Geometric algorithms for classification and retrieval in high dimension

Sanjoy Dasgupta

University of California, San Diego

Retrieval and classification

$\mathcal{X}$ = space of data items

- images
- documents
- speech recordings
- medical records
- ...

Retrieval:

- Given: collection of items $x_1, \ldots, x_n \in \mathcal{X}$
- Later: for query $x \in \mathcal{X}$, return closest match(es) amongst the x_i

Retrieval and classification

$\mathcal{X}$ = space of data items

- images
- documents
- speech recordings
- medical records
- ...

Retrieval:

- Given: collection of items $x_1, \ldots, x_n \in \mathcal{X}$
- Later: for query $x \in \mathcal{X}$, return closest match(es) amongst the x_i
- Algorithmic question: how to do this quickly?

Retrieval and classification

$\mathcal{X}$ = space of data items

- images
- documents
- speech recordings
- medical records
- $\cdots$

Retrieval:

- Given: collection of items $x_1, \ldots, x_n \in \mathcal{X}$
- Later: for query $x \in \mathcal{X}$, return closest match(es) amongst the x_i
- Algorithmic question: how to do this quickly?

Classification:

- Given: collection of labeled items $(x_1, y_1), \ldots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$
- *Learn* a classification rule $f : \mathcal{X} \to \mathcal{Y}$
- Later: for query $x \in \mathcal{X}$, predict label $f(x)$

Retrieval and classification

$\mathcal{X}$ = space of data items

- images
- documents
- speech recordings
- medical records
- ...

Retrieval:

- Given: collection of items $x_1, \ldots, x_n \in \mathcal{X}$
- Later: for query $x \in \mathcal{X}$, return closest match(es) amongst the x_i
- Algorithmic question: how to do this quickly?

Classification:

- Given: collection of labeled items $(x_1, y_1), \ldots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$
- *Learn* a classification rule $f : \mathcal{X} \to \mathcal{Y}$
- Later: for query $x \in \mathcal{X}$, predict label $f(x)$
- Statistical question: how much data is needed to find a good rule?

Dimension

E.g. Data on heart patients leaving a hospital:

$$(\text{age}, \text{weight}, \text{temp}, \text{bp1}, \text{bp2}, \ldots)$$

If d features, each data point is a vector in $\mathbb{R}^d$.

Dimension

E.g. Data on heart patients leaving a hospital:

$$(\text{age}, \text{weight}, \text{temp}, \text{bp1}, \text{bp2}, \ldots)$$

If d features, each data point is a vector in $\mathbb{R}^d$.

Problem:

- the algorithmic complexity of retrieval and
- the statistical complexity of nonparametric classification

grow very rapidly with d.

Dimension

E.g. Data on heart patients leaving a hospital:

$$(\text{age}, \text{weight}, \text{temp}, \text{bp1}, \text{bp2}, \ldots)$$

If d features, each data point is a vector in $\mathbb{R}^d$.

Problem:

- the algorithmic complexity of retrieval and
- the statistical complexity of nonparametric classification

grow very rapidly with d.

One way to conceptualize and manage this:

- Actual **degrees of freedom** are often much smaller than the apparent dimension d

Dimension

E.g. Data on heart patients leaving a hospital:

$$(\text{age}, \text{weight}, \text{temp}, \text{bp1}, \text{bp2}, \ldots)$$

If d features, each data point is a vector in $\mathbb{R}^d$.

Problem:

- the algorithmic complexity of retrieval and
- the statistical complexity of nonparametric classification

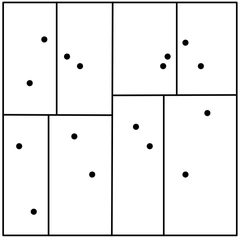
grow very rapidly with d.

One way to conceptualize and manage this:

- Actual **degrees of freedom** are often much smaller than the apparent dimension d
- Formalize a notion of **intrinsic dimension**
- Develop methods for retrieval and classification whose complexity scales with the intrinsic dimension, not with d

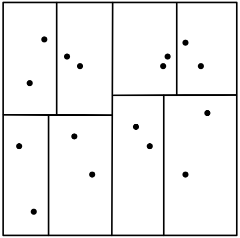
Intrinsic dimension $d_o \ll$ apparent dimension d

