Chapter 6

Classification of Diseases

6.1 Introduction

Classification is the most predominant task for disease prediction and data classification. Classification is basically the phase where decision making is done [136], [137]. In this, users utilize the features which are extracted in the features extraction stage. Decision making is done in the classification using various classifiers like naive bayes, decision tree, k nearest neighbor (KNN), support vector machine (SVM), artificial neural network (ANN), deep neural networks (DNN) etc. Figure 6.1 shows a basic classifier in which the features values (X1 to Xn) are input and the output Y is a category that is classified.

![Basic Classifier Diagram](image)

In this research, the four different classifiers that are used to classify the diseased leaves are naive bayes, decision tree, support vector machine and k nearest neighbor. For the comparative analysis of classifier, different
performance metrics are considered. A brief introduction about the performance metrics and four classifiers has been given in the following sections.

6.2 Performance Metrics

Disease classification is done on the basis of whether the apple diseased leaf belongs to apple scab or marsonina coronaria disease. So, four different possible events are possible which include pixel classifications and pixel misclassifications. True positive (TP) and true negative (TN) are the two pixel classifications and false positive (FP), and false negative (FN) is the two pixel misclassifications shown in Table 6.1. Number of these four different classifications can be used for evaluation of various performance metrics.

Table 6.1 Performance Metrics

<table>
<thead>
<tr>
<th></th>
<th>p' (Predicted)</th>
<th>n' (Predicted)</th>
</tr>
</thead>
<tbody>
<tr>
<td>P (Actual)</td>
<td>True Positive</td>
<td>False Negative</td>
</tr>
<tr>
<td>n (Actual)</td>
<td>False Positive</td>
<td>True Negative</td>
</tr>
</tbody>
</table>

An event is classified as TP if diseased pixel is correctly identified as diseased and TN if the non-diseased pixel correctly identified as non-diseased pixel. An event is classified as FN if the predicted pixel represents non-diseased pixel but actually it was diseased pixel. An event is said to be FP if the predicted pixel represents diseased pixel but actually it was non-diseased pixel. Using these possible events, two performance metrics are measured i.e. true positive rate (TPR) and false positive rate (FPR) for all the classification algorithms. True positive rate and false positive rate plays a very important role in evaluating many machine learning algorithms and
holds a great significance. True positive rate processes the section of positives that are appropriately identified and false positive rate measure the fraction of positives that are not correctly identified [138]–[142] and can be mathematically shown in Equation (6.1) and (6.2).

True positive rate = \( \frac{TP}{TP+FN} \) \hspace{1cm} (6.1)
False Positive Rate = \( \frac{FP}{FP+TN} \) \hspace{1cm} (6.2)

The important performance metrics which can be derived from the above events are sensitivity (SN), specificity (SP), accuracy (Acc), precision (Pr), f_measure (FM), goodness mean (GM) and area under the curve (AUC).

(i) Sensitivity
Sensitivity metrics represents the capability of an algorithm to identify the diseased apple leaf. Sensitivity is the ratio of TP to the sum of TP and FN. Range of SN is between 0 and 1. More sensitivity means algorithm is able to identify diseased leaf correctly. SN measure is expressed as by Equation (6.3).

\[ SN = \frac{TP}{TP+FN} \] \hspace{1cm} (6.3)

(ii) Specificity
Specificity metrics represents the capability of a classification technique to identify background or non-diseased images. Specificity is the ratio of TN to the sum of TN and FP. Range of SP is also between 0 and 1. More SP means algorithm can identify non diseased leaf correctly. SP measures is expressed by Equation (6.4).

\[ SP = \frac{TN}{TN+FP} \] \hspace{1cm} (6.4)

(iii) Accuracy
Accuracy (Acc) metrics is evaluated by taking the ratio of total number of true events which is the sum of TP and TN, to the total elements which is
aggregate number of pixels actually present in the image. Accuracy measure is expressed by Equation (6.5).

\[
\text{Acc} = \frac{\sum TP + \sum TN}{\sum \text{Total elements}} \tag{6.5}
\]

(iv) Precision

Precision metrics is evaluated by taking the ratio of true positive to the summation of true positive and false positive. Precision measure is expressed by Equation (6.6).

\[
Pr = \frac{TP}{TP + FP} \tag{6.6}
\]

