Differential Diagnosis of Erythemato-Squamous Diseases

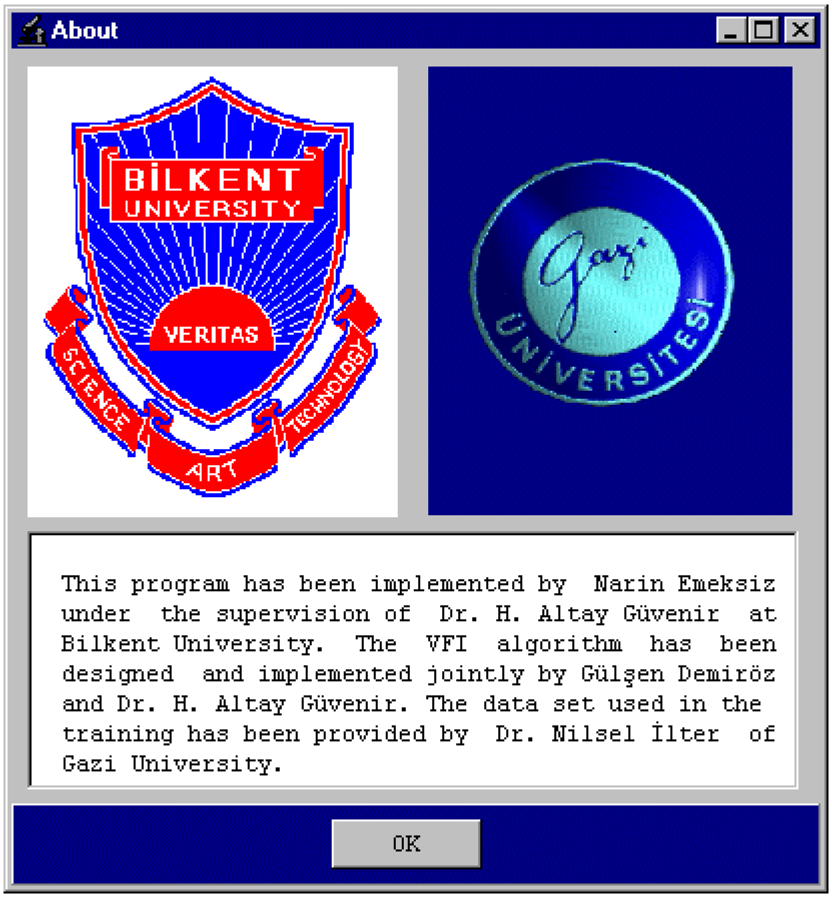
Machine Learning Application

Student: Narin Emeksiz
Supervisor: H. Altay Güvenir

Department of Computer Engineering and Information Science

Bilkent University

May,1998



Abstract

This report is about the implementation of a visual tool for Differential Diagnosis of Erythemato-Squamous Diseases based on the classification algorithms; Nearest Neighbor Classifier (NN), Naive Bayesian Classifier using Normal Distribution (NBCN) and Voting Feature Intervals-5 (VFI5). This tool enables the doctors to perform all the necessary operations occurring in the dermatology department of a hospital.

TABLE OF CONTENTS

1 INTRODUCTION

The major aim of the project is to implement a visual tool for Differential Diagnosis of Erythemato-Squamous Diseases based on the 3 different classification algorithms; Nearest Neighbor Classification on Feature Projections (NN), Naive Bayesian Classifier using Normal Distribution (NBCN) and Voting Feature Intervals-5 (VFI5).

The project has been developed under supervision of Assoc. Prof. Halil Altay Guvenir from Department of Computer Science, Bilkent University. Design goals and the targets of the projects were identified by consulting to Prof. Nilsel Ilter from Department of Dermatology, School of Medicine, Gazi University.

2 PROBLEM DESCRIPTION

2.1 What is Differential Diagnosis?

The differential diagnosis of erythemato-squamous diseases is a real problem in dermatology. They all share the clinical features of erythema and scaling, with very little differences. The diseases in this group are psoriasis (C1), seboreic dermatitis (C2), lichen planus (C3), pityriasis rosea (C4), cronic dermatitis (C5) and pityriasis rubra pilaris (C6).

These diseases are frequently seen in the outpatient departments of dermatology. At the first sight all of the diseases look very much alike with the erythema and scaling. When inspected more carefully some patients have the typical clinical features of the disease at the predilection sites (localization of the skin where a disease preters) while another group has a typical localization.

Patients were first evaluated clinically with 12 features. The degree of erythema and scaling, whether the borders of lesions are definite or not, the presence of itching and koebner phenomenon, the form of the papules, whether the oral mucosa, elbows, knees and the scalp are involved or not, whether there is a family history or not are important for the differential diagnosis.

For example the erythema and scaling of chronic dermatitis is less than of psoriasis, the koebner phenomenon is present only in psoriasis, lichen planus and pityriasis rosea. Itching and polygonal papules are for lichen planus and follicular papules are for pityriasis rubra pilaris. Oral mucosa is predilection site for lichen planus while knee, elbow and scalp involvements are of psoriasis. Family history is usually present for psoriasis and pityriasis rubra pilaris usually starts during childhood.

Some patients can be diagnosed with these clinical features only, but usually a biopsy is necessary for the correct and definite diagnosis. Skin samples were taken for the evaluation of 22 histopathological features. Another difficulty for the differential diagnosis is that a disease may show the histopathological features of another disease at the beginning stage and may have the characteristic features at the following stages. Some samples show the typical histopathological features of the disease while some do not.

Melanin incontinence is a diagnostic feature for lichen planus, fibrosis of the papillary dermis is for chronic dermatitis, exocytosis may be seen in lichen planus, pityriasis rosea and seboreic dermatitis. Acanthosis and parakeratosis can be seen in all the diseases in different degrees. Clubbing of the rete ridges, thinning of the suprapapillary epidermis are diagnostic for psoriasis. Disappearance of the granular layer, vacuolization and damage of basal layer, saw-tooth appearance of retes and a band like infiltrate are diagnostic for lichen planus. Follicular horn plug and perifollicular parakeratosis are hints for pityriasis rubra pilaris.

The features of a patient are represented as a vector of features which has 34 entries for each feature value. In the dataset, the family history feature has the value 1 if any of these diseases has been observed in the family, and 0 otherwise. The age feature simply represents the age of the patient. Every other feature (clinical and histopathological) was given a degree in the range of 0 to 3. Here, 0 indicates that the feature was not present, 3 indicates the largest amount possible, and 1, 2 indicate the relative intermediate values. Each feature has either nominal (discrete) or linear (continuous) values having different weights showing the relevance to the diagnosis.

3 SOLUTION ALGORITHMS

3.1 The Nearest Neighbor Classifier Algorithm

One of the classification algorithms that we used in this project is the NN classifier as it is both a simple and a common algorithm. The NN classification is based on the assumption that examples which are closer in the instance space are of the sample class.

NN algorithm assumes that a new testing variable ought to belong to the same class as its nearest neighbor among all stored training instances. In this project our aim is to classify a single test instance depending on the previously established training dataset. Due to this aim, we did not include the training phase to the project, the methodology that we use is directly inserting the output data into arrays after performing the training process in a separate medium. So, for the implementation of the NN classification algorithm we directly store the train data features and class values in two separate arrays as these are the datasets produced after the training process. Currently, the dataset for the domain contains 366 instances. We first used all of these instances to obtain a description of the domain. The structures of the arrays are shown in Figure 1.

```
int  train_value[366][34]={
 {2,2,0,3,0,0,0,0,1,0,0,0,0,0,0,3,2,0,0,0,0,0,0,0,0,0,0,3,0,0,0,1,0,55},
 {3,3,3,2,1,0,0,0,1,1,1,0,0,1,0,1,2,0,2,2,2,2,2,1,0,0,0,0,0,0,0,1,0, 8} ,
......................
 {2,1,3,1,2,3,0,2,0,0,0,2,0,0,0,3,2,0,0,0,0,0,0,0,3,0,2,0,1,0,0,2,3,50},
 {3,2,2,0,0,0,0,0,3,3,0,0,0,1,0,0,2,0,2,3,2,3,0,2,0,2,0,0,0,0,0,3,0,35},
};
int train_class[366]=
{2,1,3,1,3,2,5,3,4,4,1,2,2,1,3,4,2,1,3,5,6,2,5,3,5,1,6,5,2,3,
 1,2,1,1,4,2,3,2,3,1,2,4,1,2,5,3,4,6,2,3,3,4,1,1,5,1,2,3,4,2,
  ......................
 1,5,5,3,1,5,5,6,6,4,4,6,6,6,1,1,1,5,5,1,1,1,1,2, 2,4,4,3,3,1};
```

