1
Nearest-Neighbor Classification and Search

Sanjoy Dasgupta

Samory Kpotufe

Abstract

In both algorithmic analysis of nearest neighbor search and statistical rates of convergence for nearest neighbor classification, the simplest worst-case bounds are pessimistic and discouraging, and do not accurately reflect performance in practice. In this chapter, we discuss some of the more refined types of analysis that have been attempted, and argue that much remains to be done.

1.1 Introduction

Nearest neighbor search is a basic tool of information retrieval: given a new data item (such as the medical record of a new patient, or the latest measurements from a space mission), the task is to find the *most similar* items encountered in the past. These help to place the new item in context, for instance to determine whether it is something familiar that can be handled easily or something novel that demands special attention. In particular, knowledge of the outcomes, or labels, of the nearest neighbors can be used to predict an outcome for the new instance.

Nearest neighbor search raises both algorithmic and statistical questions. How can the nearest neighbor(s) be found quickly? And what is the quality of predictions made using these neighbors? These questions have been studied for many decades, yet remain rich areas of research. A large part of the difficulty is that the simplest worst-case bounds for these problems are so loose as to be meaningless, in the sense that they provide little insight into the behavior observed in reality. Thus it is of great interest to develop methods of analysis that are more refined, that gain accuracy by taking the structure or distribution of data into account.

1.2 The Algorithmic Problem of Nearest Neighbor Search

Given a set S of n points, the nearest neighbor (NN) of a query q is the point in S that is closest to q under some distance function of interest. Finding the nearest neighbor naively takes $O(n)$ time, which can be a serious deterrent in many practical settings with large n.[1] To speed this up, can a data structure be built from S that will permit subsequent queries q to be answered quickly?

For one-dimensional data, there is an easy solution: the data structure is simply a sorted version of S, using which the nearest neighbor of any query can be found in $O(\log n)$ time by binary search. But generalizing this to higher dimension is not straightforward. An especially tricky case is when the points S and query q are all chosen uniformly at random from the surface of the unit sphere in $\mathbb{R}^D$. If $D \gg \log n$, a simple calculation shows that all the points, including the query, will with high probability lie at distance $\sqrt{2} \pm o(1)$ from each other. Thus all points are just a tiny bit further from q than its very nearest neighbor. It is hard to imagine what kind of data structure might permit the nearest neighbor to quickly be identified amid such miniscule differences. In what follows, we will refer to this example as the *canonical bad case*.

There are two ways to banish this nightmare scenario. The first is to be content with a c-approximation to the nearest neighbor, for some small constant c: that is, any point that is at most c times further away from q than its nearest neighbor. For data distributed uniformly on a high-dimensional sphere, anything in S is then an acceptable answer. The second recourse is to think of this particular example as being pathological and unlikely to occur in practice, and to make assumptions about the configuration of the data under which efficient search is possible.

1.2.1 Hashing for approximate nearest neighbor search

A hugely popular and successful method for nearest neighbor search has been locality-sensitive hashing (LSH), first introduced in the late 1990s (Indyk and Motwani, 1998; Charikar, 2002; Andoni and Indyk, 2008). This is not a specific algorithm but rather a framework for boosting the performance of simple randomized hash functions. The most common instantiation, for data in $\mathbb{R}^D$, uses random linear projections for hashing:

- A point $x \in \mathbb{R}^D$ is mapped to the integer $\lfloor (u \cdot x)/b \rfloor$, where u is a direction chosen at random from the unit sphere and b is the bucket width.
- Taking m such mappings $h_1, \ldots, h_m$, point x then gets stored in an m-dimensional table at location $(h_1(x), \ldots, h_m(x))$; the value of m can be thought of as $O(\log n)$,

[1] This ignores the time taken to compute distances between points, which is $O(d)$ in d-dimensional Euclidean space, and can be a significant factor when d is large. There is quite a bit of work on mitigating this, for instance using dimensionality reduction, but it is mostly orthogonal to our discussion here and has less of the "beyond worst case" flavor.

but would in practice typically be tuned using a set of sample queries. Regardless of m, the table can be stored in $O(n)$ space using standard hashing techniques.

- A query q is answered by looking at all points falling at location $(h_1(q), \ldots, h_m(q))$ and selecting the nearest neighbor amongst them.

The probability that this fails to return a c-approximate nearest neighbor can be bounded; and by having multiple independently-built tables, the failure probability can be made as small as desired.

For n data points in Euclidean space, LSH can be used to create a data structure of size $O(n^{1+1/c^2})$ which is then subsequently able to answer c-approximate nearest neighbor queries in time $O(n^{1/c^2})$ with failure probability that is an arbitrarily small constant (Andoni and Indyk, 2008). For $c = 2$, this translates to space $O(n^{5/4})$ and query time $O(n^{1/4})$. Numerous other variants of LSH have been developed, some that handle other distance and similarity functions (Charikar, 2002; Datar et al., 2004; Andoni et al., 2018), and some that adapt to the particular data distribution (Andoni and Razenshteyn, 2015), but the Euclidean scheme described above is a useful representative case.

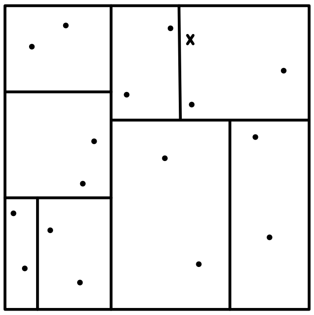
A striking feature of the analysis of LSH is that all the problem-specific characteristics that undoubtedly affect the hardness of NN search—such as the dimension of the data—are swept under the rug and a bound is given entirely in terms of the number of points n and the approximation factor c. This factor itself is somewhat hard to interpret because it means different things for different data sets. Take $c = 2$, for example (values much smaller than this lead to unreasonably large data structures): for some data sets, 2-approximation might yield points very close to the true nearest neighbor and produce usually-correct classifications, while on other data sets, 2-approximation might mean that the returned point is essentially a random draw from the data set. In short, this guarantee is not inherently reassuring.

By way of example, here is a table showing the classification error rate of c-approximate nearest neighbors, as a function of c, on the MNIST data set of handwritten digits:

c	1.0	1.2	1.4	1.6	1.8	2.0
Error rate (%)	3.1	9.0	18.4	29.3	40.7	51.4

For each value of c, the error rate shown is that of a classifier that picks a random c-approximate nearest neighbor and predicts with its label. In this case, even a small value like $c = 1.2$ leads to a substantial degradation in classification performance over the true nearest neighbor.

Having a bound that depends only on c is elegant, but the absence of other relevant parameters makes it likely to be too loose to provide guidance on specific data sets of interest. Looking back at the table for MNIST, we might be inclined to believe that we need something like a $c = 1.1$ approximation and that LSH is



```
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 $i$
  Rule($x$) = ($x_i \leq \text{median}(\{z_i : z \in S\})$)
  return (Rule)
```

Figure 1.1 The k-d tree: example and pseudocode. In the example, the split at the root of the tree is vertical, the two splits at the next level are horizontal, and the next four are a mix of horizontal and vertical. A query point q is marked by a cross.

thus a bad choice because it will require close to quadratic space. But this is far from the truth, which is that even with a much larger setting of c, the LSH scheme described above typically returns the *exact* nearest neighbor on this data set.

Locality-sensitive hashing is a beautiful algorithmic framework that is highly effective in practice. But there is scope for improvement in its analysis. It would be helpful to know the probability with which this data structure returns the exact nearest neighbor, or perhaps one of the 1% closest neighbors. This would likely depend upon the configuration of the data points, and it would be interesting to understand what structural properties of the data make for efficient search.

1.2.2 Tree structures for exact nearest neighbor search

There is an extensive literature on data structures for *exact* nearest neighbor search. Perhaps the most widely-used of these is the *k-d tree* (Bentley, 1975), a partition of $\mathbb{R}^D$ into hyper-rectangular cells, based on a given set $S \subseteq \mathbb{R}^D$ of data points. 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 1.1). The process is then recursively invoked 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 about $\log_2(n/n_o)$.

Given a k-d tree built from data points S, there are two ways to answer a nearest neighbor query q. The quick-and-dirty option 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, as is the case in Figure 1.1. Consequently, the failure probability of this scheme (taken over a random choice of queries, say) can be unacceptably high. The alternative is *comprehensive search*, which uses geometric reasoning to decide which other leaf cells might also need to be probed and always returns the true nearest neighbor, but in the worst case can take $O(n)$ time.