(v) F_Measure

F-Measure metrics is evaluated by taking ratio of double the product of precision and sensitivity to the sum of precision and sensitivity. F-Measure is expressed by Equation (6.7).

\[
\text{FM} = \frac{2 \times (\text{precision} \times \text{sensitivity})}{\text{precision} + \text{sensitivity}} \tag{6.7}
\]

(vi) Goodness Mean

G-Mean metrics is evaluated by taking ratio of square of TPR to TNR, where TPR is the true positive rate and TNR is the true negative rate. G-Mean is expressed by Equation (6.8).

\[
\text{GM} = \sqrt{\frac{TPR}{TNR}} \tag{6.8}
\]

(vii) Area Under the Curve (AUC)

It is a simple measurement metric used to measure the accuracy by taking the definite integral which is utilized to locate the region amid the graph curve and the ‘x’ axis between two given ‘x’ values, is called area under the curve. The value of this method is in the range between 0 and 1. Greater value of AUC shows a better performance of the classification. It is calculated using Equation (6.9).
\[ AUC = \int_x^y f(a) \, da \]  

(6.9)

where ‘x’ and ‘y’ are the minimum and maximum axis points in the curve.

### 6.3 Database for Training and Testing

In the experiments, two main files are generated, namely: training texture feature data and testing texture feature data. Each row in the file has 10 columns representing the 10 texture features extracted for a particular sample image. Each row has a unique number (1, 2) which represents the class (i.e. the disease) of the particular row of data. “1” represents *marsonina coronaria*, “2” represents apple scab. Then, a software program is written in MATLAB that will take in .mat files representing the training and testing data, train the classifier using the “train files”, and then use them to perform the classification task on the test data. Consequently, a Matlab routine loads all the data files (training and testing data files) and match them to generate the result which identifies the particular disease.

### 6.4 K-Nearest Neighbour (KNN)

K-NN classifier has been used by a number of researchers for the purpose of classification. In the recognition of patterns, various algorithms are used for the classification of the objects. K-NN is one of the classification methods which are based on classifying the objects on the basis of closest training examples. K-NN is a sort of occasion based learning. In this, the function is approximated locally and the computations are deferred until the classification [143]. KNN is the central and most straightforward classification method when there is very less prior information regarding the circulation of data. KNN is a standout amongst the most well-known classification algorithms for recognition of patterns. Numerous analysts have discovered that the KNN calculation achieves great results in their tests on various information sets. However KNN classification has three confinements: (i) complexity is more because all the training samples are utilized in classification (ii) the system’s performance exclusively depends on training set and (iii) no weight variation between sample points. The Nearest Neighbor (NN) govern is the easiest type
of KNN when $K = 1$. In this strategy each sample ought to be classified identically to the neighboring samples. In this manner, if the classification of a specific sample is not known, the prediction can be made by the classification of its closest neighbor samples. Given a training set and a query sample, the distance between the training set samples and the query set samples can be calculated. Therefore, the unknown sample can be classified on the with respect to classification of the nearest neighbour [144].

The execution of a KNN classifier is principally controlled by the selection of $K$ and the applied distance or separation metric. Choosing the right value of parameter ‘$K$’ usually depends on the data. Usually to remove noise in the classification, ‘$K$ must be chosen to the higher side but it makes the boundaries between the classes less distinctive. An effective value of parameter ‘$K$’ can be chosen using various techniques but commonly used technique is cross-validation. If the value of the parameter $K$ is very small, then the local estimate will be poor due to ambiguous and mislabelled points. Now in order to smoothen the estimate, the value of $K$ can be increased but a larger value of the $K$ leads to over smoothing and the performance of the classifier decreases due to outliers. To discover the $k$ closest neighbors, the similarities between the training samples have to be found [145]. When the samples for training are less, the KNN classifier is no longer ideal but if there are large number of samples, then KNN classifier needs additional time to find out the similarities. This issue can be resolved in 3 different ways: by decreasing the feature space dimensions; utilizing smaller data set; utilizing enhanced algorithms [146]. The classifier is produced just with the samples of training and it doesn't utilize any extra information. There is no distinction between the samples with modest number of information and gigantic number of information. So it doesn't coordinate the real wonder where the samples have uneven dissemination [147], [148]. Generally Euclidean distance is utilized, but other than this some other distance measures, like Manhattan distance can also be utilized. KNN algorithm can be understood by using a simple example of separating two classes of squares and triangles as shown in Fig. 6.2.
The test or query instance which is a circle can be classified to the square class or triangle class. If ‘K’ is three then it will be categorized to the triangle’s class as it has two triangles and simply one square within the interior circle. However if K is five, then it will be classified to square’s class as there are three squares and two triangles in the outer circle [149]. The feature space is divided into different regions based on labels and locations of training examples. The distance from the fresh vector to all stored vectors is calculated. There are numerous ways for classifying the new vector to the specific class, but the most commonly used technique is to envisage the new vector to the most familiar class.

