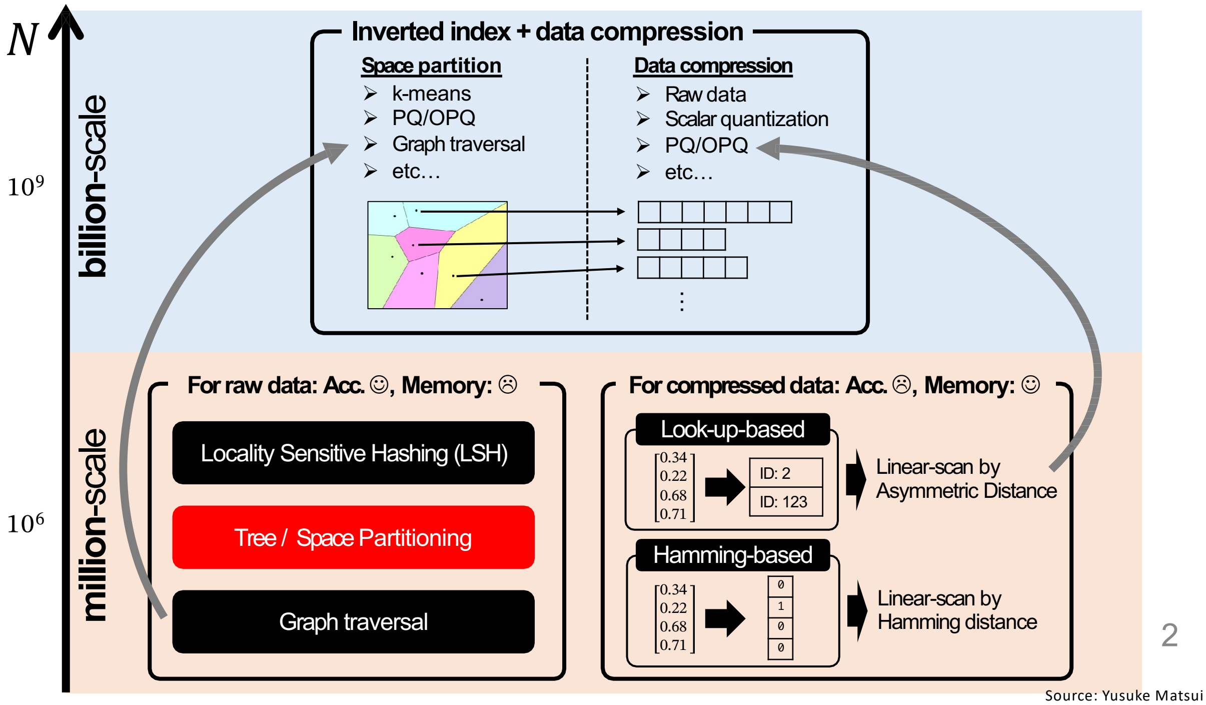
Approximate Nearest Neighbor Search

Tree and Quantization based approaches



Inverted index + data compression

Space partition
- k-means
- PQ/OPQ
- Graph traversal
- etc…

Data compression
- Raw data
- Scalar quantization
- PQ/OPQ
- etc…

For raw data: Acc. ☺, Memory: ☹

Locality Sensitive Hashing (LSH)

Tree / Space Partitioning

Graph traversal

For compressed data: Acc. ☹, Memory: ☺

Look-up-based

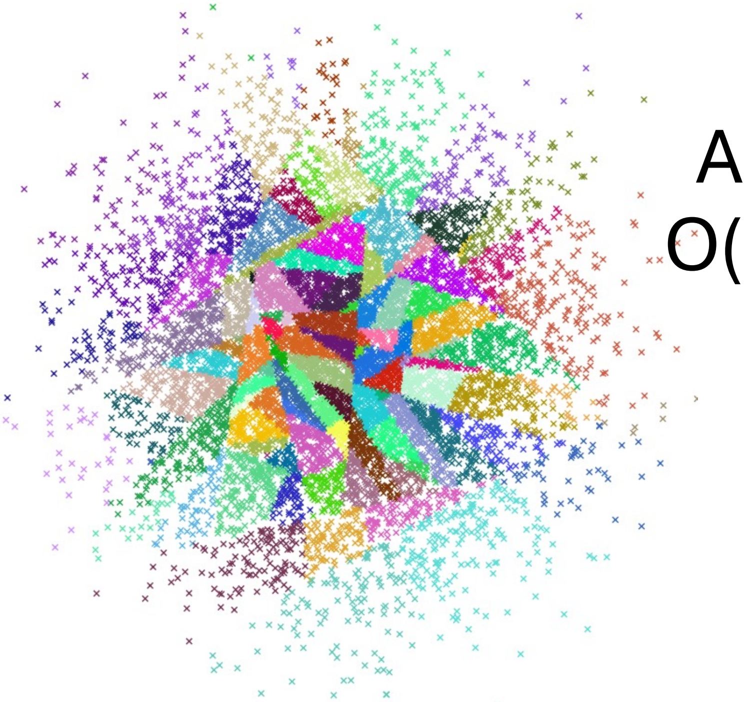
$\begin{bmatrix}0.34\\0.22\\0.68\\0.71\end{bmatrix}$ → ID: 2, ID: 123 → Linear-scan by Asymmetric Distance

