Department of Computer Science
University of Toronto
http://learning.cs.toronto.edu

6 King's College Rd, Toronto
M5S 3G4, Canada
fax: +1 416 978 1455

August 1, 2005

UTML TR 2005–004

Novelty Detection Model Selection Using Volume Estimation

Edward Meeds
Department of Computer Science, University of Toronto

Abstract

In this paper, we present an approach to selecting models for novelty (outlier) detection. Our approach minimizes the risk of accepting outliers at a fixed normal rejection rate, under the assumption that the distribution of abnormal (outlier) data is uniformly distributed in some bounded region of the input space. This risk is minimized by selecting the model with the smallest volume acceptance region, using a randomized volume estimation algorithm. The volume estimation algorithm can estimate the volume of a body in high-dimensional space and scales polynomially in dimension with the number of calls to the model. We have performed extensive experiments which show that the combined model selection criteria are able to select not only the best models from a given model class, but also among all model classes.

Novelty Detection Model Selection Using Volume Estimation

Edward Meeds
Department of Computer Science, University of Toronto

1 Introduction

The objective of novelty detection is to model normal (typical) data and to detect novel (atypical, outlier) data in such a way that a large percentage of normal data are accepted and most, if not all, novel data are rejected. Since this objective can be viewed as a trade-off between the risk of rejecting normal data and accepting outlier data, optimizing the combined risks is a common approach ([31]; [28]). Our view is that the user sets the parameter $\nu^\star$, the desired normal rejection rate, and, subsequently, the risk of accepting outliers is minimized. We therefore attempt to build the best possible boundary between a small $\nu\%$ of the normal data density and the remaining $1 - \nu\%$ of the normal data density. Such a discriminative perspective of novelty detection has led to the development of so-called one-class classification algorithms ([23]; [30]) and has been supported theoretically by [28], who demonstrated the equivalence between detecting *density levels* and classification. However, since there are many possible boundaries that will reject $\nu\%$ of normal data, the problem of model selection is paramount.

Model selection in the unsupervised learning setting is more difficult and subjective than in the supervised learning setting because we only have examples from the normal dataset. In this paper, model selection in the unsupervised learning setting is discussed with particular focus on novelty detection. We are interested in developing a model selection approach which is general to a large set of model classes, under the assumption that the acceptance regions of the novelty detection models are bounded in input space. Within each model class, ranking the quality of novelty detection solutions is possible using a number of model-class specific heuristics (see Section 2). However, since the user can build a novelty detection model by employing different model classes, having a general measure of quality is essential if all solutions from all model classes are to be ranked. Our goal is to develop general model selection criteria that are model-class independent.

This paper presents a definition of novelty detection consisting of two precise criteria for model selection. Practical methods for satisfying these criteria are also proposed. But before discussing the problem of model selection in novelty detection, we present a brief overview of novelty detection applications.

1.1 Novelty detection applications

Novelty detection problems occur primarily when we only have access to "normal" data, or when very little outlier data exists and it does not represent the entire outlier distribution. Intrusion detection systems (IDS), for example, have access to large quantities of data from the normal operation of a computer network, and possibly data from a known list of network attacks. It is possible to learn to discriminate between normal network data and data from an attacked network, but this will leave the network vulnerable to new attacks. In this case, it is not known how to obtain sufficient outlier data, because its distribution is unknown and continuously evolving. For other problems, it may be possible to generate outlier data, but to do so would be too costly or dangerous. An obvious example is modeling the normal operation of a nuclear power plant: generating meltdown situations for the sake of detecting meltdowns is not plausible. In both these examples, it is more sensible to model the abundant normal data and to flag rare data using a novelty detection model. This paper addresses the problem of selecting the best models for any given novelty detection scenario.

Novelty detection is an important problem that is applicable to a wide spectrum of fields, from biology to machine monitoring. Biological applications include brain imaging ([20]; [4]; [29]) and perceptual salience detection ([22]; [18]). Computer network intrusion detection ([37]; [39]) and text ([1]; [36]; [40]; [17]) are two other important application areas of novelty detection. Industrial process monitoring is the most common novelty detection application and it impacts business at several levels. The first level is the detection of a fault or damage to a piece of equipment ([35]). The next level involves predicting faults or wear in equipment so that appropriate intervention can be taken to reduce the chance of more costly consequences in the future. Condition monitoring and predictive maintenance are two such problem domains that are closely related to fault detection. For example, breakout detection in the steel industry is one application of condition monitoring ([2]); the airline industry has now begun to use intelligent predictive maintenance systems ([32]).

The remainder of this paper is divided into six sections: (2) model selection and novelty detection, (3) volume estimation, (4) experiments, (5) results, (6) discussion, and finally, (7) conclusion. In Section 2, we will outline the problem of model selection as it pertains to novelty detection, propose two criteria to solve it, and compare our criteria with other approaches. Our second criterion requires computing the volume of the acceptance region defined by a novelty detection model; in Section 3 we will describe several algorithms to compute volumes, from naive approaches to randomized algorithms based on Markov chain Monte Carlo sampling methods. In Section 4, we will present two sets of experiments: the first will test the volume estimation algorithm on objects of known volume, and the second will test the proposed model selection criteria using multiple model classes and several datasets, and compare our approach to other model selection heuristics. In Section 5 we will present the results of these experiments and in Section 6 we will discuss the strengths and weaknesses of our approach, including sources of error and uncertainty, and propose future work. In Section 7 we will conclude.

2 Problem setup: model selection and novelty detection

Despite the importance of novelty detection applications in industry and other fields, performance measures for novelty detection model selection are lacking. This applies to unsupervised learning in general:

> In the context of unsupervised learning, there is no direct measure of success. It is difficult to ascertain the validity of inferences drawn from the output of most unsupervised learning algorithms. One must resort to heuristic arguments not only for motivating the algorithms, ... but also for judgments as to the quality of the results. This uncomfortable situation has led to a proliferation of proposed methods, since effectiveness is a matter of opinion and cannot be verified directly ([8], pp 439).

For supervised learning problems, the measure of success is well-defined. In most cases, one wants to minimize a loss function over test data, assumed to be generated from the same distribution as the training data (i.e. the data are i.i.d.). Example loss functions are the 0/1 loss (classification) and squared error (regression). Both loss functions require targets: class labels for classification and continuous targets for regression. Model selection is inherently difficult in unsupervised learning due to the lack of such targets. After presenting our criteria, we will compare them to other common model selection heuristics for unsupervised learning.

2.1 Selection Criteria

We now describe our criteria for selecting a novelty detection model. The first criterion ensures that the normal class is correctly modeled.

Criterion 1 (C1) *Given a set of models $M = \{m_1, m_2, ..., m_t\}$, we want to select models $S \subseteq M$ such that*

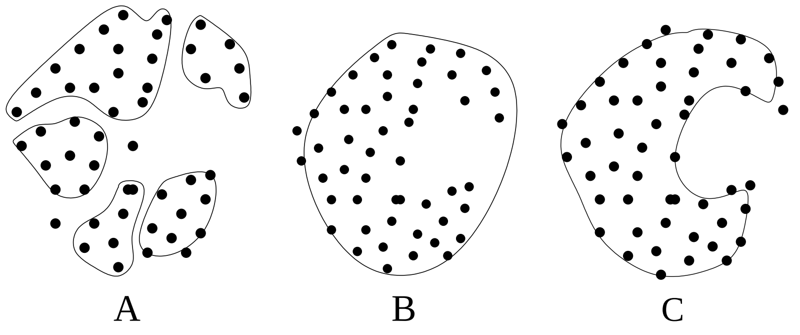
$$(1-\alpha)\nu^\star \leq E_{D_+}(m_i(x)) \leq (1+\alpha)\nu^\star,\ \forall\, m_i \in S,$$

where m_i is a model function that returns 0 for a rejected point, and 1 for an accepted point; $0 < \alpha < 1$; and D_+ is the distribution of normal points. The parameters α and $\nu^\star$ are defined a priori by the user.

This criterion selects the models whose expected rejection rate on normal data is within a small fraction α of the desired rejection rate $\nu^\star$. Since we can never evaluate $E_{D_+}(m_i(x))$ exactly, we can approximate the expectation by $\hat{\nu}$, so that C1 becomes:

$$(1-\alpha)\nu^\star \leq \hat{\nu} \leq (1+\alpha)\nu^\star,$$

Figure 1: Three possible novelty detection solutions are illustrated. On the left, the model has overfit. Although the model may reject only ν% of the training data, it will reject a much higher fraction of normal test data, and will not satisfy C1 (see text). On the other hand, both B and C satisfy C1. To select model C, we must estimate the volume of the acceptance regions of models B and C. After satisfying C2 (see text), we can select C, the best model.



where α is ideally chosen to be small. In our experiments, $\hat{\nu}$ is estimated using cross-validation.

The importance of C1 is illustrated by three possible novelty detection solutions (Figure 1). Model A shows a very small and compact representation of the data which is overfit, leaving 'holes' in the acceptance region. The $\hat{\nu}$ for this model is high and does not satisfy C1. We are left with models B and C. In both cases $\hat{\nu}$ satisfies C1; our goal is to select the intuitively more appealing model C. For this we need some notion of outliers. If the outlier distribution is known *a priori* then we can build a classifier to separate normal from outlier input data. But in general, we do not know the locations of the outlier data regions, therefore we make the following assumption:

Assumption (A1) *Abnormal data is distributed uniformly in some bounded region of the input space (D_-). Also, the normal data is contained in the same bounded region.*

Of all the models that satisfy C1, we want to keep the model with the lowest risk of accepting an abnormal test point (a false positive). If the abnormal data are assumed to be uniformly distributed in a bounded region, then the model with the smallest volume will be optimal. Given A1, we now present our second and final model selection criterion:

Criterion 2 (C2) *Select model $m_{r^\star} \in S$, where S is the set of all models satisfying C1, which is the solution to*

$$\begin{aligned} m_{r^\star} &= \arg\min_{m_r} E_{D_-}(m_r(x)) \\ &= \arg\min_{m_r} \int_{D_-} m_r(x)\, dx, \end{aligned}$$

This implies choosing the model with the smallest volume acceptance region measured in the input space.

Because we have assumed that outlier data are uniformly distributed in a bounded region, minimizing the volume of the acceptance region in input space will minimize the expected false acceptance rate. C2 therefore minimizes the false acceptance rate of abnormal data under A1.[1]

Combining C1 and C2 results in a satisfying definition of an "optimal" novelty detection model: it is stable with respect to the normal class and simultaneously has the lowest risk of misclassifying an outlier when no examples of the outlier class are available.

The practical implications of these criteria are significant. Most importantly, there is little constraint on the model class. Under these criteria, a model is essentially a black-box function which returns 1 for a test point from the normal class and 0 for an outlier, which represents the acceptance and rejection

[1]Note that others have also made the same assumption (see [28] and [31]).

regions of the input space, respectively. We can therefore use any model class with a bounded acceptance region. For example: a Gaussian Mixture Model (gmm) (see [6] for an overview), a Parzen windows model ([21]), a kmeans model (see [6] for an overview), or a one-class Support Vector Machine (**ν-svm**) ([23]; [30]).

Another related benefit of these criteria is their independence with respect to the feature space in which the user models the data. Since C2 minimizes the volume of the acceptance region in the *input* space, any model which preprocesses the input space, or projects it into lower or higher dimensions, will receive a volume estimate of the same input space. One can therefore compare novelty detection models constructed from multiple model classes and any parameterization within each model class.