6.4.1 Results of KNN Classifier

In this research work, K nearest neighbour classifier is implemented on the extracted first order and second order features from the segmented ROI images. The Matlab command used for the implementation of K nearest neighbour classifier is given below

```
Knn=ClassificationKNN.fit(Xtrain,Ytrain,'NumNeighbors',5,'NSMethod','exhaustive','Distance','euclidean','DistanceWeight','inverse','Standardize',1);
```

where ‘Xtrain’ is a matrix having different features values of all the images in training data set whereas ‘Ytrain’ is matrix having output labels of all the images in the training dataset. This command returns a binary classification k nearest neighbour named as ‘Knn’ based on the input variables ‘Xtrain’ and ‘Ytrain’. After that K nearest neighbour ‘Knn’ is used to predict the labels of
test data based upon the extracted features of test images which are saved in the ‘comfeat’. For the prediction of labels of test data, following matlab command is used which will give the labels in the variable ‘Y_knn’.

\[ [Y\_knn, score\_nb1] = knn.predict(comfeat) \]

Based upon the predicted labels ‘Y_knn’, different performance metrics like, accuracy, sensitivity specificity, precision, f_measure, goodness mean and area under the curve are evaluated whose values are given in the Table 6.2

<table>
<thead>
<tr>
<th>Performance Metrics</th>
<th>K Nearest Neighbour</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>0.9641</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>0.9000</td>
</tr>
<tr>
<td>Specificity</td>
<td>0.9827</td>
</tr>
<tr>
<td>Precision</td>
<td>0.9375</td>
</tr>
<tr>
<td>F_Measure</td>
<td>0.9184</td>
</tr>
<tr>
<td>Goodness Mean</td>
<td>0.9404</td>
</tr>
<tr>
<td>AUC</td>
<td>0.9946</td>
</tr>
</tbody>
</table>

6.5 Support Vector Machine (SVM)

In SVM, each data item is plotted as a point in k-dimensional space (where k is number of features). Each feature’s value is considered as the value of the particular coordinate. After this, classification is usually performed by taking appropriate hyper plane which distinguishes the classes well. SVM was introduced by Vapnik and this algorithm has gained interest of research community from all over the world [150]. Generally SVM classifier accepts input data & classify the input data into two distinct classes. The training of the classifier is done with some training data & a model is made for the classification of test data. Sometimes there is a problem arises named as
multiclass classification. For this there will be a need to design multiple binary classifiers. Several studies related to SVM have confirmed that SVM offers quite good classification accuracy than other classification techniques available these days. SVM can be utilized in the classification of the images. Investigational results have shown that SVMs attain considerably higher accuracy in comparison with various traditional classifiers. Though, for a number of datasets, the SVM’s performance is very susceptible to how the cost and kernel parameters are set. There are number of kernel functions in this algorithm: (1) Polynomial kernel (2) Lineal kernel (3) Gaussian radial basis function (4) Sigmoid or tangent kernel [151], [152].

From the last few years, SVMs have turned out to be exceptionally compelling in classification of real world tasks. The functional form of SVM is usually defined before the training. For training the SVM, a labelled training set is required, because SVM fits the function from the set of examples. There might be N umber of examples in the training set. In SVM, every data instance has to be represented as a vector of real numbers. However, if categorical attributes are there, then these attributes need to be converted into numerical data. Generally it is highly suggested to operate m numbers for demonstrating an m category attribute. Only one out of the m numbers is unity and others are zero. For ex. a four-category attribute such as {blue, green, red, black} can be represented as (0,0,0,1), (0,0,1,0), (0,1,0,0) and (1,0,0,0).

