

Nearest Neighbor Search: Retrieving Documents



Emily Fox & Carlos Guestrin

Machine Learning Specialization

University of Washington

Retrieving documents of interest

Document retrieval

- Currently reading article you like



Document retrieval

- Currently reading article you like
- **Goal:** Want to find similar article



Document retrieval



Challenges

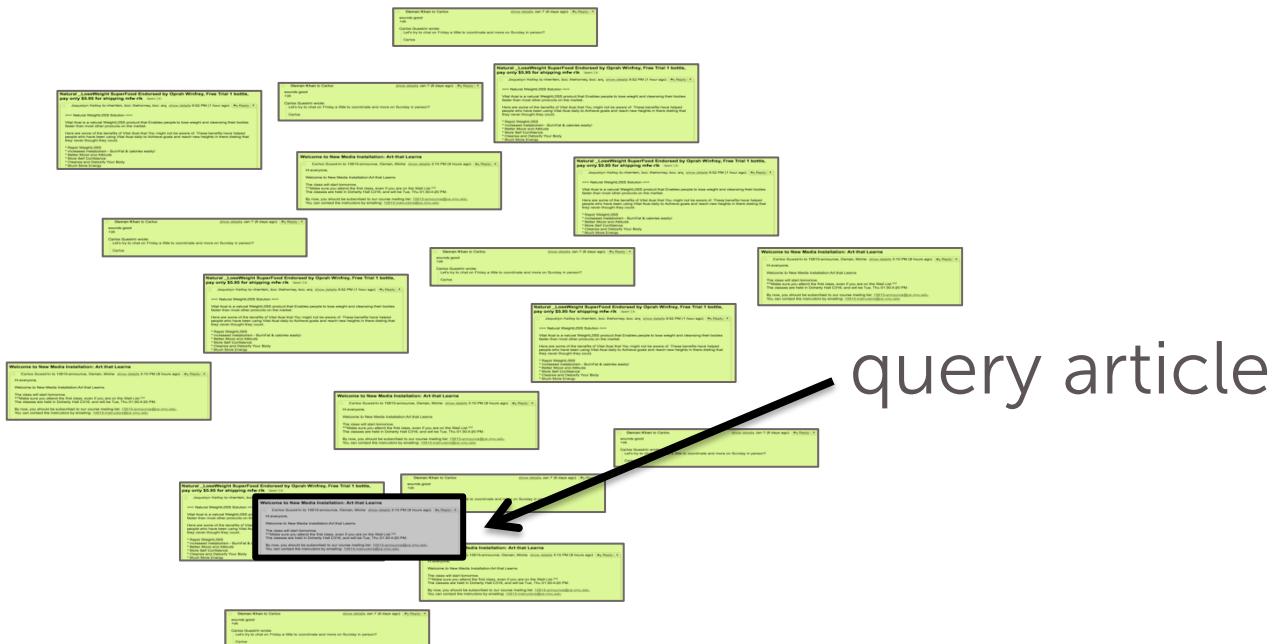
- How do we measure similarity?
- How do we search over articles?



Retrieval as k-nearest neighbor search

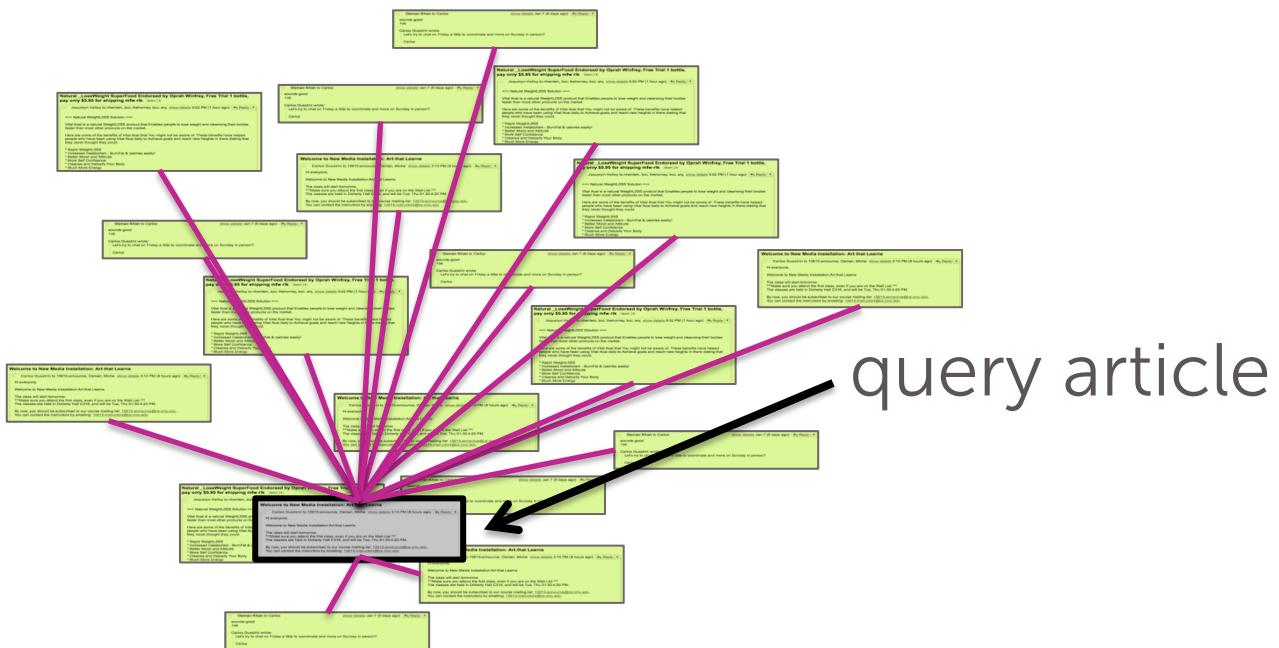
1-NN search for retrieval

Space of all articles,
organized by similarity of text



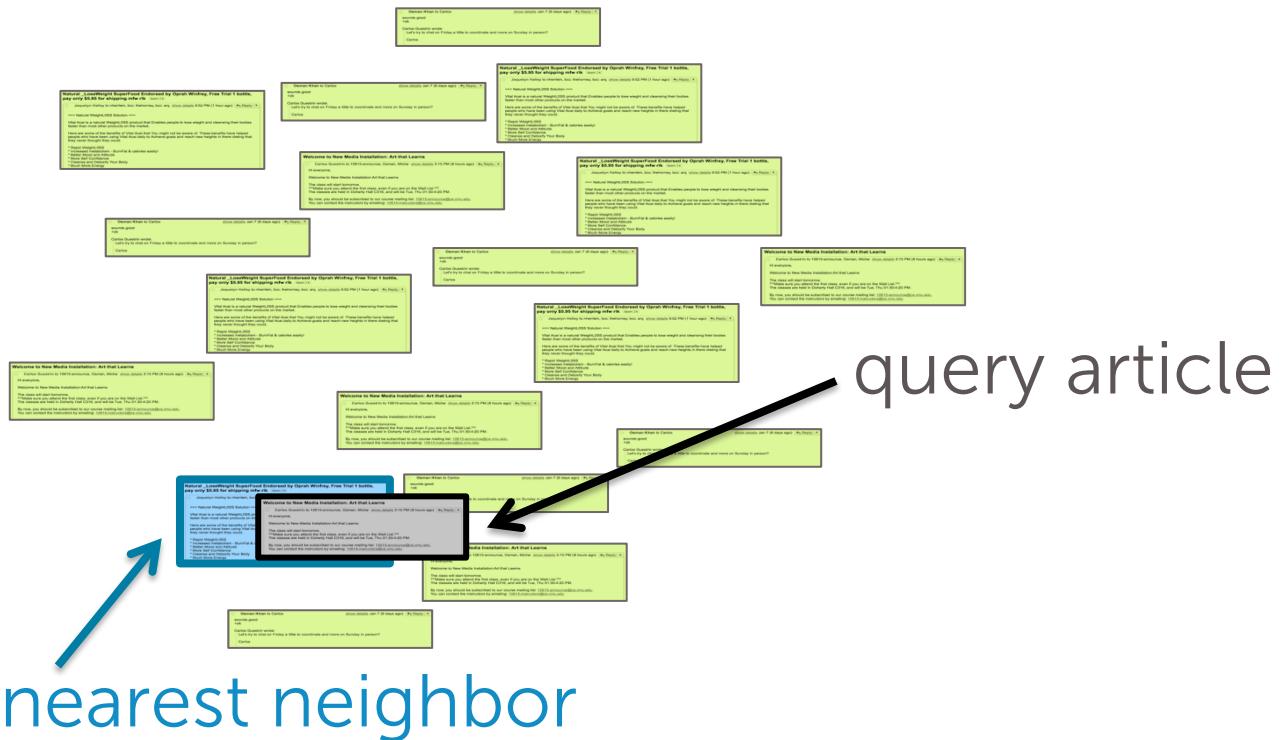
Compute distances to all docs

Space of all articles,
organized by similarity of text



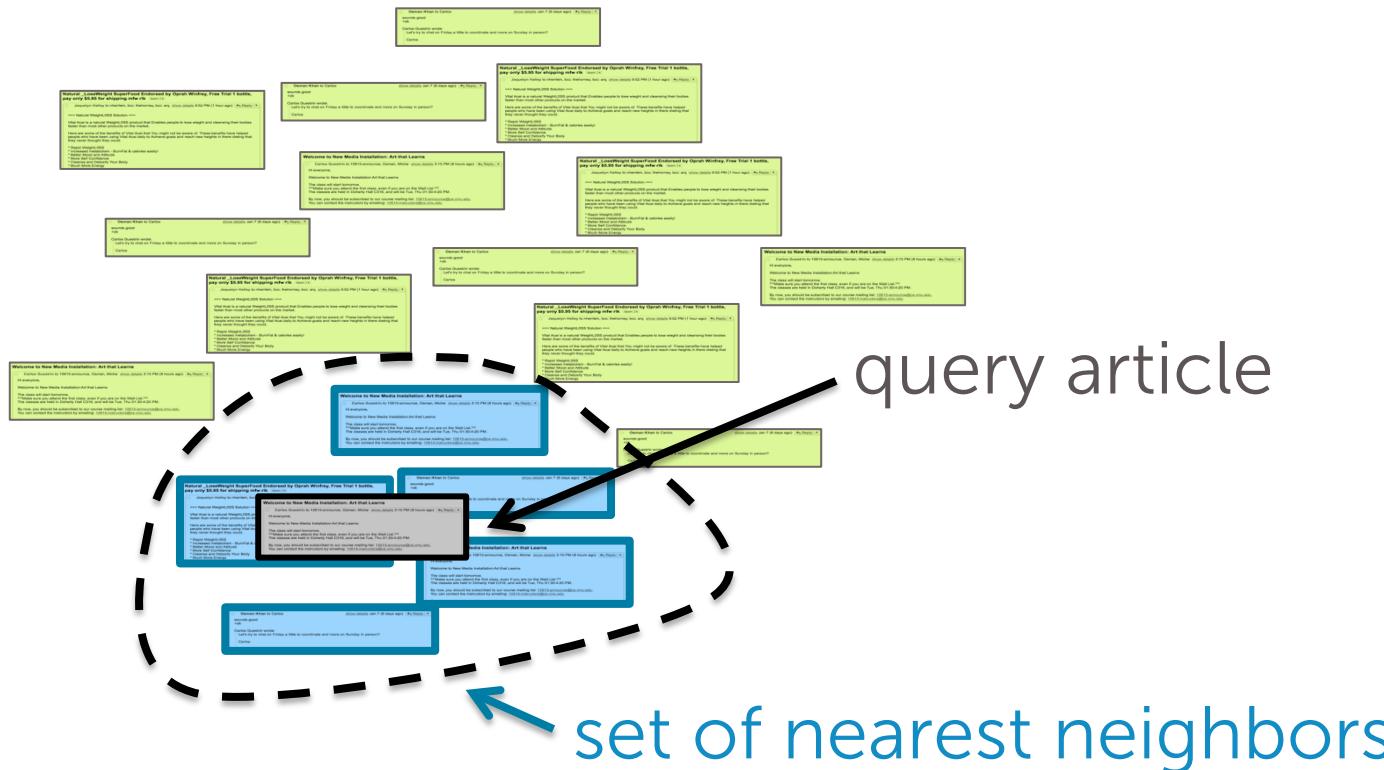
Retrieve “nearest neighbor”

Space of all articles,
organized by similarity of text



Or set of nearest neighbors

Space of all articles,
organized by similarity of text



1-NN algorithm

1 – Nearest neighbor

- **Input:** Query article  : $\underline{\mathbf{x}}_q$
Corpus of documents  $(N \text{ docs})$
- **Output:** *Most* similar article  $\leftarrow \mathbf{x}^{NN}$

Formally:

$$\mathbf{x}^{NN} = \min_{\mathbf{x}_i} \text{distance}(\mathbf{x}_q, \mathbf{x}_i)$$

1-NN algorithm

Initialize $\text{Dist2NN} = \underline{\infty}$, $= \emptyset$

For $i=1,2,\dots,N$

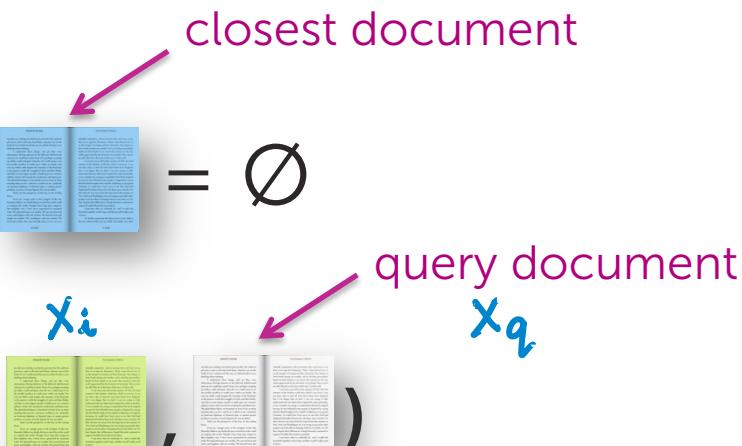
Compute: $\delta = \text{distance}(x_i, x_q)$

If $\delta < \text{Dist2NN}$

set $x_i =$

set $\text{Dist2NN} = \delta$

Return most similar document



k-NN algorithm

k – Nearest neighbor

- **Input:** Query article : \mathbf{x}_q



Corpus of documents



: $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N$

- **Output:** *List of k similar articles*



Formally:

$$X^{NN} = \{x^{NN_1}, \dots, x^{NN_k}\}$$

for all x_i not in X^{NN} , $\text{distance}(x_i, x_q) \geq \max_{x^{NN_j}, j=1..k} \text{distance}(x^{NN_j}, x_q)$

k-NN algorithm

Initialize $\text{Dist2kNN} = \text{sort}(\delta_1, \dots, \delta_k)$ ← list of sorted distances
=  $\text{sort}(\text{articles}, \dots, \text{article}_1)$  $\text{list of sorted docs}$

For $i=k+1, \dots, N$

Compute: $\delta = \text{distance}(\text{article}_i, \text{query doc}_q)$

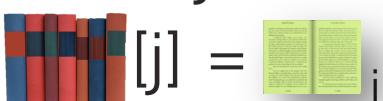
If $\delta < \text{Dist2kNN}[k]$ ← distance to k^{th} NN (furthest NN in set)

find j such that $\delta > \text{Dist2kNN}[j-1]$ but $\delta < \text{Dist2kNN}[j]$

remove furthest house and shift queue:

 $[1:k] = \underline{\text{stack of books}}[1:k-1]$

$\text{Dist2kNN}[j+1:k] = \text{Dist2kNN}[j:k-1]$

set $\text{Dist2kNN}[j] = \delta$ and  $[j] = \text{article}_i$ closest k docs to query doc

Return k most similar articles



Critical elements of NN search

Item (e.g., doc) representation

$$\mathbf{x}_q \leftarrow$$



Measure of distance between items:

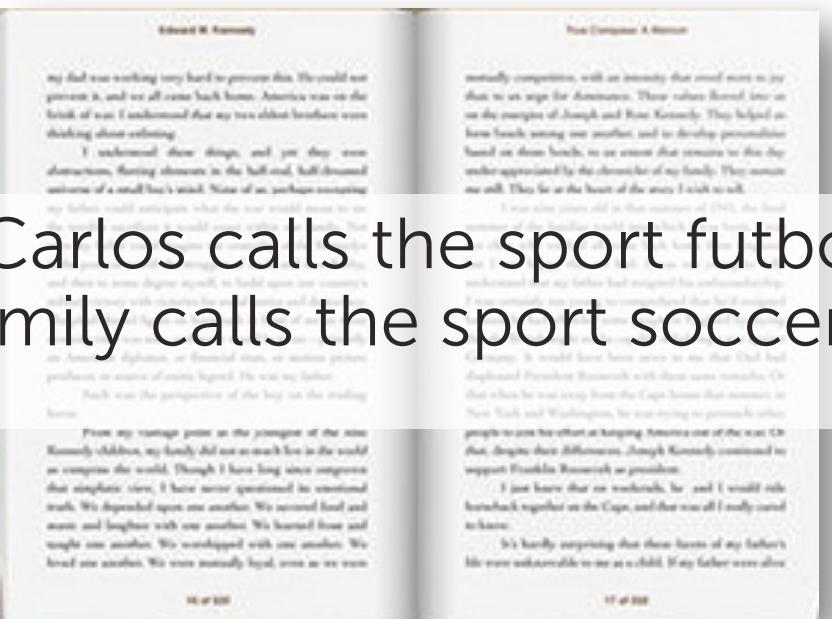
$$\delta = \text{distance}(\mathbf{x}_i, \mathbf{x}_q)$$

Document representation

Word count document representation

Bag of words model

- Ignore order of words
- Count # of instances of each word in vocabulary



"Carlos calls the sport futbol.
Emily calls the sport soccer."

Issues with word counts – Rare words



Common words in doc: “the”, “player”, “field”, “goal”

Dominate rare words like: "futbol", "Messi"

TF-IDF document representation

Emphasizes **important words**

- Appears frequently in document (**common locally**)
- Appears rarely in corpus (**rare globally**)

TF-IDF document representation

Emphasizes **important words**

- Appears frequently in document (**common locally**)

Term frequency =  word counts

- Appears rarely in corpus (**rare globally**)



TF-IDF document representation

Emphasizes **important words**

- Appears frequently in document (**common locally**)

Term frequency = 

- Appears rarely in corpus (**rare globally**)

Inverse doc freq. = $\log \frac{\# \text{ docs}}{1 + \# \text{ docs using word}}$



TF-IDF document representation

Emphasizes **important words**

- Appears frequently in document (**common locally**)

Term frequency =  word counts

- Appears rarely in corpus (**rare globally**)

Inverse doc freq. = $\log \frac{\# \text{ docs}}{1 + \# \text{ docs using word}}$



Trade off: **local frequency vs. global rarity**

tf * idf

Distance metrics

Distance metrics: Defining notion of “closest”

In 1D, just Euclidean distance:

$$\text{distance}(x_i, x_q) = |x_i - x_q|$$

In multiple dimensions:

- can define many interesting distance functions
- most straightforwardly, might want to weight different dimensions differently

Weighting different features

Reasons:

- Some features are more relevant than others



bedrooms
bathrooms
sq.ft. living
sq.ft. lot
floors
year built
year renovated
waterfront



Weighting different features

Reasons:

- Some features are more relevant than others



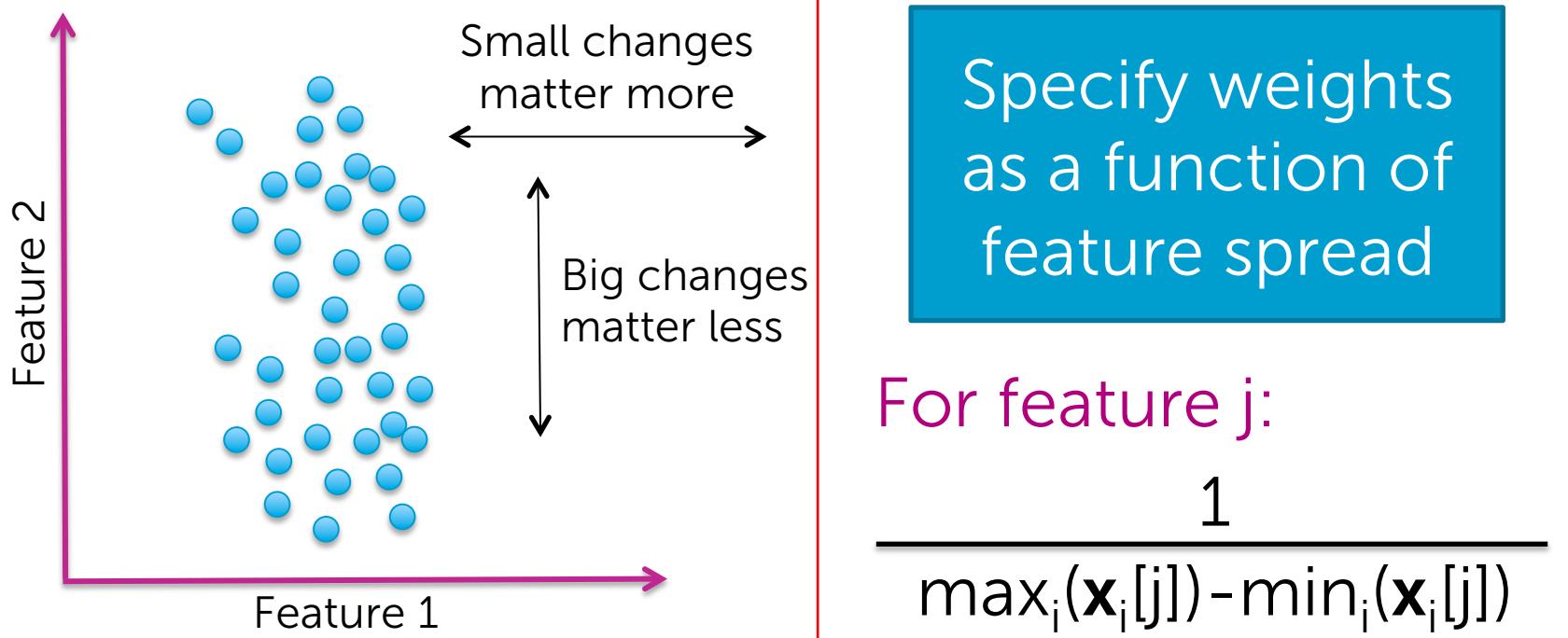
title
abstract
main body
conclusion



Weighting different features

Reasons:

- Some features are more relevant than others
- Some features vary more than others



Scaled Euclidean distance

Formally, this is achieved via

$$\text{distance}(\mathbf{x}_i, \mathbf{x}_q) = \sqrt{a_1(\mathbf{x}_i[1]-\mathbf{x}_q[1])^2 + \dots + a_d(\mathbf{x}_i[d]-\mathbf{x}_q[d])^2}$$

weight on each feature
(defining relative importance)

Effect of binary weights

$$\text{distance}(\mathbf{x}_i, \mathbf{x}_q) = \sqrt{a_1(\mathbf{x}_i[1]-\mathbf{x}_q[1])^2 + \dots + a_d(\mathbf{x}_i[d]-\mathbf{x}_q[d])^2}$$

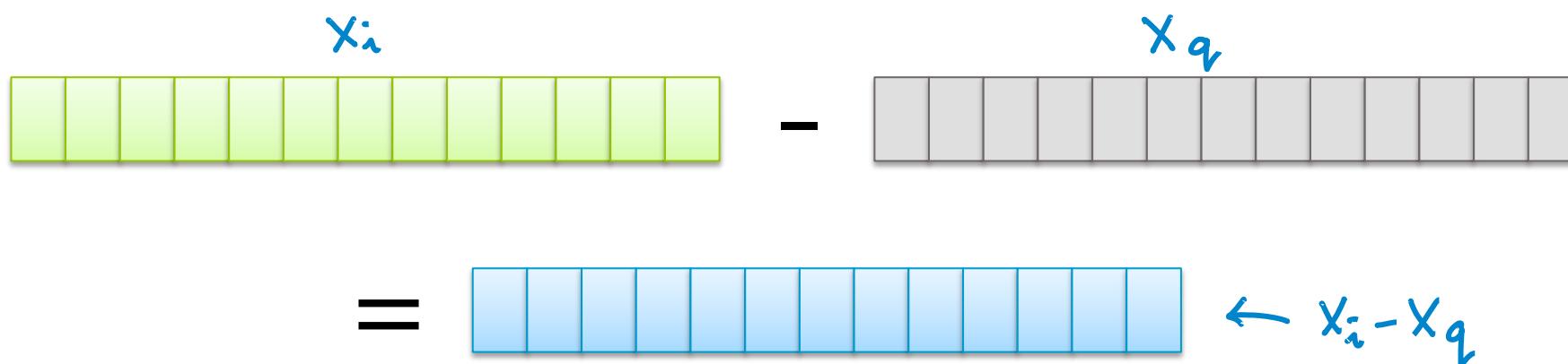
Setting weights as 0 or 1
is equivalent to
feature selection

Feature engineering/
selection is
important, but hard

(non-scaled) Euclidean distance

Defined in terms of inner product

$$\text{distance}(\mathbf{x}_i, \mathbf{x}_q) = \sqrt{(\mathbf{x}_i - \mathbf{x}_q)^T (\mathbf{x}_i - \mathbf{x}_q)}$$
$$\sqrt{(\mathbf{x}_i[1] - \mathbf{x}_q[1])^2 + \dots + (\mathbf{x}_i[d] - \mathbf{x}_q[d])^2}$$