2.2 Comparison with other model selection methods

Although our model selection criteria are general with respect to model classes, to evaluate our approach we can compare it with model-class specific heuristics.

For **ν-svm** models, we compare two heuristics for selecting the optimal radial basis function (rbf) kernel width $\sigma^\star$. One heuristic is sorting the models in ascending order of σ, and selecting the $\sigma^\star$ from the first model for which $\hat{\nu} \approx \nu^\star$ ([34]). We call this the *increasing* σ method. This makes intuitive sense and conforms with our criteria: as σ increases, so to does the "size" of the acceptance region. Once the acceptance region is large enough to model the normal data, we stop. Continuing to increase σ will only increase the acceptance region and the risk of accepting outliers. This heuristic is generally applicable to models who have parameters which directly affect the "size" of the acceptance region. Parzen windows is another model class for which this is true.

The second heuristic for selecting **ν-svm** parameters uses a naive sampling technique to measure the "size" of the acceptance region ([31]). This approach is applicable to any bounded novelty detection solution. The heuristic works as follows. Outliers are sampled uniformly from inside a hypersphere that encloses the acceptance region. The number of artificial outliers which are inside the acceptance region are counted. The ratio of this count to the total number of samples gives an estimate of the ratio between the volume of the acceptance region and the volume of the enclosing hypersphere. A volume estimate of the acceptance region could then be computed by multiplying the ratio by the volume of the hypersphere. Tax and Duin report that the method works well when the dimension is twelve or less, but that it fails for higher dimensions because the ratio estimate is zero. Although this approach will produce a volume estimate of the acceptance region, such a naive approach will require an extremely large number of samples to produce a reliable volume estimate. Their method falls into the category of naive Monte Carlo estimation procedure, and as such scales very poorly in dimension. We will discuss naive volume estimation algorithms in Section 3.[2]

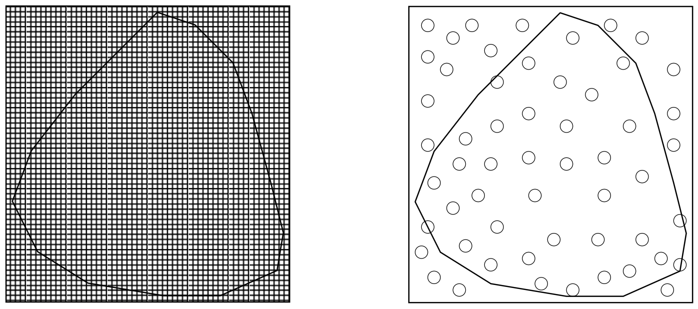
[33] present a procedure for estimating the number of clusters in a dataset. The 'gap' statistic measures the difference in the expected log of the pairwise squared distances on a reference distribution and the pairwise squared distances of the training points. The reference data is generated either uniformly from the the range of the training data, or uniformly from a hyperbox which has been aligned with the principal components of the data. The number of clusters k is chosen in such a way that the gap of k+1 is greater or equal to the gap(k). This approach is only applicable to clustering algorithms, for example kmeans. The number of clusters can also be chosen using the *within* cluster variance; this is a common quality of clustering measure (see [6] for this and others).

For density estimation models, the simplest way of selecting the best model is by computing the cross-validated likelihood for the training set. For example, cross-validating the likelihood of a gmm will detect overfitting, because, although adding mixtures will increase the likelihood for the entire (normal) training set, left-out cross-validation data will be modeled poorly. For gmms this heuristic can be used to select the number of components (e.g. [27]), and for Parzen windows it can be used to select the width parameter σ. Likelihood on the entire training set can be used to select, for a fixed number of components and width σ, the model with the best local optimum. This applies to gmms, but not to Parzen windows, because they have no local optima.

Our proposed method for estimating the volume of the acceptance region represents a major portion of this paper. As such, the rest of the paper is devoted to describing volume estimation algorithms, modifying an existing algorithm to work properly with novelty detection, its practical implementation, and extensive experiments utilizing and illustrating both C1 and C2.

[2] [5] also reasoned that measuring the volume of the acceptance region, independently of the model class and parameterization, would result in a useful model selection heuristic. However, they do not implement such a metric.

Figure 2: Simple volume estimators. **Left:** the grid method. The surrounding hyperbox is divided into a fine grid. The ratio between the number of small boxes which fall into K , and the total number of boxes, gives an estimate of the ratio between K and the enclosing hyperbox. **Right:** a naive Monte Carlo method. The same approach as the grid method is taken, except that instead of using boxes from a fine grid to estimate the volume ratio, points sampled uniformly from the enclosing hyperbox are used.



3 Volume estimation

In this section, we will describe the difficulties of volume estimation in high-dimensional spaces, followed by a brief description of a tractable, randomized algorithm for volume estimation. Finally, the algorithm used in our experiments will be described in detail.

Before describing the recent advances in volume estimation algorithms, we define some of the notation used by the volume estimation community. Let K denote a body in $\Re^n$ whose volume we wish to compute. In addition, K is normally assumed to be convex; however, in our experiments, many of the models tested will have non-convex acceptance regions. K is also known as an *oracle* function ([10]), which, in our setup, is a black-box function that returns 1 for a point inside the body, and 0 for a point outside. Essentially, an oracle "knows" when a point is inside K. For the model classes we will use, K is represented by a well-defined oracle; one that knows, for *all* points, whether the point is inside K or not ([24]).

3.1 Volume estimation in high dimensions

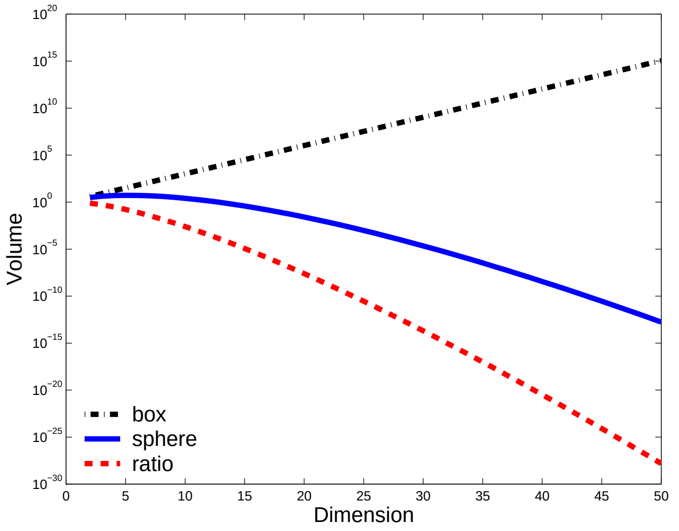
The fundamental obstacle preventing efficient volume estimation is the curse of dimensionality. This is clear from looking at two simple methods for volume estimation (Figure 2) ([24]). The first method is deterministic and can compute the volume of K to within arbitrary precision. The body K is surrounded by a hyperbox which is divided into ϵ-size boxes, and the number of the boxes that intersect with K are counted. The ratio of the number of boxes inside K to the number of total boxes gives an estimate of the ratio between the volume of K and the hyperbox. The volume is computed by multiplying the ratio by the volume of the hyperbox.

The second method is a naive Monte Carlo estimation procedure. As before, the body K is surrounded by a hyperbox. Instead of dividing the hyperbox into small boxes, points are sampled uniformly from within the hyperbox. The ratio of the number of points falling in K to the total number of sampled points gives an estimate of the ratio between the volume of K and the hyperbox. The volume of K is computed by multiplying the ratio by the volume of the hyperbox. A similar ratio trick is applied by the randomized volume estimation algorithms described later in this section.

These methods get hopelessly inefficient as the dimension increases. The first method must test an exponentially increasing number of cubes. What is worse, the volume of K gets exponentially small relative to the hyperbox. In the first approach this requires a smaller ϵ, and in the second this means one must sample a vast number of points, because each point has a very small probability of falling into K. These difficulties are illustrated in Figure 3. The volume of the unit hypercube, unit hypersphere, and their ratios are plotted versus dimension.[3] The volume of the hyperbox increases exponentially, and the volume of the hypersphere decreases exponentially. The ratio betwen the two approaches zero very fast, forcing naive Monte Carlo methods to sample huge numbers of points to estimate the volume of even these simple objects.

[3]Using a hypersphere to enclose K, rather than a hyperbox, would produce a tighter fit in high dimension. However, the ratio problem, while alleviated, would still exist.

Figure 3: The volume of a hypercube of width two, a hypersphere of radius one, and their ratios for dimensions 2 to 50. Note that the distance from the center of an edge of a hypercube of width two is one, thus tightly enclosing a hypersphere of radius one.



In 1989, volume estimation of a high-dimensional convex body was proved possible in $O^\star(n^{23})$ calls to the oracle, with the publication of “A random polynomial time algorithm for approximating the volume of convex bodies” ([7]).[4] Dyer, Frieze, and Kannan showed that by using a randomized algorithm—a multiphase Markov chain Monte Carlo method—the volume of a convex body K in dimension n could be computed with probability of at least 3/4 to within an ϵ of the true volume of K, in polynomial in dimension calls to the oracle. By repeatedly running the algorithm, and taking the median result, one can produce, with high probability, an estimate that is within ϵ of the true volume. The order of the polynomial (23), and the constants involved, however, made any practical implementation very difficult.

Fortunately, there have been a series of improvements made to the original algorithm and analysis ([14]; [15]; [13]), which have reduced the complexity to $O^\star(n^4)$ ([16]). In Section 3.2 we will present a volume estimation algorithm without its sampling details. In Section 3.3 we will discuss the problem of sampling uniformly in K, with focus on the sampling methods used in our experiments: a practical modification of Hit-and-Run (HR) ([25]) called Artificial centering Hit-and-Run (ACHR) ([11]).

3.2 Multiphase Markov chain Monte Carlo volume estimation

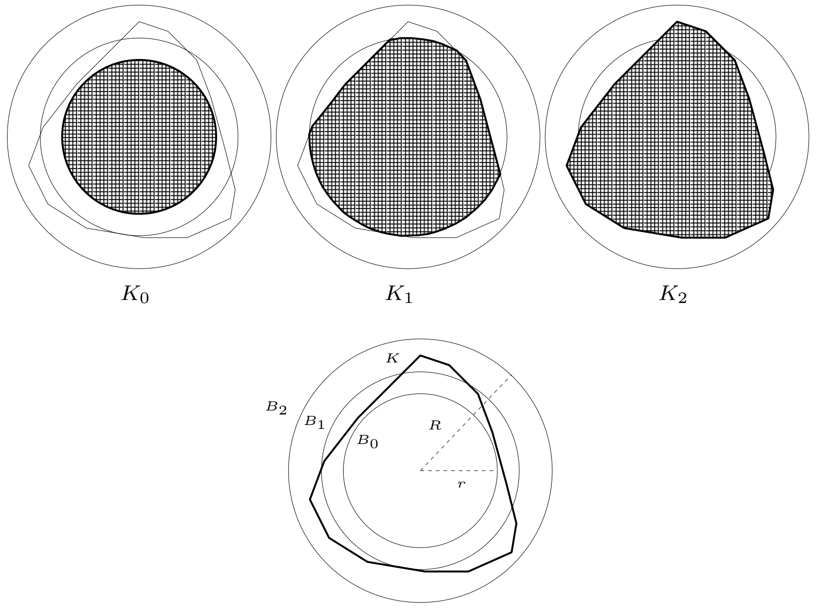
All the randomized algorithms mentioned in the previous section use the same core algorithm, multiphase Markov chain Monte Carlo (MP-MCMC), for the actual volume calculation; the differences are mainly due to sampling techniques.