Hamming-based

$\begin{bmatrix}0.34\\0.22\\0.68\\0.71\end{bmatrix}$ → 0, 1, 0, 0 → Linear-scan by Hamming distance

2

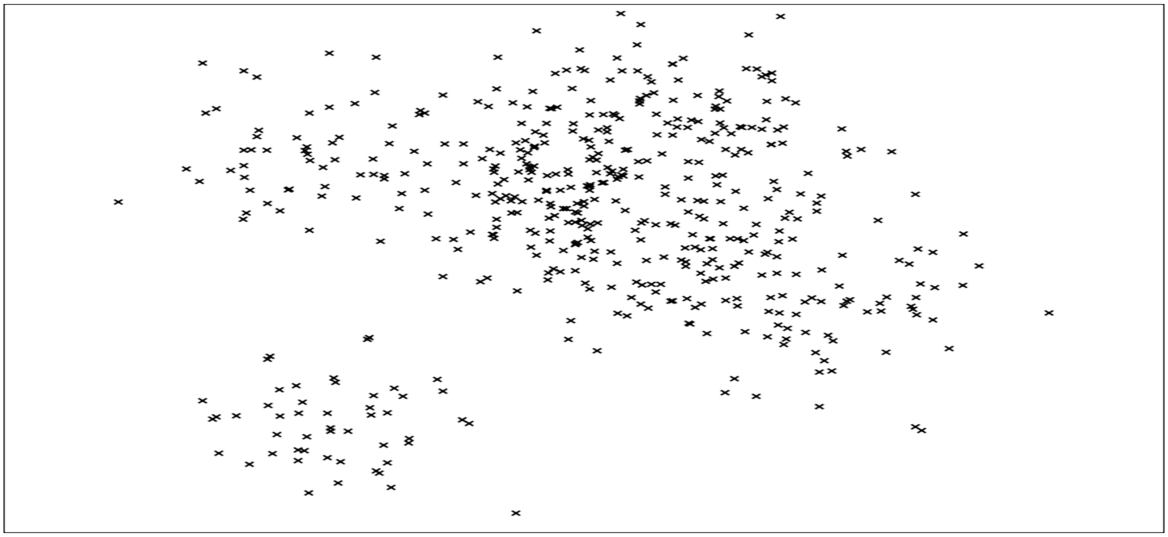
Source: Yusuke Matsui



ANNOY – from O(nd) to O(log n)

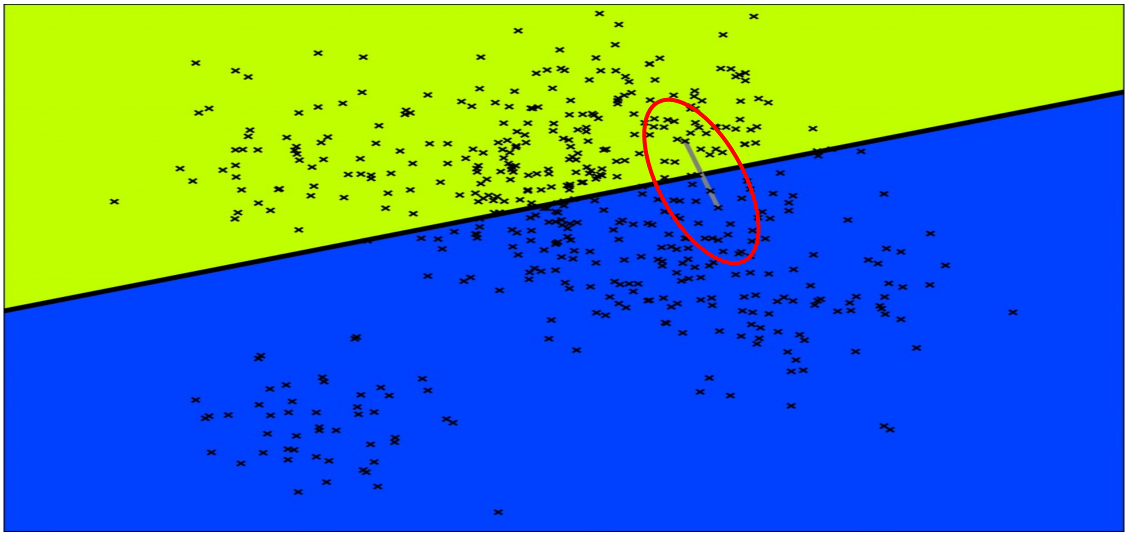
Source: Erik Bernhardsson

How do you build a tree from these vectors?