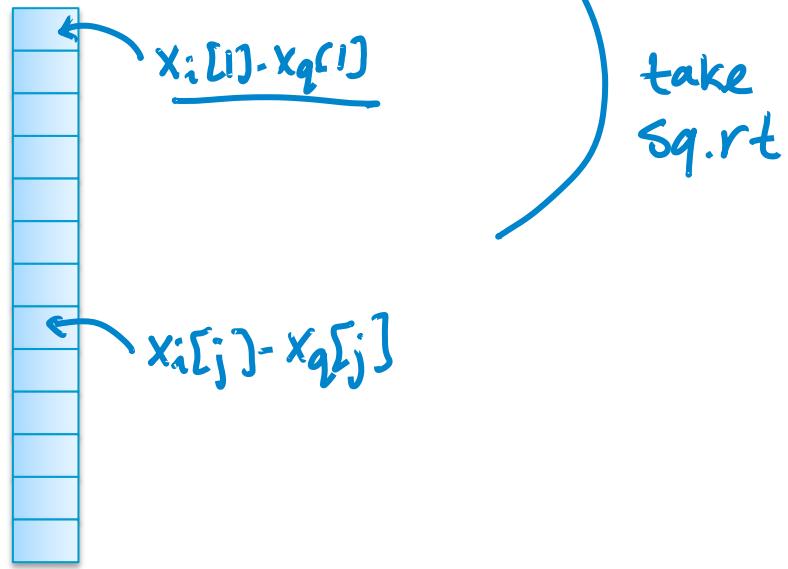
(non-scaled) Euclidean distance

Defined in terms of inner product

$$\text{distance}(\mathbf{x}_i, \mathbf{x}_q) = \sqrt{(\mathbf{x}_i - \mathbf{x}_q)^T (\mathbf{x}_i - \mathbf{x}_q)}$$

$$\sqrt{(\mathbf{x}_i[1]-\mathbf{x}_q[1])^2 + \dots + (\mathbf{x}_i[d]-\mathbf{x}_q[d])^2}$$

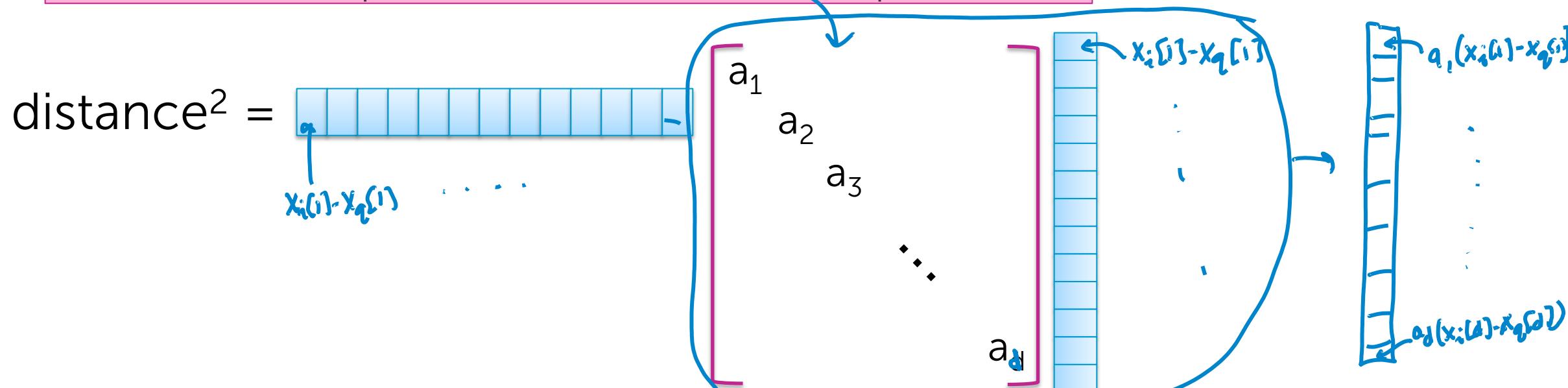
$$\text{distance}^2 = \sqrt{(x_i[j] - x_q[i])^2 + (x_i[j] - x_q[j])^2}$$



Scaled Euclidean distance

Defined in terms of inner product

$$\text{distance}(\mathbf{x}_i, \mathbf{x}_q) = \sqrt{(\mathbf{x}_i - \mathbf{x}_q)^T \mathbf{A} (\mathbf{x}_i - \mathbf{x}_q)}$$
$$= \sqrt{a_1(x_i[1]-x_q[1])^2 + \dots + a_d(x_i[d]-x_q[d])^2}$$



Another natural inner product measure



Xq



1



Similarity

$$= \mathbf{x}_i^T \mathbf{x}_q$$

$$= \sum_{j=1}^d \mathbf{x}_i[j] \mathbf{x}_q[j]$$

$$= 13$$

Another natural inner product measure

Edward W. Kennedy
my dad was working very hard to prevent that. He could not
go home, and we all came back home. America was on the
hook for us. We had to go to school, and we had to go to work.
Shaking their shoulders.

From my range point of view the things, and yet they were
distracting. Playing chess in the hall and half-dressed
and the like. I have never questioned what the war would move to
my father could imagine what the war would move to
the terrible sacrifice it would cost us one family. But
we are the ones who are the ones who are the ones who are
in the present world the strength of each, and then family,
military victory with elements for social justice and democracy.

The placed dead figure on Standard in front of us at three
memorials. We have to go to school, and we have to go to work
as American diplomats, or financial firms, or marine products
producing companies. We have to go to school, and we are now
back to the perspective of the boy on the reading
books.

From my range point of view the perspective of the nine
Family children, we did not do much for the world
in comparison. This is the reason why I have never
questioned what the war would move to
our people's views. I have never questioned in contrast
with the like. Playing chess in the hall and half-dressed
and the like. We learned from and
taught one another. We worked with one another. We
had one another. We were really good at what we are now

10 of 100
11 of 100



Similarity
= 0



Edward W. Kennedy
my dad was working very hard to prevent this. He could not
go home, and we all came back home. America was on the
hook for us. We had to go to school, and we had to go to work.
Shaking their shoulders.

From my range point of view the things, and yet they were
distracting. Playing chess in the hall and half-dressed
and the like. I have never questioned what the war would move to
my father could imagine what the war would move to
the terrible sacrifice it would cost us one family. But
we are the ones who are the ones who are the ones who are
in the present world the strength of each, and then family,
military victory with elements for social justice and democracy.

The placed dead figure on Standard in front of us at three
memorials. We have to go to school, and we have to go to work
as American diplomats, or financial firms, or marine products
producing companies. We have to go to school, and we are now
back to the perspective of the boy on the reading
books.

From my range point of view the perspective of the nine
Family children, we did not do much for the world
in comparison. This is the reason why I have never
questioned what the war would move to
our people's views. I have never questioned in contrast
with the like. Playing chess in the hall and half-dressed
and the like. We learned from and
taught one another. We worked with one another. We
had one another. We were really good at what we are now

10 of 100
11 of 100



Cosine similarity – normalize

Similarity =

$$\frac{\sum_{j=1}^d \mathbf{x}_i[j] \mathbf{x}_q[j]}{\sqrt{\sum_{j=1}^d (\mathbf{x}_i[j])^2} \sqrt{\sum_{j=1}^d (\mathbf{x}_q[j])^2}}$$

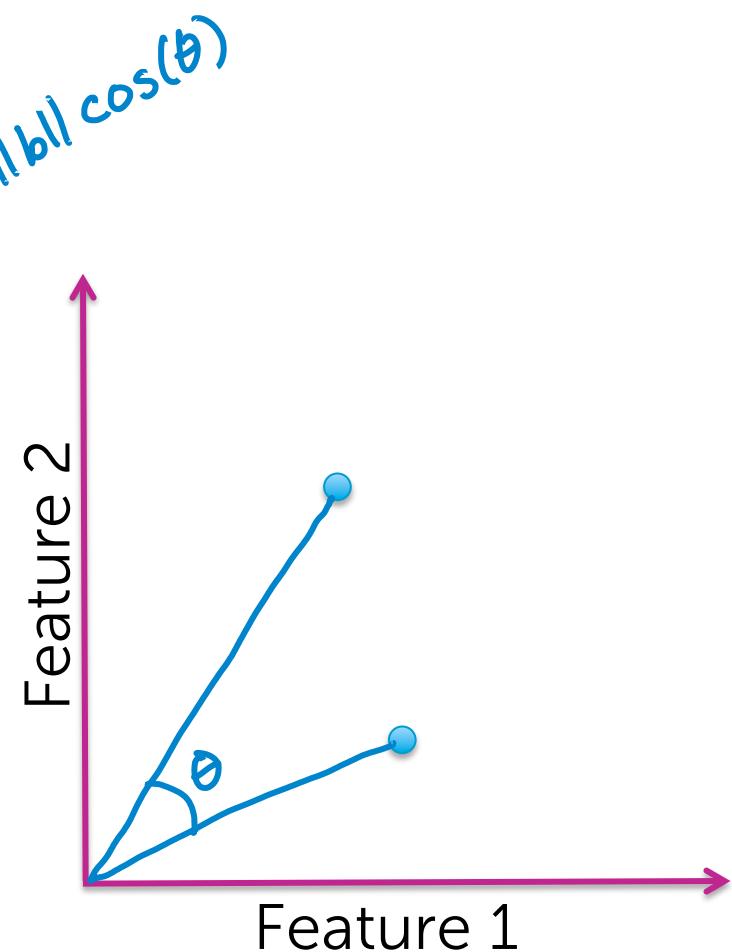
$$\mathbf{x}_i^\top \mathbf{x}_q = \cos(\theta)$$

$$\|\mathbf{x}_i\| \|\mathbf{x}_q\|$$

$$= \left(\frac{\mathbf{x}_i}{\|\mathbf{x}_i\|} \right)^\top \left(\frac{\mathbf{x}_q}{\|\mathbf{x}_q\|} \right)$$

first normalize

- Not a proper distance metric
- Efficient to compute for sparse vecs



Normalize



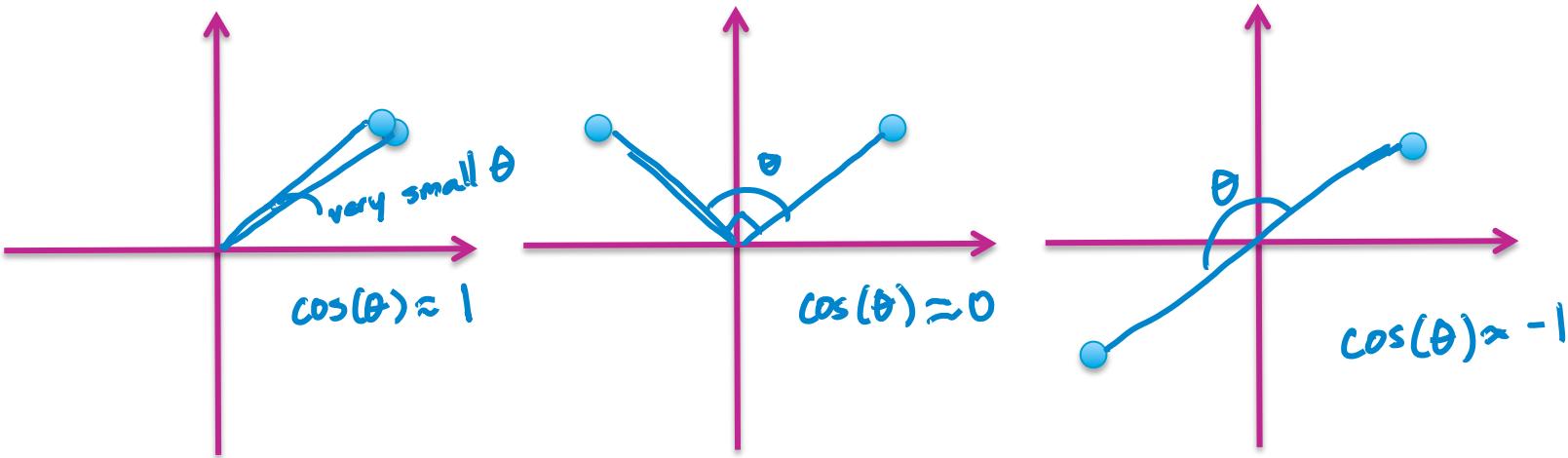
1	0	0	0	5	3	0	0	1	0	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---

$$\sqrt{1^2 + 5^2 + 3^2 + 1^2} \quad \leftarrow \|x_i\| = \sum_{j=1}^d x_{i[j]}^2$$

$\leftarrow x_i$

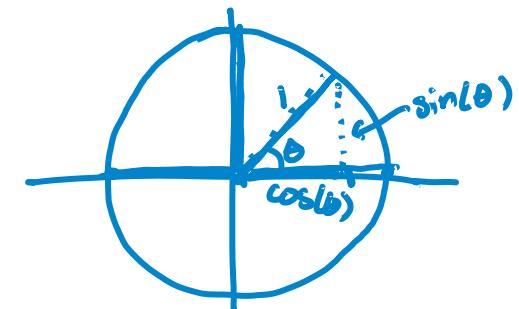
1					5	3		1				
/	0	0	0	/	/	0	0	/	0	0	0	0
6				6	6		6					

Cosine similarity



In general, $-1 < \text{similarity} < 1$

For positive features (like tf-idf)
 $-1 < \text{similarity} < 1$



Define **distance** = $1 - \text{similarity}$

To normalize or not?

1	0	0	0	5	3	0	0	1	0	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---

3	1	0	0	2	0	0	1	0	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---

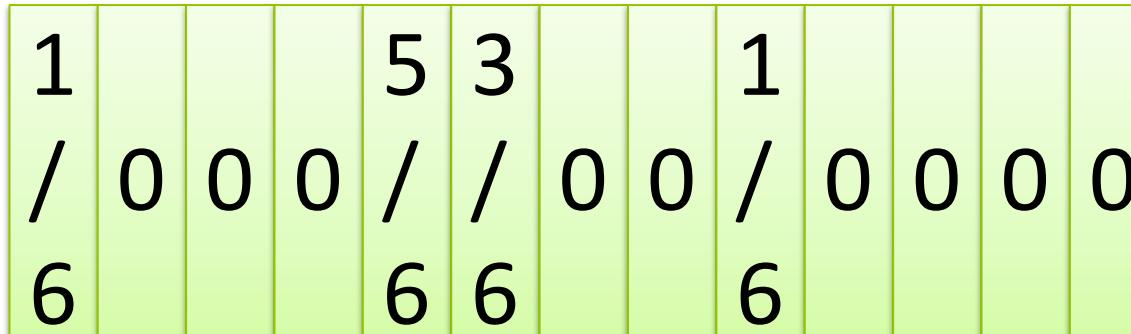
Similarity = 13

2	0	0	0	10	6	0	0	2	0	0	0	0
---	---	---	---	----	---	---	---	---	---	---	---	---

Normalize



$$\sqrt{1^2 + 5^2 + 3^2 + 1^2}$$



In the normalized case

1				5	3			1					
/	0	0	0	/	/	0	0	/	0	0	0	0	0
6				6	6			6					

3	1	0	0	1	/	0	0	1	/	0	1	0	0	0
/	/	0	0	/	0	0	/	4	0	/	4	0	0	0
4	4			2				4			4			

Similarity = 13/24

Edward W. Kennedy	Tom Cruise & Mission	Edward W. Kennedy	Tom Cruise & Mission
<p>my dad was working very hard to prevent this. He could not possibly have been more involved in the politics of the world at the time. I think he was involved in the kind of work that involved all my other brothers because they thought about politics.</p>	<p>— And you? These things, and yet you were abstaining, throwing darts in the ball room, I feel I should mention that you were abstaining from politics. My father could anticipate what the world would want to see if the world sensible would be most wise over our family. You might say that he was right, but I think he was wrong in the process while the struggle of, and then, Belden and the rest of us, were right. We were right in our political activity with respect to social justice and democracy.</p>	<p>The plot about Gruen is based on it. In fact, as one of those meetings, I was in the room with my dad, and we were discussing American diplomats, or financial, or some other problem, and he turned to me and said, "Well, you know, Tom, back then, the perspective of the guy on the trading floor."</p>	<p>From my perspective as the younger of the two Kennedy children, my family had no real link to the world of politics. I mean, we were involved in politics, but not in the sense that I have never participated in national politics. We had a brother with one son. We had a son and one daughter with another son. We had a son and one daughter. We worshipped with one another. We lived one another. We really, really loved each other.</p>
<p>— And you? These things, and yet you were abstaining, throwing darts in the ball room, I feel I should mention that you were abstaining from politics. My father could anticipate what the world would want to see if the world sensible would be most wise over our family. You might say that he was right, but I think he was wrong in the process while the struggle of, and then, Belden and the rest of us, were right. We were right in our political activity with respect to social justice and democracy.</p>	<p>The plot about Gruen is based on it. In fact, as one of those meetings, I was in the room with my dad, and we were discussing American diplomats, or financial, or some other problem, and he turned to me and said, "Well, you know, Tom, back then, the perspective of the guy on the trading floor."</p>	<p>— Well, I was working very hard to prevent this. He could not possibly have been more involved in the politics of the world at the time. I think he was involved in the kind of work that involved all my other brothers because they thought about politics.</p>	<p>— And you? These things, and yet you were abstaining, throwing darts in the ball room, I feel I should mention that you were abstaining from politics. My father could anticipate what the world would want to see if the world sensible would be most wise over our family. You might say that he was right, but I think he was wrong in the process while the struggle of, and then, Belden and the rest of us, were right. We were right in our political activity with respect to social justice and democracy.</p>
<p>The plot about Gruen is based on it. In fact, as one of those meetings, I was in the room with my dad, and we were discussing American diplomats, or financial, or some other problem, and he turned to me and said, "Well, you know, Tom, back then, the perspective of the guy on the trading floor."</p>	<p>— Well, I was working very hard to prevent this. He could not possibly have been more involved in the politics of the world at the time. I think he was involved in the kind of work that involved all my other brothers because they thought about politics.</p>	<p>— And you? These things, and yet you were abstaining, throwing darts in the ball room, I feel I should mention that you were abstaining from politics. My father could anticipate what the world would want to see if the world sensible would be most wise over our family. You might say that he was right, but I think he was wrong in the process while the struggle of, and then, Belden and the rest of us, were right. We were right in our political activity with respect to social justice and democracy.</p>	<p>The plot about Gruen is based on it. In fact, as one of those meetings, I was in the room with my dad, and we were discussing American diplomats, or financial, or some other problem, and he turned to me and said, "Well, you know, Tom, back then, the perspective of the guy on the trading floor."</p>
<p>— Well, I was working very hard to prevent this. He could not possibly have been more involved in the politics of the world at the time. I think he was involved in the kind of work that involved all my other brothers because they thought about politics.</p>	<p>— And you? These things, and yet you were abstaining, throwing darts in the ball room, I feel I should mention that you were abstaining from politics. My father could anticipate what the world would want to see if the world sensible would be most wise over our family. You might say that he was right, but I think he was wrong in the process while the struggle of, and then, Belden and the rest of us, were right. We were right in our political activity with respect to social justice and democracy.</p>	<p>— Well, I was working very hard to prevent this. He could not possibly have been more involved in the politics of the world at the time. I think he was involved in the kind of work that involved all my other brothers because they thought about politics.</p>	<p>The plot about Gruen is based on it. In fact, as one of those meetings, I was in the room with my dad, and we were discussing American diplomats, or financial, or some other problem, and he turned to me and said, "Well, you know, Tom, back then, the perspective of the guy on the trading floor."</p>

1				5	3			1					
/	0	0	0	/	/	0	0	/	0	0	0	0	0
6				6	6			6					

3	1			1			1		1				
/	/	0	0	/	0	0	/	0	/	0	0	0	0
4	4			2			4		4				

Similarity = 13/24

But not always desired...

long document

short tweet

long document

Normalizing can make dissimilar objects appear more similar

Common compromise: Just cap maximum word counts

Other distance metrics

- Mahalanobis
- rank-based
- correlation-based
- Manhattan
- Jaccard
- Hamming
- ...

Combining distance metrics

Example of document features:

1. Text of document
 - Distance metric: Cosine similarity
2. # of reads of doc
 - Distance metric: Euclidean distance

Add together with user-specified weights

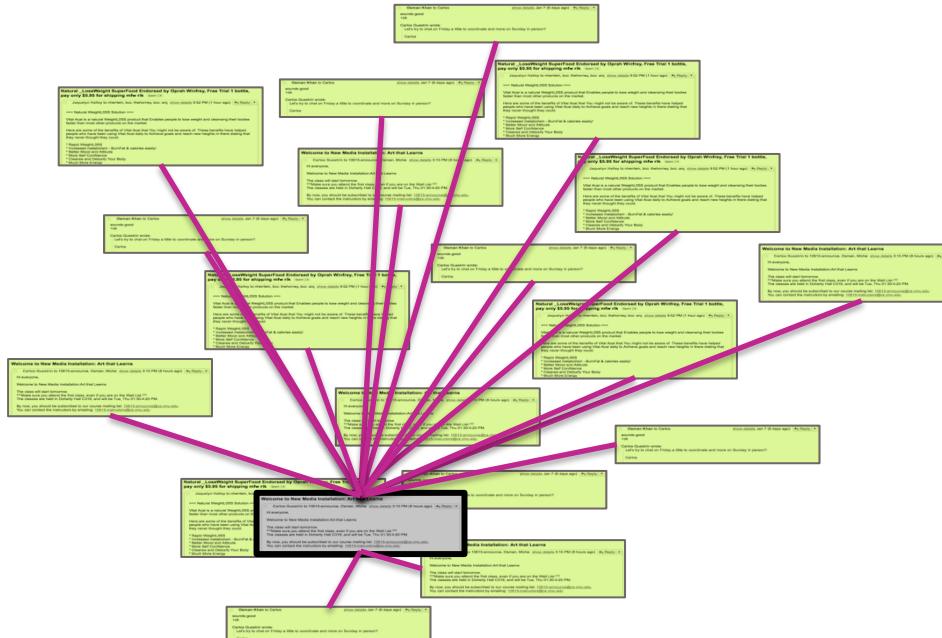
Scaling up k-NN search by storing data in a KD-tree

Complexity of search

Complexity of brute-force search

Given a query point, scan through each point

- $O(N)$ distance computations per 1-NN query!
- $O(N \log k)$ per k -NN query!



What if N is huge???
(and many queries)

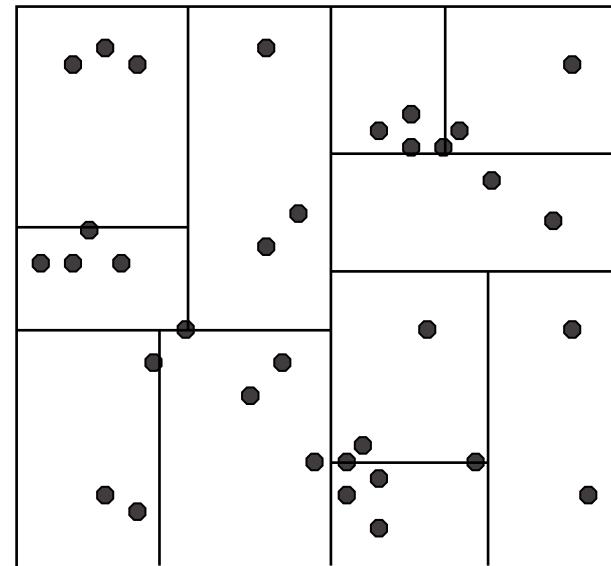
KD-tree representation

KD-trees

Structured organization of documents

- Recursively partitions points into axis aligned boxes.

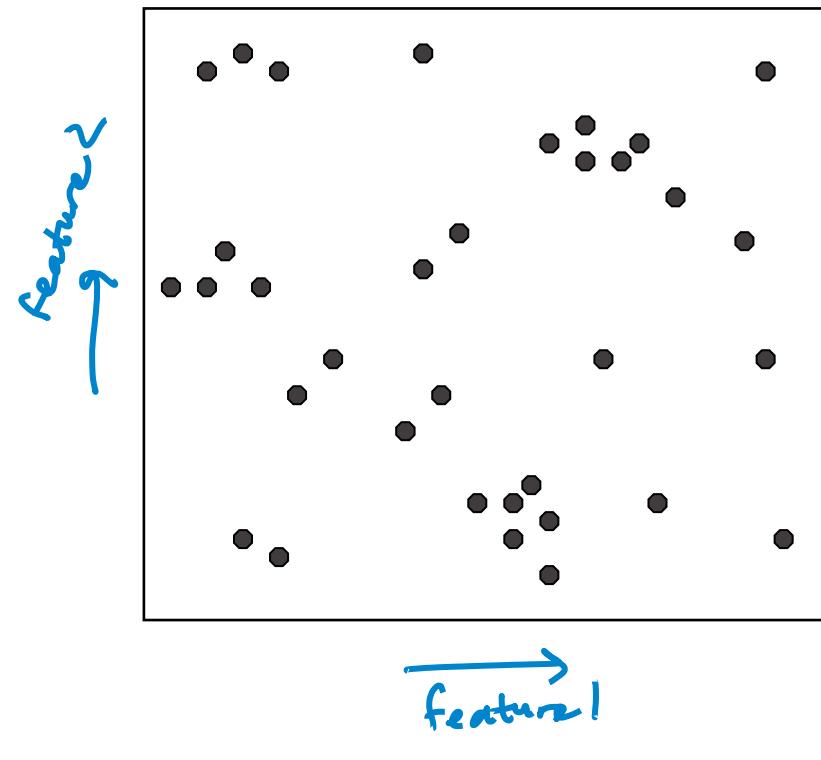
Enables more efficient pruning of search space



Works “well” in “low-medium” dimensions

- We'll get back to this...

KD-tree construction

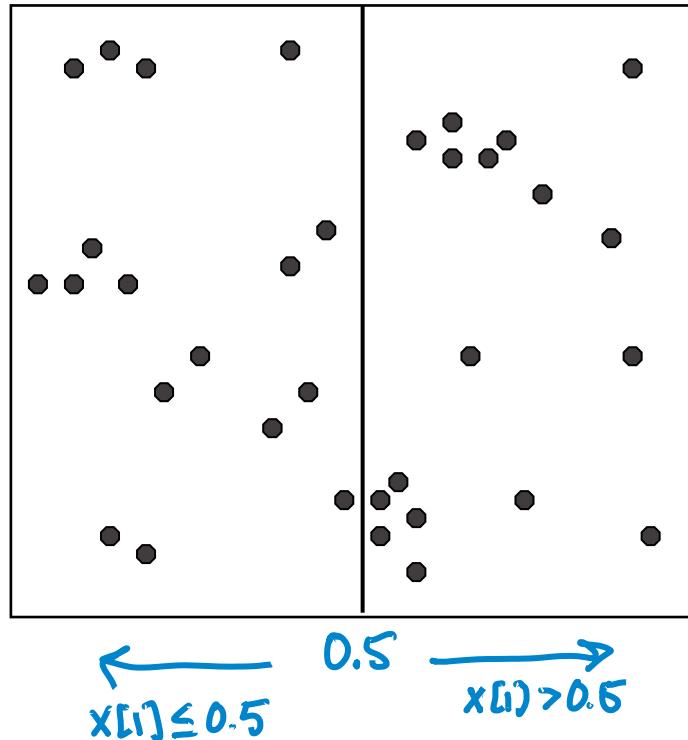


Start with a list of
d-dimensional points.