Popular prejudice holds that k-d tree performance—whether measured by the success probability of defeatist search or the query time of comprehensive search—deteriorates rapidly with dimension. This remains to be mathematically justified, however. What would be especially interesting is to identify simple conditions on high-dimensional data under which the k-d tree functions well.

Numerous variants of the k-d tree have been developed, attempting to compensate for its perceived weaknesses. One notable such example is the *principal component analysis (PCA) tree* (Sproull, 1991; McFee and Lanckriet, 2011), which splits data along directions of largest variance rather than along individual coordinates. Once again, the rigorous analysis of its query complexity remains an open problem, although there have been some attempts in this direction (Abdullah et al., 2014).

In the 1980s and 1990s, a variety of tree structures were introduced that guaranteed running times proportional to $\log n$ but exponential in D; a survey can be found in Clarkson (1999). Notice that this is in line with the canonical bad case described above: in $\mathbb{R}^D$, it is possible to have 2^D points that are roughly equidistant from each other, and thus query times proportional to this are not surprising, in the worst case. Interestingly, some of these data structures also work in arbitrary metric spaces. More recent incarnations have tried to move past the pessimism of these worst-case bounds by adapting to situations where the *intrinsic dimension* of the data is low, even it if its apparent dimension is a lot higher. Before delving into this work, we briefly discuss notions of dimension.

1.2.3 Notions of intrinsic dimension

Measures of intrinsic dimension have arisen in a variety of different fields (Cutler, 1993; Clarkson, 2005). The most common notions aim to either quantify the complexity of a (data) space $\mathcal{X}$, or that of a measure μ supported on $\mathcal{X}$ (usually the data generating distribution). We now look at two such quantities which appear most frequently in analyses of nearest neighbors methods.

For intuition behind the first such quantity, consider the fact that a d-dimensional hypercube of side-length r can be covered by 2^d hypercubes of side length $r/2$.

Definition 1.1 A metric space $(\mathcal{X}, \rho)$ is said to have **doubling dimension** d if, for all $r > 0$ and $x \in \mathcal{X}$, the ball $B(x, r)$ can be covered by 2^d balls of radius $r/2$.

Here are some common types of low-dimensional structure that are captured by doubling dimension; see Dasgupta and Freund (2008) for further details.

1. Any k-dimensional affine subspace $\mathcal{X} \subseteq \mathbb{R}^D$ has doubling dimension $\leq c_o k$, for some absolute constant c_o.
2. Any set $\mathcal{X} \subseteq \mathbb{R}^D$ in which each element has at most k nonzero coordinates (that is, a *sparse* set) has doubling dimension at most $c_o k + k \log D$. The same holds when $\mathcal{X}$ is of arbitrary dimension but can be sparsely represented under an unknown *dictionary* of size D, i.e., if there exist vectors $\{a_i\}_{i=1}^D$ such that any $x \in \mathcal{X}$ is a linear combination of at most k of them.
3. Let M be a k-dimensional Riemannian submanifold in $\mathbb{R}^D$ with *reach* τ (this is a measure of curvature: it means that every point at distance $< \tau$ of M has a unique nearest neighbor in M). Then every neighborhood of M of radius τ has doubling dimension $O(k)$.

It is also worth remarking that if $\mathcal{X}$, of doubling dimension d, is bounded (that is, $\sup_{x,x'} \rho(x, x') < \infty$), then for any $r > 0$, $\mathcal{X}$ can be covered by $C_d \cdot r^{-d}$ balls of radius r, for some constant C_d (Exercise 1.1). Any $(\mathcal{X}, \rho)$ with this property is said to have **metric dimension** d.

There is a similar-sounding notion, *doubling measure*, that attempts to capture the intrinsic dimension of a measure (usually a probability measure) on a metric space, by looking at how quickly the measure of a ball grows as its radius increases.

Definition 1.2 A measure μ on $(\mathcal{X}, \rho)$ is said to be **doubling** with exponent d, whenever for any x in the support of μ (henceforth denoted $\operatorname{supp}(\mu)$), and any $r > 0$, we have $\mu(B(x, r)) \leq 2^d \cdot \mu(B(x, r/2))$.

Unlike the doubling dimension, which depends only on the set $\mathcal{X}$, this varies according to the measure placed on $\mathcal{X}$. The relationship between the two notions is explored in Exercise 1.2.

Remark also that if $(\mathcal{X}, \rho)$ with doubling measure μ is bounded, then for any $r > 0$ and $x \in \operatorname{supp}(\mu)$, we have $\mu(B(x, r)) \geq C_d r^d$ for some constant C_d (Exercise 1.1). We then say that μ is **homogeneous** (on $\operatorname{supp}(\mu)$) with parameters (C_d, d).

1.2.4 Adaptivity to intrinsic dimension in nearest neighbor search

One way of going beyond the pessimism of worst-case analysis is to identify families of instances that occur in practice and are also "easier" in the sense of admitting better bounds. For nearest neighbor search, this enterprise has mostly focused on analyzing data sets of low intrinsic dimension. The hope is that the exponential dependence on dimension in worst-case bounds for exact nearest neighbor search can be replaced by a similar dependence on intrinsic dimension, which might be much smaller.

The excellent survey of Clarkson (2005) describes ways in which nearest neighbor data structures can be made adaptive to different types of intrinsic dimension. Perhaps the easiest assumption to work with is a finite-sample version of doubling

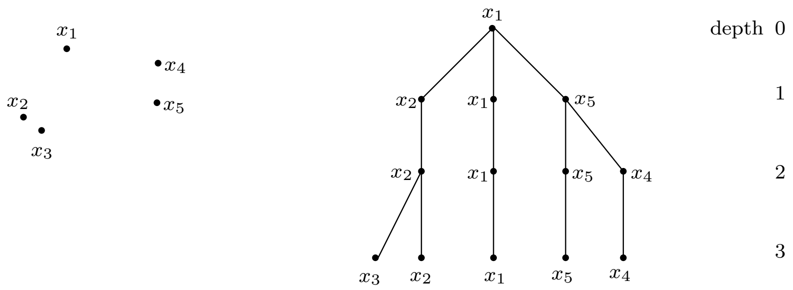


Figure 1.2 A cover tree for a data set of five points. From the structure of the tree we can conclude, for instance, that x_1, x_2, x_5 are all at distance $\geq 1/2$ from each other, since they are all at depth 1, and that the distance between x_2 and x_3 is $\leq 1/4$.

measure, which we now introduce. Suppose the data lie in a metric space $\mathcal{X}$. We say that a subset $T \subseteq \mathcal{X}$ has *expansion constant* c if for any point $p \in \mathcal{X}$ and any radius $r > 0$, we have $|T \cap B(p, 2r)| \leq c|T \cap B(p, r)|$. The assumption on the data set S is that there exists a small c such that $S \cup \{q\}$ has expansion constant at most c for any query point q. The intrinsic dimension can then be viewed as $\log c$.

One widely-used data structure that has been analyzed under this condition is the *cover tree* (Beygelzimer et al., 2006), which can be used for exact nearest neighbor search in any metric space. It works by maintaining a hierarchical covering of the data set, which we will now describe in more detail. Say the data points are $x_1, \ldots, x_n$, and assume for simplicity that all interpoint distances are ≤ 1. Then any point x_i serves as a 1-cover of the entire set; take it to be the root of the tree. The next level will consist of a subset of the x_i's that constitute a $(1/2)$-cover, and the following level will be a $(1/4)$-cover, and so on. Given level $j - 1$, level j can be built as follows: take all the points from level $j - 1$, and repeatedly add in a data point that is not within distance $1/2^j$ of those already chosen. The resulting cover tree on data points $x_1, \ldots, x_n$ is a rooted infinite tree with the following properties.

- Each node of the tree is associated with one of the data points x_i.
- If a node is associated with x_i, then one of its children is also associated with x_i.
- All nodes at depth j are at distance at least $1/2^j$ from each other.
- Each node at depth $j + 1$ is within distance $1/2^j$ of its parent (at depth j).

See Figure 1.2 for an example. In practice, there is no need to duplicate a node as its own child, so the tree takes up $O(n)$ space. Moreover, it is not hard to build the tree on-line, adding one point at a time.

