Fast and Robust Supervised Learning in High Dimensions Using the Geometry of the Data

Ujjal Kumar Mukherjee, Subhabrata Majumdar, and Snigdhansu Chatterjee(✉)

University of Minnesota, Minneapolis, MN 55455, USA
chatt019@umn.edu

Abstract. We develop a method for tracing out the shape of a cloud of sample observations, in arbitrary dimensions, called the *data cloud wrapper* (DCW). The DCW have strong theoretical properties, have algorithmic scalability and parallel computational features. We further use the DCW to develop a new fast, robust and accurate classification method in high dimensions, called the **geometric learning algorithm** (GLA). Two of the main features of the proposed algorithm are that there are no assumptions made about the geometric properties of the underlying data generating distribution, and that there are no parametric or other restrictive assumptions made either for the data or the algorithm. The proposed methods are typically faster and more robust than established classification techniques, while being comparably accurate in most cases.

1 Introduction

We propose a new method for classification, that respects the inherent geometry of the data cloud for each labeled group of observations, and this method is not subject to curse of dimensionality. Our method is based on *multivariate quantiles*, which generalize the notion of quantiles for observations in dimensions greater than one. Arising naturally from the concept of multivariate quantiles is the notion of *data depth*, which is a relative measure of proximity of a given point in space to a collection of observations. For any new or unlabeled observation in the feature space, we estimate the label by computing its depth from the data clouds corresponding to a training data.

There are two important properties of the classification technique presented below. First, we do not make assumptions about the geometrical features of the multidimensional data (for which we use the term *data cloud*) corresponding to the various labels in the training sample. Thus, the proposed method respects the geometric properties of the data, and does not impose shape restrictions on it, hence we call it *geometric learning* algorithm (GLA hereafter). Second, our method is scalable and parallelizable with respect to dimensions and sample size, and does not suffer from the curse of dimensionality, and hence is extremely fast in implementation. It can be seen from the development below that our proposed method extends readily to several other supervised learning problems.

The strengths of the proposed geometric learning method arises from the fact that it is based on multivariate quantiles. In Sect. 2 we discuss these quantiles in

P. Perner (Ed.): ICDM 2015, LNAI 9165, pp. 109–123, 2015.
DOI: 10.1007/978-3-319-20910-4_9

details. Based on projection quantiles, which are a form of multivariate quantiles, we develop a *Data Cloud Wrap* (DCW) procedure that provides a very accurate description of the geometry of any sized data set in any dimension. As an illustration, consider Fig. 1, which contains two bivariate scatter plots, the left panel being that of observations from a Gaussian distribution and the right panel is where observations are from a mixture of two Gaussian distributions. The red curves are obtained by the DCW procedure, and it can be seen that these curves quite accurately capture the geometry of the layout of the observations in either panel. The blue curves in either panel correspond to a *projection quantile* (PQ), which reasonably trace the shapes of the data clouds, but not as accurately as the DCW curves. The black curves are obtained by presuming a Gaussian distribution for the data, with only mean and functions as unknowns. Notice that while this is adequate for capturing the shape of the data cloud when the Gaussian assumption holds, it is a severe misfit when the assumption is violated. The regions enclosed by the different curves in either panel are not expected to have identical probabilistic coverage, owing to different mathematical properties.

Note that in high dimensions, it is essentially impossible to graphically or otherwise elicit how and where assumptions like Gaussian shape of the data geometry are violated. Even if such elicitation were feasible, it is unclear how to use that information for supervised learning, or other data-related tasks. The *geometric learning algorithm* we present here is a clear alternative, that does not rely on such encumbering assumptions.

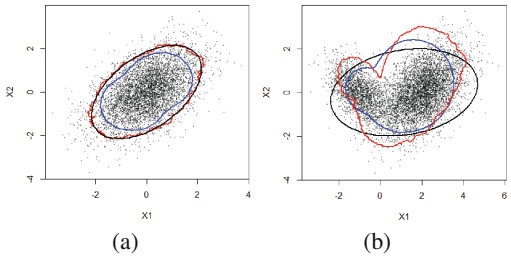

Fig. 1. Comparison of usual projection quantiles (blue) with weighted projection quantiles (red), along with a Gaussian confidence ellipsoid (black) for a Gaussian scatter in (a) and mixture of Gaussians in (b). Areas under the different curves are not expected to be equal (Color figure online).

We discuss below how the sets of Fig. 1, and their enclosing boundary curves, may be indexed by vectors of the unit sphere in the feature space. Curves and enclosed sets as in Fig. 1 are fast and accurate visualization tools that are easily available from the proposed procedure. These graphical techniques are naturally best suited for two and three dimensional projections of the data, however, the construction of such sets in any dimension is simple and quick in our proposed methodology. We can take full advantage of distributed and parallel computing tools for this purpose, since the constructions of sets like the ones depicted in

Fig. 1 is linear in both dimension (p) of the feature space, and the number of observations (n), and parallelizable in both dimensions and sample size. Since the DCW is central to the geometric learning procedure, we present theoretical properties of it in Sect. 3.