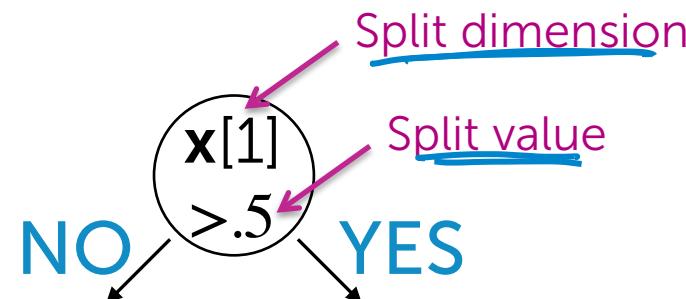
Pt	x[1]	x[2]
1	0.00	0.00
2	1.00	4.31
3	0.13	2.85
...

↑
obs.
indices
↑
feat. 1
(word 1)
↑
Feat. 2
(word 2)

KD-tree construction

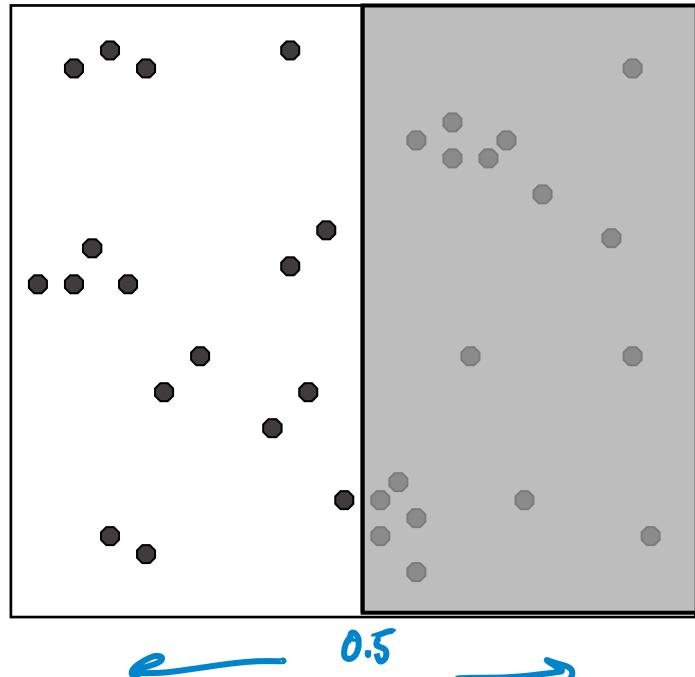


Split points into 2 groups

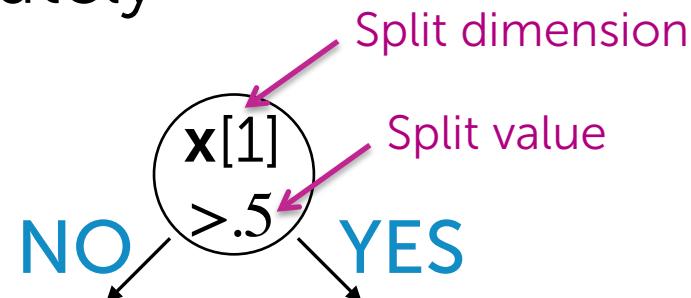


Pt	$x[1]$	$x[2]$	Pt	$x[1]$	$x[2]$
1	0.00	0.00	2	1.00	4.31
3	0.13	2.85
...			

KD-tree construction

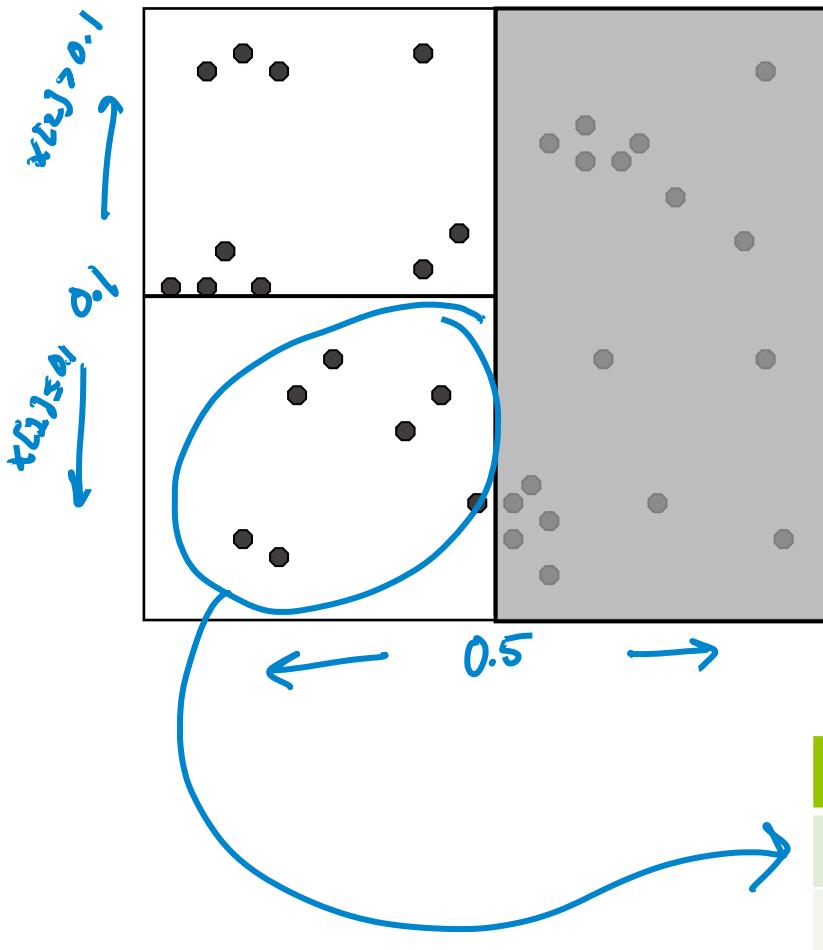


Recurse on each group separately

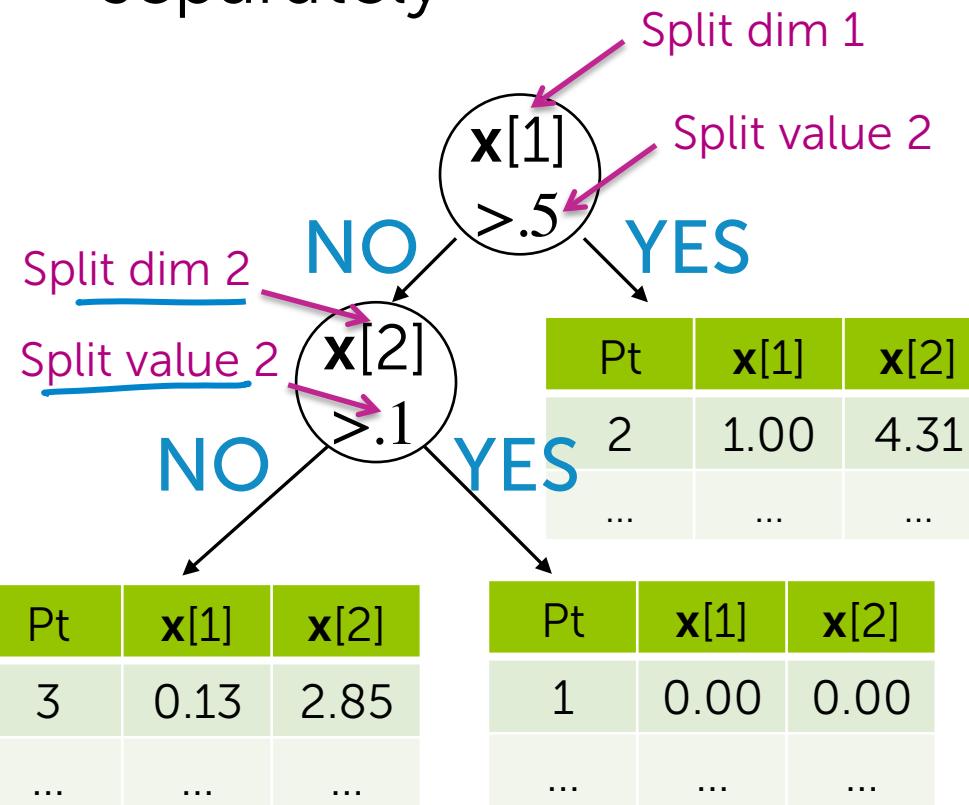


Pt	$x[1]$	$x[2]$	Pt	$x[1]$	$x[2]$
1	0.00	0.00	2	1.00	4.31
3	0.13	2.85
...			

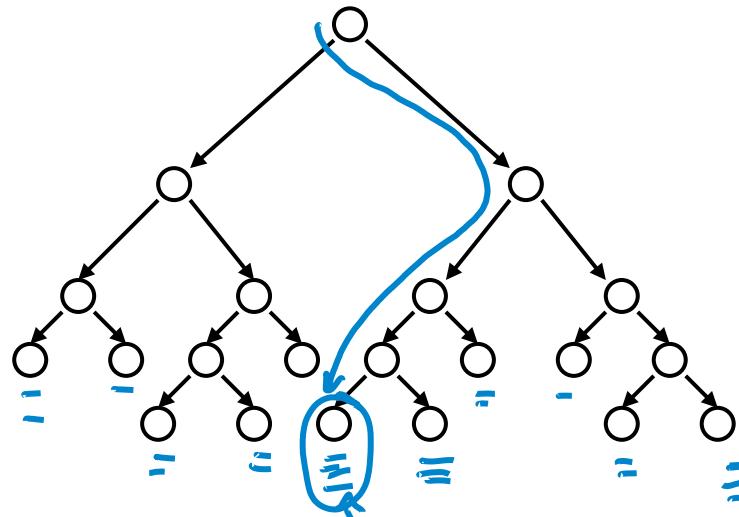
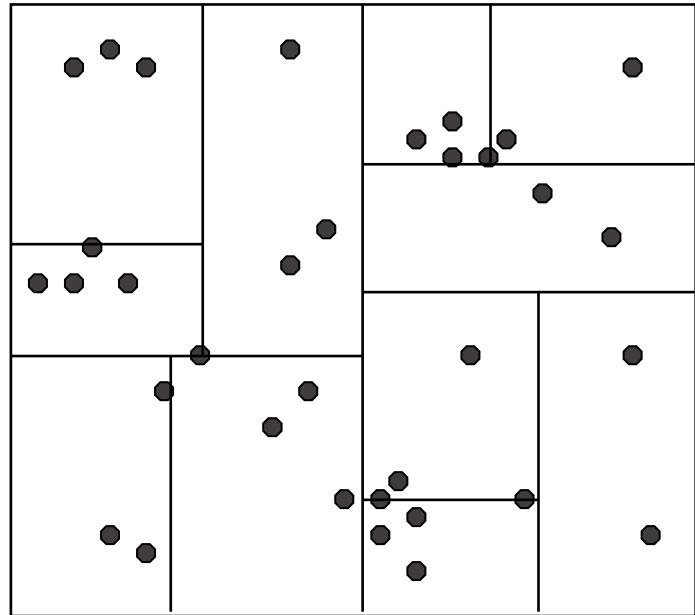
KD-tree construction



Recurse on each group separately



KD-tree construction

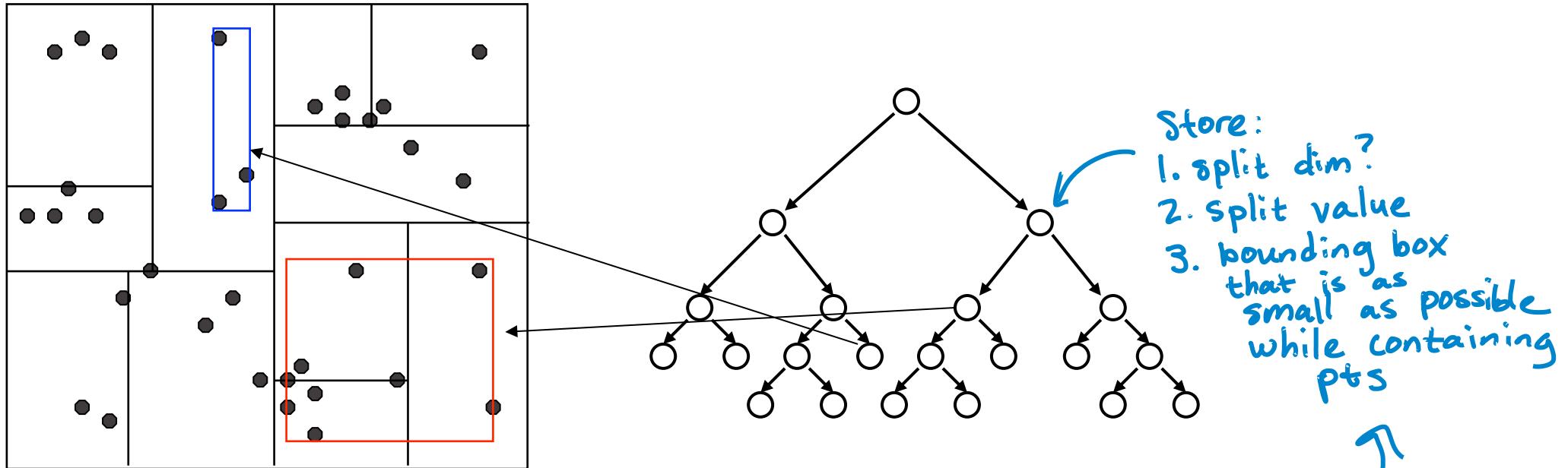


Continue splitting points at each set
- Creates a binary tree structure

points here
satisfy all
conditions down
the tree to
this leaf

Each leaf node contains a list of points

KD-tree construction



Keep one additional piece of info at each node:

#3 - The (tight) bounds of points at or below node

KD-tree construction choices

Use heuristics to make splitting decisions:

- Which dimension do we split along?

widest (or alternate)

- Which value do we split at?

median (or center point of box,
ignoring data in box)

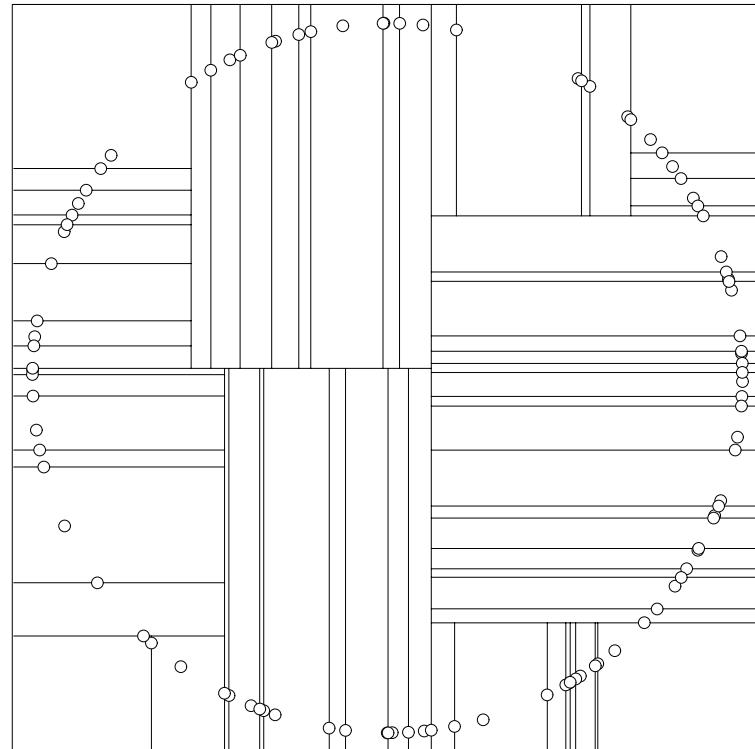
- When do we stop?

fewer than m pts left

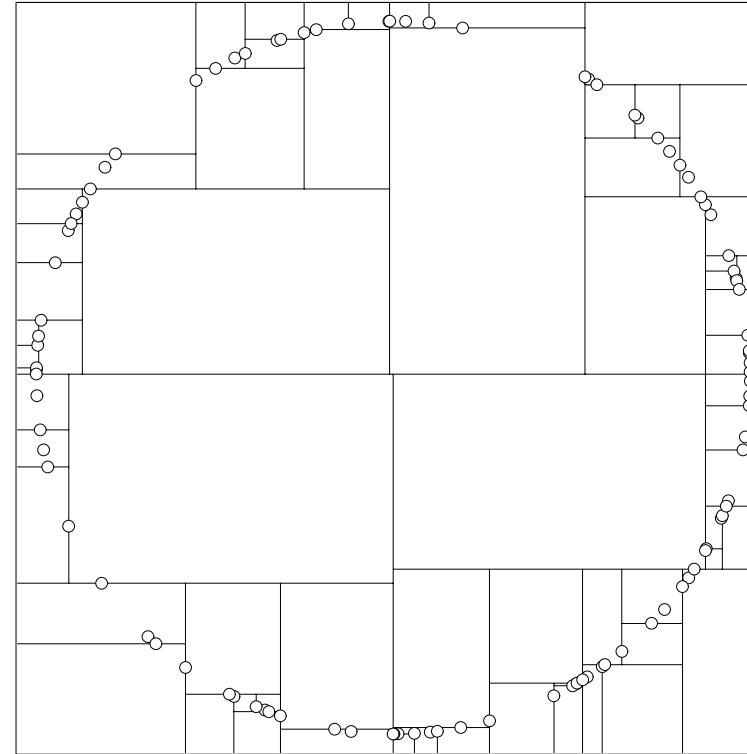
or

box hits minimum width

Many heuristics...



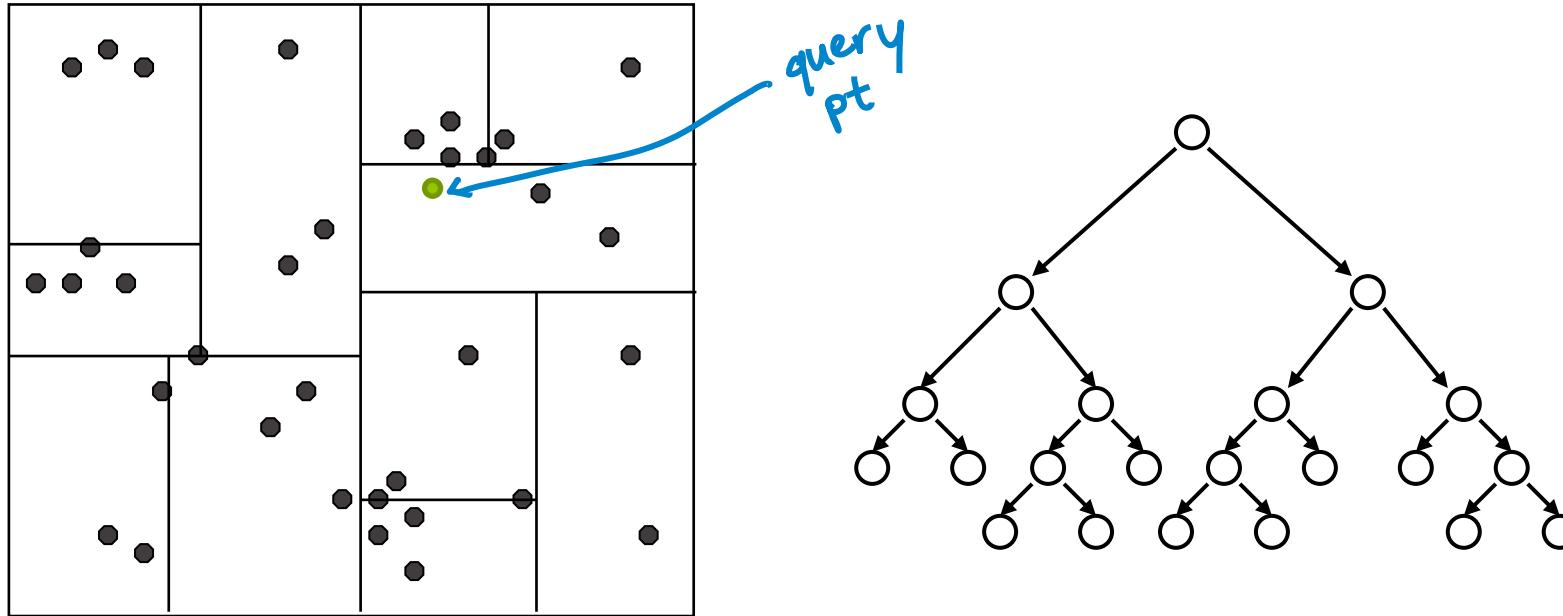
median heuristic



center-of-range
heuristic

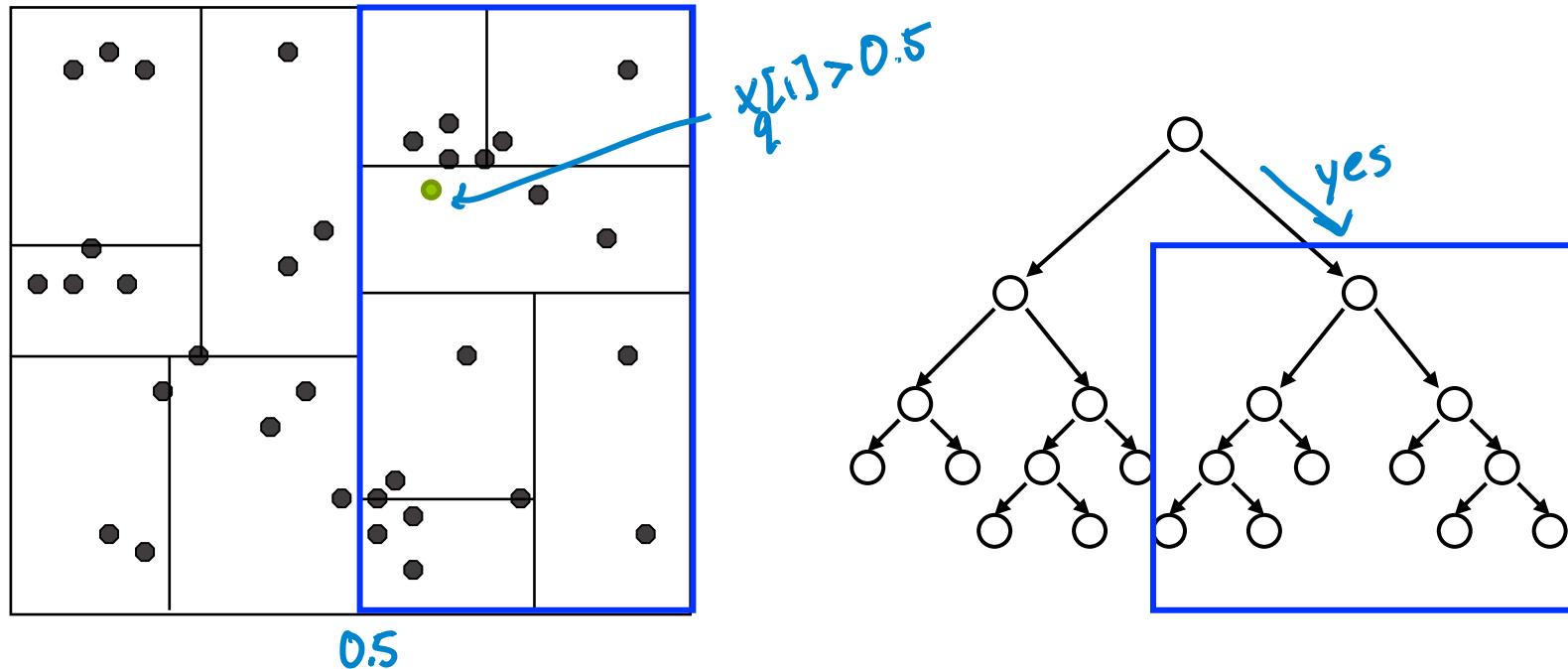
NN search with KD-trees

Nearest neighbor with KD-trees



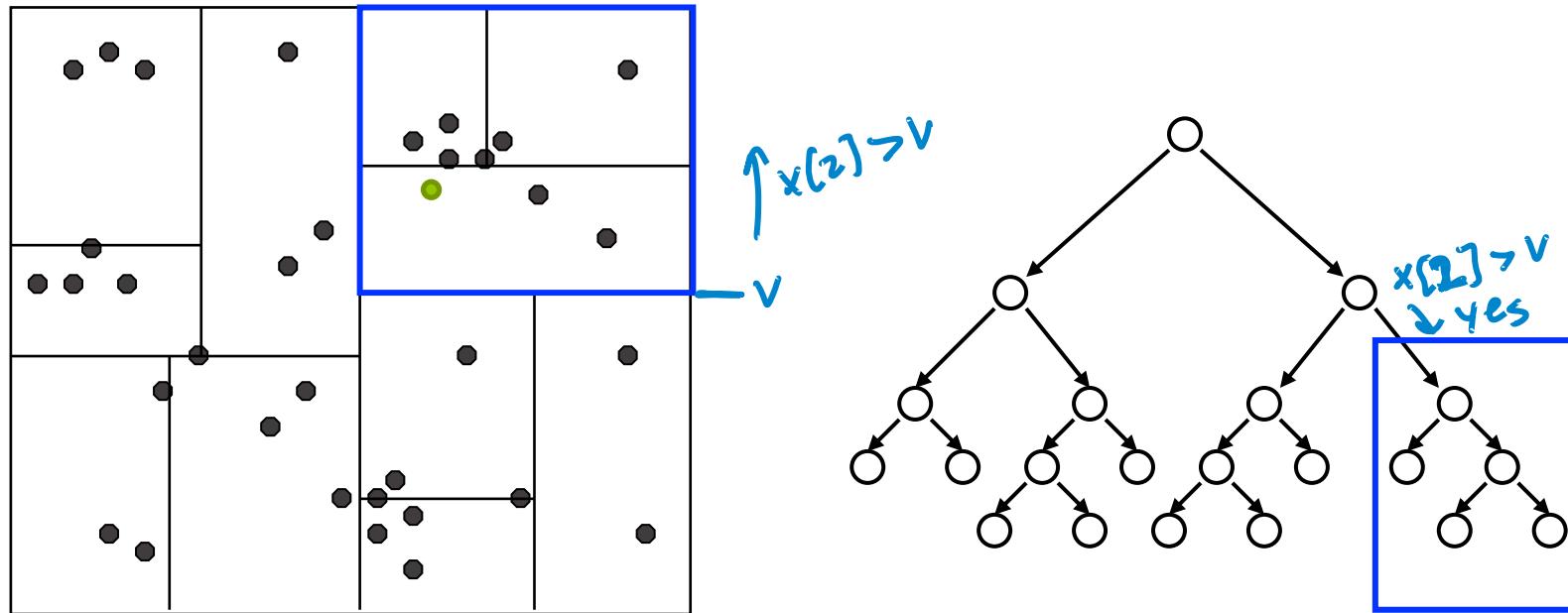
Traverse tree looking for nearest neighbor to query point

