mathematics

MDPI

Article

Using Locality-Sensitive Hashing for SVM Classification of Large Data Sets

Maria D. Gonzalez-Lima [1,*,†] and Carenne C. Ludeña [2,†]

[1] Departamento de Matemáticas y Estadística, Universidad del Norte, Barranquilla 081007, Colombia
[2] Matrix CPM Solutions, Crr 15 93A 84, Bogotá 110221, Colombia; carinludena@gmail.com
* Correspondence: limad@uninorte.edu.co
† These authors contributed equally to this work.

Abstract: We propose a novel method using Locality-Sensitive Hashing (LSH) for solving the optimization problem that arises in the training stage of support vector machines for large data sets, possibly in high dimensions. LSH was introduced as an efficient way to look for neighbors in high dimensional spaces. Random projections-based LSH functions create bins so that when great probability points belonging to the same bin are close, the points that are far will not be in the same bin. Based on these bins, it is not necessary to consider the whole original set but representatives in each one of them, thus reducing the effective size of the data set. A key of our proposal is that we work with the feature space and use only the projections to search for closeness in this space. Moreover, instead of choosing the projection directions at random, we sample a small subset and solve the associated SVM problem. Projections in this direction allows for a more precise sample in many cases and an approximation of the solution of the large problem is found in a fraction of the running time with small degradation of the classification error. We present two algorithms, theoretical support, and numerical experiments showing their performances on real life problems taken from the LIBSVM data base.

Keywords: support vector machines; locality sensitive hashing; classification problems

MSC: 90-08; 60-08

Citation: Gonzalez-Lima, M.D.; Ludeña, C.C. Using Locality-Sensitive Hashing for SVM Classification of Large Data Sets. *Mathematics* **2022**, *10*, 1812. https://doi.org/10.3390/math10111812

Academic Editor: Liangxiao Jiang

Received: 30 April 2022
Accepted: 17 May 2022
Published: 25 May 2022

1. Introduction

In this work, we deal with the problem of binary classification of large volume data sets of possible high dimension (for instance, graphs and texts). There are many techniques for solving this problem, for example, logistic regression and other linear methods, random forests and ensemble methods in general, K-nearest neighbors or neural network-deep learning models, see for example [1] for a general survey; however, one of the difficulties of the classification problem is to be able to represent high dimensional points via an appropriate embedding in a feature space. Methods based on kernel functions such as Support Vector Machines (SVM) are an interesting alternative for the classification problem because they aim exactly at this, supported by the statistical theory developed by [2]. They are based on finding a hyperplane of maximal margin that separates the data. A very appealing feature of SVMs is that the separation can be achieved in the original data space or in a higher dimensional one, via kernel functions, without explicitly forming the space transformation map [3], unlike what most algorithms do. This feature also makes it possible to apply the method in non-vectorial domains [4]. To construct the separating hyperplane, SVM only needs to identify some significant vectors, called support vectors, from the whole data set. Because of this, SVM may suffer less the effect of outliers than other techniques. With well-tuned hyperparameters, SVMs have shown to be a robust and well-behaved technique for classification in many real-world problems. We refer to the web page http://www.clopinet.com/isabelle/Projects/SVM/applist.html (accessed on 28 April 2022) for a comprehensive list of applications.

A drawback of SVM for the classification of very large data sets is the computational cost of the optimization problem to be solved in the training stage. There are a large amount of contributions in the literature that deal with this problem and several courses of action have been developed. Following the seminal work by Osuna, Girosi, and Freund [5,6], decomposition techniques based on the optimality conditions are used to find the solution of the optimization problem by solving a sequence of smaller subproblems of the same structure of the original one. Efficient methods, as well as different heuristics, have been proposed for stating and solving the optimization subproblems, as SMO [7] which gave rise to the broadly used LIBSVM [8] (that decomposes the large problem into small problems of size two). Algorithms that decompose the large optimization problem into a sequence of medium size problems include SVM$^{\text{light}}$ [9,10], GPDT [11–13], and ASL [14]. There are other approaches that do not use decomposition procedures but an efficient use of the linear algebra for some kernels [15], or identification procedures of the positive components at solution as the ones developed in [16]. All these methods strive at finding the exact solution of the SVM optimization problem by considering, in different ways, subsets of the training data set.

Other methods, although also using subsets of the training data set, are of a different nature since they consider approximations of the optimization problem in an attempt to reduce the computational cost while still obtaining similar generalization errors. In this group, we can cite [17–19] that use low-rank approximations of the kernel matrix, and [20–22] that use random samples of the data set. The use of subsamples may be combined with techniques for reducing the feature space dimension as in [23] or combined with search ideas as the one proposed in [24], followed by the extension presented in [25], where approximated solutions of the large size SVM problems are found by looking at the smaller optimization subproblems resulting from random samples of the data, and enriching the subsamples by finding the k-nearest neighbors, in the complete data set, of the support vectors associated to them. For a revision of subsampling methodologies we refer to Nalepa and Kawulok [26] and Pardis et al. [27].

The results in [24] show that great computational advantages can be obtained by using randomness and proximity ideas in the context of SVM. This has to do with the nature of SVM where the main objective is to find the separating hyperplane. It seems then natural to consider approaches that include randomness along with the idea that only points near the actual support vectors are necessary in order to obtain a good fit over a sample, thus reducing the required sample size. One way of addressing this problem is to use projections over random directions, instead of random sampling, and then choosing a sample over the projections. This approach has the advantage that it is not necessary to consider all points for a given random direction, but rather only selected representatives. Motivated by this, in this paper we look for an adequate subset of representative points of the data via projections using Locality Sensitive Hashing (LSH) [28]. LSH has the advantage of transforming the data points into a lower dimensional representation space, see for example [29,30]. One of its main applications is the efficient search of (approximated) nearest neighbors. Since only the support vectors are needed to obtain the optimal hyperplane classifier, to use LSH to select a subset of points that may be support vectors, or close to them, is especially of great benefit for reducing the training computational time for SVM; however, LSH could be used jointly with other approaches as well.

Random projections create bins so that when great probability points belonging to the same bin are close, points that are far will not be in the same bin. Based on these bins, it is not necessary to consider the whole original set but representatives in each one of them, thus reducing the effective size of the data set. A key of our proposal is that we work with the feature space and use the projections to search for closeness in this space. This is also another reason for using LSH in the context of SVM. Moreover, instead of only choosing the projection directions at random, we also choose them by solving SVM very small problems. We call these projecting directions as "directed" because they already contain useful information of the large problem to be solved.

Results relating to kernel-based classifiers and LSH are rather recent. Research efforts are most importantly related to provably approximating the similarity structure given by the kernel in the SVM in order to reduce the time and memory space required to train the SVM as in [31], where authors show accuracy improvement over a series of benchmark data sets. Another approach [32,33] is related to enhancing the prediction stage by using hash functions. Instead of using the actual solution, the authors consider hashing both the sample points and the normal to the obtained separating hyperplane in order to optimize time and memory space.

Our main objective with this paper is to propose subsampling algorithms, based on LSH, that produce a good selection of the data set for using in SVM. The goal is to improve the computational cost without degrading the prediction error significantly. Our approaches exploit the underneath idea of proximity but without looking explicitly for neighborhoods. The numerical experimentation seeks to show the effectiveness and efficiency of our methodologies comparing with the whole data set. We show improvement in time and we support our numerical findings with theoretical results. Our next work will be focused on improving the implementations (by using the embarrassing parallel nature of our algorithms) and the computational environment so that much larger problems in very high dimensions can be solved.

The article is organized as follows. In Section 3 we introduce LSH and the algorithmic general frameworks that we propose for solving the complete SVM problem. The following section includes numerical experimentation and details on the implementations. Section 5 contains theoretical results giving bounds on the obtained errors of the algorithms.

We end in Section 6 with some concluding remarks and future work.

2. Preliminaries: The SVM Problem

SVM for binary classification (the one considered in this paper) is based on the following. Given points $\{X_i \in \mathbb{R}^d, \ \ i = 1, \ldots, n\}$ belonging to two classes (identified with the corresponding tags $y_i = 1$ or $y_i = -1$), they are linearly separable if there exists an hyperplane that divides them into the two different classes. The dimension d denotes the attributes of the data, and the input (or observation) space the set formed by the data. Among the separating hyperplanes, SVM seeks to find the one that maximizes the separation margin between classes, constrained to respecting the classification of each point of the data. This problem can be modeled, after a normalization, as the optimization problem

$$\begin{array}{ll} \underset{w,b}{\text{minimize}} & \frac{1}{2}\|w\|_2^2 \\ \text{subject to} & y_i(w^t X_i + b) \geq 1 \ \ \forall \ i = 1, \ldots, n. \end{array} \tag{1}$$

Here, $\|.\|_2$ denotes the Euclidean norm.

Because the data set is usually linearly nonseparable (that is, there does not exist a solution of problem (1)) two variants are introduced in the previous problem. On one hand, a perturbation variable ξ is included in order to relax the constraints, so that a margin of error in the classification is accepted. Additionally, since the data might be separable by a nonlinear decision surface, such a surface is computed by mapping the input variables into a higher dimensional feature space, and by working with linear classification in that space. In other words, let us denote $\mathcal{H}$ as the feature space, which is a reproducing kernel Hilbert set (RKHS). We denote the inner product in $\mathcal{H}$ with $< ., . >$. Then, $x \in \mathbb{R}^d$ is mapped into $\phi(x) \in \mathcal{H}$, where $\phi(.)$ is the transformation induced by the use of the kernel function K. This is, $K(z, a) =< \phi(z), \phi(a) >$ for every $z, a \in \mathbb{R}^d$. Thus, the input vectors X_i are substituted with the new "feature vectors" $\phi(X_i)$, belonging to the "feature space" $\mathcal{H}$. In this way, for the linearly nonseparable case, the optimization problem is written as

$$\begin{array}{ll} \underset{w,b,\xi}{\text{minimize}} & \frac{1}{2}\|w\|_{\mathcal{H}}^2 + C\sum_{i=1}^n \xi_i \\ \text{subject to} & y_i(< w, \phi(X_i) > +b) \geq 1 - \xi_i \ \ \forall \ i = 1, \ldots, n, \\ & \xi_i \geq 0 \ \ \forall \ i = 1, \ldots, n \end{array} \tag{2}$$

where C is a positive constant that penalizes the errors at the constraints, and $\|w\|^2_{\mathcal{H}} = < w, w >$.

Let us denote (w^*, b^*, ξ^*) the solution to this problem. A new point $X \in \mathbb{R}^d$ is classified according to the side of the "hyperplane" where $\phi(X)$ falls. Hence, the function used to classify a new point X can be written as

$$g^*(X) = \text{sign}(< w^*, \phi(X) > + b^*) \tag{3}$$

and is usually called the classification or generalization function.