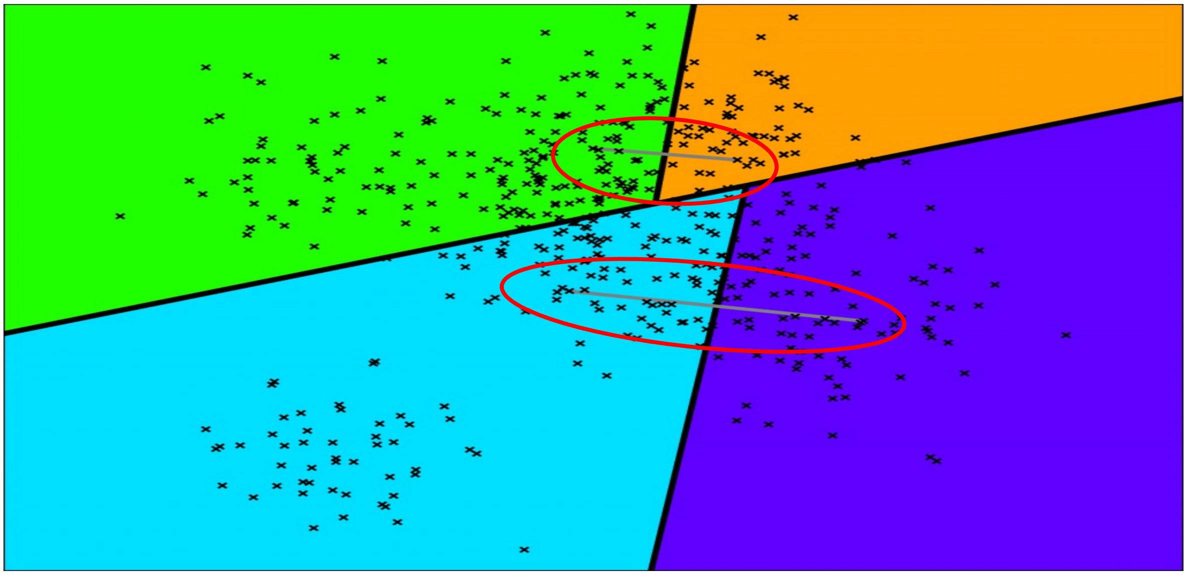
Source: Erik Bernhardsson

Pick two points randomly; split the feature space by the hyperplane equidistant from the two points.