Figure 1

All the feature values are assumed to have linear values. The distance metrics used to obtain the distance between two instances in the NN classification algorithm is the Euclidean distance metric. Suppose x is the instance that would be classified and y is an instance that is already in the dataset. The vector representation of **x** and **y** is:

$\mathbf{x} = \langle x_1, x_2, x_3, \ldots, x_n \rangle$ and $\mathbf{y} = \langle y_1, y_2, y_3, \ldots, y_n \rangle$ on an n dimensional space. The distance is computed by *euclidean_distance* function based on the equation;

$$\mathrm{dist}(x,y) = \Sigma^{n}_{f=1}\, w_f * \mathrm{diff}\,(f,x,y)^2$$

$$\mathrm{diff}\,(f,x,y): \quad \begin{array}{ll} |x_f - y_f| & \text{if } f \text{ is linear} \\ 0 & \text{if } f \text{ is nominal and } x_f = y_f \\ 1 & \text{if } f \text{ is nominal and } x_f \neq y_f \end{array}$$

The function that calculates the euclidean_distance is implemented in the project as follows:

```
feature_distance (Instance, Feature)
begin
        if  the instance feature value is not known
          return(1.0)
        if  train_value[Instance][feature] > test[feature])
          return((train_value[Instance][feature] - test[feature]) / range[feature])
        else /*test feature value is bigger*/
          return((test[feature] - train_value[Instance][feature]) / range[feature]);
end

euclidean_distance (Instance)
/* returns the square of euclidean_distance between
   training instance I and the test instance */
begin
        for each feature f
           sum += weight[f] * sqr(feature_distance (I,f));
         return(sum);
end
```

The NN algorithm is more effective when the features of the domain are equally important. It will be less effective when many of the features are misleading or irrelevant to classification. To avoid this, the features are given weights such that the irrelevant features have lower weights (w_f) and the strongly relevant features are given higher weights (w_f). Giving different weights to each feature modify the importance of the feature in the classification process such that a relevant feature becomes more important than a less relevant one.

We had used the outputs of a genetic algorithm for learning the feature weights to be used with the Nearest Neighbor classification algorithm. We applied the same genetic algorithm to determine the weights of the features in our domain to be used with the VFI5 algorithm. The weights of the 34 features, as determined by the genetic algorithm, are shown below. According to the table, koebner phenomenon has the highest weight 0.0620. Inflammatory mononuclear infiltrate is also important in the classification, with the weight of 0.0527. On the other hand, the features acanthosis, follicular horn plug, munro microabcess, and age are found to be the least relevant.

3.2 Naive Bayesian Classifier Using Normal Distribution

Bayesian classifier is an algorithm that approaches the classification problem using probabilities of the features. The probability of the instance belonging to a single class is calculated by using the prior probabilities of classes and the feature values for an instance.

Naive Bayesian Classifier assumes that features are independent. In NBC, each feature participates in the classification by assigning probability values for each class, and the final probability of a class is the product of each single feature probabilities; and the probability of the instance belonging to a class (P(x|Ci)) can be computed as follows:

$$P(x \mid C_i) = \prod_{j=1}^{n} P(x_f \mid C_i)$$

NBC estimates the conditional probability density function $P(x_f \mid C_i)$ for a given feature value x_f for the f^{th} feature using the frequency of observed instances around x_f. $P(x_f \mid C_i)$

for the nominal features is the ratio of the number of training examples of class C_i with value x_f for feature f over total number of training examples of class C_i.

$P(x_f | C_i)$ for continuous features is computed using the normal distribution. The normal distribution function is as follows: $p(x_f) = (1/\sqrt{2\pi\sigma^2})e^{-(x_f - \mu)2/2\sigma2}$

In this project our aim is to classify a single test instance depending on the previously established training dataset. In order to perform this aim we did not include the training phase of the NBCN Algorithm to the project, we directly fill in the arrays after performing the training process in a separate medium. The main components of the normal density function are the variance σ^2 and the mean μ. So, for the implementation of the NBCN classification algorithm we store the variance and the mean of the linear values in two arrays called `Variance[34]` and `Mean[34]` arrays.

The NBCN algorithm handles the missing feature values by ignoring the feature with the missing value instead of ignoring the whole instance. When x has unknown value for f, the conditional probabilities $P(x_f | C_i)$ of each class C_i is assigned to 1, which has no effect on the product of probabilities distributed by each feature. The NBCN classification algorithm is shown in Figure2.

```
probability (example x, feature f, class c)
begin
  if  feature is nominal and  there is not any value belonging to the feature and class
     return(0.0)
  if  the feature is nominal
    for each distinct value belonging to the feature for the same class
          density = distribution[c][f][i].count
 else the feature is nominal
   begin
     /*Apply normal distribution*/
     distToMean = x - Mean[c][f];
     temp = distToMean * distToMean / (2 * Variance[c][f]);
     density = exp(- temp) / sqrt(2*PI*Variance[c][f]);
    end
  Pclass_feature[f][c] = density;
  return(density*10);
end

nbcn ()
begin
  g[0]=0
  /*initial value of the class probabilities*/
  for each class
    begin
     g[c] = classProbability[c];
     for each feature value
          if test feature value is known
                 g[c] = g[c] * probability (test[f], f, c)
     /* find c with max g[c]*/
     if (g[c] > g[prediction]) prediction = c
    end
  return(prediction)
end
```

Figure 2

3.3 Voting Feature Intervals-5 Algorithm

The VFI5 classification algorithm represents a concept description by a set of feature intervals. The classification of a new instance is based on a voting among the classifications made by the value of each feature separately. It is a non-incremental classification algorithm; that is, all training examples are processed at once.

From the training examples, the VFI5 algorithm constructs intervals for each feature. An interval is either a range or point interval. A range interval is defined on a set of consecutive values of a given feature whereas a point interval is defined for a single feature value. For point intervals, only a single value is used to define that interval. For range intervals, on the other hand, it suffices to maintain only the lower bound for the range of values, since all range intervals on a feature dimension are linearly ordered. The lower bound of the range intervals obtained from the training instances are installed into an array called SegmentLower and the number of segments formed for each feature value is stored in the array No_Segments directly at the beginning of the vfi function so no training process is performed. The structure of the arrays are shown in Figure 3:

```
int No_Segments[34] = {7, 9, 9, 7, 7, 5, 9, 5, 9, 9, 5, 5, 7, 7, 9, 7, 9, 7, 7, 9,
                        9, 7, 9, 7, 7, 7, 7, 5, 7, 7, 7, 5, 9, 21};
int SegmentLower[34][22] =
{
  {-10, 0, 0, 1, 1, 3, 3},
       {-10, 0, 0, 1, 1, 2, 2, 3, 3},
       {-10, 0, 0, 1, 1, 2, 2, 3, 3},
        ..........................
   {-10, 0, 0, 1, 1, 2, 2, 3, 3},
       {-10, 0, 0, 7, 7, 8, 8, 10, 10, 12, 12, 16, 16, 22, 22, 65, 65, 70, 70, 75, 75}
};
```

Figure 3

For each interval, a single value and the votes of each class in that interval are maintained. Thus, an interval may represent several classes by storing the vote for each class. The votes given to the classes for each interval for each feature values are stored in the SegmentVotes array.

```
float SegmentVotes[34][22][7] = {
               {{0, 0, 0, 0, 0, 0, 0},
                {0, 0.145704, 0, 0.22665, 0, 0.627646, 0},
                {0, 0, 0, 0, 0, 0, 0},
                {0, 0.0596973, 0.0782916, 0.0928625, 0.214423, 0.45921, 0.0955157},
                {0, 0.155105, 0.153693, 0.187658, 0.180077, 0.11666, 0.206807},
                {0, 0.289338, 0.285058, 0.164664, 0.0967822, 0.0455993, 0.118558},
                {0, 0, 0, 0, 0, 0, 0}},
               {{0, 0, 0, 0, 0, 0, 0},

                                       }
```

The training phase is performed in another platform and the operations take place in the training process in the VFI5 algorithm is to find the end points for each class 'c' on each feature dimension 'f'. End points of a given class 'c' are the lowest and highest values on a linear feature dimension 'f' at which some instances of class 'c' are observed. On the other hand, end points on a nominal feature dimension 'f' of a given class 'c' are all distinct values of 'f' at which some instances of class 'c' are observed. There are 2k end points for each linear feature, where k is the number of classes. Then, for linear features the list of end-points on each feature dimension is sorted. If the feature is a linear feature, then point intervals from each distinct end point and range intervals between a pair of distinct end points excluding the end points are constructed. If the feature is a nominal feature, each distinct end point constitutes a point interval.