In order to solve (2), standard duality theory may be used since the problem is convex and quadratic. For more details, we refer to the book by Cristianini and J. Shawe-Taylor [3]. Using this theory, (2) can be solved by solving its dual. Some advantages of using the dual problem is that an explicit description of ϕ above is not required but a function that preserves in the higher dimensional space $\mathcal{H}$ the properties of the inner product, and this is satisfied by the kernel function K. Then, the dimension of the feature space for the classification can be increased without increasing the dimension of the optimization problem to solve, and this is particularly relevant when dealing with an infinite dimensional space $\mathcal{H}$.

Following [3], the dual problem corresponding to (2) is given, in terms of the kernel function, by

$$\begin{array}{ll} \underset{\lambda}{\text{mimimize}} & -\sum_{i=1}^n \lambda_i + \frac{1}{2}\lambda^t Q \lambda \\ \text{subject to} & y^t \lambda = 0, \\ & 0 \leq \lambda_i \leq C \quad \text{for } i = 1, \ldots, n \end{array} \tag{4}$$

where $Q \in \mathbb{R}^{n \times n}$ is a symmetric positive semidefinite matrix with positive diagonal, defined as $Q_{ij} = y_i y_j K(X_i, X_j)$. The matrix K with ij-component equal to $K(X_i, X_j)$ is called the kernel matrix. To avoid complicating notation, typically K will stand for both the generic kernel and the matrix defined by the kernel restricted to the original data of size n.

Let λ^* be a solution of (4). Using the Karush–Kuhn–Tucker (KKT) optimality conditions (see [3]), we can obtain an expression of w^* as

$$w^* = \sum_{i=1}^n \lambda_i^* y_i \phi(X_i). \tag{5}$$

Observe that in this sum, only the $\lambda_i^* > 0$ are relevant. The corresponding X_i are the so-called support vectors and their importance falls in the fact that the remaining objects are irrelevant for classification purposes.

Complementarity slackness conditions [3] also imply that, for any i with $0 < \lambda_i^* < C$, one has $y_i(< w^*, \phi(X_i) > + b^*) = 1$, and therefore an expression for b^* can also be obtained as

$$b^* = 1 - \max_{\{y_j = 1, 0 < \lambda_j^* < C\}} < w^*, \phi(X_j) > . \tag{6}$$

Let us recall that our interest is to classify a new point by means of the generalization function g^* using λ^*. This can be achieved by substituting w^* in (3) and (6), so we obtain

$$g^*(X) = \text{sign}(\sum_{i=1}^n \lambda_i^* y_i K(X_i, X) + b^*) \tag{7}$$

with $b^* = 1 - \max_{\{y_j = 1, 0 < \lambda_j^* < C\}} \sum_{i=1}^n \lambda_i^* y_i K(X_i, X_j)$.

Therefore, the classification of a new point can be made just by selecting a (hopefully small) group of points from the large original data: the support vectors. The procedure of finding these vectors (which we denote by SV) from the given data set is usually referred to as the training process or training the machine. After training, it is customary to qualify the result by using it in order to classify points that are known to be in one class or another. This set of points is called the testing set. The estimated classification or prediction error for a given data set is the percentage of points from the test set that are incorrectly predicted. This last part of the SVM procedure is known as the fitting process. Note that the number of SV vectors impact the fitting time—the fewer, the faster.

In the following section, we introduce the algorithms proposed to solve an approximation of problem (4) by the use of locality-sensitive hashing and subsamples.

3. Using LSH for SVM

Locality-sensitive Hashing (LSH) was introduced as an efficient way to look for nearest neighbors in high dimensional spaces [28]. The idea is to hash the vectors in the space using several hash functions so that, for each one, the probability of collision is much higher for points that are close to each other than for those that are far apart. Then, LSH can be used to search approximated nearest neighbors of a given query point by retrieving elements stored in the same bin containing this point. Formally, the definition follows.

Definition 1. *(LSH functions) For a given $R > 0$ and probabilities $p1 > p2$, a family of functions belonging to the set $H = \{h : D \to \mathcal{N}\}$ where D is a metric space with metric $\tilde{d}$, and $\mathcal{N}$ the set of integers, are LSH if for each $\tilde{q}, q \in D$ and each $h \in H$ the followings are satisfied*

- *if $\tilde{d}(\tilde{q}, q) \leq R$ then $PrH[h(q) = h(\tilde{q})] \geq p1$,*
- *if $\tilde{d}(\tilde{q}, q) > R$ then $PrH[h(q) = h(\tilde{q})] \leq p2$.*

In this paper we are interested in the projection-based hash functions as presented in [34]. For any p dimensional vector v, define the maps $h_{\mathbf{a},\theta}(v) : \mathbb{R}^p \to \mathcal{N}$ indexed by a choice of a α-stable random vector $\mathbf{a}$ (see [34] for a definition) and a real number θ chosen uniformly from the range $[0, r]$ in the following way. For a fixed $\mathbf{a}, \theta$ the hash function $h_{\mathbf{a},\theta}$ is given by

$$h_{\mathbf{a},\theta}(v) = \left\lfloor \frac{\mathbf{a}^t v + \theta}{r} \right\rfloor. \tag{8}$$

Here, $\lfloor . \rfloor$ denotes the floor function.

In [34] is shown that the projection-based functions h, as previously defined, are LSH.

We use these hash functions in the feature space in order to find representatives of clusters for the data set used to train the SVM problem. We do this by projecting the data several times and choosing, as a representative, a random data point at each bin, after each projection. Because the projections are computationally expensive, we project only subsamples of the whole data set. In order to choose the direction $\mathbf{a}$ we follow two alternative procedures, random and directed projections, as described below.

One interesting feature of this procedure, although we do not include this in our numerical experiments, is that repetitions are independent, so the problem is embarrassingly parallel. Each parallel iteration constructs an independent subsample, which can then be joined to form the final subsample that is used to train the SVM.

3.1. Random Projections

Following [34], we choose the entries of $\mathbf{a}$ independently from a Gaussian distribution in the input space.

This leads to the following Algorithm 1.

Algorithm 1 LSH-SVM (random projections)

Given an initial kernel K, B bins, $\eta_1 \in (0,1)$ the percentage of subsample data points, N the number of projections repetitions, η_3 the cutoff percentage, and $\mathcal{S} = \{X_1, \ldots, X_n\}$ the data set of problem (4):

1. Take a random subsample $\mathcal{S}_1$ of $\mathcal{S}$ with size $n_1 = \lfloor \eta_1 n \rfloor$.
2. Generate a vector $\mathbf{a}$ with independent entries from a α-stable distribution in the input space.
3. Find $K(\mathbf{a}, v)$ for all $v \in \mathcal{S}_1$. Calculate $R_{\max} = \max_{v \in \mathcal{S}_1} K(\mathbf{a}, v) - \min_{v \in \mathcal{S}_1} K(\mathbf{a}, v)$.
4. For given B the number of bins calculate $r = R_{\max}/B$.
5. Generate $\theta \sim Unif[0, r]$, and find $\tilde{h}_{\mathbf{a},\theta}(v) := h_{\phi(\mathbf{a}),\theta}(\phi(v))$ as (8) (in the feature space), this is, $\tilde{h}_{\mathbf{a},\theta}(v) = \left\lfloor \frac{K(\mathbf{a},v)+\theta}{r} \right\rfloor$, for all $v \in \mathcal{S}_1$.
6. Eliminate bins with η_3 percent highest and lowest values.
7. For each one of the remaining bins, randomly select a representative.
8. Repeat N times steps 2 to 5. Call $\hat{\mathcal{S}}$ the set formed by all the representatives found.
9. Solve (4) using $\hat{\mathcal{S}}$ and $\hat{y}$ their corresponding classes, instead of $\mathcal{S}$ and y. Call $\hat{\lambda}^{(1)}$ the solution, and $\hat{w}^{(1)}, \hat{b}^{(1)}$ the corresponding hyperplane values as defined in (5) and (6).

3.2. Directed Projections

In this case, the direction $\mathbf{a}$ is found by solving small random SVM subproblems. To motivate this choice, assume the solution w^* of the complete SVM problem was known beforehand. By construction, projecting in the direction of w^* immediately identifies the support vectors. By sampling a small subset and solving the associated SVM problem we can approximate w^*, and projections in this direction allow for a much more precise sample. In summary, we obtain Algorithm 2.

Algorithm 2 LSH-SVM (directed projections)

Given an initial kernel K, B number of bins, $\eta_1, \eta_2 \in (0,1)$ the percentages of subsample data points, N the number of projections repetitions, η_3 the cutoff percentage, and $\mathcal{S} = \{X_1, \ldots, X_n\}$ the data set of problem (4):

1. Take a random subsample $\mathcal{S}_1$ of $\mathcal{S}$ with size $n_1 = \lfloor \eta_1 n \rfloor$.
2. Take a random subsample $\mathcal{S}_2$ of $\mathcal{S}$ with size $n_2 = \lfloor \eta_2 n \rfloor$.
3. Find λ_{n_2} the solution to problem (4) corresponding to $\mathcal{S}_2$ and normalize $\tilde{\lambda}_{n_2 i} = \frac{\lambda_{n_2 i}}{\sum_{i=1}^{n_2} (\lambda_{n_2 i}^2)^{1/2}}$.
4. Denote the corresponding normal hyperplane direction $\tilde{w}_{n_2}$ as defined in (5) and (6).
5. Find $< \tilde{w}_{n_2}, \phi(v) >= \sum_i \tilde{\lambda}_{n_2 i} y_i K(X_i, v)$ for all $v \in \mathcal{S}_1$. Calculate $R_{\max} = \max_{v \in \mathcal{S}_1} < \tilde{w}_{n_2}, \phi(v) > - \min_{v \in \mathcal{S}_1} < \tilde{w}_{n_2}, \phi(v) >$.
6. For given B the number of bins calculate $r = R_{\max}/B$.
7. Generate $\theta \sim Unif[0, r]$ and find

$$h_{\tilde{\mathbf{w}}_{\mathbf{n_2}},\theta}(\phi(v)) = \left\lfloor \frac{< \tilde{w}_{n_2}, \phi(v) > +\theta}{r} \right\rfloor = \left\lfloor \frac{\sum_i \tilde{\lambda}_{n_2 i} y_i K(X_i, v) + \theta}{r} \right\rfloor$$

   for all $v \in \mathcal{S}_1$.
8. Eliminate bins with η_3 percent highest and lowest values.
9. For each one of the selected bins, randomly select a representative.
10. Repeat N times steps 2 to 7. Call $\bar{\mathcal{S}}$ the set formed by all the representatives found.
11. Solve (4) using $\bar{\mathcal{S}}$ and $\bar{y}$ their corresponding classes, instead of $\mathcal{S}$ and y. Call $\bar{\lambda}^{(2)}$ the solution, and $\bar{w}^{(2)}, \bar{b}^{(2)}$ the corresponding hyperplane values as defined in (5) and (6).

