Arbitrary-norm Separation by Variable Neighborhood Search

Gilles Caporossi   Pierre Hansen   Alejandro Karam

This draft: August 22, 2005

Abstract

We consider the problem of separating two sets of points in an Euclidean space with a hyperplane that minimizes the sum of p-norm distances to the plane of points lying on the "wrong" side of the plane. A Variable Neighborhood Search metaheuristic framework is used to determine the plane coefficients. For a set of examples with L_1-norm, L_2-norm and L_∞-norm, for which the exact solution can be computed, we show that our algorithm finds it in most cases, and gets good approximations in the others. The use of our heuristic solutions for problems in these norms can dramatically accelerate exact algorithms. Our method can be applied on very large instances that are intractable by exact algorithms. Since the proposed approach works for truly arbitrary norms (other than the traditional 1,2 and ∞), we can explore for the first time the effects of the choice of p on the generalization properties of p-norm hyperplane separation.

1 Introduction

Since Fisher's seminal work [Fis36], linear discriminant functions have been fundamental tools for automatic classification for several decades. Once the parameters of such models are defined, new observations can be classified with great ease and speed. This has made them, despite their limitations, popular for a wide variety of applications. Furthermore, by expansion or transformation of the original observation space, these techniques can be applied to non-linear discrimination.

The linear discriminant can be seen as a hyperplane that somehow separates the representations of the training observations of each of the two classes in an n-dimensional real space $\mathbb{R}^n$. The coefficients of the plane will provide the desired discriminant.

When the respective convex hulls of the two sets of observations intersect, perfect separation with the plane is impossible. As some points will necessarily lie on the wrong side of any plane, a natural idea is to choose the separating plane by minimizing some measure of the aggregate error implied by such misclassified points.

An intuitively appealing measure of this error is the sum of the distances to the plane of the misclassified points. As this distance is, in general, nonlinear, the resulting optimization problem is quite difficult. Alternative approaches have therefore been used, that preserve the linearity of the objective function, but in fact renounce the precise distance as the criterion. This is often done by minimizing some other measure of deviation of misclassified point with respect to the plane [Smi68, Gri72, FG81, Han81, LO90].

A recent vein in the literature has emerged that tackles explicitly the problem of minimizing the sum of distances to the plane. Analytical expressions for the distance in arbitrary L_p-norm from a point to a plane are derived in [Mel97], but [Man99] appears to be the first to apply them to the separating hyperplaneplane problem[1]. He shows that the L_1-norm case can be solved with $2n$ linear programs (each with about as many variables and as many constraints as there are training set observations), and suggests a formulation with a bilinear objective function and convex constraints for the L_2-norm case. The implications of these precise formulations for data mining applications is yet to be fully explored, and it is in this context that we consider our work.

In [AHK$^+$04a] a linear mixed integer formulation is presented for the L_∞-norm, and apply a branch and cut approach [AHJS00, Per04] to Mangasarian's formulation of the L_2-norm case.

These developments have been confined to the more usual cases of the L_1, L_2 and L_∞-norms, and except to some extent for the L_1-norm, are in practice unsuitable for application to large scale problems.

We here present a heuristic method that can be applied to a truly arbitrary (integer or fractional) norm, and that can scale up gracefully to relatively large instances. The general idea is to project the objective function (i.e. the sum of distances in the given L_p-norm, of misclassified points to the plane) to the surface of a Euclidean unit sphere representing the possible directions of the gradient of the plane. The heuristic search for a minimum is then performed over this sphere.

[1]Mangasarian also presents his own proof of the formulae. Another, perhaps simpler proof is given in [Pla]

The two key insights behind our approach are the following:

- The search for an optimal hyperplane can be decomposed in determining its direction and then, for a given direction, its position or offset with respect to the origin
- Once a direction is fixed, the conversion from the Euclidean distance to an arbitrary L_p-norm is just a constant rescaling

The procedure is implemented in two steps. We first construct a query function such that, given any point on the unit Euclidean sphere (i.e. any direction for the gradient of the candidate plane), determines the optimal position of the plane along that direction, and computes the sum of distances (in the given L_p-norm) of misclassified points to the best plane lying in that direction. In the second step, the optimization phase, the heuristic takes this function as an oracle and queries it repeatedly as it searches the surface of the sphere for the minimum.

Our implementation of the optimization phase uses Variable Neighborhood Search (VNS) [HM01b], an approach which has proved to be useful in a variety of difficult global optimization problems [BM96, CH97, HM97, BCK99, HMU00, MPKVC00].

Our numerical tests suggest that this algorithm often finds optimal or near optimal solutions. As shown in [AHK$^+$04b], these heuristic solutions can also be used to significantly accelerate exact methods.

Since the proposed approach works for truly arbitrary norms (other than the traditional 1,2 and ∞), we can explore for the first time the effects of the choice of p on the generalization properties of L_p-norm hyperplane separation.

The rest of the paper is organized as follows. Section 2 establishes the notation and explains the query function to be used as oracle in the search process. Section 3 summarizes the VNS approach and presents the search phase. Section 4 then summarizes our numerical results. The final section concludes.

2 The Oracle

Following the notation of [Man99], we denote $\mathcal{A}$ and $\mathcal{B}$ the two sets of points in $\mathbb{R}^n$ containing respectively m and k points represented by the matrices $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{k\times n}$. The discriminating hyperplane is denoted $P = \{x \,|w^t x = \gamma\}$,with $\gamma \in \mathbb{R}$, $w \in \mathbb{R}^n$, $w \neq 0$. Such plane can also be characterized by a set of $n - 1$ angles, $\alpha = (\alpha_1, \alpha_2, ..., \alpha_{n-1})$ which determine a direction perpendicular to the plane, and by an offset φ from the origin along this direction. The conversion from (α,φ) to the corresponding (w, γ) can be performed by an iterative procedure [Ken61] [Som58].