Nearest neighbor with KD-trees



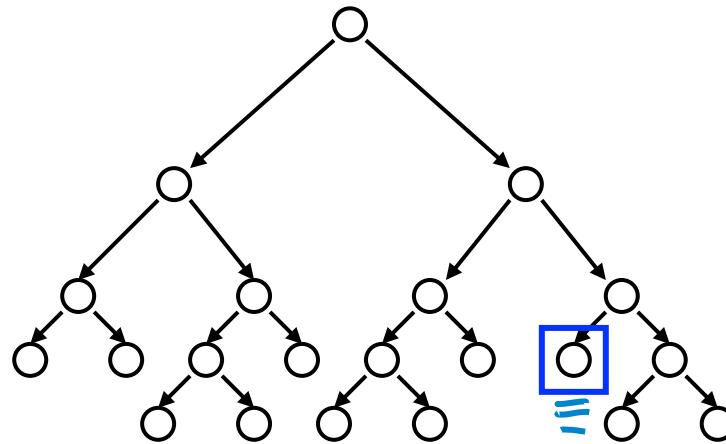
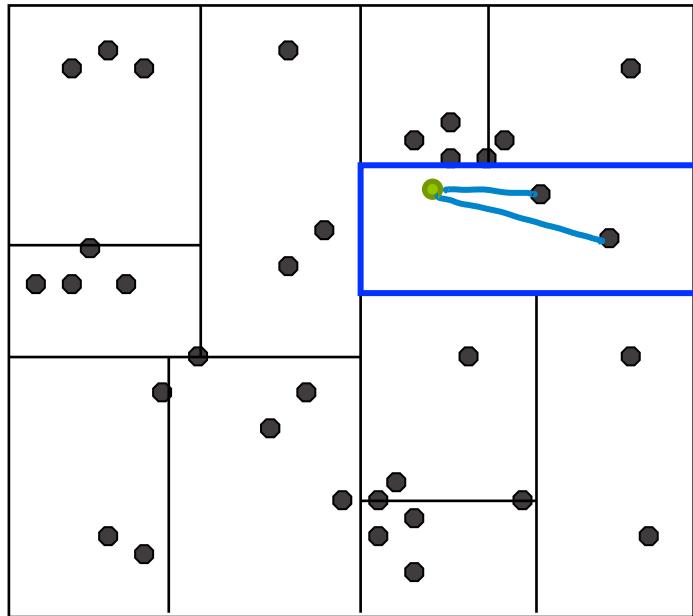
1. Start by exploring leaf node containing query point

Nearest neighbor with KD-trees



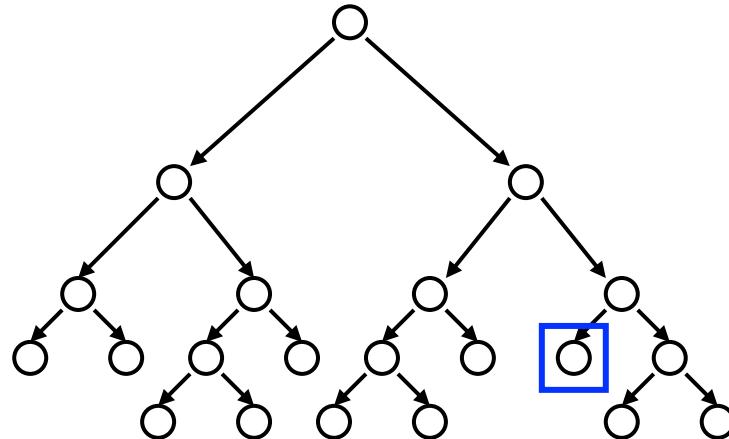
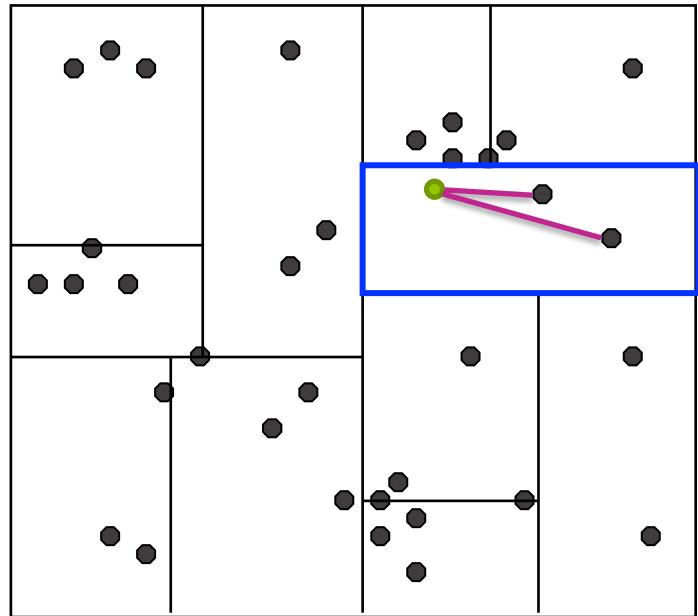
1. Start by exploring leaf node containing query point

Nearest neighbor with KD-trees



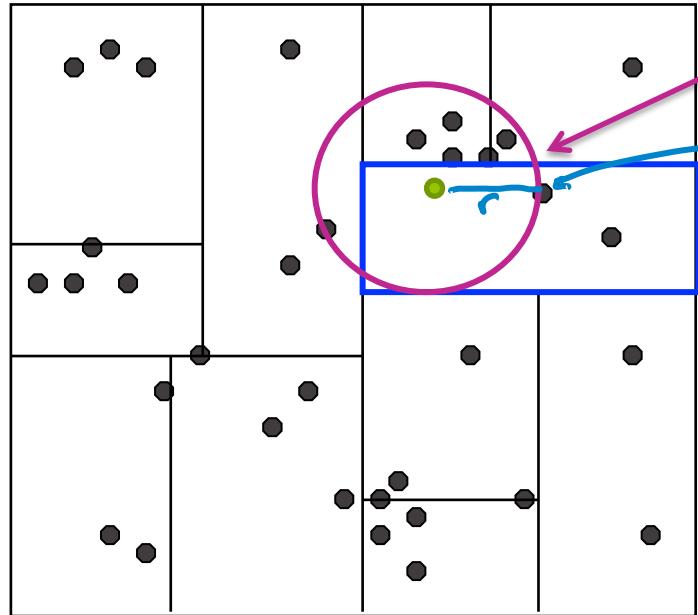
1. Start by exploring leaf node containing query point

Nearest neighbor with KD-trees



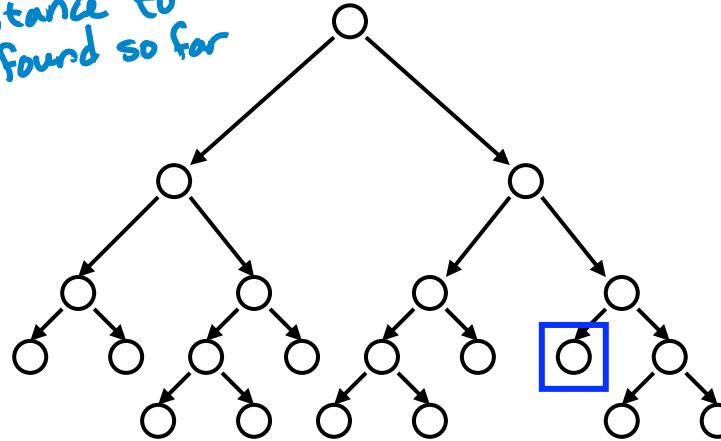
1. Start by exploring leaf node containing query point
2. Compute distance to each other point at leaf node

Nearest neighbor with KD-trees



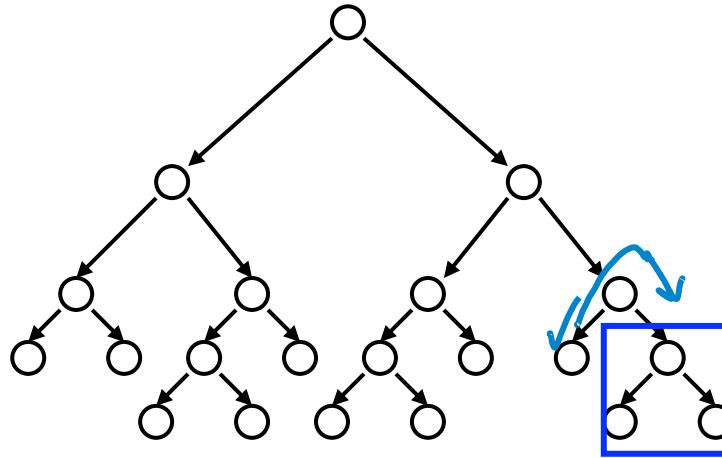
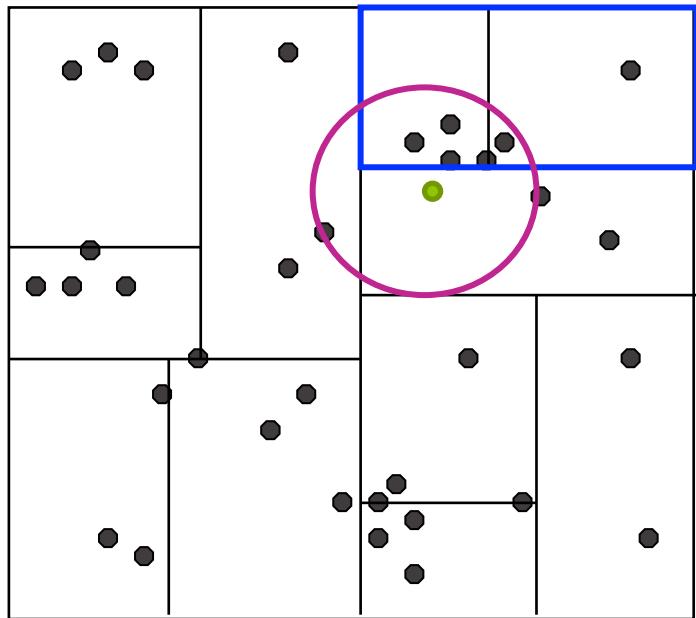
Does nearest neighbor have to live at leaf node containing query point?

distance to
NN found so far



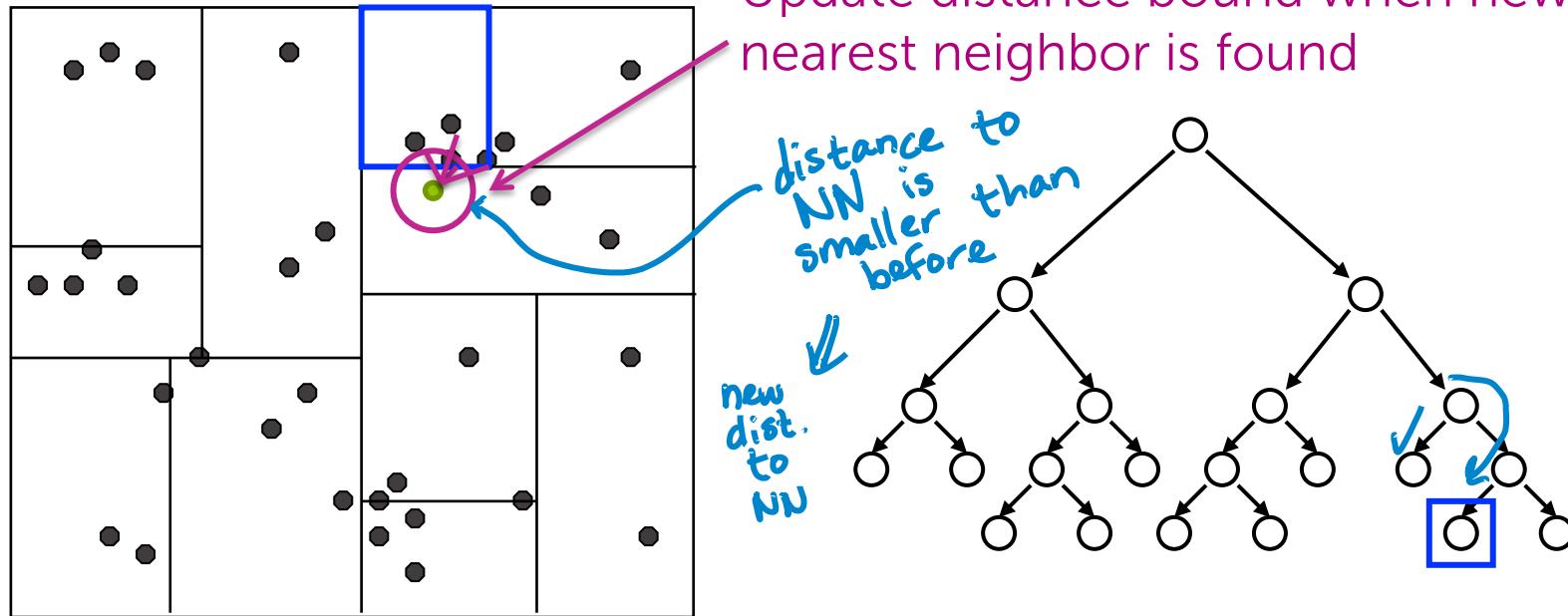
1. Start by exploring leaf node containing query point
2. Compute distance to each other point at leaf node

Nearest neighbor with KD-trees



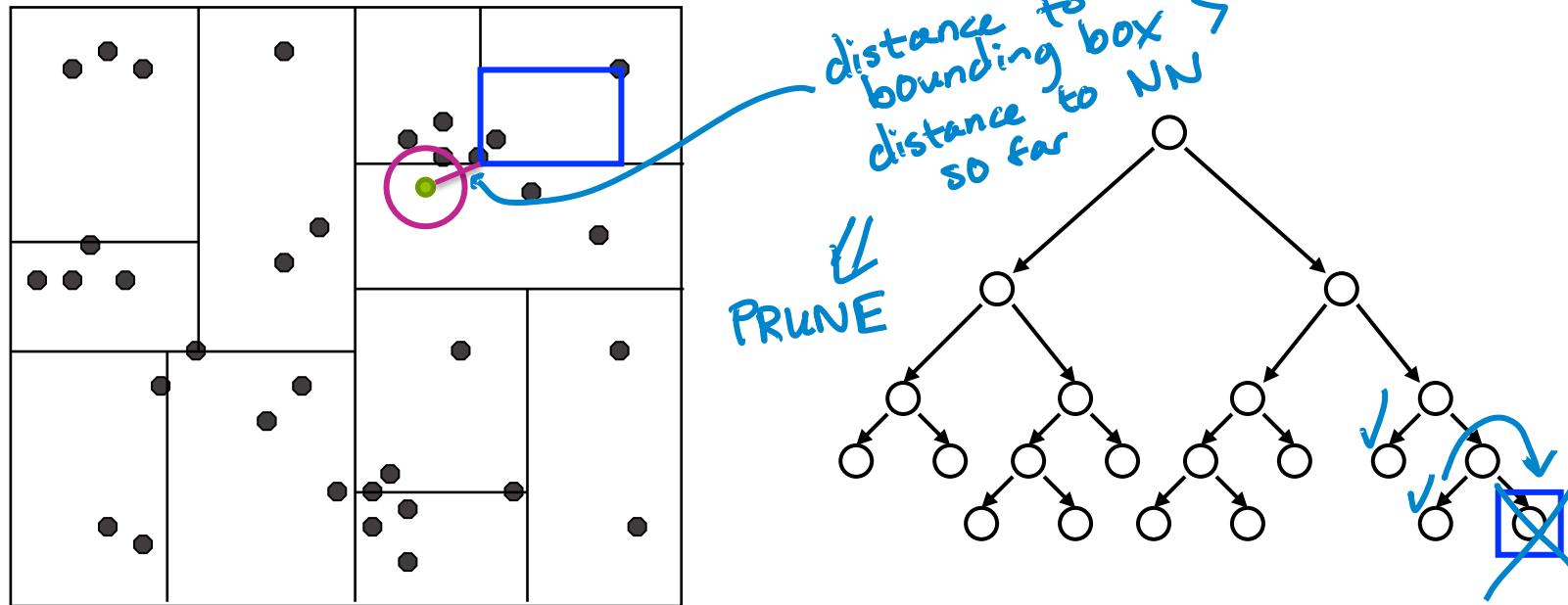
1. Start by exploring leaf node containing query point
2. Compute distance to each other point at leaf node
3. Backtrack and try other branch at each node visited

Nearest neighbor with KD-trees



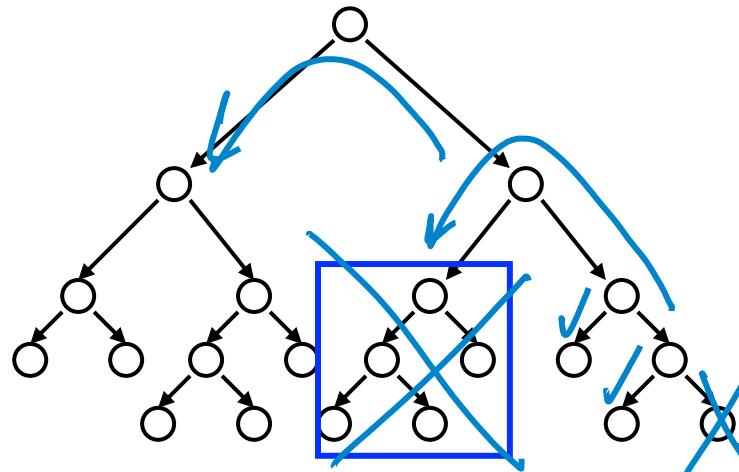
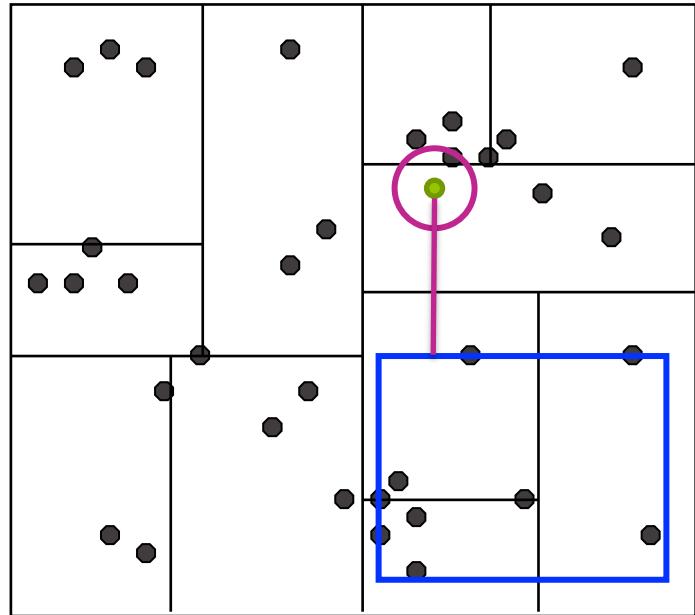
1. Start by exploring leaf node containing query point
2. Compute distance to each other point at leaf node
3. Backtrack and try other branch at each node visited

Nearest neighbor with KD-trees



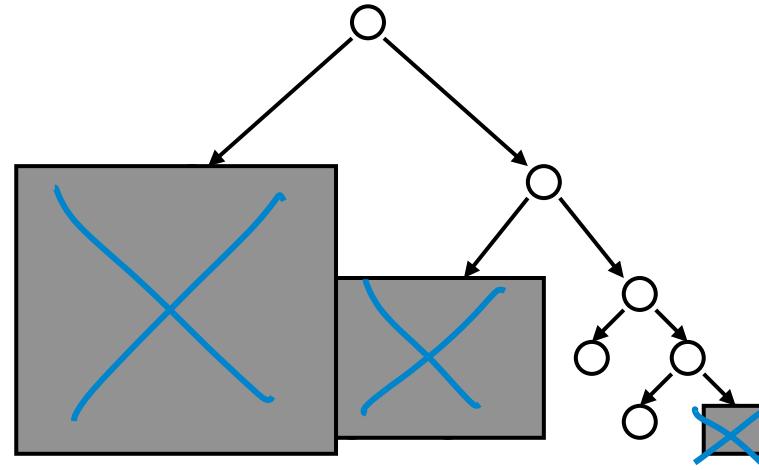
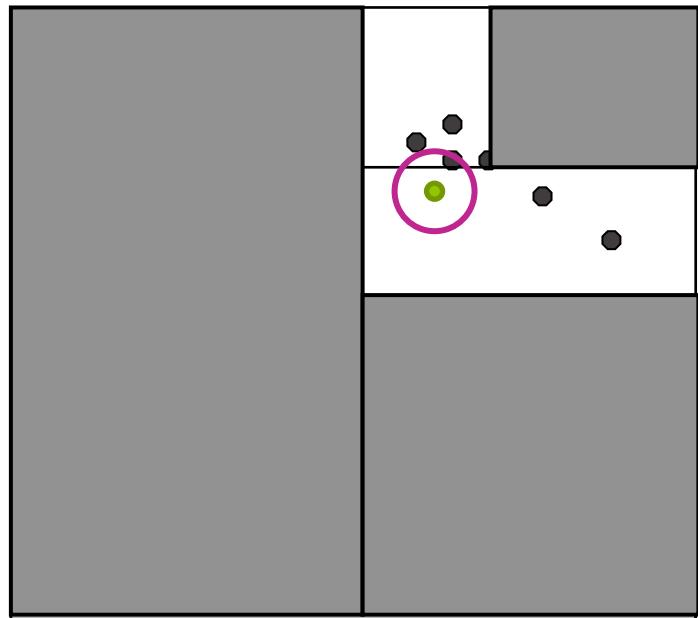
Use distance bound and bounding box of each node to
prune parts of tree that cannot include nearest neighbor

Nearest neighbor with KD-trees



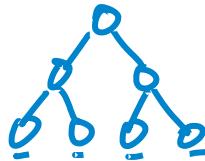
Use distance bound and bounding box of each node to
prune parts of tree that cannot include nearest neighbor

Nearest neighbor with KD-trees



Use distance bound and bounding box of each node to
prune parts of tree that **cannot include nearest neighbor**

Complexity

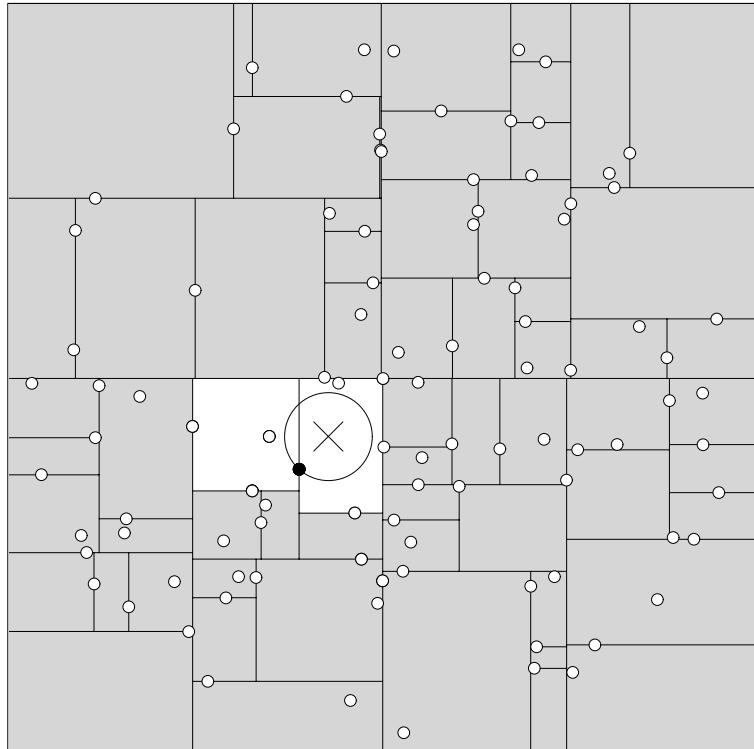


For (nearly) balanced, binary trees...