4. Numerical Experiments

In this section, we show the results of applying the LSH-SVM method to a set of real life SVM problems taken from the LIBSVM webpage www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/ (accessed on 27 April 2022). The method was implemented under R environment and the SVM problems were all solved using the Kernlab package. Kernlab [19] uses a version of the algorithm implemented in LIBSVM, based on the method SMO proposed by Platt [7].

The problems tested and their sizes can be found in Table 1. We selected 80% for the training stage of the algorithms and the remaining 20% of the data is used for finding the classification error.

Table 1. Problems tested.

Problem	Number of Data Points (n)	Number of Attributes (d)
a9a	48,844	123
w8a	49,749	300
covtype.binary	581,012	54
cod-rna	59,535	8
ijcnn1	49,990	22
skin	245,057	3
phishing	11,055	68

The objectives of the experiments were to study the performance of our approach and the effect of the parameters, as well as to analyze the impact of the directed directions in contrast with the use of random ones. All the experiments performed considered the RBF Gaussian kernel defined as $K(X_i, X_j) = \exp(-\frac{||X_i - X_j||^2}{2(0.05)^2})\ \forall i, j$, and the value of the parameter C from problem (4) was set equal to 5. We would like to highlight that the algorithms here presented are specially proposed as efficient techniques for solving SVM problems with nonlinear kernels.

In order to be more effective in the choice of representatives from each bin and to reduce running times, we considered in practice a slight change of Algorithms 1 and 2.

Step 5 in Algorithm 1 was changed by

1. Generate $\theta \sim Unif[0, r]$ and calculate the B intervals (bins) with equal (or almost equal) number of values of the collection $K(\mathbf{a}, v) + \theta, v \in \mathcal{S}_1$.
2. For each $v \in \mathcal{S}_1$ define $\tilde{h}_{\mathbf{a},\theta}(v)$ to be the corresponding bin of the value $K(\mathbf{a}, v) + \theta$.

Step 7 in Algorithm 2 was changed by

1. Generate $\theta \sim Unif[0, r]$ and calculate the B intervals (bins) with equal (or almost equal) number of values of the collection $< \tilde{w}_{n_2}, \phi(v) > +\theta, v \in \mathcal{S}_1$.
2. For each $v \in \mathcal{S}_1$ define $h_{\tilde{\mathbf{w}}_{\mathbf{n_2}}\theta}(\phi(v))$ to be the corresponding bin of the value $< \tilde{w}_{n_2}, \phi(v) > +\theta$.

In Table 2 we include the CPU training time and classification errors obtained for the tested problems using the LSH-SVM Algorithm 2 (directed projections) and Kernlab for the complete data set. After an extensive number of trials, we selected the parameters $\eta_1 = 0.001, \eta_2 = 0.005, N = 100, B = 40$ for the LSH-SVM algorithm. In each case, $N_1 = 20$ repetitions were considered and the minimum, maximum, mean, and standard deviation were calculated. For data set w8a, we set $\eta_2 = 0.008$ because the data are not balanced (approx. 3–97%) so that taking $\eta_2 = 0.001$ led to one class samples. Steps 4 and 6 in Algorithms 1 and 2, respectively, were performed by discarding the bins corresponding to the $\eta_3 = 10\%$ largest and smallest values. We also include in Table 2 the number of support vectors found by each approach at the training stage. It can be observed that the running time obtained by our proposed algorithm is much lower than when using Kernlab for the complete data set. In addition, there is not a significant degradation of the classification

error in many cases, even though the number of support vectors is much smaller. This seems to point out that much more support vectors than the ones really needed for a good classification were obtained by using the complete data set. An additional benefit of the reduced number of support vectors is reduced fitting time as shown in Table 3.

Table 2. LSH-SVM Algorithm 2 vs. Kernlab.

Problem	LSH-SVM ($N_1 = 20$)				Kernlab	Variable
	Min	Max	Mean	Std		
a9a	Directed: $\eta_1 = 0.001$, $\eta_2 = 0.005$					
	12.28	15.5	13.53	1.05	1972.58	CPU time (s)
	0.1993	0.2120	0.2047	0.0035	0.1798	Classification error
	1115	1223	1168.31	29	23,004	# Support vectors
w8a	Directed: $\eta_1 = 0.001$, $\eta_2 = 0.008$					
	21.94	31.20	26.17	2.88	2313.6	CPU time (s)
	0.0288	0.0315	0.0301	0.0006	0.0203	Classification error
	24	486	273.4	151.47	23,658	# Support vectors
covtype	Directed: $\eta_1 = 0.001$, $\eta_2 = 0.005$					
	112.55	125.79	118.72	3.44	—	CPU time (s)
	0.2062	0.2137	0.2097	0.002	—	Classification error
	1099	1199	1148.65	26.54	—	# Support vectors
cod-rna	Directed: $\eta_1 = 0.001$, $\eta_2 = 0.005$					
	3.65	4.54	4.04	0.29	84.55	CPU time (s)
	0.0513	0.0561	0.0536	0.0013	0.0460	Classification error
	461	516	484.1	15.76	6669	# Support vectors
ijcnn1	Directed: $\eta_1 = 0.001$, $\eta_2 = 0.005$					
	3.42	4.81	3.73	0.30	49.64	CPU time (s)
	0.0377	0.0518	0.0423	0.0034	0.0132	Classification error
	297	349	317.65	13.66	2900	# Support vectors
skin	Directed: $\eta_1 = 0.001$, $\eta_2 = 0.005$					
	5.83	6.59	6.04	0.21	224.45	CPU time (s)
	0.0103	0.0118	0.0111	0.0004	0.0053	Classification error
	487	522	502.3	9.23	4156	# Support vectors
phishing	Directed: $\eta_1 = 0.001$, $\eta_2 = 0.005$					
	2.98	3.16	3.04	0.063	27.31	CPU time (s)
	0.0660	0.1239	0.1007	0.0166	0.0275	Classification error
	443	484	464.2	10.84	3221	# Support vectors

Finally, for completeness sake we have included a baseline comparison to simple random sampling with $N_1 = 20$ replicas over set a9a in order to highlight the performance of Algorithms 1 and 2. In Table 4 we show results for different sample sizes ranging from $l = 0.1$ to $l = 0.001$. The latter comparable to the size of random samples used in Algorithms 1 and 2. Errors for $l = 0.1$ are essentially comparable to results of Algorithm 2 and increase as sample size decreases, as predicted. It is interesting to note however,

that the number of support vectors for a random sample $l = 0.1$ is almost double than for Algorithm 2. Support vectors decrease with sample size and are typically close to this value.

Table 3. CPU estimation times for proposed algorithms and Kernlab over the test sample. For all cases, the test sample is randomly chosen and corresponds to 20% of the complete data set.

Problem	Fit Time Kernlab	Fit Time Algorithm 1	Fit Time Algorithm 2
a9a	27.54	1.53	1.60
w8a	63.39	1.1	0.26
covtype.binary	–	19.08	11.83
cod-rna	7.12	0.40	0.54
ijcnn1	2.05	0.31	0.35
skin	18.12	1.78	2.67
phishing	0.83	0.16	0.12

Table 4. Random samples baselines for set a9a. As sample size increases, error is smaller but number of SV increase. Total error for $l = 0.1$ is essentially equal to results for LSH-SVM Algorithm 2, but number of SV is consistently larger.

Sample Size	Random Samples for Set a9a ($N_1 = 20$)				Variable
	min	max	mean	std	
$l = 0.1$	6.75	9.54	7.37	0.6	CPU time (s)
	0.18	0.20	0.20	0.003	Classification error
	2983	3096	3031	26.9	# Support vectors
$l = 0.01$	0.08	0.13	0.09	0.013	CPU time (s)
	0.2	0.23	0.22	0.005	Classification error
	361	384	373	4.86	# Support vectors
$l = 0.005$	0.03	0.04	0.04	0.003	CPU time (s)
	0.21	0.23	0.23	0.004	Classification error
	189	195	193	1.56	# Support vectors
$l = 0.001$	0.016	0.028	0.019	0.004	CPU time (s)
	0.22	0.24	0.24	0.004	Classification error
	39	39	39	0	# Support vectors

In order to study the benefit of our directed directions, we also solved the problems with the LSH-SVM Algorithm 1 using random directions for the projections. The results can be found in Table 5. Notice that, as should be expected, the running time is better when using random projections since no SVM problems need to be solved. Although in some cases the difference is not very noticeable.

In addition, as expected, the directed directions separate classes more accurately than random ones. This is illustrated in Figure 1, where histograms for the two kind of projections are shown for problems covtype.binary and a9a from LIBSVM. Each figure represents the histogram of both type of projections over the same sample. Coloring is set by the most frequent class (-1 or 1) over each bin in the histogram.

However, although errors tend to be slightly smaller with Algorithm 2, the differences seem to be related to complexity of the problem (dimension, nonlinearity).

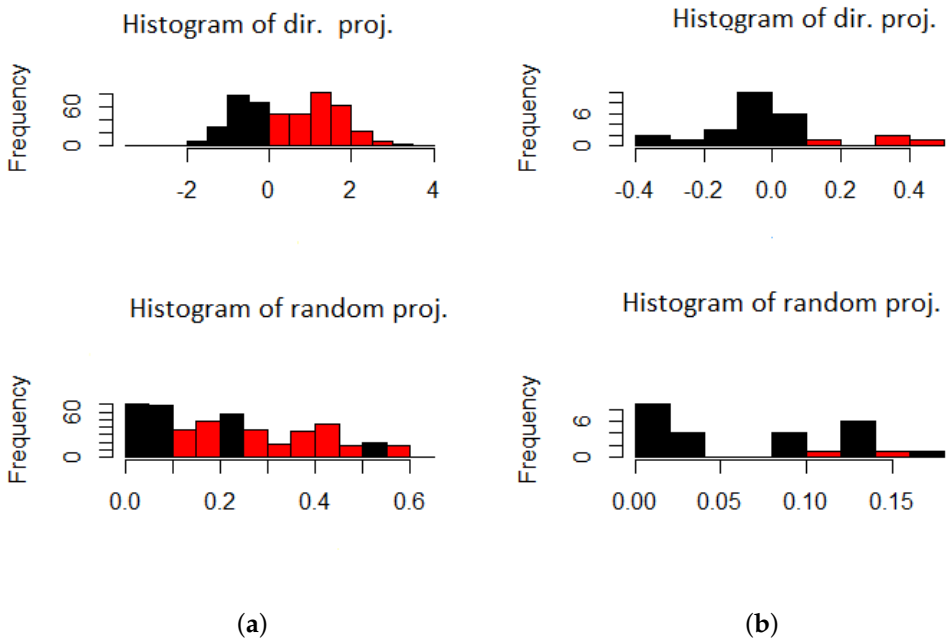
Table 5. LSH-SVM Algorithm 1: random vs. Algorithm 2: directed (mean).

