Algorithmica manuscript No.
(will be inserted by the editor)

Randomized partition trees for nearest neighbor search

Sanjoy Dasgupta · **Kaushik Sinha**

the date of receipt and acceptance should be inserted later

Abstract The k-d tree was one of the first spatial data structures proposed for nearest neighbor search. Its efficacy is diminished in high-dimensional spaces, but several variants, with randomization and overlapping cells, have proved to be successful in practice. We analyze three such schemes. We show that the probability that they fail to find the nearest neighbor, for any data set and any query point, is directly related to a simple potential function that captures the difficulty of the point configuration. We then bound this potential function in several situations of interest: when the data are drawn from a doubling measure; when the data and query distributions are identical and are supported on a set of bounded doubling dimension; and when the data are documents from a topic model.

1 Introduction

The problem of nearest neighbor search has engendered a vast body of algorithmic work. In the most basic formulation, there is a set S of n points, typically in an Euclidean space $\mathbb{R}^d$, and any subsequent query point must be answered by its nearest neighbor (NN) in S. A simple solution is to store S as a list, and to address queries using a linear-time scan of the list. The challenge is to achieve a substantially smaller query time than this.

We will consider a prototypical modern application in which the number of points n and the dimension d are both large. The primary resource constraints are the size

A preliminary abstract of this work appeared in [11].

S. Dasgupta
University of California, San Diego
E-mail: dasgupta@cs.ucsd.edu

K. Sinha
Wichita State University
E-mail: kaushik.sinha@wichita.edu

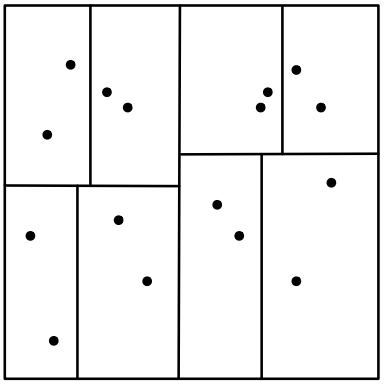
of the data structure used to store S and the amount of time taken to answer queries. For practical purposes, the former must be $O(n)$, or maybe a little more, and the latter must be $o(n)$. Secondary constraints include the time to build the data structure and, sometimes, the time to add new points to S or to remove existing points from S.

A major finding of the past two decades has been that these resource bounds can be met if it is enough to merely return a *c-approximate nearest neighbor*, whose distance from the query is at most c times that of the true nearest neighbor. One such method that has been successful in practice is *locality sensitive hashing* (LSH), which has space requirement $n^{1+\rho}$ and query time $O(n^{\rho})$, for $\rho \approx 1/c^2$ [2]. It makes use of random projection, which is also the basis of some other approximate NN methods [1,14]. A rather different approach is the *balanced box decomposition tree*, which takes $O(n)$ space and answers queries with an approximation factor $c = 1+\varepsilon$ in $O((6/\varepsilon)^d \log n)$ time [3].

In the latter result, an exponential dependence on dimension is evident, and indeed this is a familiar blot on the nearest neighbor landscape. One way to mitigate the curse of dimensionality is to consider situations in which data have low *intrinsic dimension* d_o, even if they happen to lie in $\mathbb{R}^d$ for $d \gg d_o$ or in a general metric space.

A common assumption is that the data are drawn from a *doubling measure* of dimension d_o (or equivalently, have *expansion rate* 2^{d_o}); this is defined in Section 4.1 below. Under this condition, there is a scheme [13] that gives exact answers to nearest neighbor queries in time $O(2^{3d_o} \log n)$, using a data structure of size $O(2^{3d_o} n)$. The more recent *cover tree* algorithm [6], which has been used quite widely, creates a data structure in space $O(n)$ and answers queries in time $O(2^{d_o} \log n)$. There is also work that combines intrinsic dimension and approximate search. The *navigating net* [15], given data from a metric space of *doubling dimension* d_o, has size $O(2^{O(d_o)} n)$ and gives a $(1+\varepsilon)$-approximate answer to queries in time $O(2^{O(d_o)} \log n + (1/\varepsilon)^{O(d_o)})$; the crucial advantage here is that doubling dimension, defined in Section 4.2 below, is a more general and robust notion than doubling measure. Finally, there are results for exact NN search in spaces of doubling dimension d_o, provided the queries come from the same distribution as the data [8,9]. These are similar to the bounds for doubling measures, except that there is also a dependence on the *aspect ratio* of the data: the ratio between the largest and smallest interpoint distances.

Despite these and many other results, there are two significant deficiencies in the nearest neighbor literature that have motivated the present paper. First, existing analyses have succeeded at identifying, for a given data structure, highly specific families of data for which efficient exact NN search is possible—for instance, data from doubling measures—but have failed to provide a more general characterization. Second, there remains a class of nearest neighbor data structures that are popular and successful in practice, but that have not been analyzed thoroughly. These structures combine classical k-d tree partitioning with randomization and overlapping cells, and are the subject of this paper.



function MakeTree(S)
  If $|S| < n_o$: return (Leaf)
  Rule = **ChooseRule**(S)
  LeftTree = **MakeTree**($\{x \in S : \text{Rule}(x) = \text{true}\}$)
  RightTree = **MakeTree**($\{x \in S : \text{Rule}(x) = \text{false}\}$)
  return (Rule, LeftTree, RightTree)

function ChooseRule(S)
  Choose a coordinate direction i
  $\text{Rule}(x) = (x_i \leq \text{median}(\{z_i : z \in S\}))$
  return (Rule)

Fig. 1 The k-d tree: example and pseudocode.

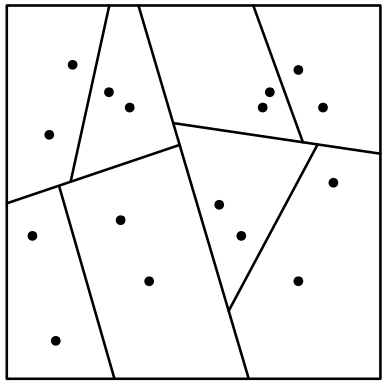
1.1 Three randomized tree structures for exact NN search

The k-d tree is a partition of $\mathbb{R}^d$ into hyper-rectangular cells, based on a set of data points [5]. The root of the tree is a single cell corresponding to the entire space. A coordinate direction is chosen, and the cell is split at the median of the data along this direction (Figure 1). The process is then recursed on the two newly created cells, and continues until all leaf cells contain at most some predetermined number n_o of points. When there are n data points, the depth of the tree is at most about $\log(n/n_o)$.

Given a k-d tree built from data points S, there are several ways to answer a nearest neighbor query q. The quickest and dirtiest of these is to move q down the tree to its appropriate leaf cell, and then return the nearest neighbor in that cell. This *defeatist search* takes time just $O(n_o + \log(n/n_o))$, which is $O(\log n)$ for constant n_o. The problem is that q's nearest neighbor may well lie in a different cell, for instance when the data happen to be concentrated near cell boundaries. Consequently, the failure probability of this scheme can be unacceptably high.

Over the years, some simple tricks have emerged, from various sources, for reducing the failure probability. These are nicely laid out by the authors of [16], who show experimentally that the resulting algorithms are effective in practice.

The first trick is to introduce randomness into the tree. Drawing inspiration from locality-sensitive hashing, [16] suggests preprocessing the data set S by randomly rotating it, and then applying a k-d tree (or related tree structure). This is rather like splitting cells along random directions as opposed to coordinate axes. In this paper, we consider a data structure that uses random split directions as well as a second type of randomization: instead of putting the split point exactly at the median, it is placed at a fractile chosen uniformly at random from the range $[1/4, 3/4]$. The resulting structure (Figure 2) is almost exactly the *random projection tree* (or RP tree) of [10]. That earlier work showed that in RP trees, the diameters of the cells decrease (down the tree) at a rate depending only on the intrinsic dimension of the data. It is a curious result, but is not helpful in analyzing nearest neighbor search, and in this paper we develop a different line of reasoning.



function ChooseRule(S)
Pick U uniformly at random from the unit sphere
Pick β uniformly at random from $[1/4, 3/4]$
Let $v = \beta$-fractile point of the projection of S on U
Rule$(x) = (x \cdot U \leq v)$
return (Rule)

Fig. 2 The random projection tree (RP tree)

A second trick suggested by [16] for reducing failure probability is to allow overlap between cells. This was also proposed in earlier work [17]. Once again, each cell C is split along a direction $U(C)$ chosen at random from the unit sphere. But now, three split points are noted: the median $m(C)$ of the data along direction U, the $(1/2)-\alpha$ fractile value $l(C)$, and the $(1/2)+\alpha$ fractile value $r(C)$. Here α is a small constant, like 0.05 or 0.1. The idea is to simultaneously entertain a *median split*

$$\text{left} = \{x : x \cdot U < m(C)\} \quad \text{right} = \{x : x \cdot U \geq m(C)\}$$

and an *overlapping split* (with the middle 2α fraction of the data falling on both sides)

$$\text{left} = \{x : x \cdot U < r(C)\} \quad \text{right} = \{x : x \cdot U \geq l(C)\}.$$

In the *spill tree* [16], each data point in S is stored in multiple leaves, by following the overlapping splits. A query is then answered defeatist-style, by routing it to a single leaf using median splits.

Both the RP tree and the spill tree have query times of $O(n_o + \log(n/n_o))$, but the latter can be expected to have a lower failure probability, and we will see this in the bounds we obtain. On the other hand, the RP tree requires just linear space, while the size of the spill tree is $O(n^{1/(1-\lg(1+2\alpha))})$. When $\alpha = 0.05$, for instance, the size is $O(n^{1.159})$.

In view of these tradeoffs, we consider a further variant, which we call the *virtual spill tree*. It stores each data point in a single leaf, following median splits, and hence has linear size. However, each query is routed to multiple leaves, using overlapping splits, and the return value is its nearest neighbor in the union of these leaves.

