A Simple Feature Space Partitioning Approach for Pattern Classification

Hina Shah, Suman K. Mitra, Asim Banerjee
Dhirubhai Ambani Institute of Information and Communication Technology,
Gandhinagar, Gujarat
e-mail: (hina_r_shah, suman_mitra, asim_banerjee)[at]daiict[dot]ac[dot]in

***Abstract*—Classification methods in general use a linear or piecewise linear boundary to define the separation between the classes in the feature space. This results into some error when the classes are not linearly separable from each other. In fact when one class is surrounded by another class, things become more complicated. Instead, if a non-linear boundary between the classes is created or areas where the class lie are marked, then classification becomes an easier task. The paper here deals with an approach of feature space partitioning wherein the feature space is partitioned continuously till some predefined resolution. At the end of the procedure, an area which a particular class occupies is defined and these areas are further utilized for assigning incoming data points to respective classes.**

I. Introduction

Classification is the primary aim of any information system. There have been supervised and unsupervised classification methods extensively generated and studied. The supervised classification methods follow a learning approach, in which first system learns, from existing training data, the separating boundaries between the classes in the feature space, and later a test data set is used for checking the correctness of the classification method. While unsupervised method requires human interpretation of the outcome. The supervised classification methods generally tend to find linear optimum separators between the classes for a minimum possible error rate of classification. However, it would be much more desirable to have non-linear separators between the classes for a better class separation representation.

The paper here presents a supervised method of classification, in which as part of learning process of the training data set, a continuous partitioning of the feature space is performed. Hyperboxes are created in the feature space at a lower resolution and these are continuously divided till a pure partition is found. Purity of a hyperbox depends on the number of classes in the hyperbox. Lesser the number of classes in the hyperbox, purer the hyperbox is.

Feature space partitioning has not been a new concept. Methods explored in [2] and [3] try to use the classical methods for classification in a sub-region of the feature space, so that classification complexity decreases. Hence feature space partitioning is used to find such subspaces in the feature space, where the classification procedure becomes extremely simple. While [2] uses genetic algorithms to perform the classification, while [3] tries to find boundaries between the two classes that are present in the feature space. The experimentation and the development of method in [3] is done for 2 classes, however, the method can be improved to work on several classes. A fuzzy classification method by partitioning the feature space into maybe overlapping hyperboxes and then generate some if-then rules for classification has been discussed in [4]. The paper suggests that first the feature space be paritioned into areas which are homogenous and non-homogenous. Homogenous areas are left as it is, while non-homogenous areas are the ones which again would be partitioned till some suitable condition is met. The paper also tries to handle some imprecise data, which might not be handled if classical methods of classification are used.

It becomes important to have a complete feature set to describe the data fully and the work presnted here takes into assumption that the feature set that has been selected is complete and so is the training set. However methods can be found to handle a case where a feature value is missing, in which case, a fuzzy assignment of the incoming input data can be done [4].

Rest of the paper is arranged as follows: Section II gives some mathematical preliminaries, Section III contains description of the method of feature space partitioning for classification, Section IV discusses experimental results, and Section V gives conclusion and scope for future work

II. Mathematical Preliminaries

Consider the feature space to be consisting of d features. Each training data point is represented as $x_i = (x_{i1}, x_{i2}, \ldots, x_{id})$ where $1 \leq i \leq N$, N is the total number of data points given in the feature space. Let C be the total number of classes in the feature space for which the classification is to be done.

- Hyperbox: Partitioning of the feature space has been experimented with several topologies [5]. However, a regular hypercube/hypercuboid division is much more easier to use and is much more representable and usable, rather than the other topologies. A generic term of hyperbox is used here for hypercuboid and hypercube. A hyperbox in d-dimension can be represented as a vector of length $2d$ as $h_m = (y_1, y_2, \ldots, y_d, z_1, z_2, \ldots z_d)$ where y_i $1 \leq i \leq d$ represents the bottom most and left most point of the hyperbox; while z_i, $1 \leq i \leq d$ gives the length of the hypercube along dimension i.