The number of training instances in each interval is counted. These counts for each class 'c' in each interval 'i' on feature dimension 'f' are computed. For each training example, the interval 'i' in which the value for feature 'f' of that training example 'e' falls is searched. If interval i is a point interval and e_f is equal to the lower bound (same as the upper bound for a point interval), the count of the class of that instance in interval i is incremented by 1. If interval i is a range interval and e_f is equal to the lower bound of i (falls on the lower bound), then the count of class e_c in both interval i and (i-1) are incremented by 0.5. But if e_f falls into interval i instead of falling on the lower bound, the count of class e_c in that interval is incremented by 1 normally. There is no need to consider the upper bounds as another case, because if e_f falls on the upper bound of an interval I, then e_f is the lower bound of interval i+1. Since all the intervals for a nominal feature are point intervals, the effect of *count\instances* is to count the number of instances having a particular value for nominal feature f.

To eliminate the effect of different class distributions, the count of instances of class ‘c’ in interval ‘i’ of feature ‘f’ is then normalized by *class\count[c]*, which is the total number of instances of class ‘c’. As these operations occured in the training phase, they are not included in our program. Only the data set formed after the training phase is directly initialized to the arrays SegmentLower, No_Segments and SegmentVotes.

The classification process starts by initializing the votes of each class to zero. The classification operation includes a separate preclassification step on each feature. The preclassification of feature ‘ f’ involves a search for the interval on feature dimension ‘f’ into which e_f falls, where e_f is the value test example ‘e’ for feature ‘f’. If that value is unknown (missing), that feature does not participate in the classification process. Hence, the features containing missing values are simply ignored. Ignoring the feature about which nothing is known is a very natural and plausible approach.

```
find_segment (value, feature f)
begin
 while ((SegmentLower[f][s]< value) && (s < No_Segments[f]))
           increase s
  if (SegmentLower[f][s] == value)
    return(s);
   else
    return(s-1);
end

feature_votes (int f, float featureVotes[])
 begin
 initalize for prediction
  if test value is known
       s = find_segment(test[f], f);
       for each claass value
               featureVotes[c] = SegmentVotes[f][s][c];
          VotesFeatures[f][c]=featureVotes[c];
end

vfi5 ()
begin
  initalize for prediction the total votes array
  initialize the votes of each feature for each class
  for each feature
    feature_votes(f, featureVotes);
    for each class
           totalVotes[c] += (featureVotes[c]* weight[f]);
   prediction = 0;
   for each class
      check for the class having the largest probability
   return (prediction)
end
```

Figure 4.

If the value for feature ‘f’ of example ‘e’ is known, the interval ‘I’ into which e_f falls is found. That interval may contain training examples of several classes. The classes in an interval are represented by their votes in that interval. For each class ‘c’, feature ‘f’ gives a vote equal to interval\vote[f,~i,~c], which is vote of class c given by interval i on feature dimension ‘f’. If e_f falls on the boundary of two range intervals, then the votes are taken from the point interval constructed at that boundary point. The individual vote of feature ‘f’ for class ‘c’, feature/vote[f,c], is then normalized to have the sum of votes of feature ‘f’ equal to 1. Hence, the vote of feature ‘f’ is a real-valued vote less than or equal to 1. Each feature ‘f’ collects its votes in an individual vote vector VotesFeatures[34][7]. After every feature completes their preclassification process, the individual vote vectors are summed up to get a total vote vector totalVotes[7]. Finally, the class with the highest vote from the total vote vector is predicted to be the class of the test instance. The implementation of the VFI algorithm is in Figure 4:

4. DESIGN OF THE PROJECT

As this application is going to be used by the doctors who are not advanced computer users, we had aimed to implement the user interface of the Erythemato-Squamous Diseases application user friendly. In order to make the usage of the program as a joy instead of a nightmare we choose Borland C++ Builder for Windows 95 & Windows which provides us the easy usage of the visual aids, and provide us to prepare a database.

Borland C++ Builder is an object-oriented, visual programming environment for rapid application development of general purpose client/server applications for Microsoft Windows 95 and Windows NT. C++ Builder enables me to perform complicated applications with a minimum coding. In C++ environment all the tools needed to design, develop, test, and debug applications are available.

At first step, in order to warm up to the C++ Builder environment I practice on the screen designs of the project, and learn the visual programming environment. In the light of advises of Prof. Ilter and Assoc. Prof. Guvenir, I defined the requirements of a dermatology department of a hospital.

Being a department of a hospital, dermatology department inherits all the processes take place in a hospital. Everyday some number of patients are applied to the department as they have symptoms which are the signs of a skin disease. In order to keep track of each patient and prepare history for the hospital, I constructed a database in which the detailed information of each patient would be kept in. The ByopsiNo is selected as the primary key so it is unique for each patient in the database. Also indexes are formed for PatientName, PatientSurname and PatientName and PatientSurname. The structure of the database table consists of the following fields:

Field
ByopsiNo
Patient Name
Patient Surname
Entrance Date
Doctor’s Diagnosis

Also 12 clinical features are stored in the database
Feature1(erythema)
Feature2(scaling)
Feature3(definite borders)
Feature4(itching)
Feature5(koebner phenomenon)
Feature6(polygonal papules)
Feature7(follicular papules)
Feature8(oral mucosal involvement)
Feature9(knee and elbow involvement)
Feature10(scalp involvement)
Feature11(family history)
Feature34(age)

Afterwards, skin samples were taken for the evaluation of 22 histopathological features which is called the biopsy process. The values of the histopathological features are determined by an analysis of the samples under a microscope. These features are:
Feature12(melanin incontinence)
Feature13(eosinophils in the infiltrate)
Feature14(PNL infiltrate)
Feature15(fibrosis of the papillary dermis)
Feature16(exocytosis)
Feature17(acanthosis)
Feature18(hyperkeratosis)
Feature19(parakeratosis)
Feature20(clubbing of the rete ridges)
Feature21(elongation of the rete)
Feature22(thinning of the suprapapillary epidermis)
Feature23(spongiform pustule)
Feature24(munro microabcess)
Feature25(focal hypergranulosis)
Feature26(disappearance of the granular layer)
Feature27(vacuolisation and damage of basal layer)
Feature28(spongiosis)
Feature29(saw-tooth appearance of retes)
Feature30(follicular horn plug)
Feature31(perifollicular parakeratosis)
Feature32(inflammatory monoluclear inflitrate)
Feature33(band-like infiltrate)

In the dataset constructed for this domain, the biopsy no is the label that is given to each patient for the differentiation, name and surname belongs to the patient, the doctor's diagnosis field stores the doctors prediction about the disease and its range is from 1 to 6 each reflecting the label of the 6 eythemato-squamous diseases, family history feature has the value 1 if any of these diseases has been observed in the family, and 0 otherwise. The age feature simply represents the age of the patient. Every other feature (clinical and histopathological) was given a degree in the range of 0 to 3. Here, a 0 indicates that the feature was not present, a 3 indicates the largest amount possible, and 1, 2 indicate the relative intermediate values.

4.1 Database Operations Take Place

Keeping the patient records; entrance of a new patient, searching for an already recorded patient or extracting a patient from the registration are some of the operations that leads to the construction of a database. All these operations are performed by specially prepared forms.

4.1.1 Patient Record Entrance

The Patient Record Entrance Form shown in Figure 5 enables the user to enter all the information about the patient.

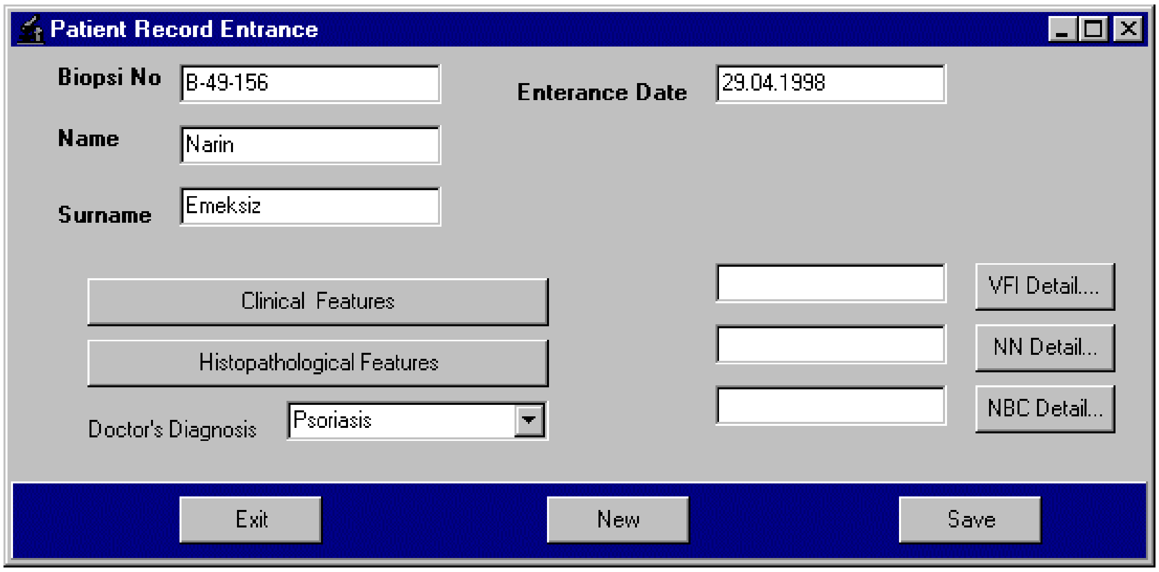


Figure 5.