SVMs can also be utilized to recognize plant leaf disease. Further SVM algorithm has also been extensively used in organic or biological sciences [154]. SVM has been utilized in an extensive variety of genuine world issues, for example, plant leaf disease classification, classification of different images, pattern recognition and detection of objects etc. [155], [156].

6.5.1 Results of SVM Classifier
In this research work, support vector machine classifier is implemented on the extracted first order and second order features from the segmented ROI.
images. The Matlab command used for the implementation of decision tree classifier is given below.

SVMModel=fitcsvm(Xtrain,Ytrain,'Standardize',true,'KernelFunction','RBF','KernelScale','auto');

where ‘Xtrain’ is a matrix having different features values of all the images in training data set whereas ‘Ytrain’ is matrix having output labels of all the images in the training dataset. This command returns a binary classification decision tree named as ‘SVMModel’ based on the input variables ‘Xtrain’ and ‘Ytrain’. After that decision tree ‘dtree’ is used to predict the labels of test data based upon the extracted features of test images which are saved in the ‘comfeat’. For the prediction of labels of test data, following matlab command is used which will give the labels in the variable ‘LL’.

[LL,score] = predict(SVMModel,comfeat);

Based upon the predicted labels ‘LL’, different performance metrics like, accuracy, sensitivity, specificity, precision, f_measure goodness mean and area under the curve are evaluated whose values are given in the Table 6.3.

<table>
<thead>
<tr>
<th>Performance Metrics</th>
<th>Support Vector Machine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>0.8879</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>0.5600</td>
</tr>
<tr>
<td>Specificity</td>
<td>0.9827</td>
</tr>
<tr>
<td>Precision</td>
<td>0.9032</td>
</tr>
<tr>
<td>F_Measure</td>
<td>0.6914</td>
</tr>
<tr>
<td>GoodnessMean</td>
<td>0.7418</td>
</tr>
<tr>
<td>AUC</td>
<td>0.9729</td>
</tr>
</tbody>
</table>
6.6 Decision Tree Classifier

Decision Tree uses a tree of observations about an item to conclusions about an item’s target. Decision Tree Classifier poses a series of carefully crafted questions about the attributes of the test record [157]. Each time it receives an answer, a follow up question is probed until a decision about the class label of the class is recorded [158].

In decision analysis decision tree and the meticulously related influence drawings are executed as a pictorial and analytical decision support tool, where the expected values of competing alternatives are calculated [159]– [161].

6.6.1 Results of Decision Tree Classifier

In this research work, decision tree classifier was implemented on the extracted 1st order and 2nd order features from the segmented ROI images. The matlab command used for the implementation of decision tree classifier is given below:

\[
dtree = \text{fitctree}(\text{Xtrain}, \text{Ytrain});
\]

where ‘Xtrain’ is a matrix having different features values of all the images in training data set whereas ‘Ytrain’ is matrix having output labels of all the images in the training dataset. This command returns a binary classification decision tree named as ‘dtree’ based on the input variables ‘Xtrain’ and ‘Ytrain’. After that decision tree ‘dtree’ is used to predict labels of test data based upon the extracted features of test images which are saved in the ‘testfeat’. For the prediction of labels of test data, following matlab command is used which will give the labels in the variable ‘Ypredict’.

\[
[Ypredict, \text{score}_{dtree}] = \text{dtree.predict}(\text{testfeat});
\]

Based upon the predicted labels ‘Ypredict’, different performance metrics like, accuracy sensitivity, specificity, precision, f_measure, goodness mean and area under the curve are evaluated whose values are given in the Table 6.4.
**Table 6.4 Performance Metrics for Decision Tree**

<table>
<thead>
<tr>
<th>Performance Metrics</th>
<th>Decision Tree</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>0.8879</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>0.8600</td>
</tr>
<tr>
<td>Specificity</td>
<td>0.8960</td>
</tr>
<tr>
<td>Precision</td>
<td>0.7049</td>
</tr>
<tr>
<td>F_Measure</td>
<td>0.7748</td>
</tr>
<tr>
<td>Goodness Mean</td>
<td>0.8778</td>
</tr>
<tr>
<td>AUC</td>
<td>0.9313</td>
</tr>
</tbody>
</table>

6.7 **Naive Bayes**

Naive bayes method is a modest technique for building classifiers models that allot class labels to problem occurrences, characterized as vectors of feature value, where the class label is drawn from a finite set [162]. There is no single system for training such classifiers, but a group of procedures grounded on a common principle. All naive bayes classifiers adopt that the value of a certain feature is liberated of the value of any other feature, given the class variable [163].

