Pattern Classification Modeled by Goal Programming

Dhruv Saksena **Shruti Garg** **Suman Mitra**
Dhirubhai Ambani Institute of Information and Communication Technology, Gandhinagar, Gujarat

Abstract— **The Goal Programming (GP) approach is used to model problems of Pattern classification. It involves finding the separating boundary lines between different classes to get minimum misclassification. A theoretical overview of solving the problem using GP is discussed and its different variants are applied to various datasets to show the effectiveness of the algorithm. The datasets considered for experimentation are taken to be in 2 – dimensional Euclidean space for better visualization of separating boundaries. Finally, the results are compared with the K-Nearest Neighbor Classifier.**

Index Terms—: **Goal Programming, Pattern Classification, K-Nearest Neighbor**

I. INTRODUCTION

Pattern Classification is a paradigm, which comes under the domain of Pattern Recognition, which aims at classifying patterns into predetermined categories/classes in a multidimensional space. This classification is based on certain attributes/features which are common to all objects. First, the features are determined for each object and then certain decision rules are formulated and based on these rules, the classification of patterns into different classes is obtained. Pattern recognition has found enormous number of applications in varied number of fields. There are many approaches to pattern classification. The algorithm for classifying objects/pattern is usually called classifier. A few of the widely used classifiers are the Bayesian Classifier, Neural networks, Support Vector Machine and the K Nearest Neighbor. The details of these are available in [1].

In this paper, an attempt is made to model pattern classification problem using Goal Programming technique (GP), an extension to linear programming, deals with multiple objectives. Unlike linear programming, the objectives can be violated up to a certain extent, but the aim is to ensure minimum deviation from them. For finding the solution to a Goal Programming problem, it is usually converted to a linear programming problem (LPP) [2].

The next section deals with mathematical formulation for Goal Programming. The proposed methodology is described in Section 3 followed by implementation and results in Section 4.

II. MATHEMATICAL FORMULATION OF GOAL PROGRAMMING

In Goal Programming, there can be different types of goals to be achieved. Let there be a linear objective function $\mathbf{c_1}^T\mathbf{x}$ whose goal is to remain less than equal to g_1 ($\mathbf{c_1}^T\mathbf{x} \le g_1$). Similarly there can be a case where a function $\mathbf{c_2}^T\mathbf{x}$ must be greater than or equal to g_2 ($\mathbf{c_2}^T\mathbf{x} \ge g_2$). Such kinds of goals are called one-sided goals. Also, there can be two sided goals such as $g_3 \le \mathbf{c_3}^T\mathbf{x} \le g_4$. The aim here is to achieve minimum deviation from goals under linear constraints if any.

The Goal Programming problem can be converted into a linear programming minimization problem as follows.

Let us consider two objective functions $\mathbf{c_1}^T\mathbf{x}$ and $\mathbf{c_2}^T\mathbf{x}$ with one-sided goals as g_1 and g_2 respectively and a constraint ($A\mathbf{x} \ge b$).

$$\mathbf{c_1}^T\mathbf{x} \le g_1 \quad (1)$$

$$\mathbf{c_2}^T\mathbf{x} \ge g_2 \quad (2)$$

Constraint $$A\mathbf{x} \ge b \quad (3)$$

Let there exists one $\mathbf{x}$ which satisfies (3).Also, $\mathbf{x}$ is such that $\mathbf{c_1}^T\mathbf{x}$ exceeds g_1 by d_1 and $\mathbf{c_2}^T\mathbf{x}$ falls short of g_2 by d_2, where d_1 and d_2 are absolute deviations from the goals.

$$\mathbf{c_1}^T\mathbf{x} - d_1 = g_1 \quad (4)$$

$$\mathbf{c_2}^T\mathbf{x} + d_2 = g_2 \quad (5)$$

Now, the whole objective of goal programming is to minimize the deviations from the objective targets. So, the whole problem boils down to Linear Programming Problem, where

Objective is Minimize $D=\alpha_1 d_1+\alpha_2 d_2$ (6)
Subject to (3),(4) and (5). [3]

Here, α_1,α_2 are the assigned weights given to the deviations if one want to give unequal wieghtages to the two given goals. Note that, when α_1 is not equal to α_2 it is called as *Preemptive Goal Programming*. [2]

If we generalize it for m number of goals then there will be 'm' number of equations each resulting in some amount of deviation.