When a query q needs to be answered, it is moved down the tree, one level at a time. At the jth level (call it L_j), geometric reasoning is used to identify a subset of nodes $S_j \subseteq L_j$ whose descendants could possibly include the nearest neighbor of

q; this is based on the distance from q to the closest point in L_j, combined with the triangle inequality. At the next level, L_{j+1}, only the children of S_j are examined and these are further restricted to a subset S_{j+1}, and so on. It turns out that with expansion constant c, only $|S_j| = O(\text{poly}(c))$ nodes need to be considered at each level, and the total time to find the exact nearest neighbor is $O(\text{poly}(c) \log n)$.

The cover tree is a popular and effective data structure, especially for non-Euclidean distance metrics. Its analysis, however, is marred by the brittleness of the expansion constant assumption. To get a sense of this, observe that even data in $\mathbb{R}^D$, under Euclidean distance, can have arbitrarily high c, unbounded by any function of D. It is thus of interest to devise more reasonable conditions under which to study this scheme.

A weaker and more realistic assumption on a data set is that it has low doubling dimension d. In this case, there are data structures of size $O(n)$ that either yield a $(1+\epsilon)$-approximate NN in time $O(2^{O(d)} \log n + (1/\epsilon)^{O(d)})$ (Krauthgamer and Lee, 2004), or, when the query distribution matches the data distribution, yield the correct NN in time $O(2^d \log n)$ (Clarkson, 1999, 2005). We will discuss another such data structure in more detail in the next section.

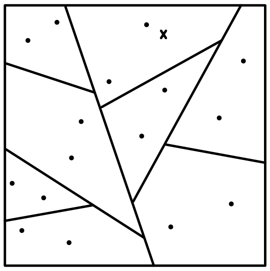
Adaptivity to doubling dimension is nontrivial: k-d trees, for instance, do not have this property (Dasgupta and Sinha, 2015). This makes it technically interesting and has led to quite a bit of computational geometry work around this notion. However, it is really just one specialized way of moving beyond the worst case in nearest neighbor search. The field of unsupervised learning has identified many varieties of geometric structure that commonly exist in data. A few of these, like manifold structure, are captured by intrinsic dimension; but many others, like cluster structure, are not. Thus it would be useful to move beyond intrinsic dimension when positing structural "niceness" assumptions under which nearest neighbor search can efficiently be performed.

An alternative to bounding the query time of a nearest neighbor data structure in terms of pre-specified geometric parameters like intrinsic dimension is to explicitly characterize the types of data on which it is efficient. Ideally, one would be able to achieve tight, instance-specific results in this way. We now turn to such a scheme.

1.2.5 A randomized tree structure with instance-specific bounds

Locality-sensitive hashing has brought a simple and highly effective paradigm to the field of nearest neighbor search: design a data structure that is quick-and-dirty and has nonzero probability of success on any instance; and then boost the success probability by making multiple copies. We now discuss a way of bringing much the same sensibility to k-d trees.

The random projection (RP) tree (Figure 1.3) injects two forms of randomness into a k-d tree: (1) instead of splitting cells along coordinate axes, it picks split directions uniformly at random from the unit sphere, and (2) instead of putting



```
function ChooseRule($S$)
  Pick $U$ uniformly at random from the unit sphere
  Pick $\beta$ 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)
```

Figure 1.3 The random projection tree (RP tree): example and pseudocode. Again, a sample query point is marked with a cross.

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 idea is to answer nearest neighbor queries using defeatist search on this randomized tree structure, which takes time $O((\log(n/n_o)) + n_o)$, where n_o is an upper bound on the number of data points in any leaf. For 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, over the randomness in the data structure, can be bounded using an elementary argument (Dasgupta and Sinha, 2015). The bound turns out to be proportional to a simple function of the point configuration,

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

where $x_{(1)}, x_{(2)}, \ldots$ denotes an ordering of the x_i by increasing distance from q.

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. This is what we get in the canonical bad case discussed at the beginning of Section 1.2. 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 an instance of nearest neighbor search.

It is not hard to give upper bounds on Φ in situations where the data has low doubling measure or doubling dimension. This leads to the following results:

- When $x_1, \ldots, x_n$ are drawn i.i.d. from a doubling measure with exponent d, the RP tree is able to answer arbitrary exact nearest neighbor queries in time $O(d)^d + O(\log n)$, with a probability of error that is an arbitrarily small constant.
- 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, then a similar result holds, but with an additional dependence on the aspect ratio of the data.

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 reduced by building multiple trees to get an *RP forest*.

Although RP forests have been found effective in practice (Hyvonen et al., 2016), one would hope to do better by having trees that are still randomized—so that error probability can be reduced by building a forest—but are more attuned to the data, in much the same way that a single PCA tree is (in practice) superior to a single RP tree. It is an interesting open problem to find a way of doing this that both works well empirically and admits a clean analysis.

1.2.6 Wrap-up: analyzing nearest neighbor search algorithms

Nearest neighbor search has been the subject of algorithmic research since the 1970s, and many data structures have been developed for it. Some of these, such as locality-sensitive hash tables, k-d trees, and cover trees, are fairly easy to implement and seem to be effective in practice. But in order to understand their relative strengths and weaknesses—to gauge, for instance, which might be preferable for a given type of data—and to develop better algorithms, it is important to have ways of analyzing these schemes. The current state-of-the-art is lacking in this regard.

For some data structures, such as the k-d tree, there is no characterization of the types of data on which it works well. On others, there is analysis that is beautiful but fails to give insight into when and why the scheme works; examples include the bounds for LSH, which are given solely in terms of an approximation factor and are thus very loose, and those for the cover tree, which are based upon a dimensionality assumption that is brittle to the point of straining plausibility.

One good open problem is to identify other structural assumptions on data—beyond low doubling dimension—that are likely to hold in many situations and that make nearest neighbor search efficient. A second is to pick any existing practical nearest neighbor algorithm, and to rigorously formulate conditions on the data under which it will work well.

1.3 Statistical Complexity of k-Nearest Neighbor Classification

We now turn to a different aspect of nearest neighbor: its statistical performance when used as a classification strategy. While statistical and computational questions are fundamentally different, and in fact are studied in different communities—machine learning and statistics on one hand, and algorithms on the other—we will see that some of the same ideas were developed to capture notions of favorable *structure* in data with similar upsides and downsides as discussed above.

Nearest neighbor classification is a form of *nonparametric estimation*: that is, it is a prediction strategy whose complexity (e.g., size) is potentially unbounded, and it is capable of modeling any decision boundary. The statistics community has developed a standard framework for analyzing nonparametric estimators, and has obtained basic bounds that provide some insights into general behavior.

1.3.1 The statistical learning framework

Let $\mathcal{X}$ be the space in which data lie, and $\mathcal{Y}$ the space of labels. We will assume for simplicity that $\mathcal{Y} = \{0, 1\}$. The standard model of statistical learning is that there is some (unknown) underlying distribution $P_{X,Y}$ on $\mathcal{X} \times \mathcal{Y}$ from which all data—past, present, and future—is drawn i.i.d. The training data $\{X_i, Y_i\}_1^n \overset{\text{i.i.d}}{\sim} P_{X,Y}$ is useful precisely because it provides some information about $P_{X,Y}$, and any model we build is evaluated according to its performance on $P_{X,Y}$.

A *classifier* is any function $h : \mathcal{X} \to \mathcal{Y}$. It can be evaluated by the *01-risk*

$$R(h) = P_{X,Y}\left(h(X) \neq Y\right).$$

There need not exist any classifier with zero risk: consider any scenario with inherent uncertainty, such as a medical prediction problem in which x is a patient's medical record and y is whether the person will suffer a stroke in the next year. Formally, this corresponds to cases in which the conditional distribution of Y given $X = x$, denoted $P_{Y|x}$, assigns non-zero probability to both outcomes, 0 and 1.

Let $\eta(x) = P_{Y|x}(1) = \mathbb{E}[Y|x]$; the 01-risk is minimized by the so-called *Bayes classifier* which predicts the most likely label at each point x:

$$h^*(x) \doteq \arg\max \left\{P_{Y|x}(1), P_{Y|x}(0)\right\} = \mathbb{1}\left\{\eta(x) \geq 1/2\right\}.$$

We will henceforth evaluate any classifier h by how much its risk exceeds that of h^*, the so-called *excess-risk*,

$$\mathcal{E}(h) \doteq R(h) - R(h^*), \text{ depending on } P_{X,Y}.$$

Now consider any learning procedure that takes n data points sampled i.i.d. from $P_{X,Y}$ and produces a classifier $\hat{h}_n$. The most basic condition we could demand of this procedure is *consistency*: that as n grows to ∞, the excess risk $\mathcal{E}(\hat{h}_n)$ goes to zero. With this assured, the next order of business is to establish the rate of convergence of the excess risk as a function of n and other problem parameters.