- Purity: Purity in this paper is a measure of the number of classes in a hyperox. A hyperbox in a feature space is said to be completely pure, if the hyperbox contains data points of only one class. In all other cases, the hyperbox is impure. However, a hyperbox can be said to be pure also when its data points are of two or more classes and the data points are evenly distributed, in the sense that all the data points of one class are together. Mathematically, if c is the number of classes in a hyperbox (this number can easily be obtained from the available data), and l is the total number of data points in the hyperbox, then the probability of class k where $1 \leq k \leq c$ in a hyperbox would be given as:

  $$p_k = \frac{l_k}{l}$$

  where, l_k gives the number of data points in the hyperbox in class k and $\sum_{k=1}^{c} l_k = l$. Hence, a hyperbox is completely pure, when p_k is 1 for any value of k and is completely impure if all p_k are equal.
- Resolution: Resolution is a very important concept for the feature space partitionin presented in this paper. It represents the number of divisions that are imposed on each dimension in the feature space. These divisions generate boundaries for hyperboxes in d-dimension which are checked for purity.

III. Proposed Methodology

Following sections discuss the algorithm for partitioning the feature space on the basis of purity. At each resolution, the feature space is partitioned into hyperboxes which are tested for their purity. The hyperboxes that are pure are stored as the indicators of area of a particular class.

A. Algorithm

Each data point in the $d-$dimensional feature space is represented by a vector of d feature values. It is assumed that the feature selection is complete, and hence the training set is able to represent the data fully. The basic idea here is to divide each dimension into some regions, hence getting the hyperboxes. Purity of these hyperboxes is checked. If the number of classes in a hyperbox is found to be one, the description of the hyperbox is stored. If any hyperbox is found to be totally contained in a hpyerbox stored at a lower resolution, the hyperbox is discarded. The approach is quite simple and results show that for complex systems also, accuracy of classification increases. The division of hyperboxes can occur till a point where one gets a hyperbox of unit dimension, i.e. length of a side of hyperbox in each dimension is one.

The algorithm runs as follows:

1) First minimum and maximum points along each dimension are calculated to form a bounding box for the feature space which contains the training data. Let the minimum data point be called $x_{min} = (min_1, min_2, \ldots, min_d)$ where min_i, $1 \leq i \leq d$ is the minimum value of all the data points in i^{th} dimension. Let a maximum data point be called $x_{max} = (max_1, max_2, \ldots, max_d)$ where max_i, $1 \leq i \leq d$ is the maximum value of all the data points in i^{th} dimension. The x_{min} and x_{max} would define the bounding hyperbox of the available data points. Using these, a range vector $R = x_{max} - x_{min}$ is calculated which helps in deciding the division length.
2) Let $r = 2$ where r gives the resolution. This means that lengths along each dimension of the bounding box of all the data will have 2 partitions.
3) Each hyperbox at a resolution r will have its $z_i = \frac{R_i}{r}$ where $1 \leq i \leq d$ while the $y_i = m * z_i + 1$ where $0 \leq m \leq r-1$ and $1 \leq i \leq d$. Using the y_i values and z_i a bounding hyperbox at a resolution r is obtained.
4) For each such hyperbox calculated above, find all the data points x_i such that $\forall l, y_l \leq x_{il} \leq y_l + z_l$ where $1 \leq l \leq d$ and $1 \leq i \leq N$ where N is the total number of data points available for the training set.
   - If the hyperbox under observation lies within the limits of the already stored hyperbox descriptions (which are of lower resolutions), then proceed to the next hyperbox. This hyperbox can be ignored, since it has already been marked as pure at a lower resolution.
   - If data points in the hyperbox come from one class, i.e. $p_k = 1$ for some $k, 1 \leq k \leq c$ where c is the number of classes in the hyperbox, then hyperbox description and the class number for which it is pure are stored. Proceed to the next hyperbox at the current resolution.
5) When all the hyperboxes at the current resolution have been checked, next higher resolution is taken into consideration. Hence, $r = r+1$ and above step is performed.
6) The procedure would end when a predefined resolution has been reached.