Problem	LSH-SVM ($N_1 = 20$)				Mean	Variable
	min	max	mean	std		
a9a	Random: $\eta_1 = 0.001$				Dir.	
	12.14	15.49	12.78	1.04	13.53	CPU time (s)
	0.2092	0.2175	0.2128	0.0026	0.2047	Classification error
	1186	1262	1235.9	22.08	1168.31	# Support vectors
w8a	Random: $\eta_1 = 0.001$				Dir.	
	13.25	44.45	19.03	8.73	26.17	CPU time (s)
	0.0293	0.0309	0.0299	0.0004	0.0301	Classification error
	34	405	189.8	125.02	273.4	# Support vectors
covtype	Random: $\eta_1 = 0.001$				Dir.	
	35.18	41.97	37.6	2.025	118.72	CPU time (s)
	0.2329	0.2511	0.2395	0.0043	0.2097	Classification error
	1685	1816	1749.85	30.98	1148.65	# Support vectors
cod-rna	Random: $\eta_1 = 0.001$				Dir.	
	2.81	3.62	3.15	0.27	4.04	CPU time (s)
	0.0525	0.0567	0.0540	0.0011	0.0536	Classification error
	373	443	399.5	18.25	484.1	# Support vectors
ijcnn1	Random: $\eta_1 = 0.001$				Dir.	
	3.29	4.50	3.83	0.33	3.73	CPU time (s)
	0.0351	0.0488	0.0432	0.0037	0.0423	Classification error
	323	407	350.75	19.59	317.65	# Support vectors
skin	Random: $\eta_1 = 0.001$				Dir.	
	5.15	9.56	5.91	0.99	6.04	CPU time (s)
	0.01405	0.0198	0.0166	0.0012	0.0111	Classification error
	362	420	393.2	17.30	502.3	# Support vectors
phishing	Random: $\eta_1 = 0.001$				Dir.	
	2.16	3.07	2.44	0.28	3.04	CPU time (s)
	0.0809	0.1542	0.1198	0.0222	0.1007	Classification error
	412	462	440.3	12.18	464.2	# Support vectors

The effect of changing parameters is shown in the next series of figures for data set covtype.binary. Figure 2 shows changes for number of bins from 10 to 250, for both Algorithms 1 and 2. Error decreases nonlinearly, as follows from Theorems 3 and 4 combining the increase in number of bins with more separated points. Increasing the number of bins in Algorithm 2 has a greater effect than on Algorithm 1, probably because of the greater effect over separating points for the former. Time increases very slowly at first for Algorithm 2 and almost linearly for Algorithm 1. As the number of bins increases, time is almost equivalent for both algorithms. The number of support vectors increases almost linearly for both algorithms; however, Algorithm 1 has a smaller slope, indicating a greater efficiency in finding significant support vectors.

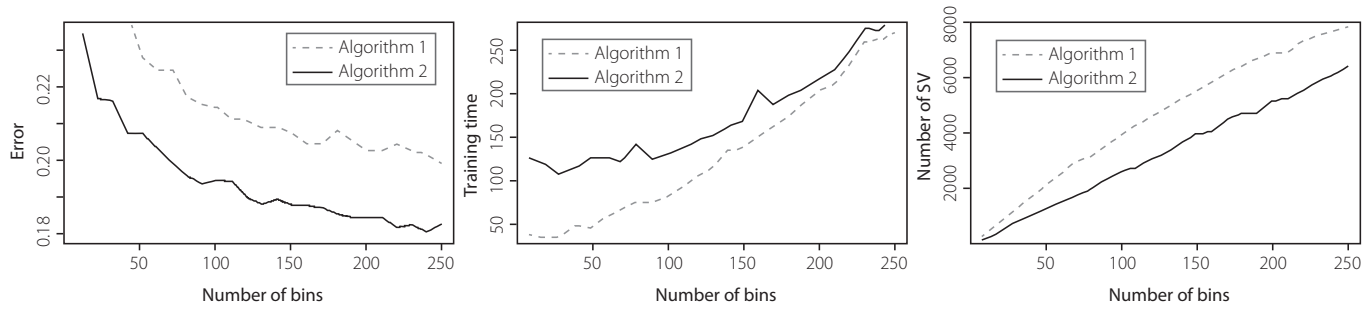
Figure 3 shows the effect of varying the number of projections. Error-wise, the effect over both algorithms is similar (curves are parallel) with bigger errors for Algorithm 1. Time

increases piecewise linearly with a slope change around 350 iterations. Before this change, Algorithm 1's slope is smaller than that of Algorithm 2. Both curves appear to be parallel however after 350 projections. As for the number of bins, the number of SV increases almost linearly. Algorithm 2 is again more efficient in selecting significant support vectors.



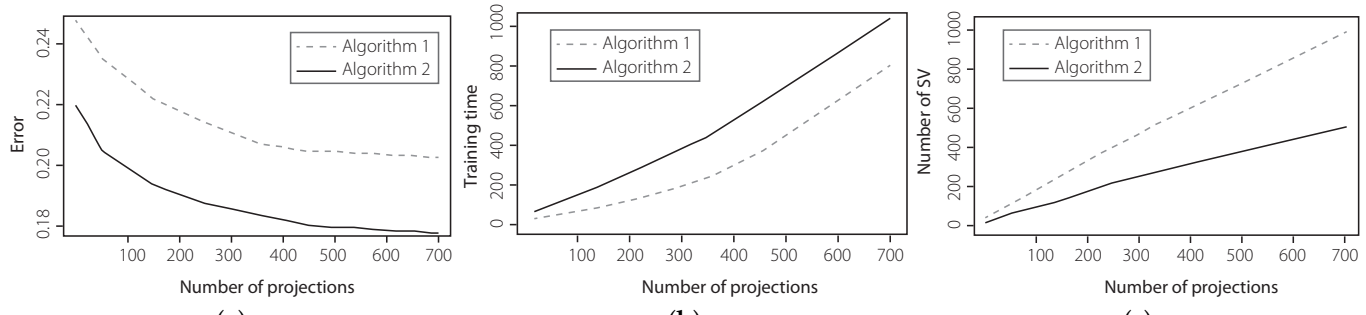
(**a**) (**b**)

Figure 1. Histograms of Directed projections and Random projections algorithms (Algorithms 1 and 2). Data sets covtype.binary (**a**) and a9a (**b**).



(**a**) (**b**) (**c**)

Figure 2. Plots of the effect of changing the number of bins over error (**a**), time (**b**) and number of SV (**c**) for data set covtype.binary.



(**a**) (**b**) (**c**)

Figure 3. Plots of the effect of changing the number of projections over error (**a**), time (**b**), and number of SV (**c**) for data set covtype.binary.

Finally, Figure 4 shows the effect of varying the size of the second sample for Algorithm 2, given by η_2. Error decreases quite quickly, reaching a plateau. Time increases nonlinearly however, probably owing to the changing effect of the proportion of time employed in finding the first approximate solution. The number of support vectors appears to find an optimal (lower) level for sample sizes around $\eta_2 = 0.005$.

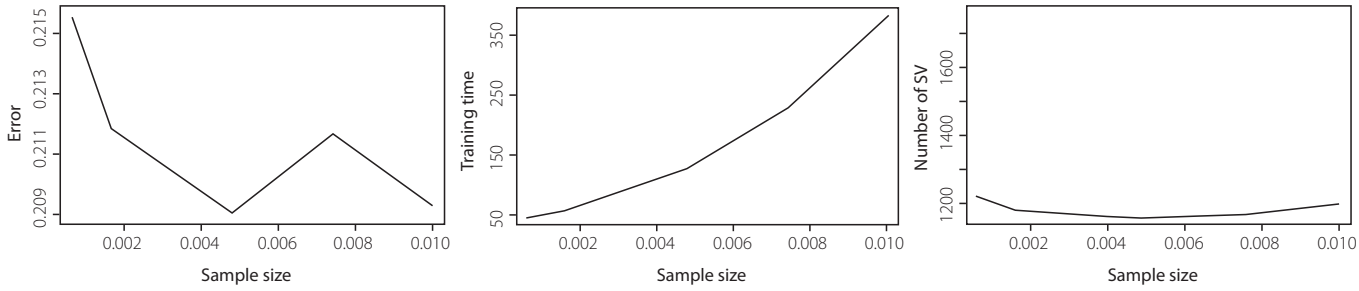


Figure 4. Plots of the effect of parameter η_2 (size of the first small SVM problem in Algorithm 2 in order to obtain projection directions) over: error (**a**), time (**b**), and number of SV (**c**) for data set covtype.binary. Larger sample sizes were not considered because of the increase in training time.

5. Theoretical Results

The procedures described in Algorithm 1 (random projections) and Algorithm 2 (directed projections) produce subsamples of size $N \times B$ by appropriately selecting points from a series of N samples each distributed in B bins. In each case a SVM model is adjusted based on the representatives selected. In this section, we give approximation results for both proposed methods and give some intuition as to when directed sampling is better than random sampling. Approximations are based on two main results: general risk minimization theory [35,36] used to bound, with high probability, the supremum of the differences between the original loss function (to be introduced in the preliminaries) and its approximation using a subsample of the data. The bounds are shown to depend on the trace, the spectral and the Frobenious norm of kernel K over the subsample. In addition, deterministic approximation lemmas allow us to bound the difference between the solution w^* and that obtained by subsampling, as in Theorem 3. This then allows us to argue in the case of directed projections, that chosen points are more correlated to the normal vector of the optimal separating hyperplane with high probability. Finally, our results can be used to improve bounds over the unobserved theoretical misclassification error.

We begin by introducing some preliminaries.

5.1. Preliminaries

We assume that the n observations (X_i, y_i) $i = 1, \ldots, n$, $X_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$, are independent with identical joint distribution P. In what follows, $E(h(X, y))$ stands for the expectation of any function h of the random vector (X, y) with respect to probability P. Our aim is to construct the data-dependent function $g = g(w, b)$ with values in $\{-1, 1\}$, as defined in (3), over an appropriate function space such that $P(g(X) \neq y)$ is small. For this, we choose a vector (w, b) minimizing an optimization problem equivalent to (2) defined over a class of data-defined functions over a subsample of the original observations (X_i, y_i) $i = 1, \ldots, n$.