The various splits are summarized in Figure 3, and the three trees use them as follows:

	Routing data	Routing queries
RP tree	Perturbed split	Perturbed split
Spill tree	Overlapping split	Median split
Virtual spill tree	Median split	Overlapping split

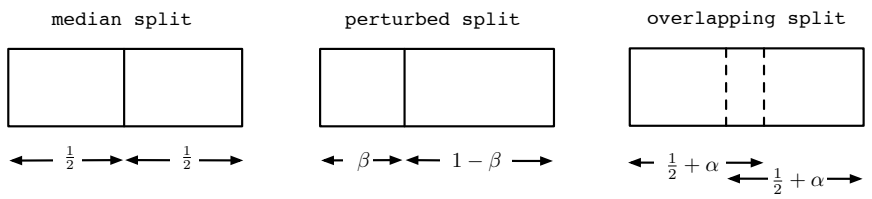


Fig. 3 Three types of split. The fractions refer to probability mass. α is some constant, while β is chosen uniformly at random from $[1/4, 3/4]$.

One small technicality: if, for instance, there are duplicates among the data points, it might not be possible to achieve a median split, or a split at a desired fractile. We will ignore these discretization problems.

1.2 Analysis of failure probability

Our three schemes for nearest neighbor search—the RP tree and the two spill trees—can be analyzed in a simple and unified framework. Pick any data set $x_1, \ldots, x_n \in \mathbb{R}^d$ and any query $q \in \mathbb{R}^d$. The probability of not finding the nearest neighbor, assessed over the randomness in the data structure, can be shown to be directly related to the quantity

$$\Phi(q, \{x_1, \ldots, x_n\}) \;=\; \frac{1}{n}\sum_{i=2}^{n} \frac{\|q - x_{(1)}\|}{\|q - x_{(i)}\|},$$

where $x_{(1)}, x_{(2)}, \ldots$ denotes an ordering of the x_i by increasing distance from q. For RP trees, the failure probability is proportional to $\Phi \log(1/\Phi)$ (Theorem 4); for the two spill trees, it is proportional to Φ (Theorem 3). The results extend easily to the problem of searching for the k nearest neighbors. Moreover, these bounds are roughly tight: a failure probability proportional to Φ is inevitable unless there is a significant amount of collinearity within the data (Corollary 1).

Let's take a closer look at this potential function. If Φ is close to 1, then all the points are roughly the same distance from q, and so we can expect that the NN query is not easy to answer. On the other hand, if Φ is close to zero, then most of the points are much further away than the nearest neighbor, so the latter should be easy to identify. Thus the potential function is an intuitively reasonable measure of the difficulty of NN search.

This general characterization of data configurations amenable to efficient exact NN search, by the three data structures, is our main result. To illustrate our theorem, we bound Φ for three commonly-studied data types.

- When $x_1, \ldots, x_n$ are drawn i.i.d. from a doubling measure (Section 4.1). As we discussed earlier, this is the assumption under which many other results for exact NN search have been obtained.
- When the query q is *exchangeable* with the data $x_1, \ldots, x_n$—that is, q is a random draw from $\{x_1, \ldots, x_n, q\}$—and they together form a set of bounded doubling dimension (Section 4.2).

– When $x_1, \ldots, x_n$ are documents drawn from a topic model (Section 4.3).

In the first case, when data are from a doubling measure of dimension d_o, we show that the spill tree is able to answer arbitrary exact nearest neighbor queries in time $O(d_o)^{d_o} + O(\log n)$, with a probability of error that is an arbitrarily small constant, while the RP tree is slower by only a logarithmic factor (Theorem 7). These are close to the best results that have been obtained using other data structures. (The failure probability is over the randomization in the tree structure, and can be further reduced by building multiple trees.) Similar results hold for the second case (Theorem 8), with d_o now denoting the doubling dimension and with an additional dependence on the aspect ratio of the data, as in prior work [8,9]. Finally, we chose the topic model as an example of a significantly harder case: its data distribution is more concentrated than that of a doubling measure, in the sense that there are a lot of data points that are only slightly further away than the nearest neighbor. The resulting savings are far more modest though non-negligible: for large n, the time to answer a query is roughly $n \cdot 2^{-O(\sqrt{L})}$, where L is the expected document length.

In some situations, the time to construct the data structure, and the ability to later add or remove data points, are significant factors. It is readily seen that the construction time for the spill tree is proportional to its size, while that of the RP tree and the virtual spill is $O(n \log n)$. Adding and removing points is also easy: all guarantees hold if these are performed locally, while rebuilding the entire data structure after every $O(n)$ such operations.

2 A potential function for point configurations

To motivate the potential function Φ, we start by considering what happens when there are just two data points and one query point.

2.1 How random projection affects the relative placement of three points

Consider any three points $q, x, y \in \mathbb{R}^d$, such that x is closer to q than is y; that is, $\|q-x\| \leq \|q-y\|$.

Now suppose that a direction U is chosen uniformly at random from the unit sphere S^{d-1}, and that the points are projected onto this direction. What is the probability that y falls between q and x on this line? The following lemma answers this question exactly. An approximate solution, with different proof method, was given earlier in [14].

Lemma 1 *Pick any $q, x, y \in \mathbb{R}^d$ with $\|q-x\| \leq \|q-y\|$. Pick a direction U uniformly at random from the unit sphere. The probability, over the choice of U, that $y \cdot U$ falls strictly between $q \cdot U$ and $x \cdot U$ is*

$$\frac{1}{\pi} \arcsin\left(\frac{\|q-x\|}{\|q-y\|} \sqrt{1 - \left(\frac{(q-x)\cdot(y-x)}{\|q-x\|\,\|y-x\|} \right)^2} \right).$$

Proof We may assume that U is drawn from $N(0, I_d)$, the d-dimensional Gaussian with mean zero and unit covariance. This gives exactly the right distribution if we scale U to unit length, but we can skip this last step since it has no effect on the question we are considering.

We can also assume, without loss of generality, that q lies at the origin and that x lies along the (positive) x_1-axis: that is, $q = 0$ and $x = \|x\|e_1$. It will then be helpful to split the direction U into two pieces, its component U_1 in the x_1-direction, and the remaining $d-1$ coordinates U_R. Likewise, we will write $y = (y_1, y_R)$.

If $y_R = 0$ then x, y, and q are collinear, and the projection of y cannot possibly fall between those of x and q. In what follows, we assume $y_R \neq 0$.

Let E denote the event of interest:

$$\begin{aligned} E &\equiv y \cdot U \text{ falls between } q \cdot U \text{ (that is, 0) and } x \cdot U \text{ (that is, } \|x\|U_1) \\ &\equiv y_R \cdot U_R \text{ falls between } -y_1 U_1 \text{ and } (\|x\| - y_1)U_1 \end{aligned}$$

The interval of interest is either $(-y_1|U_1|, (\|x\| - y_1)|U_1|)$, if $U_1 \geq 0$, or $(-(\|x\| - y_1)|U_1|, y_1|U_1|)$, if $U_1 < 0$. To simplify things, $y_R \cdot U_R$ is independent of U_1 and is distributed as $N(0, \|y_R\|^2)$, which is symmetric and thus assigns the same probability mass to the two intervals. We can therefore write

$$\Pr_U(E) \;=\; \Pr_{U_1}\Pr_{U_R}(-y_1|U_1| < y_R \cdot U_R < (\|x\| - y_1)|U_1|).$$

Let Z and Z' be independent standard normals $N(0,1)$. Since U_1 is distributed as Z and $y_R \cdot U_R$ is distributed as $\|y_R\|Z'$,

$$\Pr_U(E) \;=\; \Pr(-y_1|Z| < \|y_R\|Z' < (\|x\| - y_1)|Z|) \;=\; \Pr\left(\frac{Z'}{|Z|} \in \left(-\frac{y_1}{\|y_R\|}, \frac{\|x\| - y_1}{\|y_R\|}\right)\right).$$

Now $Z'/|Z|$ is the ratio of two standard normals, which has a standard Cauchy distribution. Using the formula for a Cauchy density,

$$\begin{aligned} \Pr(E) &= \int_{-y_1/\|y_R\|}^{(\|x\| - y_1)/\|y_R\|} \frac{\mathrm{d}w}{\pi(1+w^2)} \\ &= \frac{1}{\pi}\left(\arctan\left(\frac{\|x\| - y_1}{\|y_R\|}\right) - \arctan\left(\frac{-y_1}{\|y_R\|}\right)\right) \\ &= \frac{1}{\pi}\arctan\frac{\|x\|\,\|y_R\|}{\|y\|^2 - y_1\|x\|} \\ &= \frac{1}{\pi}\arcsin\left(\frac{\|x\|}{\|y\|} \cdot \sqrt{\frac{\|y\|^2 - y_1^2}{\|y\|^2 + \|x\|^2 - 2y_1\|x\|}}\right), \end{aligned}$$

which is exactly the expression in the lemma statement once we invoke $y_1 = (y \cdot x)/\|x\|$ and factor in our assumption that $q = 0$.

To simplify the expression, define an index of the *collinearity* of q, x, y to be

$$\mathrm{coll}(q,x,y) \;=\; \frac{|(q-x)\cdot(y-x)|}{\|q-x\|\,\|y-x\|}.$$

This value, in the range $[0,1]$, is 1 when the points are collinear, and 0 when $q - x$ is orthogonal to $x - y$.

Corollary 1 *Under the conditions of Lemma 1,*

$$\frac{1}{\pi}\frac{\|q-x\|}{\|q-y\|}\sqrt{1-\text{coll}(q,x,y)^2} \;\leq\; \Pr_U(y\cdot U \text{ falls between } q\cdot U \text{ and } x\cdot U) \;\leq\; \frac{1}{2}\frac{\|q-x\|}{\|q-y\|}.$$

Proof Apply the inequality $\theta \geq \sin\theta \geq 2\theta/\pi$ for all $0 \leq \theta \leq \pi/2$.

The upper and lower bounds of Corollary 1 are within a constant factor of each other unless the points are approximately collinear.