For specific sorts of probability models, naive bayes classifiers can be trained very proficiently in supervised learning background. In several practical uses, parameter approximation for naive bayes models uses the scheme of maximum likelihood in other words, an individual can work with the naive bayes model devoid of accepting bayesian probability or using any bayesian methods [164]. Naive Bayes classifiers are a set of supervised learning systems based on relating bayes theorem with the “naive” assumption of independence between every pair of features [165].
6.7.1 Results of Naive Bayes Classifier

In this research work, k nearest neighbour classifier is implemented on the extracted 1st order and 2nd order features from the segmented ROI images. The matlab command used for the implementation of decision tree classifier is given below:

\[ \text{Nb} = \text{NaiveBayes.fit}(\text{Xtrain}, \text{Ytrain}) ; \]

where ‘Xtrain’ is a matrix having different features values of all the images in training data set whereas ‘Ytrain’ is matrix having output labels of all the images in the training dataset. This command returns a binary classification decision tree named as ‘Nb’ based on the input variables ‘Xtrain’ and ‘Ytrain’. After that decision tree ‘Nb’ is used to calculate the labels of test data based upon the extracted features of test images which are saved in the ‘comtarget’. For the prediction of labels of test data, following Matlab command is used which will give the labels in the variable ‘Ypredict’.

\[ [\text{X11}, \text{Y11}, \text{T11}, \text{AUC11}] = \text{perfcurve} (\text{comtarget}, \text{Y Nb}, 2) ; \]

Based upon the predicted labels ‘Y Nb’, different performance metrics, accuracy sensitivity, specificity, precision, f_measure, goodness mean and area under the curve are evaluated whose values are given in the Table 6.5.

<table>
<thead>
<tr>
<th>Performance Metrics</th>
<th>Naive Bayes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>0.5067</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>0.8400</td>
</tr>
<tr>
<td>Specificity</td>
<td>0.4104</td>
</tr>
<tr>
<td>Precision</td>
<td>0.2917</td>
</tr>
<tr>
<td>F_Measure</td>
<td>0.4330</td>
</tr>
<tr>
<td>GoodnessMean</td>
<td>0.5871</td>
</tr>
<tr>
<td>AUC</td>
<td>0.6252</td>
</tr>
</tbody>
</table>
6.8 Comparative Analysis of Classifiers

For this research, four different classifiers namely decision tree, naïve bayes, support vector machine and k nearest neighbour are used for classifying the apple leaves diseases into marsonina coronaria and apple scab. The results for the different classifier are given in the previous sections. Here comparative analysis of different classifiers has been done based upon the different performance metrics which is shown in the Table 6.6.

Table 6.6 Comparison of Different Classifiers

<table>
<thead>
<tr>
<th>Performance Metrics</th>
<th>Naive Bayes</th>
<th>Decision Tree</th>
<th>Support Vector Machine</th>
<th>K-Nearest Neighbour</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>0.5067</td>
<td>0.8879</td>
<td>0.8879</td>
<td>0.9641</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>0.8400</td>
<td>0.8600</td>
<td>0.5600</td>
<td>0.9000</td>
</tr>
<tr>
<td>Specificity</td>
<td>0.4104</td>
<td>0.8960</td>
<td>0.9827</td>
<td>0.9827</td>
</tr>
<tr>
<td>Precision</td>
<td>0.2917</td>
<td>0.7049</td>
<td>0.9032</td>
<td>0.9375</td>
</tr>
<tr>
<td>F_Measure</td>
<td>0.4330</td>
<td>0.7748</td>
<td>0.6914</td>
<td>0.9184</td>
</tr>
<tr>
<td>Goodness Mean</td>
<td>0.5871</td>
<td>0.8778</td>
<td>0.7418</td>
<td>0.9404</td>
</tr>
<tr>
<td>AUC</td>
<td>0.6252</td>
<td>0.9313</td>
<td>0.9729</td>
<td>0.9946</td>
</tr>
</tbody>
</table>

6.8.1 Accuracy Comparison

For the comparison of accuracy of all the classifiers, a graph is drawn depicting their accuracies as shown in Figure 6.3.
From the Figure it is clear that K nearest neighbour gives best accuracy as compared to other classifiers. Its accuracy is coming out to be 0.9641 which is 4.3% better than SVM, 4.3% better than decision tree and 43.8% better than naïve bayes classifier.