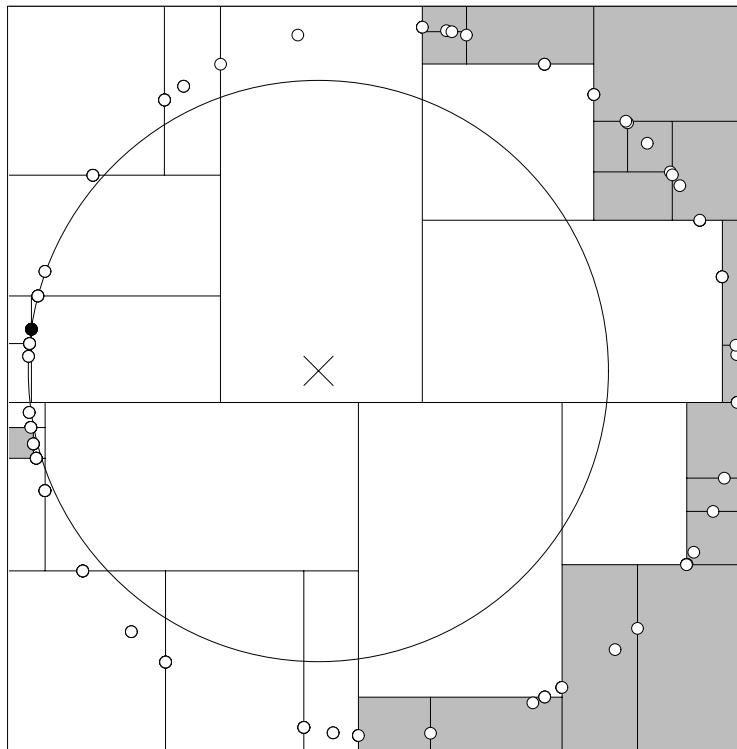
- Construction
 - Size: $2N-1$ nodes if 1 datapt at each leaf $\rightarrow O(N)$
 - Depth: $O(\log N)$
 - Median + send points left right: $O(N)$ at every level of the tree
 - Construction time: $O(N \log N)$
- 1-NN query
 - Traverse down tree to starting point: $O(\log N)$
 - Maximum backtrack and traverse: $O(N)$ in worst case
 - Complexity range: $O(\log N) \rightarrow O(N)$

Under some assumptions on distribution of points,
we get $O(\log N)$ but exponential in d

Complexity



pruned many
(closer to $O(\log N)$)



pruned few
(closer to $O(N)$)

Complexity for N queries

- Ask for nearest neighbor to each doc

N queries

- Brute force 1-NN:

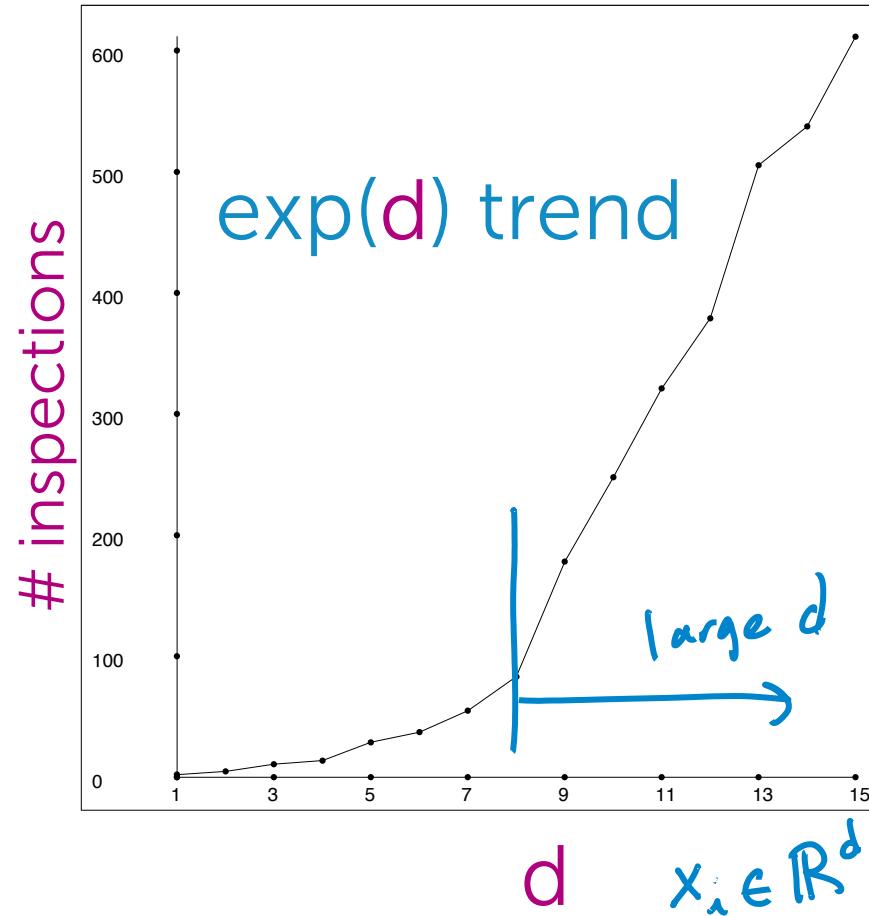
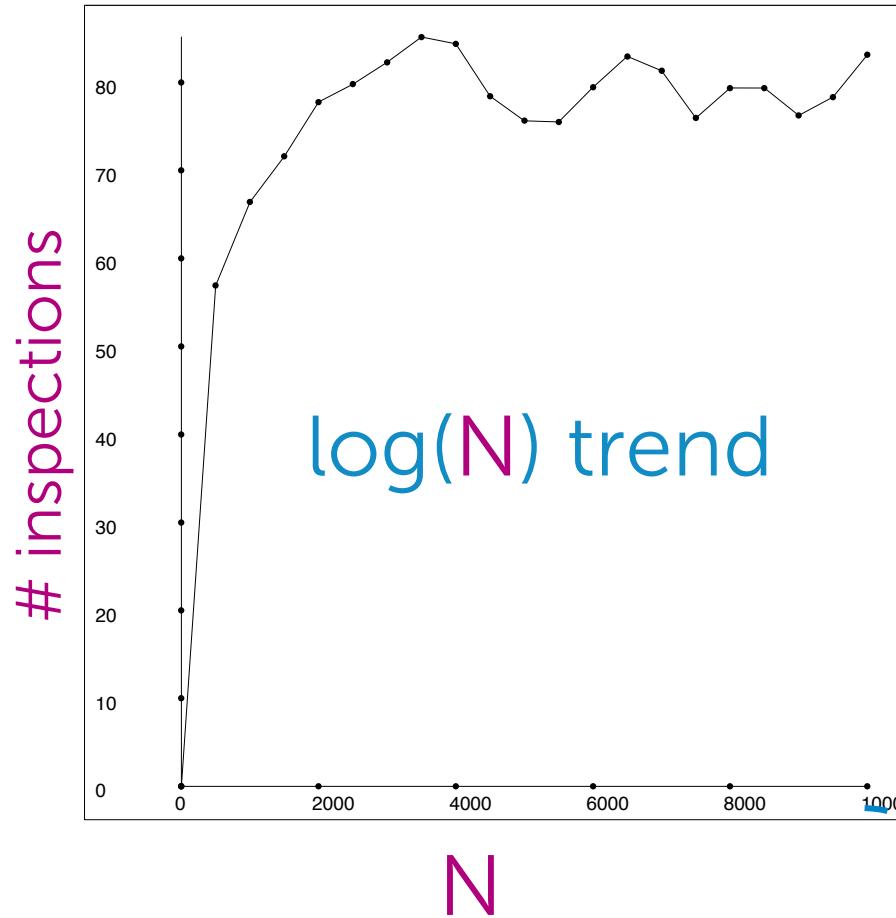
$O(N^2)$

- kd-trees:

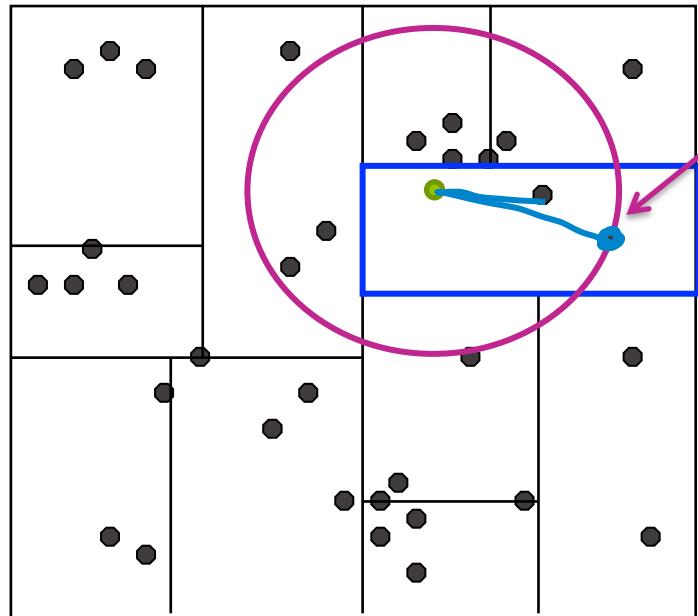
$O(N \log N) \rightarrow O(N^2)$

↑ potentially
very large
savings for
large N !

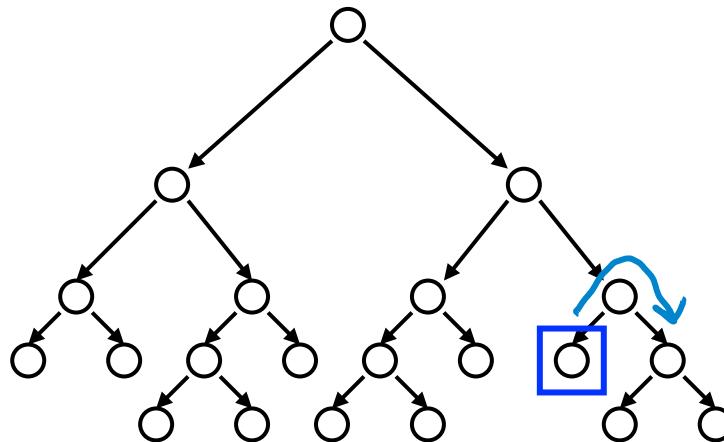
Inspections vs. N and d



k-NN with KD-trees



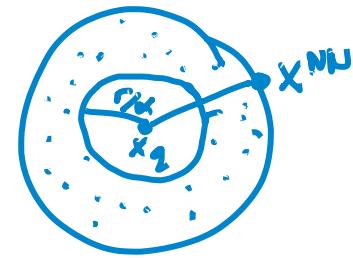
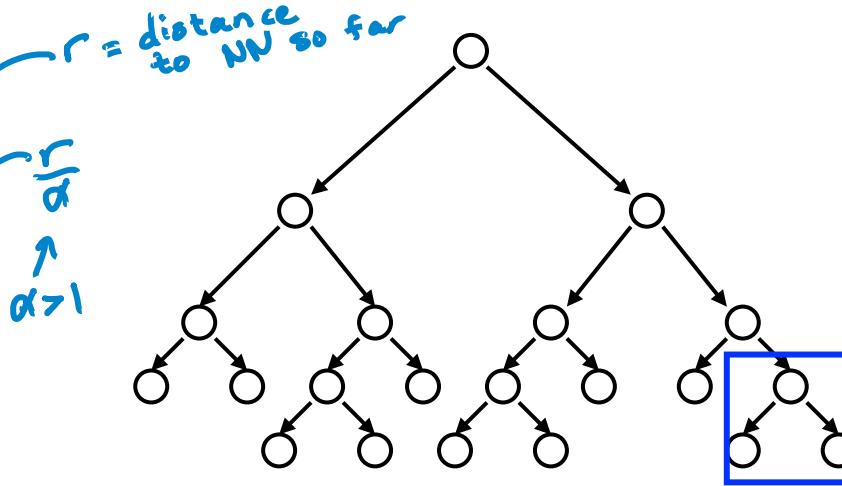
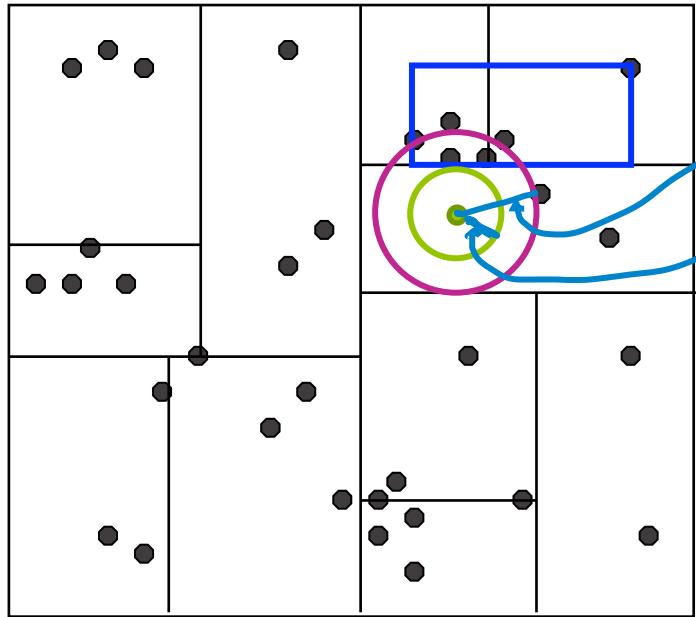
distance to 2nd nearest neighbor
in 2-NN example



Exactly same algorithm, but maintain distance to
furthest of current k nearest neighbors

Approximate k-NN search

Approximate k-NN with KD-trees



Before: Prune when distance to bounding box $> r$

Now: Prune when distance to bounding box $> r/\alpha$

Prunes more than allowed, but can **guarantee** that if we return a neighbor at distance r , then there is **no neighbor closer** than r/α

Bound loose...In practice, often closer to optimal.

Saves lots of search time at little cost in quality of NN!

Closing remarks on KD-trees

Tons of variants of kd-trees

- On construction of trees
(heuristics for splitting, stopping, representing branches...)
- Other representational data structures for fast NN search
(e.g., ball trees,...)

Nearest Neighbor Search

- Distance metric and data representation crucial to answer returned

For both, high-dim spaces are hard!

- Number of kd-tree searches can be exponential in dimension
 - Rule of thumb... $N \gg 2^d$... **Typically useless for large d.**
- Distances sensitive to irrelevant features
 - Most dimensions are just noise → everything is far away
 - Need technique to learn which features are important to given task

Acknowledgements

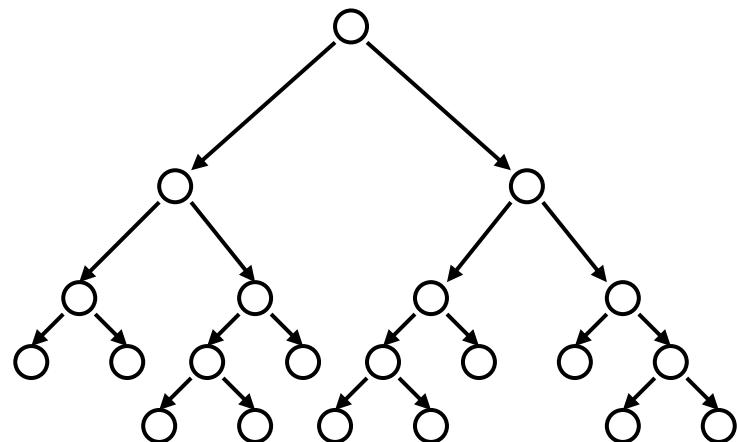
Thanks to Andrew Moore
(<http://www.cs.cmu.edu/~awm/>)
for the KD-trees slide outline



Locality sensitive hashing for approximate NN search

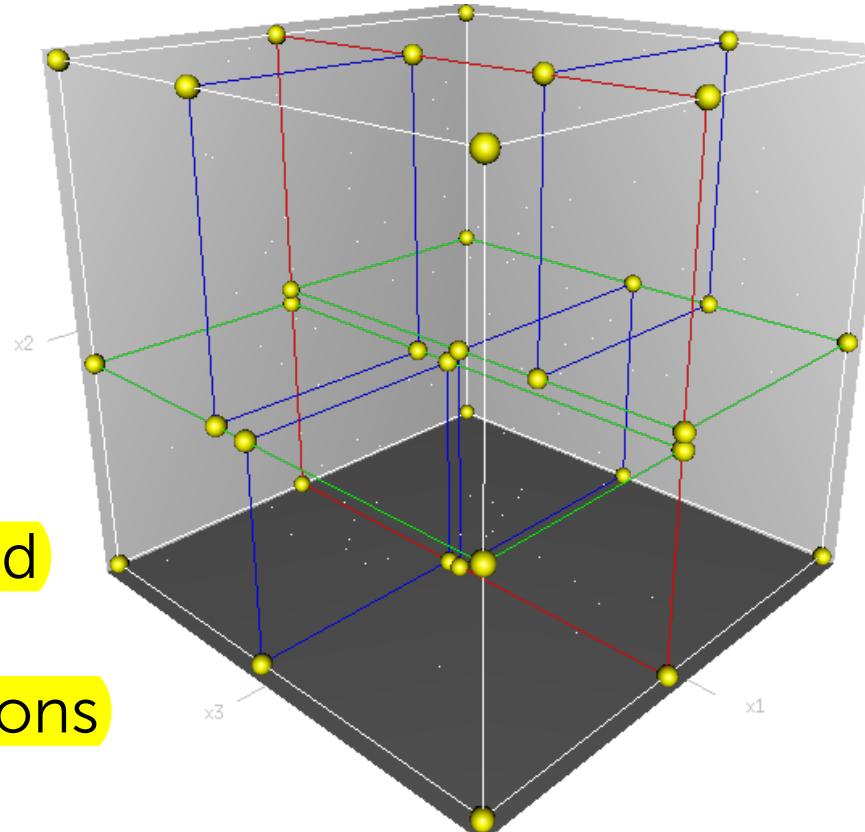
Motivating alternative approaches to approximate NN search

- KD-trees are cool, but...
 - Non-trivial to implement efficiently
 - Problems with high-dimensional data



KD-trees in high dimensions

- Unlikely to have any data points close to query point
- Once “nearby” point is found, the search radius is likely to intersect many hypercubes in at least one dim
- Not many nodes can be pruned
- Can show under some conditions that you visit at least 2^d nodes



Moving away from exact NN search

- Approximate neighbor finding...
 - Don't find exact neighbor, but that's okay for many applications



Out of millions of articles, do we need the closest article or just one that's pretty similar?

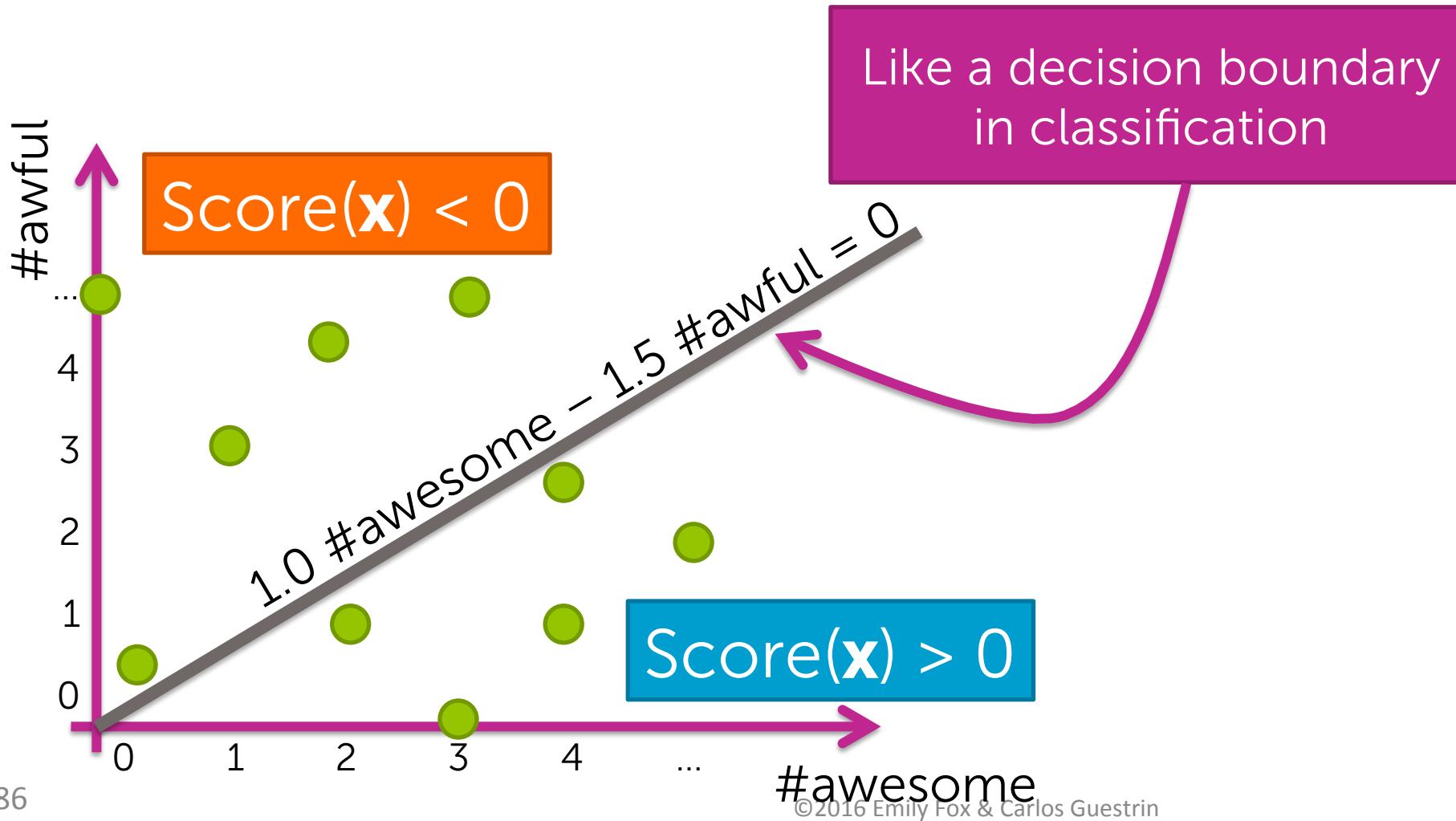
Do we even fully trust our measure of similarity???

- Focus on methods that provide good probabilistic guarantees on approximation

LSH as an alternative to KD-trees

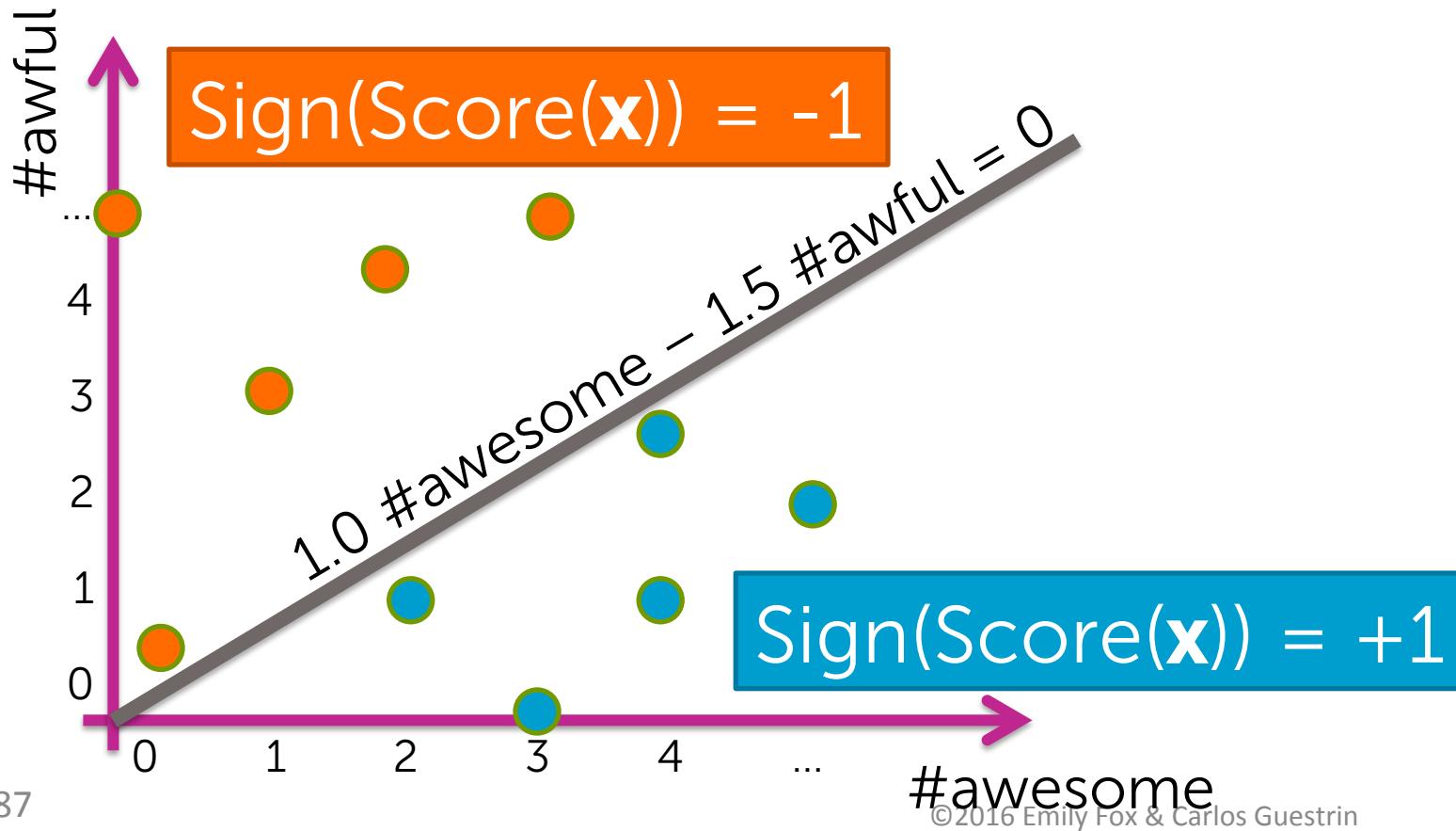
Simple “binning” of data into 2 bins

$$\text{Score}(\mathbf{x}) = 1.0 \# \text{awesome} - 1.5 \# \text{awful}$$



Simple “binning” of data into 2 bins

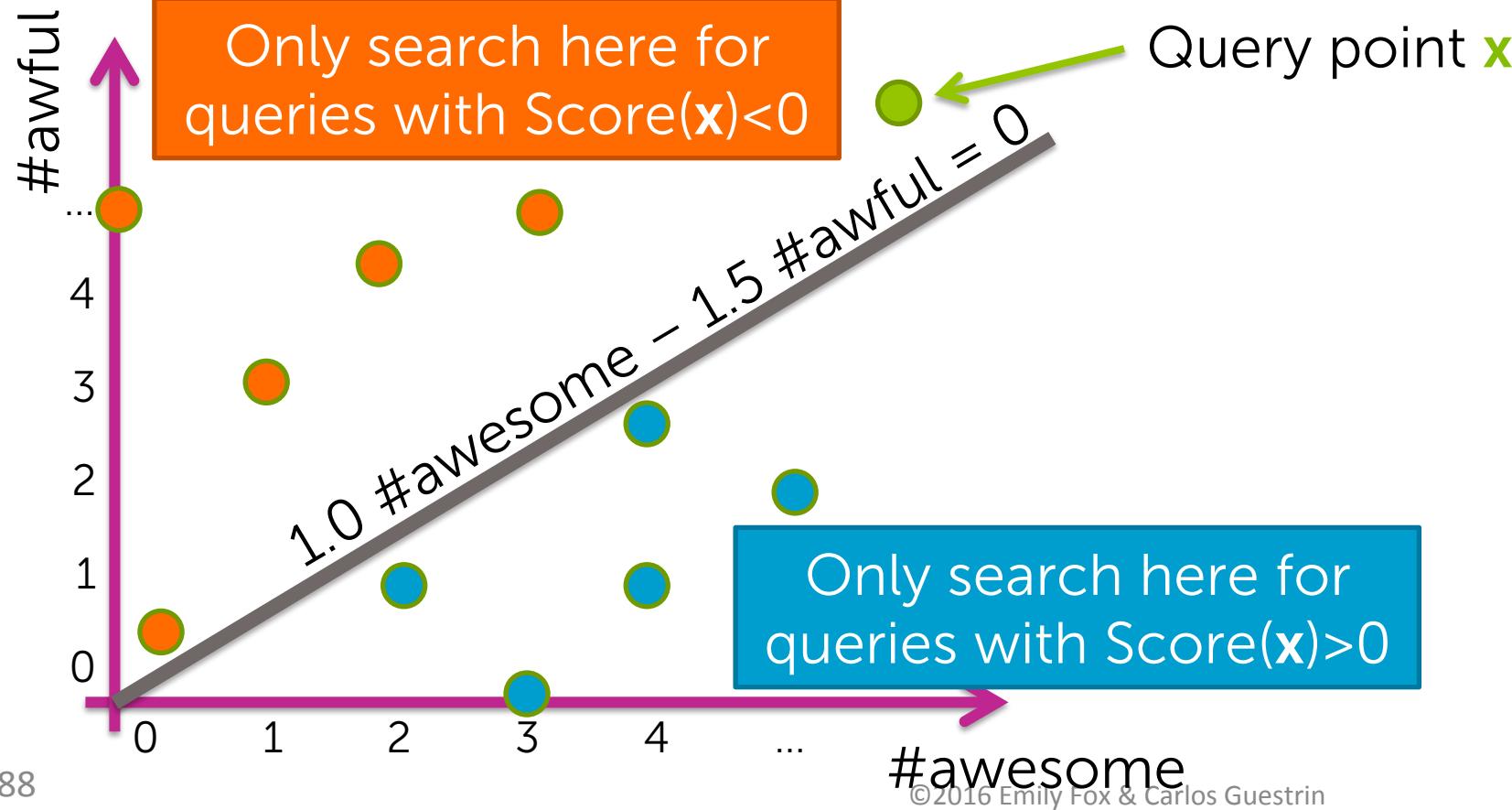
2D Data	Sign(Score)
$\mathbf{x}_1 = [0, 5]$	-1
$\mathbf{x}_2 = [1, 3]$	-1
$\mathbf{x}_3 = [3, 0]$	1
...	...