2.2 By how much does random projection separate nearest neighbors?

For a query q and data points $x_1,\ldots,x_n$, let $x_{(1)},x_{(2)},\ldots$ denote a re-ordering of the points by increasing distance from q. Consider the potential function

$$\Phi(q,\{x_1,\ldots,x_n\}) \;=\; \frac{1}{n}\sum_{i=2}^{n}\frac{\|q-x_{(1)}\|}{\|q-x_{(i)}\|}.$$

Theorem 1 *Pick any points $q,x_1,\ldots,x_n \in \mathbb{R}^d$. If these points are projected to a direction U chosen at random from the unit sphere, then*

$$\mathbb{E}_U(\textit{fraction of the projected } x_i \textit{ that fall between } q \textit{ and } x_{(1)}) \;\leq\; \frac{1}{2}\Phi(q,\{x_1,\ldots,x_n\}).$$

Proof Let Z_i be the event that $x_{(i)}$ falls between q and $x_{(1)}$ in the projection. By Corollary 1,

$$\Pr_U(Z_i) \;\leq\; \frac{1}{2}\frac{\|q-x_{(1)}\|}{\|q-x_{(i)}\|}$$

The lemma now follows by linearity of expectation.

The upper bound of Theorem 1 is fairly tight, as can be seen from Corollary 1, unless there is a high degree of collinearity between the points.

In the tree data structures we analyze, most cells contain only a subset of the data $\{x_1,\ldots,x_n\}$. For a cell that contains m of these points, the appropriate variant of Φ is

$$\Phi_m(q,\{x_1,\ldots,x_n\}) \;=\; \frac{1}{m}\sum_{i=2}^{m}\frac{\|q-x_{(1)}\|}{\|q-x_{(i)}\|}.$$

Corollary 2 *Pick any points $q,x_1,\ldots,x_n$ and let S denote any subset of the x_i that includes $x_{(1)}$. Project q and the points in S onto a direction U chosen at random from the unit sphere. Then for any $0<\alpha<1$,*

$$\Pr_U(\textit{at least an } \alpha \textit{ fraction of } S \textit{ falls between } q \textit{ and } x_{(1)} \textit{ when projected})$$
$$\leq \frac{1}{2\alpha}\Phi_{|S|}(q,\{x_1,\ldots,x_n\}).$$

Proof This follows immediately by applying Theorem 1 to S, noting that the corresponding value of Φ is maximized when S consists of the points closest to q, and then applying Markov's inequality.

2.3 Extension to k nearest neighbors

If we are interested in finding the k nearest neighbors, a suitable generalization of Φ_m is

$$\Phi_{k,m}(q,\{x_1,\ldots,x_n\}) \;=\; \frac{1}{m}\sum_{i=k+1}^{m}\frac{(\|q-x_{(1)}\|+\cdots+\|q-x_{(k)}\|)/k}{\|q-x_{(i)}\|}.$$

Theorem 2 *Pick any points $q,x_1,\ldots,x_n$ and let S denote any subset of the x_i that includes $x_{(1)},\ldots,x_{(k)}$. Suppose q and the points in S are projected to a direction U chosen at random from the unit sphere. Then, for any $0<\alpha<1$,*

$$\Pr_U(\exists 1\le j\le k \text{ such that } \ge \alpha|S| \text{ points fall between } q \text{ and } x_{(j)} \text{ when projected}) \le \frac{k}{2(\alpha-(k-1)/|S|)}\Phi_{k,|S|}(q,\{x_1,\ldots,x_n\}).$$

provided $k<\alpha|S|+1$.

Proof Set $m=|S|$. As in Corollary 2, the probability of the bad event is maximized when $S=\{x_{(1)},\ldots,x_{(m)}\}$, so we will assume as much.

For any $1\le j\le k$, let N_j denote the number of points in $\{x_{(k+1)},\ldots,x_{(m)}\}$ that fall (strictly) between q and $x_{(j)}$ in the projection. Reasoning as in Theorem 1, we have

$$\Pr_U(N_j\ge \alpha m-(k-1)) \;\le\; \frac{\mathbb{E}_U N_j}{\alpha m-(k-1)} \;\le\; \frac{1}{2(\alpha m-(k-1))}\sum_{i=k+1}^{m}\frac{\|q-x_{(j)}\|}{\|q-x_{(i)}\|}.$$

Taking a union bound over all $1\le j\le k$,

$$\begin{aligned}&\Pr_U(\exists 1\le j\le k : N_j\ge \alpha m-(k-1))\\ &\le \frac{1}{2(\alpha m-(k-1))}\sum_{i=k+1}^{m}\frac{\|q-x_{(1)}\|+\cdots+\|q-x_{(k)}\|}{\|q-x_{(i)}\|}\\ &= \frac{k}{2(\alpha-(k-1)/m)}\Phi_{k,m}(q,\{x_1,\ldots,x_n\}),\end{aligned}$$

as claimed.

3 Randomized partition trees

We'll now see that the failure probability of the random projection tree is proportional to $\Phi\ln(1/\Phi)$, while that of the two spill trees is proportional to Φ. We start with the second result, since it is the more straightforward of the two.

3.1 Randomized spill trees

In a randomized spill tree, each cell is split along a direction chosen uniformly at random from the unit sphere. Two kinds of splits are simultaneously considered: (1) a split at the median (along the random direction), and (2) an overlapping split with one part containing the bottom $1/2+\alpha$ fraction of the cell's points, and the other part containing the top $1/2+\alpha$ fraction, where $0 < \alpha < 1/2$ (recall Figure 3).

We consider two data structures that use these splits in different ways. The *spill tree* stores each data point in (possibly) multiple leaves, using overlapping splits. The tree is grown until each leaf contains at most n_o points. A query is answered by routing it to a single leaf, using median splits, and returning the NN in that leaf.

The time to answer a query is just $O(n_o + \log(n/n_o))$, but the space requirement of this data structure is super-linear. Its depth is $\ell = \log_{1/\beta} n/n_o$ levels, where $\beta = (1/2)+\alpha$, and thus the total size is

$$n_o 2^\ell \;=\; n_o \left(\frac{n}{n_o}\right)^{\log_{1/\beta} 2}.$$

We will take n_o to be a constant independent of n, so this size is $O(n^{\log_{1/\beta} 2})$. When $\alpha = 0.05$, for instance, the size is $O(n^{1.159})$. When $\alpha = 0.1$, it is $O(n^{1.357})$.

A *virtual spill tree* stores each data point in a single leaf, using median splits, once again growing the tree until each leaf has n_o or fewer points. Thus the total size is just $O(n)$ and the depth is $\log_2(n/n_o)$. However, a query is answered by routing it to multiple leaves using overlapping splits, and then returning the NN in the union of these leaves.

Theorem 3 *Suppose a randomized spill tree is built using data points $\{x_1,\ldots,x_n\}$, to depth $\ell = \lceil \log_{1/\beta}(n/n_o)\rceil$, where $\beta = (1/2)+\alpha$ for regular spill trees and $\beta = 1/2$ for virtual spill trees. Assume for convenience that the $\beta^i n$ are integers for all $0 \le i \le \ell$. If this tree is used to answer a query q, then the probability (over randomization in the construction of the tree) that it fails to return $x_{(1)}$ is at most*

$$\frac{1}{2\alpha}\sum_{i=0}^{\ell} \Phi_{\beta^i n}(q, \{x_1,\ldots,x_n\}).$$

The probability that it fails to return the $k > 1$ nearest neighbors $x_{(1)},\ldots,x_{(k)}$ is at most

$$\frac{k}{\alpha}\sum_{i=0}^{\ell} \Phi_{k,\beta^i n}(q, \{x_1,\ldots,x_n\}),$$

provided $k \le \alpha n_o/2$.

Proof Let's start with the regular spill tree. Consider the internal node at depth i on the root-to-leaf path of query q; this node contains $\beta^i n$ data points, for $\beta = (1/2)+\alpha$. What is the probability that q gets separated from $x_{(1)}$ when the node is split? This bad event can only happen if q and $x_{(1)}$ lie on opposite sides of the median and if $x_{(1)}$ is transmitted only to one side of the split, that is, if at least α fraction of the points

lie between $x_{(1)}$ and the median. This means that at least an α fraction of the cell's projected points must fall between q and $x_{(1)}$, which occurs with probability at most $(1/2\alpha)\Phi_{\beta^i n}(q,\{x_1,\ldots,x_n\})$ by Corollary 2. The lemma follows by summing over all levels i.

The argument for the virtual spill tree is identical, except that we use $\beta = 1/2$ and we swap the roles of q and $x_{(1)}$; for instance, we consider the root-to-leaf path of $x_{(1)}$.

The generalization to k nearest neighbors is immediate for spill trees. The probability of something going wrong at level i of the tree is, by Theorem 2, at most

$$\frac{k}{2(\alpha-(k-1)/n_o)}\Phi_{k,\beta^i n} \;\leq\; \frac{k}{\alpha}\Phi_{k,\beta^i n}.$$

Virtual spill trees require a slightly more careful argument. If the root-to-leaf path of each $x_{(j)}$, for $1 \leq j \leq k$, is considered separately, it can be shown that the total probability of failure at level i is again bounded by the same expression.

We will encounter two functional forms of Φ_m: either $1/m^{1/d_o}$, where d_o captures the intrinsic dimension of the data, or a small constant $1/\sqrt{L}$, where L is the expected document size under a topic model. In the former case, the failure probability of the spill tree is roughly $1/(\alpha n_o^{1/d_o})$, and in the latter case it is $(1/(\alpha\sqrt{L}))\log(n/n_o)$. Further details are in Sections 4.1–4.3.

3.2 Random projection trees

In an RP tree, a cell is split by choosing a direction uniformly at random from the unit sphere S^{d-1}, projecting the points in the cell onto that direction, and then splitting at the β fractile, for β chosen uniformly at random from $[1/4,3/4]$. As in a k-d tree, each point is mapped to a single leaf. Likewise, a query point is routed to a particular leaf, and its nearest neighbor within that leaf is returned.

In many of the statements below, we will drop the arguments $(q,\{x_1,\ldots,x_n\})$ of Φ in the interest of readability.