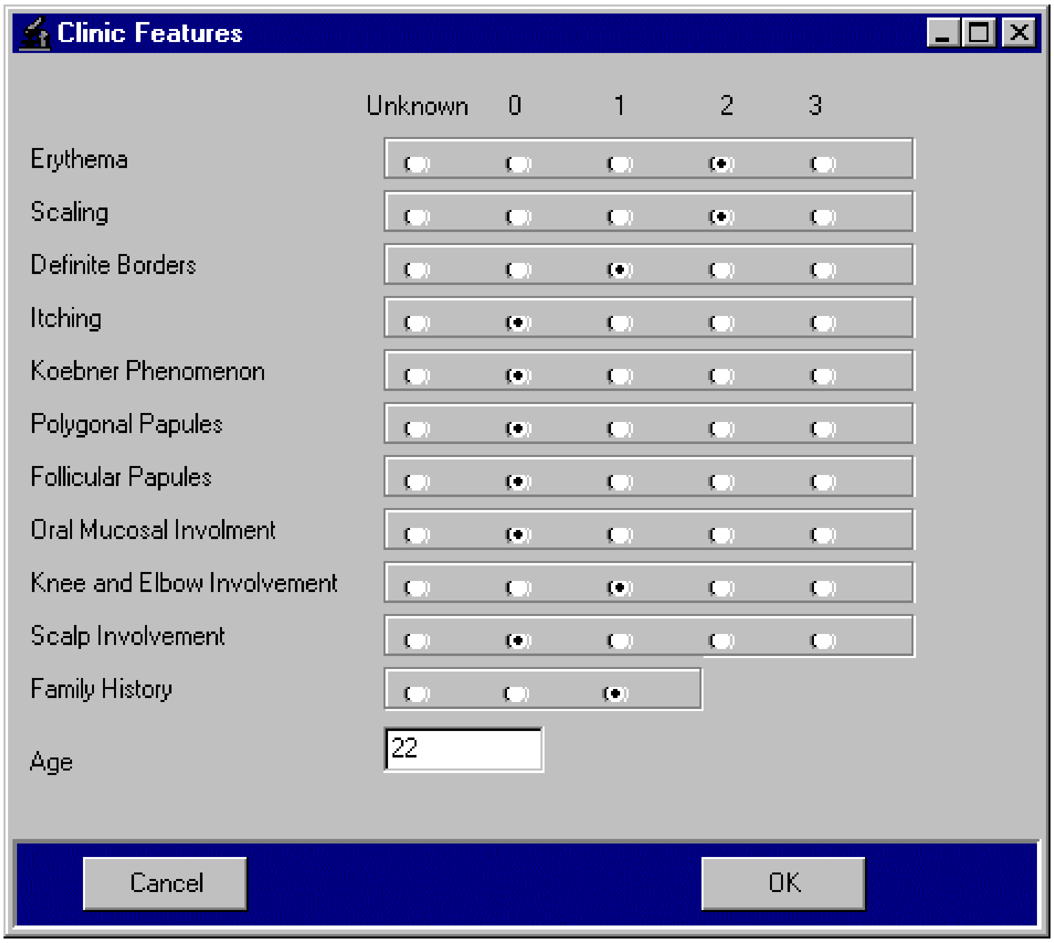


Figure 6.

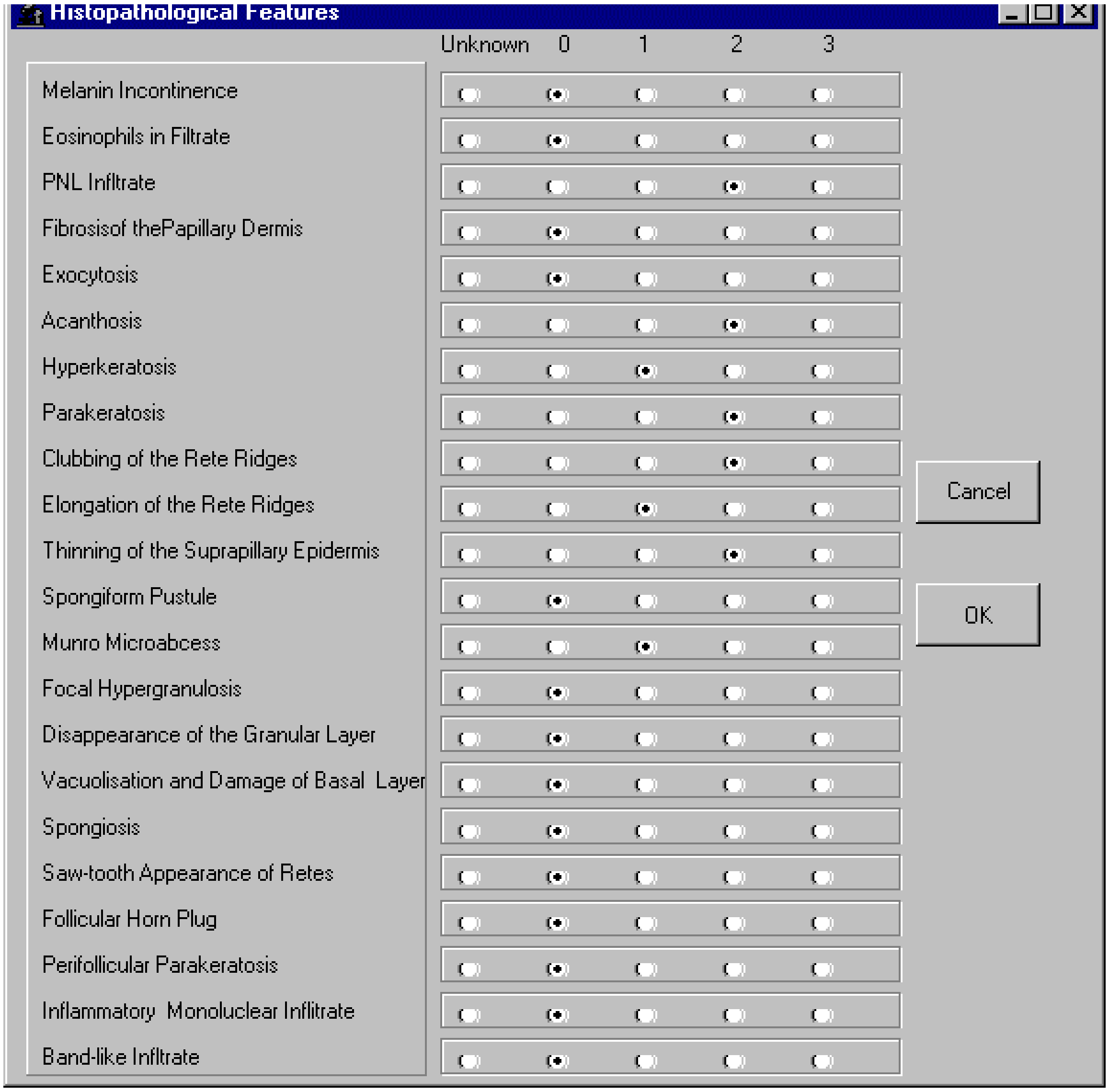


Figure 7.

The save button is enabled only if a new biopsyno is entered. When pressed it attempts to save the entered data, but initially checks the previous records if this biopsyno already exists or not. If it exists it shows a warning messages and forces the user to enter a unique value for the biopsy no. By this method it preserves the primary key property of the biopsyno. If the biopsyno is unique, then the database is prepared for the insertion. If the buttons labeled Clinical Features or Histopathological Features is pressed one of the following forms in Figure 6 or Figure 7 is opened and enables the user to enter the feature values only by marking the corresponding values.

If a value is not entered in these forms their values are recorded as unknown to the database and each prediction algorithm handles these unknowns in a specific way depending on the handling mechanism of the algorithm. Classification algorithms make prediction even if one of the feature values of clinical or histopathological features is entered. The result of one prediction is shown in Figure8.