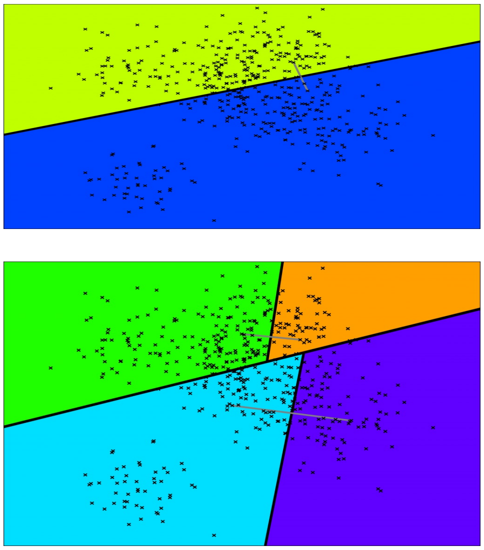
Source: Erik Bernhardsson

Split each subspace recursively



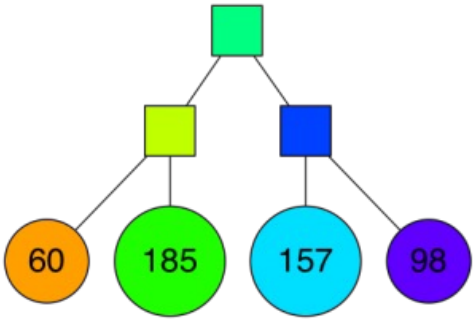
Source: Erik Bernhardsson

Split each subspace recursively



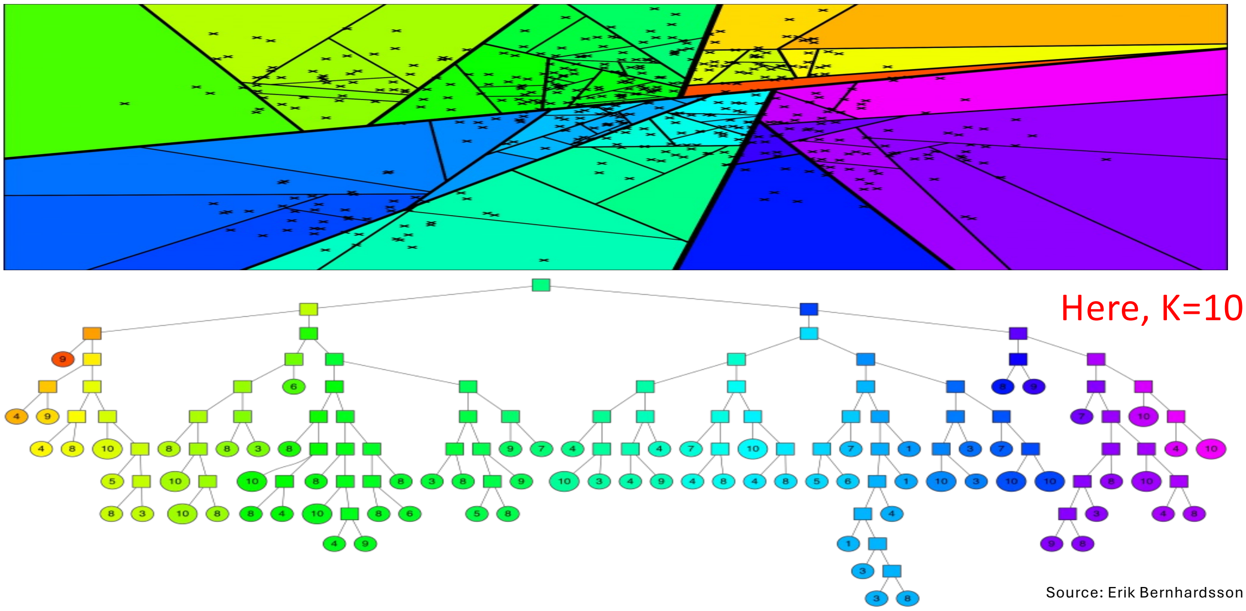
and the tree starts to evolve

(Intermediate node defines a hyperplane)



Source: Erik Bernhardsson

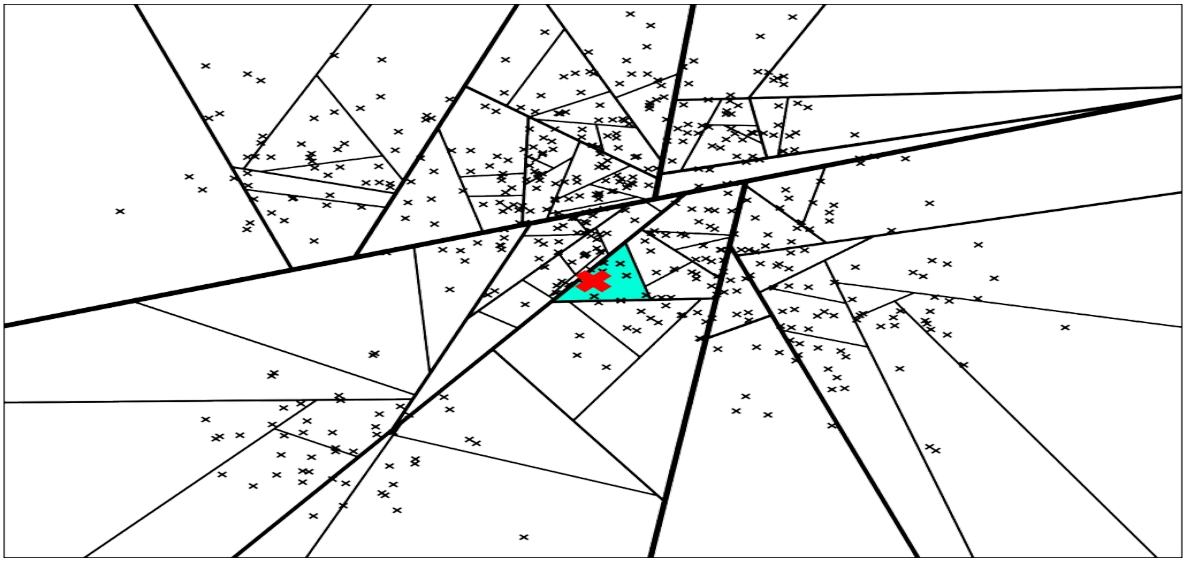
Repeat until at most K items are left in each node