Using bins for NN search

2D Data	Sign(Score)	Bin index
$\mathbf{x}_1 = [0, 5]$	-1	0
$\mathbf{x}_2 = [1, 3]$	-1	0
$\mathbf{x}_3 = [3, 0]$	1	1
...

candidate
neighbors if
 $\text{Score}(\mathbf{x}) < 0$



Using score for NN search

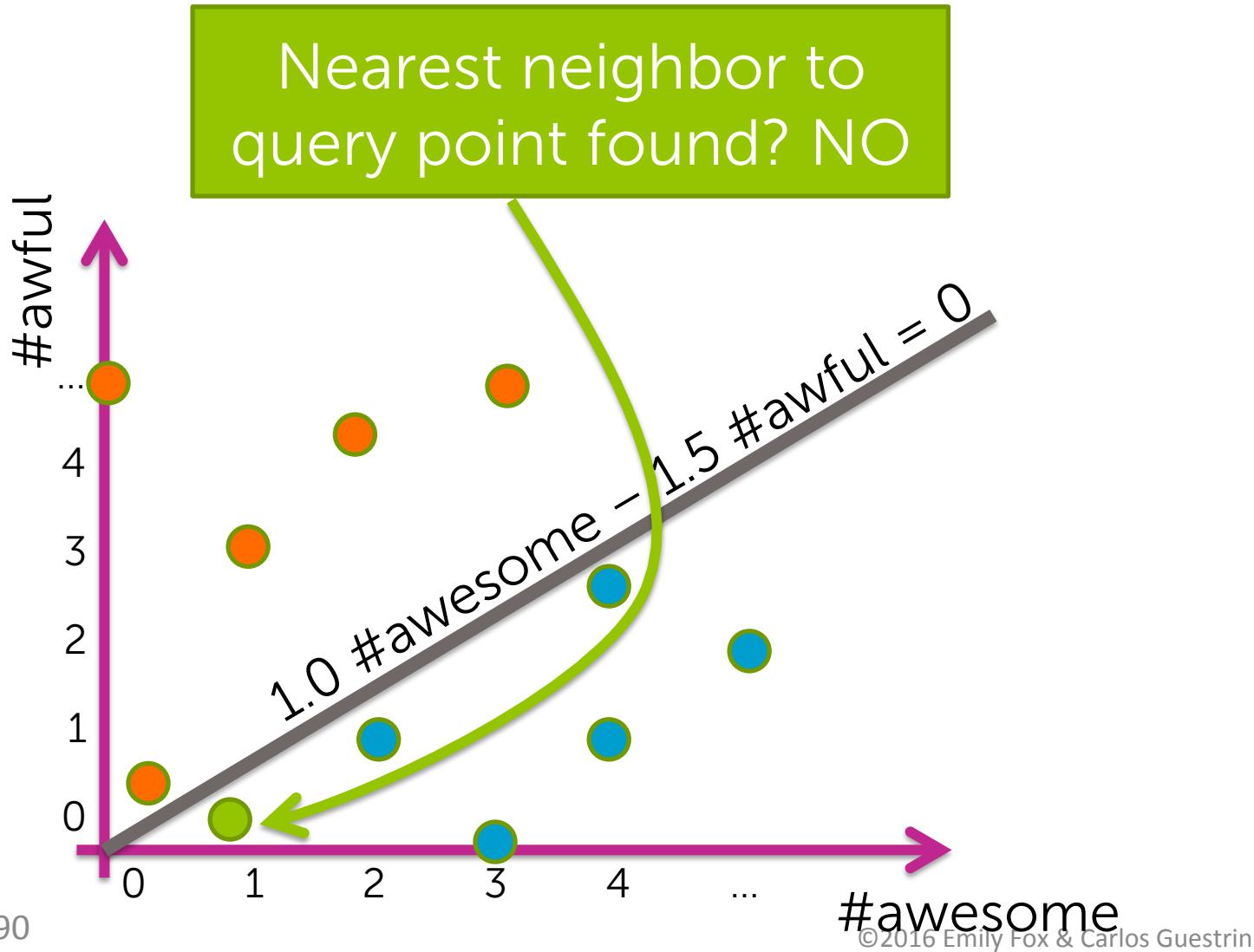
2D Data	Sign(Score)	Bin index
$\mathbf{x}_1 = [0, 5]$	-1	0
$\mathbf{x}_2 = [1, 3]$	-1	0
$\mathbf{x}_3 = [3, 0]$	1	1
...

candidate
neighbors if
 $\text{Score}(x) < 0$

Bin	0	1	HASH TABLE
List containing indices of datapoints:	{1,2,4,7,...}	{3,5,6,8,...}	

search for NN
amongst this set

Provides approximate NN



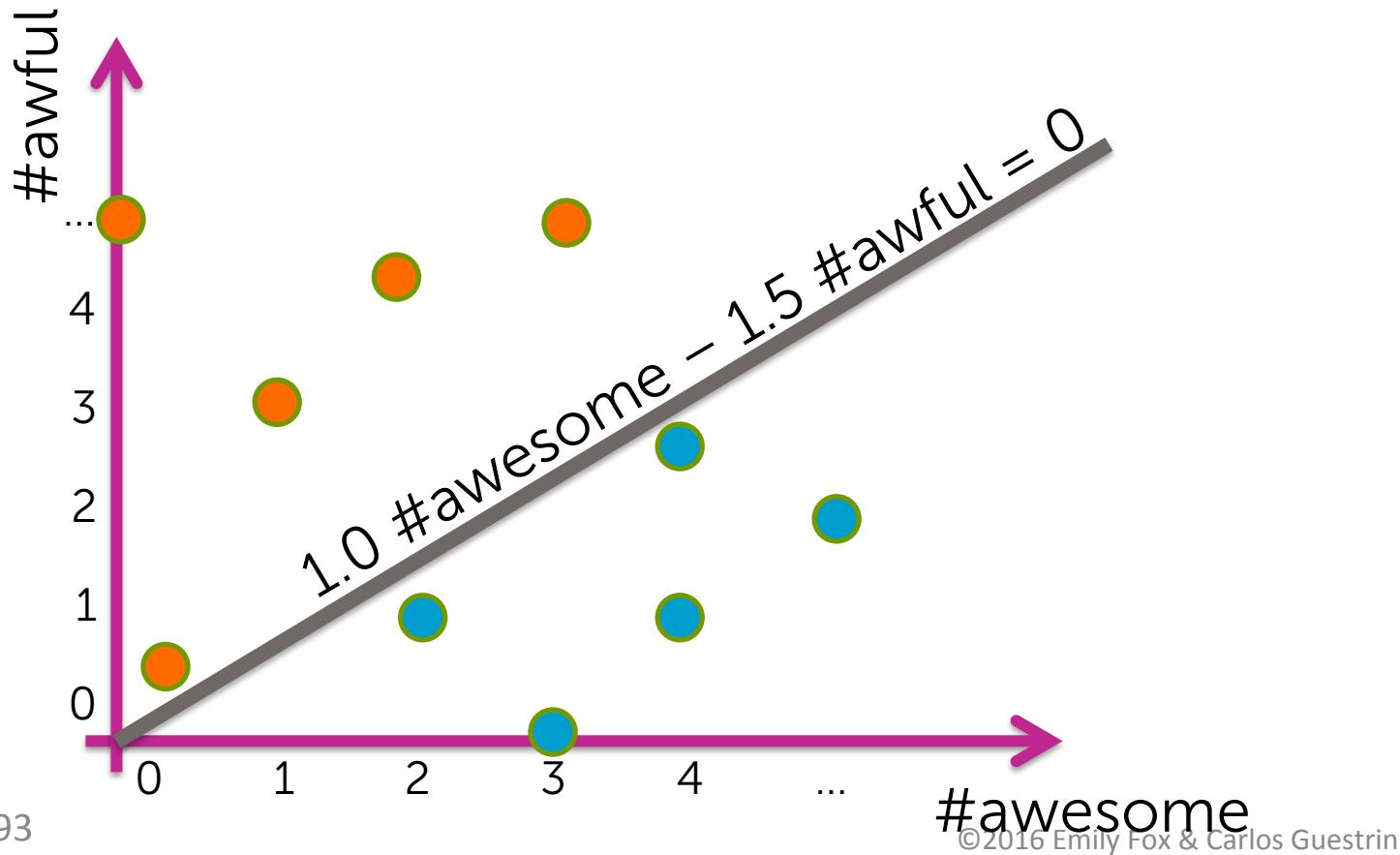
A practical implementation

Three potential issues with simple approach

1. Challenging to find good line
2. Poor quality solution:
 - Points close together get split into separate bins
3. Large computational cost:
 - Bins might contain many points, so still searching over large set for each NN query

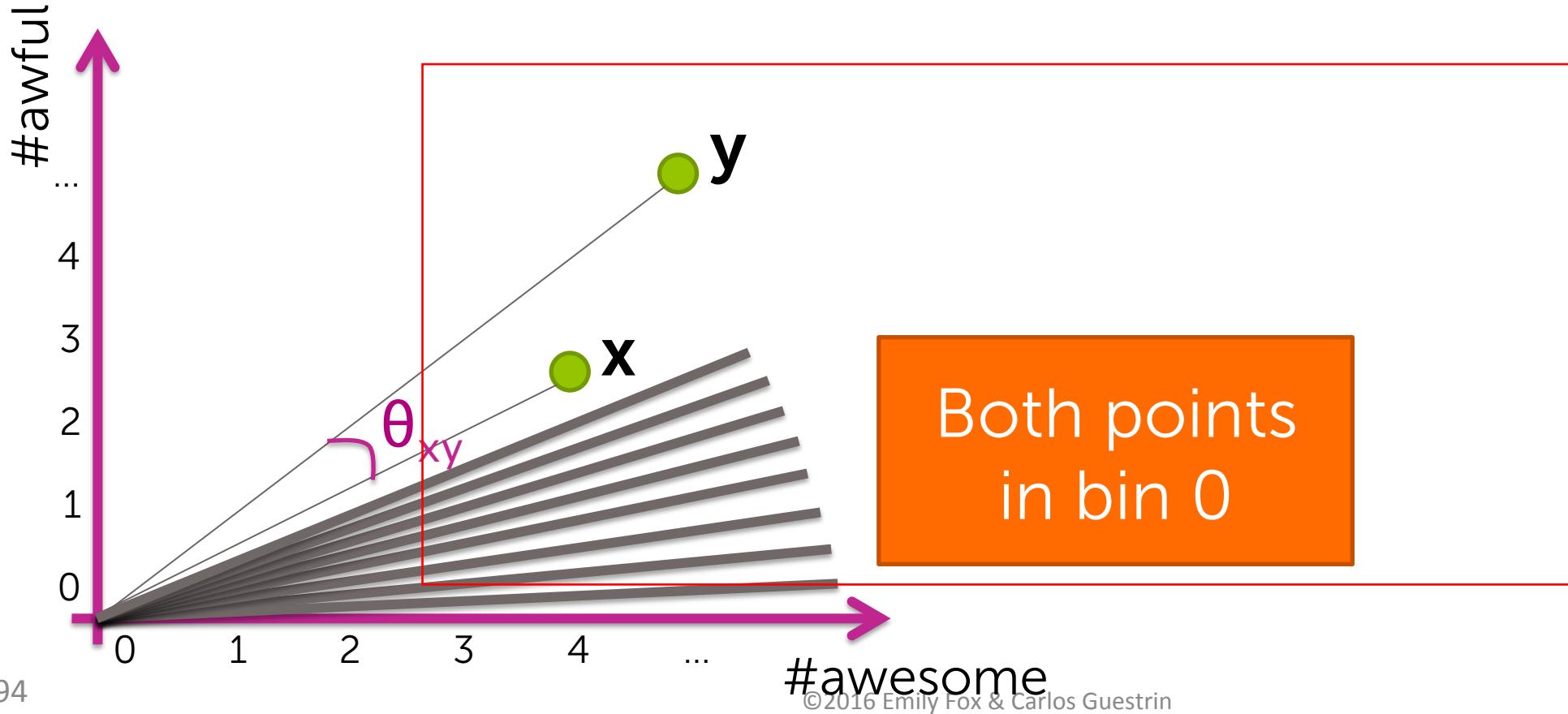
How to define the line?

Crazy idea:
Define line randomly!



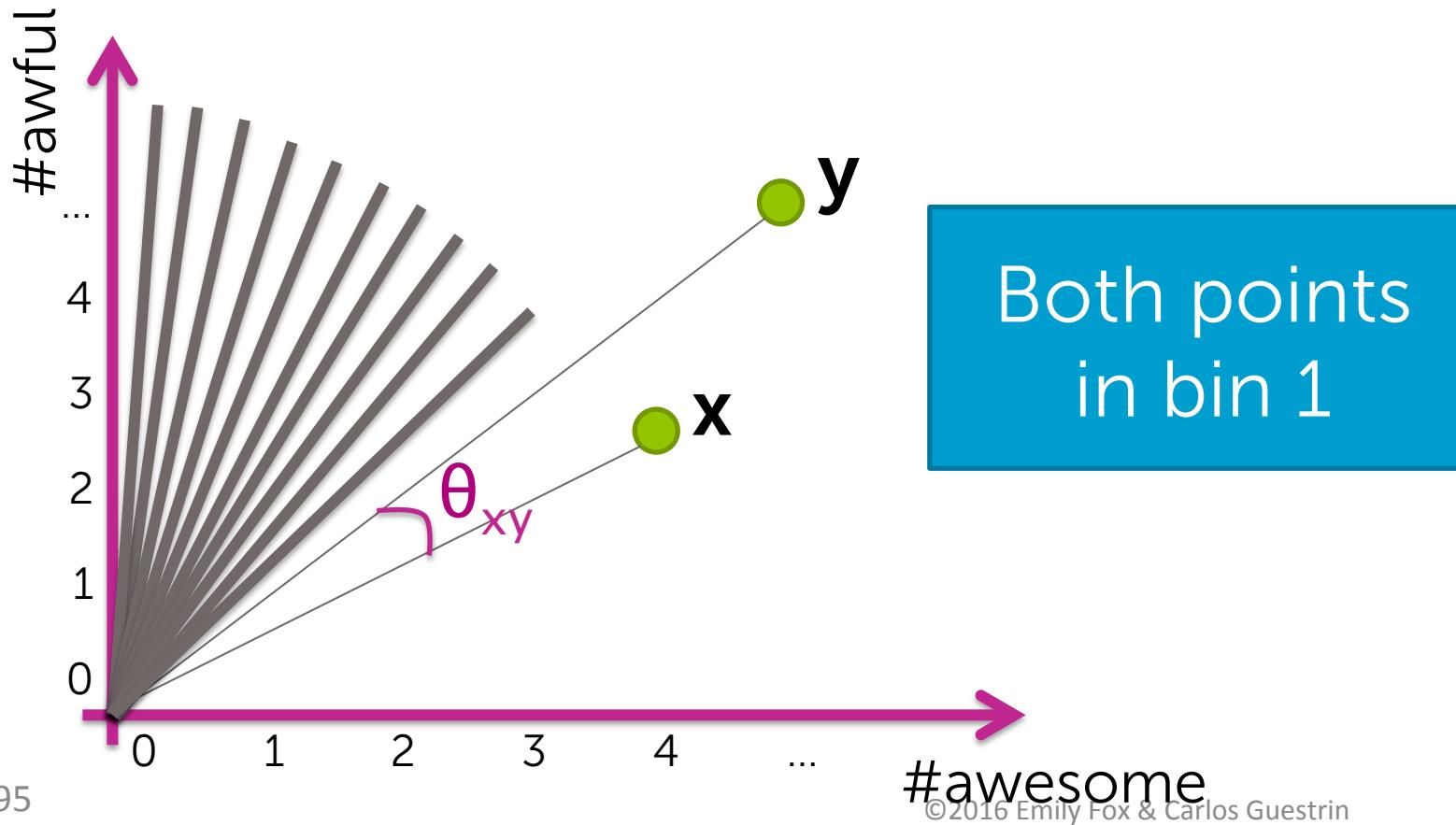
How bad can a random line be?

Goal: If x, y are close (according to cosine similarity),
want binned values to be the same.



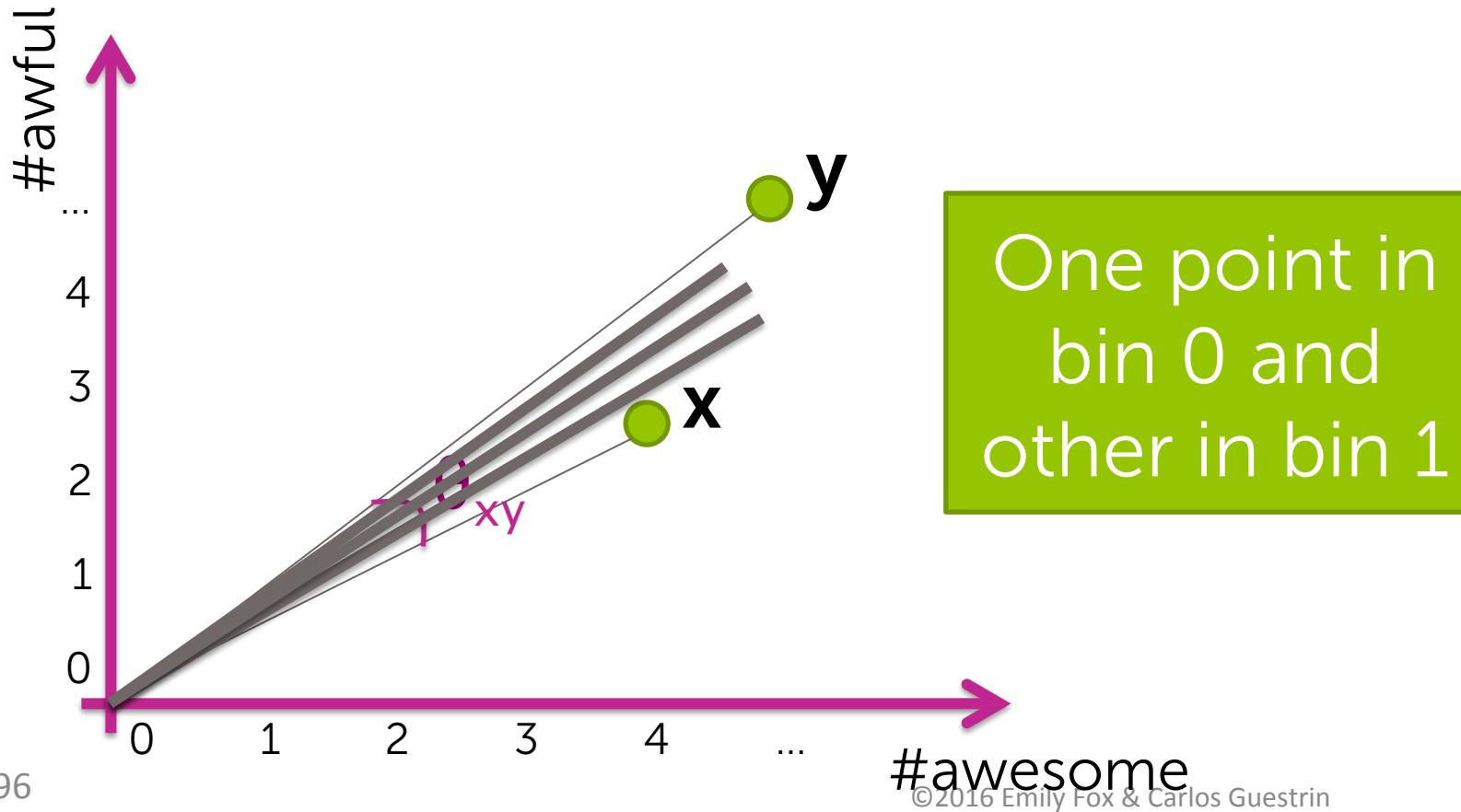
How bad can a random line be?

Goal: If x, y are close (according to cosine similarity),
want binned values to be the same.



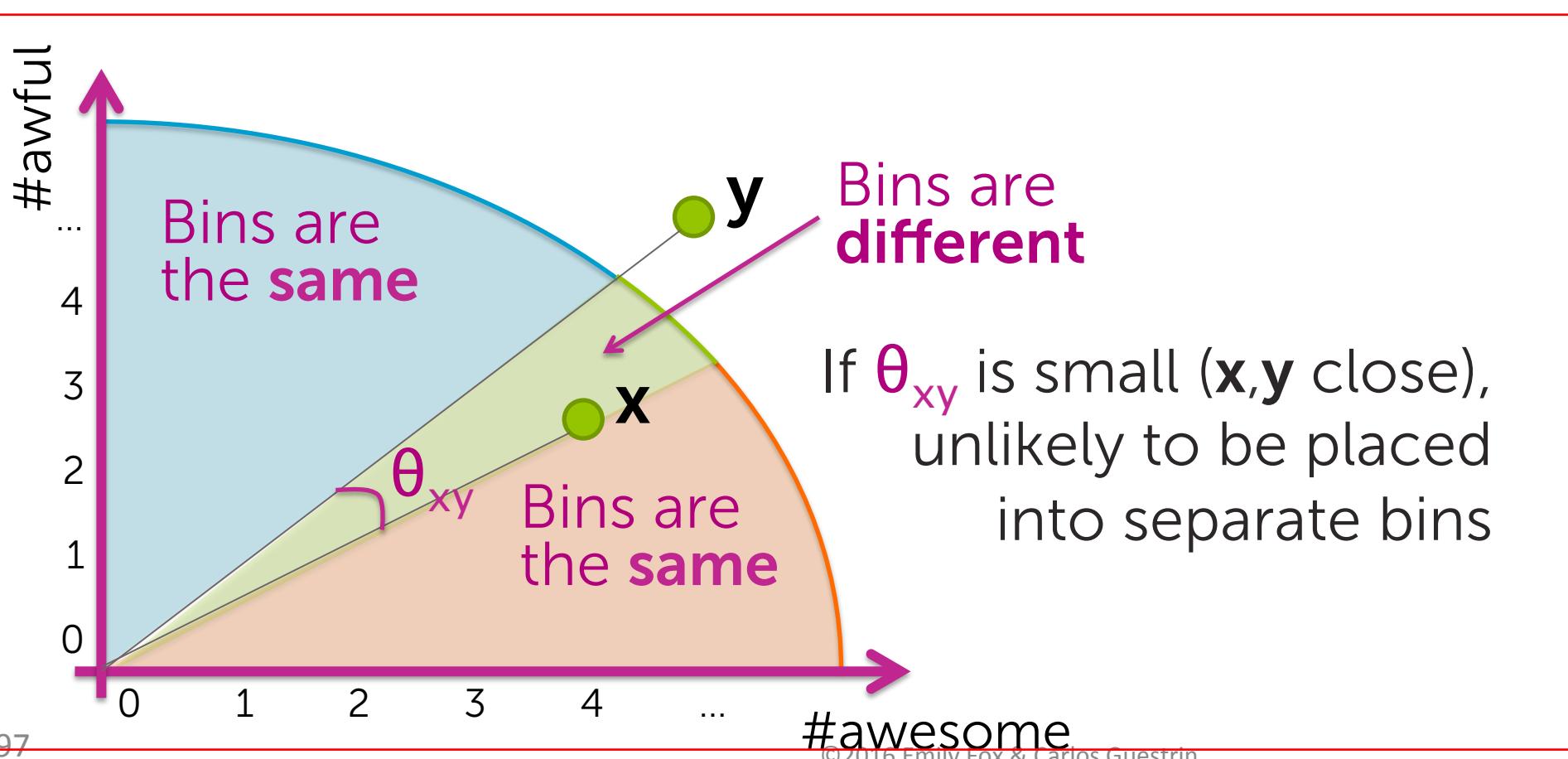
How bad can a random line be?