Intrinsic dimension $d_o \ll$ apparent dimension d

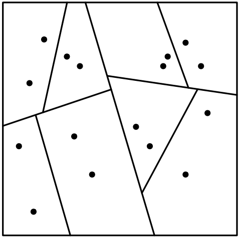


Classification error depends on d

Intrinsic dimension $d_o \ll$ apparent dimension d

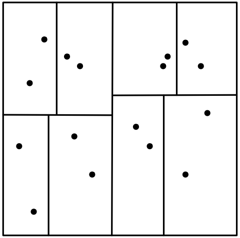


Classification error depends on d

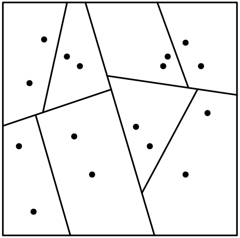


Classification error depends on d_o

Intrinsic dimension $d_o \ll$ apparent dimension d



Classification error depends on d



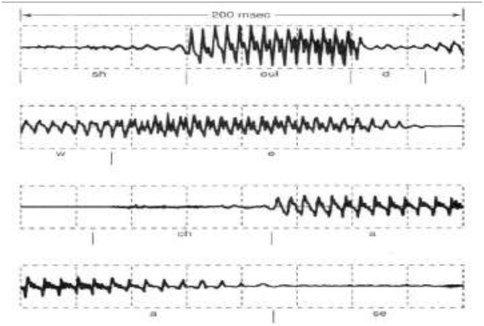
Classification error depends on d_o

The same method yields state-of-the-art nearest neighbor search.

Outline

1. Intrinsic dimension
2. Classification
3. Retrieval

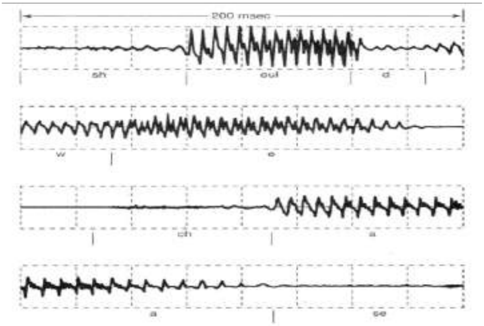
Degrees of freedom



Common representation of speech:

- Take overlapping windows of the speech signal
- Apply many filters within each window
- More filters $\Rightarrow$ higher dimensional

Degrees of freedom



Common representation of speech:

- Take overlapping windows of the speech signal
- Apply many filters within each window
- More filters $\Rightarrow$ higher dimensional

But the speech has been produced by a physical system (vocal tract) with a fixed number of degrees of freedom.

Low dimensional manifolds

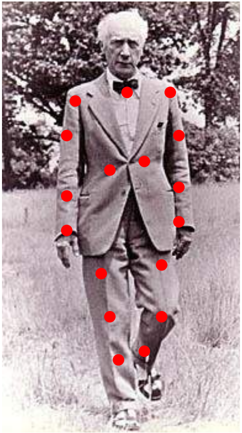
Manifold learning: handling data in a high-dimensional space $\mathbb{R}^d$ that lie close to a d-dimensional manifold, for $d_o \ll d$

- Speech example

Low dimensional manifolds

Manifold learning: handling data in a high-dimensional space $\mathbb{R}^d$ that lie close to a d-dimensional manifold, for $d_o \ll d$

- Speech example
- Motion capture
  M markers on a human body yields data in $\mathbb{R}^{3M}$

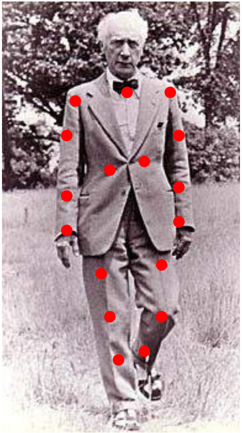


Low dimensional manifolds

Manifold learning: handling data in a high-dimensional space $\mathbb{R}^d$ that lie close to a d-dimensional manifold, for $d_o \ll d$

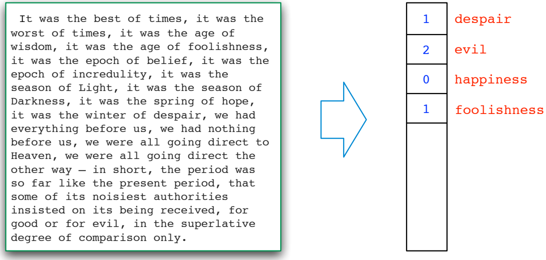
- Speech example
- Motion capture
  M markers on a human body yields data in $\mathbb{R}^{3M}$

Typical approach: approximately identify the manifold and use this to reduce dimension



Another example of low intrinsic dimension

Bag-of-words document model



- Fix a vocabulary of size d
- A document is represented by a d-dimensional vector indicating, for each word, whether it appears (or how often)

Average number of nonzero entries in these vectors is $d_o \ll d$.

Unifying notion of intrinsic dimension?

There are several widely-occurring types of low intrinsic dimension.

Can we:

- Find a broad notion of dimensionality that captures at least a few of these?
- Develop methods for classification and retrieval whose complexity depends only on this refined notion rather than on the superficial apparent dimension?

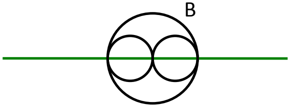
Doubling dimension

Set $S \subset \mathbb{R}^d$ has *doubling dimension* d_o if for any (Euclidean) ball B, the subset $S \cap B$ can be covered by 2^{d_o} balls of half the radius [Assouad, Gupta-Krauthgamer-Lee].