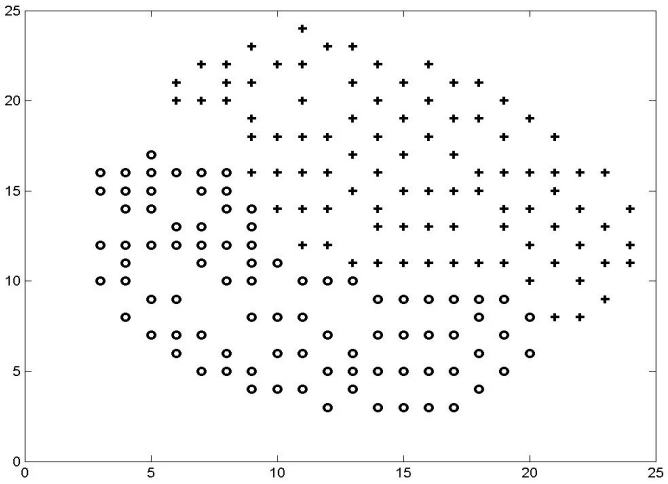

Fig. 1. Plot of Kidney Data

Figure 1 gives a sample plot of data. The data has 2-classes and the featuer space is of 2-dimensions. The plot consists of 181 points. Figure 2 shows the divisions in the feature space

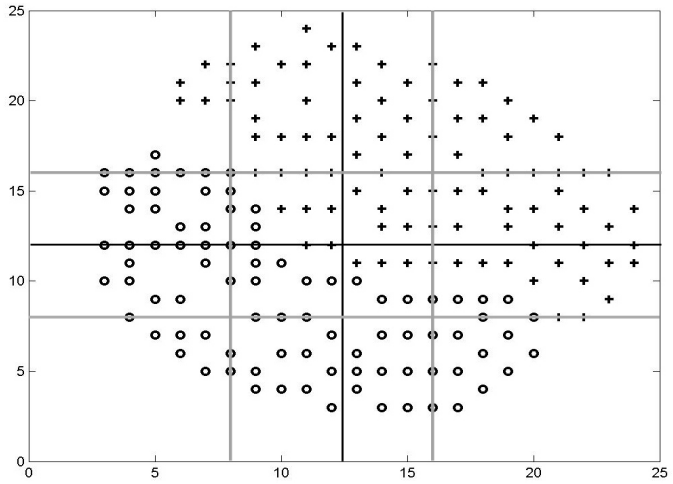

Fig. 2. Kidney Data with r=2 and r=3; black lines division for r=2; gray lines division for r=3

