```

In [9]:

```
from sklearn.metrics import confusion_matrix
```

In [10]:

```
cm = confusion_matrix(y_test,y_pred)
```

```
print(cm)
```

```
[[10  1  1]
 [ 0  7  1]
 [ 3  1 12]]
```

In [11]:

```
print(y_pred)
```

```
[2 0 2 2 2 0 0 2 1 2 0 2 1 1 2 0 0 2 0 2 1 1 0 0 0 0 2 2 2 1 1 1 1 0 2 0]
```

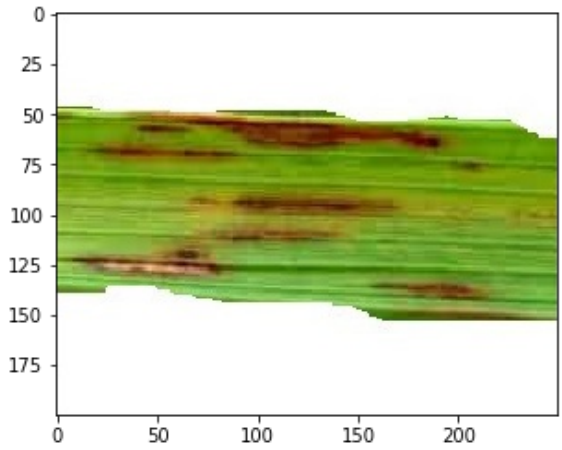
In [16]:

```
from PIL import Image
import os
def load_image(file):
    dimension=(104, 104)
    image = Image.open(file)
    flat_data = []
    img = skimage.io.imread(file)
    img_resized = resize(img, dimension, anti_aliasing=True, mode='reflect')
    flat_data.append(img_resized.flatten())
    return image,flat_data
```

In [17]:

```
plot , img = load_image(r'C:/Users/kusha/PycharmProjects/Rice_Disease_prediction_GUI/rice_leaf_diseases/Leaf smut/DSC_0504.jpg')
%matplotlib inline
plt.imshow(plot)
plt.show()
k = image_dataset.target_names
p = clf.predict(img)

s = [str(i) for i in p]
a = int("".join(s))
print("Predicted Disease is", k[a])
```




```
Predicted Disease is Leaf smut
```

In [20]:

```
import pickle
# now you can save it to a file
with open(r'C:\Users\kusha\PycharmProjects\Rice_Disease_prediction_GUI\rice_pred.pkl', 'wb') as f:
    pickle.dump(clf, f)
```

In [21]:

```
with open(r'C:\Users\kusha\PycharmProjects\Rice_Disease_prediction_GUI\rice_pred.pkl', 'rb') as f:
    clf1 = pickle.load(f)
```

In [23]:

```
cy = clf1.predict(img)
print(cy)
```

```
[2]
```

In [19]:

```
import sklearn
print(sklearn.__version__)
```

```
0.21.3
```

In [ ]:

```

```