Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 186 (2021) 381–386

Procedia Computer Science

www.elsevier.com/locate/procedia

14th International Symposium “Intelligent Systems”

Machine learning algorithm based on convex hull analysis

A.P. Nemirko[a,*], J.H. Dulá[b]

[a]*Saint Petersburg Electrotechnical University “LETI”, Prof. Popov st., 5, Saint Petersburg 197376, Russia*
[b]*University of Alabama, Box 870226, Tuscaloosa AL 35487, U.S.A.*

Abstract

In this paper machine learning methods for automatic classification problems using computational geometry are considered. Classes are defined with convex hulls of points sets in a multidimensional feature space. Classification algorithms based on the estimation of the proximity of the test point to convex class shells are considered. Several ways of such estimation are suggested when the test point is located both outside the convex hull and inside it. A new method for estimating proximity based on linear programming is proposed, and the corresponding nearest convex hull classifier is described. The results of experimental studies on the real medical diagnostics problem are presented. An efficiency comparison of the proposed classifier and other types of classifiers, both based on convex hull analysis and not, has shown the high efficiency of the proposed method for estimating proximity based on linear programming.

Keywords: Intelligent systems; computational geometry; pattern recognition; nearest convex hull classification; linear programming; automatic medical diagnostics.

1. Introduction

In many machine learning cases, the analyzed objects are described by a set of vectors in a multidimensional feature space. Such objects are considered and analyzed mainly from a mathematical statistics perspective. While this approach has its advantages, it is worth noting that the probabilistic models sometimes provide only a rough approximation of the real data. In this paper, the application of computational geometry methods to machine learning problems is examined. Despite the computational difficulties in constructing convex hulls, this approach has been used in problems of feature space reduction [1], for studying the intersection of classes, visualization of their intersection area [2], classification [3, 4, 5, 6] and clusterization [7]. Instead of a set of points in a feature space that describes a specific class, we suggest using their convex hull and treating the intersection of classes as the intersection of their convex hulls. Many machine learning algorithms based on the geometric approach calculate the distance from a point to a convex hull and the distance between convex hulls. In the case of intersecting convex hulls in multidimensional

* Corresponding author. Tel.: +7-921-744-42-15.
E-mail address: apn-bs@yandex.ru

10.1016/j.procs.2021.04.160

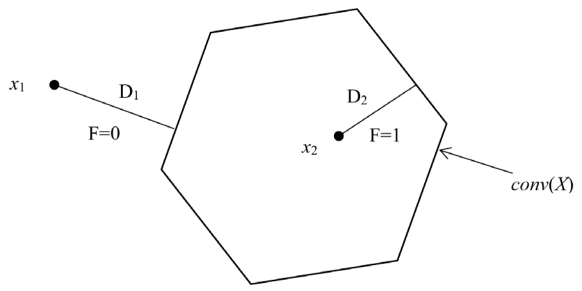

Fig. 1. The distance between a point and a convex hull: x_1 – a point outside the convex hull, x_2 – a point inside the convex hull, F – location labels of the points, D1, D2 – distances to the convex hull.

space, or when the test point is located inside a convex hull, calculating these distances may pose some difficulties. In this paper, the use of the approximation method based on the computational geometry results [8] is considered.

2. Classification based on the nearest convex hull

Many computational geometry methods are based on the concept of the convex hull (CH). The convex hull of a set X is the smallest convex set containing X. Let us consider the set of m vectors in multidimensional space as $X = \{x, x_i \in \Re^n; i = 1, \ldots, m\}$. The convex hull generated by this set is defined as:

$$\mathcal{P} = \text{conv}(X) = \left\{ v | v = \sum_{i=1}^{m} x_i \lambda_i;\ \sum_{i=1}^{m} \lambda_i = 1;\ \lambda_i \geq 0; x_i \in X \right\};$$

where λ_i are non-negative scalar coefficients. Let the set of extreme points (vertices) of a convex hull $\mathcal{P}$ be a frame FR of X: FR $\subseteq \mathcal{P}$.

There are methods for constructing CHs in 2-dimensional and 3-dimensional spaces: algorithms of Graham [9], Jarvis [10], Chan [11], etc. Several algorithms are known for calculating convex hulls in a high dimensional feature space, such as Quickhull [12], but their implementation is currently limited by both the dimension of the feature space and the power of the training set.