Once the training data set $\mathcal{A} \cup \mathcal{B}$ and the desired norm p are fixed, the input to the oracle is a direction α from the origin. We adopt the convention that the set $\mathcal{A}$ is meant to lie on the side of the plane to which w points.

The output PNORMDIST(α) is the sum of p-distances of misclassified points by the best possible plane along the given direction.

The algorithm is outlined in Figure 1.

The points in $\mathcal{A}$ and $\mathcal{B}$ are projected into the ray defined by α. Their distances to the origin are recorded and ranked (distances to the origin are signed, a negative value indicating the direction opposite to the origin with

```
• $missA \longleftarrow 0$      /* counter of missed points from $\mathcal{A}$ */
• $missB \longleftarrow |\mathcal{B}|$      /* counter of missed points from $\mathcal{B}$ */
• compute $distA$ and $distB$ /* vectors of projections to ray $\alpha$ */
• sort $distA$ and $distB$
• $sumdist \longleftarrow$ sum of all entries in $distB$
• $bestdist \longleftarrow sumdist$
• $bestpoint \longleftarrow$ first point
• $current \longleftarrow \min(\min(distA), \min(distB))$      /* initial position of plane */
• REPEAT
  −      move $current$ to next point
  −      update $missA$ , $missB$ and $sumdist$
  −      IF ($sumdist < bestdist$)
     *          $bestdist \longleftarrow sumdist$
     *          $bestpoint \longleftarrow current$
  −      ENDIF
  − UNTIL (all points considered)
• $pbest \longleftarrow bestdist/DualNorm$
• return $pbest$
```

Figure 1: Pseudocode for the $PNORMDIST(\alpha)$ oracle.

respect to w).

The candidate plane, perpendicular to the ray, is initially positioned at the first point along the ray, i.e. the one with minimum distance. All the observations start thus on the same side of this plane, with $\mathcal{A}$ being in the correct half-space, while all $\mathcal{B}$ points are misclassified. The corresponding counters $missedA = 0$ and $missedB = |\mathcal{B}|$ are initialized. The initial sum of distances of misclassified points $sumdist$ corresponds to the distance of all points in $\mathcal{B}$ to the initial plane.

The plane is then moved along the ray, in the direction of α, to the position corresponding to each successive point. The distance between successive projections and the updated counters $missedA$ and $missedB$ are used to compute the new *sumdist*. The minimum distance *best* found along the way is recorded. The position where this minimum occurs is also noted.

By equation (7) in [Man99], the conversion of this value to the corresponding sum of distances in p-norm is obtained by dividing it by the dual norm $\|w\|_p'$ of the gradient w that corresponds to the given α. For $1 < p < \infty$, $\|w\|_p' = \|w\|_q$ where q is such that $\frac{1}{p} + \frac{1}{q} = 1$. We therefore compute it as

$$\|w\|_p' = \sqrt[\frac{p-1}{p}]{\sum w^{\frac{p}{p-1}}} \ .$$

We also recall that $\|w\|_1' = \|w\|_\infty$ and vice versa, and deal with these cases separately.

The adjusted value $p{-}best$ is then the minimum sum of L_p-norm distances of points misclassified by a plane in direction α. This value is returned by the oracle.

The oracle function $PNORMDIST(\alpha)$ can then be queried by the optimization routine. Note that the angles (in radians) given by α are automatically reduced modulo 2π within the oracle[2]. This implies that the optimization routine can search over the unconstrained domain $\alpha \in \mathbb{R}^{n-1}$.

Sorting the distance vector is theoretically the bottleneck step in the oracle function. We use the C++ standard library function *sort* for this task[3]. The distances for A and B are stored and sorted separately to speed this step.

[2]The projections of the points to the plane are computed using the standard trigonometric functions.

[3]Modern implementations of the C++ standard library use *introsort*, which has a worst case $O(n \log n)$ complexity, but is known to do much better on average [Mus97].

3 The Heuristic Search

Variable Neighborhood Search is a metaheuristic framework that has been successfully applied to a variety of difficult optimization problems. The basic methodology and several extensions can be found in [MH97], [HM99], [HM01b], and [HM01c]. Applications to unsupervised classification include [HM01a], [BHM02] and [HJM98].

In the realm of supervised classification, [GTMM03] use VNS to deal with combinatorial challenges arising in variants and refinements of the k-nearest neighbors approach.

A crucial insight underlying VNS is the observation that in many practical optimization problems with multiple local minima, these tend to be somehow clustered or correlated. Under these circumstances, a local minimum might contain information useful to find other, perhaps better, minima. The general idea in VNS is thus to try to escape from local minima by first exploring subsets of the domain that are in some sense "close" to the incumbent, and then going on to test for solutions that are increasingly different to it.

Among the family of metaheuristic frameworks, VNS has a number of advantages that in our view make it suitable for the optimization phase of this data mining problem. It is simple and transparent, with a very small number of parameters; it is also quite flexible, allowing for any local search procedure and objective function to be used without changing the general framework.

The only two exogenous ingredients for the basic version of VNS are a metric defined on the solution domain and a local descent procedure. The metric is used to build a problem-dependent structure consisting of K ordered neighborhoods, $\mathcal{N}_1(\alpha^*), \mathcal{N}_2(\alpha^*), ..., \mathcal{N}_K(\alpha^*)$ centered around the current incumbent solution point α^*. The first neighborhood $\mathcal{N}_1(\alpha^*)$ includes only solutions near the incumbent and subsequent neighborhoods include regions farther from it.

A local descent procedure is applied starting from a randomly chosen point within $\mathcal{N}_1(\alpha^*)$, and then from increasingly farther neighborhoods until a better solution is found or the stopping criterion is met. If an improved solution is found, the whole structure is recentered around it and the process restarts.