Doubling dimension

Set $S \subset \mathbb{R}^d$ has *doubling dimension* d_o if for any (Euclidean) ball B, the subset $S \cap B$ can be covered by 2^{d_o} balls of half the radius [Assouad, Gupta-Krauthgamer-Lee].

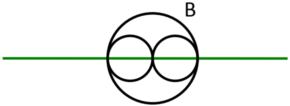
1. Example: $S =$ line has doubling dimension 1.



Doubling dimension

Set $S \subset \mathbb{R}^d$ has *doubling dimension* d_o if for any (Euclidean) ball B, the subset $S \cap B$ can be covered by 2^{d_o} balls of half the radius [Assouad, Gupta-Krauthgamer-Lee].

1. Example: $S =$ line has doubling dimension 1.

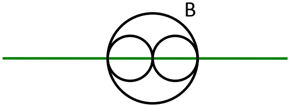


2. A k-dimensional flat has doubling dimension $c_o k$ for some absolute constant c_o.

Doubling dimension

Set $S \subset \mathbb{R}^d$ has *doubling dimension* d_o if for any (Euclidean) ball B, the subset $S \cap B$ can be covered by 2^{d_o} balls of half the radius [Assouad, Gupta-Krauthgamer-Lee].

1. Example: $S =$ line has doubling dimension 1.

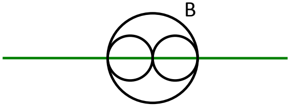


2. A k-dimensional flat has doubling dimension $c_o k$ for some absolute constant c_o.
3. If a k-dimensional Riemannian submanifold of $\mathbb{R}^d$ has "condition number" $1/\tau$, then its neighborhoods of radius τ have doubling dimension $O(k)$.

Doubling dimension

Set $S \subset \mathbb{R}^d$ has *doubling dimension* d_o if for any (Euclidean) ball B, the subset $S \cap B$ can be covered by 2^{d_o} balls of half the radius [Assouad, Gupta-Krauthgamer-Lee].

1. Example: $S =$ line has doubling dimension 1.

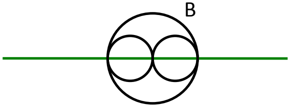


2. A k-dimensional flat has doubling dimension $c_o k$ for some absolute constant c_o.
3. If a k-dimensional Riemannian submanifold of $\mathbb{R}^d$ has "condition number" $1/\tau$, then its neighborhoods of radius τ have doubling dimension $O(k)$.
4. If points in $S \subset \mathbb{R}^d$ have $\leq k$ nonzero coordinates, then S has doubling dimension $\leq c_o k + k \log(d/k)$.

Doubling dimension

Set $S \subset \mathbb{R}^d$ has *doubling dimension* d_o if for any (Euclidean) ball B, the subset $S \cap B$ can be covered by 2^{d_o} balls of half the radius [Assouad, Gupta-Krauthgamer-Lee].

1. Example: $S =$ line has doubling dimension 1.



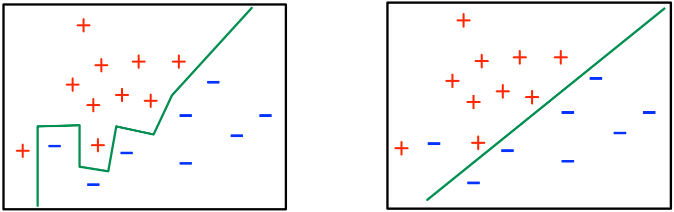
2. A k-dimensional flat has doubling dimension $c_o k$ for some absolute constant c_o.
3. If a k-dimensional Riemannian submanifold of $\mathbb{R}^d$ has "condition number" $1/\tau$, then its neighborhoods of radius τ have doubling dimension $O(k)$.
4. If points in $S \subset \mathbb{R}^d$ have $\leq k$ nonzero coordinates, then S has doubling dimension $\leq c_o k + k \log(d/k)$.
5. If S has doubling dimension d_o, then so does any subset of S.

Outline

1. Intrinsic dimension
2. Classification
3. Retrieval

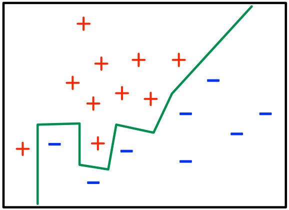
Nonparametric classification

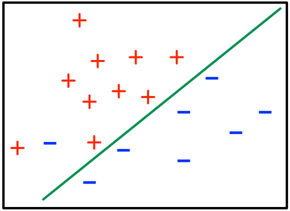
Nonparametric methods can fit any function.



Nonparametric classification

Nonparametric methods can fit any function.

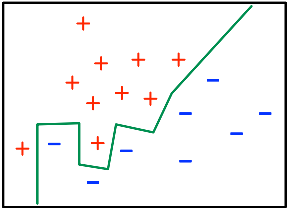


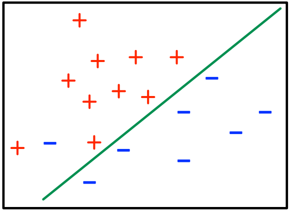


But they suffer a severe curse of dimension.