The nearest convex hull classifier determines the distance from the test point to the convex hull of each class. Similar to the nearest neighbor method, a point belongs to the class the convex hull of which is closest. The distance between x and the convex hull conv(X) in Euclidean space can be calculated as:

$$D = \min_j ||x - x_j||;\ x_j \in \text{conv}(X). \tag{1}$$

Expression (1) will not change whether the point x is located outside or inside of the convex hull. However, if it is possible to determine the location of the point, it is beneficial to introduce a label F in addition to the D, which is equal to 0 for external and 1 for the internal location of a point relative to CH (Fig.1). In the case of internal point placement, the shell is not convex, but concave and the expression (1) can be applied to different facets of the convex hull.

The main issue of all algorithms, using the nearest convex hull, is the method of determining the distance from a point to a convex set.

3. Methods for measuring the proximity of a test point to a convex hull.

For linearly separable classes, the problem of finding the minimum distance between convex hulls of classes (NPP - nearest point problem) is solved by algorithms: SVM [13, 14], SK-algorithm (Schlesinger–Kozinec algorithm) [15], MDM-algorithm (Mitchell–Dem'yanov–Malozemov algorithm) [16]. The same algorithms are applied when measuring the distance between a point, located outside the convex hull, and a convex hull.

In the case of convex shells intersection, as well as in the case of the test point located inside the convex shell, the concept of penetration depth (PD) is used [17, 18]. This measure shows the degree to which objects intersect. Let us assume that object A can move, while object B is stationary. For a given direction vector u, the *directional penetration depth* between A and B is defined as a minimum distance A has to be shifted in the u direction to avoid any intersection between A and B. The *overall penetration depth* between A and B stands for the minimum distance A has to be shifted in *any direction* to avoid intersection between A and B. These parameters can be applied to linearly separable classes. In that case, the distance between classes is calculated as the minimum distance A has to be shifted in any direction (or in the u direction when calculating the directional penetration depth), so that A starts to touch B, but still does not collide with it.

There are algorithms for finding the overall PD for 2D and 3D cases [17, 18], but for the nD case, finding the overall PD is difficult and computationally costly. It is much simpler to find a directed PD, which is the task when considering projections of convex hulls to a specific direction in a multidimensional space. In [6], directed PD is used as an approximation to the overall PD for creating an algorithm for classification with a teacher. The u direction was always chosen from the test point to the centroid of the class. The proximity calculation was done by analyzing the projections of sets of class points on the u direction. Since only the convex hull's extremal points were involved in this analysis, instead of projections of full sets X_i on u, one can only use projections of the vertices of convex hulls on u, which gives the same results. It allows us to describe classes using only the vertices of their convex hulls, rather than the original sets X_i. In order to do this, the set of vertices of convex class shells must be explicitly calculated from the original training set. Calculating the list of vertices during the training step is a more straightforward task compared to the complete construction of a convex hull, including its vertices and faces. Besides, fewer points have to be projected on the selected directions during the classification phase, which reduces processing time. The use of vertexes dramatically simplifies both the task of training and the task of making decisions when classifying test points.

4. A new method for estimating the proximity of a test point to a convex hull.

In [8], an efficient algorithm for identifying a frame FR of a point set X is described. In this work a linear program, LP1, is formulated the optimal solution of which provides information about the location of a point with respect to the boundary of the convex hull of a point set. Consider a test point b and a point set X such that conv(X) contains the origin. The "gauge" LP is given by:

$$\begin{aligned} z^* = \min_{\lambda\in\Re^m} \quad & \textstyle\sum_{j=1}^{m} \lambda_j, \\ \text{s.t.} \quad & \textstyle\sum_{j=1}^{m} x_i\lambda_i = b, \\ & \lambda_j \geq 0;\ j = 1, \dots, m; \end{aligned}$$

where b is an arbitrary, non-zero, vector in $\Re^n$. The optimal solution to the LP provides information about the vector b as follows [8]

1. $z^* < 1$: The point b is interior of conv(X).
2. $z^* = 1$: The point b is on the boundary of conv(X).
3. $z^* > 1$: The point b is exterior of conv(X).

This relation between a right-hand vector b and the solution to the gauge LP can be illustrated with the help of Fig. 2. The figure shows three test points in the gauge LP: b_1, b_2, b_3 located in the interior, boundary, and exterior, respectively, of the convex hull $\mathcal{P}$ of a two-dimensional point set X. These three points are right-hand sides for the gauge LP. The optimal objective function value of the gauge LP, z^*, is the scaling factor for extending or contracting b so that it lands on the boundary of the convex hull. That is, the point $(1/z^*)b$ is on the boundary of conv(X).

