THE UNIVERSITY OF TEXAS AT AUSTIN
Department of Statistics and Data Sciences
College of Natural Sciences

SDS 385: Stat Models for Big Data

Lecture 9: KD trees

Purnamrita Sarkar
Department of Statistics and Data Science
The University of Texas at Austin

https://psarkar.github.io/teaching

Background

- Has a long history–invented in 1970 by Jon Bentley
- k represents the number of dimensions
- Idea is to partition the data spatially, by using only one dimension at any level.
- While searching, this helps pruning most of the search space.

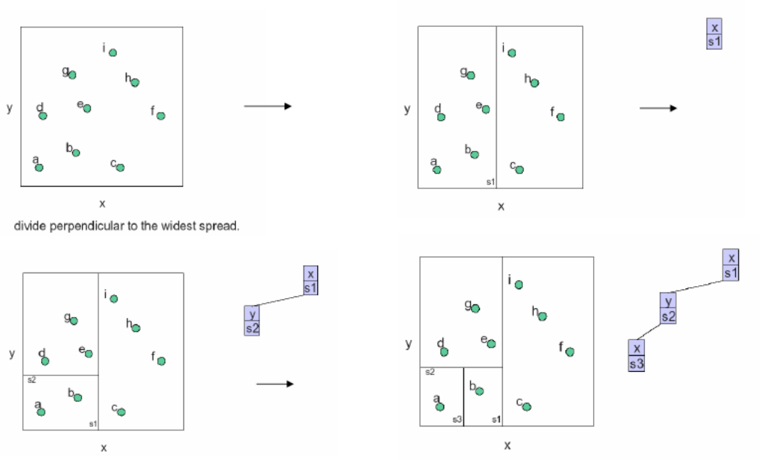
General idea

- Cycle through the dimensions for each level
- Call this cut-dim (cutting dimension)
- Node in tree contains $P = (x, y)$
- So, to find a point, only need to compare the cutting dimension.

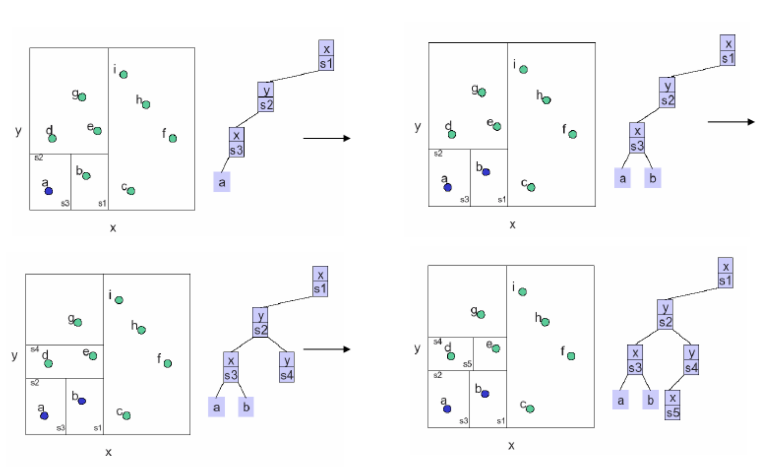
Construct

- If there is one point, just form a leaf node
- Otherwise divide the points in half along the cutting axis
  - Find the axis with the widest spread
  - divide in alternative/round robin fashion
- recursively build kdtrees from each half
- Complexity $dn \log n$