We also use the DCW algorithm to compute the *data-depth* of any point in the feature space, with respect to any probability distribution function, or data cloud. A data-depth is a relative measure of how close is the given point in space to the center of a data cloud or a median of a (multivariate) probability distribution function. We discuss technical results of data-depths in the current context in Sect. 4 below. The crucial component of obtaining the depth of a given point with respect to a cloud of observations is to project the observations *in a single direction*, which is extremely fast and easy, apart from being an simple parallel procedure.

One immediate application of the DCW and the related data-depth algorithm is in supervised learning, presented in Sect. 5. Owing to the speed and efficiency of the DCW and data-depth algorithms, such classification of observations can be carried out extremely quickly, and the proposed **geometric learning** procedure may be used for *online supervised learning*. Thus, this can be adapted for a real-time analytics tool for classification in big data. Moreover, since we do not make assumptions about the geometry of the description of the data cloud $\mathbf{X}_k$ for any k, the proposed procedure is *robust* against failures of statistical assumption. Note that most statistical assumptions are essentially unverifiable declarations in high-dimensional data, hence such robustness properties are essential.

Apart from being fast and robust, the geometric learning procedure is surprisingly versatile and efficient. In the different datasets we have analysed, some of which are presented below, it seems that the proposed procedure is competitive, if not better, than standard supervised learning methods that are in popular usage. Note however, our goal here is to obtain (i) the shape of the data cloud, and (ii) fast classification without encumbering assumptions, and we do not claim to have an algorithm that will be "most accurate always". It nevertheless turns out that the proposed methodology that is typically hundreds or thousands of times faster than, say, random forest or support vector machine-based algorithm, we generally have a comparably high classification accuracy.

Results are presented in Sect. 7 for some simulated data examples, and in Sect. 8 for several real data examples. We conclude this paper with Sect. 9, where we present some caveats about using geometric learning, and some future research directions.

2 The Projection Quantile

We denote the open unit ball in p-dimensional Euclidean plane as $\mathcal{B}_p = \{x \in \mathbb{R}^p : ||x|| < 1\}$. The notation $||\mathbf{a}||$ stands for the Euclidean norm of a vector $\mathbf{a}$, while $\langle \mathbf{a}, \mathbf{b} \rangle$ stands for the Euclidean inner product between two vectors. For convenience, we reserve the notation $\mathbf{0}$ for a vector of zeroes, and $\mathbf{1}$ for a vector of ones, in appropriate dimensions that will be specified in the right contexts.

Also, we reserve the notation $\mathbf{u}$ to denote a typical element in this open unit ball. We further reserve the notation $\mathbf{e_u}$ for the unit vector in the direction of $\mathbf{u} \in \mathcal{B}^p$. Thus, $\mathbf{e_u} = \frac{\mathbf{u}}{||\mathbf{u}||}$ when $\mathbf{u} \neq \mathbf{0} \in \mathbb{R}^p$ and $\mathbf{0}$ otherwise. For any vector $x \in \mathbb{R}^p$, we define $x_\mathbf{u} = \langle x, \mathbf{e_u} \rangle$. The projection of x in the direction of $\mathbf{u}$ is, $x_\mathbf{u}\mathbf{e_u} = ||\mathbf{u}||^{-2}\langle \mathbf{x}, \mathbf{u} \rangle \mathbf{u}$.

Let $X \in \mathbb{R}^p$ be a random variable in p-dimensional Euclidean space. For the moment, assume that the center of the distribution of X is the origin. Let, $\mathbf{q_u}$ be the $(1 + ||\mathbf{u}||)/2$-th quantile of $X_\mathbf{u}$, that is, $\mathbb{P}[X_\mathbf{u} \leq \mathbf{q_u}] = (1 + ||\mathbf{u}||)/2$. The $\mathbf{u}$-th projection quantile (PQ) is defined in [9] as

$$Q_{proj}(\mathbf{u}) = \mathbf{q_u} \frac{\mathbf{u}}{||\mathbf{u}||} = \mathbf{q_u}\mathbf{e_u}. \tag{1}$$

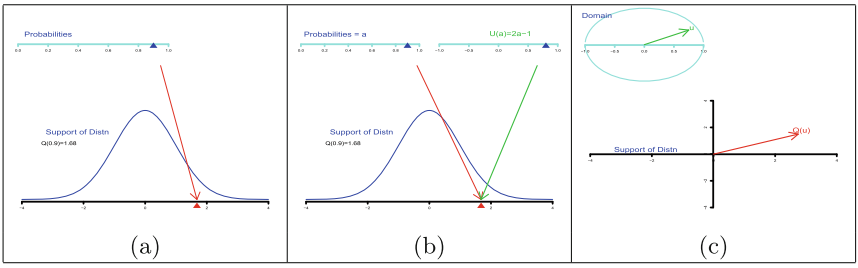

Fig. 2. A graphical depiction of the quantile function in one and two dimensions

The PQ (projection quantile) has several interesting properties, which makes it attractive from both theoretical and algorithmic points of view. It is linearly dependent on the number of dimensions p in calculation of $X_\mathbf{u}$. The sample PQ computation is linear in n also. Additionally, it can be easily seen that the computation of projection quantiles in different directions are unrelated to each other, and can be trivially distributed over a network of computing cores.