Nonparametric classification

Nonparametric methods can fit any function.





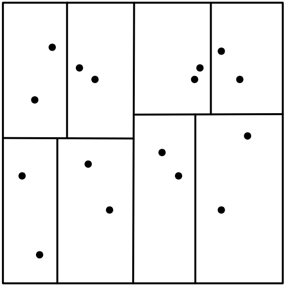
But they suffer a severe curse of dimension.

Consider random pair (X, Y), where $X \in \mathbb{R}^d$ and $Y \in \{0, 1\}$ is a label.

- Want to infer $f(x) = \mathbb{E}[Y|X = x]$.
- Let f_n be an estimator based on n data points. It is common to judge it by its squared loss $\mathbb{E}(f_n(X) - f(X))^2$.
- Stone 1982: Loss $\geq n^{-2p/(2p+d)}$, where p captures smoothness of f.

Spatial partitioning for nonparametric estimation

e.g. the k-d tree:



To split a cell with points S:

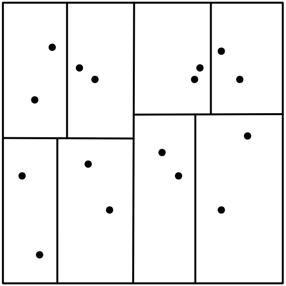
- Choose a coordinate direction
- Split at the median along that direction

Once the tree is built:

- Fit a simple model (e.g. constant) in each leaf.
- Answer a query by routing it to a leaf and applying the leaf's model.

Spatial partitioning for nonparametric estimation

e.g. the k-d tree:



To split a cell with points S:

- Choose a coordinate direction
- Split at the median along that direction

Once the tree is built:

- Fit a simple model (e.g. constant) in each leaf.
- Answer a query by routing it to a leaf and applying the leaf's model.

These estimators are consistent if, as $n \to \infty$,

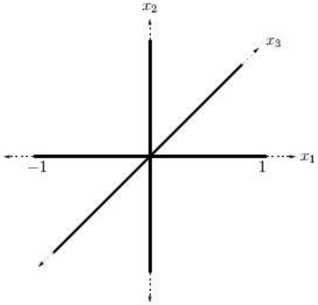
1. the diameter of the leaf cells goes to zero, and
2. the number of samples in each leaf goes to infinity.

Rate of convergence depends on relative speed of these two effects.

k-d trees are not adaptive to intrinsic dimension

As one moves down a k-d tree, how rapidly does the cell diameter shrink?

Consider the data set $S = \cup_{i=1}^{d} \{te_i : -1 \leq t \leq 1\}$.

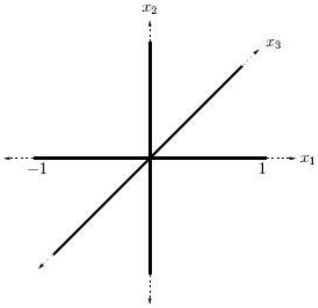


At least d levels are needed to halve the diameter.

k-d trees are not adaptive to intrinsic dimension

As one moves down a k-d tree, how rapidly does the cell diameter shrink?

Consider the data set $S = \cup_{i=1}^{d} \{te_i : -1 \leq t \leq 1\}$.

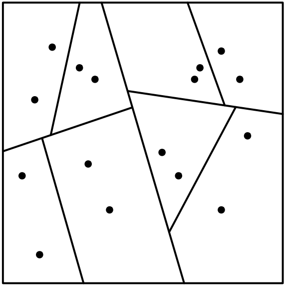


At least d levels are needed to halve the diameter.

Yet S has doubling dimension just $d_o = 1 + \log d$.

Random projection trees

A randomized variant of the k-d tree

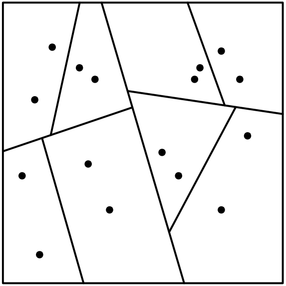


To split a cell with points $S \subset \mathbb{R}^d$:

- Choose a direction v at random from the unit sphere
- Split at the median along that direction, perturbed slightly

Random projection trees

A randomized variant of the k-d tree



To split a cell with points $S \subset \mathbb{R}^d$:

- Choose a direction v at random from the unit sphere
- Split at the median along that direction, perturbed slightly

Theorem: Pick any cell C in the tree. With probability at least $1/2$, every descendant cell C' which is more than $d_o \log d_o$ levels below C has $\text{diam}(C') \leq \text{diam}(C)/2$.

Here, $\text{diam}(C)$ is the maximum interpoint distance of data in cell C.