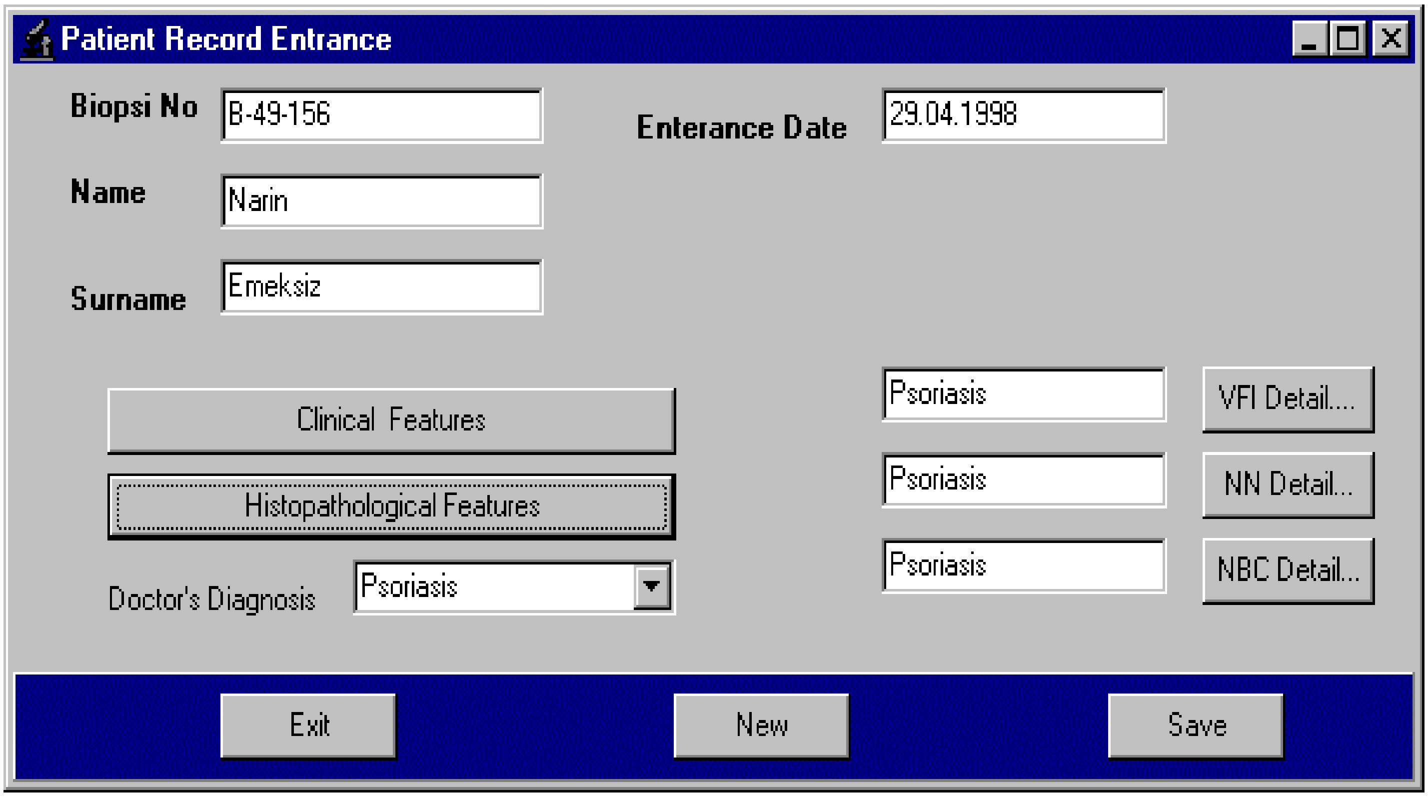


Figure 8.

4.1.2 Patient Record Search

As keeping Biopsyno in mind is a difficult task for a human being, we based our searching methodology on different indexes. The user can reach the target patient by defining its search criteria on form in Figure 9 after choosing the search patients option from the main menu.

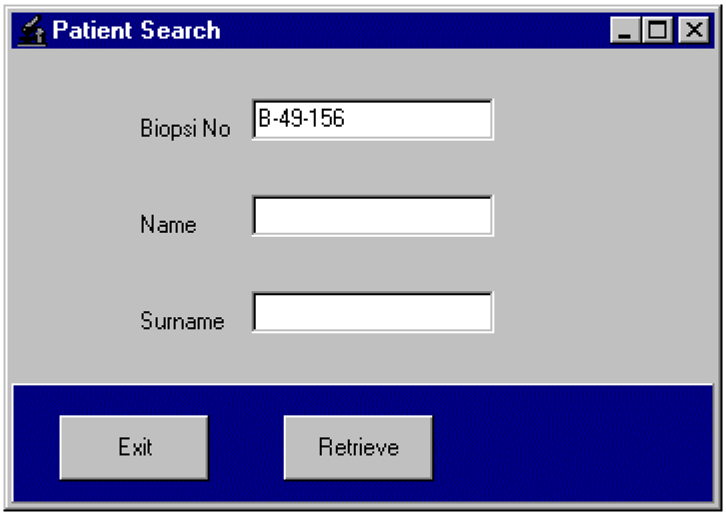


Figure 9.

We have four searching craters;

1). BiopsyNo : If an existing BiopsyNo is entered then the patient having this BiopsyNo is displayed.
2). Name : All the patients in the database having this name is returned.
3). Surname : All the patients in the database having this surname is returned.
4). Name Surname: All the patients in the database having this name and surname is returned.

The arrows appeared on the form enables to see all the retrieved patients depending on the selected search criteria. The database is opened depending on the selected search criteria before each search.

If the detail button is pressed than the form in Figure 10 which contains the detailed information about the patient is retrieved. This form enables us to make any update on the previously recorded dataset; to examine the previous patients details and to see the predictions. The VFI, NN and NBCN Algorithms functions depending on the same methodology as explained in the Patient Record Entrance Form.

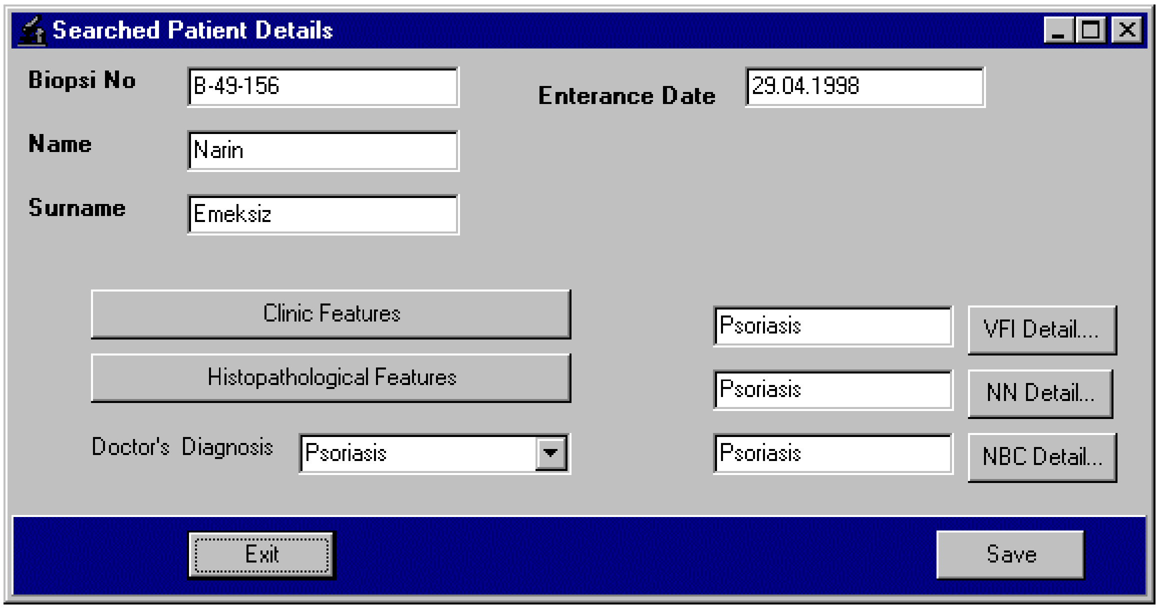


Figure 10.

For the update operation; the biopsy no which is on the form is taken and the database is opened as indexed by the biopsyno. Then the record which would be updated is found by the GotoKey() function. This function is specific for the primary keys and directly goes to the searched key.

4.1.3 Patient Record Deletion

If the data about a patient becomes out of date or a patient is entered by mistaken; the removal of the patient from the database comes to scene. The special treatments done for the removal enables the user to clean the database from the unnecessary data. The methodology to reach the target patient data which would be removed from the database is like the searching process but this time the options of the form displayed, when the detail button pressed, is different. The detail form, which is shown in Figure11, is only for reassuring the user that he is deleting the correct patient. It does not have update option and the algorithms do not do any prediction and all clinical and histopathological forms are disabled against the user interrupt.