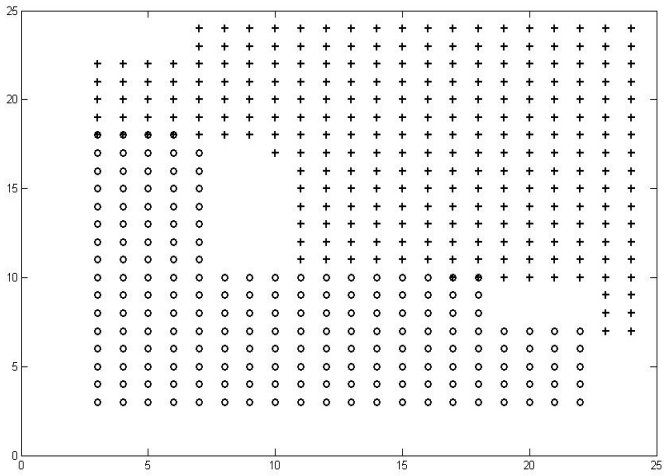

Fig. 3. Result of feature space partitioning till $r = 6$

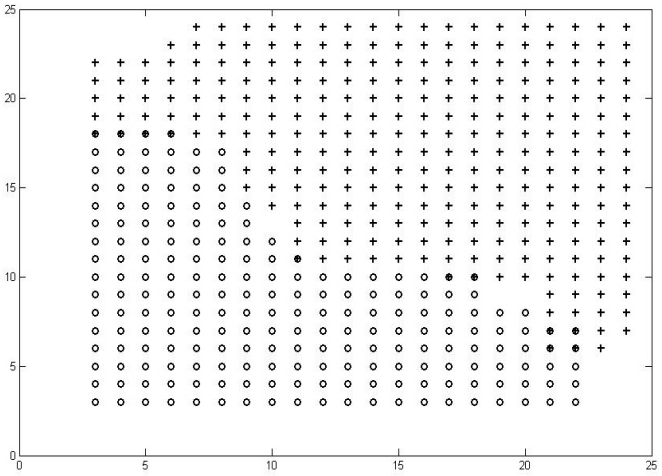

Fig. 4. Result of feature space partitioning till $r = 15$

at resolution of 2 and 3. At $r = 2$, as can be seen from Figure 2, there is one pure box (top right) whose description wil be stored. At $r = 3$ there are 6 boxes which are pure. However one of these boxes lies completely within the pure box obtained at $r = 2$. So the description of the other 5 boxes would stored. This way at subsequent resolutions, the number of pure boxes increases whose description is stored.

Above procedure generates a list of hyperboxes with their two extreme corners and the class which it contains.

B. Classification

Once above training or learning has been done, the classification problem now becomes a find-and-assign problem. An incoming input data point say $in = (in_1, in_2, \ldots, in_d)$ is assigned to a class k if the hyperbox to which it belongs has an assignment to class k. The belongingness to a hyperbox is satisfied if $\forall i, y_i \leq in_i \leq y_i + z_i$ where $1 \leq i \leq d$ for a hyperbox in the list generated in the algorithm for feature space partitioning.

IV. Experimental Results

Feature Space Partitioning was performed on 3 types of data. One of the data was generated synthetically to test the feature space partitioning algorithm on a data where the boundary between the two classes is not that simple to find. This data is plotted in Figure 1 and we call it kidney data.

Figures 3 and 4 show the feature space partitioning method as given by the above algorithm till resolutions of 6 and 15 partitions respectively. As can be seen tll $r = 6$ there is some area between the two classs which is left blank since no pure hyperboxes were encountered here. However, till $r = 15$ this blank area is covered, and the separation between the two regions is complete and a defined area is seen in terms of hyperboxes (limits have not been drawn here).

Other data on which experiments of classification were performed is the vowel data. This is a 3-feature and 6-class data. The feature space was partitioned till a resolution of 13 and 32. 570 points were used for training (i.e. feature space partitioning)and some 300 points were used for testing. Out of the 871 points, 570 points are selected randomly for training and the other 300 points are used for testing. Classification with different sets of random points each time give results which have been tabulated in Table I. As is seen the number of misclassified points and the unclassified points would result into only 10% of the testing data set. However, at resolution 32, the error is much less than that at 13. Table I shows results for 4 separate runs on the vowel data set at 2 different resolutions.

Another data which was experimented on is plotted in Figure 5. As can be seen, one class (class 1) is completely covered by another class (class 2). Feature space partitioning starting from $r = 2$ would generate an area which is inside the other class. This is not wrong. However this also results into an inclusion of those areas which are not part of class 1 at all! For demonstrating the thing, we have done the following setup: 357 points are used for training while other 220 points are used