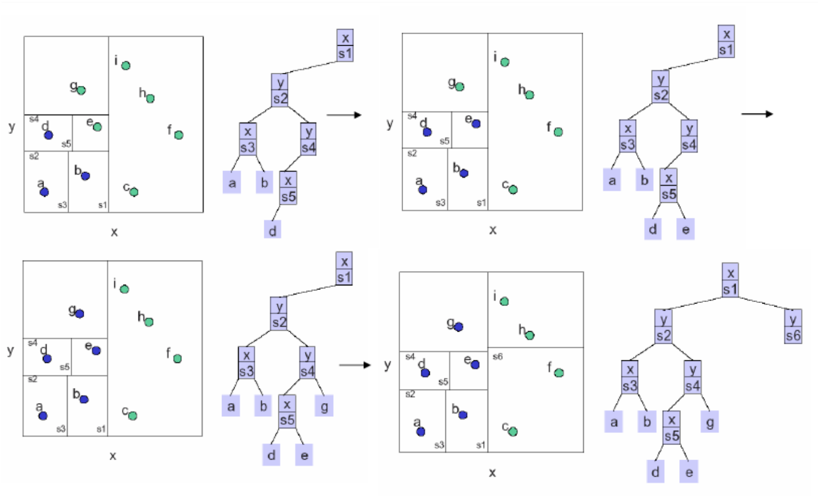
Insert



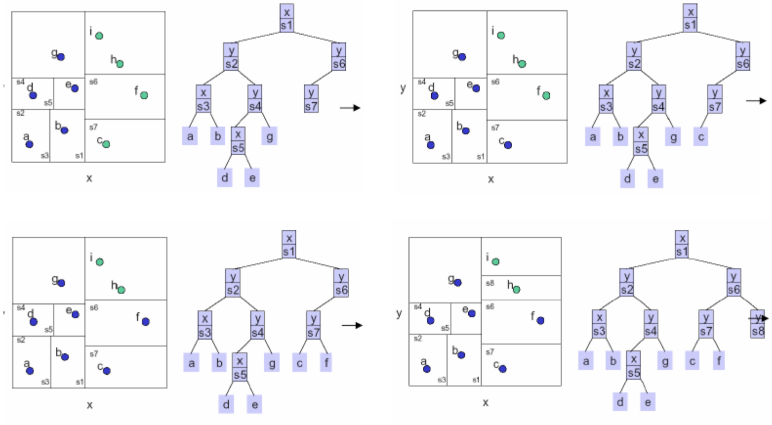
Insert



Insert



Insert



Find point with the smallest element in dimension a

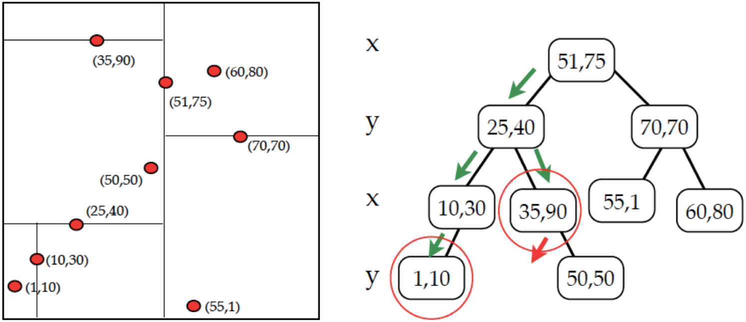
- If cutdim at current node equals a,
  - the min cannot be in the right subtree
  - recurse on the left subtree

Base case: if there are no left children, stop and return current point.

- Otherwise
  - the min could be in either
  - recurse on both left and right subtrees

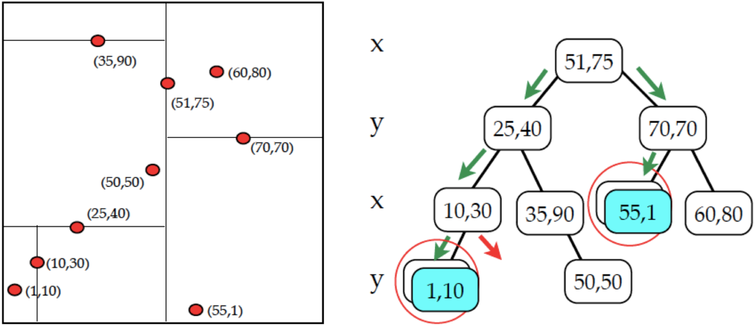
Find point with the smallest element in dimension x

FindMin(x-dimension):



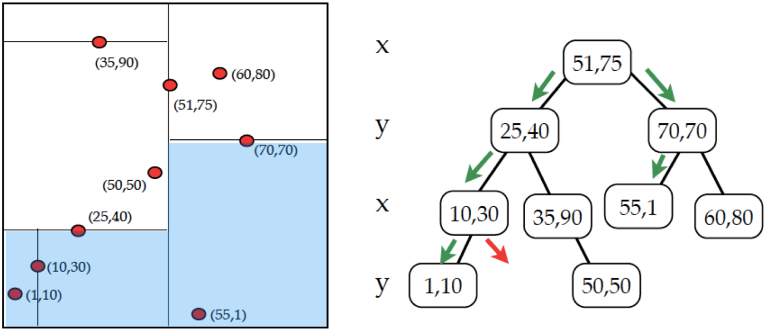
Find point with the smallest element in dimension y

FindMin(y-dimension):



Find point with the smallest element in dimension y

FindMin(y-dimension): space searched



Nearest neighbor queries

- Given point Q, find the closest point R
- Have to be careful, because its possible that two points are far away in the tree but close in the Eucidean space.
- For each node store a bounding box