Theorem 4 *Suppose an RP tree is built using points $\{x_1,\ldots,x_n\}$ and is then used to answer a query q. The probability (over the randomization in tree construction) that it fails to return the nearest neighbor of q is at most*

$$\sum_{i=0}^{\ell} \Phi_{\beta^i n} \ln \frac{2e}{\Phi_{\beta^i n}},$$

where $\beta = 3/4$ and $\ell = \lceil \log_{1/\beta}(n/n_o)\rceil$, and we are assuming for convenience that the $\beta^i n$ are integers for all $0 \leq i \leq \ell$. The probability that it fails to return the k nearest neighbors of q is at most

$$\left(2k\sum_{i=0}^{\ell} \Phi_{k,\beta^i n} \ln \frac{2e}{k\Phi_{k,\beta^i n}}\right) + \frac{16(k-1)}{n_o}.$$

Proof Consider any internal node of the tree that contains q as well as m of the data points, including $x_{(1)}$. What is the probability that the split at that node separates q from $x_{(1)}$? To analyze this, let $F \in \{0/m, 1/m, \ldots, (m-1)/m\}$ denote the fraction of the m points that fall between q and $x_{(1)}$ along the randomly-chosen split direction. Since the split point is chosen at random from an interval of mass $1/2$, the probability that it separates q from $x_{(1)}$ is at most $F/(1/2)$. Summing out F, we get

$$\begin{aligned}
\Pr(q \text{ is separated from } x_{(1)}) &\leq \sum_{i=0}^{m-1} \Pr(F = i/m)\frac{i/m}{1/2} \\
&= \frac{2}{m}\sum_{i=1}^{m-1} \Pr(F \geq i/m) \\
&\leq \frac{2}{m}\sum_{i=1}^{m-1} \min\left(1, \frac{\Phi_m}{2i/m}\right) \\
&= \frac{2}{m}\sum_{i=1}^{\lfloor m\Phi_m/2\rfloor} 1 + \frac{2}{m}\sum_{i=\lfloor m\Phi_m/2\rfloor+1}^{m-1} \frac{\Phi_m}{2i/m} \\
&= \frac{2}{m}\left\lfloor \frac{m\Phi_m}{2} \right\rfloor + \Phi_m \sum_{i=\lfloor m\Phi_m/2\rfloor+1}^{m-1} \frac{1}{i} \;\leq\; \Phi_m \ln \frac{2e}{\Phi_m},
\end{aligned}$$

where the second inequality uses Corollary 2.

The lemma follows by taking a union bound over the path that conveys q from root to leaf, in which the number of data points per level shrinks geometrically, by a factor of $3/4$ or better.

The same reasoning generalizes to k nearest neighbors. This time, F is defined to be the fraction of the m points that lie between q and the furthest of $x_{(1)}, \ldots, x_{(k)}$ along the random splitting direction. Then q is separated from one of these neighbors only if the split point lies in an interval of mass F on either side of q, an event that

occurs with probability at most $2F/(1/2)$. Using Theorem 2,

$$\begin{aligned}
&\Pr(q \text{ is separated from some } x_{(j)},\ 1 \le j \le k) \\
&\le \sum_{i=0}^{m-1} \Pr(F = i/m) \frac{2i/m}{1/2} \\
&= \frac{4}{m} \sum_{i=1}^{m-1} \Pr(F \ge i/m) \\
&\le \frac{4}{m} \sum_{i=1}^{m-1} \min\left(1, \frac{k\Phi_{k,m}}{2(i-(k-1))_+/m}\right) \\
&= \frac{4}{m} \sum_{i=1}^{\lfloor km\Phi_{k,m}/2\rfloor+k-1} 1 + \frac{4}{m} \sum_{i=\lfloor km\Phi_{k,m}/2\rfloor+k}^{m-1} \frac{k\Phi_{k,m}}{2(i-(k-1))/m)} \\
&= \frac{4}{m}\left(\left\lfloor \frac{km\Phi_{k,m}}{2} \right\rfloor + k - 1\right) + 2k\Phi_{k,m} \sum_{i=\lfloor km\Phi_{k,m}/2\rfloor+1}^{m-k} \frac{1}{i} \\
&\le 2k\Phi_{k,m} \ln \frac{2e}{k\Phi_{k,m}} + \frac{4(k-1)}{m},
\end{aligned}$$

and as before, we sum this over a root-to-leaf path in the tree.

3.3 Could coordinate directions be used?

The tree data structures we have studied make crucial use of random projection for splitting cells. It would not suffice to use coordinate directions, as in k-d trees.

To see this, consider a simple example. Let q, the query point, be the origin, and suppose the data points $x_1, \ldots, x_n \in \mathbb{R}^d$ are chosen as follows:

- x_1 is the all-ones vector.
- Each $x_i, i > 1$, is chosen by picking a coordinate at random, setting its value to M, and then setting all remaining coordinates to uniform-random numbers in the range $(0,1)$. Here M is some very large constant.

For large enough M, the nearest neighbor of q is x_1. By letting M grow further, we can push $\Phi(q, \{x_1, \ldots, x_n\})$ arbitrarily close to zero, which means that our random projection methods will work admirably. However, any coordinate projection will create a disastrously large separation between q and x_1: on average, a $(1-1/d)$ fraction of the data points will fall between them.

4 Bounding Φ

The exact nearest neighbor schemes we analyze have error probabilities related to Φ, which lies in the range $[0,1]$. The worst case is when all points are equidistant, in which case Φ is exactly 1, but this is a pathological situation. Is it possible to bound Φ under simple assumptions on the data?

In this section we study three such scenarios.

4.1 Data drawn from a doubling measure

Suppose the data points are drawn from a distribution μ on $\mathbb{R}^d$ which is a *doubling measure*: that is, there exist a constant $C > 0$ and a subset $\mathscr{X} \subseteq \mathbb{R}^d$ such that

$$\mu(B(x,2r)) \;\leq\; C \cdot \mu(B(x,r)) \quad \text{for all } x \in \mathscr{X} \text{ and all } r > 0.$$

Here $B(x,r)$ is the closed Euclidean ball of radius r centered at x. To understand this condition, it is helpful to also look at an alternative formulation that is essentially equivalent: there exist a constant $d_o > 0$ and a subset $\mathscr{X} \subset \mathbb{R}^d$ such that for all $x \in \mathscr{X}$, all $r > 0$, and all $\alpha \geq 1$,

$$\mu(B(x,\alpha r)) \;\leq\; \alpha^{d_o} \cdot \mu(B(x,r)).$$

In other words, the probability mass of a ball grows polynomially in the radius. Comparing this to the standard formula for the volume of a ball, we see that the degree of this polynomial, d_o (which is $\log_2 C$), can reasonably be thought of as the "dimension" of the measure μ.

Theorem 5 *Suppose μ is continuous on $\mathbb{R}^d$ and is a doubling measure with dimension $d_o \geq 2$. Pick any $q \in \mathscr{X}$ and draw $x_1, \ldots, x_n$ independently at random from μ. Pick any $0 < \delta < 1/2$. Then with probability at least $1 - 3\delta$ over the choice of the x_i, for all $2 \leq m \leq n$,*

$$\Phi_m(q, \{x_1, \ldots, x_n\}) \;\leq\; 6\left(\frac{2}{m}\ln\frac{1}{\delta}\right)^{1/d_o}.$$

Proof We will consider a collection of balls $B_o, B_1, B_2, \ldots$ centered at q, with geometrically increasing radii $r_o, r_1, r_2, \ldots$, respectively. For $i \geq 1$, we will take $r_i = 2^i r_o$. Thus by the doubling condition, $\mu(B_i) \leq C^i \mu(B_o)$, where $C = 2^{d_o} \geq 4$.

Define r_o to be the radius for which $\mu(B(q,r_o)) = (1/n)\ln(1/\delta)$. This choice implies that $x_{(1)}$ is likely to fall in B_o: when points $X = \{x_1, \ldots, x_n\}$ are drawn randomly from μ,

$$\Pr(\text{no point falls in } B_o) \;=\; (1 - \mu(B_o))^n \;\leq\; \delta.$$

Next, for $i \geq 1$, the expected number of points falling in ball B_i is at most $nC^i\mu(B_o) = C^i \ln(1/\delta)$, and by a multiplicative Chernoff bound,

$$\Pr(|X \cap B_i| \geq 2nC^i\mu(B_o)) \;\leq\; \exp(-(nC^i\mu(B_o)/3)) \;=\; \delta^{C^i/3} \;\leq\; \delta^{iC/3}.$$

Summing over all i, we get

$$\Pr(\exists i \geq 1 : |X \cap B_i| \geq 2nC^i\mu(B_o)) \;\leq\; 2\delta^{C/3} \;\leq\; 2\delta.$$

We will henceforth assume that $x_{(1)}$ lies in B_o and that each B_i has at most $2n\mu(B_o)C^i = 2C^i \ln(1/\delta)$ points.

Pick any $2 \leq m \leq n$, and recall the expression for Φ:

$$\Phi_m(q, \{x_1, \ldots, x_n\}) \;=\; \frac{1}{m} \sum_{i=2}^{m} \frac{\|q - x_{(1)}\|}{\|q - x_{(i)}\|}.$$

Once $x_{(1)}$ is fixed, moving other points closer to q can only increase Φ. Therefore, the maximizing configuration has $\lfloor 2n\mu(B_o)C \rfloor$ points in B_1, followed by $\lfloor 2n\mu(B_o)C^2 \rfloor$ points in B_2, and then $\lfloor 2n\mu(B_o)C^3 \rfloor$ points in B_3, and so on. Each point in $B_j \setminus B_{j-1}$ contributes at most $1/2^{j-1}$ to the Φ summation.