Here, K=10

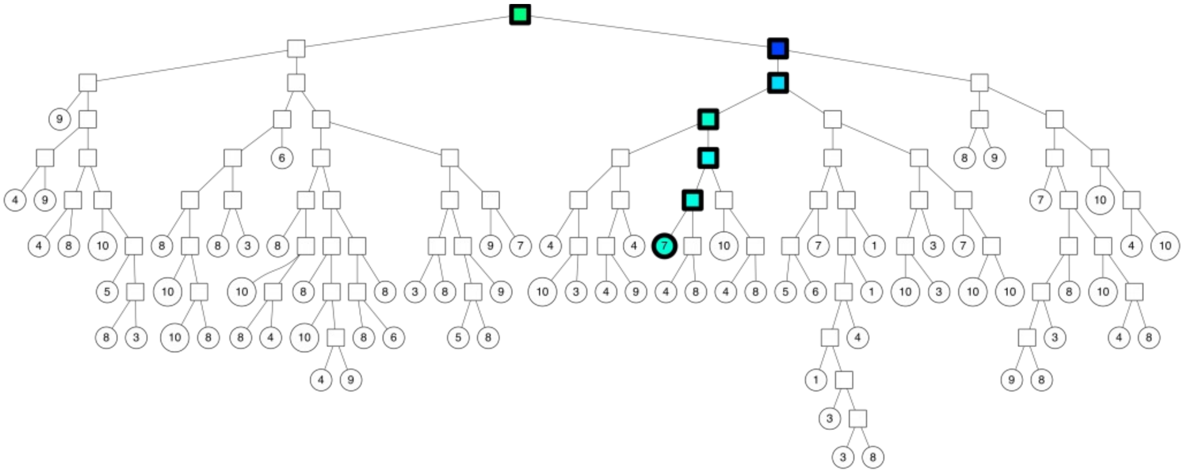
Source: Erik Bernhardsson

How do you find documents similar to a query?



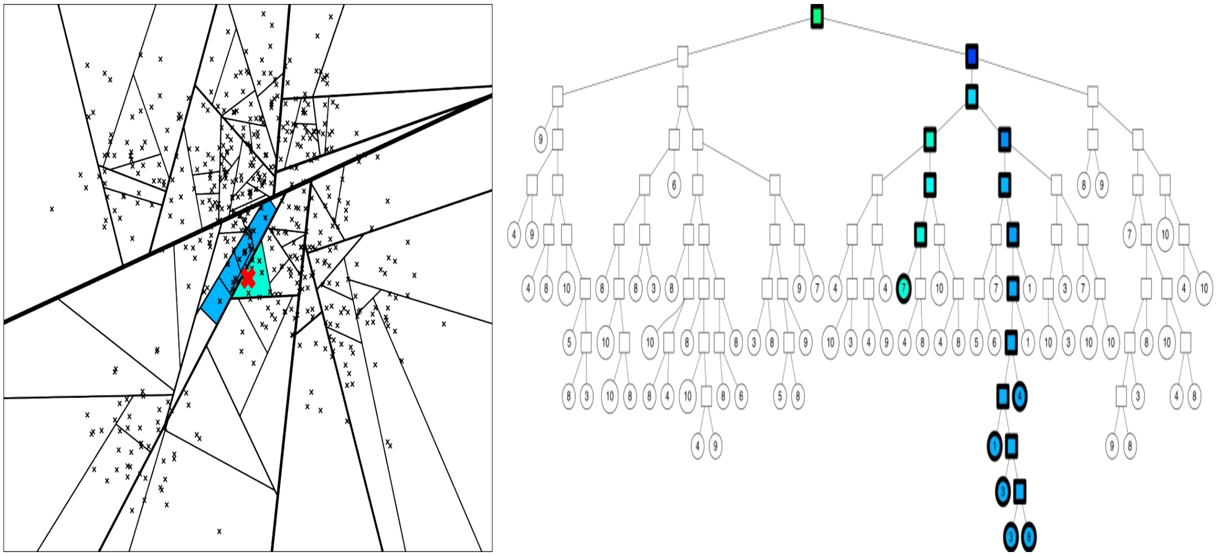
Source: Erik Bernhardsson

How do you search the tree?