TABLE I
CLASSIFICATION STATISTICS FOR CLASSIFICATION ON VOWEL DATA BY APPLYING FEATURE SPACE PARTITIONING TILL $r = 13$ AND $r = 32$. EACH RUN HAS A RANDOMLY SELECTED TRAINING AND TESTING SET

till $r = 13$	Run 1	Run 2	Run 3	Run 4
Misclassified Points (out of 300)	19	13	14	13
Unclassified Points (out of 300)	23	18	9	20
Total error points:	42	31	23	33
Percentage Correct classification	86	89.67	92.33	89
till $r = 32$	Run 1	Run 2	Run 3	Run 4
Misclassified Points (out of 300)	16	17	15	13
Unclassified Points (out of 300)	9	11	6	11
Total error points:	25	28	21	24
Percentage Correct classification	91.67	90.67	93	92

TABLE II
CLASSIFICATION STATISTICS FOR CLASSIFICATION ON SYNTHETIC DATA IN FIGURE 5 BY APPLYING FEATURE SPACE PARTITIONING FROM $r = 12$ TO $r = 14$ AND $r = 12$ TO $r = 17$. EACH RUN HAS A RANDOMLY SELECTED TRAINING AND TESTING SET

$r = 12$ to $r = 14$	Run 1	Run 2	Run 3	Run 4
Misclassified Points (out of 220)	0	0	0	0
Unclassified Points (out of 220)	3	2	0	3
Total error points:	3	2	0	3
Percentage Correct classification	98.64	99.09	100	98.64
$r = 12$ to $r = 17$	Run 1	Run 2	Run 3	Run 4
Misclassified Points (out of 220)	0	0	0	0
Unclassified Points (out of 220)	2	0	0	0
Total error points:	2	0	0	0
Percentage Correct classification	99.09	100	100	100

TABLE III
CLASSIFICATION STATISTICS FOR CLASSIFICATION ON IRIS DATA BY APPLYING FEATURE SPACE PARTITIONING TILL $r = 10$ AND TILL $r = 15$. EACH RUN HAS A RANDOMLY SELECTED TRAINING AND TESTING SET

till $r = 10$	Run 1	Run 2	Run 3	Run 4
Misclassified Points (out of 45)	0	0	1	0
Unclassified Points (out of 45)	1	5	1	2
Total error points:	1	5	2	2
Percentage Correct classification	97.77	88.89	95.56	95.56
till $r = 15$	Run 1	Run 2	Run 3	Run 4
Misclassified Points (out of 45)	1	0	0	2
Unclassified Points (out of 45)	0	1	1	1
Total error points:	1	1	1	3
Percentage Correct classification	97.77	97.77	97.77	93.33

for testing. Feature space partitioning is done using these 357 points. Figure 6 shows the boundaries of all the hyperboxes created during the feature space partitioning starting from $r = 2$ to $r = 4$. All other hyperboxes at a higer resolution would definitely fall into these boundaries. The classification turns out to be all correct (because class 2 is completely surrounded by class 1). However, the two partitions in class 1 itself have now been merged which is undesirable. Hence if the feature space patitioning is started with some higher resolution and end up at a suitable higher resolution then the