For the search over $\mathbb{R}^{n-1}$ in our problem, we adopt the following neighborhood structure:

$$\mathcal{N}_1(\alpha^*) = \prod_{i=1}^{n-1} \left[\alpha_i^* - \frac{1}{2K}, \alpha_i^* + \frac{1}{2K}\right]$$

$$\mathcal{N}_j = \left\{\prod_{i=1}^{n-1} \left[\alpha_i^* - \frac{j}{2K}, \alpha_i^* + \frac{j}{2K}\right]\right\} \setminus \bigcup_{l=1}^{j-1} \mathcal{N}_l(\alpha^*) \quad \text{for } j = 2, ..., K$$

where $\prod$ denotes Cartesian product. A unit hyperbox is thus centered on α and sliced into K successive symmetric layers with equal thickness. Note that, the $\mathcal{N}_j(\alpha^*)$ being mutually exclusive, they do not correspond to topological neighborhoods[4].

Also note that the volume of each $\mathcal{N}_j(\alpha^*)$ is increasing in j. Since at each iteration one point for restarting the local search is drawn at random

[4] $\mathcal{N}_1$, however, would be a neighborhood under the L_1-norm, in the topological sense of the word.

```
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
• Get initial point $\alpha^*$
• $best \leftarrow \infty$
• REPEAT
    – $k \leftarrow 0$
    – REPEAT
        ∗ Get random candidate $\alpha \in N_k(\alpha^*)$
        ∗ Call local search; returns new local min
        ∗ $new = PNORMDIST(\hat{\alpha})$
        ∗ IF $(new < best)$
            · $best \leftarrow new$
            · $\alpha^* \leftarrow \hat{\alpha}$
            · $k \leftarrow 0$
        ∗ ENDIF
        ∗ ELSE $k \leftarrow k + 1$
    – UNTIL $(k = K)$
• UNTIL (stop criterion met)
• return $\alpha^*$ and $best$
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

Figure 2: Pseudocode for the VNS heuristic.

from each neighborhood, the implication is that the search is relatively more intensive near the incumbent. Figure 2 summarizes the algorithm.

The input to the local descent function is the starting point α and the output is $\widehat{\alpha}$, the new minimum found. $PNORMDIST(\widehat{\alpha})$, the objective function evaluated at this point, is thus the sum of distances of misclassified points to the best plane in direction $\widehat{\alpha}$.

For the local descent during the optimization phase, we use the downhill simplex method of [NM65]. Our implementation is roughly based on the structure suggested in [PFTV88].

The first neighborhood structure is arbitrarily centered at the origin. We use as stopping criterion for VNS the number of runs through all neighborhoods without improvement[5].

We now present the results of our numerical experiments.

4 Experimental Results

We tested our algorithm on several series of random problems, as well as on a set of real life instances from the UCI Repository [CM98]. For moderately sized instances, exact solutions can be obtained for p values of 1, 2 and ∞. We first compare our heuristic's performance to these exact results, and then explore the algorithms's speed on larger problems, for which exact benchmarks are impossible or impractical to obtain.

Although known exact methods can be applied to moderately sized problems in 2-norm and ∞-norm, and to some extent to larger problems for the 1-norm, for the case of a truly arbitrary, integer or fractional $p > 1$, there are no available general exact methods. An interesting issue for practical data mining applications is whether a choice of p results in better generalization for a class of problems. Our heuristic approach allows this issue to be tackled. In the final part of this section, we study the effects of using a range of values for p on the generalization properties of the resulting planes, as estimated by k-cross validation on a set of UCI problems.

[5]Unless otherwise indicated, we set the limit of passes without improvement at 6, i.e. the algorithm stops if it has visited all neighborhoods six times without improvement. This yielded reasonable results in moderate time in most of our test problems.

4.1 Benchmarks to Known Exact Solutions

For a set of real life instances from the UCI Repository and two sets of artificial problems, we compare the heuristic solutions found by our algorithm for the L_1-norm, L_2-norm and L_∞-norms to the corresponding exact solutions.