Notice that (2) can be written as the unconstrained minimization problem

$$\underset{w,b}{\text{minimize}} \; \frac{1}{2}\|w\|_{\mathcal{H}}^2 + C \sum_{i=1}^{n} \psi(-y_i(< w, \phi(X_i) > + b)) \tag{9}$$

with $\psi(x) = \max(1 + x, 0)$. Recall we denote by (w^*, b^*) the solution.

Define $A_n(w,b) := \frac{1}{n}\sum_{i=1}^{n}\psi(-y_i(f_w(X_i)+b))$, with $f_w(x) =< w, \phi(x) >$. This function is known as the hinge loss function. Divide the objective function of problem (9) by nC, and let $M = \frac{1}{2nC}$. Then, we can rewrite problem (9) as

$$\underset{w,b}{\text{minimize}} \quad M\|w\|_{\mathcal{H}}^2 + A_n(w,b). \tag{10}$$

Clearly, using (5) and (6), the minimization problem (10) can in turn be restated as

$$\begin{array}{ll} \underset{w,b}{\text{minimize}} & A_n(w,b) \\ \text{over the set} & \mathcal{F}_n \end{array} \tag{11}$$

with

$$\mathcal{F}_n = \{(w,b) : \|w\|_{\mathcal{H}} \leq \|w^*\|_{\mathcal{H}}, |b| \leq 1 + \|w^*\|_{\mathcal{H}} \max_j \sqrt{K(X_j,X_j)}\}.$$

The next proposition gives bounds for the feasible points of problem (11) in terms of C,n and kernel K. For any $w = \sum_{i=1}^{n} w_i\phi(X_i)$ we use the following notation: $\|w\|_\infty = \max_{1\leq i\leq n}(|w_i|)$, $\|K\|_2 = \sup_{\|w\|_2=1}\|Kw\|_2$, $\|K\|_F^2 = \sum_{i,j=1} K_{i,j}^2$, and $K^{1/2} := [|K_{i,j}|^{1/2}]_{1\leq i,j\leq n}$ the entry-wise square root matrix. We also introduce the following assumptions on Kernel K.

K1 There exists a positive constant C_K such that for all x, $K(x,x) \leq C_K$.
K2 Given $d_1 > 0$, for all x, y with $\|x-y\| > d_1$ there exists a positive constant $\varepsilon(d_1)$ such that $K(x,y) < \varepsilon(d_1)$.

Proposition 1. *Let $(w,b) \in \mathcal{F}_n$. Then $\|w\|_{\mathcal{H}} \leq R := C\min(\sqrt{n}\|K\|_2^{1/2}, \|K^{1/2}\|_F)$ and $|b| \leq 1 + R\sqrt{C_K}$.*

Proof. The proof follows easily by bounding w^* using (5). We have that $w^* = \sum_{i=1}^{n} w_i^*\phi(X_i)$ with $w_i^* = \lambda_i^* y_i$ and $0 \leq \lambda_i^* \leq C$. Then, $\|w^*\|_{\mathcal{H}}^2 = \sum_{i,j=1}^{n} w_i^* w_j^* K(X_i,X_j)$ satisfying $\|w^*\|_{\mathcal{H}}^2 \leq \min(\|K\|_2\|w^*\|_2^2, \|w^*\|_\infty^2\|K^{1/2}\|_F^2) \leq R$. On the other hand, $|b| \leq 1 + ||w^*||_{\mathcal{H}} \max_j \sqrt{K(X_j,X_j)}$. The result follows bounding $K(X_j,X_j) \leq C_K$. □

Observe that if K^S denotes the submatrix of K formed by the values corresponding to the position of the support vectors, zeroing the other components, that is, $K_{ij}^S = K_{ij}$ if $i \in SV$ or $j \in SV$ and $K_{ij}^S = 0$ if not, then the solution of (11) satisfies being in the set $\mathcal{F}^S = \{(w,b) : \|w\|_{\mathcal{H}} \leq R^S, |b| \leq R^S\sqrt{C_K} + 1\}$ where $R^S = C\min(\sqrt{|SV|}\|K^S\|_2^{1/2}, \|(K^S)^{1/2}\|_F)$.

The next lemma from [35] gives a bound, in probability, between the hinge loss function and its expectation, i.e., between $A(w,b) := E\psi(-y(f_w(X)+b))$ and $A_n(w,b)$. Clearly $E(A_n(w,b)) = A(w,b)$. This result will be at the heart of our theory.

Lemma 1. *Let us define $\Delta := sup_{(w,b)\in\mathcal{F}_n}[A(w,b) - A_n(w,b)]$. Then, with probability greater than $1-\delta$ we have that*

$$\Delta < \frac{4}{n}\sqrt{Tr(K)} + \Big(\sup_{(w,b)\in\mathcal{F}_n}\|f_w + b\|_\infty + 1\Big)\sqrt{\frac{2\log(1/\delta)}{n}}$$

with $Tr(K) = \sum_{i=1}^{n} K(X_i,X_i)$.

Proof. The proof follows [35], page 8, bounding $\sup_{x,y}\psi(-yf(x)) \leq \|f\|_\infty + 1$ and $E\Delta$ by $\frac{2}{n}\sqrt{Tr(K)}$ using Radamacher averages [35]. □

In the following subsections, we use the previous lemma to compare the values of the hinge loss function for the given data set and for subsamples of the data set generated randomly or by the LSH-SVM method. For some results, we require the following assumptions on the original sample:

S1 There exists a unique (w^*, b^*) which minimizes $A_n(w, b)$ in problem (12) over $\tilde{\mathcal{F}}_n$.
S1' With probability one, the collection of classes $\mathcal{F} = \cup_n \mathcal{F}_n$ defined by an infinite sample $\{(X_i, y_i)\}_{i\geq 1}$ is such that $(w_{0,n}, b_{0,n}) = \arg\min_{(w,b)\in\mathcal{F}_n} A(w, b)$ converges to an overall solution $(w_0, b_0) \in \mathcal{F}$.

A sufficient condition for [S1] is that there are a different number of support vectors for each class (see [37]). This is usually the case for practical SVM problems. In addition, the overall minimum of $A(w, b)$ is the Bayes classifier (see [35]). If, with probability one, the class is rich enough, condition [S1'] then implies $(w_{0,n}, b_{0,n})$ converges to the Bayes classifier.

In Section 5.2, we consider the case when the subsamples are randomly chosen. Sections 5.3 and 5.4 cover the cases when the subsamples are generated by the LSH-SVM Algorithms 1 and 2, respectively. In these latter subsections, we show that the approximation bounds improve respect to the random case, and we also show how these bounds relate for the two algorithms.

5.2. Random Samples

Consider an index set $M \subset \{1, \ldots, n\}$ corresponding to a random sample of the original data set (without replacement) of size l. Recall, as shall be required below, that any function of the random sample may be thought of as a function of the sampled variables $(Z_1, t_1), \ldots, (Z_l, t_l)$, where $P((Z_k, t_k) = (X_i, y_i)) = 1/n$. Since the sample is without replacement, if $Z_i = 1$ if (X_i, y_i) is selected among the l trials, then $P(Z_{i_1} \cdots Z_{i_r} = 1) = \frac{\binom{m}{r}}{\binom{|SV|}{r}}$. Set $\tilde{S}$ to be the sample based on the index set M, and let $\tilde{K}$ be the associated kernel matrix, that is $\tilde{K}_{st} = K(Z_s, Z_t)$.

Following the presentation above, define $A_l(w, b) := \frac{1}{l}\sum_{k=1}^{l} \psi(-t_k(f_w(Z_k) + b))$.

As for problem (10), the unconstrained minimization problem with solution (w^l, b^l) can be stated as minimizing A_l over the set $\mathcal{F}_{n,l} = \{(w, b) : w = \|w\| \leq \|w^l\|_{\mathcal{H}}, |b| \leq 1 + \|w^l\|_{\mathcal{H}} \max_j \sqrt{\tilde{K}(Z_j, Z_j)}\}$.

For our theory, we use a closely related minimization problem, where the feasible set $\mathcal{F}_{n,l}$ is substituted by the set $\tilde{\mathcal{F}}_{n,l}$ defined as

$$\tilde{\mathcal{F}}_{n,l} = \begin{cases} \mathcal{F}_{n,l} & \text{if } \|w^l\|_{\mathcal{H}} \leq \|w^*\|_{\mathcal{H}} \\ \mathcal{F}_n & \text{if } \|w^l\|_{\mathcal{H}} > \|w^*\|_{\mathcal{H}} \end{cases}$$

This is, the minimization problem

$$\begin{array}{ll} \underset{w,b}{\text{minimize}} & A_l(w, b) \\ \text{over the set} & \tilde{\mathcal{F}}_{n,l} \end{array} \tag{12}$$

Observe that Problem (12) is equivalent to an unconstrained problem of the type of (9) for some parameter $\tilde{C}$. The followings are satisfied

(1) $\tilde{\mathcal{F}}_{n,l} \subseteq \mathcal{F}_{n,l}$, $\tilde{\mathcal{F}}_{n,l} \subseteq \mathcal{F}_n$.
(2) $\cup_l \tilde{\mathcal{F}}_{n,l} = \mathcal{F}_n$.
(3) Following Proposition 1, for any $(w, b) \in \tilde{\mathcal{F}}_{n,l}$ we have that $\|w\|_{\mathcal{H}} \leq R_l$ and $|b| \leq C\sqrt{l}\sup_M \|\tilde{K}\|_2 + 1 \leq 1 + R_l\sqrt{C_K}$ with

$$R_l := C\min(\sqrt{l}\sup_M \|\tilde{K}\|_2^{1/2}, \sup_M \|\tilde{K}^{1/2}\|_F).$$

(4) $A_l(w^l, b^l)$ is a lower bound for $A_l(\tilde{w}, \tilde{b})$ with $(\tilde{w}, \tilde{b})$ the solution of problem (12).

Next, our objective is to bound the difference of the hinge loss functions corresponding to the original dataset and the random sample.

Set $\Delta_M := sup_{(w,b)\in\tilde{\mathcal{F}}_{n,l}}[A_n(w,b) - A_l(w,b)]$. In order to bound Δ_M, we use the following Mc Diardmid type inequality for symmetric functions of samples due to Cortes et al., 2008 (cited from Kumar et al. [36] (Th. 2: pages 998 to 1000)).