Because decision boundaries can be arbitrarily complex, it is well-known that in nonparametric estimation there are no universal rates of convergence without conditions on the data distribution (Devroye et al., 1997). But what are reasonable assumptions to make on $P_{X,Y}$? Over the past few decades, a certain set of assumptions has become entrenched in the statistics literature, perhaps more for mathematical convenience than anything else, and has become the standard backdrop for convergence rates. We will talk about these, about the resulting bounds and the estimators that achieve them, and about whether this theory provides an adequate picture of when nearest neighbor classification works well.

1.3.2 Minimax optimality

We are interested in the limits of performance, assessed in terms of excess risk $\mathcal{E}(\hat{h})$ (as a function of sample size n) achievable by any procedure[2] $\hat{h}$ having little to no information on the Bayes classifier h^*, i.e., little information on $P_{X,Y}$. Assuming $P_{X,Y}$ belongs to some class $\mathcal{P}$, encoding information on h^*, performance limits are captured by the *minimax* classification risk over $\mathcal{P}$:

$$\mathcal{E}^*(\mathcal{P}) \doteq \inf_{\hat{h}} \sup_{P_{X,Y} \in \mathcal{P}} \mathbb{E}_{P^n_{X,Y}} \mathcal{E}(\hat{h}).$$

The sup denotes the worst-case excess risk over $\mathcal{P}$ achievable by any given $\hat{h}$. Any classifier $\hat{h}$ achieving excess risk $O(\mathcal{E}^*)$ for all $P_{X,Y} \in \mathcal{P}$ is called *minimax-optimal* for $\mathcal{P}$. As a classical example, $\mathcal{P} \doteq \{P_{X,Y}\}$ corresponds to assuming $\mathcal{X} \subset \mathbb{R}^D$, while $\eta(x)$ is λ-Lipschitz over $\mathcal{X}$, i.e., $|\eta(x) - \eta(x')| \leq \lambda \|x - x'\|$, for some $\lambda > 0$ – encoding the hope that nearby points in $\mathcal{X}$ have similar Y values. Under these assumptions, $\mathcal{E}^*(\mathcal{P})$ is known to be of order $n^{-1/(2+D)}$; such rate is achieved for instance by k-NN classification with a suitable choice of $k \propto n^{2/(2+D)}$. Unfortunately, this is a rather slow rate whenever D is large, since a number of samples $n = \Omega\left(\epsilon^{-(2+D)}\right)$ seems required to achieve excess risk $0 < \epsilon < 1$, a *curse of dimensionality.* While this rate is unavoidable in the worst-case over $\mathcal{P}$, one would hope that there are more favorable distributions $P_{X,Y}$ in $\mathcal{P}$ where procedures such as k-NN would do much better. This is indeed the case and is the focus of the rest of this section.

1.3.3 Adaptive Rates versus Worst-case Rates

As in the above discussion, let $\mathcal{P}$ denote the class of all distributions $P_{X,Y}$, with marginal P_X supported on $\mathcal{X} \subset \mathbb{R}^D$ ($\mathcal{X}$ perhaps unknown), with λ-Lipschitz *regression function* $\eta(x)$. For simplicity, in the following discussion, we will let $\mathcal{X}$ be bounded; hence, w.l.o.g., let $\sup_{x,x' \in \mathcal{X}} \|x - x'\| = 1$.

Now note that, $\mathcal{P}$ contains – among other favorable distributions – subclasses $\mathcal{P}_d$ of those distributions $P_{X,Y}$ such that $\mathcal{X} \doteq \text{supp}(P_X)$ (or P_X itself) is of lower intrinsic dimension $d \ll D$, where *intrinsic dimension* is formalized as any of the concepts defined in Section 1.2.3. If we knew a priori that $P_{X,Y} \in \mathcal{P}_d \subset \mathcal{P}$, we could do much better than the minimax rate $\mathcal{E}^*(\mathcal{P}) \propto n^{-1/(2+D)}$: this is immediate to see when d stands for Euclidean dimension, i.e., $\mathcal{X}$ is an affine subspace of dimension d, since as per the above discussion, we would then have $\mathcal{E}^*(\mathcal{P}_d) \propto$

[2] We often will not distinguish between the classification procedure $\hat{h}$, which maps data in $\mathcal{X}^n$ to a classifier $\mathcal{X} \to \mathcal{Y}$, and the classifier that it returns. In other words, $\mathcal{E}(\hat{h})$ is the excess risk of the classifier returned by the procedure $\hat{h}$.

$n^{-1/(2+d)} \ll n^{-1/(2+D)}$, which is achieved, e.g., by k-NN with $k \propto n^{2/(2+d)}$. The question is therefore whether such a better rate is achievable (i) under general notions of intrinsic dimension d where $\mathcal{X}$ is nonlinear (e.g., a manifold of dimension d that occupies much of $\mathbb{R}^D$), and (ii) without the knowledge that $P_{X,Y} \in \mathcal{P}_d$. A classification procedure which (nearly) achieves the rates $\mathcal{E}^*(\mathcal{P}_d)$ simultaneously for all $\mathcal{P}_d \subset \mathcal{P}$ (i.e. under (ii) above) is called *minimax adaptive* over the collection $\{\mathcal{P}_d\}_{d \le D}$, or colloquially, *adaptive to intrinsic d.*

In the sequel, we will show that this is indeed the case for k-NN for any of the notions of intrinsic dimension d of Section 1.2.3, as it happens that key quantities controlling performance – namely, typical distances to nearest neighbors – depend only on d rather than on the ambient dimension D. To develop this theme, let us first assume that P_X is *homogeneous* on $\mathcal{X}$ with parameters (C_d, d), i.e., balls of radius r have P_X-mass at least $C_d r^d$ (Section 1.2.3).

In this case we have the following theorem. Throughout we assume that the k-NN estimate at any x is defined on exactly $k \le n$ points, i.e., either there are no ties in distances to x, or a deterministic rule is employed to break ties (e.g. pick the first k ordered indices); we let $k\mathrm{NN}(x)$ be the retained set of k closest neighbors to x.

With this notation, k-NN classification is given by $\hat{h} \doteq \mathbb{1}\{\hat{\eta} \ge 1/2\}$, where

$$\hat{\eta}(x) \doteq \frac{1}{k} \sum_{X_i \in k\mathrm{NN}(x)} Y_i. \tag{1.1}$$

Theorem 1.3 *Let P_X be (C_d, d) homogeneous on bounded support $\mathcal{X} \subset \mathbb{R}^D$, and let $\eta(x)$ be λ-Lipschitz. Let $\hat{h}$ denote a k-NN estimate with $k \propto n^{2/(2+d)}$. We have*

$$\mathbb{E}\,\mathcal{E}(\hat{h}) \le C\left(\frac{1}{\sqrt{k}} + \left(\frac{k}{n}\right)^{1/d}\right) \le C' n^{-1/(2+d)},$$

where the expectation is over the random draw of $\{X_i, Y_i\}_{i=1}^n$, and C, C' depend on C_d, d and λ, but not on D.

Without further distributional assumption, the rate is tight as it matches the minimax rate for distributions on $\mathbb{R}^d$. The result is obtained by a reduction from classification to regression, where we recall the fact that the Bayes classifier is given by $h^* = \mathbb{1}\{\eta \ge 1/2\}$, $\eta(x) \doteq \mathbb{E}[Y|x]$. Hence, k-NN performance can be assessed through how well $\hat{\eta}$ estimates the *regression function* η. Let $\|\hat{\eta} - \eta\|_1 \doteq \mathbb{E}\,|\hat{\eta} - \eta|$:

Proposition 1.4 (Regression to Classification) $\mathcal{E}(\hat{h}) \le 2\|\hat{\eta} - \eta\|_1$.