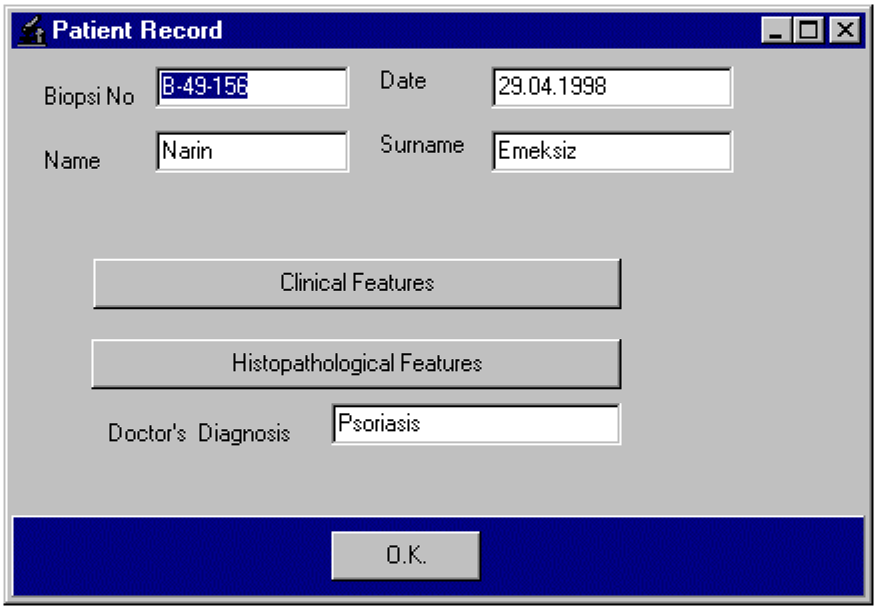
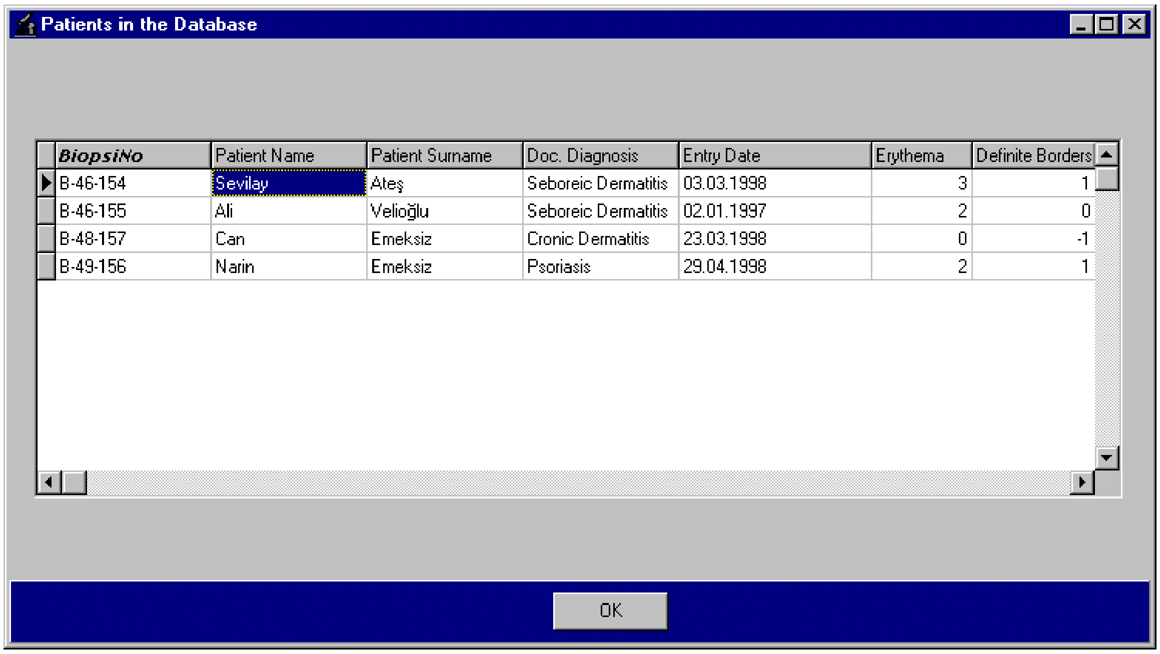


Figure 11

4.2 Patients in the Database

When the database option is selected from the main menu, all the patient details displayed. For this option I used the query functions of the Borland C++ Builder. I wrote an SQL statement to the QueryComponent such as “Select * from Patient”. The result of a selection is displayed in Figure 12.



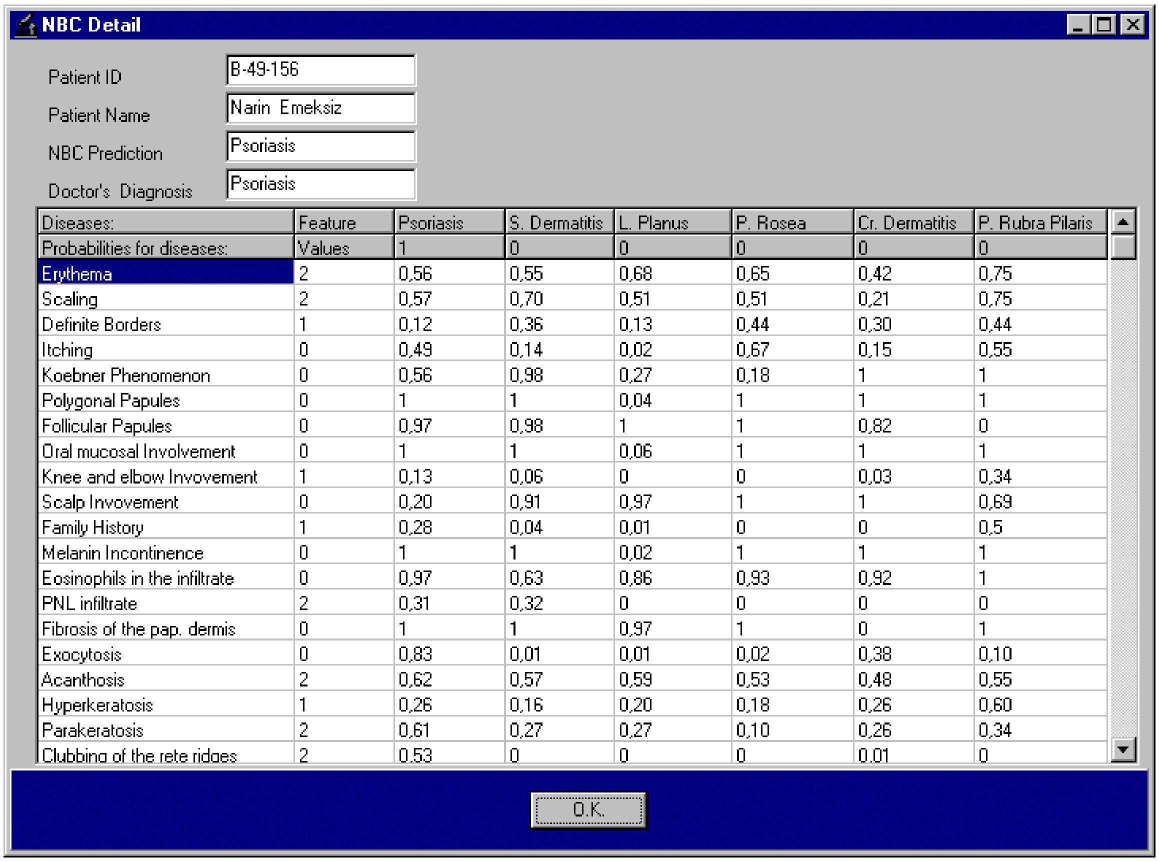
BiopsiNo	Patient Name	Patient Surname	Doc. Diagnosis	Entry Date	Erythema	Definite Borders
B-46-154	Sevilay	Ateş	Seboreic Dermatitis	03.03.1998	3	1
B-46-155	Ali	Velioğlu	Seboreic Dermatitis	02.01.1997	2	0
B-48-157	Can	Emeksiz	Cronic Dermatitis	23.03.1998	0	-1
B-49-156	Narin	Emeksiz	Psoriasis	29.04.1998	2	1

Figure 12.

4.3 Algorithm Displays

As one of the main aims of the project is to be an assistant tool in the training of the dermatology diseases; the implementation of the 3 different classification algorithms are placed in both Patient Data Entrance and Searched Patient Details forms by giving the doctor the chance to compare his own classification with the prediction of the algorithms. The detailed information given for each of the classification algorithms can provide the flexibility to the application to be used both in the hospital and in the education process of the intern-doctors.

If the detail button for the NBCN is pressed then the form which shows the probability of each of 34 features belonging to any erythemato-squamous diseases is displayed. The NBCN Form is shown in Figure13.