Under the worst-case configuration, points $x_{(1)}, \ldots, x_{(m)}$ lie within B_ℓ, for ℓ such that

$$2n\mu(B_o)C^{\ell-1} \;<\; m \;\leq\; 2n\mu(B_o)C^{\ell}. \qquad (*)$$

We then have

$$\begin{aligned}
\Phi_m &\leq \frac{1}{m}\left(|X \cap B_1| + \left(\sum_{j=2}^{\ell-1} |X \cap (B_j \setminus B_{j-1})| \cdot \frac{1}{2^{j-1}} \right) + (m - |X \cap B_{\ell-1}|) \cdot \frac{1}{2^{\ell-1}} \right) \\
&= \frac{1}{m}\left(|X \cap B_1| + \sum_{j=2}^{\ell-1} \left(\frac{|X \cap B_j|}{2^{j-1}} - \frac{|X \cap B_{j-1}|}{2^{j-1}} \right) + (m - |X \cap B_{\ell-1}|) \cdot \frac{1}{2^{\ell-1}} \right) \\
&= \frac{1}{m}\left(\frac{m}{2^{\ell-1}} + \sum_{j=1}^{\ell-1} \frac{|X \cap B_j|}{2^j} \right) \\
&\leq \frac{1}{m}\left(\frac{m}{2^{\ell-1}} + 2n\mu(B_o) \sum_{j=1}^{\ell-1} \left(\frac{C}{2} \right)^j \right) \\
&\leq \frac{1}{m}\left(\frac{m}{2^{\ell-1}} + 4n\mu(B_o) \left(\frac{C}{2} \right)^{\ell-1} \right) \\
&\leq \frac{1}{m}\left(\frac{m}{2^{\ell-1}} + \frac{2m}{2^{\ell-1}} \right) \;=\; \frac{6}{2^\ell},
\end{aligned}$$

where the last inequality comes from (*). To lower-bound 2^ℓ, we again use (*) to get $C^\ell \geq m/(2n\mu(B_o))$, whereupon

$$2^\ell \;\geq\; \left(\frac{m}{2n\mu(B_o)} \right)^{1/\log_2 C} \;=\; \left(\frac{m}{2\ln(1/\delta)} \right)^{1/\log_2 C}$$

and we're done.

This extends easily to the potential function for k nearest neighbors.

Theorem 6 *Under the same conditions as Theorem 5, for any $k \geq 1$, we have*

$$\Phi_{k,m}(q, \{x_1, \ldots, x_n\}) \;\leq\; 6\left(\frac{8}{m} \max\left(k, \ln \frac{1}{\delta} \right) \right)^{1/d_o}.$$

Proof The only big change is in the definition of r_o; it is now the radius for which

$$\mu(B_o) \;=\; \frac{4}{n}\max\left(k, \ln\frac{1}{\delta}\right).$$

Thus, when $x_1,\ldots,x_n$ are drawn independently at random from μ, the expected number of them that fall in B_o is at least $4k$, and by a multiplicative Chernoff bound is at least k with probability $\geq 1-\delta$.

The balls $B_1, B_2, \ldots$ are defined as before, and once again, we can conclude that with probability $\geq 1-2\delta$, each B_i contains at most $2nC^i\mu(B_o)$ of the data points.

Any point $x_{(i)} \notin B_o$ lies in some annulus $B_j \setminus B_{j-1}$, and its contribution to the summation in $\Phi_{k,m}$ is

$$\frac{(\|q-x_{(1)}\|+\cdots+\|q-x_{(k)}\|)/k}{\|q-x_{(i)}\|} \;\leq\; \frac{1}{2^{j-1}}.$$

The relationship (*) and the remainder of the argument are exactly as before.

We can now give bounds on the failure probabilities of the three tree data structures.

Theorem 7 *There is an absolute constant c_o for which the following holds. Suppose μ is a doubling measure on $\mathbb{R}^d$ of intrinsic dimension $d_o \geq 2$. Pick any query $q \in \mathscr{X}$ and draw $x_1,\ldots,x_n$ independently from μ. Then with probability at least $1-3\delta$ over the choice of data:*

(a) For either variant of the spill tree, if $k \leq \alpha n_o/2$,

$$\Pr(\textit{fails to return k nearest neighbors}) \;\leq\; \frac{c_o d_o k}{\alpha}\left(\frac{8\max(k,\ln 1/\delta)}{n_o}\right)^{1/d_o}.$$

(b) For the RP tree with $n_o \geq c_o(3k)^{d_o}\max(k,\ln 1/\delta)$,

$$\Pr(\textit{fails to return k nearest neighbors}) \leq c_o k(d_o+\ln n_o)\left(\frac{8\max(k,\ln 1/\delta)}{n_o}\right)^{1/d_o}.$$

These probabilities are over the randomness in tree construction.

Proof These bounds follow immediately from Theorems 3, 4, and 6, using Lemma 6 from the appendix to bound the summation.

In order to make the failure probability an arbitrarily small constant, it is sufficient to take $n_o = O(d_o k)^{d_o}\max(k,\ln 1/\delta)$ for spill trees and $O(d_o k\ln(d_o k))^{d_o}\max(k,\ln 1/\delta)$ for RP trees.

4.2 Data of low doubling dimension

The notion of doubling measure is quite brittle: it rules out, for instance, the possibility of an isolated point mass. A more robust and widely-applicable quantity is the *doubling dimension*, a variant of the Assouad dimension [4] introduced in [12]. It applies to any metric space; here, of course, we are interested only in subsets of Euclidean space.

A set $S \subset \mathbb{R}^d$ has doubling dimension d_o if for any (Euclidean, closed) ball B, the subset $S \cap B$ can be covered by 2^{d_o} balls of half the radius. For instance, a line in $\mathbb{R}^d$ has doubling dimension 1. Here are some further examples: see [10] for details.

1. A k-dimensional affine subspace of $\mathbb{R}^d$ has doubling dimension $\leq c_o k$, for some absolute constant c_o.
2. A set $S \subset \mathbb{R}^d$ in which each element has at most k nonzero coordinates (that is, a *sparse* set) has doubling dimension at most $c_o k + k \log d$.
3. A k-dimensional Riemannian submanifold $M \subset \mathbb{R}^d$ with condition number $1/\tau$ has the property that every neighborhood of M of radius τ has doubling dimension $O(k)$.

In short, the doubling dimension appears to capture some common types of low intrinsic dimension.

We will show, roughly, that if a data set $x_1, \ldots, x_n$ has doubling dimension d_o and if a query point q has the same distribution, then $\Phi(q, \{x_1, \ldots, x_n\})$ is small. The final bound closely resembles the one for doubling measures, except that there is also a dependence on the *aspect ratio* of the data set: the ratio of the largest interpoint distance to the smallest interpoint distance. More precisely, for finite $S \subset \mathbb{R}^d$,

$$\Delta(S) \;=\; \frac{\max_{x,y \in S} \|x-y\|}{\min_{x,y \in S, x \neq y} \|x-y\|}.$$

Our result is similar to that obtained by Clarkson [8,9] for a different data structure. We use much the same formulation and the same key lemma. Specifically, fix any points $S = \{x_1, \ldots, x_n\} \subset \mathbb{R}^d$. Pick a query q at random from this set and treat the remainder of S as the data points from which the nearest neighbor to q is sought. We bound the expected value of Φ over the n possible choices of query q.

Theorem 8 *Let $S = \{x_1, \ldots, x_n\} \subset \mathbb{R}^d$ be a set of doubling dimension $d_o \geq 2$ and aspect ratio $\Delta(S)$. Suppose q is chosen uniformly at random from S. Then for any $2^{d_0+1} \leq m < n$ and any $k \geq 1$,*

$$\mathbb{E}\Phi_{k,m}(q, S - \{q\}) \;\leq\; 6 \cdot 8^{d_o} \cdot \log \Delta(S) \cdot \left(\frac{k}{m}\right)^{1/d_o},$$

where the logarithm is base two and the expectation is over the choice of q.

Proof For any point z, any set A, and any integer $\ell \geq 1$, let $\mathrm{NN}_\ell(z, A)$ denote the ℓth nearest neighbor of z in A, breaking ties arbitrarily. For integers $\ell_1, \ell_2 \geq 1$, let

$\mathrm{NN}_{\ell_1:\ell_2}(z,A)$ be the set $\{\mathrm{NN}_\ell(z,A) : \ell_1 \le \ell \le \ell_2\}$. We will also need to work with approximate nearest neighbors. For $\gamma \ge 1$, we say $x \in A$ is an (ℓ,γ)-NN of z in A if

$$\|x-z\| \le \gamma \|z - \mathrm{NN}_\ell(z,A)\|;$$

in words, x is at most γ times further away than z's ℓth nearest neighbor.

We make use of a key fact from [9, Lemma 5.1]. For the convenience of the reader, and to resolve minor discrepancies between our notation and Clarkson's, the brief proof of this fact is reproduced in the appendix (Lemma 7).

> If S has doubling dimension d_o, then any $s \in S$ can be an (ℓ,γ)-NN nearest neighbor of at most $(8\gamma)^{d_o}\ell \log \Delta(S)$ other points of S.

We will henceforth write this as $C\ell\gamma^{d_o}$, where $C = 8^{d_o}\log\Delta(S)$. Now, for q chosen uniformly at random from S,

$$\begin{aligned}
&\mathbb{E}\Phi_{k,m}(q, S-\{q\}) \\
&= \frac{1}{n}\sum_{i=1}^n \Phi_{k,m}(x_i, S-\{x_i\}) \\
&\le \frac{1}{nm}\sum_{i,j=1}^n \frac{\|x_i - \mathrm{NN}_k(x_i, S-\{x_i\})\|}{\|x_i - x_j\|} \cdot 1(x_j \in \mathrm{NN}_{k+1:m}(x_i, S-\{x_i\})) \\
&\le \frac{1}{m}\max_{1\le j\le n}\sum_{i=1}^n \frac{\|x_i - \mathrm{NN}_k(x_i, S-\{x_i\})\|}{\|x_i - x_j\|} \cdot 1(x_j \in \mathrm{NN}_{k+1:m}(x_i, S-\{x_i\})).
\end{aligned}$$

