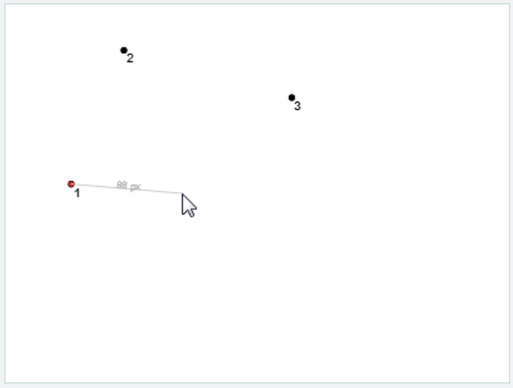
2D Nearest Neighbour Search

There are many data structures and algorithms to do a nearest neighbour search in low- and high-dimensional spaces. We've implemented some of them for a 2-dimensional case and created an interactive web application that allows everyone interested to try out those approaches for NNS.

http://nns.tume-maailm.pri.ee

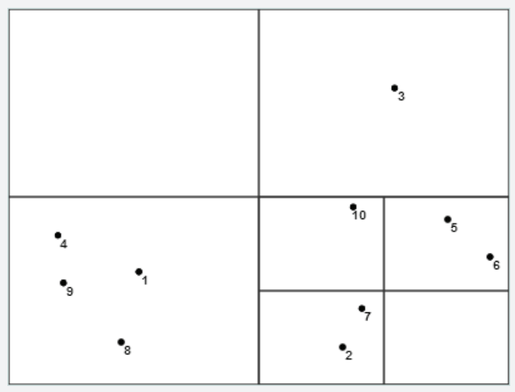
Linear Search

The most basic technique that one might even call the brute-force solution. Linear search calculates distances from query point to every point in the dataset.



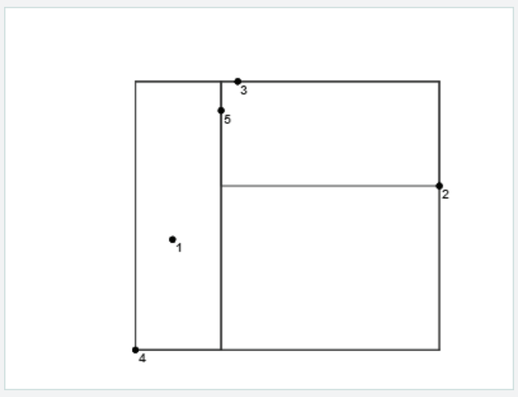
Quadtree

One of the more common solutions for a low-dimensional case. Quadtree divides the space into 4 equal parts recursively until the datapoint count in one node is above some threshold.



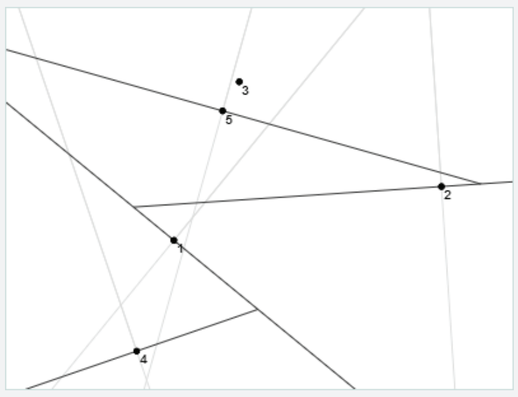
K-Dimensional Tree

Another quite common approach in low-dimensional space. Improvement over a quadtree comes because space is now divided based on the dataset. On every level space is divided into 2 from the median at an alternating axis.



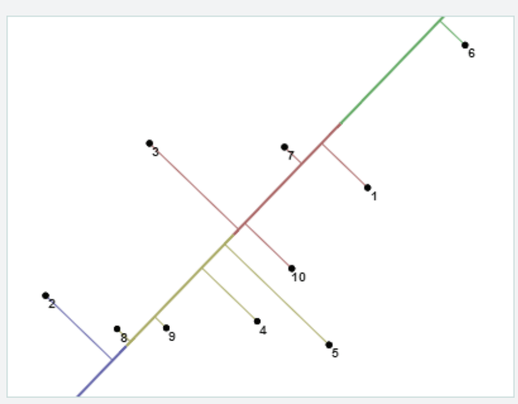
Random-Projection Tree

Following the K-D tree's approach of dividing the space into 2 in regard to the actual dataset, RP tree will divide via a randomly directed projection. This causes the nearest-neighbour search to be probabalistic. With high probability we can get an accurate result in logarithmic time.



Locality-Sensitive Hashing

This method uses a number of hash-tables with some bucket width to collect together nearby datapoints in random projections. When finding a nearest neighbour, we hash the query point and only search among the collisions in our random projection hash-tables.



Authors

Raimond Tunnel
Anastassia Soikonen
Jordan Valdma

Master's curriculum of Computer Science
Institute of Computer Science
Faculty of Mathematics and Computer Science
University of Tartu

Project is managed in Assembla.com
https://www.assembla.com/spaces/2d-nearest-neighbour-search

TARTU ÜLIKOOL · UNIVERSITAS TARTUENSIS · 1632

Study IT in .ee