NBC Detail

Patient ID: B-49-156
Patient Name: Narin Emeksiz
NBC Prediction: Psoriasis
Doctor's Diagnosis: Psoriasis

Diseases:	Feature	Psoriasis	S. Dermatitis	L. Planus	P. Rosea	Cr. Dermatitis	P. Rubra Pilaris
Probabilities for diseases:	Values	1	0	0	0	0	0
Erythema	2	0,56	0,55	0,68	0,65	0,42	0,75
Scaling	2	0,57	0,70	0,51	0,51	0,21	0,75
Definite Borders	1	0,12	0,36	0,13	0,44	0,30	0,44
Itching	0	0,49	0,14	0,02	0,67	0,15	0,55
Koebner Phenomenon	0	0,56	0,98	0,27	0,18	1	1
Polygonal Papules	0	1	1	0,04	1	1	1
Follicular Papules	0	0,97	0,98	1	1	0,82	0
Oral mucosal Involvement	0	1	1	0,06	1	1	1
Knee and elbow Invovement	1	0,13	0,06	0	0	0,03	0,34
Scalp Invovement	0	0,20	0,91	0,97	1	1	0,69
Family History	1	0,28	0,04	0,01	0	0	0,5
Melanin Incontinence	0	1	1	0,02	1	1	1
Eosinophils in the infiltrate	0	0,97	0,63	0,86	0,93	0,92	1
PNL infiltrate	2	0,31	0,32	0	0	0	0
Fibrosis of the pap. dermis	0	1	1	0,97	1	0	1
Exocytosis	0	0,83	0,01	0,01	0,02	0,38	0,10
Acanthosis	2	0,62	0,57	0,59	0,53	0,48	0,55
Hyperkeratosis	1	0,26	0,16	0,20	0,18	0,26	0,60
Parakeratosis	2	0,61	0,27	0,27	0,10	0,26	0,34
Clubbing of the rete ridges	2	0,53	0	0	0	0,01	0

O.K.

Figure13.

When the detail button is pressed for seeing the logic that lies behind the NN algorithm's prediction the form called NN-Detail which is shown in Figure 14 is displayed. As NN algorithm assumes that a new patient has the same disease as its nearest neighbor; the design of the NN-Detail form includes both the patient for whom the NN makes classification and the patient, which has the most similar feature values.

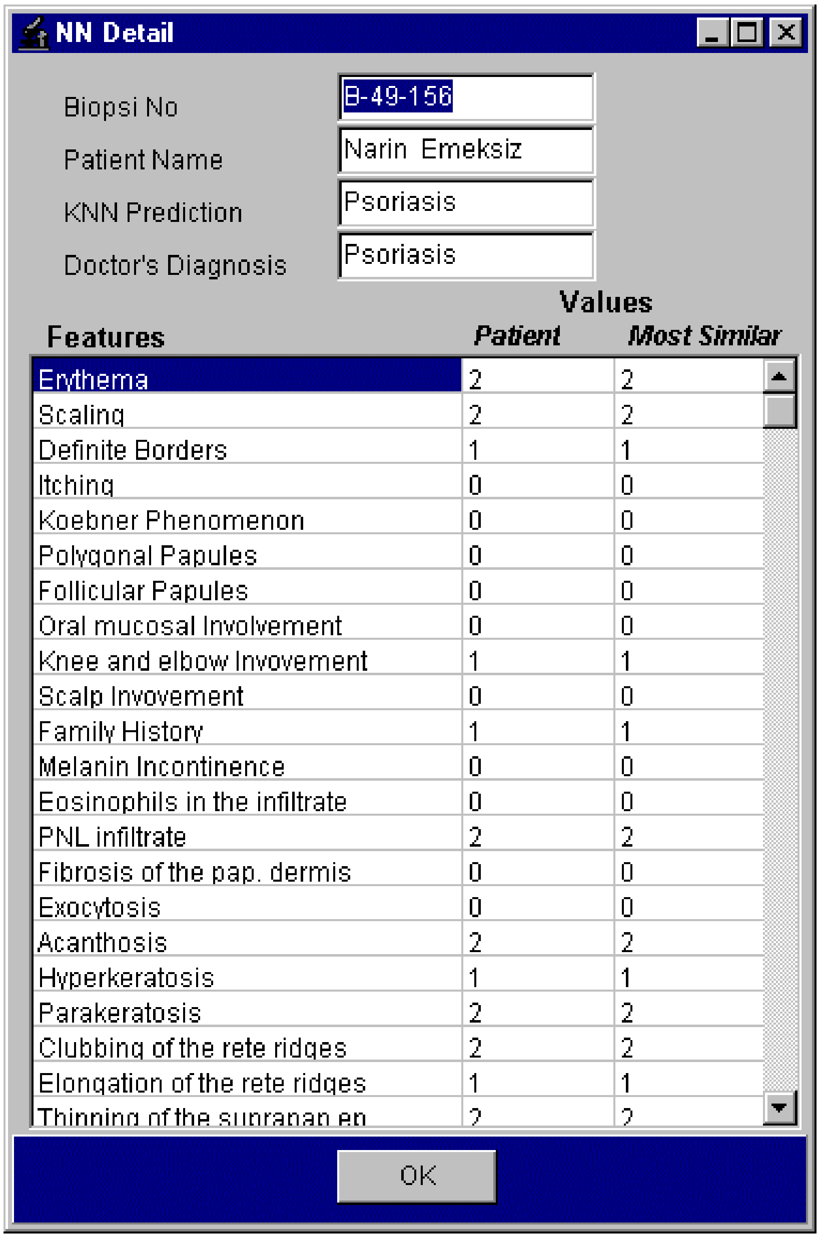
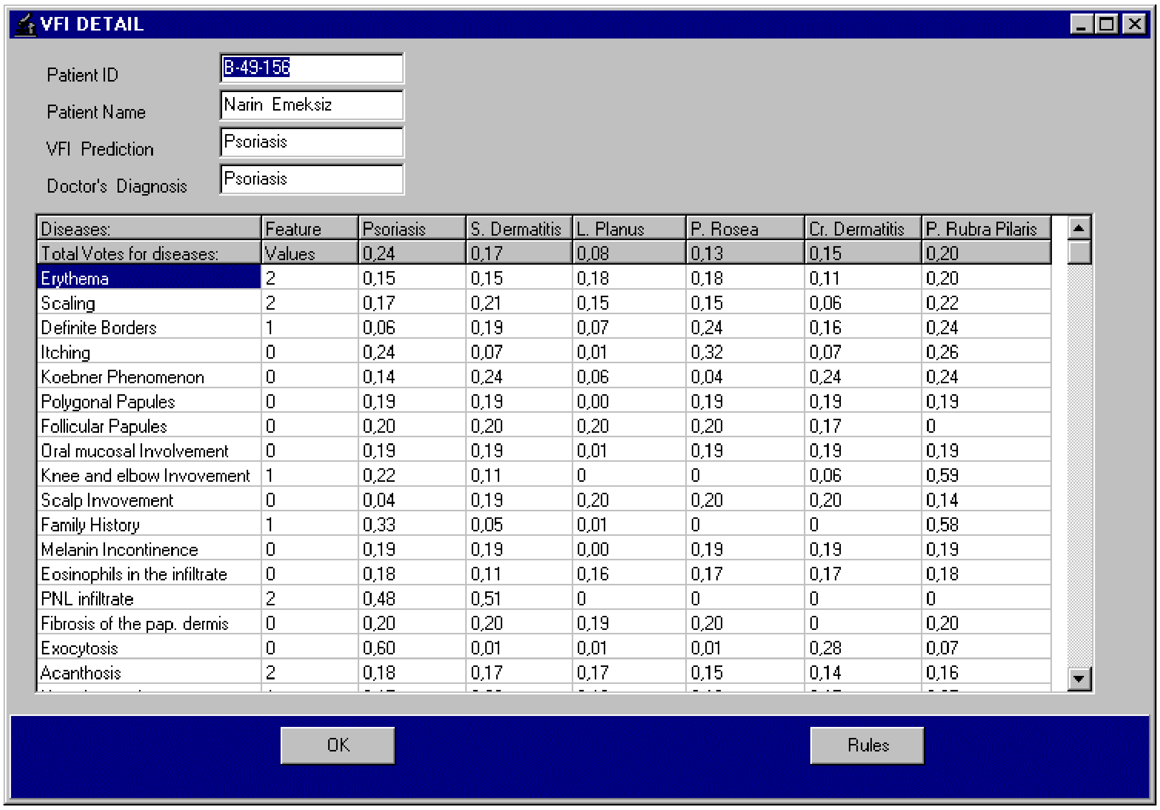


Figure14.

When the detail button is pressed for seeing the logic that lies behind the VFI-5 algorithm's classification the form called VFI-Detail which is shown in Figure 15 is displayed.