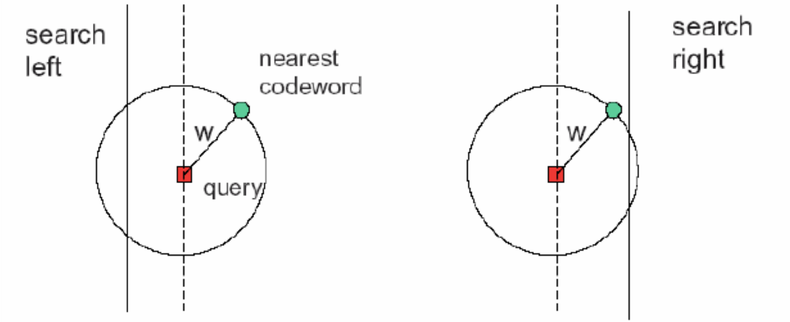
Nearest neighbor queries

- Given point Q, find the closest point R
- Have to be careful, because its possible that two points are far away in the tree but close in the Eucidean space.
- For each node store a bounding box
- Remember the closest point to Q seen so far (call this R')

Nearest neighbor queries

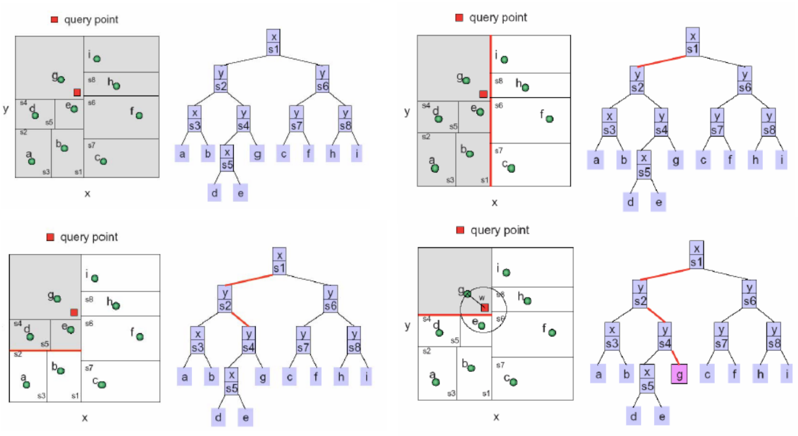
- Given point Q, find the closest point R
- Have to be careful, because its possible that two points are far away in the tree but close in the Eucidean space.
- For each node store a bounding box
- Remember the closest point to Q seen so far (call this R')
- Prune subtrees where bounding boxes cannot contain R'

Nearest neighbor queries

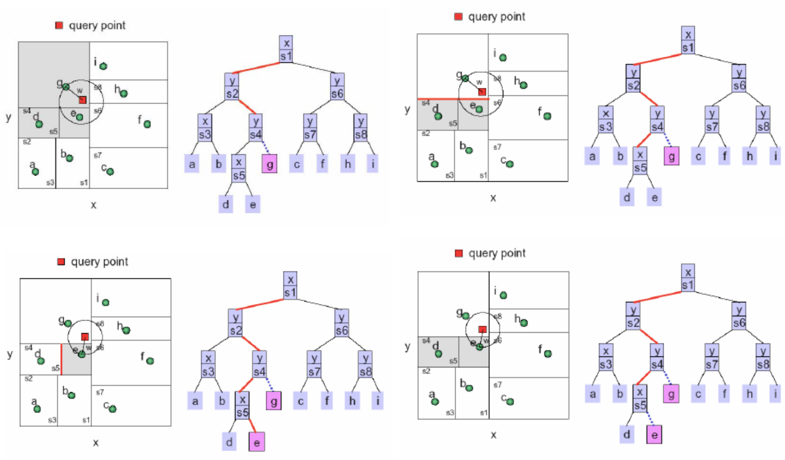


- If circle overlaps with left subtree, search left subtree
- If circle overlaps with right subtree search right subtree
- Has been shown to work in about $O(\log n)$ time.