The above results suggest an approach for estimating proximity of a point to a convex hull based on solving a gauge LP. The ratio of the Euclidean distance, D, from a test point, b, to the boundary of a convex hull, along the

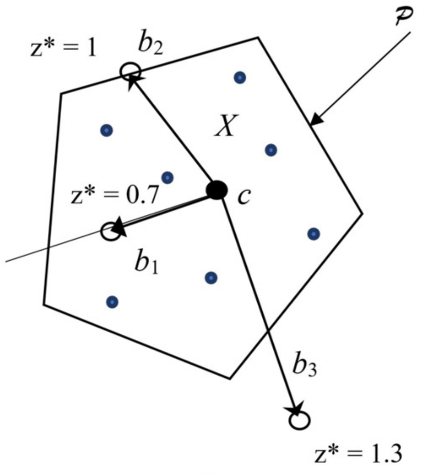

Fig. 2. A two-dimensional set X; its convex hull, $\mathcal{P}$; the origin (centroid), c; test points (vectors) b_1, b_2, b_3, which are different right-hand sides of the gauge LPs; and the optimal objective function values for these LPs for these points, z^*.

length of the vector b, is equal to the ratio $|z^* - 1|$ to z^*. That is, regardless of the test point location ($F = 0$ or $F = 1$) the following is true:

$$\frac{D}{||b||} = \frac{|z^* - 1|}{z^*} \tag{2}$$

which can be rewritten as

$$D = ||b||\ |z^* - 1|/z^*. \tag{3}$$

The value for D will be the actual Euclidean distance of b to the specific point on the boundary of the convex hull at where it pierces it if b is orthogonal to that face. Otherwise, and because different shapes and asymmetries may mean a different face is closer, the defined distance D is an upper bound for the true distance to the convex hull. Making the origin the centroid of the set X, better yet, that of its frame, can be expected to enhance the performance of this approximation.

5. The nearest convex hull classifier

The basics described above allow one to approximate the distance between the test point and the convex hull. They are implemented in the following algorithm.

```
Algorithm DISTANCE_LP
[INPUT:] Data matrix $X_i, b$.                    /*Point set for class $i$ and test point.*/
[OUTPUT:] $F_i$, $D_i$      /* Crossing label and distance between test point and the convex hull for class $i$.*/
1.0 Initialization:
    1.1 Convert point set into data matrix $\mathbf{X_i}$.
    1.2 Calculate centroid, $c$, of $\mathbf{X_i}$.
    1.3 Set $F_i = 0$.
    1.4 Translate data by centroid: $\mathbf{X'_i} = \mathbf{X_i} - c;\ b' = b - c$.
2.0 Solve gauge LP with $\mathbf{X'_i}, b'$ to obtain $z^*$.
3.0 If $z^* < 1$ then $F_i = 1$.
4.0 Calculate $D_i$ using Equation (3).
5.0 End.
```

Notes about Algorithm `DISTANCE_LP`**.**

1. $\mathbf{X_i'}$, is a data matrix, evaluated from the set X_i; F_i is a label assigned to a test point based on its location relative to $\mathcal{P} = \text{conv}(X_i)$ where $F_i = 1$ if the point lies inside $\mathcal{P}$ and $F_i = 0$ for points outside $\mathcal{P}$.
2. For the purpose of using the gauge LP to approximate distances, it is beneficial to use only the frame, FR, of the data; that is, only the extreme points of the convex hulls. In addition to the advantages of having fewer variables in the gauge LP, excluding elements of the a class's interior point for the purpose of calculating D will remove the unpredictable effect of these points would have in the location of the centroid.

For the case of a multiclass problem with k classes, $X_i; i = 1, \dots, k$, one gets k unions $(F_i, D_i); i = 1, \dots, k$. With these unions, the test point classification should be done using the following rules:

1. If no union contains $F = 1$, the label of the recognized class corresponds to the class with the minimal distance to the test point.
2. If a single union contains $F = 1$, the label of the recognized class corresponds to that single union.
3. If several unions (or even all of them) contain $F = 1$ and its unions indexes form a set G, the label of the recognized class corresponds to the class in G with the minimal distance to the test point.

6. Experiment results.

The proposed classification algorithm was tested on a well-known problem of medical diagnosis of breast cancer [19]. The results are shown in Table 1. In the table p_1 & p_2 are the decision error rates for classes B (benign) and M (malignant) respectively; $p = \frac{1}{2}(p_1 + p_2)$. The accuracy metric for the SNCH algorithm is provided in paper [3]. The results of the proposed algorithm without cross-validation using a set of 444 "benign" cases and 239 "malignant" cases are shown under the label NCH-1. The second test, labeled NCH-2, was conducted with a train and test sets of equal power (each is half of the original data set). The convex hulls were calculated using train set data. For reference, the results of SVM and kNN algorithms are provided.