In that case objective function will become $D=\sum_{i=1}^{i=m}\alpha_i d_i$, subject to constraints that result into these deviations. Based upon the above given mathematical formulation an algorithm is proposed to model Pattern Classification using Goal Programming.

III. Description of Proposed Methodology

Let us consider a two class problem for which two features (X,Y) of each class are given. Figure 1 shows the plot of the data points. Our aim is to find a separating hyperplane (a straight line in this case) as indicated by H in the figure that divides the data points into two classes. The equation of H is

$$y - mx - c = 0 \qquad (7)$$

where, m and c have their usual meanings.

Any point (x_1, y_1) in class A when substituted in (7) should give a positive value. The first goal then becomes

$$y_1 - mx_1 - c > 0 \qquad (8)$$

Similarly, for any point (x_2, y_2) in class B when substituted in (7) should give negative value. Hence, second goal is

$$y_2 - mx_2 - c < 0 \qquad (9)$$

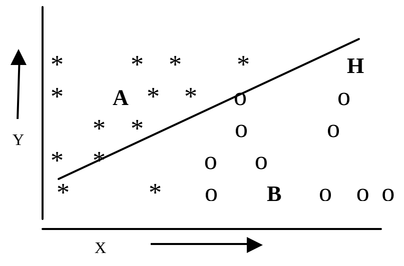

Fig.1 A hyper plane H separating two classes A and B.

But due to some misclassifications as indicated in Figure 1, (8), (9) may not be satisfied. For a misclassified point (x_1', y_1') in class A, the deviation from the goal would be

$$|y_1' - mx_1' - c| / (1 + m^2)^{1/2} = \alpha \qquad (10)$$

For a misclassified point (x_2', y_2') in B, the deviation is

$$|y_2' - mx_2' - c| / (1 + m^2)^{1/2} = \beta \qquad (11)$$

Where α and β are the distances of the respective points from the separating line.

So, our aim is to minimize this overall deviation ($\alpha+\beta$) .Again if we have i=1,2,…p number of misclassifications for class A and j=1,2,…q number of misclassifications for class B, then the entire problem can be looked upon as

$$\text{Minimize} \qquad D = \sum_{i=1}^{i=p}\alpha_i + \sum_{j=1}^{j=q}\beta_j, \qquad (12)$$

subject to (10) and (11).

The above optimization problem is solved to get the values of m and c. Note that, the current optimization problem is no longer a Linear Programming Problem as equations (10),(11) are not linear in nature. We are proposing an approach, which is almost like an exhaustive search to find the optimal values of m and c, that give the separating boundary.

IV. Determination of Seperating Boundary

The problem now is to find 'm' and 'c' (or the line y = mx + c) such that (12) is minimized. We assume 'c' is bounded in [c_{min}, c_{max}], thus for each choice of c, we find the optimum 'm'. The search space for m is also restricted as shown in Figure 2. Both classes A and B are bounded by rectangles. The intersection of these two rectangles provides the bounded search space for m as indicated in Figure 2. The choice of m is then restricted within m_{min} and m_{max} where,

$$m_{min} = \begin{cases} (y_{min} - c)/x_{min} & c <= y_{min} \\ (y_{max} - c)/x_{min} & y_{min} < c < y_{max} \\ (y_{max} - c)/x_{max} & c >= y_{max} \end{cases}$$

$$m_{max} = \begin{cases} (y_{min} - c)/x_{max} & c <= y_{min} \\ (y_{max} - c)/x_{max} & y_{min} < c < y_{max} \\ (y_{min} - c)/x_{min} & c >= y_{max} \end{cases}$$