For the L_1-norm case, the exact solutions were found by solving $2n$ linear programs, as suggested in [Man99]. We solved these $2n$ LPs sequentially[6], but accelerated the solution process by adding at each step the constraint that the objective value be at most the best found so far. This technique can save a number of phase II solutions and reduces considerably total computation time.

The exact L_2-norm and L_∞-norm solutions were found with the techniques proposed in [AHK$^+$04b].

The random problems were created with D. Musicant's NDC generator [Mus98], which produces normally distributed clusters. This generator is publicly available and has been used in other discrimination studies (e.g., [FM01, MM00]).

For the first series of random problems, we fixed the dimension at 6 and explored the effects of increasing the number of points from 2000 to 20000 (by steps of 2000). As the exact solution in L_1-norm is much easier than for the other norms, we were able to extend the series to 100000 by increments of 20000 for this case.

We then generated a second series of problems with 2000 points, with dimensions ranging from 4 to 13. For each problem size, we generated 10

[6]As the $2n$ linear programs are independent, a parallel implementation would be possible.

instances and report the corresponding mean values of the results.

The parameters used for the generation of the artificial sets, as well as the preprocessing steps for the real life instances from the UCI Repository are detailed in Appendix A.

Tables 1, 2 and 3 present, respectively, the results for the Irvine datasets and each of the two families of artificial problems. The gap columns shows the percentage above the exact minimum of the objective function value found by the heuristic. The fit column shows the full-set accuracy of the discriminant (i.e. the percentage of points correctly classified by the corresponding hyperplane). The computation time in seconds is shown in the CPU column. Results are shown for the L_1-norm, L_2-norm and L_∞-norms.

			L1-norm					L2-norm					L∞-norm				
			VNS			EXACT		VNS			EXACT		VNS			EXACT	
problem	dim	obs	gap %	fit	CPU	fit	CPU	gap %	fit	CPU	fit	CPU	gap %	fit	CPU	fit	CPU
cancer	9	683	0.00%	96.78%	5.3	97.07%	0.08	0.50%	97.07%	6.2	97.07%	156	0.01%	97.37%	7.6	97.51%	1.1
echocardiogram	6	74	0.00%	75.68%	0.5	77.03%	0.01	0.03%	75.68%	0.6	75.68%	5	0.00%	72.97%	0.5	77.42%	0.1
glass	9	214	2.33%	95.79%	3.3	95.79%	0.03	0.79%	94.86%	3.5	95.33%	5	0.82%	94.39%	2.8	95.79%	0.4
hepatitis	16	150	0.60%	88.00%	4.3	90.00%	0.06	1.17%	86.67%	4.4	88.00%	14,181	3.49%	89.33%	7.4	90.00%	65.7
housing	13	506	0.17%	85.77%	28.8	84.58%	0.14	0.57%	84.19%	13.9	84.39%	550	4.50%	86.76%	13.2	88.14%	30.4
pima	8	768	0.00%	72.79%	4.7	73.05%	0.42	0.00%	75.39%	4.4	75.39%	612	0.00%	76.04%	4.4	76.04%	8.1

Table 1: UCI instances.

Exact solution times are not reported for the problems in L_2-norm for 12 and 13 dimensions, since these exact solutions could not be obtained within the predetermined maximum allowed CPU time without the heuristic bound[7].

[7]The use of heuristic solutions to accelerate exact solution in L_2- and L_∞-norms is discussed in [AHK$^+$04b].

As far as the objective function is concerned, the heuristic finds the exact solution in most of our datasets, and is close on many of the cases where it does not. As we can see from the results for *housing* and *hepatitis*, as well as the last lines of table 2, performance deteriorates on dimensions above about 10, but appears to be robust to the number of observations[8]. The *glass* problem, although only in 9 dimensions, appears to have a very large number of local minima under the L_1-norm, and the heuristic fails to find the optimum within the limits set by our stopping criterion. Interestingly, the full set fit of the heuristic plane is the same as that of the exact solution.

For the L_1-norm, the gap was zero in nine out of the 10 problems with 18000 observations in 6 dimensions, while the remaining one failed by almost 13%; the heuristic found the exact solution for all but one of the problems with 80000 observations, which was missed by an (abysmal) 32%.

The small but systematic apparent disadvantage of the heuristic solution in full set accuracy for some problem sets appears to be due to round-off and precision differences between our algorithm and the exact solver.