Goal: If x, y are close (according to cosine similarity), want binned values to be the same.



How bad can a random line be?

Goal: If x, y are close (according to cosine similarity), want binned values to be the same.



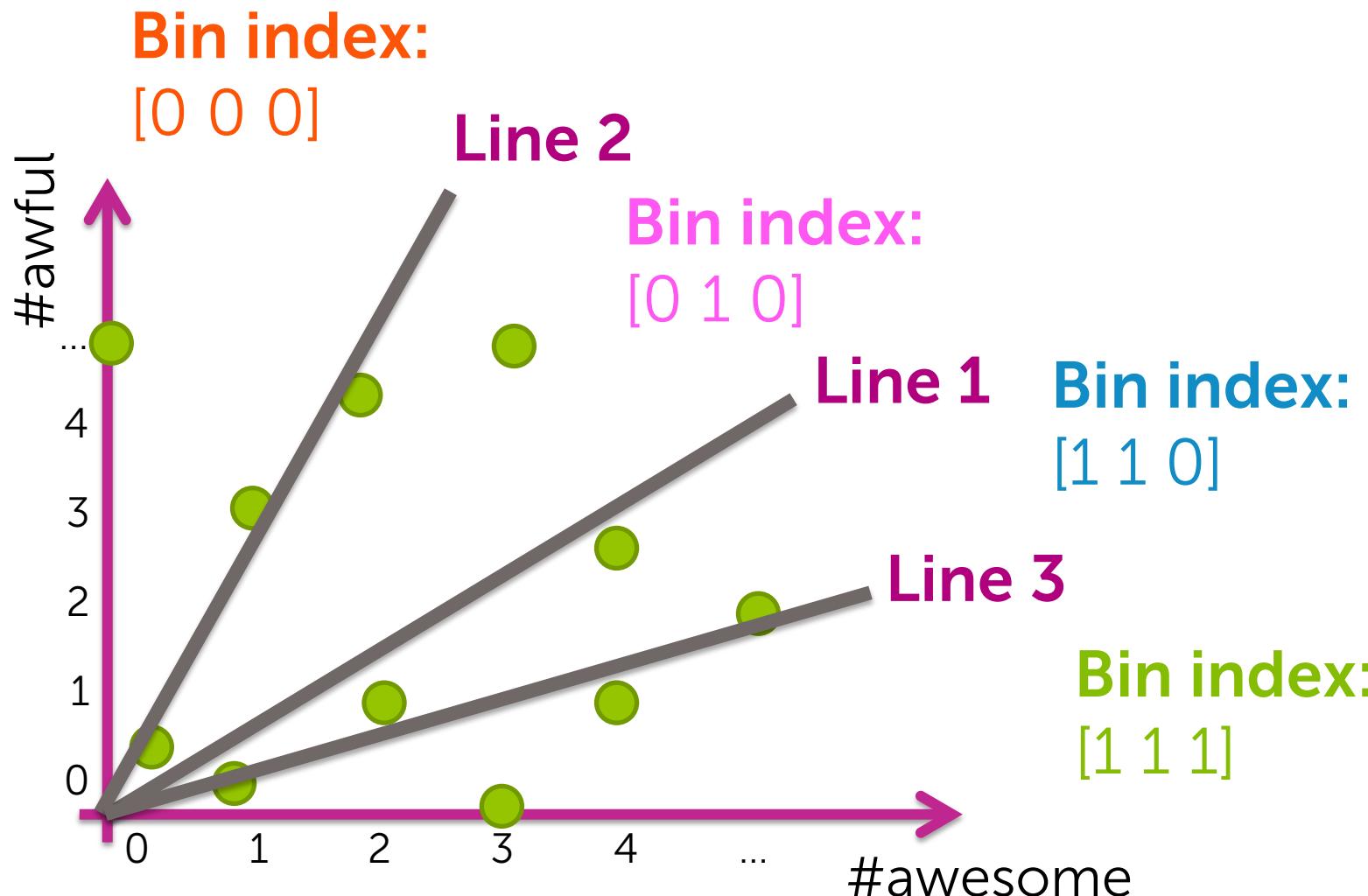
Three potential issues with simple approach

1. Challenging to find good line
2. Poor quality solution:
 - Points close together get split into separate bins
3. Large computational cost:
 - Bins might contain many points, so still searching over large set for each NN query

Bin	0	1
List containing indices of datapoints:	{1,2,4,7,...}	{3,5,6,8,...}

Improving efficiency:
Reducing # points examined per query

Reducing search cost through more bins



Using score for NN search

2D Data	Sign (Score ₁)	Bin 1 index	Sign (Score ₂)	Bin 2 index	Sign (Score ₃)	Bin 3 index
$\mathbf{x}_1 = [0, 5]$	-1	0	-1	0	-1	0
$\mathbf{x}_2 = [1, 3]$	-1	0	-1	0	-1	0
$\mathbf{x}_3 = [3, 0]$	1	1	1	1	1	1
...

Bin	[0 0 0] = 0	[0 0 1] = 1	[0 1 0] = 2	[0 1 1] = 3	[1 0 0] = 4	[1 0 1] = 5	[1 1 0] = 6	[1 1 1] = 7
Data indices:	{1,2}	--	{4,8,11}	--	--	--	{7,9,10}	{3,5,6}

search for NN
amongst this set

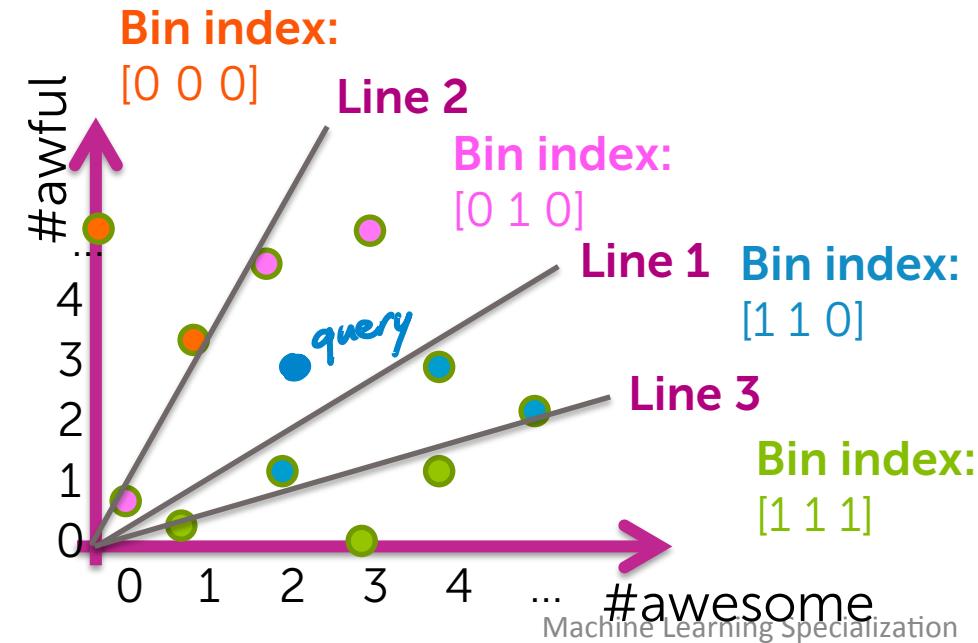
Improving search quality by searching neighboring bins

Bin	[0 0 0] = 0	[0 0 1] = 1	[0 1 0] = 2	[0 1 1] = 3	[1 0 0] = 4	[1 0 1] = 5	[1 1 0] = 6	[1 1 1] = 7
Data indices:	{1,2}	--	{4,8,11}	--	--	--	{7,9,10}	{3,5,6}

Query point here,
but is NN?

Not necessarily

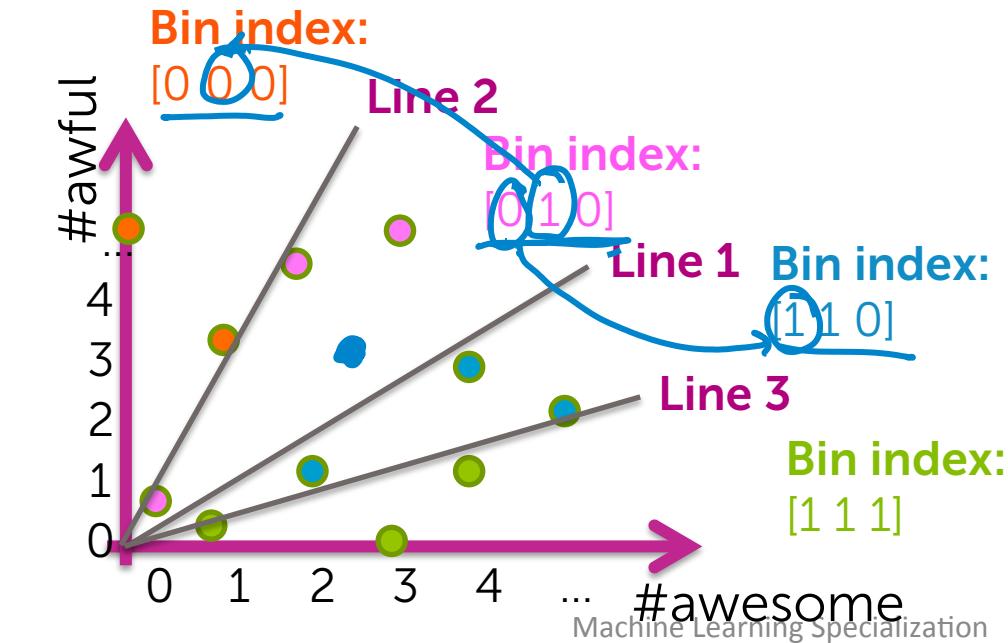
Even worse than before...Each line can split pts.
Sacrificing accuracy for speed



Improving search quality by searching neighboring bins

Bin	[0 0 0] = 0	[0 0 1] = 1	[0 1 0] = 2	[0 1 1] = 3	[1 0 0] = 4	[1 0 1] = 5	[1 1 0] = 6	[1 1 1] = 7
Data indices:	{1,2}	--	{4,8,11}	--	--	--	{7,9,10}	{3,5,6}

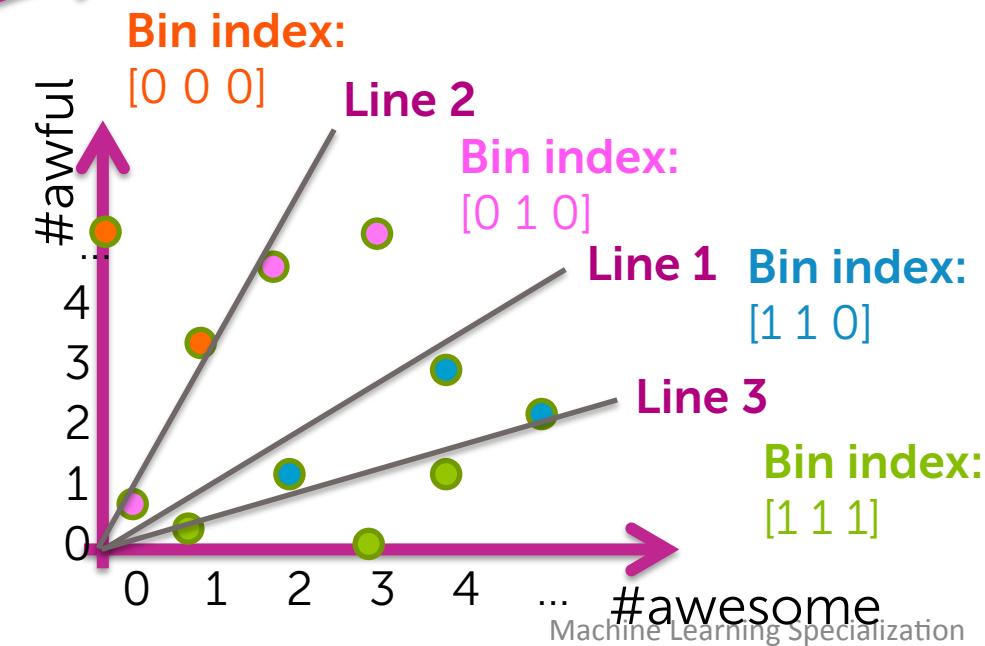
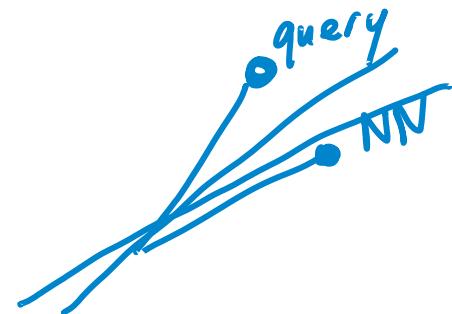
Next closest
bins
(flip 1 bit)



Improving search quality by searching neighboring bins

Bin	[0 0 0] = 0	[0 0 1] = 1	[0 1 0] = 2	[0 1 1] = 3	[1 0 0] = 4	[1 0 1] = 5	[1 1 0] = 6	[1 1 1] = 7
Data indices:	{1,2}	--	{4,8,11}	--	--	--	{7,9,10}	{3,5,6}

Further bin
(flip 2 bits)



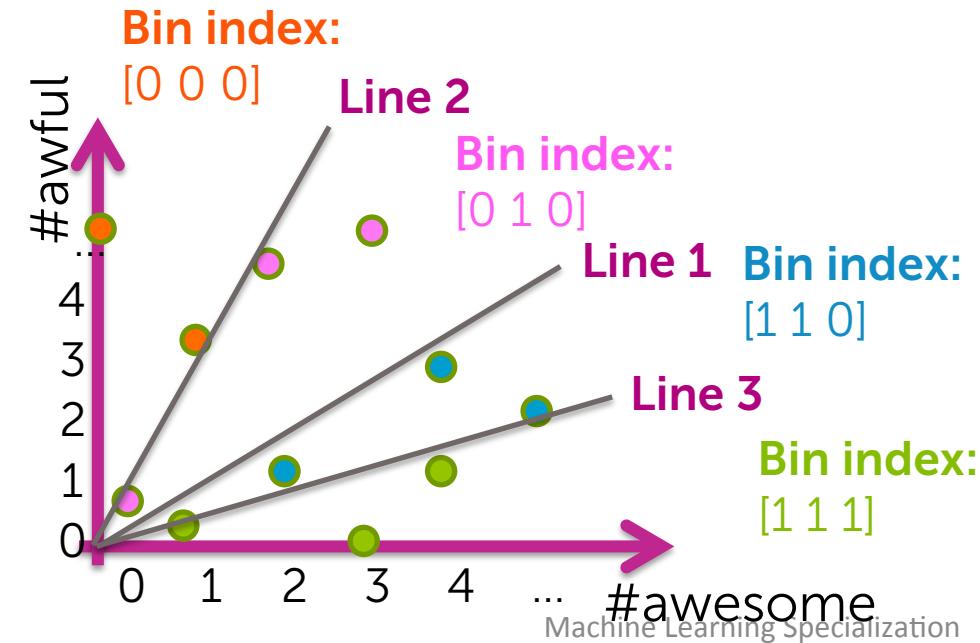
Improving search quality by searching neighboring bins

Bin	[0 0 0] = 0	[0 0 1]	[0 1 0] = 2	[0 1 1]	[1 0 0]	[1 0 1]	[1 1 0]	[1 1 1] = 7
Data indices:	{1,2}	--	{4,8,11}	--	--	--	{7,9,10}	{3,5,6}

Quality of retrieved NN can only improve with searching more bins

Algorithm:

Continue searching until computational budget is reached or quality of NN good enough



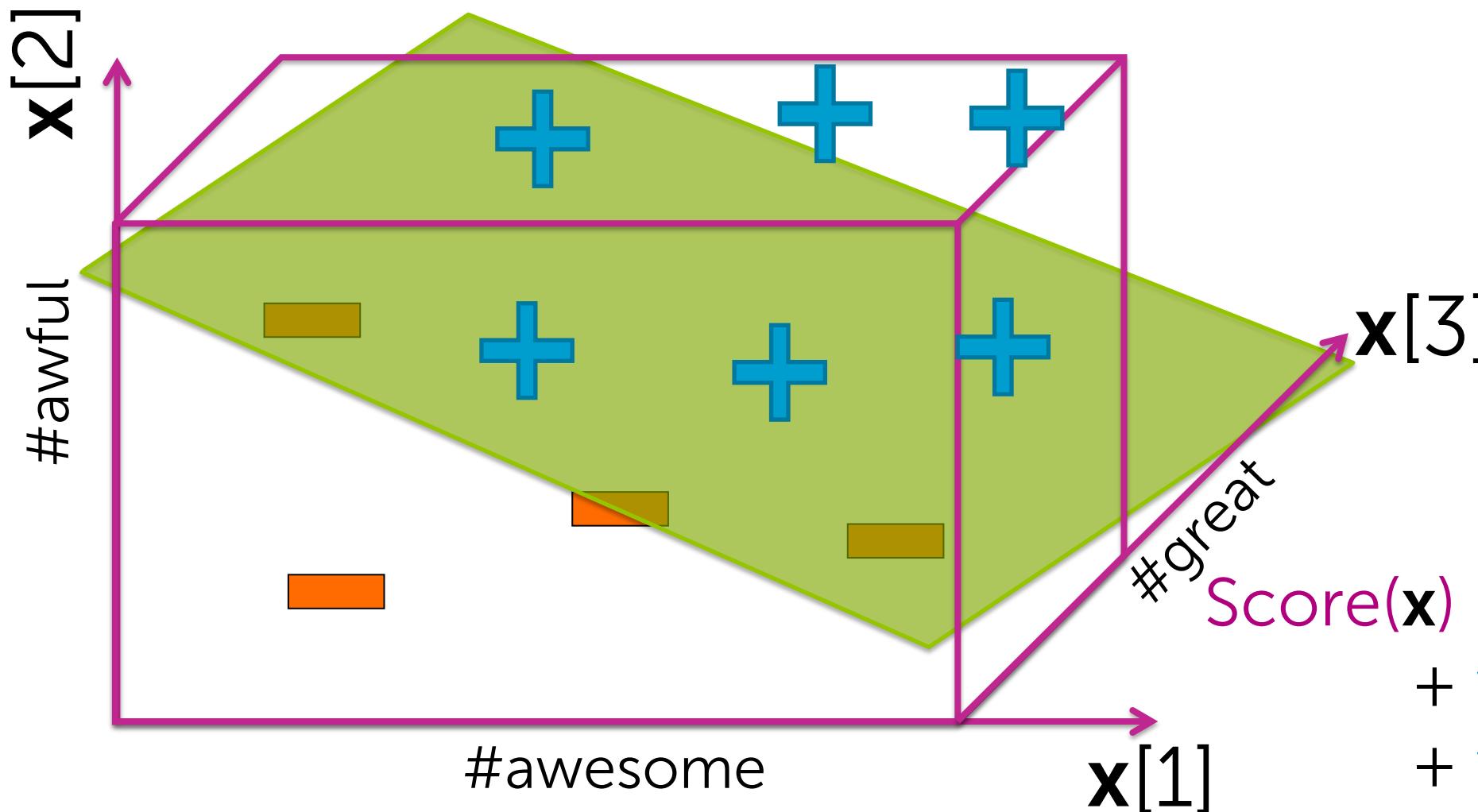
LSH recap

kd-tree competitor
data structure

- Draw h random lines
 - Compute “score” for each point under each line and translate to binary index
 - Use h -bit binary vector per data point as bin index
 - Create hash table
-
- For each query point \mathbf{x} , search $\text{bin}(\mathbf{x})$, then neighboring bins until time limit

Moving to higher dimensions d

Draw random planes



$$\begin{aligned}\text{Score}(\mathbf{x}) &= v_1 \# \text{awesome} \\ &+ v_2 \# \text{awful} \\ &+ v_3 \# \text{great}\end{aligned}$$

Cost of binning points in d-dim

$$\text{Score}_i(\mathbf{x}) = v_1^{(i)} \# \text{awesome} + v_2^{(i)} \# \text{awful} + v_3^{(i)} \# \text{great}$$

↑
ith hyperplane

In high-dim, (and some applications)
this is often a sparse mult.

} Per data point,
need **d multiplies**
to determine bin
index **per plane**

One-time cost offset if many queries of fixed dataset

Using multiple tables for even greater efficiency in NN search

OPTIONAL



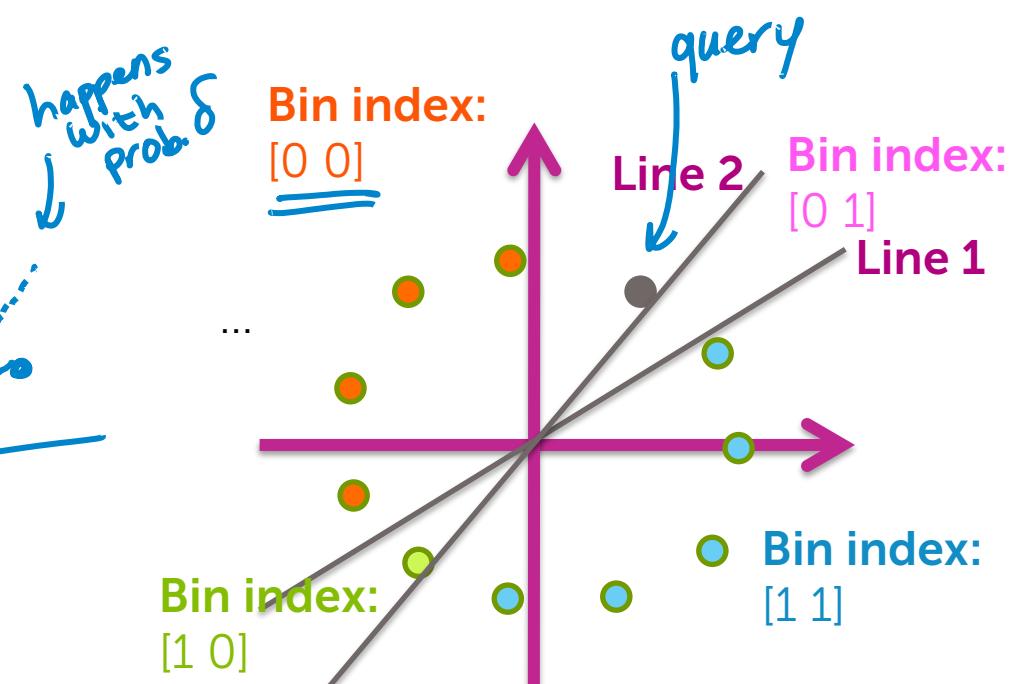
If I throw down 2 lines...

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3

For simplicity, assume we **search bins 1 bit off** from query

Let δ be the probability of a line falling between points θ apart

Search 3 bins and do not find NN with probability δ^2

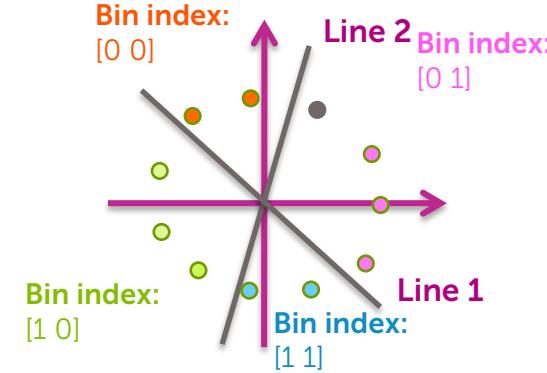
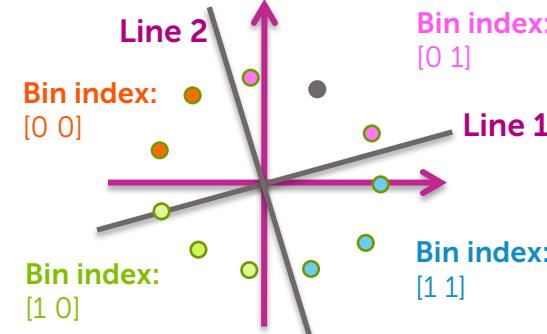
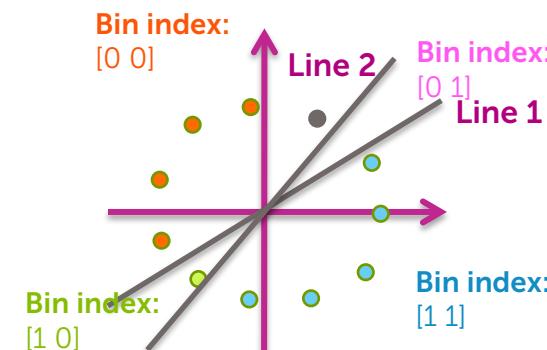


What if I repeat the 2-line binning?