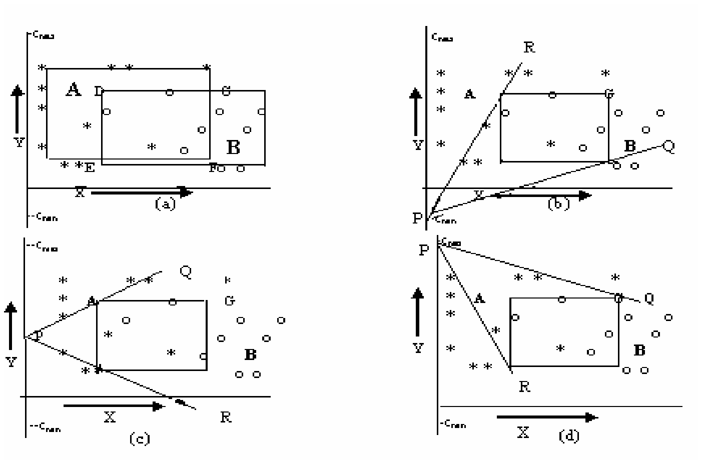

Fig.2(a) Enclosing two classes with rectangles. Rectangle DEFG is the intersection of these rectangles and (b), (c), (d) show the search space for m for a given choice of c.

To find optimal c and m, the entire range of c = [c_{min}, c_{max}] is searched with a step size (dist/2) where dist = min {d(x_1, x_2) | x_1,x_2 are two data points} and d is the distance between x_1 and x_2. Similar argument is given in [4]. Also the entire range of m is searched with a step size 1/n where n is the number of hyper planes that we check for a particular value of c. Though an exhaustive search is carried out at present, but a Genetic Algorithm based search similar to that given in [4] can also be carried out.

As there can be more than one combination of m and c for which deviation D comes out to be minimum, there is a possibility of getting more than one separating lines.

We are mostly presenting the results for 2 dimensional datasets using the present method. The results are also compared with the K – Nearest Neighbor classifier.

V. IMPLEMENTATION AND RESULTS

The algorithm was implemented on several datasets even with more than two classes. Figure 3 shows the simulation result of a 3 class artificial data. Each class can be identified by a combination of signs [5] of the equation of the two lines **H_1** and **H_2** as shown in figure 3(a).

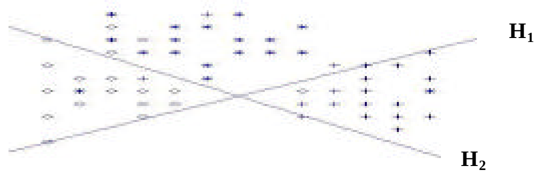

Fig.3(a) Results obtained for a multi-class dataset where we have 3 classes.

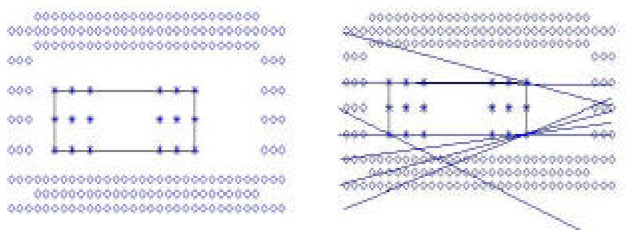

Fig. 4(a) An artificial dataset with two classes where one class is within another class (b) Boundary lines separating inner class from the outer class.