Proof Let $\mathcal{X}_{\neq} \doteq \left\{x \in \mathcal{X} : \hat{h}(x) \neq h(x)\right\}$, and notice that

$$\mathcal{E}(\hat{h}) = \int_{\mathcal{X}_{\neq}} |P_{Y|x}(1) - P_{Y|x}(0)|\, \mathrm{d}P_X = \int_{\mathcal{X}_{\neq}} |2\eta(x) - 1|\, \mathrm{d}P_X,$$

while, whenever $\hat{h} \neq h$, we necessarily have $|\hat{\eta} - \eta| \ge |\eta - 1/2|$. □

Now we aim to bound $\mathbb{E}\,\|\hat{\eta}-\eta\|_1$, one approach being to bound $\|\hat{\eta}-\eta\|_1$ by $\|\hat{\eta}-\eta\|_2 \doteq \left(\mathbb{E}_X|\hat{\eta}(X)-\eta(X)|^2\right)^{1/2}$. We will first condition on $\mathbf{X} \doteq \{X_i\}_{i=1}^n$ while considering just the randomness in $\mathbf{Y} \doteq \{Y_i\}_{i=1}^n$. Let $\tilde{\eta}(x)$ denote the conditional expectation $\mathbb{E}_{\mathbf{Y}|\mathbf{X}}\hat{\eta} = \frac{1}{k}\sum_{X_i \in k\mathrm{NN}(x)} \eta(X_i)$. Clearly, $\tilde{\eta}$ relates most directly to η. Using the fact that, for any random variable Z, $\mathbb{E}[Z-c]^2 = \mathbb{E}(Z-\mathbb{E}Z)^2 + (\mathbb{E}Z-c)^2$, we have the following *bias-variance* decomposition:

$$\mathbb{E}_{\mathbf{Y}|\mathbf{X}}|\hat{\eta}(x)-\eta(x)|^2 = \underbrace{\mathbb{E}_{\mathbf{Y}|\mathbf{X}}|\hat{\eta}(x)-\tilde{\eta}(x)|^2}_{\text{Variance}} + \underbrace{|\tilde{\eta}(x)-\eta(x)|^2}_{\text{Squared Bias}}. \tag{1.2}$$

Variance Bound. Using the independence of Y_i values upon conditinoning, we have:

$$\mathbb{E}_{\mathbf{Y}|\mathbf{X}}|\hat{\eta}(x)-\tilde{\eta}(x)|^2 = \frac{1}{k^2}\sum_{X_i \in k\mathrm{NN}(x)} \mathrm{Var}(Y_i) \le \frac{1}{k}. \tag{1.3}$$

Bias Bound. Given the Lipchitz assumption on η, we have that

$$|\tilde{\eta}(x)-\eta(x)| \le \frac{1}{k}\sum_{X_i \in k\mathrm{NN}(x)} |\eta(X_i)-\eta(x)| \le \max_{X_i \in k\mathrm{NN}(x)} \lambda\|X_i - x\|.$$

Nearest Neighbor Distances. Let $r_k(x) \doteq \max_{X_i \in k\mathrm{NN}(x)} \|X_i - x\|$ denote the distance from x to its k-th closest neighbor in $\mathbf{X}$. As it turns out, typical values of r_k depend on d rather than on the ambient dimension D. For intuition, notice that the ball $B(x, r_k(x))$ will likely have mass at most $c \cdot \frac{k}{n}$ (since it has empirical mass at least $\frac{k}{n}$); we will therefore have the inequality $c \cdot \frac{k}{n} \ge P_X(B(x, r_k(x))) \ge C_d r_k^d(x)$ (following from P_X being (C_d, d) homogeneous), implying that $r_k(x) \le C_d' \left(\frac{k}{n}\right)^{1/d}$. This is formalized as follows.

Let $r_k^*(x) = \inf\left\{1 \ge r > 0 : P_X(B(x,r)) \ge 2\frac{k}{n}\right\}$. First notice that we must have $P_X(B(x, r_k^*(x))) \ge 2\frac{k}{n}$ (by continuity of P_X over monotone sequences of events). Also, since $P_X(B(x, \frac{1}{2}r_k^*(x))) < 2\frac{k}{n}$, we must have that $r_k^*(x) \le C_d' \left(\frac{k}{n}\right)^{1/d}$. Now, we just need to argue that $r_k(x) \le r_k^*(x)$ with high probability, in other words, that the ball $B(x, r_k^*(x))$ contains at least k points; this is certainly the case since empirical masses of balls concentrate around their expectation. Namely, let $P_{X,n}$ denote the empirical distribution induced by $\mathbf{X}$, by a multiplicative Chernoff bound:

$$\begin{aligned}\mathbb{P}\left(P_{X,n}(B(x, r_k^*(x))) < \frac{k}{n} \le \frac{1}{2}P_X(B(x, r_k^*(x)))\right) &\le \exp\left\{-\frac{1}{8}n \cdot P_X(B(x, r_k^*(x)))\right\} \\ &\le \exp\left\{-\frac{k}{4}\right\} \le \frac{4}{k}.\end{aligned}$$

It follows that

$$\mathbb{E}_{\mathbf{X}}\left[r_k^2(x)\right] \le r_k^{*2}(x) + \mathbb{P}\left(r_k(x) > r_k^*(x)\right) \le {C_d'}^2\left(\frac{k}{n}\right)^{2/d} + \frac{4}{k}. \tag{1.4}$$

Combining (1.4) and (1.3) by invoking the bias-variance decomposition in (1.2), and then taking expectation over $\mathbf{X}$ yields the result of Theorem 1.3. $\square$

Thus, k-NN classification achieves an excess risk that depends only on $d \ll D$ (for even nonlinear support $\mathcal{X}$) provided k is set according to d. A loose-end is therefore whether the parameter k can be set, optimally, without knowledge of d.

Data-driven Choice of k. The simplest approach is cross-validation, i.e., splitting the sample into 2 (nearly) equal size independent subsamples, where one subsample is used to define classifiers – corresponding to choices of k – and the other is used to test their performance. W.l.o.g., assume both samples are of size n; define $\hat{h}_k$ as a classifier on subsample $\{X_i, Y_i\}_{i=1}^n$ using the parameter choice $k \in [n]$ (admitting a choice of $k \propto n^{2/(2+d)}$, for unknown d). Now, define the empirical risk $R'_n(h_k) \doteq \frac{1}{n}\sum_i \mathbb{1}\{h(X'_i) \neq Y'_i\}$ on validation sample $\{X'_i, Y'_i\}_{i=1}^n$, and the choice $\hat{k} \doteq \arg\min_{k\in[n]} R'_n(\hat{h}_k)$. Let $k^* \doteq \arg\min_{k\in[n]} R(\hat{h}_k)$; notice that

$$R(\hat{h}_{\hat{k}}) \leq R(\hat{h}_{k^*}) + 2\max_{k\in[n]} |R(h_k) - R'_n(h_k)|.$$

Combining Chernoff and union bounds, we have that, with probability at least $1-\delta$:

$$\max_{k\in[n]} |R(h_k) - R'_n(h_k)| \leq \sqrt{\frac{\log(2n/\delta)}{2n}}, \text{ of lower order than } n^{-1/(2+d)}.$$

In other words, picking $\delta = 1/n$, we have with probability at least $1 - 1/n$ that $\mathcal{E}(\hat{h}_{\hat{k}}) \leq \mathcal{E}(\hat{h}_{k^*}) + 2\sqrt{\frac{2\log(2n)}{2n}}$. Now, use the fact that $\mathbb{E}\,\mathcal{E}(\hat{h}_{k^*}) \leq \min_{k\in[n]} \mathbb{E}\,\mathcal{E}(\hat{h}_k)$:

Corollary 1.5 *Under the assumptions of Theorem 1.3, the empirical $\hat{k}$ satisfies*

$$\mathbb{E}\,\mathcal{E}(\hat{h}_{\hat{k}}) \leq C' n^{-1/(2+d)}.$$

Similar arguments as the above extend to more general settings, overviewed next.

General Metrics and Notions of Intrinsic Dimension. First, notice that the above arguments extend directly to any metric space $(\mathcal{X}, \rho)$ admitting a (C_d, d)-homogenous measure P_X. Also, we could have assumed P_X to be doubling since it is then homogeneous (Section 1.2.3). Suppose instead we only assumed that $(\mathcal{X}, \rho)$ has metric dimension d (allowing spaces of doubling dimension d). The adaptive rate of $n^{-1/(2+d)}$ still holds. However, such a result requires a more refined analysis of k-NN distances r_k: while at any given point x, $r_k(x)$ might not scale with d, it can be shown that $\mathbb{E}\, r_k(X)$ is of the order $(k/n)^{1/d}$ (by adapting a covering argument of Györfi et al. (2006) to metric $\mathcal{X}$), which is sufficient.