Fix any x_j. By applying Clarkson's lemma to $(m,1)$-nearest neighbors, we see that x_j is an m-NN of at most Cm points in S, and thus the summation has at most Cm nonzero terms. By applying it to $(k,2^\ell)$-nearest neighbors, x_j is a $(k,2^\ell)$-NN of at most $2^{\ell d_o}kC$ points $x_i \in S$. For all other x_i, we have $\|x_i - \mathrm{NN}_k(x_i, S-\{x_i\})\|/\|x_i - x_j\| \le 1/2^\ell$. Thus, for any p such that $2^{pd_o}k \le m$,

$$\begin{aligned}
&\sum_{i=1}^n \frac{\|x_i - \mathrm{NN}_k(x_i, S-\{x_i\})\|}{\|x_i - x_j\|} \cdot 1(x_j \in \mathrm{NN}_{k+1:m}(x_i, S-\{x_i\})) \\
&\le 1\cdot 2^{d_o}kC + \frac{1}{2}\cdot(4^{d_o}kC - 2^{d_o}kC) + \frac{1}{4}\cdot(8^{d_o}kC - 4^{d_o}kC) + \cdots \\
&\quad + \frac{1}{2^{p-1}}\cdot(2^{pd_o}kC - 2^{(p-1)d_o}kC) + \frac{1}{2^p}\cdot(Cm - 2^{pd_o}kC) \\
&= \left(\frac{2^{d_o}}{2} + \frac{4^{d_o}}{4} + \cdots + \frac{2^{pd_o}}{2^p}\right)kC + \frac{mC}{2^p} \\
&\le \left(2k\cdot 2^{p(d_o-1)} + \frac{m}{2^p}\right)C \;\le\; \frac{3Cm}{2^p} \;\le\; 6Cm\left(\frac{k}{m}\right)^{1/d_o},
\end{aligned}$$

if we choose $p = \lfloor(\log_2(m/k))/d_o\rfloor$. The theorem now follows immediately from the earlier characterization of $\mathbb{E}\Phi_{k,m}$.

This expression has the same functional form, $(k/m)^{1/d_o}$, as that of Theorem 6 and thus leads to similar bounds on query failure probabilities to those of Theorem 7, except that they hold only in expectation over the choice of query. The additional ingredients in the argument are linearity of expectation and (for RP trees) the inequality $\mathbb{E}(\Phi\ln(1/\Phi)) \le (\mathbb{E}\Phi)\ln(1/\mathbb{E}\Phi)$.

4.3 A document model

In a bag-of-words model, a document is represented as a binary vector in $\{0,1\}^N$, where N is the size of the vocabulary and the ith coordinate is 1 if the document happens to contain the corresponding word. This is a sparse representation in which the number of nonzero positions is typically much smaller than N.

Pick any query document $q \in \{0,1\}^N$, and suppose that $x_1, \ldots, x_n$ are generated i.i.d. from a topic model μ. We will consider a simple such model with t topics, each of which follows a product distribution. The distribution μ is parametrized by the mixing weights over topics, $w_1, \ldots, w_t$, which sum to one, and the word probabilities $(p_1^{(j)}, \ldots, p_N^{(j)})$ for each topic $1 \leq j \leq t$. Here is the generative process for a document x:

- Pick a topic $1 \leq j \leq t$, where the probability of picking j is w_j.
- Set the coordinates of $x \in \{0,1\}^N$ independently; the ith coordinate is 1 with probability $p_i^{(j)}$.

The overall distribution is thus a mixture $\mu = w_1\mu_1 + \cdots + w_t\mu_t$ whose jth component is a Bernoulli product distribution $\mu_j = B(p_1^{(j)}) \times \cdots \times B(p_N^{(j)})$. Here $B(p)$ is a shorthand for the distribution on $\{0,1\}$ with expected value p. It will simplify things to assume that $0 < p_i^{(j)} < 1/2$; this is not a huge assumption if, say, stopwords have been removed.

For the purposes of bounding Φ, we are interested in the distribution of $d_H(q,X)$, where X is chosen from μ and d_H denotes Hamming distance. This is a sum of small independent quantities, and it is customary to approximate such sums by a Poisson distribution. In the current context, however, this approximation is rather poor, and we instead use counting arguments to directly bound how rapidly the distribution grows. The results stand in stark contrast to those we obtained for doubling measures, and reveal this to be a substantially more difficult setting for nearest neighbor search. For a doubling measure, the probability mass of a ball $B(q,r)$ doubles whenever r is multiplied by a constant. In our present setting, it doubles whenever r is increased by an additive constant. Specifically, it turns out (Lemma 3) that

$$\frac{\Pr(d_H(q,X) = \ell + 1)}{\Pr(d_H(q,X) = \ell)} \geq 4.$$

for $\ell \leq L/8$. Here $L = \min(L_1, \ldots, L_t)$, where $L_j = p_1^{(j)} + \cdots + p_N^{(j)}$ is the expected number of words in a document drawn from μ_j.

We start with the case of a single topic.

4.3.1 Growth rate for one topic

Let $q \in \{0,1\}^N$ be any fixed query and let X be drawn from a Bernoulli product distribution $B(p_1) \times \cdots \times B(p_N)$. Then the Hamming distance $d_H(q,X)$ is distributed as a sum of Bernoullis,

$$d_H(q,X) \sim B(a_1) + \cdots + B(a_N),$$

where

$$a_i \;=\; \begin{cases} p_i & \text{if } q_i = 0 \\ 1-p_i & \text{if } q_i = 1 \end{cases}$$

To understand this distribution, we start with a general result about sums of Bernoulli random variables. Notice that the result is exactly correct in the situation where all $p_i = 1/2$.

Lemma 2 *Suppose $Z_1, \ldots, Z_N$ are independent, where $Z_i \in \{0,1\}$ is a Bernoulli random variable with mean $0 < a_i < 1$, and $a_1 \geq a_2 \geq \cdots \geq a_N$. Let $Z = Z_1 + \cdots + Z_N$. Then for any integer $0 \leq \ell < N$,*

$$\frac{\Pr(Z=\ell+1)}{\Pr(Z=\ell)} \;\geq\; \frac{1}{\ell+1} \sum_{i=\ell+1}^{N} \frac{a_i}{1-a_i}.$$

Proof Define $r_i = a_i/(1-a_i) \in (0,\infty)$; then $r_1 \geq r_2 \geq \cdots \geq r_N$. Now, for any $\ell \geq 0$,

$$\begin{aligned}
\Pr(Z=\ell) &= \sum_{\{i_1,\ldots,i_\ell\} \subset [N]} a_{i_1} a_{i_2} \cdots a_{i_\ell} \prod_{j \notin \{i_1,\ldots,i_\ell\}} (1-a_j) \\
&= \prod_{i=1}^{N} (1-a_i) \sum_{\{i_1,\ldots,i_\ell\} \subset [N]} \frac{a_{i_1}}{1-a_{i_1}} \frac{a_{i_2}}{1-a_{i_2}} \cdots \frac{a_{i_\ell}}{1-a_{i_\ell}} \\
&= \prod_{i=1}^{N} (1-a_i) \sum_{\{i_1,\ldots,i_\ell\} \subset [N]} r_{i_1} r_{i_2} \cdots r_{i_\ell}
\end{aligned}$$

where the summations are over subsets $\{i_1,\ldots,i_\ell\}$ of ℓ distinct elements of $[N]$. In the final line, the product of the $(1-a_i)$ does not depend upon ℓ and can be ignored. Let's focus on the summation; call it S_ℓ. We would like to compare it to $S_{\ell+1}$.

$S_{\ell+1}$ is the sum of $\binom{N}{\ell+1}$ distinct terms, each the product of $\ell+1$ r_i's. These terms also appear in the quantity $S_\ell(r_1+\cdots+r_N)$; in fact, each term of $S_{\ell+1}$ appears multiple times, $\ell+1$ times to be precise. The remaining terms in $S_\ell(r_1+\cdots+r_N)$ each contain $\ell-1$ unique elements and one duplicated element. By accounting in this way, we get

$$\begin{aligned}
S_\ell(r_1+\cdots+r_N) &= (\ell+1)S_{\ell+1} + \sum_{\{i_1,\ldots,i_\ell\} \subset [N]} r_{i_1} r_{i_2} \cdots r_{i_\ell} (r_{i_1}+\cdots+r_{i_\ell}) \\
&\leq (\ell+1)S_{\ell+1} + S_\ell(r_1+\cdots+r_\ell)
\end{aligned}$$

since the r_i's are arranged in decreasing order. Hence

$$\frac{\Pr(Z=\ell+1)}{\Pr(Z=\ell)} \;=\; \frac{S_{\ell+1}}{S_\ell} \;\geq\; \frac{1}{\ell+1}(r_{\ell+1}+\cdots+r_N),$$

as claimed.

We now apply this result directly to the sum of Bernoulli variables $Z = d_H(q,X)$.

Lemma 3 *Suppose that $p_1,\ldots,p_N \in (0,1/2)$. Pick any query $q \in \{0,1\}^N$, and draw X from distribution $\mu = B(p_1)\times\cdots\times B(p_N)$. Then for any $\ell \geq 0$,*

$$\frac{\Pr(d_H(q,X)=\ell+1)}{\Pr(d_H(q,X)=\ell)} \;\geq\; \frac{L-\ell/2}{\ell+1},$$

where $L = \sum_i p_i$ is the expected number of words in X.

Proof Suppose q contains k_o nonzero entries. Without loss of generality, these are $q_1,\ldots,q_{k_o}$.

As we have seen, $d_H(q,X)$ is distributed as the Bernoulli sum $B(1-p_1)+\cdots+B(1-p_{k_o})+B(p_{k_o+1})+\cdots+B(p_N)$. Define

$$r_i \;=\; \begin{cases} (1-p_i)/p_i \text{ if } i \leq k_o \\ p_i/(1-p_i) \text{ if } i > k_o \end{cases}$$

Notice that $r_i > 1$ for $i \leq k_o$, and ≤ 1 for $i > k_o$; and that $r_i > p_i$ always.