For this range of problem sizes, the computing time for our heuristic appears to be linear in the number of observations, and quadratic in the dimension[9]. The time gains are of course most dramatic with respect to the L_2-norm, which is the hardest of the three to compute exactly. For the problems with 2000 observations in 11 dimensions, the heuristic found the optimum in about $\frac{1}{600}^{th}$ of the time of the exact algorithm. Exact solutions

[8]Poor performance on higher dimensions can be traced to the limitations of the local search technique, the Nelder-Mead simplex search.

[9]Least squares fit of the model $CPUtime = \beta_0 + \beta_1 * dim + \beta_2 * dim^2$ yields an R^2 of .977 with $\beta_0 = 17.0918$, $\beta_1 = -5.09$ and $\beta_2 = 0.6314$.

under the L_1-norm are of course the easiest to obtain, and the heuristic approach is not worthwhile for small problems (such as our Irvine examples and problems with up to 10000 observations. However, even under the L_1-norm, the heuristic looks advantageous in all instances larger than 20000 observations, and the difference reaches a full order of magnitude for 100000 points, with very little loss of average accuracy.

		L1-norm					L2-norm					L∞-norm				
		VNS			EXACT		VNS			EXACT		VNS			EXACT	
dim	obs	gap %	fit	CPU	fit	CPU	gap %	fit	CPU	fit	CPU	gap %	fit	CPU	fit	CPU
4	2000	0.00%	94.33%	8.5	94.36%	0.4	0.00%	94.41%	10.5	94.41%	5	0.00%	94.29%	8.3	94.31%	0.8
5	2000	0.00%	93.68%	7.6	93.70%	0.5	0.00%	93.83%	11.1	93.84%	11	0.00%	93.69%	7.1	93.71%	1.5
6	2000	0.00%	92.95%	7.5	92.98%	0.6	0.00%	93.21%	13.9	93.23%	34	0.00%	93.11%	6.9	93.14%	3.9
7	2000	0.51%	90.98%	11.3	90.92%	1.0	0.00%	91.73%	21.8	91.75%	118	0.00%	91.49%	9.4	91.54%	8.8
8	2000	0.53%	89.29%	15.3	89.46%	1.4	0.00%	90.13%	29.2	90.16%	557	0.00%	89.63%	14.0	89.69%	20.6
9	2000	0.01%	89.85%	22.8	89.96%	1.6	0.00%	90.13%	44.3	90.17%	1,796	0.05%	89.92%	21.4	90.06%	49.4
10	2000	0.01%	88.86%	27.5	88.94%	1.9	0.00%	89.51%	56.8	89.55%	3,194	0.00%	89.37%	28.2	89.46%	102.8
11	2000	0.00%	82.32%	43.4	82.45%	3.4	0.00%	83.65%	64.0	83.74%	38,531	0.02%	83.48%	31.4	83.59%	330.9
12	2000	2.09%	87.61%	49.0	88.25%	2.8	0.01%	88.87%	78.5	89.02%	N/A	0.04%	88.41%	42.3	88.58%	530.5
13	2000	10.12%	84.49%	53.7	84.71%	4.0	0.01%	86.02%	93.8	86.15%	N/A	0.27%	85.56%	49.7	85.73%	1324.6

Table 2: Artificial problems with fixed number of observations.

		L1-norm					L2-norm					L∞-norm				
		VNS			EXACT		VNS			EXACT		VNS			EXACT	
dim	obs	gap %	fit	CPU	fit	CPU	gap %	fit	CPU	fit	CPU	gap %	fit	CPU	fit	CPU
6	2000	0.00%	93.12%	8.1	93.14%	0.7	0.00%	93.14%	14.6	93.15%	36	0.00%	93.10%	7.1	93.13%	4
6	4000	0.00%	92.16%	15.2	92.17%	3.0	0.00%	92.22%	28.3	92.22%	213	0.00%	91.90%	13.5	91.92%	15
6	6000	0.21%	91.08%	18.5	91.21%	8.9	0.00%	91.59%	34.2	91.60%	636	0.00%	91.51%	16.5	91.53%	41
6	8000	0.00%	91.65%	36.0	91.66%	16.1	0.00%	92.27%	54.9	92.28%	960	0.00%	92.24%	33.8	92.25%	65
6	10000	0.00%	91.22%	50.5	91.22%	27.3	0.00%	91.04%	77.7	91.04%	1,433	0.00%	91.08%	45.2	91.08%	126
6	12000	0.00%	90.70%	35.4	90.71%	44.2	0.00%	91.01%	69.6	91.01%	2,593	0.00%	90.94%	32.9	90.94%	204
6	14000	0.00%	89.34%	40.3	89.35%	62.6	0.00%	89.68%	74.9	89.69%	2,858	0.00%	89.56%	37.2	89.58%	321
6	16000	0.00%	92.09%	79.1	92.10%	54.3	0.00%	92.25%	112.4	92.25%	2,146	0.00%	92.21%	70.6	92.22%	288
6	18000	1.27%	92.57%	57.8	92.61%	77.5	0.00%	92.80%	115.5	92.80%	5,916	0.00%	92.82%	52.1	92.82%	391
6	20000	0.00%	94.48%	58.1	94.48%	96.7	0.00%	94.59%	118.5	94.59%	5,777	0.00%	94.49%	57.1	94.49%	368
6	40000	0.00%	92.18%	193.28	92.18%	528.4										
6	60000	0.00%	95.06%	162.25	95.06%	1000.7										
6	80000	3.23%	91.58%	233.16	91.84%	2039.2										
6	100000	0.00%	90.51%	306.83	90.51%	4384.7										

Table 3: Artificial problems with fixed dimension.

4.2 Test on Larger Problems

We explored empirically the computation time of our algorithm on larger problems, for which exact benchmarks are unavailable. In order to control for problem difficulty, we constructed an additional set of nine test problems, with a simpler structure than those used for benchmarking accuracy[10]. The dimension was fixed at 10, and the number of observations was increased from 200000 to one million, by steps of 100000.