Properties of random projection

Pick a random vector U from the unit sphere in $\mathbb{R}^d$. Mapping:

$$\Pi(x) = U \cdot x$$

Almost the same: pick $U \sim N(0, (1/d)I_d)$.

Properties of random projection

Pick a random vector U from the unit sphere in $\mathbb{R}^d$. Mapping:

$$\Pi(x) = U \cdot x$$

Almost the same: pick $U \sim N(0, (1/d)I_d)$.

1. Effect of projection on a single point.

   Pick any x. As U varies, projection $\Pi(x)$ has a Gaussian distribution with mean zero and variance $\|x\|^2/d$.

   Therefore, concentrated in $[-\|x\|/\sqrt{d},\ \|x\|/\sqrt{d}]$.

Properties of random projection

Pick a random vector U from the unit sphere in $\mathbb{R}^d$. Mapping:

$$\Pi(x) = U \cdot x$$

Almost the same: pick $U \sim N(0, (1/d)I_d)$.

1. Effect of projection on a single point.

   Pick any x. As U varies, projection $\Pi(x)$ has a Gaussian distribution with mean zero and variance $\|x\|^2/d$.

   Therefore, concentrated in $[-\|x\|/\sqrt{d}, \ \|x\|/\sqrt{d}]$.

2. To extend to sets of points, generally need to take a union bound.

Properties of random projection

Pick a random vector U from the unit sphere in $\mathbb{R}^d$. Mapping:

$$\Pi(x) = U \cdot x$$

Almost the same: pick $U \sim N(0, (1/d)I_d)$.

1. Effect of projection on a single point.

   Pick any x. As U varies, projection $\Pi(x)$ has a Gaussian distribution with mean zero and variance $\|x\|^2/d$.

   Therefore, concentrated in $[-\|x\|/\sqrt{d},\ \|x\|/\sqrt{d}]$.

2. To extend to sets of points, generally need to take a union bound.

3. Median of projected points.

   If $S \subset B(x_o, \Delta)$, then

$$|\text{median}(\Pi(S)) - \Pi(x_o)| \ \leq\ O\left(\frac{\Delta}{\sqrt{d}}\right).$$

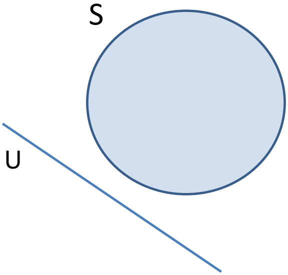
Random projection and diameter

For $S \subset \mathbb{R}^d$, how does the diameter of $\Pi(S)$ compare to that of S?

Random projection and diameter

For $S \subset \mathbb{R}^d$, how does the diameter of $\Pi(S)$ compare to that of S?

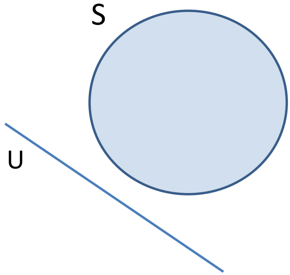
If S is full-dimensional, the diameter could be unchanged.



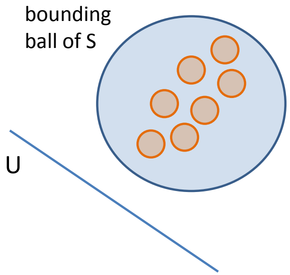
Random projection and diameter

For $S \subset \mathbb{R}^d$, how does the diameter of $\Pi(S)$ compare to that of S?

If S is full-dimensional, the diameter could be unchanged.



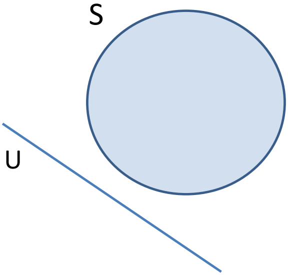
But if S has doubling dimension $d_o \ll d$, the diameter ought to shrink.



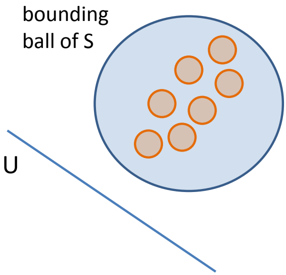
Random projection and diameter

For $S \subset \mathbb{R}^d$, how does the diameter of $\Pi(S)$ compare to that of S?

If S is full-dimensional, the diameter could be unchanged.



But if S has doubling dimension $d_o \ll d$, the diameter ought to shrink.



In the latter case, $\text{diam}(\Pi(S))$ is at most about $\text{diam}(S) \cdot \sqrt{d_o/d}$.

Random projection and diameter

Theorem: If $S \subset \mathbb{R}^d$ has doubling dimension d_o, then with probability at least $1-\delta$, the diameter of $\Pi(S)$ is at most

$$4 \cdot \frac{\text{diam}(S)}{\sqrt{d}} \cdot \sqrt{2\left(d_o + \ln \frac{2}{\delta}\right)}.$$

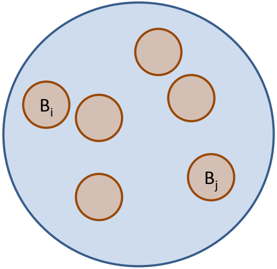
Proof: We'll prove a weaker version with factor $\sqrt{(d_o \log d)/d}$.