We present a brief motivation of the above PQ here, by using the illustrative example of univariate and bivariate Gaussian random variables. Note that for a real random variable X, the quantile function is defined on the interval $[0, 1]$ of probabilities and has as its range as the support of the random variable, and is traditionally defined as $Q(a) = \inf\{q : \mathbb{P}[X \leq q] \geq a\}$ for any $a \in [0, 1]$. For the standard Gaussian distribution, this is illustrated in the left panel of Fig. 2. Note, however, the following is also true [3,4]:

Theorem 1. *The a^{th} quantile is the smallest minimizer of the function* $\mathbb{E}$ $[|X - q| + (2a - 1)(X - q)]$.

Existence and uniqueness of $Q(a)$ is not an issue, owing to convexity of the criterion function, and hereafter we assume adequate conditions to ensure that in the *population*, the above convex criterion function has a unique minimizer. Assuming that the random variable X is absolutely continuous is sufficient for this

purpose, and hereafter we assume all feature vectors are absolutely continuous random variables.

In view of the above, we may alternatively define the quantile function as being indexed by $u = 2a - 1 \in [-1, 1]$, and $Q(u)$ as the (unique) minimizer of the convex function $\mathbb{E}\left[|X - q| + u(X - q)\right]$, as illustrated by the middle panel in Fig. 2. This definition of a quantile function was extended by [3] for p-dimensional random variables as being indexed by vectors $\mathbf{u} \in \mathcal{B}_p = \{x \in \mathbb{R}^p : ||x|| < 1\}$, and defined as minimizers $Q(\mathbf{u})$ of $\mathbb{E}\left[||X - q||+ < \mathbf{u}, X - q >\right]$, as illustrated in the right panel of Fig. 2. This is a generalization of one of the earliest attempts at defining multivariate median by [7]. Note that Chaudhuri's multivariate quantiles cannot be computed for $p > n$ using the algorithm given in [3], and requires iterative methods for even $p \leq n$. Additionally, it was seem that (a) this definition of multivariate quantiles does not capture the data geometry adequately, and (b) $Q(\mathbf{u})$ and $\mathbf{u}$ were nearly parallel in several simulated data examples. These observations motivate the projection quantile, where instead of using the full Euclidean norm of $X - q$, we only use that part of $X - q$ that is parallel to $\mathbf{u}$. Some amount of algebra reduces this procedure to the description of PQ provided above, and Fig. 1 shows its efficacy.

3 The Data Cloud Wrapper

The PQ described above does not fully capture the shape of the data geometry, mainly because of two issues. First, the spread of the data in different directions $\mathbf{e_u}$ from the center is different, and PQ does not accomodate for that. Second, all information related to any feature vector X_i in the directions orthogonal to $\mathbf{e_u}$ is discarded. The *data cloud wrapper* (DCW) algorithm attempts to correct these two discrepancies in the PQ, by introducing two weight functions. First, we adjust the *direction specific scaling* using $w_\mathbf{u}$ defined below. Then, we incorporate the information from the i^{th} observation in the directions orthogonal to $\mathbf{e_u}$ using another weight factor w_{2i}, also detailed below.

Recall that in accordance with the notation developed earlier, $X_{\mathbf{u}i}\mathbf{e_u}$ is the projection of X_i along $\mathbf{e_u}$. After centering (at the co-ordinatewise median) and scaling (using the median absolute deviation) the data, we first compute $Q_{proj}(\mathbf{u})$, the projection quantile along $\mathbf{u}$. We then compute global weights for the direction vector $\mathbf{u}$ by k-mean distance. Define d_i is the Euclidean distance of X_i from $Q_{proj}(\mathbf{u})$, and $d_{(1)} < \ldots < d_{(n)}$ are the ordered distances. We then define the k-mean distance as $\bar{d}_k = \frac{1}{n}\sum_{i=1}^{n} d_i \mathbb{I}_{\{d_i < d_{(k)}\}}$. Here, k is a tuning parameter that we choose depending on the application. We then define $w_\mathbf{u} = \exp(-a\bar{d}_k)$ as a scaling factor to be used in the direction $\mathbf{e_u}$. Our next step is to compute the norms of the vectors $||X_{\mathbf{u}\perp i}|| = ||X_i - X_{\mathbf{u}i}\mathbf{e_u}||$, which we use in the weight function $w_{2i} = \exp\left[-b\frac{||X_{\mathbf{u}\perp i}||}{||X_i||}\right]\mathbb{I}_{\{||X_{\mathbf{u}\perp i}|| \leq \epsilon\}}$. Here, b and ϵ are tuning parameters. Suppose $\{j_1, \ldots, j_{n_u}\}$ are the indices for which w_{2i} is non-zero. We now define $\tilde{X}_{\mathbf{u}j_k} = w_\mathbf{u} w_{2j_k} X_{\mathbf{u}j_k}$, for $k = 1, \ldots, n_u$. The DCW in the direction $\mathbf{e_u}$ is obtained by selecting the $\alpha = (1 + ||\mathbf{u}||)/2$-th quantile of $\tilde{X}_{\mathbf{u}j_1}, \ldots, \tilde{X}_{\mathbf{u}j_{n_u}}$. Let it be $\tilde{q}_\mathbf{u}$. The DCW is the direction $\mathbf{e_u}$ is defined as $\tilde{Q}_{proj}(\mathbf{u}) = \tilde{q}_\mathbf{u}\mathbf{e_u}$.