One of the shortcomings of using a simple Monte Carlo technique for volume estimation is that the ratio of K to the enclosing hyperbox/hypersphere decreases exponentially. A point sampled from the enclosing region will have very small chance of falling into K. A multiphase technique avoids this problem by ensuring that the ratio between the enclosing region and K is large, for example between one and two. This means that points sampled uniformly from the enclosing region will have high probability of falling in K.

Figure 4 illustrates the MP-MCMC algorithm which is described in Figure 5. First, a ball B_0 is fitted into K which is as large as possible. Next, a ball B_m is fitted around K which is as small as possible. Dilations of B_0, called B_1 to B_{m-1}, are constructed such that each dilation's volume is twice that of the previous dilation. Then N points are sampled uniformly from the intersection of K and balls B_1 to B_m (denoted as K_1 to K_m). For each K_i, the number of these points that fall into the intersection of K and previous dilation, K_{i-1}, is counted (T_i). The fraction N/T_i is the estimate of the ratio between

[4] $O(\cdot)^\star$ notation refers to the number of calls to the oracle. This is the accepted way of presenting algorithm complexity in the volume estimation literature. The actual complexity includes constants and other terms of lower order, but the number of calls to the oracle is the greatest inefficiency.

Figure 4: MP-MCMC setup. The upper diagrams diagrams represent an instantiation of MP-MCMC, defined by the initialization illustrated in the bottom diagram. Given the center of K, the radius of the enclosed and enclosing ball are computed: r and R, respectively. To calculate the volume estimate, we compute the ratios between K_i and K_{i-1}, for $i = 1$ to m.



the volume of K_i and K_{i-1}. Because the dilations were specified to be twice the volume of the previous dilation, the ratio is bounded as:

$$1 \le \frac{vol(K_i)}{vol(K_{i-1})} \le 2.$$

Since $vol(K_0) = vol(B_0)$ and $vol(K_m) = vol(K)$, we can write the volume estimate as a product of volume ratios and $vol(K)$:

$$\begin{aligned}
vol(K) &= vol(K_0) \cdot [\frac{N}{T_1} \cdot \frac{N}{T_2} \cdots \frac{N}{T_m}] \\
&= vol(K_0) \cdot \prod_{i=1}^{m} \frac{vol(K_i)}{vol(K_{i-1})} \\
&= vol(K_0) \cdot \frac{vol(K_1)}{vol(K_0)} \cdot \frac{vol(K_2)}{vol(K_1)} \cdot \cdots \cdot \frac{vol(K_{m-1})}{vol(K_{m-2})} \cdot \frac{vol(K_m)}{vol(K_{m-1})} \\
&= vol(K_m) \\
&= vol(K)
\end{aligned}$$

3.3 Sampling

In this section, we will discuss how to sample uniformly from a body in high dimension. At each phase of the MP-MCMC algorithm, a sampling algorithm must supply N points uniformly distributed in K_i. Before describing actual random walks on K_i, we will clarify three of the important concepts and difficulties of random walks in high dimension: step size, sandwiching, and convexity.

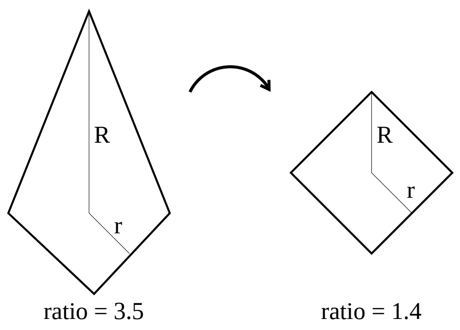
To sample efficiently, a random walk should explore K as quickly as possible. This is easily achieved by setting the step size large. The downside of setting the step size large is that, at each step, the walk has a high probability of stepping outside K. Near the edge of the body, for instance, it can take many attempts to remain in K. High-dimensional corners, although difficult to actually reach, can be extremely difficult to escape ([24]).

Algorithm 1: MP-MCMC VOLUME ESTIMATION

```
inputs
    $K =$ Oracle in $\Re^n$
    $N =$ number of points to sample per phase.
initialization
    $m = \lceil n \log_2 n \rceil$
    $B_0 = B(\mathbf{0}, 1)$
for i = 1 : m
    $B_i = B(\mathbf{0}, 2^{i/n})$
    $K_i = K \cap B_i$
    $S_i = Sample(N, K_i)$
    $T_i = |S_i \cap K_{i-1}|$
output
    $vol(K) = vol(K_0) \cdot \prod_i \frac{N}{T_i}$
```

Figure 5: The MP-MCMC VOLUME ESTIMATION algorithm for computing the volume of a body K which has been transformed (rounded) such that B($\mathbf{0}$,1) fits inside K, and B($\mathbf{0}$,n) encloses K ([7], [24]). Note that B($\mathbf{0}$,n) defines a ball with the zero vector as the center and having radius n. Our implementation is a modified version of this algorithm; see Section 4 for details.

Figure 6: Sandwiching (rounding). The body on the left is transformed into the body on the right via an affine transformation. As a result, the sandwiching ratio decreases from 3.5 to 1.4.

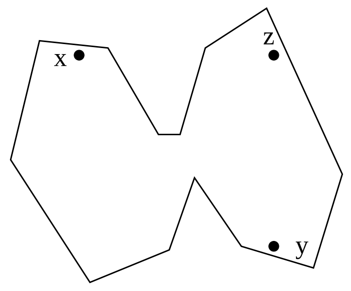


Step size is directly related to the concept of local conductance. Local conductance is the probability of a point remaining inside K after taking a step ([10]). In the interior of K, the local conductance almost one (nearly every step will remain inside), but it decreases near the boundary. The goal of many algorithms is to sample from bodies that have high *average* local conductance, so that the fewest number of points are rejected per sample. A large step size implies lower local conductance. Decreasing the step size will increase the local conductance at the expense of increasing the number of random steps needed to explore K. In our experiments, step size is not an issue because our sampling method (ACHR) does not take steps in the sense discussed above.

Sandwiching, or rounding, is an important preprocessing step for many samplers ([10]). Rounding K can be accomplished by an affine transformation of K, such that the ratio of the radii R and r is as small as possible (see Figure 6). When the sandwiching ratio is one, for example, we know that the body is spherical: the two radii are equal. By improving the sandwiching ratio, the local conductance improves. This is not only an important initialization step for sampling proofs, but also leads to improved practical efficiency ([9]). In our experiments, sandwiching is not performed; instead, we will use a sampling method that is robust to large sandwiching ratios.

All randomized volume estimation algorithms assume that K is a convex body. Convexity is important for these algorithms mainly to prove that a random walk from point A can reach any other point C, just as easily as from point B, and for any A, B, and C in K. Consider two convex bodies in high dimension

Figure 7: A non-convex body. In this example, a random walk starting at point x will only reach z after a very large number of steps. This is mainly due to the low probability of passing from the region where x is located to the region where y and z are located. In high-dimensional spaces, the gap between the two regions may be very small, making random walks very slow to converge (i.e. they are not rapidly mixing). A point y, on the other hand, can reach z much more easily because they are both in the same convex region.



that have small intersection (see Figure 7). For point x to reach z by a random walk, it must find a path through the intersection. As the dimension increases and the intersection shrinks, the probability of x reaching z becomes much smaller than the probability of y reaching z. If we guarantee convexity, this problem is alleviated. In other words, a random walk must *rapidly* approach a uniform walk on K, which is much easier to accomplish if K is convex ([24]). Imagine two disjoint regions. Obviously a random walk starting from one region will have zero probability of reaching the other region if its step size is small. Unfortunately, the acceptance regions of our models are not guaranteed to be convex. In fact, many of our models will generate non-convex or even disjoint regions. But as we will see below, HR-based random walks can sample uniformly from non-convex bodies.

3.4 Random Walks

We now describe, in more detail, the random walks used in the volume estimation literature. The grid walk was used in the original randomized volume estimation algorithm paper ([7]). Improvements to the volume estimation algorithms were made by changing the random walk from a grid walk a ball walk ([15]). This improvement came at the cost of making the analysis more difficult, mainly because the ball walk has infinite state space.

Using the HR walk ([25]) further improved the complexity of the volume estimation algorithms ([13]). HR has many attractive properties, and as such forms the basis of the walk chosen for our implementation. HR is a time-reversible random walk which is able to reach any part of a convex (or non-convex) body in a single step ([26]). The HR algorithm is shown in Figure 8, and illustrated in Figure 9. From any position x inside K, a random direction μ is chosen uniformly from the unit sphere. The next position is chosen by selecting a step uniformly along the line segment passing through x, in the direction of μ, with end points at the boundary of K. In practice, finding the end points does not pose a problem: a simple bisection method can quickly find a step inside K along the line. For MP-MCMC volume estimation, bisecting is made even easier because we know that the end points initially intersect the ball of the current dilation.

As mentioned above, HR has a major advantage over the two other walks: we can sample uniformly from K even if K is non-convex. This is demonstrated by looking at the same non-convex example as before (Figure 10). Any point x can easily reach any other region in K, even if K is disjoint.

Nevertheless, HR has its own shortcomings. Notably, if K is an elongated body (with a large sandwiching ratio), drawing directions from a uniform direction distribution can lead to inefficient exploration of K ([11]), making sandwiching a necessary preprocessing step for any efficient and accurate implementation. Fortunately, this shortcoming is addressed below.

3.5 Artificial centering Hit-and-Run

Artificial centering Hit-and-Run (ACHR) ([11]) was developed as a practical improvement to HR. Kaufman and Smith showed that ACHR significantly increases the number of uniform samples over

Algorithm 2: HIT-AND-RUN SAMPLER (UNIFORM TARGET DISTRIBUTION)

```
inputs
        $K$ = Oracle in $\Re^n$
        $J$ = bounding surface
        D = uniform direction distribution
        N = number of points to sample
initialization
        $x_0$ = initial point $\in K$
        Set $i = 0$
for i = 1 : N
        Generate direction $\mu_{i+1} \sim D$.
        Find end points $a$ and $b$ of the line projected from $x_i$ which intersect $J$.
        Sample $\lambda$ uniformly along line.
          while $x_{i+1} = x_i + \lambda\mu_{i+1} \notin K$
                Update $a$ and $b$.
                Sample $\lambda$ uniformly along line segment.
output
        Samples $x_1$ through $x_N$.
```

Figure 8: The HIT-AND-RUN algorithm for sampling uniformly from K (adapted from [25]).

regular HR on elongated bodies. The ACHR algorithm is shown in Figure 11, and illustrated in Figure 12. ACHR estimates an optimal direction distribution from an empirical sample.[5] Selecting a direction uniformly from the unit hypersphere is an optimal direction distribution if the body is spherical. For an elongated hyperellipsoid, a uniform direction distribution will oversample from the "thin" directions, even though, to sample efficiently, it should concentrate on the elongated directions. ACHR attempts to estimate an optimal direction choice by selecting directions not from the uniform direction distribution, but an empirical estimate of the optimal direction. This direction distribution is the set of directions from all previous samples to an "artificial" center. The artificial center is the mean of the previous samples. Initially, there is a "warm-up" period where samples (which are later thrown away) are generated from the hypersphere defined by r. As the sampling continues, points away from the center of the body will contribute to directions to the elongated extremes of the body, avoiding "thin" directions.