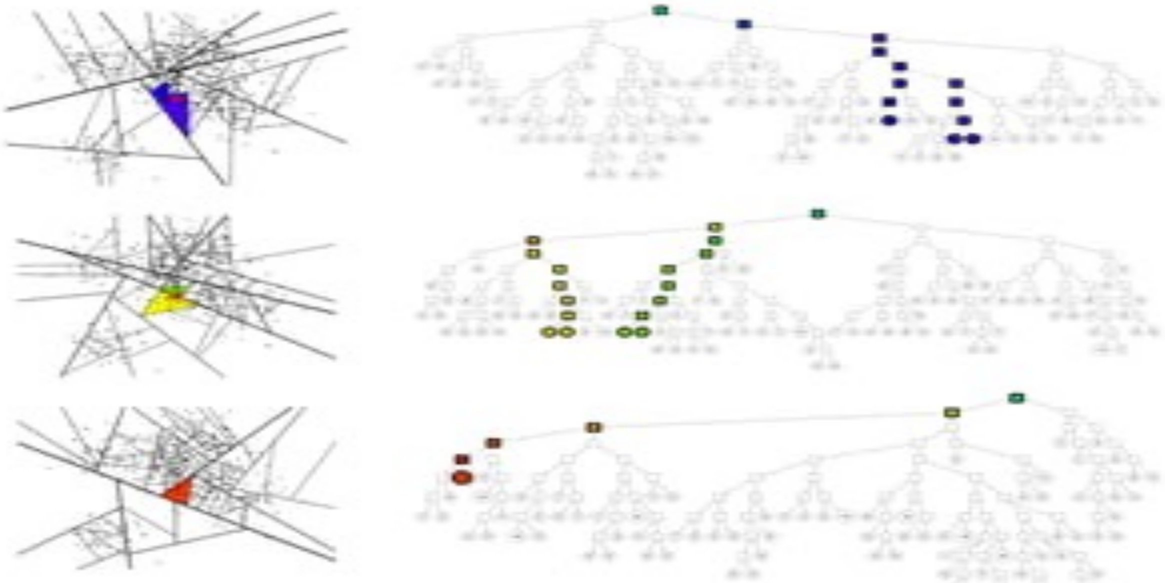
Source: Erik Bernhardsson

A better solution: priority queue



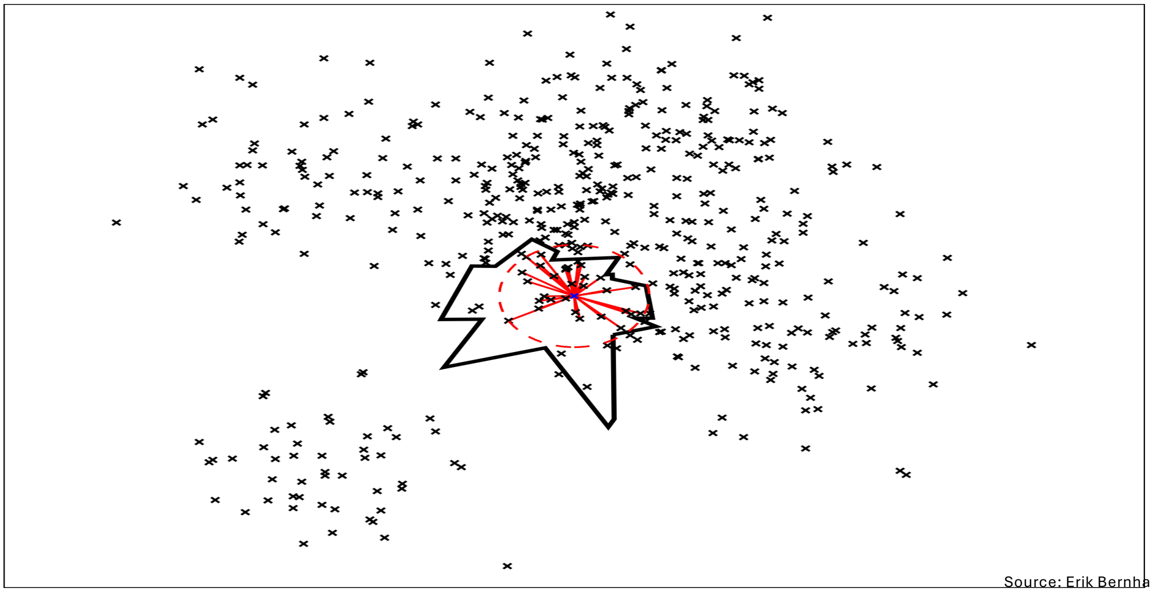
Source: Erik Bernhardsson

Even better solution: build a forest of trees

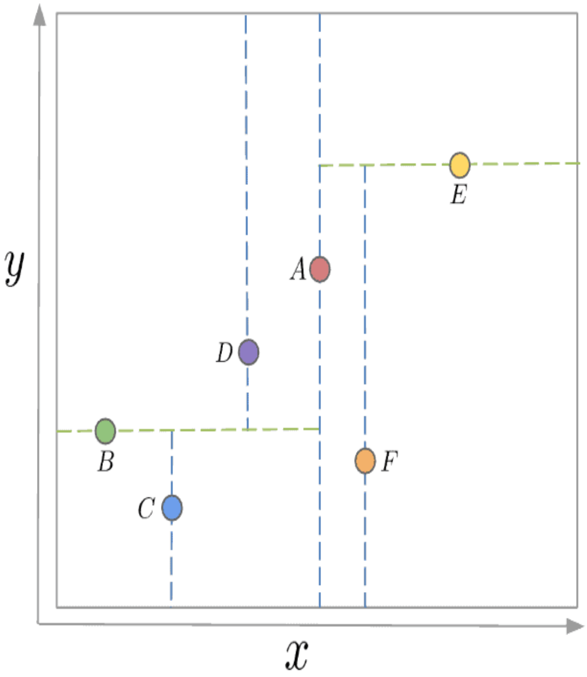


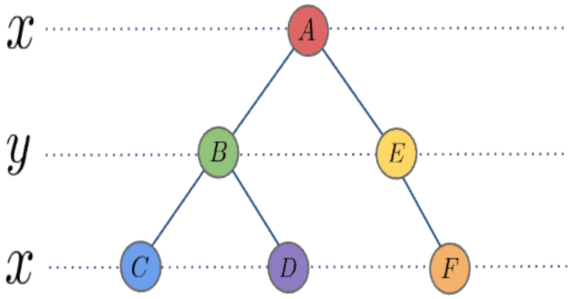
Source: Erik Bernhardsson

Even better solution: build a forest of trees



Source: Erik Bernhardsson

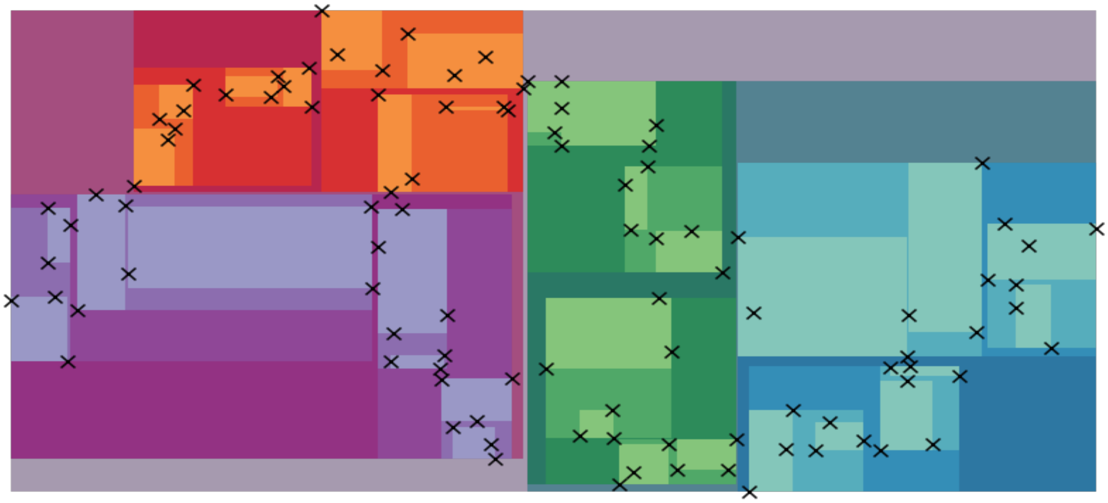




K-D Tree

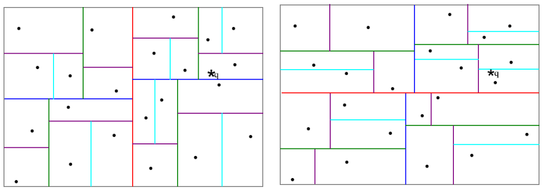
Source: https://www.baeldung.com

KD-Tree

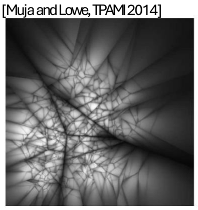