Figure 3 shows the CPU time of the heuristic (for the Euclidean norm) as function of problem size. For this range of problem sizes, computation time still appears to grow linearly; the instance with one million points in ten dimensions took a little under two hours to run.

We used for these experiments the same stopping criterion as in the accuracy benchmarks discussed in the previous section. However, for all of these large problems, the heuristic solution that's eventually reported is found very quickly (within the first 3% to 5% of the total time), the rest of the time being spent spanning the neighborhoods without further improvement. In our example with one million points, the final solution was found in only 4.5 minutes.

We suspect this phenomenon to be due to the fact that in problems with simple structures, with a large number of points in moderate dimensions, the effect of each single point on the surface of the objective function is relatively small, and deep, narrow valleys are unlikely to exist. Under these circum-

[10]The problems were constructed by fixing two arbitrary centers, assigning one half of the points to each and generating independent columns from a normal distribution for each of them.

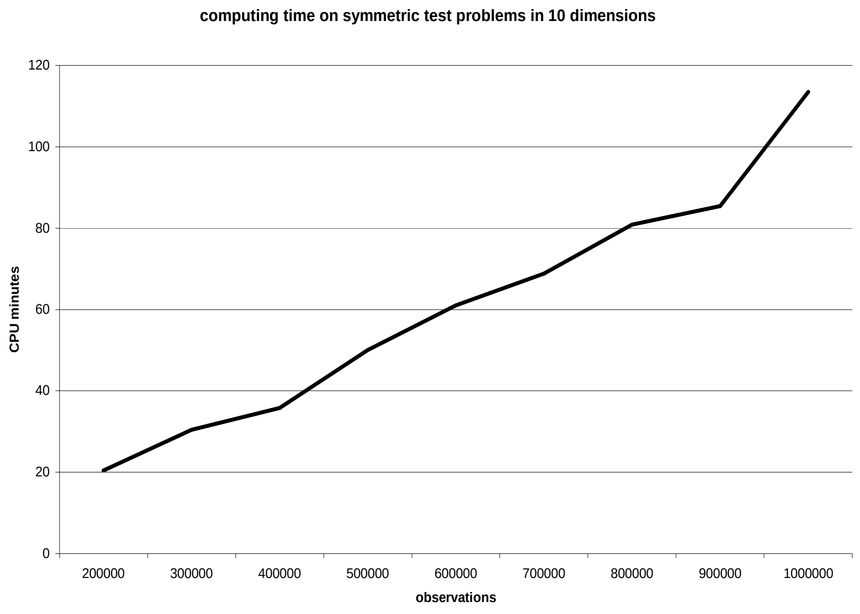

Figure 3: Running time for large problems.

stances, just a few runs of the local descent lead to the global minimum. This suggests that, when this kind of problem is identified, and complex clusters of local minima are not present, the use of our oracle with a suitable descent routine can provide a good, quick answer even in fairly large instances.

4.3 Classification Accuracy with Arbitrary Norm

We now turn to the study of the effect on classification accuracy of the choice of the norm p for the UCI datasets. Our experimental set up is as follows. We compute the 10-cross validation accuracy of separating hyperplanes that minimize the p-norm distance of misclassified points, as found by our heuristic, for 30 values of p, starting from 1 with increments of .2. In order to smooth out the effect of the partition, we repeat the process 10 times for each data set[11].

As accuracy was more important than speed for this exercise, the stopping criterion for VNS runs was raised to 10 passes through all neighborhoods without improvement. As we did not expect the optimal planes to vary dramatically for nearby values of p, the VNS search was initiated at the best point found for the previous value of p considered[12].

The results are plotted in figures 4 to 7. The lighter lines represent the 10-cross validation accuracies obtained with each partition; the darker line is the average. For *echocardiogram, glass* and *hepatitis*, there seems to be no systematic effect of the value of p on testing set classification accuracy. For *cancer*, despite some fluctuations, it appears that the planes with $p = 1$ perform generally at least as good as any others.

Results for the *housing* data set show an increasing trend on p, with average accuracy raising from 82% to 84% as p goes from 1.6 to 6.8. For the *pima* data set the effect is quite remarkable: for all the partitions considered, accuracy is lowest under the L_1-norm and increases until about $p = 3$.

The choice of p appears thus to be relevant in some, but not all instances. A practical implication is that a range of values (including fractions) might be tried when L_p-norm separation is being considered.

[11]we considered alternative approaches to deal with this issue. Performing 20- or 30-cross validation would have been a reasonable choice only for the larger datasets, whereas leave-one-out was computationally impractical. Averaging over several partitions with 10-cross validation appeared to be a solution appropriate for all the datasets.

[12]the case $p = 1$, for which no previous solution is available, was started at the origin

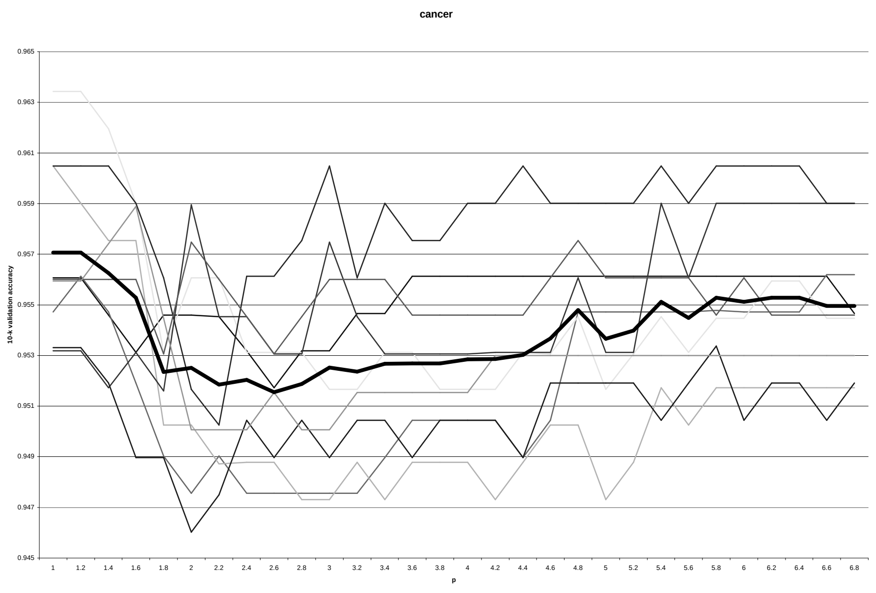

Figure 4: 10-fold validation accuracy vs. norm chosen: *Cancer* Dataset.

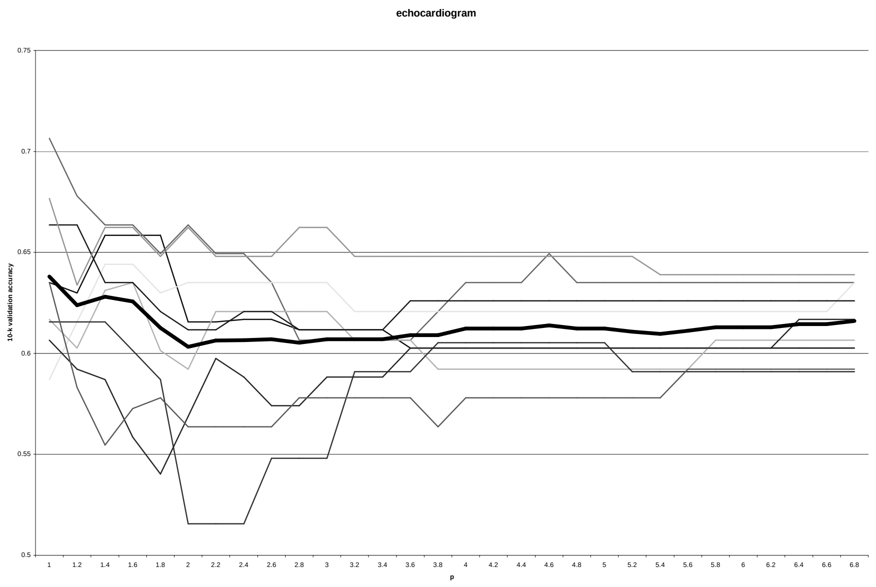

Figure 5: 10-fold validation accuracy vs. norm chosen: *Echocardiogram* Dataset.

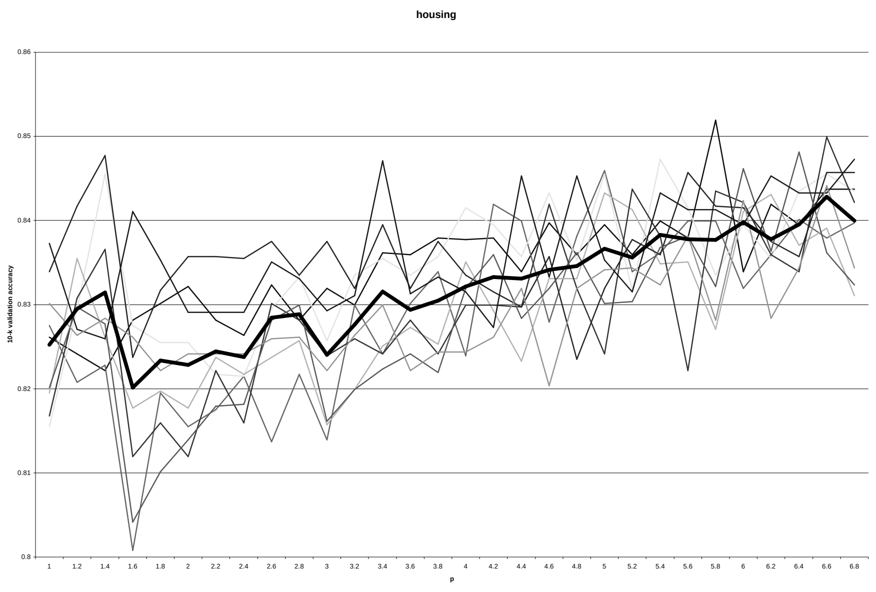