Unfortunately, the improvement comes at a theoretical cost: a random walk by ACHR is no longer a Markov chain, thus making all proofs of convergence for the volume algorithms invalid. Nevertheless, we feel that the improvements are justified. There are two foreseeable advantages of ACHR over HR. The first is that because ACHR directions will tend to point towards longer portions of the body, it will spend less time bisecting as it attempts to find a sample inside K. The second advantage is that ACHR will sample points that are closer than HR to the uniform distribution on K ([11]). Since we will not be throwing out serial correlations (neighboring samples), the quality of the samples becomes more important. One can imagine an HR sampler at a position near the tip of an elongated hyperellipsoid. Because there are few directions pointing to the other end of the hyperellipsoid, HR will tend to stay at the tip region much longer than ACHR. For a small number of samples, this may overestimate the volume. If the HR is near the center of K, then it can underestimate the volume, because few directions will point to the ends of the hyperellipsoid. Overall, because of these localized, and highly correlated, sets of samples, HR may produce volume estimates with higher variance than ACHR.

3.6 Implementation

We now discuss the implementation details of the volume estimation algorithm used in our experiments.

The initialization of the MP-MCMC algorithm is important because incorrect values of r and R will result in incorrect volume estimates, even if the sampling is of high quality. Before defining the

[5]If there are a sufficient number of examples, an alternative approach to estimating the optimal direction distribution is to sample directions from a multivariate normal distribution with mean zero and covariance equal to the inverse Hessian of the data points ([38]).

Figure 9: Hit-and-Run algorithm. A single step is illustrated from initial state x_0 to x_1. First, a direction μ_1 is selected uniformly from the unit hypersphere. A point is then sampled along the line segment defined by μ_1, passing through x_0, and with end points a and b on the enclosing surface, J. The first sampled point b' is outside K. The endpoint b is moved to b', and another point is sampled along the same line, but with different end points. The next point x_1 is accepted because it is inside K. A new direction μ_2 is then chosen for the next step.

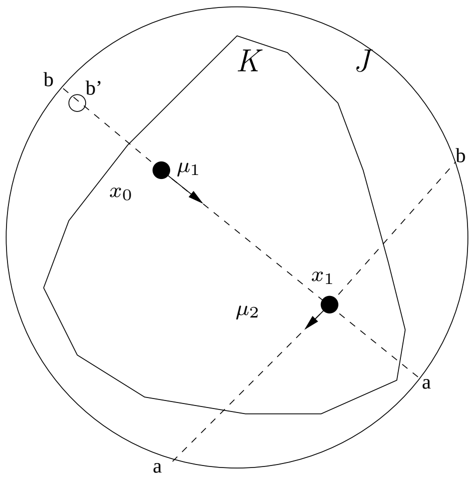
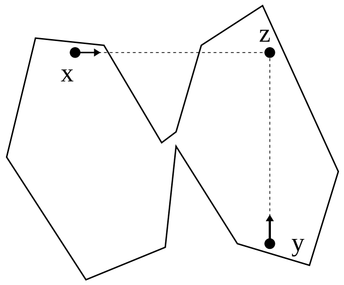


Figure 10: The Hit-and-Run walk is able to reach point z from both x and y is a single step.



Algorithm 3: ARTIFICIAL CENTERING HIT-AND-RUN SAMPLER

```
inputs
        $K$ = Oracle in $\Re^n$
        $J$ = bounding surface
        D = uniform direction distribution
        $V$ = number of warm-up points
        $N$ = number of points to sample
initialization
        $x_0$ = initial point $\in K$
for i = 0 : N
        Select random index j, from 1 to V+i
        If $j \leq V$, set $\mu_{i+1} \sim D$, otherwise $\mu_{i+1} = x_{j-V}$.
        Find end points $a$ and $b$ of line projected from $x_i$ which intersect $J$.
        Sample $\lambda$ uniformly along line.
          while $x_{i+1} = x_i + \lambda\mu_{i+1} \notin K$
                Update $a$ and $b$.
                Sample $\lambda$ uniformly along line segment.
output
        Samples $x_1$ through $x_N$.
```

Figure 11: The ARTIFICIAL CENTERING HIT-AND-RUN algorithm for sampling uniformly from K (adapted from [11]). Note that there is no artificial center: we assume that K has already been centered at the origin. See Figure 12 for an illustration of ACHR.

radii, however, we must select the "center" of K. The model classes that we use in our experiments all have some measure which can be used to select the most "likely" training point. For gmm, this is the probability density function; the center is chosen as the training point with highest density. For kmeans, the measure is the distance of each point to its closest mean; the center is chosen as the training point with the smallest distance.

We first make initial guesses for r and R using an ad hoc heuristic (see Figure 14). We set r to be the smallest distance of all *rejected* training points to the center. We set R to be the largest distance of all non-rejected training points to the center. Because this heuristic explicitly incorporates error into our volume estimations, we refine our values of r and R as follows. For r, we iteratively sample ten thousand points from $B(\mathbf{0}, r)$ until none are rejected. If any samples are rejected, we decrease r by 10%. For R, we project all training points so that they all lie on the hypersphere of radius R. If any of the points are accepted, then we increase R by 10% and re-project the points. We iterate until all projected training points are rejected. These heuristics are fast and they produce a more precise initialization of r and R, but they can still contain error.

Once the center and the two radii are defined, we compute the dilations of $B(\mathbf{0}, r)$. The role of the dilations is to maintain a large ratio between the volume of K_i and K_{i-1}. Following the examples of [24] and [10], we find dilations such that each dilation approximately doubles the volume of the previous dilation. We compute m, the number of dilations of $B(\mathbf{0}, r)$, such that each dilation approximately doubles in volume:

$$m = \lceil \log_2 \frac{V_R}{V_r} \rceil,$$

where $V_r = vol(B(\mathbf{0}, r))$ and $V_R = vol(B(\mathbf{0}, R))$. Since we may not be able to exactly double the volume of each dilation, we calculate Δ such that:

$$\Delta^m = \frac{V_R}{V_r}.$$

Because the volume of a hypersphere of fixed dimensionality is only a function of its radius, we can solve for Δ by as follows:

$$\Delta^m = \frac{R^n}{r^n}$$

Figure 12: Artificial centering Hit-and-Run algorithm. The center of K is shown as an unfilled circle. The direction distribution is represented by the arrows extending from the center. Dashed arrows represent warm-up directions and solid arrows represent directions derived from steps in the ACHR walk. Once x_0 has been chosen, the direction from the center to x_0 is added to the direction distribution (shown as dashed line). Initially, x_0 selects μ_1 from the warm-up directions. Once at x_1, the direction from the center to x_1 is added to the direction distribution etc.

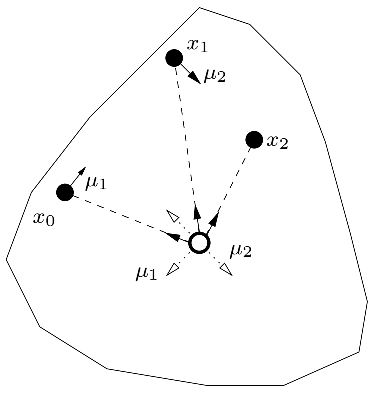


Figure 13: Direction distributions. The steady-state direction distributions of HR and ACHR are shown for an ellipsoid body. HR maintains a uniform direction distribution, whereas ACHR will concentrate its direction density on directions pointing to the elongated regions of the ellipsoid.

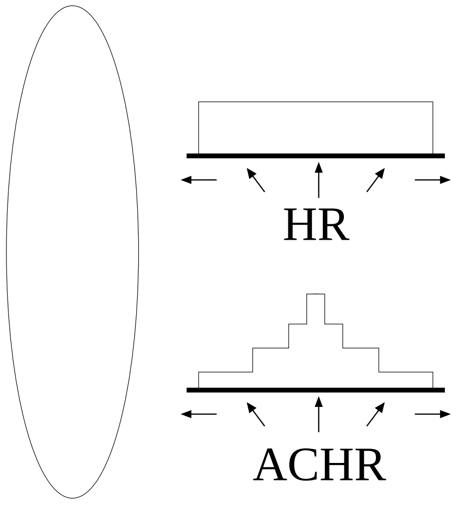
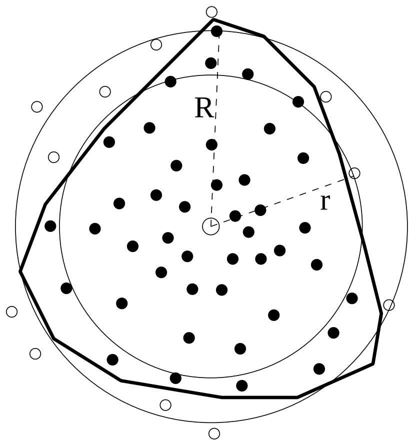


Figure 14: MP-MCMC initial estimates of r and R. Filled circles represent non-rejected training points., while unfilled circles rejected training points. The large unfilled circle is the center of K, defined as as the most normal training point. The Euclidean distance is then computed between the center and all other training points. We set r to be the smallest distance of a rejected point to the center, and R to be the largest distance of a non-rejected point to the center.



$$
\begin{aligned}
m \log_2 \Delta &= n \log_2 R - n \log_2 r \\
\log_2 \Delta &= \frac{n \log_2 R - n \log_2 r}{m} \\
&= \delta.
\end{aligned}
$$

We can now solve for the radii of each dilation i using the follwing:

$$
\begin{aligned}
\frac{r_i^n}{r^n} &= \Delta^i \\
n \log_2 r_i - n \log_2 r &= i \log_2 \Delta \\
r_i &= r2^{\delta(\frac{i}{n})}, i = \{1, 2, \cdots, m\}
\end{aligned}
$$

We can verify that r_0 and r_m are exactly r and R, respectively:

$$r_0 = r2^0 = r$$

$$r_m = r2^{\frac{n \log_2 R - n \log_2 r}{m}(\frac{m}{n})} = r\frac{R}{r} = R.$$

After the dilations have been initialized, we can begin running MP-MCMC. For each dilation i, we run three Monte Carlo simulations simultaneously. Each simulation samples points from K_i using the ACHR walk. At each dilation, ACHR uses one hundred warm-up points sampled uniformly from $B(\mathbf{0}, r)$. The statistic monitored during the simulations is the ratio between the number of points in K_{i-1} and K_i. When the between and within variance of the statistic falls below 10^{-5}, or when each simulation has already sampled ten thousand points, the simulations are stopped and the mean value of the ratio is returned. This is an approach similar to one used in [3]. Finally, we use the equations from Section 3.2 to compute the volume, replacing each N/T_i with the ratios computed by the simulations.

4 Experiments

In this section, we will describe two sets of experiments. The first will test the accuracy of our volume estimation algorithm. We will run our algorithm on simple geometric shapes with known volume, and test how well the algorithm performs as the dimension increases. Comparisons will be made with a simple naive approach, MP-MCMC with HR walks, and MP-MCMC with ACHR walks. The second set of experiments will test the model selection criteria. To do this we will use classification datasets, and train models using one of the classes, and test on the remaining classes which act as an outlier test set.

4.1 Experiments: volume estimation

These experiments will compare three volume estimation procedures: *naive*, HR, and ACHR. The *naive* method is used by [31] and was discussed in Section 2.2.[6] It consists of sampling uniformly from the outer ball, and counting the number of samples that fall inside K. HR and ACHR use the MP-MCMC implementation discussed in Section 3.6.