1. WLOG S has diameter 1 and $S \subset B(0,1)$.
2. Cover S by balls of radius $\sqrt{d_o/d}$. At most $(d/d_o)^{d_o/2}$ balls are needed.
3. Pick any of these balls. With probability $1-(1/d)^{d_o}$, its center is projected to a point within distance $\sqrt{(d_o \log d)/d}$ of the origin; and thus the entire projected ball lies in an interval within distance $\sqrt{(d_o \log d)/d} + \sqrt{d_o/d}$ of the origin.
4. Take a union bound over all the balls.

Proof outline for RP trees

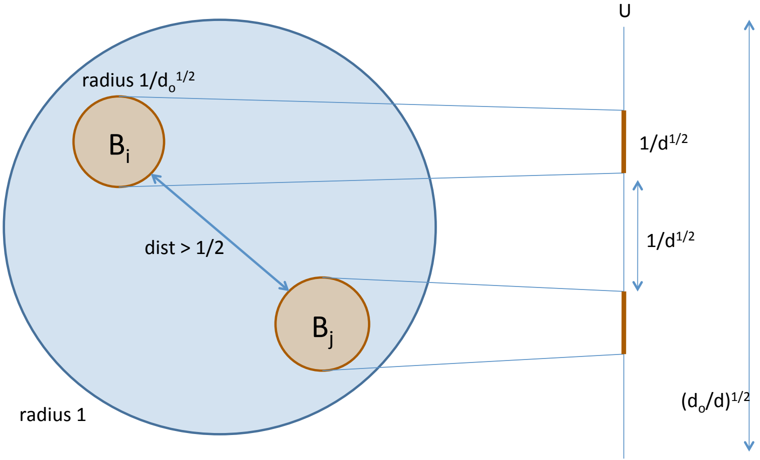
Suppose $S \subset \mathbb{R}^d$ has doubling dimension d_o and lies in a ball of radius 1. We need to show that if an RP tree is built on S, then with constant probability, every cell $O(d_o \log d_o)$ levels below is contained in ball of radius $1/2$.

Current cell (radius ≤ 1):



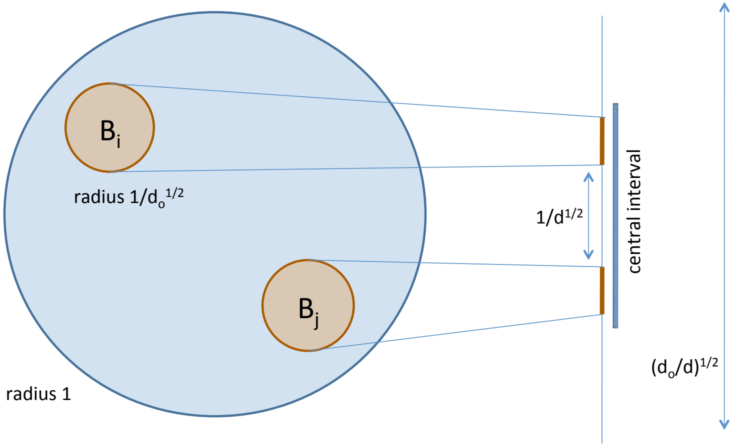
1. Cover S by $d_o^{d_o/2}$ balls B_i of radius $1/\sqrt{d_o}$.
2. Consider any pair of balls B_i, B_j that are distance $> 1/2 - 1/\sqrt{d_o}$ apart. We'll see that a single random split has constant probability of cleanly separating them.
3. There are at most $d_o^{d_o}$ such pairs, so after $O(d_o \log d_o)$ splits, with constant probability every faraway pair of balls will be separated. Thus all cells at that level will have radius $\leq 1/2$.

The big picture



Recall that random projection shrinks diameter by $\sqrt{d_o/d}$ and individual vectors by $1/\sqrt{d}$.

The big picture



Most projected points (and the median) fall in a central interval of size $1/\sqrt{d}$.

Outline

1. Intrinsic dimension
2. Classification
3. Retrieval

Nearest neighbor search

Given a data set of n points in $\mathbb{R}^d$, build a data structure for efficiently answering subsequent nearest neighbor queries q.

- Data structure should take space $O(n)$
- Query time should be $o(n)$

Nearest neighbor search

Given a data set of n points in $\mathbb{R}^d$, build a data structure for efficiently answering subsequent nearest neighbor queries q.

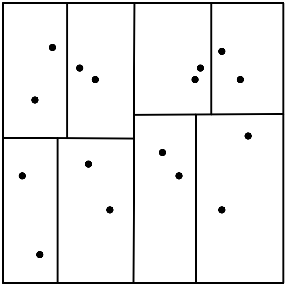
- Data structure should take space $O(n)$
- Query time should be $o(n)$

Unproven but common conjecture: for data structures of linear size, query time will be exponential in d.