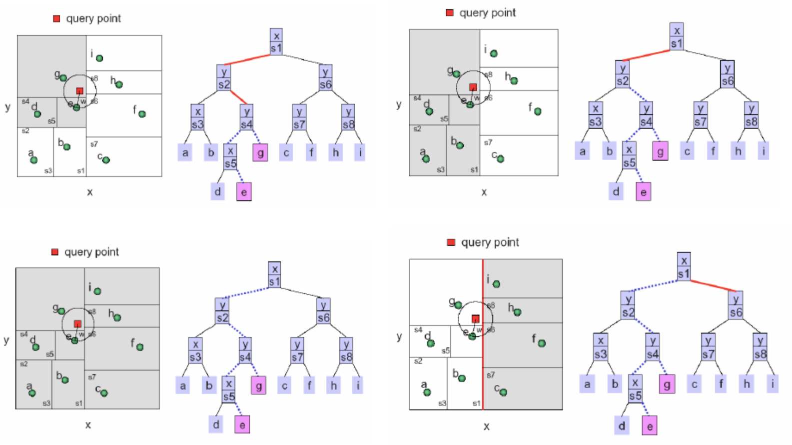
NN search



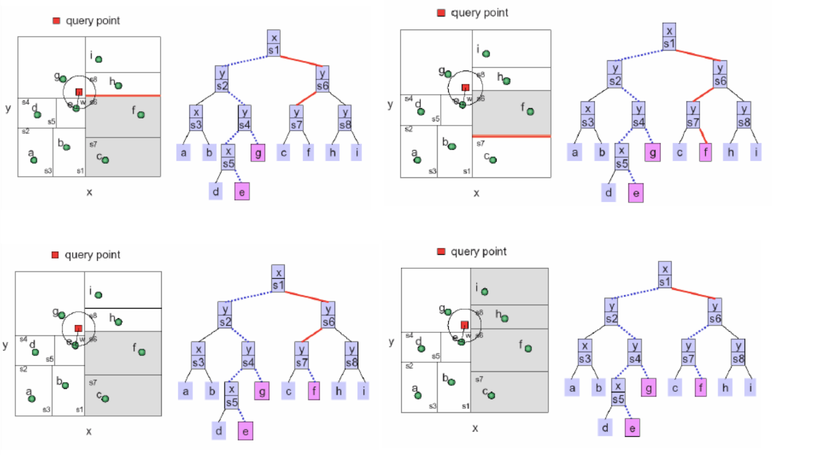
NN search



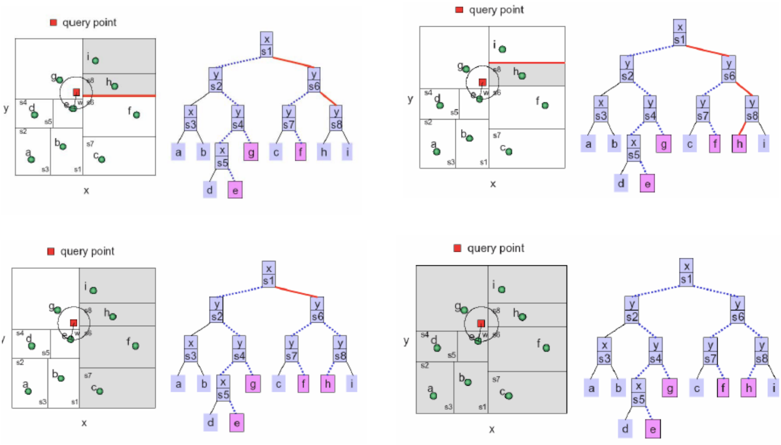
NN search



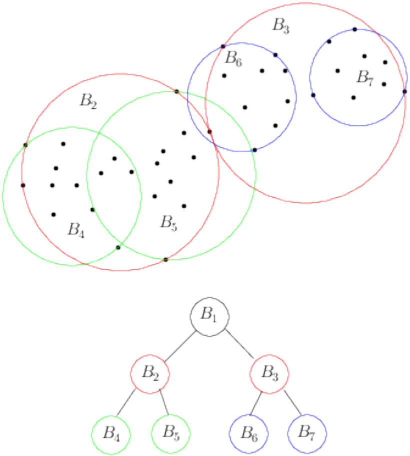
NN search



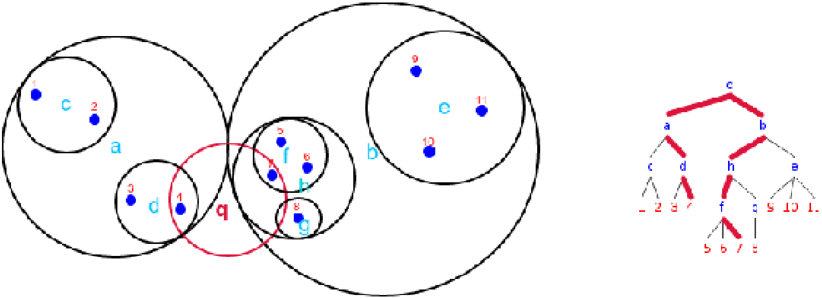
NN search



Ball trees



Ball tree search



(a) (b)

Acknowledgment

- Ullman's lecture notes from "Mining of Massive Datasets".
- Some slides from `http://infolab.stanford.edu/~ullman/mining/2009/similarity3.pdf`
- The S curve plot was taken from Scribe notes of EE381V at UT from Fall 2012