To compare these methods, we will test their accuracy on objects of known volume. The first experiment will test the algorithms on a difficult convex object: an elongated hyperellipsoid. The radius of each dimension of the hyperellipsoid is one, except along one dimension which has radius ten. In the second experiment, the algorithms are tested on their ability to estimate the volume of a non-convex body: three non-overlapping hyperspheres, aligned along one axis, each having radius one.

We vary the dimensionality of the objects from two to ten. Because we know the minimum and maximum radii, we will initialize r and R exactly, so that all three methods will use the same inner (if applicable) and outer balls. A fair comparison will be made between *naive* and HR and ACHR by allowing *naive* to sample as many points as the maximum allowed by the other methods. For example, if HR and ACHR use a maximum of ten thousand samples per dilation for five dilations, each with three simultaneous simulations, then we would sample one hundred fifty thousand samples for *naive*. For these experiments, we will estimate the volume of each object ten times.

4.2 Experiments: model selection

To demonstrate the flexibility of our approach, we will build models from several different model classes, and exclusively use C1 and C2 to pick the best model.[7]

Although we will build models on a range of $\nu^\star$ values for each dataset, in all our experiments we assume that $\nu^\star$ is a parameter defined by the user. In manufacturing, for example, $\nu^\star$ could represent the operation engineer's upper limit of how many objects to stop and test. We will build and test a model for each parameter setting and for each $\nu^\star$, which varies over the values 1%, 5%, 10%, 20%, and 50%. Note that we are most interested in the results when $\nu^\star$ is small, e.g. 1% and 5%.

For each parameter setting, we run ten-fold cross-validation three times and estimate $\hat{\nu}$ as the mean of the three runs. All models whose $\hat{\nu}$ differs relatively from $\nu^\star$ by less than 15% satisfy C1. We select, from those models satisfying C1, the model with the lowest estimated volume, using MP-MCMC with ACHR sampling (which we will henceforth call *volume*). The volume is computed on the model trained using the entire training set. We make one modification to the MP-MCMC algorithm described earlier: we do not compute an "artificial" center. We assume that the center is the origin, and that the most likely point in the training set acts as an offset from the origin. We will also compare the naive volume estimation technique (*naive*) to our approach.

The datasets used in these experiments are all classification sets from the UCI Machine Learning Repository (`http://www.ics.uci.edu/∼mlearn/MLRepository.html`). We use one of the classes as the normal class, and all the other classes combined as the outlier class (or, equivalently, the abnormal class). All model selection is performed exclusively using the normal dataset. The outlier set is used as an objective estimate of the outlier error rate (false positive rate). Table 1 describes the size and dimension of the datasets. The datasets were modified prior to our experiments: any objects with missing attributes were removed; attributes which have zero variance in the normal class were removed; and both normal and abnormal classes were whitened by removing the mean of the normal class, and by dividing by the standard deviation of the normal class. Whitening is not necessary, but was done so that the parameter ranges (especially σ) would have generally the same effect on all datasets.

The models selected by C1 and C2 are directly compared with two-class classification results. The normal dataset represents one class, and the outlier dataset the other class. We use a rbf two-class svm whose optimal parameters are selected using ten-fold crossvalidation. All svm optimization is performed using the libsvm toolbox (`http://www.csie.ntu.edu.tw/ ∼cjlin/libsvm/`).

[6] See [31] for details of how to sample uniformly from a ball.

[7] In practice, however, we may not be interested in the single best model, but in ranking the models. For example, the volume estimates could be used as votes in a committee machine. Such extensions are beyond the scope of this paper.

Table 1: Data sets used in experiments.

Name	n	nbr D_+	nbr D_-	Description	Normal	Outlier
gauss	2	100	2500	Toy data	two gaussians	uniform
iris	4	50	100	Iris Plant	Virginica	Setosa and Versicolour
yeast	8	463	1021	Protein localization	CYT	other nine classes
letter	16	789	19211	Letter recognition	A	B-Z
derm	25	111	247	Dermatology	psoriasis	other five conditions
ion	32	225	126	Ionosphere	good returns	bad returns

Table 2: Model classes used in our experiments.

Model Class	Parameters	Ranges
ν-gmm	k	1-5
ν-parzen	σ	0.1:1000
ν-kmeans	k	1-5
ν-svm	σ	0.1:1000

4.2.1 Model classes

In our experiments, we use several common density estimation and one-class classification algorithms. Table 2 lists the model classes and their adjustable parameters. It should be noted that although we build models on a limited range of parameter values, our results are not affected: we are interested in model selection, that is, finding the best model from a set of candidate models, and not in searching all model classes and parameter spaces for the absolute best model.

We now describe how the models are modified to reject ν training points. We will refer to models that have been modified in such a way as **ν-ModelName** (e.g. **ν-gmm**).

A Parzen windows model that rejects data when $p(x) < \tau_\nu$ is called a **ν-parzen** model. The ν threshold, τ_ν, is defined as follows. Sort the densities of the training data in ascending order. Let the ν-th point to be rejected be x_m, where $m = \lceil \nu \cdot N \rceil$. Set $\tau_\nu = (p(x_m) + p(x_{m+1}))/2$.

A kmeans model that rejects data when $d(x) < \tau_\nu$ is called a **ν-kmeans** model. In our experiments, the ν threshold, τ_ν, is defined as follows. Sort the distances of each point in the training data from the nearest cluster center, d_i, in *descending* order. Let the ν-th point to be rejected be x_m, where $m = \lceil \nu \cdot N \rceil$. Set $\tau_\nu = (d(x_m) + d(x_{m+1}))/2$. A kmeans model that rejects data when $d(x) < \tau_\nu$ is called a **ν-kmeans** model.

A gmm that rejects data when $p(x) < \tau_\nu$ is called a **ν-gmm** model. The ν threshold, τ_ν, is defined in the same as manner as **ν-parzen**. In our experiments, we only consider full-covariance matrices; we therefore only build gmm models when $n^2 < N$, where N is the number of training examples. This excludes *derm* and *ion*.

In our experiments, we use an rbf kernel so that low density regions are mapped close to the origin in feature space, and so that the acceptance region is bounded in input space. Therefore, the only parameter for **ν-svm** is the rbf width parameter σ. All **ν-svm** optimization is performed using the libsvm toolbox (`http://www.csie.ntu.edu.tw/∼cjlin/libsvm/`).

5 Results

We divide the presentation of our results into separate sections for the volume estimation experiments and the model selection experiments (Sections 5.1 and 5.2, respectively). We also briefly describe the results of other model selection heuristics in Section 5.3.

5.1 Results: volume estimation

Table 4 shows the results of the volume estimation experiments on an elongated hyperellipsoid. We report the mean volume estimates (μ_{vol}), the standard deviation of the volume estimates (σ_{vol}), and the mean absolute error between the volume estimates and the true volume (μ_{err}). All volumes are reported

Table 3: Volume estimation results for a hyperellipsoid of radius one for all but one dimension. The remaining dimension has radius ten. We report the mean volume estimates (μ_{vol}), the standard deviation of the volume estimates (σ_{vol}), and the mean absolute error between the volume estimates and the true volume (μ_{err}). Results are reported in units of log volume. The result with the lowest μ_{err} is bold faced. The ratio between the volume of the enclosing hypersphere and the volume of the hyperellipsoid is shown ($vr = \lceil \log_2 \frac{vol(B(\mathbf{0},R))}{vol(K)} \rceil$).

n	vr	true	naive			HR			ACHR		
			μ_v	σ_v	μ_{err}	μ_v	σ_v	μ_{err}	μ_v	σ_v	μ_{err}
2	4	3.4473	3.4486	0.0095	**0.0078**	3.4385	0.0189	0.0128	3.4532	0.0188	0.0152
3	7	3.7350	3.7224	0.0354	0.0303	3.7041	0.0179	0.0309	3.7293	0.0211	**0.0169**
4	10	3.8989	3.8783	0.0776	0.0659	3.8605	0.0510	0.0533	3.9056	0.0290	**0.0253**
5	14	3.9634	3.9408	0.3862	0.2814	3.9399	0.0578	0.0422	3.9455	0.0268	**0.0241**
6	17	3.9450	3.8873 (9)	0.6062	0.5128	3.8472	0.0654	0.1054	3.9286	0.0187	**0.0210**
7	20	3.8554	5.3251 (1)	0	1.4697	3.7500	0.0583	0.1054	3.8365	0.0247	**0.0237**
8	24	3.7035	7.3531 (1)	0	3.6497	3.6278	0.0976	0.0855	3.6700	0.0382	**0.0379**
9	27	3.4961	n/a	n/a	n/a	3.3399	0.1079	0.1696	3.4552	0.0299	**0.0409**
10	30	3.2387	n/a	n/a	n/a	3.0723	0.0358	0.1665	3.1905	0.0453	**0.0567**

in natural log. For *naive*, we display the number of non-zero volume estimates in parentheses if there are less than ten non-zero estimates. For $n = 9$ and $n = 10$ *naive* has volume estimates of zero for all ten runs. For all dimensions except $n = 2$, ACHR has the best μ_{err}. HR, in turn, outperforms *naive* on all but $n = 2$ and $n = 3$. Note how the estimates tend to underestimate the true volume by a greater degree as the dimension increases.

Table 4 shows the results of the volume estimation experiments on three aligned hyperspheres. In this experiment, *naive* performs well for all dimensions. This is mainly explained by the ratio between the volume of $B(\mathbf{0}, R)$ and K, which remains relatively larger than in the previous experiment. ACHR once again outperforms HR and *naive*, and does not seem to suffer performance loss even though K is no longer convex.

5.2 Results: model selection

For each dataset we present two plots (see Figures 15 and 16). The left-most plots illustrate model selection using C1 and C2 (*volume*). The x-axis represents $\hat{\nu}$ and the y-axis represents the log of the volume estimate for each model. All the models are shown as *x's*, models satisfying C1 are shown as circles, and the model with the smallest volume, according to *volume*, for each $\nu^\star$, is shown as a square. The smallest models according to *naive* are shown as triangles. For all plots, $\hat{\nu}$ is plotted on the log scale. Note that in classification terms, $\hat{\nu}$ is the estimate of the false-negative rate.

The symbols in the right-most plots represent the same models as on the left, but here the y-axis represents the test error on the outlier class (false positives). The two-class svm cross-validation errors are also plotted as stars. Errors for the normal class (false negatives) are plotted on the x-axis, and errors for the abnormal class (false positives) are plotted on the y-axis.

The results are summarized by Table 5 and Figure 17. Table 5 shows the outlier error rate for the model selected using *volume*. For comparison purposes, the error for the smallest model for each of the model classes is also shown, as is the model selected using *naive*. Note that *volume* generally picks the best model among all the model classes—many times more than *naive*—and that there is never one model class which always produces the best model.