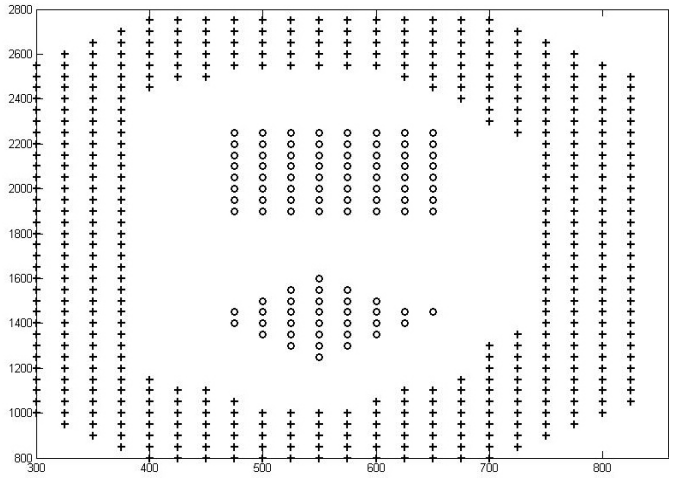

Fig. 5. Plot of data where one class lies within the other

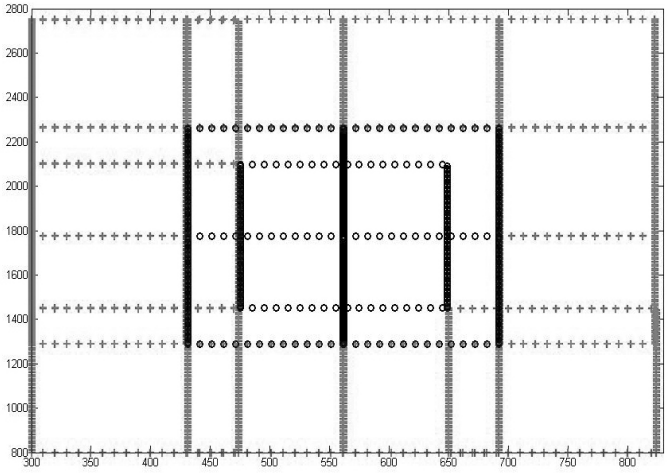

Fig. 6. Plot of boundaries of feature space partitioning on plot of Figure 5, with $r = 2$ to $r = 4$. o is for class 1 and + is for class 2

two partitions in class 1 would be retained and so would be the boundary of class 2. This is illustrated in Figure 7, where resolution has been taken from $r = 12$ to $r = 14$. For this particular run, out of 220 points, only 2 could not be classified. For another randomly selected set of points, Figure 8 shows the boundaries of the hyperboxes created at resolutions $r = 13$ to $r = 15$. Classification statistics for synthetic have been shown in Table II. Hence actually, the classification in this type of data results into a success rate of above 90% if the partitioning is done at the correct resolutions. Note that none of the points here are misclassified but there are points which could not be assigned to any class since there are no hyperboxes listed which could contain them.

Experiments were also done on the standard Iris data. The data is a 4 feature, 3 class data. Here one class is completely separate from the other two classes and the remaining classes are slightly overlapped in the feature space. Performing feature space partitioning on Iris data gives classification rates as shown in Table III.

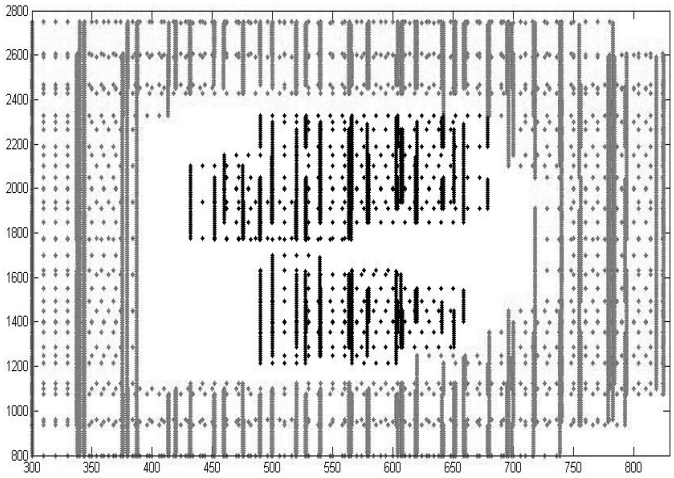

Fig. 7. Plot of boundaries of feature space partitioning on plot of Figure 5, with $r = 12$ to $r = 14$. Black dots are for class 1 and gray ones for class 2