In order to state the theoretical properties of the DCW, first define $\tilde{X}_{\mathbf{u}i} = w_{\mathbf{u}} w_{2j_k} X_{\mathbf{u}i}$, for $i = 1, \ldots, n$. Also, consider the following two functions

$$\Psi_{\mathbf{u}}(X, q) = \mathbb{I}_{\{||X_{\mathbf{u}\perp i}||\leq\epsilon\}} \left[|\tilde{X}_{\mathbf{u}i} - q| + ||\mathbf{u}||(\tilde{X}_{\mathbf{u}i} - q) \right],$$
$$g_{\mathbf{u}}(X, q) = \mathbb{I}_{\{||X_{\mathbf{u}\perp i}||\leq\epsilon\}} \left[\left(2\mathbb{I}_{\{\tilde{X}_{\mathbf{u}i}\leq q\}} - 1 \right) - ||\mathbf{u}|| \right].$$

Our results are based on the *population level* properties of the functions $\Psi_{\mathbf{u}}(X, q)$ and $g_{\mathbf{u}}(X, q)$, that is, their behavior when we take an expectation of these functions with respect to the measure extended by X. Such properties are not assumed for sample level functions. An extremely easy example where population and sample values differ may be seen in the context of a Binomial (n, θ) random variable Z. Note that the expectation of Z/n is θ, which is a smooth function on $(0, 1)$. However, the sample expectation, i.e., the same functional computed under the empirical distribution function, is just Z/n, which is supported only on discretely many values, and is not a smooth function.

We assume that $\mathbb{E}\Psi_{\mathbf{u}}(X, q)$ is finite for all potential choices of q, and has a unique minimizer, which we call $q^*_{\mathbf{u}}$. This merely states that there is a unique population parameter to estimate. The sample version does not require uniqueness, but that may be enforced, as is traditionally done, by defining the minimizer to be the infimum over all possible values at which the minimum is reached. In this framework, we have the following results:

Theorem 2. *The sample DCW is a consistent estimator of the population DCW, that is $q_{\mathbf{u}} \to q^*_{\mathbf{u}}$ almost surely as sample size $n \to \infty$.*

Theorem 3. *Under the additional* **population level** *conditions that $\mathbb{E}g^2_{\mathbf{u}}(X, q^*_{\mathbf{u}}) < \infty$, and that the function $\mathbb{E}\Psi_{\mathbf{u}}(X, q)$ is twice continuously differentiable at $q^*_{\mathbf{u}}$ with the second derivative H being positive definite, then as $n \to \infty$*

$$n^{1/2}(q_{\mathbf{u}} - q^*_{\mathbf{u}}) = -n^{-1/2} H^{-1} S_n + o_P(1),$$

where $S_n = \sum_{i=1}^n g_{\mathbf{u}}(X_i, q^*_{\mathbf{u}})$. This implies, in particular, that $n^{1/2}(q^*_{\mathbf{u}} - q^*_{\mathbf{u}})$ is asymptotically Normal, with asymptotic variance $H^{-1}VH^{-1}$ where we have $V = \mathrm{V}ar\ g_{\mathbf{u}}(X, q^*_{\mathbf{u}})$.

The proofs of these results, and other theorems that follow, require considerable mathematical details. We present a very brief sketch of the main line of argument in the supplementary material, and the details can be made available as needed.

4 Data Depth Using the DCW

We consider the support of the feature vector, $\mathcal{X}$ to be a convex set in $\mathbb{R}^p$. For any given point $\mathbf{p} \in \mathcal{X} \setminus \{\mathbf{0}\}$ the support of a feature of an absolutely continuous random vector X with cumulative distribution function F, we define the *data-depth* as $D(\mathbf{p}, F) = \exp(-\alpha_{\mathbf{p}})$, where $\mathbf{u} = \alpha_{\mathbf{p}}\mathbf{p}/||\mathbf{p}||$, and $\tilde{Q}_{proj}(\mathbf{u}) = \mathbf{p}$. We

extend this to the point $\mathbf{p} = \mathbf{0}$ by defining $D(\mathbf{0}, F) = 1$. This essentially means that $\alpha_{\mathbf{p}} \in (0, 1]$ is the norm of $\mathbf{u}$, which has the same direction as $\mathbf{p}$, such that $\mathbf{u}^{th}$ DCW is exactly $\mathbf{p}$.