Bad case: for any $0 < \epsilon < 1$,

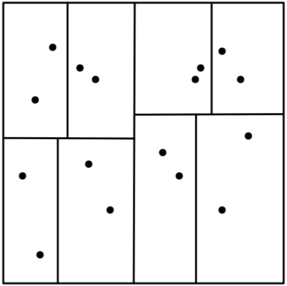
- Pick $2^{O(\epsilon^2 d)}$ points uniformly from the unit sphere in $\mathbb{R}^d$
- With high probability, all interpoint distances are $(1 \pm \epsilon)\sqrt{2}$

The k-d tree, again



Defeatist search: return NN in query's leaf node (may not be true NN).

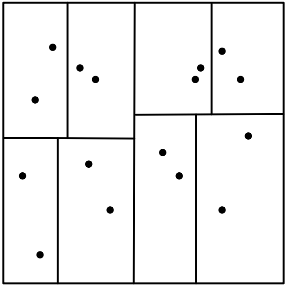
The k-d tree, again



Defeatist search: return NN in query's leaf node (may not be true NN).

Curse of dimension: chance of returning the true NN tends to drop dramatically with dimension.

The k-d tree, again



Defeatist search: return NN in query's leaf node (may not be true NN).

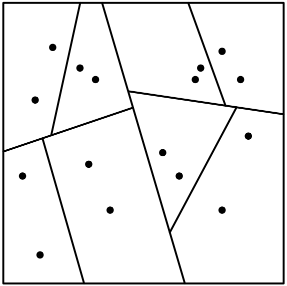
Curse of dimension: chance of returning the true NN tends to drop dramatically with dimension.

Some variants:

- Better split direction: PCA tree
- Overlapping cells (Maneewongvatana and Mount; Liu et al)
- Random split directions (Liu, Moore, Gray; Muja, Lowe)

Random projection trees

In each cell of the tree, pick split direction uniformly at random from the unit sphere in $\mathbb{R}^d$



Perturbed split: after projection, pick $\beta \in_R [1/4, 3/4]$ and split at the β-fractile point.

Failure probability

Pick any data set $x_1, \ldots, x_n$ and any query q.

- Let $x_{(1)}, \ldots, x_{(n)}$ be the ordering of data by distance from q.
- Probability of not returning the NN depends directly on

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

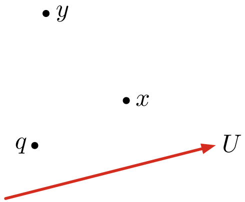
(This probability is over the randomization in tree construction.)

Random projection of three points

Let $q \in \mathbb{R}^d$ be the query, x its nearest neighbor and y some other point:

$$\|q - x\| < \|q - y\|.$$

Bad event: when the data is projected onto a random direction U, point y falls between q and x.

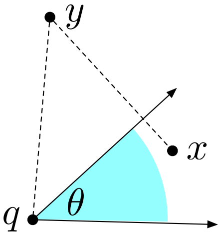


What is the probability of this?

This is a 2-d problem, in the plane defined by q, x, y.

- Only care about projection of U on this plane
- Projection of U is a random direction in this plane

Random projection of three points



Probability that U falls in this bad region is $\theta/2\pi$.

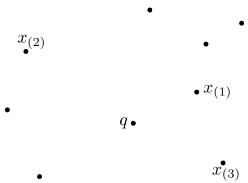
Lemma

Pick any three points $q, x, y \in \mathbb{R}^d$ such that $\|q - x\| < \|q - y\|$. Pick U uniformly at random from the unit sphere S^{d-1} in $\mathbb{R}^d$. Then

$$Pr(y \cdot U \text{ falls between } q \cdot U \text{ and } x \cdot U) \;\leq\; \frac{1}{2} \frac{\|q - x\|}{\|q - y\|}.$$

(Tight within a constant unless the points are almost-collinear)

Random projection of a set of points



Lemma

Pick any $x_1, \ldots, x_n$ and any query q. Pick $U \in_R S^{d-1}$ and project all points onto direction U. Then the expected fraction of the projected x_i that fall between q and $x_{(1)}$ is at most

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

Proof: Probability that $x_{(i)}$ falls between q and $x_{(1)}$ is at most $\frac{1}{2} \frac{\|q-x_{(1)}\|}{\|q-x_{(i)}\|}$. Now use linearity of expectation.

Bad event: this fraction is $\Omega(1)$. Happens with probability $O(\Phi)$.

Failure probability of NN search

Fix any data points $x_1, \ldots, x_n$ and query q. For $m \leq n$, define

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

Theorem

Suppose an RP tree is built for data set $x_1, \ldots, x_n$ with leaf nodes of size n_o. For any query q, the probability that the NN query does not return $x_{(1)}$ is at most

$$\sum_{i=0}^{\ell} \Phi_{(3/4)^i n}(q, \{x_1, \ldots, x_n\})$$

where $\ell = \log_{4/3}(n/n_o)$ is the tree's depth.