Table 1. Recognition errors for various algorithms. NCH-1 - proposed algorithm without cross-validation, NCH-2 – proposed algorithm with 2-fold cross-validation

Data	p_1	p_2	p
SVM (linear)	4.10	2.00	3.05
SNCH [3]	–	–	2.70
LNCH [6]	2.93	4.18	3.56
kNN ($k = 5$)	2.25	3.15	2.70
NCH-1	0.45	0	0.23
NCH-2	1.8	2.5	2.15

7. Conclusion.

A new method for estimating the proximity of a test point to convex class hulls is proposed. This method is based on applying a "gauge" linear programming formulation that estimates the distance from the test point to the convex hull along the line from the class centroid to the test point. Using only the hull's vertices as a class description instead of the full set of points improves algorithm performance. Experiments show that the proposed method provides substantial improvements compared to the previously proposed LNCH method in terms of recognition quality. The method is easy to implement, does not require user-defined parameter settings, and can be used for multi-class problems. The challenge of calculating the frame of the convex hull are addressed in the large literature which includes several efficient algorithms.

Acknowledgments.

This research was supported by Russian Foundation for Basic Research (RFBR), projects 18-07-00264 and 19-29-01009.

References

[1] Ostrouchov, G. and Samatova, N.F. (2005). On FastMap and the convex hull of multivariate data: toward fast and robust dimension reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8), pp.1340–1343.

[2] Nemirko, A.P. (2018). Multidimensional Data Visualization Based on the Minimum Distance. *Pattern Recognition and Image Analysis Between Convex Hulls of Classes*,28(4), pp.712–719.

[3] Nalbantov, G. and Smirnov, E. (2010). Soft nearest convex hull classifier. In: *Proceedings of the 2010 conference on ECAI 2010: 19th European Conference on Artificial Intelligence*. pp.841–846.

[4] Zhou, X. and Shi, Y. (2009). Nearest neighbor convex hull classification method for face recognition. In: *International Conference on Computational Science*. pp.570–577.

[5] Qing, J., Huo, H, and Fang, T. (2008). Nearest convex hull classifiers for remote sensing classification. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 37, pp.589–594.

[6] Nemirko, A.P. (2019). Lightweight nearest convex hull classifier. *Pattern Recognition and Image Analysis*, 29(3), pp.360–365.

[7] Theljani, F., Laabidi, K., Zidi, S. and Ksouri, M. (2013). Convex hull based clustering algorithm. *International Journal of Artificial Intelligence*, 10, pp.51–70.

[8] Dulá, J.H. and Helgason, R.V. (1996). A new procedure for identifying the frame of the convex hull of a finite collection of points in multidimensional space. *European Journal of Operational Research*, 92, pp.352–367.

[9] Graham, R.L. (1972). An efficient algorithm for determining the convex hull of a finite planar set. *Information Processing Letters*, 1(4), pp.132–133.

[10] Jarvis, R.A. (1973). On the identification of the convex hull of a finite set of points in the plane. *Information Processing Letters*, 2(1), pp.18–21.

[11] Chan, T.M. (1995). *Output-sensitive construction of convex hulls*. University of British Columbia.

[12] Barber, C., Dobkin, D.P. and Huhdanpaa, H. (1996). The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS)*, 22(4), pp.469–483.

[13] Vapnik, V. (1998). *Statistical learning theory*. Wiley.

[14] Bennett, K.P. and Bredensteiner, E.J. (2000). Duality and geometry in SVM classifiers. *ICML*, pp.57–64.

[15] Franc, V. and Hlaváč, V. (2003). An iterative algorithm learning the maximal margin classifier. *Pattern Recognition*, 36(9), pp.1985–1996.

[16] Mitchell, B.F., Demyanov, V.F. and Malozemov, V.N. (1974). Finding the point of a polyhedron closest to the origin. *SIAM Journal on Control*, 12(1), pp.19–26.

[17] Weller, R. (2013). *New geometric data structures for collision detection and haptics*. Springer Science & Business Media.

[18] Lin, M.C., Manocha, D. and, Kim, Y.J. (2018). Collision and proximity queries. In: Goodman, J.E., O'Rourke J. and Toth C.D., ed. *Handbook of Discrete and Computational Geometry*. CRC Press, pp.1029–1056.

[19] UCI Machine Learning Repository (2020). *Breast Cancer Wisconsin (Original) Data Set*. [online] Available at: https://archive.ics.uci.edu/ml/datasets/breast+cancer+wisconsin+(original).