The following properties are available for the data depth function $D(\mathbf{p}, F)$:

Theorem 4. *1.* $D(\mathbf{p}, F) = 1$ *if and only if* $\mathbf{p} = \mathbf{0}$.
2. $D(t\mathbf{p}, F) \leq D(\mathbf{p}, F)$ *for all* $\mathbf{p} \in \mathcal{X}$, *and all* $t \in [0, 1]$.
3. The directional derivative of $D(\mathbf{p}, F)$ *with respect to* $\mathbf{p}$ *exists.*
4. The depth function $D(\mathbf{p}, F)$ *is smooth in te second argument, in the sense that the Gateux derivative exists.*
5. $D(\mathbf{p}, F) \to 0$ *as* $||\mathbf{p}|| \to \infty$.

Note that the first two properties are the essential properties of a data-depth, while the rest of the results are technical properties that help understand the depth function better.

5 Geometric Classification Technique

One immediate application of the DCW and the related data-depth algorithm is in supervised learning. Consider a feature vector $X \in \mathcal{X} \subseteq \mathbb{R}^p$ for some (potentially high) dimension p, associated with a label $Y \in \mathcal{Y} = \{0, 1, \ldots, K-1\}$. Assumed that the observed data is a random sample $\{(X_i, Y_i) \in \mathcal{X} \times \mathcal{Y} \subseteq \mathbb{R}^p \times \{0, 1, \ldots, K-1\}\}$ of such (X, Y) pairs. Here K, the total number of labels, is assumed known. Without loss of generality, we assume that observations indexed by $S_k = \{i_{k1}, \ldots, i_{kn_k}\}$ share the common label $Y_{i_{kj}} = k$, for any $k \in \mathcal{Y} = \{0, \ldots, K-1\}$. We assume that $\cup_{k=0}^{K-1} S_k = \{1, \ldots, n\}$, and $S_k \cap S_{\tilde{k}} = \varnothing$ whenever $k \neq \tilde{k}$. We denote the n_k feature observations corresponding to S_k by $\mathbf{X}_k = (X_{i_{k1}}, \ldots, X_{i_{kn_k}})$.

We elicit the label of any new or unlabeled feature vector $x \in \mathcal{X}$ by computing its depths with respect to the K different data clouds of observations $\mathbf{X}_k = (X_{i_{k1}}, \ldots, X_{i_{kn_k}})$. We may then choose the label that corresponds to the highest depth value, or construct labels using more complex usage of the K depth values obtained for x. For example, one alternative to simply choosing the highest-depth label would be to not classify an unlabeled observation if the maximum depth is below a threshold, thus paving the way for potentially extending this algorithm to unsupervised and semi-supervised classification problems, which we do not pursue here.

Notice that the geometric learning-based separating boundary between two labeled classes in the feature space is essentially the locus of the points where the data-depths are equal from both the classes. Thus, the separating curves between labeled classes are essentially *isodepth curves*. This notion can be extended to multiple labeled classes easily, and is of independent interest.

Nevertheless, note that in order to classify a new or unlabeled observation, it is not necessary to obtain the entire isodepth curves. The only computation that needs to be performed is to obtain the data-depth *in the direction of the unlabeled observation* relative to each labeled class, which is trivially a parallelizable

procedure requiring at most n one-dimensional projections and few other simple computations. This leads to the geometric learning algorithm being very much amenable to active learning as well as online learning of class labels.

It may be noted that existing techniques for supervised learning either impose shape restrictions explicitly or tacitly, or suffer from curse of dimensionality, or are slow, sequential procedures requiring multiple passes through the data. Many methods of supervised learning suffer from multiple of these issues. For example, methods like logistic regression for two or more class labels, as well as linear or quadratic discriminant analysis explicitly make parametric statistical assumptions, which imply strong restrictions on the shape of the allowed data layout. The classical version of these algorithms are inapplicable for data in high dimensions, and require additional assumptions, typically of sparsity of some functional of the feature vector distributions, for usability in big data. Some of these additional assumptions are unverifiable in data. Nearest neighbor rules implicitly make similar assumptions with a choice of metric and tuning parameters, while support vector machines make an implicit choice of allowable geometry using the kernel function. Methods like (multivariate) density estimation and subsequent learning procedures suffer from the curse of dimensionality. Decision trees and ensemble-based methods like classification and regression trees, bagging, random decision forests, boosting require iterative and often sequential computation, and may impose shape restrictions on the data and may suffer from curse of dimensionality depending on the algorithmic details.

6 Data Geometric Feature Selection

The key benefit of the weighted projection quantile (WPQ) and *geometric learning algorithm* (GLA)-based classification is in high dimensional data. However, the application of the algorithm also provides us high accuracy of classification when applied in conjunction with a suitable feature selection algorithm. To retain the speed benefits of the GLA algorithm, it is important that the feature selection can also be done using a quick and fast method. We have used a very simple and quick algorithm for feature selection which we describe in details below.

A very simple feature selection can be done using how well individual features are correlated with the classes. Using a Spearman's correlation can provide us with a very basic and quick feature ranking. However, the problem with this method is that non-monotonic relations of features with the response.