Figure 6: 10-fold validation accuracy vs. norm chosen: *Housing* Dataset.

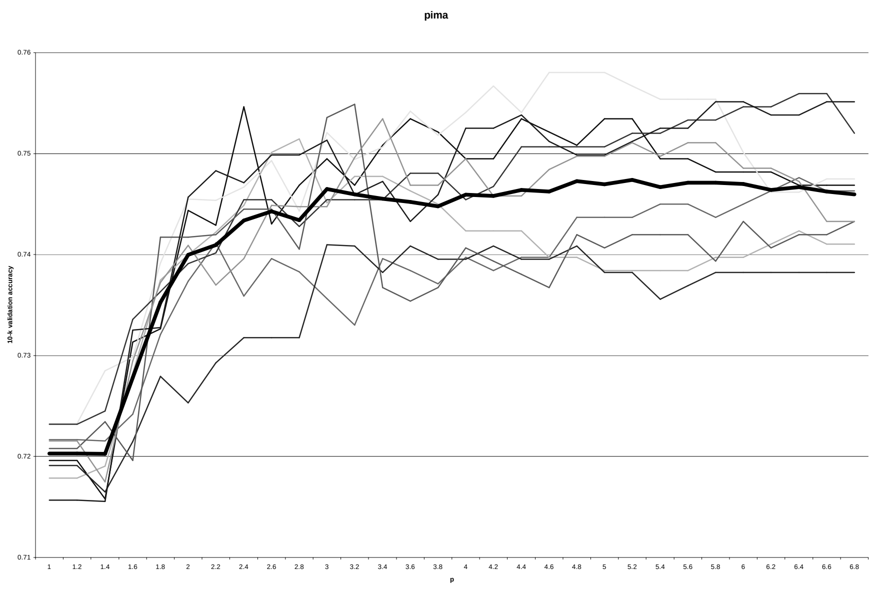

Figure 7: 10-fold validation accuracy vs. norm chosen: *Pima* Dataset.

5 Conclusion

Our heuristic approach for the L_p-norm separating hyperplane problem appears to be very promising for data mining applications for which this criterion is chosen. It is able to find good solutions in reasonable time, for large instances in up to about 10 dimensions. The quality of the solutions found deteriorates in higher dimensions but appears to be robust to the number of observations. In moderate dimensions, the algorithm scales very gracefully (apparently linearly) to very large problems.

The proposed method can furthermore deal with problems for which no practical method for exact solution is known, namely when a truly arbitrary p is considered. Our exploration of the generalization properties of L_p-norm separation with respect to the choice of p suggest that there are classes of problems for which the norm matters, while for others the resulting hyperplane is essentially the same regardless of the norm.

Possible avenues for further work on the optimization phase include exploring the use of local search procedures other than the downhill simplex method within VNS, or of metaheuristic approaches other than VNS.

A Data Set Details

We considered the instances from the UCI Machine Learning Repository [CM98] which either have only two classes or could be readily converted into a binary classification problem. We then retained those with very few or no categorical variables.

Cancer refers to the Wisconsin Breast Cancer database. Rows with missing attributes were deleted.

Pima refers to the Pima Indians Diabetes database.

For the *Echocardiogram* problem, all instances with missing labels were deleted, and missing attributes were replaced by the corresponding class means.

For the *Glass* database the two classes considered were window versus non-window sources.

Housing refers to the Boston Housing database.

In the *Hepatitis* database, all observations with more than 6 missing attributes were deleted, as were columns 16 and 18, which had too many missing entries. Missing observations were then replaced with column means (if continuous) or modes. Column 3 trivially separates the set, and was also removed.

Musicant's NDC generator is a matlab program. It locates randomly a given number of centers, assigns them to one of two classes by splitting the set with a randomly generated plane, and then produces multivariate normal observations from these centers, using a randomly generated covariance matrix. This approach provides some more generality than one might have with other common practices. Note, however, that even within the class of normally distributed problems, some reasonable, interesting configurations (such as having a small cluster centered on the "wrong" side of the plane) are not spanned by NDC. These limitations are inevitable in any exercise with artificial data, and we feel that replicability is facilitated with the use of a publicly available generator. The number of centers was made to be equal to the dimension of the problem and the dispersion parameter $nExpandFactor$

was fixed at 15.

All databases were linearly standardized to the range $[0, 1]$.

The original files are available at http:XXXXX SAY WHERE

References

[AHJS00] C. Audet, P. Hansen, B. Jaumard, and G. Savard. A branch and cut algorithm for nonconvex quadratically constrained quadratic programming. *Math. Program.*, 87(1, Ser. A):131–152, 2000.

[AHK$^+$04a] C Audet, P Hansen, A Karam, C NG, and S Perron. Exact solution of l_∞-norm and l_1-norm plane separation. Technical Report G-2004-84, GERAD, november 2004.

[AHK$^+$04b] C. Audet, P. Hansen, A. Karam, C.T. Ng, and S. Perron. Arbitrary-norm plane separation by variable neighborhood search. *submitted to ...*, 2004.

[BCK99] E.K. Burke, P. Cowling, and R. Keuthen. New local and variable neighborhood search heuristics for a sequencing problem in printed circuit board assembly applied to the travelling salesman problem. Technical report, University of Nottingham, 1999.