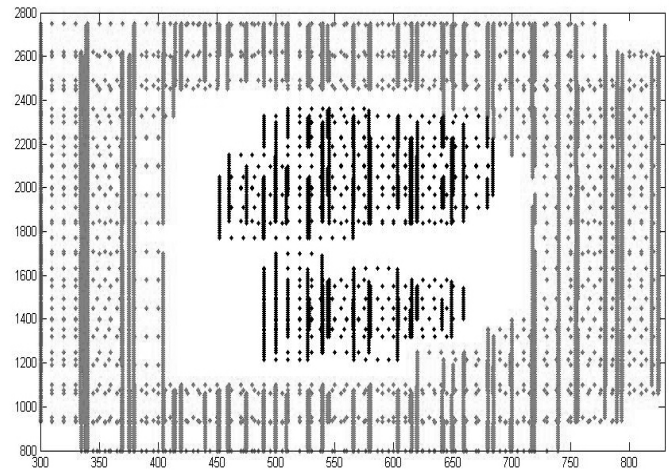

Fig. 8. Plot of boundaries of feature space partitioning on plot of Figure 5, with $r = 13$ to $r = 15$. Black dots are for class 1 and gray ones for class 2

V. Conclusion and Future Work

The classification procedure discussed here is able to create a non-linear boundary between the classes in the feature space allowing for a classification with accuracy of almost above 90%. For completely exclusive classes, the method works quite well even if the structures of the classes are complicated. The only disadvantage of the algorithm can be considered in terms of the high number of hyperboxes generated and the complexity. But this problem can be easily curbed down since the list generates a number of hyperboxes which are overlapping over two or more hyperboxes together.

As discussed in the Results section. an initial resolution can be calculated as in [4] so that unnecessary calculations are removed for lower resolutions, if the data seems to be better representable at a higher resolution. A final resolution where the procedure would end can also be found out. Moreover, such an initial resolution would also be desirable in the cases like the 3rd data set, where one class lies compeletely within another class.

The list of hyperboxes has the chances of having hyperboxes which actually lie completely in areas of two hyperboxes, i.e. some part of the hyperbox lies completely within one hyperbox and some part completely within the other one. The hyperbox in such a condition has to be ignored, since it is already pure and its boundaries stored at a lower resolutions. The algorithm above does not ignore such hyperboxes and hence many unnecessary hyperboxes are created in the list.

The algorithm also lists down all the hyperboxes and the corresponding classes. However, a non-linear separating boundary can be represented easily using those hyperboxes which lie at the edges of the class area keeping in mind how the class distribution falls in the feature space.

Hence the feature space patitioning algorithm discussed here is an effective way of training the data set and hence generating areas of classes resulting into classification successes of as high as 95% accuracy. The way the algorithm works and its pros and cons have been discussed here with explanations given on 3 data set which actually can represent any type of data. With proper improvements in the algorithm, there still exists a possiblity of reducing the number of hyperboxes selected without affecting the overall area finally defined by the method.

Acknowledgement

Authors acknowledge Dr. Sanghamitra Bandyopadhyay of Indian Statistical Institute, Calcutta for sharing the vowel data used in this work.

References

[1] R. O. Duda,P. E. Hart and D. G. Stork, *Pattern Classification*, Second Edition John Wiley (2000)

[2] J. K. Kishore and L. M. Patnaik and V. Mani and V. K. Agrawal *Genetic programming based pattern classification with feature space partitioning*, Information Science 2001, Vol. 131, pp. 65-86, Publisher: Elsevier Inc.

[3] Yinghua He, Bo Zhang and Jianmin Li, *A new multiresolution classification model based on partitioning of feature space*, Proceedings for 2005 IEEE International Conference on Granular Computing 25-27 July 2005, Vol 2 pp. 462-467(6)

[4] D. P. Mandal, *Partitioning of Feature Space for Pattern Classification*, Pattern Recognition, December 1997, Vol 30 no. 12, pp. 1971-1990(20) Publisher: Elsevier Science Ltd.

[5] S. Singh, *PRISM - A novel framework for pattern recognition*, Pattern Analysis and Application Springer London June 2003, Vol 6. no. 2, pp.134-149 (16)