Another method could be constructed based on a measure of distance of distributions of a feature in each class. If a feature is informative, then the distribution of the feature between the classes will have some separation or distance. On the other hand, if a feature is not informative at all, then the distributions would not be substantially separated. We can use the Kolmogorov-Smirnov test statistic $D_x = sup\|F_1(x) - F_0(x)\|$ for feature x for a two class $(1, 0)$ response. The Spearman's correlation and the Kolmogorov-Smirnov test statistic would be significantly correlated. However, Kolmogorov-Smirnov statistic D would be

more sensitive to non-monotonic relations than the Spearman's correlation. This can be illustrated by Fig. 3 comprising of a the graph of a highly informative feature and a relatively non-informative feature from the Dexter dataset from the UCI repository, discussed in details later.

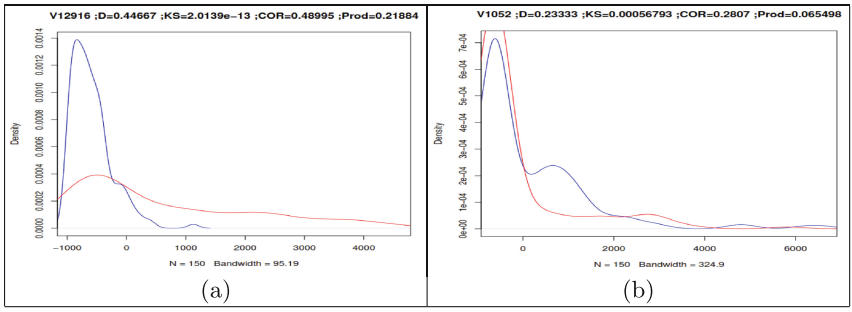

Fig. 3. Plot of two features from the Dexter dataset. The left panel (a) represents a highly informative feature, while the right panel (b) represents a low information feature. Our feature selection protocol quantifies this.

The third scheme of feature ranking can be a constructed using a combination of the two. We used a product of the Kolmogorov-Smirnov distance $\in (0, 1)$ with the Spearman's correlation coefficient $\in (0, 1)$. We ranked the features in decreasing order based on the product of the Spearman's correlation coefficient and the Kolmogorov-Smirnov test statistic D. Usually a combination of the top few features provide the best predictive ability. So at the second stage we chose a p fraction of features. If there are k features we chose pk features and the response, and constructed a precision matrix, or inverse covariance matrix, of these $pk + 1$ variables together using generalized lasso. Based on the strength of relationship with the response, we re-ranked the features based on the generalized lasso.

Based on the final feature ranking we started with first feature and computed the predictive accuracy on the validation set. Then we progressively included features and kept the feature in the final feature set or deselected a feature based on the increased in prediction accuracy on the validation set. If inclusion of a feature increased the predictive accuracy on the validation set then we kept that feature in the list of selected feature or else we deselected the feature. We continued to do this until the increase in prediction accuracy by inclusion of further features did not benefit the classification.

7 Geometric Learning Example: Simulated Data

In this section we use two simulated datasets to demonstrate the key characteristics of the proposed geometry learning algorithm, and compare its performance with state-of-the-art alternatives.

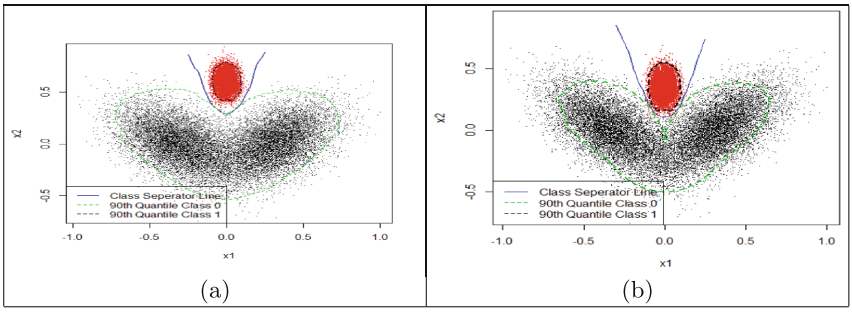

Fig. 4. Isodepth separation curves for a binary classification problem in two datasets

The simulated data contains two classes, and is depicted in Figure 4 with black and red points. We present the two simulated datasets in this figure, in the two panels. In both panels, the black points are clustered in the apple shaped figure, while the red points are in the top leaf cluster. The dotted lines in green and black are the DCW curves corresponding to the black and red observations respectively, corresponding to $||\mathbf{u}|| = 0.9$. The solid blue curve indicates the isodepth curve for this dataset, which acts as the curve of separation for the two groups of data. In order to evaluate the classification accuracy and speed of our algorithm, we randomly selected 80 % of the observations from each group for training, and evaluated the geometric learning algorithm on the rest 20 % test data from each group. For comparison purposes, we also used several standard classification algorithms on identical training and testing datasets. This process was repeated 1000 times to ensure sufficient randomization. All results from this section and the next section are based on 1000 randomization runs as described above.

As can be seen from Table 1, the geometric learning algorithm is comparable or better in terms of classification accuracy and speed compared to the state-of-the-art methods. In fact, it is considerably fast in comparison to the other algorithms which achieve similar levels of accuracy in different datasets, of which we have chosen two illustrative examples. In particular, the geometric learning technique is considerably faster than random forest and support vector machines (SVM). In general, parametric methods like linear or quadratic discriminant analysis (LDA, QDA), and logistic regression, or procedures like neural networks do not achieve a good class separator, especially in dataset (b) where the classes are not as distinct as in case of dataset (a).