The proposed method is implemented on a non-convex dataset as shown in figure 4(a). In this case, we used preemptive Goal Programming and assigned more weightage to the inner class, as we wanted to separate it from the outer one. For every value of *c*, we choose the value of *m* which shows minimum deviation To get the final separating boundaries, we sort the deviations obtained in increasing order. The first value in this sorted sequence is selected to get the separating boundary corresponding to that deviation. If the resulting line does not classify the dataset completely, we consider the next line corresponding to the next values of the deviations in the sequence and find out the deviations as a result of the combination of these lines. The process is repeated until the combined deviation tends to stabilize. The final decision boundaries are as shown in Figure 4(b). Similar approach was used for the dataset shown in the Figure 5(a).

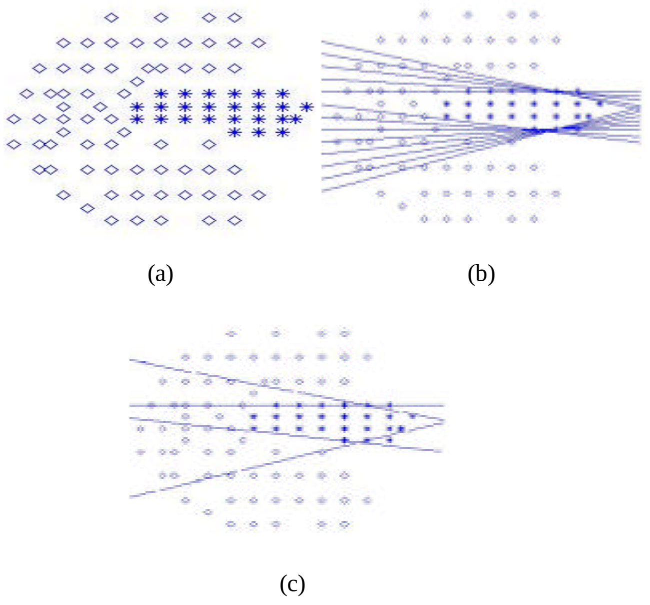

Fig.5 (a) An artificial dataset (b) Simulations using *preemptive goal programming* (c) Boundary lines separating inner class from the outer class.

The classification results obtained by the proposed method are compared with the results obtained by K-Nearest Neighbor. The results are presented in Table 1.

Dataset	Goal Programming	K – Nearest Neighbour			
		k =1	k=2	k=4	k=8
Fig. 3(a)	82.4%	66.67 %	66.67 %	66.67%	66.67 %
Fig 4(a)	95.75%	96.45%	96.45%	96.45%	93.61 %
Fig 5(a)	93.75%	86.2%	88.3%	86.2%	87.25 %

Table 1. Results of different datasets using different algorithms

VI. CONCLUSION

Goal Programming is used to model problems of pattern classification and optimal separating boundaries can be achieved using exhaustive search. The algorithm can also be extended to Genetic Algorithm based search. This algorithm when applied to different datasets, worked satisfactorily as compared to the K-Nearest Neighbor classifier. Also, note that, while in K-Nearest Neighbor approach, a prior availability of an already classified dataset is required [1], there is no such requirement in case of Goal Programming.

REFERENCES

[1] Duda Richard O. , Hart Peter E. and Stork David G "*Pattern Classification*", Second Edition, 2007, Page(s) 20 –27.

[2] Rardin Ronald L., "*Optimization in operations Research*", 2005 , Pearson Education ,Page(s) 389-408.

[3] H. Poor, *An Introduction to Signal Detection and Estimation*. New York: Springer-Verlag, 1985, ch. 4.

[4] Nakayama Hirotaka and Asada Takeshi "*Support Vector Machines using multiple objective programming and goal programming*" Proceedings of the 9th International Conference on NIPS , Vol.2,2002.

[5] Bandhopadhyay S. , Murthy C.A., Pal S.K., "*Pattern classification with genetic algorithms*", *Elsevier Science, Pattern Recognition Letters 16*, pp 801-808, 1995.

[6] Hiller Federick S. and Lieberman Gerald J. "*Introduction to Mathematical Programming*", Second Edition, 1995, Page(s) 285 – 292