Theorem 1. *Let $Z_1, \ldots, Z_\ell$ be a sequence of random variables sampled uniformly without replacement from a fixed set of $\ell + u$ elements. Let $\Gamma : \mathbb{Z}^\ell \to \mathbb{R}$ be a symmetric function such that for all $j \in \{1, \ldots, \ell\}$, $|\Gamma(Z_1, \ldots, Z_j, \ldots, Z_\ell) - \Gamma(Z_1, \ldots, Z'_j, \ldots, Z_\ell)| \leq c$. Then,*

$$P(\Gamma - E(\Gamma) > \varepsilon) \leq e^{-\frac{2\varepsilon^2}{\alpha(\ell,u)c^2}}$$

where $\alpha(\ell, u) = \frac{\ell u}{\ell+u} \frac{1}{1-1/(2\max(\ell,u))}$

Based on Lemma 1 and Theorem 1, we obtain the following theorem

Theorem 2. *Assume kernel K satisfies assumption [K1]. Consider a sample of size l taken without replacement from an original sample of size n. Define $\Delta_M := sup_{(w,b)\in\tilde{\mathcal{F}}_{n,l}}[A_n(w,b) - A_l(w,b)]$ and $T_l(K) := 2R_l\sqrt{C_K} + 1$. Then, with probability greater than $1 - \delta$*

$$\Delta_M \leq \frac{4}{l}\sqrt{Tr(\tilde{K})} + T_l(K)\sqrt{\frac{1}{2}\log(1/\delta)\frac{n-l}{n\ l\ (1 - \frac{1}{2\max(l,n-l)})}}.$$

Proof. The proof follows directly from Lemma 1 and Theorem 1, bounding $|\frac{1}{l}(\Delta_M(Z) - \Delta_M(Z'))|^2 \leq \frac{(\sup_{w,b}\sup_j |f_w(X_j)+b|+1)^2}{l^2}$ using that $E_M A_l(w,b) = A_n(w,b)$, where E_M stands for the expectation with respect to the subsampling procedure, that is, conditional to the original sample $\mathcal{S}$. Next we bound $\sup_{w,b}\sup_j |f_w(X_j) + b|$. By definition $f_w(x) =< w, \phi(x) >$ so $\sup_{w,b}\sup_j |f_w(X_j) + b| \leq \|w\|_{\mathcal{H}}\sup_j\sqrt{K(X_j, X_j)} + |b| \leq 2R_l\sqrt{C_K} + 1 = T_l(K)$ by our assumptions over w (see (3)) and Kernel K. □

Using Theorem 2, we obtain the following bound

$$A_n(w^*, b^*) - A_l(\tilde{w}, \tilde{b}) \leq \min_{(w,b)\in\tilde{\mathcal{F}}_{n,l}} A_n(w,b) - A_l(\tilde{w}, \tilde{b}) \leq \Delta_M.$$

Theorem 2 also allows us to give a bound between the minimizers (w^*, b^*) and $(\tilde{w}, \tilde{b})$. For this purpose, we assume [S1] applies to the minimizer $(\tilde{w}, \tilde{b})$ of problem (11), and we present the next two lemmas.

Lemma 2. *Let $V : \mathbb{R}^n \to$ be a convex function, $\mathcal{F} \subseteq \mathbb{R}^n$ a not empty convex, closed, and bounded set, and x^* a unique global minimizer of V over $\mathcal{F}$. Then, there exists ε_o such that for all $0 < \epsilon < \varepsilon_o$ there exists $\delta = \delta(\epsilon)$ such that if $||x - x^*|| > \delta$ we have that $V(x) - V(x^*) > \epsilon$ for any $x \in \mathcal{F}$.*

Proof. Let $\varepsilon_o := \arg\max_{x\in\mathcal{F}} V(x) + V(x^*)$, which is well defined because V is continuous and $\mathcal{F}$ compact. Moreover, since V cannot be the constant function equal to zero, ε_o is positive. Given any $\varepsilon_o > \epsilon > 0$ consider the level set $\mathcal{E} = \{x \in \mathcal{F} : V(x) = \epsilon + V(x^*)\}$. This set is not empty because the function V is coercive, this is $V(x) \to \infty$ if $||x|| \to \infty$. Let us define $\delta = \delta(\epsilon)$ the distance from x^* to the set $\mathcal{E}$. This distance exists because $\mathcal{E}$ is a not empty closed set inside a compact, therefore is compact.

Let $x \in \mathcal{F}$ such that $||x^* - x|| > \delta$. Then, there exists $x_\epsilon \in \mathcal{E}$ such that $||x^* - x_\epsilon|| = \delta$ and $\lambda \in (0,1)$ with $x_\epsilon = \lambda x^* + (1-\lambda)x$; therefore, by convexity, $V(x_\epsilon) \leq \lambda(V(x) - V(x^*)) + V(x) < V(x)$ and we obtain that $V(x) - V(x^*) > \epsilon$. □

Lemma 3. *Consider a function V satisfying the assumptions of previous lemma and $\tilde{V}$ another function with unique minimizer $\tilde{x}$ over $\mathcal{F}$ and such that $sup_{x\in\mathcal{F}}|V(x)-\tilde{V}(x)| \leq \epsilon$, for some $\frac{\varepsilon_0}{2} > \epsilon > 0$. Then $\|x^*-\tilde{x}\| \leq \delta(2\epsilon)$ with δ given as in the previous lemma.*

Proof. Assume that $\|x^*-\tilde{x}\| > \delta(2\epsilon)$. Then, by Lemma 2, we have that $2\epsilon < V(\tilde{x}) - V(x^*)$; therefore, $2\epsilon + V(x^*) + \tilde{V}(x^*) - \tilde{V}(x^*) < V(\tilde{x}) - \tilde{V}(\tilde{x}) + \tilde{V}(\tilde{x})$. Using the assumption at each side of the inequality, we obtain $2\epsilon - \epsilon + \tilde{V}(x^*) < \epsilon + \tilde{V}(\tilde{x})$. This contradicts the fact that $\tilde{x}$ is the minimizer of $\tilde{V}$ over $\mathcal{F}$. □

Observe that previous lemmas apply to the Hilbert space $\mathcal{H}$ being finite dimensional or being infinite dimensional (by considering the restriction to the subspace of finite linear combinations of the data set, based on the dual representation [3]). Now the theorem can be stated.

Theorem 3. *Assume kernel K satisfies [K1]. Assume [S1] and [S1'] are satisfied. Then, there exist n, l such that with probability greater than $1-2\delta$, $\|(w^*,b^*)-(\tilde{w},\tilde{b})\| \leq \delta(\varepsilon/2)+\delta(\varepsilon)+\varepsilon$ with*

$$\varepsilon = \frac{4}{l}\sqrt{Tr(\tilde{K})} + T_l(K)\sqrt{\frac{1}{2}\log(1/\delta)\frac{n-l}{n\ l\ (1-\frac{1}{2\max(l,n-l)})}},$$

$T_l(K) := 2R_l\sqrt{C_K}+1$ as in the previous theorem, and $||.||$ denoting the induced norm in the Cartesian product space $\mathcal{H}\times\mathbb{R}$.

Proof. By [S1'] and [S1], $A(w,b)$ satisfies the assumptions of Lemma 2 over $\mathcal{F}$, so there exists ε_0 and $\delta(\epsilon)$ satisfying Lemma 1 for all $\epsilon < \varepsilon_0$ over $\mathcal{F}_n$ for large enough n. In addition, [S1] is satisfied for $A_n(w,b)$ over $\mathcal{F}_n$ and for $A_l(w,b)$ over $\tilde{\mathcal{F}_{n,l}}$. Then $\sup_{\mathcal{F}_n}|A(w,b)-A_n(w,b)| \leq \Delta < \varepsilon_0/2$, $\sup_{\tilde{\mathcal{F}}_{n,l}}|A_n(w,b)-A_l(w,b)| \leq \Delta_M < \varepsilon_0/2$ and finally $\sup_{\tilde{\mathcal{F}}_{n,l}}|A(w,b)-A_l(w,b)| \leq \Delta_M+\Delta < \varepsilon_0$, for n,l large enough with probability greater than $(1-\delta)^2 > 1-2\delta$. On the other hand, since $\mathcal{F}_n = \cup_l \mathcal{F}_{n,l}$ (4.2), setting $(w_{0,l},b_{0,l}) = \arg\min_{(w,b)\in\mathcal{F}_{n,l}} A(w,b)$ we have that $||(w_{0,l},b_{0,l})-(w_{0,n},b_{0,n})||$ converges to zero when n,l goes to infinity.

Then, with probability greater than $1-2\delta$, $||(w_{0,n},b_{0,n})-(w^*,b^*)|| \leq \delta(\varepsilon/2)$ and $||(w_{0,l},b_{0,l})-(\tilde{w},\tilde{b})|| \leq \delta(\varepsilon_0)$, for $\varepsilon < \varepsilon_0$, by applying the lemmas. Moreover, for any given ϵ, $||(w_{0,n},b_{0,n})-(w_{0,l},b_{0,l})|| < \epsilon$ for n,l large enough; therefore, using the triangle inequality we obtain $||(w^*,b^*)-(\tilde{w},\tilde{b})|| \leq \delta(\epsilon/2)+\delta(\epsilon)+\epsilon$. □

Finally, Lemma 1 and Theorem 2 can be used to improve bounds over the misclassification error given by $L(w,b) = P(sign(< w,\phi(X) > +b) \neq y)$.

Indeed, following [35], we have that $L(\tilde{w},\tilde{b}) \leq A_l(\tilde{w},\tilde{b}) + sup_{(w,b)\in\tilde{\mathcal{F}_{n,l}}}[A_n(w,b)-A_l(w,b)] + sup_{(w,b)\in\tilde{\mathcal{F}_{n,l}}}[A_n(w,b)-A(w,b)]$.

Then,

$$L(\tilde{w},\tilde{b}) \leq A_l(\tilde{w},\tilde{b}) + \Delta + \Delta_M.$$

Therefore, the unobserved theoretical misclassification error is bounded by the observed hinge loss function plus two approximation errors. The bound denoted by Δ_M can be further improved if the subsample is generated by the LSH-SVM algorithm. The improved bound is established in the following subsections by Theorems 4 and 5, for the LSH-SVM algorithm with random projections and directed projections, respectively.

5.3. Random Projections

In Theorem 2, the quantity $T_l(K)$ can be improved if we are able to control the characteristics of the random sample. The main idea behind random projections (Algorithm 1) is being able to improve this bound by selecting with high probability a sample over a set of data points belonging to B blocks satisfying $d(X_j,X_k) > d$ forj,k in different blocks and

some $d > 0$. In order to show this, we have the following lemma. We denote $\hat{\mathcal{S}}$ the sample created by Algorithm 1.