We used the standard packages and routines in the open source software `R`, namely `glm, lda (MASS), qda (MASS), randomForest, svm (e1071), nnet, knn (class)` for the existing supervised learning methods reported here. In the random forest procedure, the number of trees for each run was kept at 500, sampling was done with replacement, and the rest of the parameters were run with default setting. Radial basis kernel ($= \exp(-\gamma|u - v|^2)$) was used for the SVM fit of type *C-Classification* using a scaling of 1 and a class weight

equal to the proportion of observation in each class in the train set. For neural nets, the size of the hidden layer was set at 2, case-wise sample weights were set at 1, and entropy fit was used. Note that the random forest algorithm has an inbuilt feature selection capability. Hence, without feature selection random forest tends to perform much better than other comparable algorithms. More details on the classification techniques used for our analysis, and detailed graphical results from this simulation study may found in the supplementary material. Algorithmic and statistical details of these learning techniques can be found in [8]. The standard error of classification accuracy results under any method may be estimated using the relation $s.e. = \sqrt{a(1-a)/1000}$ where is the proportion of accurate classifications, since the results are based on 1000 randomization runs.

8 Performance of the Algorithm on Standard Datasets

We show the performance of the dataset on multiclass multivariate datasets, namely the celebrated Fisher's iris dataset, the colon dataset from the `R` package `cepp`, Arcene and Dexter from the UCI Machine Learning dataset library [2] available at https://archive.ics.uci.edu/ml/datasets.html.

8.1 Fisher's Iris Dataset

This is perhaps the best known dataset to be found in the pattern recognition, learning and statistical classification literature. Performance on this dataset is a litmus test for any new proposed machine learning method. The data contains three classes of Iris plants of 50 instances each. The three classes are *Iris setosa*, *Iris versicolor* and *Iris virginica*. There are four features related to the Iris plant and flower in this dataset: sepal length, sepal width, petal length and petal width, all in centimeters. We use the same methods and techniques as in the simulated data analysis discussed above.

In Table 2 we present the average classification accuracy in the test data sets from the Iris data. The running time of all the algorithms are very similar since the dataset is small with 150 observations and 4 predictors. In terms of performance, while all methods perform well, the proposed geometric learning algorithm is the best.

8.2 The Colon Cancer Dataset

The colon dataset is a publicly available dataset for $n = 62$ individuals related to colon cancer. This dataset was generated using Affymetrix oligonucleotide arrays, and contains expressions levels for 40 tumor and 22 normal colon tissues. Out of the originally measured 6500 human genes, $p = 2000$ with the highest minimal intensity across the tissues are selected for classification purposes. Each score represents a gene intensity derived in a process described in [1]. Note that in this case dimension p is considerably higher than sample size n, and thus represents a high dimensional supervised learning problem (Table 3).

It can be seen that the geometric learning algorithm is the most successful among the supervised learning methodologies that were usable for this dataset, both in terms of prediction accuracy, as well as in terms of speed. Methods not reported here were not applicable without additional unverifiable technical assumptions. Support vector machines perform marginally better in terms of accuracy, but requires an 80-fold increase in computing time.

8.3 Arcene Dataset

The Arcene dataset consists of mass spectrometry data obtained with the SELDI technique, and is accessed from the UCI repository [2]. The data contains data from both cancer patients (ovarian or prostate) or healthy individuals. The classification task is to identify the cancer tissues. The data consists of 900 observations and 10000 attributes. Out of these 10000 attributes 7000 real attributes and 3000 artificial random probes.

We use this data to illustrate two things: the speed of the proposed geometric learning algorithm, as well as the efficacy of the proposed fast feature selection procedure. First, we present in Table 4 the results for the geometric learning and several other algorithms, when no feature selection is performed a priori. Notice that the proposed GLA takes negligible amount of time compared to random forest or support vector machine (SVM)-based methodology. It achieves a 82 % accuracy, which is marginally lower than SVM at 84 %, and somewhat lower than 89 % achieved by random forest method which is about 4550-times slower.

We next performed the fast feature selection as discussed in Sect. 6. With the feature selection we were able to improve the accuracy of classification to 0.92 with a balanced error rate (BER) of 0.08603 with 9 features out of 10000. Balanced error rate (BER) is defined as the mean prediction error in all classes. This is better performance the accuracy achieved for the NIPS feature selection competition from which this data originated [5,6]. The mean BER obtained at the NIPS challenge is 0.119 ± 0.012 and the best BER obtained was 0.10. We ran some of the standard classification algorithms, and the results are presented in Table 5. Once a small number of features are selected, classical methods like naive Bayes (discriminant analysis) and generalized linear model (GLM), of which logistic regression is a special case, become viable, and are reported here. This table shows that after feature selection, the proposed method is about three times faster than the sate-of-the-art random forest method, and achieves greater accuracy. None of the classical methods perform as well.