FLANN: Fast Library for Approximate Nearest Neighbors

Images are from [Muja and Lowe, TPAMI 2014]



Randomized KD Tree



k-means Tree

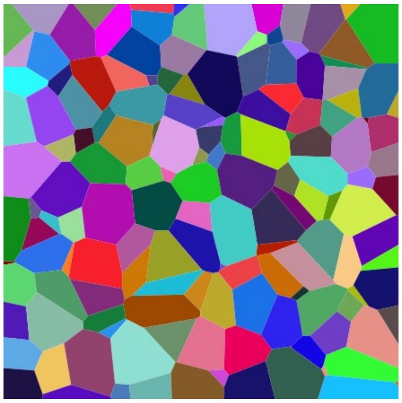
➢ Automatically select “Randomized KD Tree” or “k-means Tree”
https://github.com/mariusmuja/flann

☺ Good code base. Implemented in OpenCV and PCL
☺ Very popular in the late 00's and early 10’s
☹ Large memory consumption. The original data need to be stored
☹ Not actively maintained now

Source: Yusuke Matsui

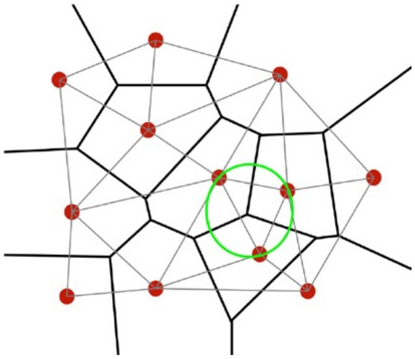
Inverted File (IVF)