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3

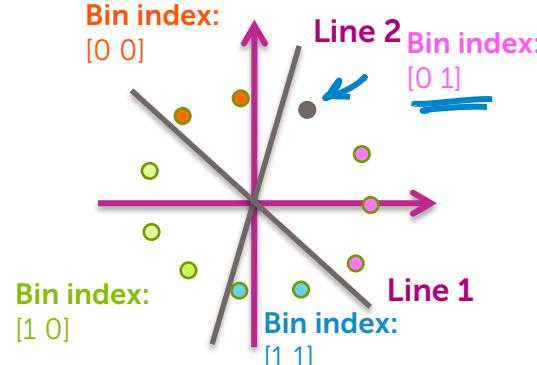
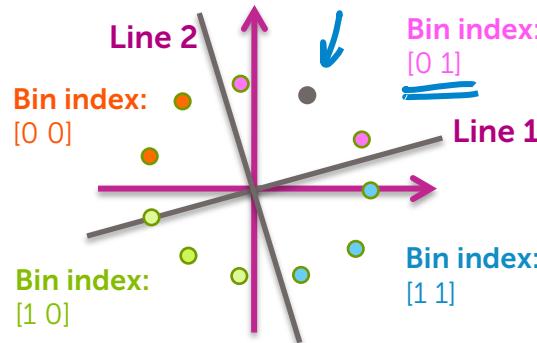
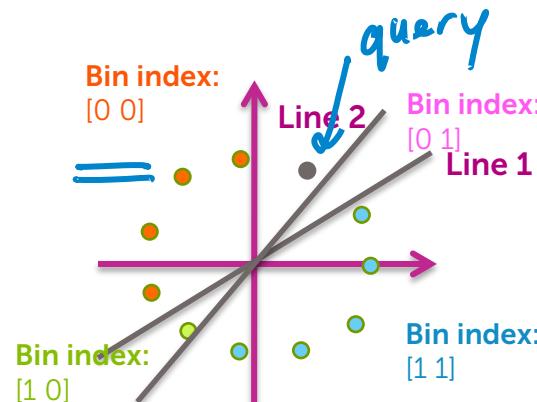


What if I repeat the 2-line binning?

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3



Now, search
only query bin
per table

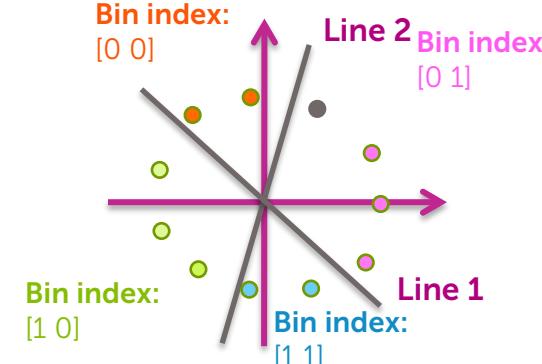
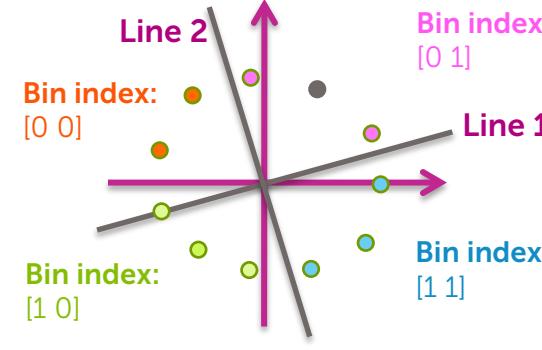
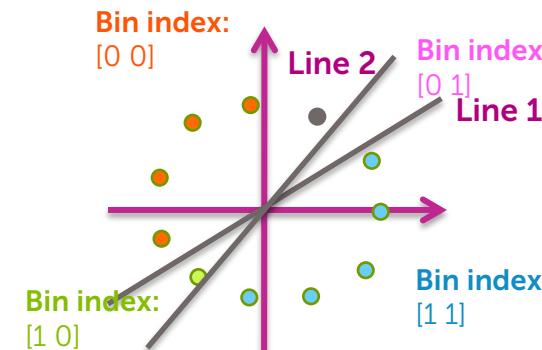
Still searching 3
bins, but what is
chance of not
finding NN?

What if I repeat the 2-line binning?

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3



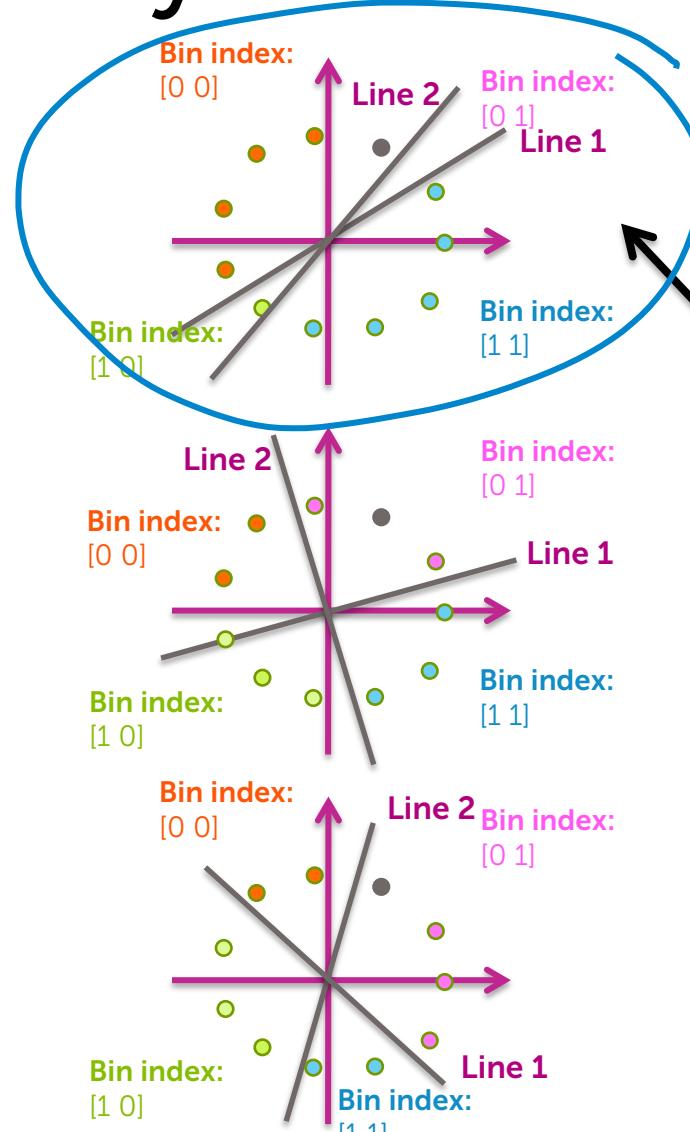
What is chance
that query pt
and NN are split
in **all tables**?

Probability of splitting neighboring points many times

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3



Probability NN is
in different bin:

$$\begin{aligned} \text{Prob} &= 1 - \text{Pr}(\text{same bin}) \\ &= 1 - (1 - \delta)^2 \\ &= 2\delta - \delta^2 \end{aligned}$$

$1 - \delta$ = prob. that 1 line
does not split
query + NN

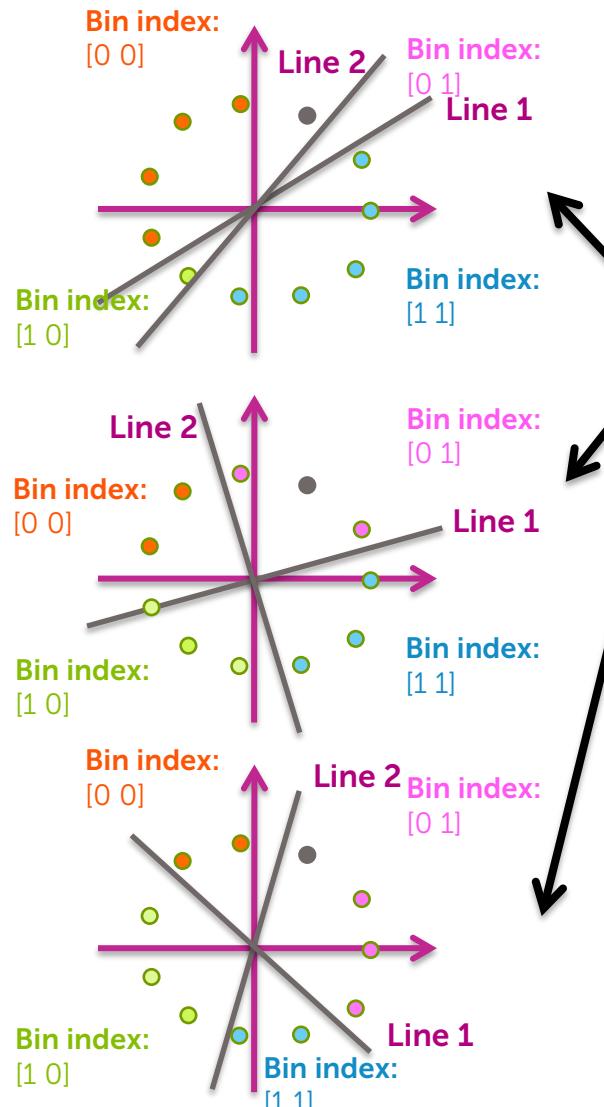
$(1 - \delta)^2$ = prob. that 2 lines
don't split pts

Probability of splitting neighboring points many times

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3



Probability NN is
in different bin in
all 3 tables:

$$\text{Prob} = (2\delta - \delta^2)^3$$

of hash tables

Comparing approaches for 2-bit tables

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3

bins
searched

3

prob. of
no NN

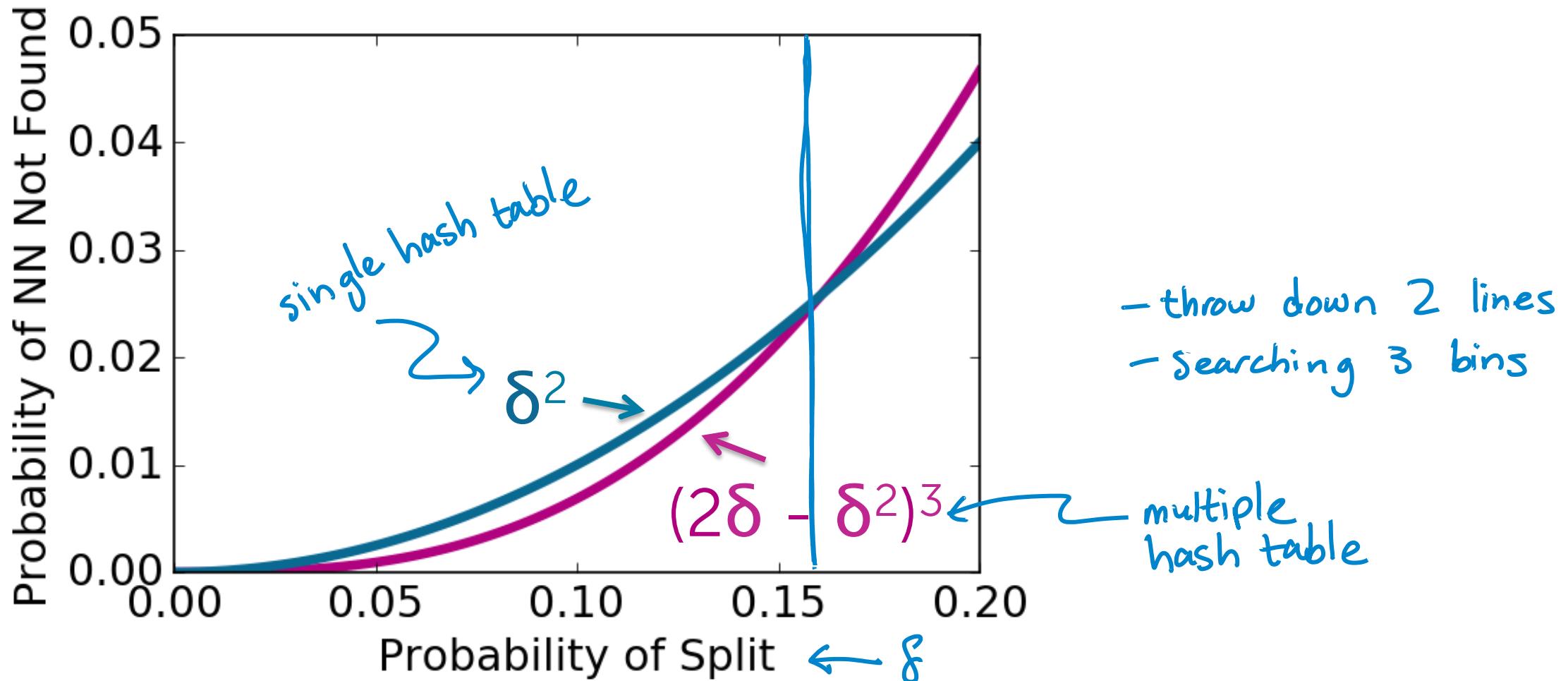
$$\underline{\delta^2}$$

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3

3

$$(2\underline{\delta} - \underline{\delta^2})^3$$

Comparing probabilities



If I throw down h lines...

Bin	[0 0 0] = 0	[0 0 1] = 1	[0 1 0] = 2	[0 1 1] = 3	[1 0 0] = 4	[1 0 1] = 5	[1 1 0] = 6	[1 1 1] = 7
indices	L0	L1	L2	L3	L4	L5	L6	L7

Still assume we search bins 1 bit off from query

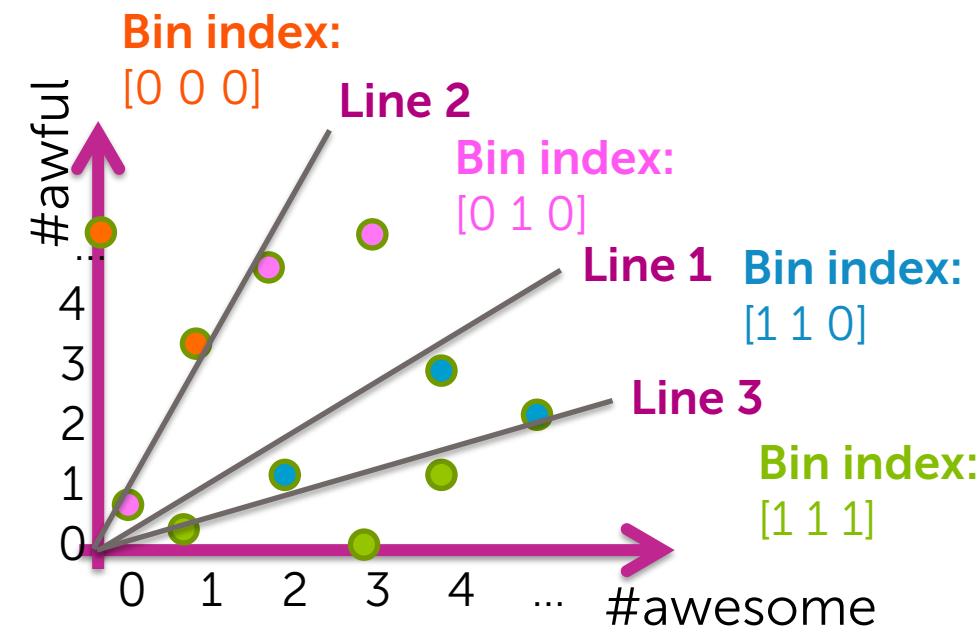
Prob. of being > 1 bit away

$$= 1 - \Pr(\text{same bin}) - \Pr(1 \text{ bin away})$$

$$= 1 - \Pr(\text{no split lines}) - \Pr(1 \text{ split line})$$

$$= 1 - (1 - \delta)^h - h\delta(1 - \delta)^{h-1}$$

↑ Prob. of no split
is $1 - \delta$
+ we have h
lines → Prob. none of
 h lines split is $(1 - \delta)^h$



If I throw down h lines...

Bin	[0 0 0] = 0	[0 0 1] = 1	[0 1 0] = 2	[0 1 1] = 3	[1 0 0] = 4	[1 0 1] = 5	[1 1 0] = 6	[1 1 1] = 7
indices	L0	L1	L2	L3	L4	L5	L6	L7

Still assume we **search bins 1 bit off** from query

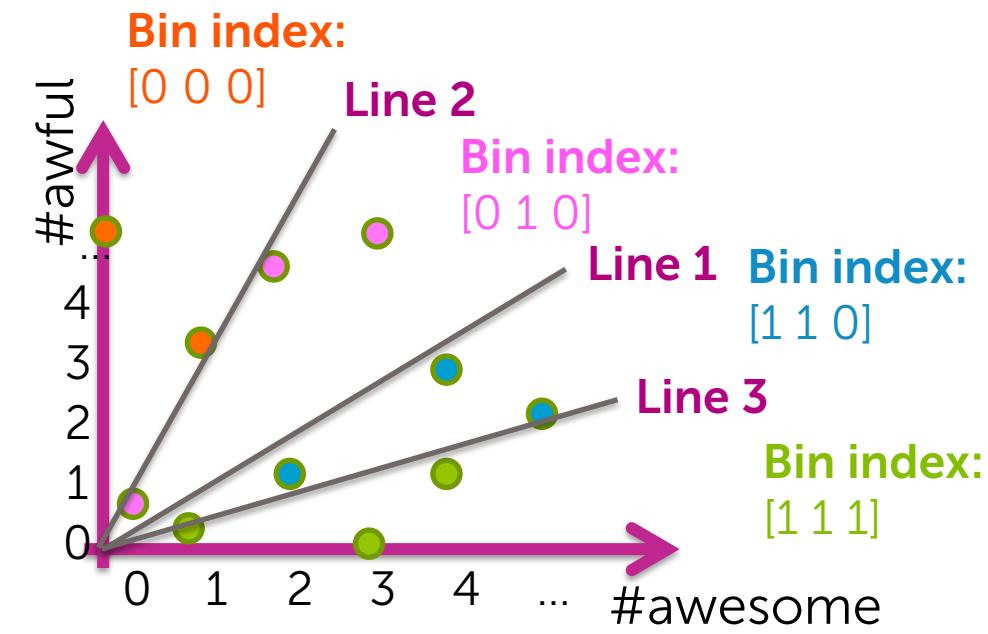
Prob. of being > 1 bit away

$$= 1 - \Pr(\text{same bin}) - \Pr(1 \text{ bin away})$$

$$= 1 - \Pr(\text{no split lines}) - \Pr(1 \text{ split line})$$

$$= 1 - (1 - \delta)^h - h\delta(1 - \delta)^{h-1}$$

Search $h+1$ bins and do not find NN
with probability $1 - (1 - \delta)^h - h\delta(1 - \delta)^{h-1}$



Probability of splitting neighboring points many times

Bin	[0 0 0] = 0	[0 0 1] = 1	[0 1 0] = 2	[0 1 1] = 3	[1 0 0] = 4	[1 0 1] = 5	[1 1 0] = 6	[1 1 1] = 7
indices	L0	L1	L2	L3	L4	L5	L6	L7

Bin	[0 0 0] = 0	[0 0 1] = 1	[0 1 0] = 2	[0 1 1] = 3	[1 0 0] = 4	[1 0 1] = 5	[1 1 0] = 6	[1 1 1] = 7
indices	L0	L1	L2	L3	L4	L5	L6	L7

Bin	[0 0 0] = 0	[0 0 1] = 1	[0 1 0] = 2	[0 1 1] = 3	[1 0 0] = 4	[1 0 1] = 5	[1 1 0] = 6	[1 1 1] = 7
indices	L0	L1	L2	L3	L4	L5	L6	L7

Bin	[0 0 0] = 0	[0 0 1] = 1	[0 1 0] = 2	[0 1 1] = 3	[1 0 0] = 4	[1 0 1] = 5	[1 1 0] = 6	[1 1 1] = 7
indices	L0	L1	L2	L3	L4	L5	L6	L7

Probability NN is
in different bin in
all $h+1$ tables

$$= (1 - \Pr(\text{same bin}))^{h+1}$$

$$= (1 - \Pr(\text{no split line}))^{h+1}$$

$$= (1 - (1 - \delta)^h)^{h+1}$$

prob. of no
split from any
of h lines thrown
down

holds
for all
hash
tables

Comparing approaches for h-bit tables

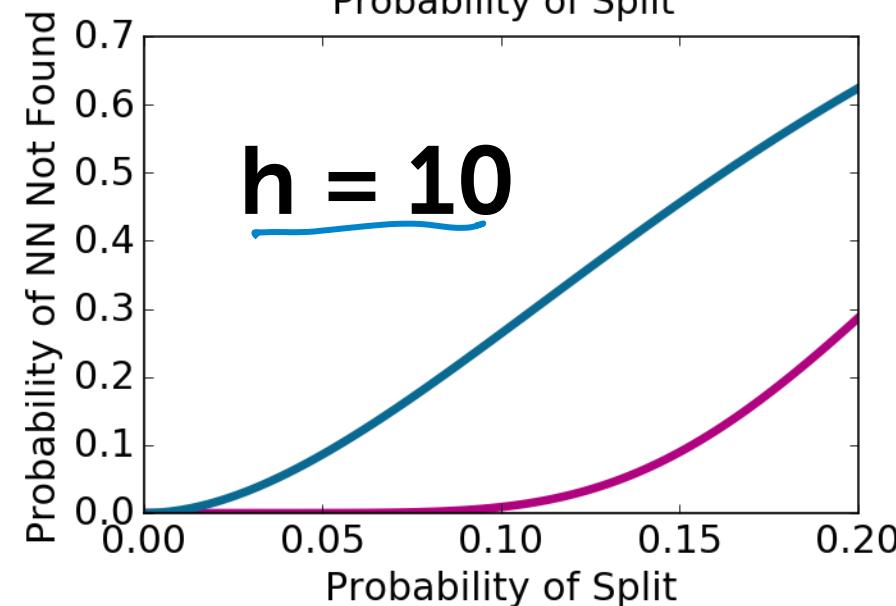
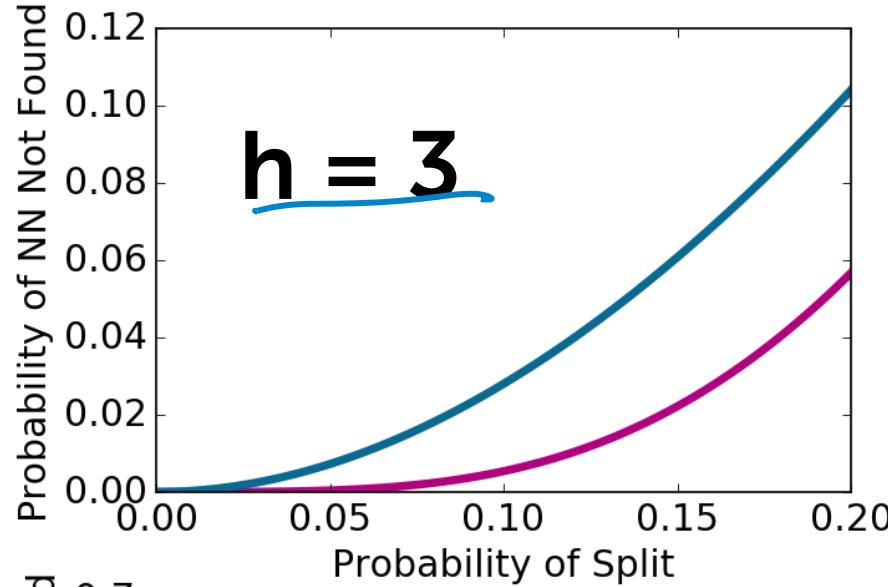
throw down lines

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3

# bins searched	prob. of no NN
$h+1$	$1-(1-\delta)^h - h\delta(1-\delta)^{h-1}$
$h+1$	$(1-(1-\delta)^h)^{h+1}$

Comparing probabilities



one hash table
 $1 - (1 - \delta)^h - h\delta(1 - \delta)^{h-1}$

$(1 - (1 - \delta)^h)^{h+1}$
multiple hash table

Fix #bits and increase depth

Bin	[0 0 0] = 0	[0 0 1] = 1	[0 1 0] = 2	[0 1 1] = 3	[1 0 0] = 4	[1 0 1] = 5	[1 1 0] = 6	[1 1 1] = 7
indices	L0	L1	L2	L3	L4	L5	L6	L7

Bin	[0 0 0] = 0	[0 0 1] = 1	[0 1 0] = 2	[0 1 1] = 3	[1 0 0] = 4	[1 0 1] = 5	[1 1 0] = 6	[1 1 1] = 7
indices	L0	L1	L2	L3	L4	L5	L6	L7

Bin	[0 0 0] = 0	[0 0 1] = 1	[0 1 0] = 2	[0 1 1] = 3	[1 0 0] = 4	[1 0 1] = 5	[1 1 0] = 6	[1 1 1] = 7
indices	L0	L1	L2	L3	L4	L5	L6	L7

Bin	[0 0 0] = 0	[0 0 1] = 1	[0 1 0] = 2	[0 1 1] = 3	[1 0 0] = 4	[1 0 1] = 5	[1 1 0] = 6	[1 1 1] = 7
indices	L0	L1	L2	L3	L4	L5	L6	L7

Bin	[0 0 0] = 0	[0 0 1] = 1	[0 1 0] = 2	[0 1 1] = 3	[1 0 0] = 4	[1 0 1] = 5	[1 1 0] = 6	[1 1 1] = 7
indices	L0	L1	L2	L3	L4	L5	L6	L7

Probability NN is in different bin in **all tables** falls off exponentially fast

hash tables
↓

Prob

$$= (1 - \text{Pr}(\text{same bin}))^{m \cdot b}$$

$$= (1 - \text{Pr}(\text{no split line}))^{m \cdot b}$$

$$= (1 - (1 - \delta)^b)^{m \cdot b}$$

of lines (cbits)

Fix #bits and increase depth

Bin	[0 0 0] = 0	[0 0 1] = 1	[0 1 0] = 2	[0 1 1] = 3	[1 0 0] = 4	[1 0 1] = 5	[1 1 0] = 6	[1 1 1] = 7
indices	L0	L1	L2	L3	L4	L5	L6	L7

Typically higher probability of finding NN than searching m bins in 1 table

Summary of LSH approaches

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3

Cost of binning points is **lower**,
but likely need to search
more bins per query

Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
Bin	[0 0] = 0	[0 1] = 1	[1 0] = 2	[1 1] = 3
indices	L0	L1	L2	L3

Cost of binning points is **higher**,
but likely need to search
fewer bins per query

Summary for retrieval using nearest neighbors, KD-trees, and locality sensitive hashing

What you can do now...

- Implement nearest neighbor search for retrieval tasks
- Contrast document representations (e.g., raw word counts, tf-idf,...)
 - Emphasize important words using tf-idf
- Contrast methods for measuring similarity between two documents
 - Euclidean vs. weighted Euclidean
 - Cosine similarity vs. similarity via unnormalized inner product
- Describe complexity of brute force search
- Implement KD-trees for nearest neighbor search
- Implement LSH for approximate nearest neighbor search
- Compare pros and cons of KD-trees and LSH, and decide which is more appropriate for given dataset