6.8.2 Sensitivity Comparison

For the comparison of accuracy of all the classifiers, a graph is drawn depicting their accuracies as shown in Figure 6.4.

---

**Fig. 6.3 Accuracy Values for Different Classifiers**

**Fig. 6.4 Sensitivity Values for Different Classifiers**
From the Figure it is clear that sensitivity is best for K nearest neighbour as compared to other classifiers. Its sensitivity is coming out to be 0.900 which is 37.7% better than SVM, 4.4% better than decision tree and 6.6% better than naïve bayes classifier.

**6.8.3 Specificity Comparison**

For the comparison of accuracy of all the classifiers, a graph is drawn depicting their accuracies as shown in Figure 6.5.

![Graph showing Specificity Values for Different Classifiers](image)

**Fig. 6.5 Specificity Values for Different Classifiers**

From the Figure it is clear that specificity is best for K nearest neighbour as compared to other classifiers. Its specificity is coming out to be 0.9827 which is same as SVM, but 8.8% better than decision tree and 58.2% better than naïve bayes classifier.

**6.8.4 Precision Comparison**

For the comparison of accuracy of all the classifiers, a graph is drawn depicting their accuracies as shown in Figure 6.6.
From the Figure it is clear that K nearest neighbour gives best precision as compared to other classifiers. Its precision is coming out to be 0.9375 which is 3.6% better than SVM, 24.8% better than decision tree and 68.8% better than naïve bayes classifier.

6.8.5 F_Measure Comparison

For the comparison of accuracy of all the classifiers, a graph is drawn is drawn depicting their accuracies as shown in Figure 6.7.
From the Figure it is clear that f_measure is best for K nearest neighbour as compared to other classifiers. Its f_measure is coming out to be 0.9184 which is 24.7% better than SVM, 15.63% better than decision tree and 52.8% better than naïve bayes classifier.

6.8.6 Goodness Mean Comparison
For the comparison of accuracy of all the classifiers, a graph is drawn depicting their accuracies as shown in Figure 6.8.

![G_Mean Values for Different Classifiers](image)

Fig. 6.8 G_Mean Values for Different Classifiers

From the Figure it is clear that goodness mean is best for K nearest neighbour as compared to other classifiers. Its goodness mean is coming out to be 0.9404 which is 21.1% better than SVM, 6.65% better than decision tree and 37.5% better than naïve bayes classifier.

6.8.7 Area Under Curve Comparison
For the comparison of accuracy of all the classifiers, a graph is drawn depicting their accuracies as shown in Figure 6.9.
Fig. 6.9 AUC Results of Four Classifiers

From the Figure it is clear that K nearest neighbour gives best area under the curve as compared to other classifiers. Its area under the curve is coming out to be 0.9946 which is 2.1% better than SVM, 6.36% better than decision tree and 37.4% better than naïve bayes classifier.

From the observation of the four classifiers, it was found that K nearest neighbour outperforms as compared to other three classifiers in. For this all the performance metrics like accuracy, sensitivity, specificity, precision f_measure, goodness mean and area under the curve comes better than other classifiers.

6.9 Graphical User Interface (GUI)

Figures 6.10 to 6.13 show the graphical user interface results snapshots for Marsonina Coronaria and Apple Scab Disease in Apple leaves for three different backgrounds images. In the snapshots of GUI, input query image, enhanced image, segmented ROI, features values, AUC curve and the type of disease detected are shown.
1) **Type of Disease:** Marsonina Coronaria having Live Background  
**Result:** Pass

![Fig. 6.10 Classification Result of Marsonina Coronaria Diseased Leaf](image)

2) **Type of Disease:** Marsonina Coronaria having black Background  
**Result:** Pass

![Fig. 6.11 Classification Result of Marsonina Coronaria Diseased Leaf](image)
3) **Type of Disease:** Apple Scab with Live Background  
**Result:** Pass

![Fig. 6.12 Classification Result of Apple Scab Diseased Leaf](image)

4) **Type of Disease:** Apple Scab with Live Background  
**Result:** Pass

![Fig. 6.13 Classification Result of Apple Scab Diseased Leaf](image)