Lemma 4. *Assume kernel K satisfies [K1]. Let $0 < c, \delta$ be fixed constants. Set $p(c,\delta) := P(\max_{X,X' \in \hat{\mathcal{S}}}(K(\mathbf{a},X) - K(\mathbf{a},X')) > Bc/\delta)$, and for each $X \in \hat{\mathcal{S}}$ define $V_d(X) = \{X' \in \hat{\mathcal{S}} : \|\phi(X) - \phi(X')\|_{\mathcal{H}} \leq d\}$. Then*

$$P(\max_X |V_{c/2C_K}(X)| \leq N-1) \geq 1 - BN(1-\delta) - Np(c,\delta).$$

Proof. Recall LSH-SVM projecting Algorithm 1 creates the sample $\hat{\mathcal{S}}$ of $B\,N$ points, with N projection directions and B bins for each direction, where $r = \max_{X,X' \in \hat{\mathcal{S}}}(K(\mathbf{a},X) - K(\mathbf{a},X')/B$. By construction, for any given direction, $r > Bc/(\delta B)$ and then $c/r < \delta$ with probability at least $p(c,\delta)$. For $\mathbf{a} \sim N(0,I)$ consider the random variables $K(\mathbf{a},X)$ for any given X. Let $f_{\mathbf{a}}(X,X')$ be the density of the random variable $K(\mathbf{a},X) - K(\mathbf{a},X')$. The probability $P(\tilde{h}_{\mathbf{a},\theta}(X) = \tilde{h}_{\mathbf{a},\theta}(X')) = \int_0^r f_{\mathbf{a}}(X,X')(x)(1-x/r)dx$.

In particular, if $c_{\mathbf{a}} = |K(\mathbf{a},X) - K(\mathbf{a},X')|$, then conditional to $\mathbf{a}$, $P(\tilde{h}_{\mathbf{a},\theta}(X) = \tilde{h}_{\mathbf{a},\theta}(X')|\, c_{\mathbf{a}}) = \int_{c_{\mathbf{a}}}^r (1-\theta/r)d\theta$. If $c_{\mathbf{a}} \leq r$ then X, X' are in contiguous bins with probability $P = 1 - P(\tilde{h}_{\mathbf{a},\theta}(X) = \tilde{h}_{\mathbf{a},\theta}(X')) = \int_0^{c_{\mathbf{a}}}(1-\theta/r)d\theta = c_{\mathbf{a}}/r$. Then, conditional on $\mathbf{a}$, the event $\{|\tilde{h}_{\mathbf{a},\theta}(X')(X) - \tilde{h}_{\mathbf{a},\theta}(X')| = 1\} = \{\theta < c_{\mathbf{a}}\}$ and for any $0 < c < r$, with probability greater than c/r, $c < c_{\mathbf{a}} \leq \|\phi(X) - \phi(X')\|_{\mathcal{H}}\, \|\phi(\mathbf{a})\|_{\mathcal{H}} \leq C_K \|\phi(X) - \phi(X')\|_{\mathcal{H}}$. The last inequality by [K1].

In addition, if $|\tilde{h}_{\tilde{h}_{\mathbf{a},\theta}(X'),\theta}(X) - \tilde{h}_{\mathbf{a},\theta}(X')| \geq 2$ then $r < c_{\mathbf{a}} \leq C_K \|\phi(X) - \phi(X')\|_{\mathcal{H}}$ using the same bounds as before on kernel K.

It follows that with probability at least $1 - B\,N(1-\delta) - Np(c,\delta)$ selected points from contiguous bins in any given projection will be apart at least c/C_K. Given any $X \in \hat{\mathcal{S}}$ from projection j, for each projection $j' \neq j$, if there exists X' from projection j' with $\|\phi(X) - \phi(X')\|_{\mathcal{H}} < c/2C_K$ then $\|\phi(X) - \phi(X'')\| > c/2C_K$ for all other X'' from projection j'. Thus $\|\phi(X) - \phi(X')\|_{\mathcal{H}} \leq c/2C_K$ for at most $N-1$ points from $\hat{\mathcal{S}}$. □

Set $\hat{K}$ to be the kernel matrix defined over the sample $\hat{\mathcal{S}}$ and consider the corresponding minimization problem (12) with the feasible set denoted as $\hat{\mathcal{F}}_{n,l}$. Define $\hat{\Delta} := \sup_{(w,b) \in \hat{\mathcal{F}}_{n,l}} [A_n(w,b) - A_l(w,b)]$. We have the following result

Theorem 4. *Let $\hat{\mathcal{S}}$ be the sample obtained by Algorithm 1. Set $p_1 := 1 - max(1 - B\,N(1-\delta) - Np(c,\delta), 0)$. Assume [K1] and [K2] are satisfied. Set*

$$C_1 = \min(C(NC_K + N(B-1)\,\varepsilon(c(1-\delta)/2C_K)), R_{\hat{\mathcal{S}}}\sqrt{C_k}),$$

where $R_{\hat{\mathcal{S}}} = C\min(\sqrt{NB}\|\hat{K}\|_2^{1/2C_K}, \|\hat{K}^{1/2}\|_F)$, $T_1(K) = 2C_1 + 1$, and $\varepsilon(d)$ is defined in [K2]. Then, with probability greater than $1 - p_1 - \delta$,

$$\hat{\Delta} \leq \frac{4}{NB}\sqrt{Tr(\hat{K})} + T_1(K)\sqrt{\frac{1}{2}\log(1/\delta)\frac{n-NB}{n\;NB\;(1 - \frac{1}{2\max(NB,n-NB)})}}.$$

Proof. The sample $\hat{\mathcal{S}}$ is a subset of $\mathcal{S}$ satisfying the property that for each $v \in \hat{\mathcal{S}}$ there exist at most $N-1$ points at distance smaller than $r(1-\delta)$ over a set S_1 with $P(S_1) > 1 - \delta$. We now bound $P(\hat{\Delta} \leq \varepsilon) \leq P(\{\hat{\Delta} \leq \varepsilon\} \cap S_1) + P(S_1^c)$. The proof follows from bounding $P(\{\hat{\Delta} \leq \varepsilon\} \cap S_1)$. We use Theorem 1, and the bound $\sup_{w,b}(\|f_w + b\|_\infty + 1) \leq 2C_1 + 1$. For this, use that over the set S_1, $\max_j \sum_i |\hat{K}(X_i, X_j)| \leq NC_K + N(B-1)\,\varepsilon(c(1-\delta)/2C_K)$ by Lemma 4.

Then, we bound $E(\hat{\Delta}1_{S_1}) \leq E(\hat{\Delta})$ and use the bounds on the Radamacher averages following [35]. □

Remark 1. *By Lemma 4, matrix $\hat{K}$ will satisfy with high probability a block-like property. This improves the bounds over $\|\hat{K}\|_2$ from the case of a completely random sample.*

5.4. Directed Projections

Random projections (Algorithm 1) are effective because they are able to assure a minimum distance among most part of elements of the sample; however, it may happen that certain randomly chosen directions are non-informative: i.e., directions that are close to the normal of the optimal solution w^*. Algorithm 2, selects directions that are informative with high probability, i.e., close to the optimal solution w^*. This has a two-wise benefit. On the one hand, maintain a minimal distance among selected points, but on the other concentrate the sample over a set of points that are close to optimal separating hyperplane. Thus, by choosing a representative from each bin we are able to reduce the quantity of effective support vectors required to estimate the solution. Although our theoretical bounds are not as strong as we would like, they do offer a key to understanding why this method works better in certain cases.

Lemma 4 also holds for Algorithm 2 thus assuring the minimum distance property; however, Theorem 3 allows for a different approach. First we introduce some notation.

Let $Z_{i,j}(w) :=< w, \phi(X_i) - \phi(X_j) >$. Then, $|Z_{i,j}(w)| \leq \|w\|_{\mathcal{H}} \|\phi(X_i) - \phi(X_j)\|_{\mathcal{H}}$. In addition, for any given direction, w we have the distances are ordered: that is $|Z_{i,j}(w)| > r$ for 2 apart bins, $|Z_{i,j}(w)| > 2r$ for 3 apart bins and so on. Theorem 3 then yields that for the direction $\tilde{w}_{n_2}$ obtained by Algorithm 2, with probability greater than $1 - \delta$,

$$\|\phi(X_i) - \phi(X_j)\|_{\mathcal{H}} \geq (p-1)(r - \eta(\varepsilon))/\|w^*\|_{\mathcal{H}}$$

for points that are p bins apart. Bins that are contiguous are harder to bound so we just consider non-contiguous bins.

The rest of the approximation result is just as in the random projection case. This yields the following result whose proof is omitted as it is exactly as the proof of Theorem 4. Here the subsample is denoted as $\bar{\mathcal{S}}$, $\bar{K}$ is the kernel matrix defined over $\bar{\mathcal{S}}$ and we consider the corresponding minimization problem (12) with feasible set $\bar{\mathcal{F}}_{n,l}$. In addition, $\bar{\Delta}$ stands for the supremal among the original and approximated hinge loss functions. This is, $\bar{\Delta} := \sup_{(w,b)\in\bar{\mathcal{F}}_{n,l}} [A_n(w,b) - A_l(w,b)]$.

Theorem 5. *Let $\bar{\mathcal{S}}$ be the sample obtained by Algorithm 2. Assume [K1], [K2], [S1], and [S1'] are satisfied. Set*

$$C_2 = \min(C(3NC_K + \sum_{p=2}^{B-1} \varepsilon((p-1)(r-\eta(\varepsilon))/\|w^*\|_{\mathcal{H}})), R_{\bar{\mathcal{S}}}\sqrt{C_K}),$$

with $\varepsilon(d)$ from [K2], $T_2(K) = 2C_2 + 1$, and $R_{\bar{\mathcal{S}}} = C\min(\sqrt{NB}\|\bar{K}\|_2^{1/2}, \|\bar{K}^{1/2}\|_F)$. Then, with probability greater than $1 - \min((N+1)\delta, 1)$,

$$\bar{\Delta} \leq \frac{4}{NB}\sqrt{Tr(\bar{K})} + T_2(K)\sqrt{\frac{1}{2}\log(1/\delta)\frac{n - NB}{n\ NB\ (1 - \frac{1}{2\max(NB, n-NB)})}}.$$

Moreover, directed projections satisfy an additional property stemming from the fact that with high probability, the projections in the optimal direction w^* are zero for the actual support vectors. We have the following result.