The results in Figures 15 and 16 are summarized in Figure 17. Along the x-axis is the log volume of the optimal models chosen either by *volume* (squares) or by *naive* (triangles) from Figures 15 and 16. Along the y-axis the abnormal test error is plotted in the log scale, for clarity. The dashed lines connect the optimal models of the same dataset, for different values of $\nu^\star$. Because the test error is plotted in log, models with zero error are plotted at $1e-5$, so they appear on the plot. The solid lines join *volume* and *naive* models for the same $\nu^\star$. There are two observations to be made. One is that the abnormal test error monotonically increases with the volume estimates. The second is that on several

Table 4: Volume estimation results for three hyperspheres of radius one, aligned along the first dimension. We report the mean volume estimates (μ_{vol}), the standard deviation of the volume estimates (σ_{vol}), and the mean absolute error between the volume estimates and the true volume (μ_{err}). Results are reported in units of log volume. The result with the lowest μ_{err} is bold faced. The log of the ratio between the volume of the enclosing hypersphere and the volume of the hyperellipsoid is shown ($vr = \log_2 \frac{vol(B(\mathbf{0},R))}{vol(K)}$), which corresponds to the number of dilations.

n	vr	true	naive			HR			ACHR		
			μ_v	σ_v	μ_{err}	μ_v	σ_v	μ_{err}	μ_v	σ_v	μ_{err}
2	2	2.2433	2.2456	0.0068	0.0060	2.2465	0.0164	0.0131	2.2453	0.0086	**0.0066**
3	4	2.5310	2.5232	0.0140	0.0125	2.5304	0.0243	0.0199	2.5396	0.0127	**0.0106**
4	5	2.6949	2.6998	0.0148	0.0114	2.6944	0.0240	0.0191	2.6917	0.0124	**0.0084**
5	7	2.7595	2.7396	0.0440	0.0352	2.7662	0.0282	0.0242	2.7553	0.0193	**0.0130**
6	8	2.7410	2.7386	0.0610	0.0482	2.7148	0.0653	0.0536	2.7269	0.0154	**0.0169**
7	10	2.6514	2.6818	0.0718	0.0642	2.6202	0.0548	0.0496	2.6348	0.0170	**0.0231**
8	12	2.4995	2.4981	0.0661	0.0530	2.3943	0.0980	0.1175	2.4620	0.0112	**0.0374**
9	13	2.2921	2.0471	0.2606	0.2746	2.1496	0.1995	0.1910	2.1400	0.0269	**0.1520**
10	15	2.0348	1.9469	0.1962	**0.1739**	1.7124	0.2165	0.3317	1.7959	0.0484	0.2389

datasets the model selected by *volume* is significantly better than the homologous *naive* model. This is most apparent in *letter* and *ion*. For these sets, *naive* returned few volume estimates greater than zero, thus restricting model selection from only the largest of models.

5.3 Results: other selection heuristics

We also compared *volume* with other model selection heuristics besides *naive*: for **ν-gmm** and **ν-parzen** models, we compared the cross-validated log likelihood; for **ν-parzen** and **ν-svm** models, we compared the *increasing* σ method of [34]; and for **ν-kmeans** models, we compared using the *within* cluster variance as a selection heuristic. For all heuristics, we found that *volume* either selected a similarly performing model or a better model. In most cases, for the **ν-parzen** and **ν-svm** models, the *increasing* σ method selected the same models as *volume*.

6 Discussion

We divide the discussion of our results into separate sections for the volume estimation experiments and the model selection experiments (Sections 6.1 and 6.2, respectively). We also briefly describe possible sources of error (Section 6.3) and future directions (Section 6.4).

6.1 Discussion: volume estimation

The results from the volume estimation experiments clearly show the superiority of ACHR over HR and *naive*. The results are accurate even on such a difficult object as the hyperellipsoid. At $n = 10$, the volume of the hyperellipsoid must double 30 times to equal that of $B(\mathbf{0}, R)$. At this dimension, HR commits three times the error of ACHR.

The inherent difficulties of a naive approach are clearly demonstrated by this experiment. At $n = 4$, *naive* is already making as much error as ACHR does at $n = 10$. At $n = 6$, *naive* has begun to miss K altogether, achieving only nine of ten runs with at least one point in K. For dimensions greater than eight, *naive* requires a huge number of attempts to have just one point fall in K.

The results of *volume* on the three hyperellipsoids indicate that the non-convexity of the shape was not a severe handicap to the algorithm.

Though not reported in our results, *naive* can potentially have a significant efficiency advantage over both HR and ACHR. This is because both HR and ACHR must sample from within K_i, which requires testing, for each sample inside K_i, many points outside of K_i that are never used as samples. It should be emphasized that the advantage is not always present, and is most pronounced when the body

Table 5: The abnormal test error (false positive rate) from model-class specific optimal models using *volume* are shown. The selection process was performed on each model class independently of each other. The best test error is bold faced. If there are no models that satisfy C1, their entry is '?'. There were no **ν-gmm** models produced for *derm* and *ion*. The best model selected using *naive* is also shown.

dataset	$\nu^\star$	*volume*	*naive*	**ν-kmeans**	**ν-parzen**	**ν-gmm**	**ν-svm**
gauss	0.01	**0.4348**	**0.4348**	?	**0.4348**	0.4680	?
	0.05	**0.2736**	0.2756	0.2852	0.2828	0.2796	**0.2736**
	0.1	**0.2084**	0.2104	**0.2084**	0.2160	0.2168	0.2204
	0.2	**0.1480**	0.1488	0.1544	0.1600	**0.1480**	0.1580
	0.5	**0.0692**	**0.0692**	0.0720	0.0768	**0.0692**	0.0820
iris	0.01	**0.1900**	**0.1900**	?	0.2600	0.2600	**0.1900**
	0.05	**0.2300**	**0.2300**	?	?	**0.2300**	?
	0.1	**0.0500**	0.1400	**0.0500**	0.1800	0.1400	?
	0.2	**0.0200**	**0.0200**	**0.0200**	**0.0200**	0.0300	0.0300
	0.5	**0.0000**	**0.0000**	0.0100	0.0100	**0.0000**	0.0100
yeast	0.01	**0.9775**	**0.9775**	?	?	**0.9775**	?
	0.05	**0.7081**	0.7728	0.7111	**0.7081**	0.7679	0.7747
	0.1	**0.5896**	0.6376	0.5955	**0.5896**	0.6572	0.6405
	0.2	0.4466	0.4466	0.4466	0.4574	**0.4310**	0.4701
	0.5	0.1783	0.1783	0.1783	0.2233	**0.1606**	0.2223
letter	0.01	**0.0994**	0.3480	?	0.3480	0.7109	**0.0994**
	0.05	**0.0440**	0.1406	0.1631	0.0922	0.1309	**0.0440**
	0.1	**0.0035**	0.0632	0.0681	0.0407	0.0095	**0.0035**
	0.2	**0.0003**	0.0123	0.0042	0.0105	0.0010	**0.0003**
	0.5	**0.0000**	**0.0000**	**0.0000**	0.0005	**0.0000**	**0.0000**
derm	0.01	**0.6113**	**0.6113**	?	**0.6113**	n/a	?
	0.05	0.1255	0.1255	?	0.1255	n/a	**0.0648**
	0.1	**0.0121**	**0.0121**	?	?	n/a	**0.0121**
	0.2	0.0081	0.0081	0.0040	0.0081	n/a	**0.0000**
	0.5	**0.0000**	**0.0000**	**0.0000**	?	n/a	**0.0000**
ion	0.01	**0.5238**	**0.5238**	?	?	n/a	**0.5238**
	0.05	**0.1905**	0.3254	?	**0.1905**	n/a	0.3254
	0.1	**0.0873**	**0.0873**	**0.0873**	0.1190	n/a	0.2540
	0.2	**0.0317**	0.0556	**0.0317**	0.0556	n/a	0.0873
	0.5	0.0079	0.0159	0.0079	**0.0000**	n/a	0.0079

is composed of small, disjoint regions in input space. Fortunately, these are the models that are rejected by C1, and therefore, in practice, the speed advantage of *naive* over HR and ACHRis not necessarily significant. For high-dimensional problems, since we do not know the volume of K in advance, it would be risky to use *naive* because we have no knowledge of how many samples are necessary to have one fall into K. ACHR, in turn, is more efficient than HR, because it tends to avoid sampling in directions where K_i is "thin", and therefore uses less time bisecting.

The volume estimation results empirically confirm our use of ACHR in our experiments. Not only does ACHR result in better volume estimates, but it is faster than HR in practice.

6.2 Discussion: model selection

For all datasets, the implemented selection criteria are able to choose the best model with respect to outlier test error for nearly all cases of $\nu^\star$. That is, among all models selected using C1, the models with the smallest volume using our procedure, are also the models with the lowest false positive rate (25/30 from Table 5). This is not true for *naive* (13/30 from Table 5). Although, for many cases, the best models for *volume* and *naive* coincide, *naive* is unreliable, especially for high-dimensional datasets, because fewer models have non-zero volume estimates, and of those, their estimates are more susceptible to error.

The first criterion C1 is important so that very small models are rejected. For all the datasets, there are model classes and parameterizations which produce very small solutions, but which are easily identified using a cross-validation estimate of $\nu^\star$. In practice, we would not estimate the volume of these overfit models. This allows computational resources to be used for estimating the volume of larger acceptance regions. Overfit models can have extremely disjoint regions, requiring more dilations, and also causing bisection to be very inefficient.

The main advantage of our criteria is that it can almost always select the best model from all the model classes. For each dataset and $\nu^\star$, a particular model class may select the best model, but there is never one model class which always performs best. Our criteria, however, tends to select the best model for all datasets. This allows the user to experiment with a wide variety of model classes, and have confidence in the selected models.

6.3 Discussion: sources of error

There are several potential sources of error in both our model selection procedure and in the implementation of the volume algorithm.

By estimating $\nu^\star$ using cross-validation, we introduce error into our estimate. We may be able to achieve more stable results using either a different number of folds, or by averaging more than three cross-validation estimates, as we have done. Leave-one-out cross-validation, though more time consuming, would be more accurate. For generative models, drawing samples from $p(x)$ can also be used to estimate the quality of the rejection rate on normal data ([19]).

Our implementation of C1 is arbitrary; correctly selecting the C1 threshold is an open problem. If there are many models with stable $\hat{\nu}$, then the threshold could be decreased to focus on models that are most stable.

C2 selects, from the models satisfying C1, the model with the smallest volume; however, since the volume estimates are not error free, selecting the smallest model, without taking into account the volume variability, is inappropriate. In our experiments, we do not take into account this noise, although one could select, from the models satisfying C1, all the models whose volumes are within some ϵ of the smallest volume.

The advised number of Markov chain steps in the volume estimation literature are based upon model assumptions that we did not make (i.e. convexity, sandwiching ratios). We therefore used our own heuristics to measure convergence. Since our approach is ad hoc, there is no reason to conclude that our ratio estimates in the MP-MCMC algorithm result from converged Markov chains (ACHR does not produce a proper Markov chain). Indeed, using three simultaneous MP-MCMC runs is arbitrary; certainly more runs, of longer length, would result in more accurate estimates.

Despite the ability of ACHR to sample from bodies with large sandwiching ratios, simple rounding techniques should be considered because it could significantly reduce the number of dilations required by MP-MCMC. By whitening our datasets prior to the experiments, some rounding was already incorporated into our procedure.

Because we are testing our outlier acceptance rate using classification data, we are limited by these distributions. The classification outlier test sets, from our experiments, occupy only part of the input space. Our model selection criteria may select the smallest model, but another, larger model may be better at discriminating between the normal and outlier data sets. This source of error is beyond our control.

All these sources of error may help to explain why the model selection criteria do not perfectly select the best model in all cases. We hope that our results are encouraging enough to inspire further research.

6.4 Discussion: future directions and open problems

Despite the success of selecting novelty detection models, there are other improvements that could be made to a novelty detection *system* to increase its usability. Since a novelty detection model only expresses novelty as a binary event, using a continuous measure of novelty would be useful. This is an advantage that density estimation has over novelty detection. Most novelty detection models do, however, have an output that can be interpreted as density, allowing the user to know *by how much* a test point is an outlier.