Smoothness Conditions on η. The above arguments extend easily to the case where η is Hölder continuous, i.e., $|\eta(x) - \eta(x')| \leq \lambda\rho^\alpha(x, x')$ for some $0 < \alpha \leq 1, \lambda > 0$; we would instead obtain the minimax rate $n^{-1/(2+d/\alpha)}$, attained by setting $k \propto$

$n^{2/(2+d/\alpha)}$ (or using $\hat{k}$ as defined above). This is obtained by bounding the bias by $\lambda r_k^\alpha(x)$. Notice that the rate $n^{-1/(2+d/\alpha)}$ worsens as $\alpha \to 0$, attesting to the fact that classification is hardest when η changes too fast over $\mathcal{X}$.

While Hölder or Lipschitz conditions on η capture the desired condition that Y should not change too fast over $\mathcal{X}$, they do not allow discontinuities in η, which goes against practical intuition in classification. One way to address this is to instead assume that η is piecewise Hölder, or likely to be *locally* Hölder over $\mathcal{X}$, appropriately formalized (see e.g. (Urner et al., 2011; Willett et al., 2006)). More recently Chaudhuri and Dasgupta (2014) formalized the intuition that, all that is needed for k-NN success – irrespective of continuity of η – is that the average Y value in a neighborhood of any x be close to $\eta(x)$ (the average Y at x), especially as the P_X-mass of the neighborhood gets small. This intuition is parametrized as follows: for any set $B \subset \mathcal{X}$, let $\eta(B) = \mathbb{E}[\eta \mid B]$, then it is assumed that

$$\forall x \in \mathcal{X}, r > 0, \quad |\eta(B(x,r)) - \eta(x)| \leq C_\gamma \cdot P_X(B(x,r))^\gamma, \text{ for some } C_\gamma, \gamma > 0.$$

Intuitively, letting $r = r_k(x)$, we would have $\tilde{\eta}(x) \approx \eta(B(x,r))$, while $P_X(B(x,r)) \approx k/n$, that is, the bias $|\tilde{\eta}(x) - \eta(x)|$ would be of order $(k/n)^\gamma$; this together with a variance of order $1/k$ yields an excess risk of order $n^{-1/(2+\gamma)}$ by optimizing over k.

In particular, under our earlier Hölder conditions, it can be shown that $\gamma = \alpha/d$ holds. This more general condition therefore yields a similar bias bound of order $(k/n)^{\alpha/d}$, and recovers the above minimax rates.

Inhomogeneous data, and extensions of k-NN. The above distributional conditions, while classical, do not account for spatial variations in $P_{X,Y}$. For instance the density of P_X (e.g., with respect to Lebesgue on $\mathcal{X} = \mathbb{R}^d$) might vary significantly over space; $\mathcal{X}$ might be made up of subregions $\mathcal{X}_i$ of varying intrinsic dimension $d_i \ll D$, and varying complexity in $P_{Y|X}$ (e.g. η might satisfy different Hölder conditions across $\mathcal{X}_i$'s). The support $\mathcal{X}$ might be unbounded, allowing for far outliers. While these situations might be common in practice, they have only now started receiving theoretical attention. In particular, they are commonly handled by extensions of k-NN such as *local* k-NN, where a local choice of $k = k(x)$ is made at every $x \in \mathcal{X}$.While these procedures in essence have *an infinite number of hyperparameters*, e.g. $\{k(x) : x \in \mathcal{X}\}$, they can be shown to *generalize*, i.e., attain nearly minimax rates of convergence, even under data-driven choices of $k(x)$ (see e.g., Kpotufe (2011); Samworth *et al.* (2012); Gadat et al. (2016) for general treatments which extend to weighted versions of k-NN prediction, under relaxations of traditional assumptions on the marginal P_X).

1.3.4 Low Noise Conditions and Fast Rates

Another favorable situation in classification is one where Y labels are deterministic (or nearly so). In particular, suppose $\eta(x)$ has margin away from $1/2$ at some point

x, i.e., $|\eta(x) - 1/2| > \tau$ for some $0 < \tau < 1/2$. Recall that the Bayes classifier satisfies $h^*(x) = \mathbb{1}\{\eta(x) \geq 1/2\}$, while the k-NN estimate $\hat{h}(x) = \mathbb{1}\{\hat{\eta}(x) \geq 1/2\}$ where $\hat{\eta}$ estimates η. Thus, if $|\hat{\eta}(x) - \eta(x)| \leq \tau$, we must have $\hat{h}(x) = h^*(x)$, i.e., the excess risk at x is then 0.

Under the conditions of Theorem 1.3 above, for $k \propto n^{2/(2+d)}$, and n sufficiently large, we will have $|\hat{\eta}(x) - \eta(x)| \leq Cn^{-1/(2+d)} \leq \tau$ with high probability: this follows from $|\hat{\eta}(x) - \eta(x)| \leq |\hat{\eta}(x) - \tilde{\eta}(x)| + |\tilde{\eta}(x) - \eta(x)|$, and bounding the *variance* and *bias* terms in high probability (by order of $(1/k)$ and $(k/n)^{1/d}$), rather than in expectation. As it turns out, such result holds uniformly over $x \in \mathcal{X}$: let $0 < \delta < 1$,

$$\mathbb{P}\left(\sup_x |\hat{\eta}(x) - \eta(x)| \leq C\left(\frac{\log(n/\delta)}{n}\right)^{1/(2+d)}\right) \geq 1 - \delta. \tag{1.5}$$

One way to obtain the above is to use uniform Vapnik-Chervonenkis (VC) concentration arguments over the class of balls centered at $x \in \mathcal{X}$; the constant C above now also depends on the VC dimension of this class (see e.g. Kpotufe (2011)).

Now assume the so-called *Massart's noise condition* that $\forall x \in \mathcal{X}$, $|\eta(x) - 1/2| > \tau$. It then follows from (1.5) that, if n is greater than some $n_0(\tau)$, there is high probability that $\mathcal{E}(\hat{h}) = 0$, which is remarkable. This corresponds to an exponentially fast rate in expectation, i.e., $\mathbb{E}\,\mathcal{E}(\hat{h}) \leq \delta$, for large n, provided $\delta = \omega(e^{-n})$.

A common relaxation of Massart's condition, is the so-called *Tsybakov noise condition* which parametrizes the likelihood of having a margin τ:

$$\forall\, 0 < \tau < 1/2, \quad P_X\left(x : |\eta(x) - 1/2| \leq \tau\right) \leq C_\beta \tau^\beta, \text{ for some } C_\beta, \beta > 0.$$

Now, define $\tau_{n,\delta} \doteq C\left(\frac{\log(n/\delta)}{n}\right)^{1/(2+d)} < 1/2$, for n sufficiently large. Under the event of (1.5), the excess risk is 0 at all points in $\mathcal{X}_> \doteq \{x : |\eta(x) - 1/2| > \tau_{n,\delta}\}$. Let $\mathcal{X}_\leq \doteq \mathcal{X} \setminus \mathcal{X}_>$. We therefore have that, with probability at least $1 - \delta$,

$$\mathcal{E}(\hat{h}) \leq \int_{\mathcal{X}_\leq} 2|\eta(x) - 1/2|\, \mathrm{d}P_X \leq 2\tau_{n,\delta} \cdot \int_{\mathcal{X}_\leq} \mathrm{d}P_X \leq 2C_\beta \cdot \tau_{n,\delta}^{\beta+1}.$$

Thus, we have $\mathbb{E}\,\mathcal{E}(\hat{h}) \leq C\left(\frac{\log(n/\delta)}{n}\right)^{(\beta+1)/(2+d)} + \delta$. In other words, the rate is much faster than $n^{-1/(2+d)}$ for large β. For example, let $\delta = 1/n$, and $\beta \geq d/2$, and the rates are at most $n^{-1/2}$.