By Lemma 2, we have that for any $\ell \geq 0$,

$$\frac{\Pr(d_H(q,X)=\ell+1)}{\Pr(d_H(q,X)=\ell)} \;\geq\; \frac{1}{\ell+1}\sum_{i>\ell} r_{(i)},$$

where $r_{(1)} \geq \cdots \geq r_{(N)}$ denotes the reordering of $r_1,\ldots,r_N$ into descending order. Since each $r_i > p_i$, and each p_i is at most $1/2$,

$$\sum_{i>\ell} r_{(i)} \;\geq\; (\text{sum of } N-\ell \text{ smallest } p_i\text{'s}) \;\geq\; (\sum_i p_i) - \ell/2 \;=\; L-\ell/2.$$

4.3.2 Growth rate for multiple topics

Now let's return to the original model, in which X is chosen from a mixture of t topics $\mu = w_1\mu_1+\cdots+w_t\mu_t$, with $\mu_j = B(p_1^{(j)})\times\cdots\times B(p_N^{(j)})$. Then for any ℓ,

$$\Pr(d_H(q,X)=\ell \mid X\sim\mu) \;=\; \sum_{j=1}^{t} w_j \Pr(d_H(q,X)=\ell \mid X\sim\mu_j).$$

Combining this relation with Lemma 3, we immediately get the following.

Corollary 3 *Suppose that all $p_i^{(j)} \in (0,1/2)$. Let $L_j = \sum_i p_i^{(j)}$ denote the expected number of words in a document from topic j, and let $L = \min(L_1,\ldots,L_t)$. Pick any query $q \in \{0,1\}^N$, and draw $X \sim \mu$. For any $\ell \geq 0$,*

$$\frac{\Pr(d_H(q,X)=\ell+1)}{\Pr(d_H(q,X)=\ell)} \;\geq\; \frac{L-\ell/2}{\ell+1}.$$

4.3.3 Bounding Φ

Fix a particular query $q \in \{0,1\}^N$, and draw $x_1, \ldots, x_n$ from distribution μ. Let the random variable S_ℓ denote the points at Hamming distance exactly ℓ from q, so that $\mathbb{E}|S_\ell| = n\Pr_{X\sim\mu}(d_H(q,X) = \ell)$. The next lemma captures some empirical consequences of Corollary 3, obtained by Chernoff bounds.

Lemma 4 *There is an absolute constant c_o for which the following holds. Pick any $0 < \delta < 1/2$ and any integer $k \geq 1$, and let v denote the smallest integer for which $\Pr_{X\sim\mu}(d_H(q,X) \leq v) \geq (8/n)\max(k, \ln 1/\delta)$. With probability at least $1-3\delta$,*

(a) $|S_0| + \cdots + |S_v| \geq 4k$.
(b) If $v \leq c_o L$ then $|S_0| + \cdots + |S_{v-1}| \leq |S_v|$.
(c) For all $v \leq \ell \leq c_o L - 1$, we have $|S_{\ell+1}|/|S_\ell| \geq 2$.

Proof Let $S_{\leq \ell}$ be a shorthand for $S_0 \cup S_1 \cup \cdots \cup S_\ell$. Then $|S_\ell|$ and $|S_{\leq\ell}|$ are each sums of n i.i.d. 0–1 random variables. Since $\mathbb{E}|S_{\leq v}| \geq 8\ln(1/\delta)$, it follows by a multiplicative Chernoff bound that

$$\Pr(|S_{\leq v}| \leq (1/2)\mathbb{E}|S_{\leq v}|) \leq \exp(-(\mathbb{E}|S_{\leq v}|)/8) \leq \delta. \tag{1}$$

This yields part (a) since $\mathbb{E}|S_{\leq v}| \geq 8k$.

For (b) and (c), pick the constant c_o so that

$$\mathbb{E}|S_{\ell+1}| \geq 16\,\mathbb{E}|S_\ell| \quad \text{for all} \quad 0 \leq \ell \leq c_o L - 1. \tag{2}$$

By Corollary 3, it is enough to take $c_o \leq 2/33$.

Assume $v \leq c_o L$. Then $\mathbb{E}|S_{<v}| \leq (1/16)\mathbb{E}|S_{\leq v}|$ and by another Chernoff bound,

$$\Pr(|S_{<v}| \geq (1/4)\mathbb{E}|S_{\leq v}|) \leq \exp(-(5/32)\mathbb{E}|S_{\leq v}|) \leq \delta. \tag{3}$$

If neither of the bad events in (1) and (3) holds, then $|S_{\leq v}| > (1/2)\mathbb{E}|S_{\leq v}| > 2|S_{<v}|$ and hence $|S_v| > |S_{<v}|$, giving statement (b).

By (2), we have $\mathbb{E}|S_v| \geq (1/2)\mathbb{E}|S_{\leq v}| \geq 4\ln(1/\delta)$, and $\mathbb{E}|S_{v+i}| \geq 16^i \cdot 4\ln(1/\delta)$ for all $0 \leq i \leq c_o L - v$. We again invoke Chernoff bounds to show that with probability at least $1-\delta$, for all $v \leq \ell \leq c_o L - 1$, we have $|S_{\ell+1}| \geq (1/2)\mathbb{E}|S_{\ell+1}|$ and $|S_\ell| \leq 4\mathbb{E}|S_\ell|$, whereupon $|S_{\ell+1}| \geq 2|S_\ell|$.

The fast growth rate of $|S_\ell|$ implies that most points lie at distance $\approx c_o L$ or greater from q. If the nearest neighbor is much closer than this, then Φ is easily bounded.

Lemma 5 *Let δ, k, v, and c_o be defined as in Lemma 4. If statements (a, b, c) of that lemma hold, then for any $m \leq n$,*

$$\Phi_{k,m}(q, \{x_1, \ldots, x_n\}) \;\leq\; 6\sqrt{\frac{v}{c_o L - \log_2(n/m) - 2}}.$$

Proof Suppose that for some $i > k$, point $x_{(i)}$ is at Hamming distance ℓ from q, that is, $x_{(i)} \in S_\ell$. Then

$$\frac{(\|q-x_{(1)}\|+\cdots+\|q-x_{(k)}\|)/k}{\|q-x_{(i)}\|} \;\leq\; \sqrt{\frac{v}{\ell}}$$

since Euclidean distance is the square root of Hamming distance. In bounding $\Phi_{k,m}$, we need to gauge the range of Hamming distances spanned by $x_{(k+1)},\ldots,x_{(m)}$.

Let $u = d_H(q,x_{(m)})$. To obtain a lower bound on u, observe that if $u \leq c_o L$, then by (b,c) of the previous lemma,

$$\begin{aligned}|S_{\leq u}| &= |S_u| + |S_{u-1}| + \cdots + |S_v| + |S_{<v}| \\ &\leq |S_u|\left(1+\frac{1}{2}+\frac{1}{4}+\cdots+\frac{1}{2^{u-v}}+\frac{1}{2^{u-v}}\right) \;=\; 2|S_u|.\end{aligned}$$

Similarly, $|S_u| \leq |S_{\lfloor c_o L\rfloor}|/2^{\lfloor c_o L\rfloor - u} \leq n/2^{c_o L-u-1}$. Stringing together the inequalities, $m \leq |S_{\leq u}| \leq 2|S_u| \leq 2n/2^{c_o L-u-1}$, whereupon $u \geq c_o L - \log_2(n/m) - 2$.

Assume $v \geq 1$ and take $u' = \min(u, \lfloor c_o L\rfloor)$. For $v \leq \ell < u'$, we have $|S_\ell|/m \leq |S_\ell|/|S_{u'-1}| \leq 1/2^{u'-\ell-1}$. Thus

$$\begin{aligned}\Phi_{k,m}(q,\{x_1,\ldots,x_n\}) &= \frac{1}{m}\sum_{i>k}\frac{(\|q-x_{(1)}\|+\cdots+\|q-x_{(k)}\|)/k}{\|q-x_{(i)}\|} \\ &\leq \frac{1}{m}\left(|S_{<v}| + \sum_{\ell=v}^{u'-1}|S_\ell|\sqrt{\frac{v}{\ell}} + (m-|S_{<u'}|)\sqrt{\frac{v}{u'}}\right) \\ &\leq \frac{|S_v|}{m} + \sum_{\ell=v}^{u'-1}\frac{|S_\ell|}{m}\sqrt{\frac{v}{\ell}} + \sqrt{\frac{v}{u'}} \\ &\leq \sqrt{\frac{v}{u'}}\left(\frac{1}{2^{u'-v-1}}\sqrt{\frac{u'}{v}} + \sum_{\ell=v}^{u'-1}\frac{1}{2^{u'-\ell-1}}\sqrt{\frac{u'}{\ell}} + 1\right) \\ &\leq \sqrt{\frac{v}{u'}}\left(1+\left(2^{1/2}+2^0+2^{-1/2}+2^{-1}+\cdots\right)\right) \;\leq\; 6\sqrt{\frac{v}{c_o L-\log_2(n/m)-2}},\end{aligned}$$

where the second-last inequality uses $u'/\ell \leq 2^{u'-\ell}$.

The implication of this lemma is that for any of the three tree data structures, the failure probability at a single level is roughly $\sqrt{v/L}$. This means that the tree can only be grown to depth $O(\sqrt{L/v})$, and thus the query time is dominated by $n_o = n \cdot 2^{-O(\sqrt{L/v})}$.

When n is large, we expect v to be small, and thus the query time improves over exhaustive search by a factor of roughly $2^{-\sqrt{L}}$.

5 Open problems

It is possible that the same data structures have substantially lower failure probability for *approximate* nearest neighbor queries. Suppose, for instance, that it suffices to find any one of the k nearest neighbors, for some small constant k, rather than the very closest neighbor. Denote by p_k the probability of failing this condition, for a specific data set and query. Is it the case that $p_k \ll p_1$ for some interesting and widely-occurring types of data, for instance $p_k \propto p_1^k$?

The trees we have described here make heavy use of randomness to avoid being tripped up by pathological data configurations like those of Section 3.3. Nevertheless it is likely possible to do much better with data structures that are more adaptive to the distribution of data and queries. One modest such adaptation is to set each cell's split direction to the principal eigenvector of the covariance matrix of data falling in that cell. Some promising experimental results for this scheme, called the *PCA spill tree*, have been obtained in [18]. A general framework for adaptivity, with discussion of generalization issues, can be found in [7].

Finally, it is increasingly common to work with non-Euclidean distances in applications of nearest neighbor search. It is therefore of interest to develop data structures that provide similar guarantees for general metric spaces or for Bregman spaces.