We may also want to recognize specific outlier patterns. This would entail clustering or learning models of outlier *signatures*, which could then be recognized as specific classes in the future. A related issue is diagnosis. If the model detects an outlier, it would be useful to diagnose the cause (location) of the alarm ([35]).

High performance novelty detection requires quality sensor measurements. It is obvious that if input data are not the result of monitoring the correct aspect of a machine, novelty detection has no hope of producing correct results ([35]). On the other hand, operations engineers have difficulty in monitoring multiple sensors at the same time, i.e. sensor fusion. In such cases, automated model building of high-dimensional processes would greatly improve monitoring processes in industrial situations.

In this paper we did not consider time-series data specifically. A state-space model of measurements from a sliding time window could be used with our setup, but another useful novelty detection approach for time-series data is to build a regression model of the sensors, detecting novelty when the predicted sensor values differ from the actual sensor values at the next time step.

Domains such as text and gene expression (discrete data) were not considered in this paper. Since we assumed a continuous input space, building a volume estimate intuitively made sense. But for discrete spaces, computing the volume using Euclidean geometry may be inappropriate. Modifying the volume estimation algorithms to work in a non-Euclidean space, for example on the multinomial manifold ([12]), would be a useful endeavor.

7 Conclusion

In this paper we have presented clearly defined criteria for selecting a novelty detection model among a large set of models which are built from multiple model classes and parameterizations. We have presented a volume estimation algorithm which is *one* possible solution to the second criterion.

Our experimental results confirm volume estimation as an essential criterion for a *generalized* model-selection approach. ACHR is shown to be a superior sampling technique to HR, and in turn, in conjunction with MP-MCMC, vastly superior to a naive sampling technique for objects of high dimension. It should be emphasized that for low dimensions, or for very well-rounded acceptance regions, *naive* can be a good approach.

Compared with other model-class specific heuristics, our criteria perform the same or better. Our selection criteria are able to select the best model from a wide range of model classes and parametrizations.

Acknowledgements

The author is grateful to Rodrigo Fernandez, Radford Neal, and Sam Roweis for useful discussions. Initial work on this project was done while the author worked at Mathématiques Appliquées S.A.

References

[1] James Allan, Victor Lavrenko, and Hubert Jin. First story detection in tdt is hard. In *Proceedings of the 9th international conference on Information and knowledge management*, pages 374–381, 2000.

[2] Salah Bouhouche. *Contribution to Quality and Process Optimisation in Continuous Casting Using Mathematical Modelling*. PhD thesis, Freiberg Technical University, 2002.

[3] Stephen Brooks and Andrew Gelman. General methods for monitoring convergence of iterative simulations. *Journal of Computational and Graphical Statistics*, 7:434–455, 1998.

[4] M. Costa and L. Moura. Automatic assessment of scintmammographic images using a novelty filter. In *Proc Annu Symp Comput Appl Med Care*, pages 537–41, Sao Paulo, 1995.

[5] Christopher P. Diehl and John B. Hampshire II. Real-time object classification and novelty detection for collaberative video surveillance. In *International Joint Conference on Neural Networks*, volume 3, pages 2620–2625, 2002.

[6] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. Wiley, 2nd edition, 2001.

[7] Martin Dyer, Alan Frieze, and Ravi Kannan. A random polynomial time algorithm for approximating the volume of convex bodies. *Journal of the ACM (JACM)*, 38(1):1–17, 1991.

[8] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer, 1st edition, 2003.

[9] Mark Jerrum. *Volume of a convex body*, chapter 6, pages 71–105. 2002.

[10] Ravi Kannan, Lasloz Lovasz, and Miklos Simonovits. Random walks and an $o^\star(n^5)$ volume algorithm for convex bodies. *Random Structures and Algorithms*, 11:1–50, 1997.

[11] David E. Kaufman and Robert L. Smith. Direction choice for accelerated covergence in hit-and-run sampling. *Operations Research*, 46(1):84–95, 1998.

[12] Guy Lebanon and John Lafferty. Hyperplane margin classifiers on the multinomial manifold. In *International Conference on Machine Learning*, volume 21, 2004.

[13] Laszlo Lovasz. Hit-and-run mixes fast. *Math. Programming*, 86:443–461, 1998.

[14] Laszlo Lovasz and Miklos Simonovits. Mixing rate of markov chains, an isoperimetric inequality, and computing the volume. In *31st Annual Symposium on Foundations of Computere Science*, pages 346–355, 1990.

[15] Laszlo Lovasz and Miklos Simonovits. Random walks in a convex body and an improved volume algorithm. *Random Structures and Algorithms*, 4:359–412, 1993.

[16] Laszlo Lovasz and Santosh Vempala. Simulated annealing in convex bodies and an $o^\star * (n^4)$ volume algorithm. In *44th IEEE Foundations of Computer Science (FOCS 03)*, 2003.

[17] Larry M. Manevitz and Malik Yousef. One-class svms for document classification. *Journal of Machine Learning Theory*, 2:139–154, 2001.

[18] Stephen Marsland, Ulrich Nehmzow, and Jonathan Shapiro. Novelty detection for robot neotaxis. In *Proceedings of the 2nd Symposium on Neural Computation*, pages 554–559, 2000.

[19] A. Nairac, T. Corbett-Clark, R. Ripley, N. Townsend, and L. Tarassenko. Choosing an appropriate model for novelty detection. In *Proceedings of the 5th IEE International Conference on Artificial Neural Networks*, pages 117–122, 1997.

[20] Finn Nielsen and Lars Hansen. Modeling of activation data in the brainmap database: Detection of outliers. *Human Brain Mapping*, 15:146–156, 2002.

[21] Emanuel Parzen. On estimation of a probability function and mode. *Annals of Mathematical Statistics*, 33(3):1065–1076, 1962.

[22] Paul Sajda and Feng Han. Perceptual salience as novelty detection in cortical pinwheel space. In *IEEE EMBS Conference on Neural Engineering*, pages 43–46, 2003.

[23] Bernhard Scholkopf, John C. Platt, John Shawe-Taylor, Alexander Smola, and Robert Wolliams. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

[24] Miklos Simonovits. How to compute the volume in high dimension? *Math. Program.*, B(97):337–374, 2003.

[25] Robert L. Smith. Efficient monte carlo procedures for generating points uniformly distributed over bounded regions. *Operations Research*, 32:1296–1308, 1984.

[26] Robert L. Smith. The hit-and-run sampler: A globally reaching markov chain sampler for generating arbitrary multivariate distributions. In *Winter Simulation Conference*, 1996.

[27] Padhraic Smyth. Model selection for probabilistic clustering using cross-validated likelihood. *Statistics and Computing*, 9:63–72, 2000.

[28] Ingo Steinwart, Don Hush, and Clint Scovel. Density level detection is classification. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*. MIT Press, Cambridge, MA, 2005.

[29] L. Tarassenko, P. Hayton, N. Cerneaz, and M. Brady. Novelty detection for the identication of masses in mammograms. In *Fourth International Conference on Articial Neural Networks*, Cambridge, UK, 1995.

[30] David M.J. Tax and Robert P.W. Duin. Data domain description by support vectors. In M. Verleysen, editor, *ESANN*, volume 2, pages 251–256, Brussels, 1999.

[31] David M.J. Tax and Robert P.W. Duin. Uniform object generation for optimizing one-class classifiers. *Journal of Machine Learning Research*, 2:155–173, 2001.

[32] C. Teal and D. Sorensen. Condition based maintenance. In *Proceedings of the 20th Conference on Digital Avionics Systems*, 2001.

[33] Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of clusters in a dataset via the gap statistic. *Journal of the Royal Statistics Society: Series B (Statistical Methodology)*, 63(2):411–423, 2001.

[34] Runar Unnthorsson, Thomas Philip Runarsson, and Magnus Thor Jonsson. Model selection in one-class ν-svms using rbf kernels. *16th Conference on Condition Monitoring and Diagnostic Engineering Management*, pages 27–29, 2003.

[35] K. Worden and J. M. Dulieu-Barton. An overview of intelligent fault detection in systems and structures. *Structural Health Monitoring*, 3(1):85–98, 2004.

[36] Yiming Yang, Jian Zhang, Jaime Carbonell, and Chun Jin. Topic-conditioned novelty detection. In *Proceedings of the 8th Conference on Knowledge Discovery in Data archive*, pages 688–693, 2002.

[37] D.Y. Yeung and C. Chow. Parzen-window network intrusion detection. In *16th International Conference on Pattern Recognition*, pages 11–15, Quebec City, Canada, 2002.

[38] Z. Zabinsky, R.L. Smith, J.F. McDonald, H.E. Romeijn, and D.E. Kaufman. Improving hit-and-run for global optimization. *Journal of Global Optimization*, 3:171–192, 1993.

[39] Stefano Zanero and Sergio M. Savaresi. Unsupervised learning techniques for an intrusion detection system. In *Proceedings of the 2004 ACM symposium on Applied computing*, pages 412–419, Nicosia, Cyprus, 2004.

[40] Rong Zhang and Alexander I. Rudnicky. A large scale clustering scheme for kernel k-means. In *4th International Conference on Pattern Recognition (ICPR)*, 2002.