NN search in spaces of bounded doubling dimension

Need to bound

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

Suppose:

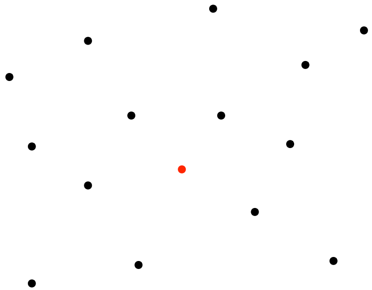
- Pick any $n+1$ points in $\mathbb{R}^d$ with doubling dimension d_o
- Randomly pick one of them as q; the rest are $x_1, \ldots, x_n$

Then $\mathbb{E}\Phi_m \leq 1/m^{1/d_o}$.

For constant failure probability, use tree with leaf size $n_o = O(d_o^{d_o})$, and query time $O(n_o + \log n)$.

How does doubling dimension help?

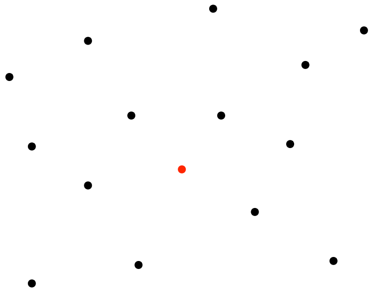
Pick any n points in $\mathbb{R}^d$. Pick one of these points, x. At most how many of the remaining points can have x as its nearest neighbor?



How does doubling dimension help?

Pick any n points in $\mathbb{R}^d$. Pick one of these points, x. At most how many of the remaining points can have x as its nearest neighbor?

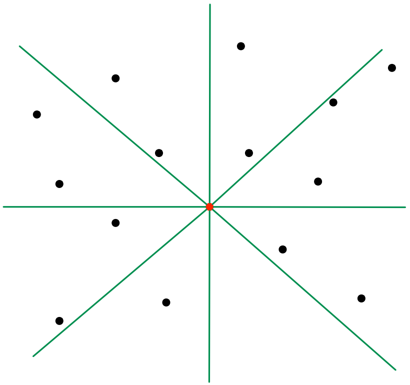
At most c^d, for some constant c [Stone].



How does doubling dimension help?

Pick any n points in $\mathbb{R}^d$. Pick one of these points, x. At most how many of the remaining points can have x as its nearest neighbor?

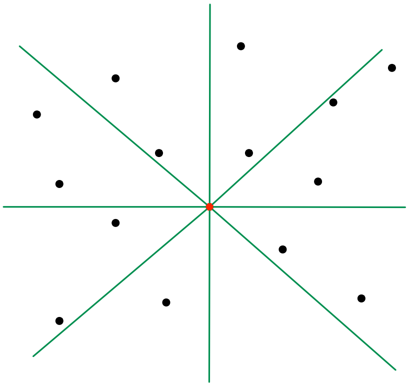
At most c^d, for some constant c [Stone].



How does doubling dimension help?

Pick any n points in $\mathbb{R}^d$. Pick one of these points, x. At most how many of the remaining points can have x as its nearest neighbor?

At most c^d, for some constant c [Stone].



Can (almost) replace d by the doubling dimension [Clarkson].

Randomized forests

To exploit randomization in the data structure:

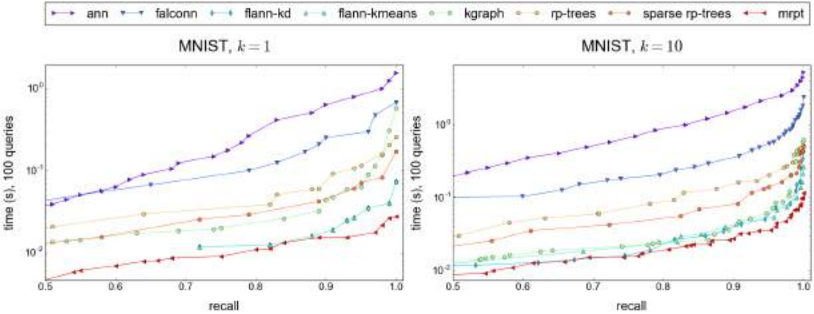
- Build multiple RP trees
- Upon query: return the closest among the NN results from each

Randomized forests

To exploit randomization in the data structure:

- Build multiple RP trees
- Upon query: return the closest among the NN results from each

Experiments by Roos et al:



Open problems

1. Working in general metric spaces.
   Simple randomized partition trees for metric spaces?

Open problems

1. Working in general metric spaces.
   Simple randomized partition trees for metric spaces?

2. More general notions of intrinsic dimension.
   Get closer to underlying "degrees of freedom" of input space?