Acknowledgements

The authors are grateful to the National Science Foundation for support under grant IIS-1162581, and to the anonymous reviewers for their detailed feedback.

References

1. Ailon, N., Chazelle, B.: The fast Johnson-Lindenstrauss transform and approximate nearest neighbors. SIAM Journal on Computing **39**, 302–322 (2009)
2. Andoni, A., Indyk, P.: Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. Communications of the ACM **51**(1), 117–122 (2008)
3. Arya, S., Mount, D., Netanyahu, N., Silverman, R., Wu, A.: An optimal algorithm for approximate nearest neighbor searching. Journal of the ACM **45**, 891–923 (1998)
4. Assouad, P.: Plongements lipschitziens dans $\mathbb{R}^n$. Bull. Soc. Math. France **111**(4), 429–448 (1983)
5. Bentley, J.: Multidimensional binary search trees used for associative searching. Communications of the ACM **18**(9), 509–517 (1975)
6. Beygelzimer, A., Kakade, S., Langford, J.: Cover trees for nearest neighbor. In: Proceedings of the 23rd International Conference on Machine Learning (2006)
7. Cayton, L., Dasgupta, S.: A learning framework for nearest-neighbor search. In: Advances in Neural Information Processing Systems (2007)
8. Clarkson, K.: Nearest neighbor queries in metric spaces. Discrete and Computational Geometry **22**, 63–93 (1999)
9. Clarkson, K.: Nearest-neighbor searching and metric space dimensions. In: Nearest-Neighbor Methods for Learning and Vision: Theory and Practice. MIT Press (2005)
10. Dasgupta, S., Freund, Y.: Random projection trees and low dimensional manifolds. In: ACM Symposium on Theory of Computing, pp. 537–546 (2008)
11. Dasgupta, S., Sinha, K.: Randomized partition trees for exact nearest neighbor search. In: 26th Annual Conference on Learning Theory (2013)
12. Gupta, A., Krauthgamer, R., Lee, J.R.: Bounded geometries, fractals, and low-distortion embeddings. In: 44th Annual IEEE Symposium on Foundations of Computer Science, pp. 534–543 (2003)

13. Karger, D., Ruhl, M.: Finding nearest neighbors in growth-restricted metrics. In: ACM Symposium on Theory of Computing, pp. 741–750 (2002)
14. Kleinberg, J.: Two algorithms for nearest-neighbor search in high dimensions. In: 29th ACM Symposium on Theory of Computing (1997)
15. Krauthgamer, R., Lee, J.: Navigating nets: simple algorithms for proximity search. In: ACM-SIAM Symposium on Discrete Algorithms (2004)
16. Liu, T., Moore, A., Gray, A., Yang, K.: An investigation of practical approximate nearest neighbor algorithms. In: Advances in Neural Information Processing Systems (2004)
17. Maneewongvatana, S., Mount, D.: The analysis of a probabilistic approach to nearest neighbor searching. In: Seventh International Worshop on Algorithms and Data Structures, pp. 276–286 (2001)
18. McFee, B., Lanckriet, G.: Large-scale music similarity search with spatial trees. In: 12th Conference of the International Society for Music Retrieval (2011)
19. Stone, C.: Consistent nonparametric regression. Annals of Statistics **5**, 595–645 (1977)

A A summation lemma

Lemma 6 *Suppose that for some constants $A, B > 0$ and $d_o \geq 1$,*

$$F(m) \;\leq\; A\left(\frac{B}{m}\right)^{1/d_o}$$

for all integers $m \geq n_o$. Pick any $0 < \beta < 1$ and define $\ell = \log_{1/\beta}(n/n_o)$. Assume for convenience that ℓ is an integer. Then:

$$\sum_{i=0}^{\ell} F(\beta^i n) \;\leq\; \frac{Ad_o}{1-\beta}\left(\frac{B}{n_o}\right)^{1/d_o}$$

and, if $n_o \geq B(A/2)^{d_o}$,

$$\sum_{i=0}^{\ell} F(\beta^i n)\ln\frac{2e}{F(\beta^i n)} \;\leq\; \frac{Ad_o}{1-\beta}\left(\frac{B}{n_o}\right)^{1/d_o}\left(\frac{1}{1-\beta}\ln\frac{1}{\beta}+\ln\frac{2e}{A}+\frac{1}{d_o}\ln\frac{n_o}{B}\right).$$

Proof Writing the first series in reverse,

$$\begin{aligned}\sum_{i=0}^{\ell} F(\beta^i n) \;=\; \sum_{i=0}^{\ell} F\left(\frac{n_o}{\beta^i}\right) &\leq \sum_{i=0}^{\ell} A\left(\frac{B\beta^i}{n_o}\right)^{1/d_o} \\ &= A\left(\frac{B}{n_o}\right)^{1/d_o}\sum_{i=0}^{\ell}\beta^{i/d_o} \\ &\leq \frac{A}{1-\beta^{1/d_o}}\left(\frac{B}{n_o}\right)^{1/d_o} \;\leq\; \frac{Ad_o}{1-\beta}\left(\frac{B}{n_o}\right)^{1/d_o}.\end{aligned}$$

The last inequality is obtained by using

$$(1-x)^p \geq 1-px \quad \text{for } 0<x<1,\ p\geq 1$$

to get $(1-(1-\beta)/d_o)^{d_o} \geq \beta$ and thus $1-\beta^{1/d_o} \geq (1-\beta)/d_o$.

Now we move on to the second bound. The lower bound on n_o implies that $A(B/m)^{1/d_o} \leq 2$ for all $m \geq n_o$. Since $x\ln(2e/x)$ is increasing when $x \leq 2$, we have

$$\sum_{i=0}^{\ell} F(\beta^i n)\ln\frac{2e}{F(\beta^i n)} \;\leq\; \sum_{i=0}^{\ell} A\left(\frac{B}{\beta^i n}\right)^{1/d_o}\ln\frac{2e}{A(B/(\beta^i n))^{1/d_o}}.$$

The lemma now follows from algebraic manipulations that invoke the first bound as well as the inequality

$$\sum_{i=0}^{\ell} iA\left(\frac{B\beta^i}{n_o}\right)^{1/d_o} \;\leq\; \frac{Ad_o^2}{(1-\beta)^2}\left(\frac{B}{n_o}\right)^{1/d_o},$$

which in turn follows from

$$\sum_{i=0}^{\ell} i\beta^{i/d_o} \;\leq\; \sum_{i=1}^{\infty} i\beta^{i/d_o} \;=\; \sum_{i=1}^{\infty}\sum_{j=i}^{\infty} \beta^{j/d_o} \;=\; \sum_{i=1}^{\infty} \frac{\beta^{i/d_o}}{1-\beta^{1/d_o}} \;=\; \frac{\beta^{1/d_o}}{(1-\beta^{1/d_o})^2} \;\leq\; \frac{d_o^2}{(1-\beta)^2}.$$

B Clarkson's lemma

Suppose we are given a finite set of points $S \subset \mathbb{R}^d$. How many of these points can have a specific $x \in S$ as one of their ℓ nearest neighbors? Stone [19] showed that the answer is $\leq \ell\gamma_d$, where γ_d is a constant exponential in d but independent of $|S|$ and ℓ. This was a key step towards establishing the universal consistency of nearest neighbor classification in Euclidean spaces.

Clarkson [9] extended this result to metric spaces of bounded doubling dimension and to approximate nearest neighbors. Before stating his result, we introduce some notation. For any point $z \in \mathbb{R}^d$, any set $A \subset \mathbb{R}^d$, and any integer $\ell \geq 1$, let $\mathrm{NN}_\ell(z,A)$ denote the ℓth nearest neighbor of z in A, breaking ties arbitrarily. For $\gamma \geq 1$, we say $x \in A$ is an (ℓ,γ)-NN of z in A if

$$\|x-z\| \leq \gamma \|z - \mathrm{NN}_\ell(z,A)\|,$$

that is, x is at most γ times further away than z's ℓth nearest neighbor.

Recall also that we define the aspect ratio of a finite set $S \subset \mathbb{R}^d$ to be

$$\Delta(S) \;=\; \frac{\max_{x,y\in S}\|x-y\|}{\min_{x,y\in S, x\neq y}\|x-y\|}.$$

The following is shown in [9, Lemma 5.1].

Lemma 7 *Pick any integer $\ell \geq 1$ and any $\gamma \geq 1$. If a finite set $S \subset \mathbb{R}^d$ has doubling dimension d_o, then any $s \in S$ can be an (ℓ,γ)-NN nearest neighbor of at most $(8\gamma)^{d_o}\ell\log_2\Delta(S)$ other points of S.*

Proof Pick any $s \in S$ and any $r > 0$. Consider the annulus $A_r = \{x \in S : r < \|x-s\| \leq 2r\}$. By Lemma 8, A_r can be covered by $\leq (8\gamma)^{d_o}$ balls of radius $r/(2\gamma)$. Consider any such ball B: if $B \cap A_r$ contains $\geq \ell+1$ points, then each of these points has ℓ neighbors within distance r/γ, and thus does not have s as an (ℓ,γ)-NN. Therefore, there are at most $\ell(8\gamma)^{d_o}$ points in A_r that have s as an (ℓ,γ)-NN.

We finish by noticing that by the definition of aspect ratio, S can be covered by $\log_2\Delta(S)$ annuli A_r, with successively doubling radii.

Lemma 8 *Suppose $S \subset \mathbb{R}^d$ has doubling dimension d_o. Pick any $r \geq \varepsilon > 0$. If B is a ball of radius r, then $S \cap B$ can covered by $(2r/\varepsilon)^{d_o}$ balls of radius ε.*

Proof By the definition of doubling dimension, $S \cap B$ can be covered by 2^{d_o} balls of radius $r/2$, and thus 2^{2d_o} balls of radius $r/4$, and so on. More generally, $S \cap B$ can be covered by $2^{\ell d_o}$ balls of radius $r/2^\ell$ for any integer $\ell \geq 0$. Now take $\ell = \lceil \log_2(r/\varepsilon) \rceil \leq \log_2(2r/\varepsilon)$.