[BHM02] N. Belacel, P. Hansen, and N. Mladenović. Fuzzy J-means: a new heuristic for fuzzy clustering. *Pattern Recognition*, 35(10):2193–2200, 2002.

[BM96] J. Brimberg and N. Mladenović. A variable neighborhood algorithm for solving the continuous location-allocation problem. *Studies in Locational Analysis*, 10:1–12, 1996.

[CH97] G. Caporossi and P. Hansen. Variable neighborhood search for extremal graphs 1: The autographix system. *Les Cahiers du GERAD*, G-97-41, 1997. Also in: Discrete Mathematics 212:29-44, 2000.

[CM98] J.M. Christopher and P.M. Murphy. UCI repository of machine learning databases, 1998.

[FG81] Ned Freed and Fred Glover. A linear programming approach to the discriminant problem. *Decision Sciences*, 12(1):68, 1981.

[Fis36] R.A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.

[FM01] G. Fung and O.L. Mangasarian. Proximal support vector machine classifiers. In *Knowledge Discovery and Data Mining*, pages 77–86, 2001.

[Gri72] Richard Grinold. Mathematical programming methods of pattern classification. *Management Science (pre-1986)*, 3:272–290, 1972.

[GTMM03] M. García-Torres, J. A. Moreno Pérez, and J. Marcos Moreno Vega. Vns para clasificacin supervisada. In *MAEB 2003*, pages 343–352, Gijón, Spain, 2003.

[Han81] D. J. Hand. *Discrimination and Classification.* John Wiley & Sons, 1981.

[HJM98] P. Hansen, B. Jaumard, and N. Mladenović. Minimum sum of squares clustering in a low dimensional space. *Journal of Classification*, 15:37–56, 1998.

[HM97] P. Hansen and N. Mladenović. Variable neighborhood search for the p-median. *Location Science*, 5:207–226, 1997.

[HM99] P. Hansen and N. Mladenović. An introduction to variable neighborhood search. In S. Voss et al., editor, *Metaheuristics: Advances and Trends in Local Search Paradigms for Optimization*, pages 433–458. Kluwer, 1999.

[HM01a] P. Hansen and N. Mladenović. J-means: A new local search heuristic for minimum sum-of-squares clustering. *Pattern Recognition*, 34(2):405–413, 2001.

[HM01b] P. Hansen and N. Mladenović. Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, 130:449–467, 2001.

[HM01c] P. Hansen and N. Mladenović. Variable neighbourhood search. In P. Pardalos and M. Resende, editors, *Handbook of Applied Optimization.* Oxford University Press, 2001.

[HMU00] P. Hansen, N. Mladenović, and D. Urosevic. Variable neighborhood search for the maximum clique problem. *Les Cahiers du GERAD*, G-2000-45, 2000.

[Ken61] M.G. Kendall. *A Course in the Geometry of n Dimensions*. Charles Griffin, London, 1961.

[LO90] Chau-Kwor Lee and J. Keith Ord. Discriminant analysis using least absolute deviations. *Decision Sciences*, 21(1):86, 1990.

[Man99] O. Mangasarian. Arbitrary-norm separating plane. *Operations Research Letters*, 24(1–2):15–23, 1999.

[Mel97] E. Melachrinoudis. An analytical solution to the minimum $\mathcal{L}_p$-norm of a hyperplane. *Journal of Mathematical Analysis and Applications*, 211:172–179, 1997.

[MH97] N. Mladenović and P. Hansen. Variable neighborhood search. *Computers & Operations Research*, 24:1097–1100, 1997.

[MM00] O.L. Mangasarian and D.R. Musicant. Active support vector machine classification. In *NIPS*, pages 577–583, 2000.

[MPKVC00] N. Mladenović, J. Petrovic, V. Kovacevic-Vujcic, and M. Cangalovic. Solving spread spectrum radar polyphase code design problem by tabu search and variable neighbourhood search. *Les Cahiers du GERAD*, G-2000-60, 2000. appeared in EJOR 151, 389-399.

[Mus97] David R. Musser. Introspective sorting and selection algorithms. *Software Practice and Experience*, 27(8):983–993, 1997.

[Mus98] D.R. Musicant. NDC: normally distributed clustered datasets, 1998.

[NM65] J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

[Per04] S. Perron. *Applications jointes de l'optimisation combinatoire et globale.* PhD thesis, École Polytechnique de Montréal, 2004.

[PFTV88] William H. Press, Brian Flannery, Saul Teukolosky, and William Vetterling. *Numerical recipes in C : the art of scientific computing.* Cambridge University Press, Cambridge, 1988.

[Pla]

[Smi68] Fred W Smith. Pattern classifier design by linear programming. *IEEE Transactions on Computers*, C-17:367–372, 1968.

[Som58] D.M.Y. Sommerville. *An introduction to the geometry of n dimensions.* Dover Publications, New York, 1958.