Lemma 5. *Assume $r > 2$. Let w^* be the solution of problem (1). Then, $P(h_{\mathbf{w}^*,\theta}(X_j) = h_{\mathbf{w}^*,\theta}(X_i)) \geq \frac{r-2}{r}$ for any X_j, X_i support vectors. In any case, $P(|h_{\mathbf{w}^*,\theta}(X_j) - h_{\mathbf{w}^*,\theta}(X_i)| > 1) = 0$. This result applies to the feature space H. This is, $P(|\tilde{h}_{\mathbf{w}^*,\theta}(\phi(X_j)) - \tilde{h}_{\mathbf{w}^*,\theta}(\phi(X_i))| > 1) = 0$.*

Proof. If X_j is a support vector, $(w^*)^t X_j = y_j - \hat{b}$ with $y_j = 1$ or -1 and $h_{\mathbf{w}^*,\theta}(X_j) = \left\lfloor \frac{y_j - b^* + \theta}{r} \right\rfloor$. Thus for two support vectors X_i, X_j we have that $|(X_j - X_i)^t w^*| \leq 2$. Then, as in the proof of Lemma 4, use that for a given direction $P(h_{\mathbf{w}^*,\theta}(X_j) = P(h_{\mathbf{w}^*,\theta}(X_i)) = \frac{r - |Z_{i,j}(w^*)|}{r}$. □

Even though b^* is not known, Lemma 5, suggests looking at the central bins in order to select support vectors or points such that $\phi(X_j)$ is close to the separating hyperplane. Thus a reasonable guess based on projections is to eliminate the highest and lowest value bins. As we have seen, Lemma 3 assures points in these center bins will concentrate support vectors and points that are close to the margins of the separating hyperplane.

Thus, directed projections improve over random projections. The extent of this improvement is hard to assess though, and depends on the geometry of $\mathcal{S}$. Experimental results show a greater improvement for high dimensional problems with many support vectors for the optimal solution.

6. Concluding Remarks

In this article, we propose a novel methodology for dealing with classification problems in very large data sets using support vector machines. Our findings are that using projections based on Locality-Sensitive Hashing (LSH) can lead to improved estimation and fitting times by selecting a smaller yet significant data set for model training. Moreover, we show that previous knowledge based on the problem at hand, such as solving the problem over a small sample and then projecting in the normal direction to the obtained hyperplane, can further improve error rates in some cases. One byproduct of the proposed algorithms is that reducing the number of support vectors of the solution yields an important reduction in fitting times without affecting the overall error.

Although we restricted our attention to SVM, the approach here presented can be readily applied to other classification methods. Theoretical results, albeit not as conclusive as experimental results indicate, show improvement is related to better bounds over the target function class by eliminating less informative points, such as very close points or far from actual support vectors. This suggests that our method acts as a dimension reduction technique as measured by the decrease in the supreme of the infinity norm over the target class. An important related problem is considering sampling schemes not only for rows but also for columns in order to address problems with a huge number of features as well. Further research is necessary to fully understand how efficient column sampling can be achieved with similar heuristics, for example by sub-setting based on correlations with chosen support vectors over a smaller sample. Finally, since our method is embarrassingly parallel, future research will consider parallel implementation for further training time reductions for huge problems.

Author Contributions: These authors contributed equally to the theoretical, methodological, and computational aspects of this work. M.D.G.-L. work focuses on numerical optimization and support vector machines. C.C.L. work focuses on mathematical statistics and data mining. This paper is a result of complementing their areas of expertise. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Informed Consent Statement: Not applicable

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/ (accessed on 20 April 2022).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Bishop, C.M. *Pattern Recognition and Machine Learning*; Springer: Berlin/Heidelberg, Germany, 2006.
2. Vapnik, V. *The Nature of Statistical Learning Theory*; Springer: New York, NY, USA, 1995.
3. Cristianini, N.; Shawe-Taylor, J. *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*; Cambridge University Press: Cambridge, UK, 2000.
4. Bunke, H.; Neuhaus, M. *Bridging the Gap between Graph Edit Distance and Kernel Machines (Machine Perception and Artificial Intelligence)*; World Scientific Publishing Company: Hackensack, NJ, USA, 2007.
5. Osuna, E.; Freund, R.; Girosi, F. *Support Vector Machines: Training and Applications*; Technical Report A.I. Memo No. 1602, C.B.C.L. Paper No. 144; Massachusetts Institute of Technology: Cambridge, MA, USA, 1997.
6. Osuna, E.; Freund, R.; Girosi, F. Training support vector vector machines: An application to face detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Juan, PR, USA, 17–19 June 1997; pp. 130–136.
7. Platt, J. Fast training of support vector machines using sequential minimal optimization. In *Advances in Kernel Methods. Support Vector Learning*; Scholkopf, B., Burges, C., Smola, A., Eds.; MIT Press: Cambridge, MA, USA, 1998; pp. 41–65.
8. Fan, R.E.; Chen, P.H.; Lin, C.J. Working set selection using second order information for training support vector machines. *J. Mach. Learn. Res.* **2005**, *6*, 1889–1918.
9. Joachims, T. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In Proceedings of the 10th European Conference on Machine Learning (ECML-98), Chemnitz, Germany, 21–23 April 1998; Nédellec, C., Rouveirol, C., Eds.; Springer: Berlin/Heidelberg, Germany, 1998; pp. 137–142.
10. Lin, C.J. On the Convergence of the Decomposition Method for Support Vector Machines. *IEEE Trans. Neural Netw.* **2001**, *12*, 1288–1298. [PubMed]
11. Serafini, T.; Zanghirati, G.; Zanni, L. Gradient projection methods for quadratic programs and applications in training support vector machines. *Optim. Methods Softw.* **2003**, *20*, 353–378. [CrossRef]
12. Serafini, T.; Zanni, L. On the Working set selection in gradient-based descomposition techniques for support vector machines. *Optim. Methods Softw.* **2005**, *20*, 586–593. [CrossRef]
13. Zanni, L. An Improved Gradient Projection-based Decomposition Techniques for Support Vector Machines. *Comput. Manag. Sci.* **2006**, *3*, 131–145. [CrossRef]
14. Gonzalez-Lima, M.D.; Hager, W.W.; Zhang, H. An Affine-Scaling Interior-Point Method for Continuous Knapsack Constraints with Application to Support Vector Machines. *SIAM J. Optim.* **2011**, *21*, 361–390. [CrossRef]
15. Woodsend, K.; Gondzio, J. Exploiting Separability in Large Scale Linear Support Vector Machine Training. *Comput. Optim. Appl.* **2011**, *49*, 241–269. [CrossRef]
16. Jung, J.; Leary, D.O.; Tits, A. Adaptive constraint reduction for training support vector machines. *Electron. Trans. Numer. Anal.* **2008**, *31*, 156–177.
17. Ferris, M.C.; Munson, T. Interior point methods for massive support vector machines. *SIAM J. Optim.* **2002**, *13*, 783–804. [CrossRef]
18. Fine, S.; Scheinberg, K. Efficient svm training using low-rank kernel representations. *J. Mach. Learn. Res.* **2002**, *2*, 243–264.
19. Karatzoglou, A.; Smola, A.; Hornik, K.; Zeileis, A. kernlab—An S4 Package for Kernel Methods in R. *J. Stat. Softw.* **2004**, *11*, 1–20. [CrossRef]
20. Krishnan, S.; Bhattacharyya, C.; Hariharan, R. A randomized algorithm for large scale support vector learning. In Proceedings of the Advances in 20th, Neural Information Processing Systems, Vancouver, BC, Canada, 3–6 December 2008; pp. 793–800.
21. Balcazar, J.L.; Dai, Y.; Tanaka, J.; Watanabe, O. Fast training algorithms for Support Vector Machines. *Theory Comput. Syst.* **2008**, *42*, 568–595. [CrossRef]
22. Jethava, V.; Suresh, K.; Bhattacharyya, C.; Hariharan, R. Randomized algorithms for large scales SVMs. *arXiv* **2009**, arXiv:0909.3609.
23. Paul, S.; Boutsidis, C.; Mwagdon-Ismail, M.; Drineas, P. Random projections for Support Vector Machines. In Proceedings of the 16th International Conference on Artificial Intelligence and Statistics, Scottsdale, AZ, USA, 29 April–1 May 2013; pp. 498–506.
24. Camelo, S.; Gonzalez-Lima, M.D.; Quiroz, A. Nearest neighbors methods for support vector machines. *Ann. Oper. Res.* **2015**, *235*, 85–101. [CrossRef]
25. Montañés, D.M.; Quiroz, A.; Dulce, M.; Riascos, A. Efficient nearest neighbors methods for support vector machines in high dimensional feature spaces. *Optim. Lett.* **2021**, *15*, 391–404. [CrossRef]
26. Nalepa, J.; Kawulok, M. Selecting training sets for support vector machines: A review. *Artif. Intell. Rev.* **2019**, *52*, 857–900. [CrossRef]
27. Birzandi, P.; Kim, K.T.; Youn, H.Y. Reduction of training data for support vector machine: A survey. *Soft Comput.* **2022**, *26*, 1–14.
28. Indyk, P.; Motwani, R. Approximate nearest neighbors: Towards removing the curse of dimensionality. In Proceedings of the 30th Annual ACM Symposium on Theory of Computing, Dallas, TX, USA, 24–26 May 1998; pp. 604–613.
29. Yong, L.; Wenliang, H.; Yunliang, J.; Zhiyong, Z. Quick attribute reduct algorithm for neighborhood rough set model. *Inf. Sci.* **2014**, *271*, 65–81. [CrossRef]
30. Chen, Y.; Wang, P.; Yang, X.; Mi, J.; Liu, D. Granular ball guided selector for attribute reduction. *Knowl.-Based Syst.* **2021**, *229*, 107326. [CrossRef]

31. Mu, Y.; Hua, G.; Fan, W.; Chang, S.F. Hash-SVM: Scalable Kernel Machines for Large-Scale Visual Classification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Washington, DC, USA, 23–28 June 2014. [CrossRef]
32. Litayem, S.; Joly, A.; Boujemaa, N. Hash-Based Support Vector Machines Approximation for Large Scale Prediction. In *British Machine Vision Conference (BMVC)*; BMVA Press: Durham, UK, 2012; Volume 86, pp. 1–11.
33. Ju, X.; Wang, T. A Hash Based Method for Large Scale Nonparallel Support Vector Machines Prediction. *Procedia Comput. Sci.* **2017**, *108*, 1281–1291. [CrossRef]
34. Datar, M.; Indyk, P.; Immorlica, N.; Mirrokni, V. Localitity Sensitivity Hashing scheme based on p-stable distributions. In Proceedings of the 20th Annual Symposium on Computational Geometry, Brookling, NY, USA, 8–11 June 2004; pp. 253–262.
35. Boucheron, S.; Bousquet, O.; Lugosi, G. Theory of Classification: A Survey of Some Recent Advances. *ESAIM Probab. Stat.* **2005**, *9*, 323–375. [CrossRef]
36. Kumar, S.; Mohri, M.; Talwalkar, A. Sampling Methods for the Nyström Method. *J. Mach. Learn. Res.* **2012**, *13*, 981–1006.
37. Burges, C.; Crisp, D.J. Uniqueness of the SVM Solution. In *Advances in Neural Information Processing Systems 12*; Solla, S.A., Leen, T.K., Müller, K., Eds.; MIT Press: Cambridge, MA, USA, 2000; pp. 223–229.