Remark (Tension between parameters) Larger values of $\beta > d$ only happen in restricted situations where η crosses 1/2 outside of $\text{int}(\mathcal{X})$, due to the fact that the Lipschitz assumption on η prohibits sharp transitions from 1/2 (see Audibert and Tsybakov (2007)). Such tension disappears for more general distributions that homogeneous P_X (which corresponds to so-called *strong density* conditions). However, assuming more general conditions on P_X, e.g., only that it has support $\mathcal{X}$ of metric dimension d, minimax rates are slower of the form $n^{-(\beta+1)/(2+d+\beta)}$.

Data Dependent Choice of k. It remains unclear whether a global choice of k, e.g., by cross-validation, achieves the above rates in terms of β. In particular the above arguments required pointwise guarantees over x as in (1.5), while cross-validation only yields guarantees on global error. However, suitable local choices of $k = k(x)$, e.g., by variants of so-called Lepski's method, yield the above rates – up to log terms – without prior knowledge of d or β (see e.g. Kpotufe and Martinet (2018)).

Multi-class Settings. In common classification problems, e.g., object detection, speech, we are in fact dealing with a large number of classes. Therefore, let $Y \in \{1, 2, \ldots, L\}$, and for convenience consider the equivalent encoding $\tilde{Y} \in \{0, 1\}^L$, with coordinate $\tilde{Y}_l = \mathbb{1}\{Y = l\}$. We can now let the regression function $\eta(x) \doteq \mathbb{E}[\tilde{Y}|x]$, with corresponding k-NN estimate $\hat{\eta}(x) = \frac{1}{k}\sum_{X_i \in k\mathrm{NN}(x)} \tilde{Y}_i$.

Now the Bayes classifier is given by $h^*(x) = \arg\max_{l\in[L]} \eta_l(x)$, and similarly obtain the k-NN classifier as $\hat{h}(x) = \arg\max_{l\in[L]} \hat{\eta}_l(x)$. Whether $\hat{h}(x) \neq h^*(x)$ has to do with how well $\hat{\eta}(x)$ estimates $\eta(x)$, as a function of how it is to distinguish the largest coordinate of $\eta(x)$ – say $\eta_{(1)}(x)$ – from the second largest, say $\eta_{(2)}(x)$. Hence, a natural extension to the above noise conditions is as follows:

$$\forall\, 0 < \tau < 1/2, \quad P_X\left(x : \eta_{(1)}(x) \le \eta_{(2)}(x) + \tau\right) \le C_\beta \tau^\beta, \text{ for some } C_\beta, \beta > 0.$$

The resulting rates are similar under Lipschitz conditions on η (albeit, with an additional $\log L$ term in the rates; see e.g. Reeve and Brown (2018)).

1.3.5 Wrap-up: Statistical Complexity

We presented an overview of conditions, or parametrizations of data spaces, going from worst-case to more favorable to statistical performance:

(a). Notions of dimension similar to those used in analyzing NN search algorithms. These are not enough on their own, i.e., rates of convergence can be arbitrarily slow even with this condition since η (or $P_{Y|X}$) can be arbitrarily complex.

(b). Lipschitz or Holder conditions on the smoothness of η, together with (a), can give bounds of the form $n^{-1/(2+d)}$ that are adaptive to $d \ll D$ for $X \in \mathbb{R}^D$.

(c). Massart/Tsybakov conditions on the "margin": how much of η stays away from $1/2$. Under these conditions, much better rates are possible, e.g., $1/\sqrt{n}$.

Conditions (a) are sometimes verifiable, e.g. by appealing to manifold structure or sparsity. But (b), and (c) can be hard to check in practice, although they might be expected to approximately hold.

Together, the above conditions alleviate the worst-case nature of the minimax-approach by identifying favorable distributional parameters. Yet, they are still not refined enough, given that many predictors can be shown to be rate-optimal under these conditions (e.g., k-NN, ϵ-NN, various classification trees) but are observed to achieve rather different performance in practice.

Tradeoffs with Fast Search. It is interesting to note that the above analysis and rates remain relevant – up to constants – whenever fast search methods return *approximate* nearest neighbors, since in any case we only needed to bound nearest neighbor distances approximately to obtain the above rates. However, changes in constants matter in practice (c.f. the discussion of MNIST in Section 1.2.1), but unfortunately are not captured by the type of analysis outlined above. There is also a general need for statistical considerations in the design of fast search methods – which largely involve decisions based on marginal X and do not take signal in Y into account, e.g., how slowly labels Y change over X space.

1.4 Bibliographic Notes

References for algorithmic aspects of nearest neighbor search are mostly provided in the main text. The article of Clarkson (1999) on nearest neighbor methods in metric spaces is especially recommended, as is the survey of Cutler (1993) on notions of dimension. For recent developments in locality-sensitive hashing, there is a webpage maintained by Andoni, at `https://www.mit.edu/~andoni/LSH/`.

Universal consistency of nearest neighbor methods are first established in Fix and Hodges (1951), Stone (1977), and Devroye et al. (1994) with recent generalizations by Chaudhuri and Dasgupta (2014) and Hanneke et al. (2019) to metric spaces and beyond. Early rates of convergence were given by Cover (1968), Wagner (1971), Fritz (1975), Kulkarni and Posner (1995), and Gyorfi (1981). Various other predictors – local in nature – can be shown to converge at rates adaptive to the unknown intrinsic dimension of data, see e.g., Scott and Nowak (2006), Bickel and Li (2007), Kpotufe and Dasgupta (2012), and Yang and Dunson (2016). Finally, a recent book of Chen et al. (2018) gives a comprehensive theoretical survey of nearest neighbor methods.

References

Abdullah, A., Andoni, A., Kannan, R., and Krauthgamer, R. 2014. Spectral approaches to nearest neighbor search. In: *55th Annual Symposium on Foundations of Computer Science.*

Andoni, A., and Indyk, P. 2008. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Communications of the ACM*, **51**(1), 117–122.

Andoni, A., and Razenshteyn, I. 2015. Optimal Data-Dependent Hashing for Approximate Near Neighbors. In: *ACM Symposium on Theory of Computing.*

Andoni, A., Naor, A., Nikolov, A., Razenshteyn, I., and Waingarten, E. 2018. Data-dependent hashing via nonlinear spectral gaps. In: *ACM Symposium on Theory of Computing.*

Audibert, J.-Y., and Tsybakov, A.B. 2007. Fast learning rates for plug-in classifiers. *Ann. Statist.*, **35**(2), 608–633.

Bentley, J.L. 1975. Multidimensional Binary Search Trees Used for Associative Searching. *Communications of the ACM*, **18**(9), 509–517.

Beygelzimer, A., Kakade, S., and Langford, J. 2006. Cover trees for nearest neighbor. In: *Proceedings of the 23rd International Conference on Machine Learning.*

Bickel, P.J., and Li, B. 2007. Local polynomial regression on unknown manifolds. Pages 177–186 of: *Complex datasets and inverse problems.* Institute of Mathematical Statistics.

Charikar, M. 2002. Similarity estimation techniques from rounding algorithms. Pages 380–388 of: *Proceedings of the 34th ACM Symposium on Theory of Computing.*

Chaudhuri, K., and Dasgupta, S. 2014. Rates of convergence for nearest neighbor classification. Pages 3437–3445 of: *Advances in Neural Information Processing Systems.*

Chen, George H, Shah, Devavrat, et al. 2018. Explaining the success of nearest neighbor methods in prediction. *Foundations and Trends® in Machine Learning*, **10**(5-6), 337–588.

Clarkson, K. 1999. Nearest neighbor queries in metric spaces. *Discrete and Computational Geometry*, **22**, 63–93.

Clarkson, K. 2005. Nearest-neighbor searching and metric space dimensions. In: *Nearest-Neighbor Methods for Learning and Vision: Theory and Practice.* MIT Press.

Cover, T.M. 1968. Rates of convergence for nearest neighbor procedures. In: *Proceedings of The Hawaii International Conference on System Sciences.*

Cutler, C. 1993. A review of the theory and estimation of fractal dimension. Pages 1–107 of: Tong, H. (ed), *Dimension Estimation and Models.* World Scientific.

Dasgupta, S., and Freund, Y. 2008. Random projection trees and low dimensional manifolds. Pages 537–546 of: *ACM Symposium on Theory of Computing.*

Dasgupta, S., and Sinha, K. 2015. Randomized partition trees for nearest neighbor search. *Algorithmica*, **72**(1), 237–263.