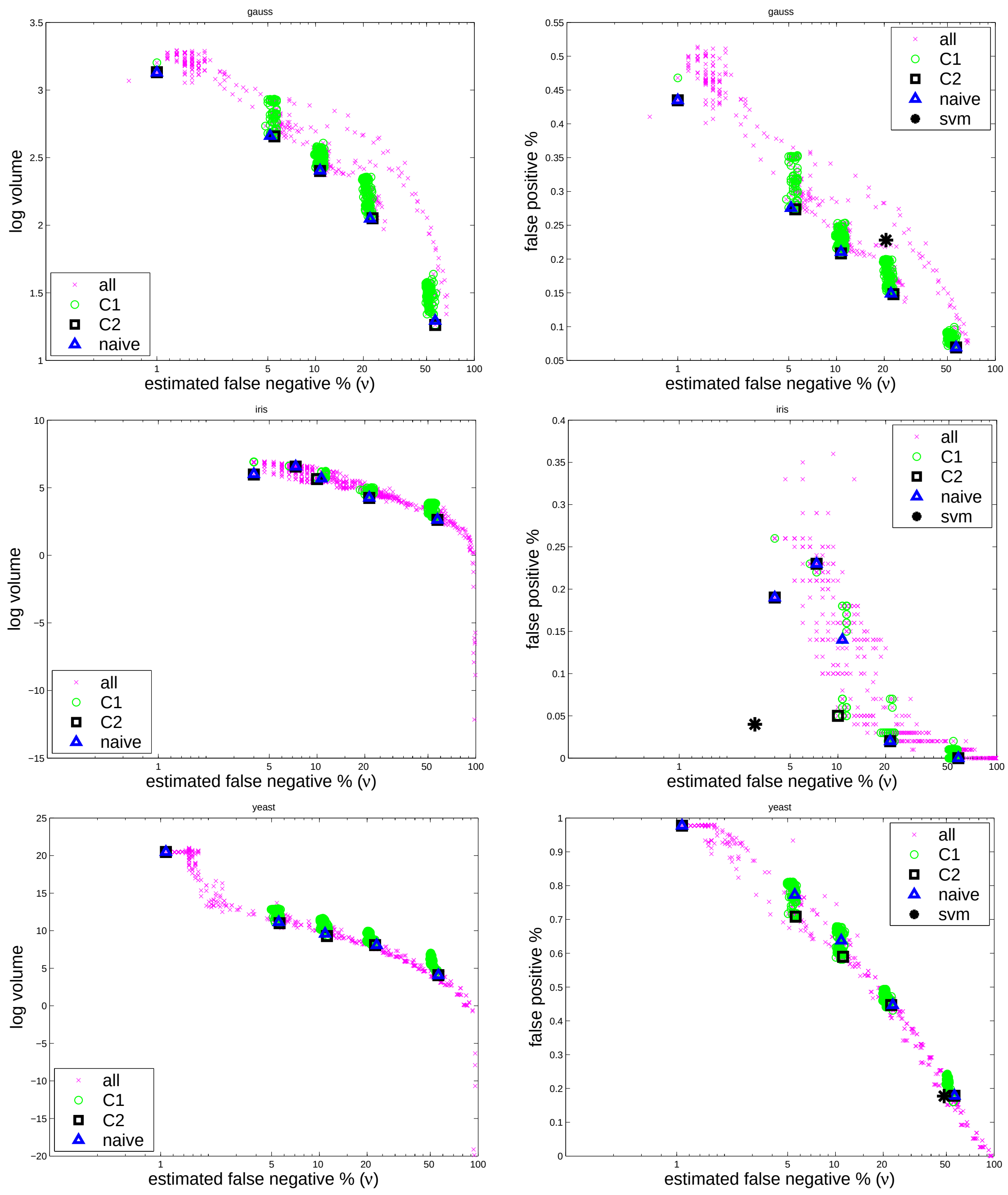


Figure 15: Model selection for *gauss*, *iris*, and *yeast*. **Left**: model selection results. The x-axis represents the estimated false negative rate ($\hat{\nu}$), and the y-axis log volume. All the models tested are shown as x's. Models satisfying C1 are shown as circles. The models with the smallest volume, for a given $\nu^{\star}$, are shown as squares (*volume*). The models selected from C1 using *naive* are shown as triangles. **Right**: test performance. The same models are shown, but the y-axis now shows the test error on the abnormal data class (false positives). The x-axis is unchanged. The optimal two-class svm is shown as a star.

Figure 16: Model selection for *letter*, *derm*, and *ion*. **Left**: model selection results. The x-axis represents the estimated false negative rate ($\hat{\nu}$), and the y-axis log volume. All the models tested are shown as x's. Models satisfying C1 are shown as circles. The models with the smallest volume, for a given $\nu^\star$, are shown as squares (*volume*). The models selected from C1 using *naive* are shown as triangles. **Right**: test performance. The same models are shown, but the y-axis now shows the test error on the abnormal data class (false positives). The x-axis is unchanged. The optimal two-class svm is shown as a star.

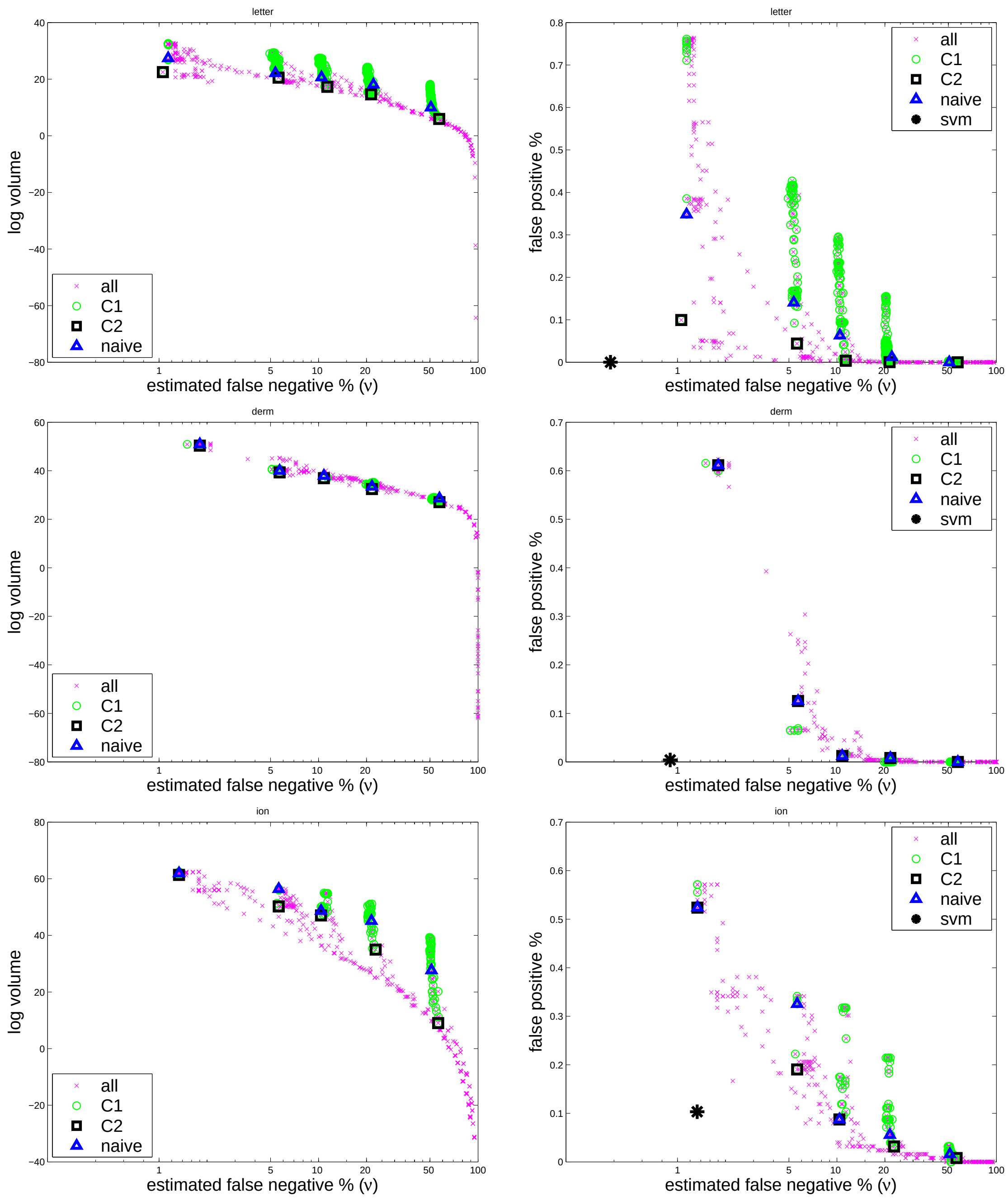
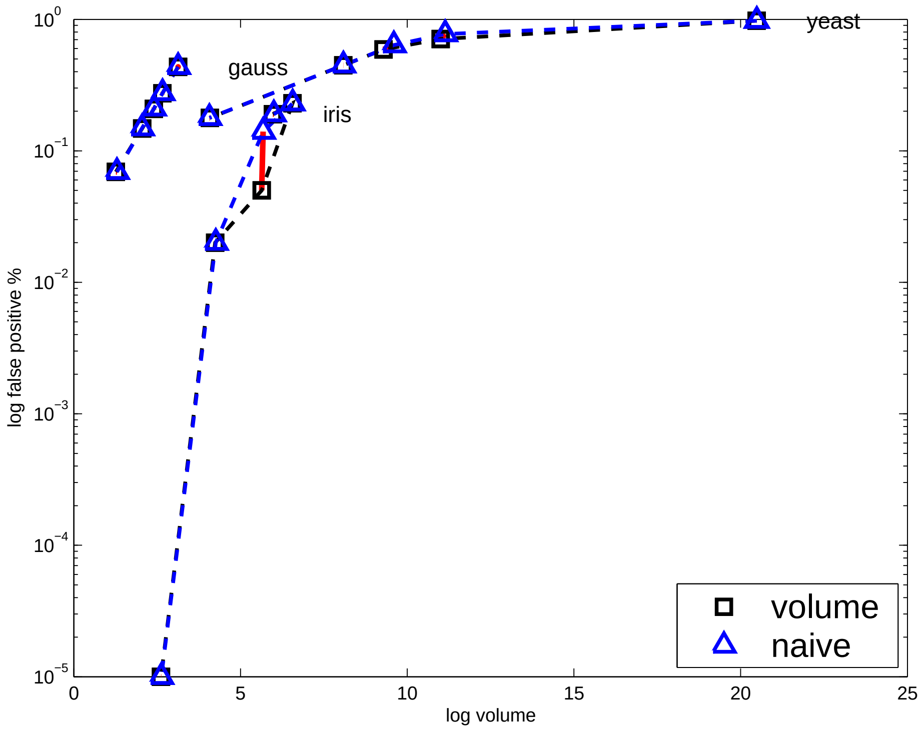
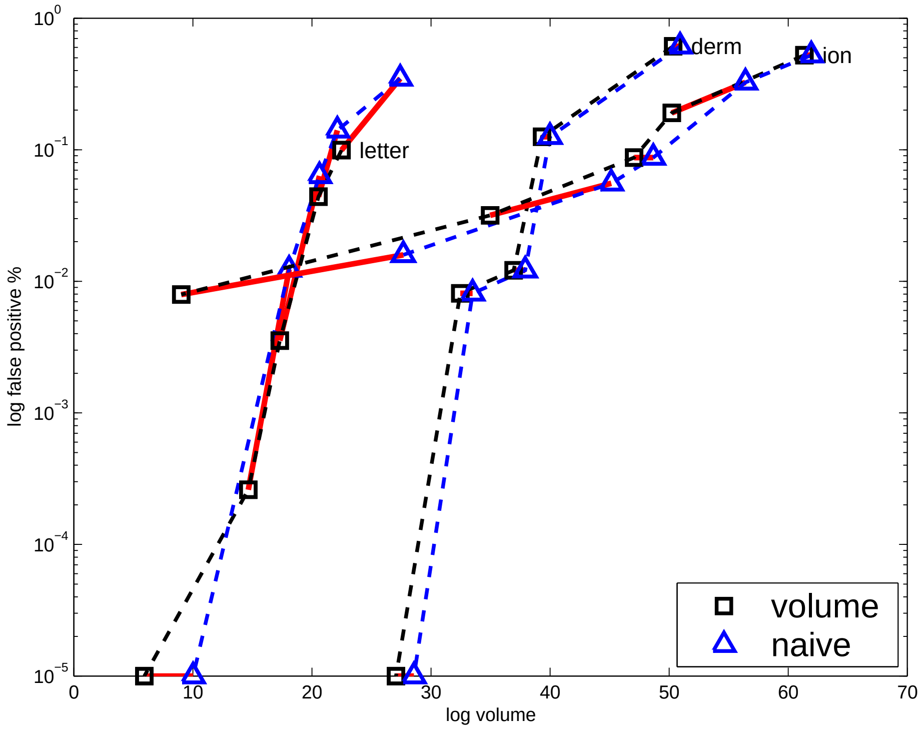


Figure 17: Summary: log volume vs. log false positive rate. **Top**: summary of the results of *gauss*, *iris*, and *yeast*. **Bottom**: summary of the results of *letter*, *derm*, and *ion*. For each value of $\nu^\star$, the optimal model chosen using either our *volume* (squares), or *naive* (triangles), is plotted using a dashed line. Solid lines link the chosen models for given a $\nu^\star$ and indicate where there is a gap in either test error performance or volume. This is used to illustrate the performance improvements of *volume* over *naive*. Zero error values are replaced with $1e-5$ so that they appear on the plots.


gauss
iris
yeast
log false positive %
log volume
volume
naive


letter
derm
ion
log false positive %
log volume
volume
naive