VFI DETAIL

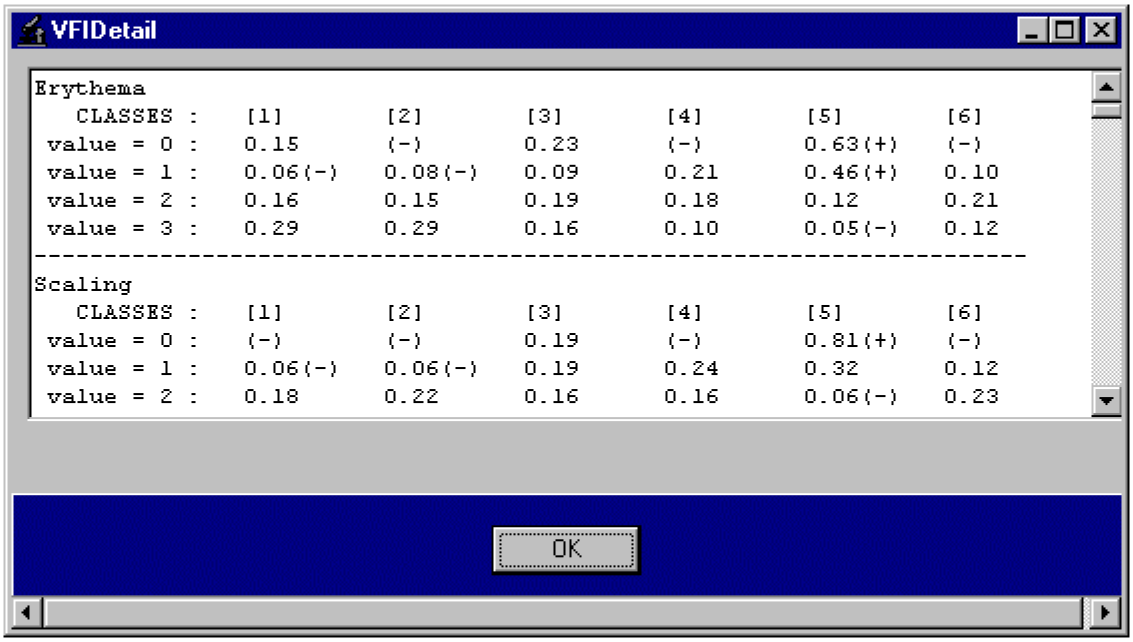
Patient ID: B-49-156
Patient Name: Narin Emeksiz
VFI Prediction: Psoriasis
Doctor's Diagnosis: Psoriasis

Diseases:	Feature	Psoriasis	S. Dermatitis	L. Planus	P. Rosea	Cr. Dermatitis	P. Rubra Pilaris
Total Votes for diseases:	Values	0,24	0,17	0,08	0,13	0,15	0,20
Erythema	2	0,15	0,15	0,18	0,18	0,11	0,20
Scaling	2	0,17	0,21	0,15	0,15	0,06	0,22
Definite Borders	1	0,06	0,19	0,07	0,24	0,16	0,24
Itching	0	0,24	0,07	0,01	0,32	0,07	0,26
Koebner Phenomenon	0	0,14	0,24	0,06	0,04	0,24	0,24
Polygonal Papules	0	0,19	0,19	0,00	0,19	0,19	0,19
Follicular Papules	0	0,20	0,20	0,20	0,20	0,17	0
Oral mucosal Involvement	0	0,19	0,19	0,01	0,19	0,19	0,19
Knee and elbow Invovement	1	0,22	0,11	0	0	0,06	0,59
Scalp Invovement	0	0,04	0,19	0,20	0,20	0,20	0,14
Family History	1	0,33	0,05	0,01	0	0	0,58
Melanin Incontinence	0	0,19	0,19	0,00	0,19	0,19	0,19
Eosinophils in the infiltrate	0	0,18	0,11	0,16	0,17	0,17	0,18
PNL infiltrate	2	0,48	0,51	0	0	0	0
Fibrosis of the pap. dermis	0	0,20	0,20	0,19	0,20	0	0,20
Exocytosis	0	0,60	0,01	0,01	0,01	0,28	0,07
Acanthosis	2	0,18	0,17	0,17	0,15	0,14	0,16

OK   Rules

Figure 15.

The rules table in Figure 16 displays the logic that lies behind the votes given to each class for each of the 34 features.



VFIDetail

```
Erythema
   CLASSES :   [1]       [2]       [3]       [4]       [5]       [6]
 value = 0 :   0.15      (-)       0.23      (-)       0.63(+)   (-)
 value = 1 :   0.06(-)   0.08(-)   0.09      0.21      0.46(+)   0.10
 value = 2 :   0.16      0.15      0.19      0.18      0.12      0.21
 value = 3 :   0.29      0.29      0.16      0.10      0.05(-)   0.12
-----------------------------------------------------------------------
Scaling
   CLASSES :   [1]       [2]       [3]       [4]       [5]       [6]
 value = 0 :   (-)       (-)       0.19      (-)       0.81(+)   (-)
 value = 1 :   0.06(-)   0.06(-)   0.19      0.24      0.32      0.12
 value = 2 :   0.18      0.22      0.16      0.16      0.06(-)   0.23
```

OK

Figure 16.

4.4 Help

As this program would be used by the doctors who are not much familiar with the classification algorithms; we include a brief description for each of the algorithms. In the below the description which is written for NN Classifier is given (Figure 17).

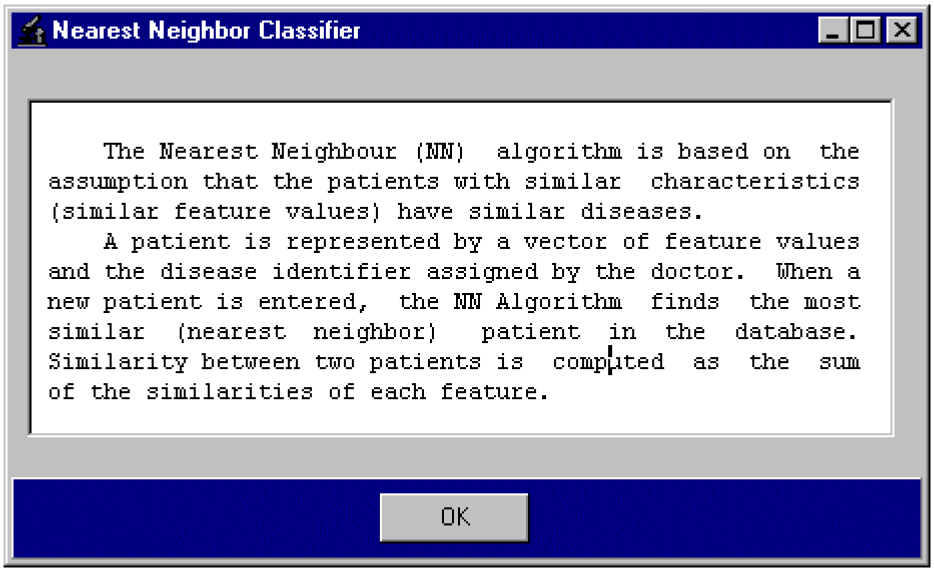


Figure 17.

5 CONCLUSION

In our opinion using this tool in the education process provides a more colorful environment for the doctors than huge hard covered materials. Also the students of the Gazi Medical School can use the tool for testing their knowledge by comparing their predictions with the classifications done by the algorithms. Also another advantage of the tool is to be a guide to the doctors in constructing their own classification mechanisms by examining the working methodologies of the algorithms presented in the detail sections.

In this project I worked on classification algorithms which were written for UNIX. As the original version of the algorithm was implemented for both the training and testing processes. Adapting it for the classification of a single patient and adding the functions, which would provide the application's easy usage, was the main theme of the project.

Today, this visual tool for the differentiation of erythemato squamous diseases is ready for the usage by the doctors and the computer scientists who are interested in machine learning algorithms.

I would like to express my gratitude to Assoc. Prof. H.Altay Guvenir, for his endurance, help in the implementation, suggestions, solutions, understanding thorough the implementation of the project and for everything.

REFERENCES

[1] G. Demiroz. Non-Incremental Classification Learning Algorithms Based On Voting Feature Intervals. Bilkent University, Dept. Of Computer Engineering and Information Science, Msc. Thesis, 1997.

[2] G. Demiroz, and H.A. Guvenir, and Nilsel Ilter. Differential Diagnosis of Erythemato-Squamous Diseases Using Feature Intervals. In Prooceedings of the Sixth Turkish Symposium on Artificial Intelligence and Neural Networks (TAINN'97), 190-194, 1997.

[3] H.A. Guvenir and I.Sirin, Classification by Feature Partitioning, Machine Learning, 23:47-67, 1996.