Datar, M., Immorlica, N., Indyk, P., and Mirrokni, V. 2004. Locality-sensitive hashing based on p-stable distributions. In: *Proceedings of the Twentieth Annual Symposium on Computational Geometry.*

Devroye, L., Gyorfi, L., Krzyzak, A., Lugosi, G., et al. 1994. On the strong universal consistency of nearest neighbor regression function estimates. *The Annals of Statistics*, **22**(3), 1371–1385.

Devroye, L., Gyorfi, L., and Lugosi, G. 1997. *A Probabilistic Theory of Pattern Recognition.* Springer.

Fix, E., and Hodges, J. 1951. Discriminatory analysis, nonparametric discrimination. *USAF School of Aviation Medicine, Randolph Field, Texas, Project 21-49-004, Report 4, Contract AD41(128)-31.*

Fritz, J. 1975. Distribution-free exponential error bound for nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, **21**(5), 552–557.

Gadat, Sébastien, Klein, Thierry, Marteau, Clément, et al. 2016. Classification in general finite dimensional spaces with the k-nearest neighbor rule. *The Annals of Statistics*, **44**(3), 982–1009.

Gyorfi, L. 1981. The rate of convergence of k_n-NN regression estimates and classification rules. *IEEE Transactions on Information Theory*, **27**(3), 362–364.

Györfi, L., Kohler, M., Krzyzak, A., and Walk, H. 2006. *A distribution-free theory of nonparametric regression.* Springer Science & Business Media.

Hanneke, S., Kontorovich, A., Sabato, S., and Weiss, R. 2019. Universal Bayes consistency in metric spaces. *arXiv preprint arXiv:1906.09855.*

Hyvonen, V., Pitkanen, T., Tasoulis, S., Jaasaari, E., Tuomainen, R., Wang, L., Corander, J., and Roos, T. 2016. Fast nearest neighbor search through sparse random projections and voting. In: *Proceedings of the 2016 IEEE International Conference on Big Data*.

Indyk, P., and Motwani, R. 1998. Approximate nearest neighbors: Towards removing the curse of dimensionality. Pages 604–613 of: *Proceedings of the 30th Annual ACM Symposium on Theory of Computing.*

Kpotufe, S. 2011. k-NN regression adapts to local intrinsic dimension. Pages 729–737 of: *Advances in Neural Information Processing Systems.*

Kpotufe, S., and Dasgupta, S. 2012. A tree-based regressor that adapts to intrinsic dimension. *Journal of Computer and System Sciences*, **78**(5), 1496–1515.

Kpotufe, S., and Martinet, G. 2018. Marginal singularity, and the benefits of labels in covariate-shift. *arXiv preprint arXiv:1803.01833.*

Krauthgamer, R., and Lee, J.R. 2004. Navigating nets: simple algorithms for proximity search. In: *ACM-SIAM Symposium on Discrete Algorithms.*

Kulkarni, S., and Posner, S. 1995. Rates of convergence of nearest neighbor estimation under arbitrary sampling. *IEEE Transactions on Information Theory*, **41**(4), 1028–1039.

Luukkainen, J., and Saksman, E. 1998. Every complete doubling metric space carries a doubling measure. *Proceedings of the American Mathematical Society*, **126**(2), 531–534.

McFee, B., and Lanckriet, G. 2011. Large-scale music similarity search with spatial trees. In: *12th Conference of the International Society for Music Retrieval.*

Reeve, H.W.J., and Brown, G. 2018. Minimax rates for cost-sensitive learning on manifolds with approximate nearest neighbours. *arXiv preprint arXiv:1803.00310.*

Saksman, E. 1999. Remarks on the nonexistence of doubling measures. Pages 155–164 of: *Annales-Academiae Scientiarum Fennicae Series A1 Mathematica*, vol. 24. Academia Scientiarum Fennicae.

Samworth, Richard J, et al. 2012. Optimal weighted nearest neighbour classifiers. *The Annals of Statistics*, **40**(5), 2733–2763.

Scott, C., and Nowak, R.D. 2006. Minimax-optimal classification with dyadic decision trees. *IEEE Transactions on Information Theory*, **52**(4), 1335–1353.

Sproull, R.F. 1991. Refinements to nearest-neighbor searching in k-dimensional trees. *Algorithmica*, **6**(1), 579–589.

Stone, C.J. 1977. Consistent nonparametric regression. *The Annals of Statistics*, 595–620.

Urner, R., Shalev-Shwartz, S., and Ben-David, S. 2011. Access to unlabeled data can speed up prediction time. Pages 641–648 of: *Proceedings of the 28th International Conference on Machine Learning (ICML-11).*

Wagner, T.J. 1971. Convergence of the nearest neighbor rule. *IEEE Transactions on Information Theory*, **17**(5), 566–571.

Willett, R., Nowak, R., and Castro, R.M. 2006. Faster rates in regression via active learning. Pages 179–186 of: *Advances in Neural Information Processing Systems.*

Yang, Y., and Dunson, D.B. 2016. Bayesian manifold regression. *The Annals of Statistics*, **44**(2), 876–905.

Exercises

1.1 *Implications of doubling properties.*

(a) Show that if $(\mathcal{X}, \rho)$ is a bounded metric, with doubling dimension d, then it has metric dimension d.

(b) Show that if μ is a doubling measure with exponent d on a bounded metric $(\mathcal{X}, \rho)$, then it is homogeneous (on its support) with parameters (C_d, d) for some C_d.

1.2 *Relation between doubling measures and metrics.*

(a) Show that if there exists a doubling measure μ on the metric $(\mathcal{X}, \rho)$ with exponent d, then $(\mathcal{X}, \rho)$ must be *doubling*, with doubling dimension $O(d)$. *Hint: consider maximal packings of a ball by smaller balls.*

The reverse is often true, i.e., every *complete* doubling metric admits a doubling measure (Luukkainen and Saksman, 1998; Saksman, 1999).

(b) Show that if there exists a doubling measure μ on the metric $(\mathcal{X}, \rho)$ with exponent d, then $(\mathcal{X}, \rho)$ has metric dimension d (in fact every ball $B(x, r)$ can be covered by $C_d \epsilon^{-d}$ balls of radius ϵr, $\forall \epsilon \in (0, 1]$ and some constant C_d independent of x and r).

1.3 *Comprehensive search for k-d trees.* Given a k-d tree built on a data set $S \subset \mathbb{R}^D$ and a query q, a *comprehensive search* begins by finding the nearest point in the leaf cell containing q; call this point x_o. It then expands its search to other leaf cells that might potentially contain an even closer point: namely, those that intersect the ball $B(q, r)$, where $r = \|q - x_o\|$. Along the way, it keeps updating its current-best nearest neighbor and search radius r, and is guaranteed to return the true nearest neighbor. Flesh out an algorithm that implements this logic via a suitable tree traversal.

1.4 *ϵ-NN classification.* Under the assumptions of Theorem 1.3, let $\epsilon = Cn^{-1/(2+d)}$, for some $C > 0$. Let $\hat{h}(x) = \mathbb{1}\{\hat{\eta}(x) \geq 1/2\}$, where, for $n_\epsilon(x) \doteq |\mathbf{X} \cap B(x, \epsilon)|$,

$$\hat{\eta}(x) = \frac{1}{n_\epsilon(x)} \sum_{X_i \in B(x,\epsilon)} Y_i \cdot \mathbb{1}\{n_\epsilon(x) \geq 1\}, \; \forall x \in \operatorname{supp}(P_X).$$

(a) Argue that $\mathbb{E}_{\mathbf{Y}|\mathbf{X}} \|\hat{\eta}(x) - \eta(x)\|^2 \leq \frac{1}{n_\epsilon(x)} \mathbb{1}\{n_\epsilon(x) \geq 1\} + \lambda\epsilon^2 + \mathbb{1}\{n_\epsilon(x) = 0\}$.

(b) Argue that $\mathbb{E}_{\mathbf{X}} \mathbb{1}\{n_\epsilon(x) = 0\} \leq C'/(nP_X(B(x, \epsilon))$ for suitable C, C'.

(c) Using the fact that for a Binomial Z s.t. $\mathbb{E}Z \geq 1$, we have $\mathbb{E}\frac{\mathbb{1}\{Z \geq 1\}}{Z} \leq 3/\mathbb{E}Z$ (Lemma 4.1 of Györfi et al. (2006)), bound $\mathbb{E}\,\mathcal{E}(\hat{h})$, and conclude that $\hat{h}$ achieves the same rates as derived for k-NN in Theorem 1.3.