8.4 Dexter Dataset

The Dexter dataset is a text classification dataset with a bag-of-word representation, and is also available at the UCI repository [2]. This is a two class classification problem based on corporate acquisition text collected from Reuters news items. The dataset consists of 20000 attributes. Out of these 20000 attributes, real attributes are 9947 and the rest of the 10053 attributes are random probes for the NIPS feature selection and classification contest. The train set consists

Table 1. Comparison of several supervised learning algorithms on the simulated example, in randomly selected test sets. The classification accuracy is the average proportion of test sets observations correctly classified.

	Data (a)		Data (b)	
Method	Accuracy	Run time	Accuracy	Run time
Geometric learning	0.996	1.46	0.976	1.48
Logistic regression	0.981	0.28	0.901	0.31
LDA	0.969	0.22	0.881	0.20
DA	0.992	0.27	0.974	0.21
Random forest (500 trees)	0.998	23.61	0.976	31.77
Neural network	0.967	4.76	0.962	3.56
SVM	0.984	6.45	0.974	8.54

Table 2. Performance of different classification algorithms on the Fisher iris dataset

Method	Accuracy
Geometric learning	0.9733
LDA	0.9600
QDA	0.9667
Random forest (500 trees)	0.9667
Neural network	0.9267
SVM	0.9533

Table 3. Classification results in the colon cancer data

Method	# Mis-classified	Accuracy	Run time
Geometric learning	9	0.854	0.21
Random forest (500 trees)	10	0.839	226.92
LDA	15	0.758	45.12
SVM	8	0.871	17.36

Table 4. Arcene classification without feature selection

Method	CPU time	Accuracy
Geometric learning	3.67	0.825
Random forest	16714.20	0.895
SVM	966.86	0.842

Table 5. Arcene classification output with feature selection

Method	CPU time	Accuracy
Geometric learning	0.32	0.92
Random forest	0.98	0.90
SVM	0.33	0.91
Naive bayes	0.28	0.89
GLM	0.35	0.74

Table 6. Dexter classification output

Method	CPU time	Accuracy
Geometric learning	0.43	0.89
Random forest	1.23	0.91
SVM	0.51	0.90
Naive bayes	0.37	0.86
GLM	0.35	0.69

of 300 observations and the validation set consists of another 300 observations. With feature selection we achieved a best classification accuracy of 0.89 with a BER of 0.116 which is also comparable to the NIPS performance. The best accuracy was achieved using top 75 features. Results from this analysis is presented in Table 6. The results show that the proposed geometric learning algorithm is again one of the fastest methods, which achieves nearly comparable accuracy with the best existing techniques.

9 Conclusions and Future Directions

The geometric learning algorithm can be seen to be very versatile, and applicable in complicated supervised learning problems, as well as in high dimensions. One future research to pursue is on theoretical quantification of its classification error bound. Another important consideration in this line of research is to evaluate its performance in unsupervised learning problems.

The illustrative examples of the previous section demonstrates several things. First, the proposed procedure captures the geometry of the data reasonably well. Second, owing to its geometric properties, the speed of the algorithm is not particularly affected by the dimensionality of the feature space. However, there is plenty of scope of parallelization, both in dimensions as well as in sample size. Third, our method is robust against lack of assumptions, and seems to be fairly functional even without feature selection in many examples. Naturally, feature selection improves performance, but the rough and fast feature selection procedure suggested here seems adequate, showing additional robustness. In all

examples, the geometric learning algorithm is either the best in terms of accuracy, or reasonably close to the best, this showing the speed and robustness does not drastically compromise its efficiency.

Some of the limitations of the geometric learning method arise from the topological restrictions it imposes on the data corresponding to any labeled class. While such restrictions are minimal, it is nevertheless unlikely to succeed in classification problems where the data cloud for any class consists of more than one connected component, or have genus greater than zero. We can envisage how to extend the geometric learning algorithm to address these kind of data features, but nevertheless additional research needs to be carried out.

Acknowledgements. This research is partially supported by NSF grant # IIS-1029711, NASA grant #-1502546) the Institute on the Environment (IonE), and College of Liberal Arts (CLA) at the University of Minnesota.

References

1. Alon, A., et al.: Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. Proc. Natl. Acad. Sci. USA **96**, 6745–6750 (1999)
2. Bache, K., Lichman, M.: UCI machine learning repository (2013)
3. Chaudhuri, P.: On a geometric notion of quantiles for multivariate data. J. Am. Stat. Assoc. **91**, 862–872 (1996)
4. Ferguson, T.S.: Mathematical Statistics. A Decision Theoretic Approach. Academic Press, New York (1967)
5. Guyon, I., et al.: Feature selection with the CLOP package. Technical report (2006)
6. Guyon, I., et al.: Competitive baseline methods set new standards for the NIPS 2003 feature selection benchmark. Pattern Recogn. Lett. **28**, 1438–1444 (2007)
7. Haldane, J.B.S.: Note on the median of a multivariate distribution. Biometrika **35**, 414–415 (1948)
8. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, New York (2009)
9. Mukhopadhyay, N., Chatterjee, S.B.: High dimensional data analysis using multivariate generalized spatial quantiles. J. Mult. Anal. **102–4**, 768–780 (2011)