IVF : Intuition



This Photo by Unknown Author is licensed under CC BY-SA

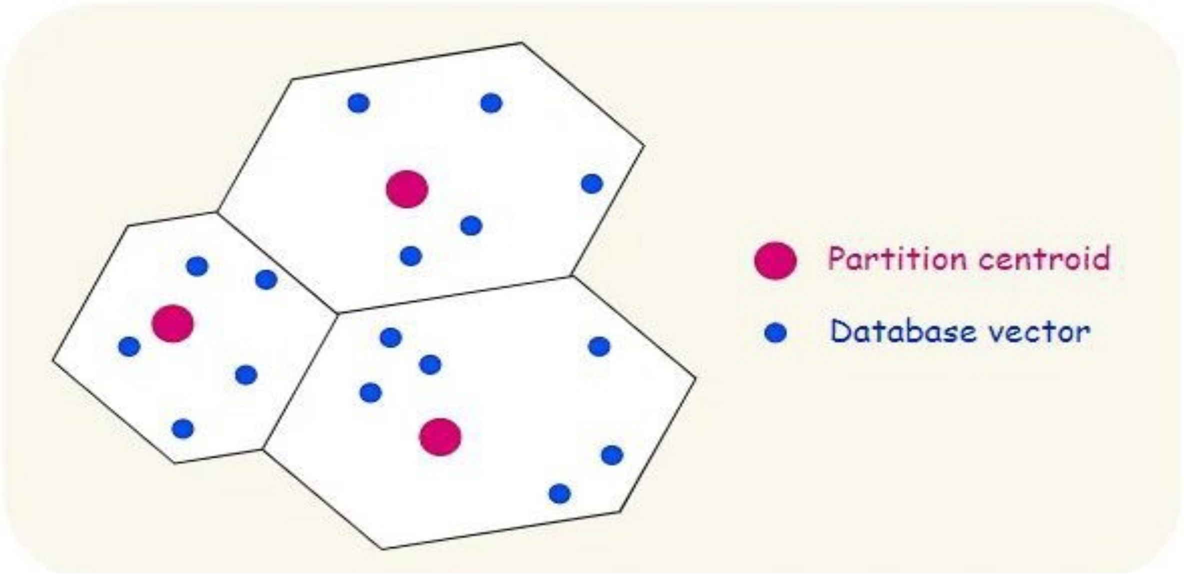
1-NN: VORONOI DIAGRAM



This Photo by Unknown Author is licensed under CC BY-SA

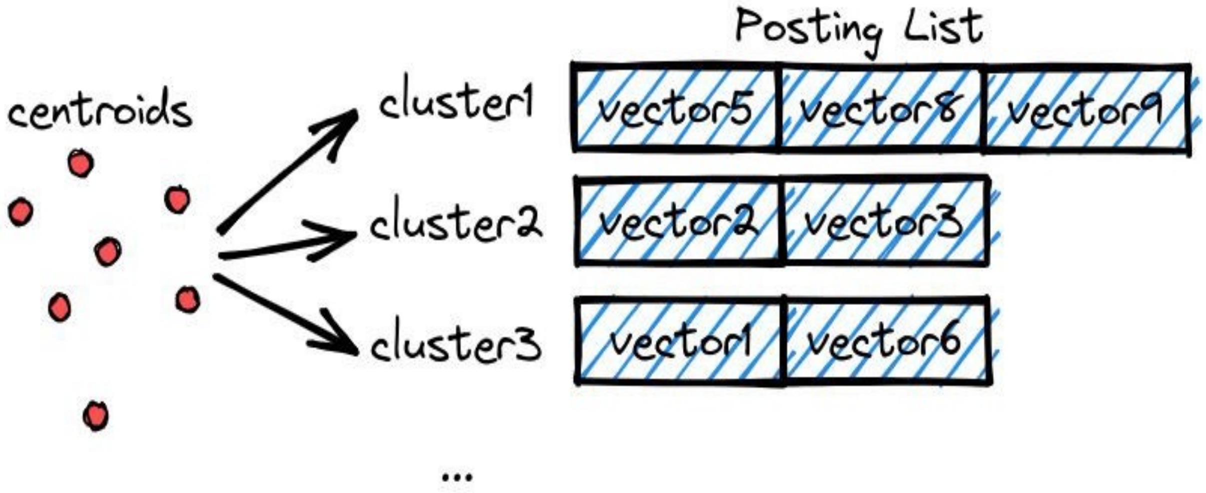
Visualization of the Induced Decision Boundary

Inverted File Index (IVF): Centroids represent clusters



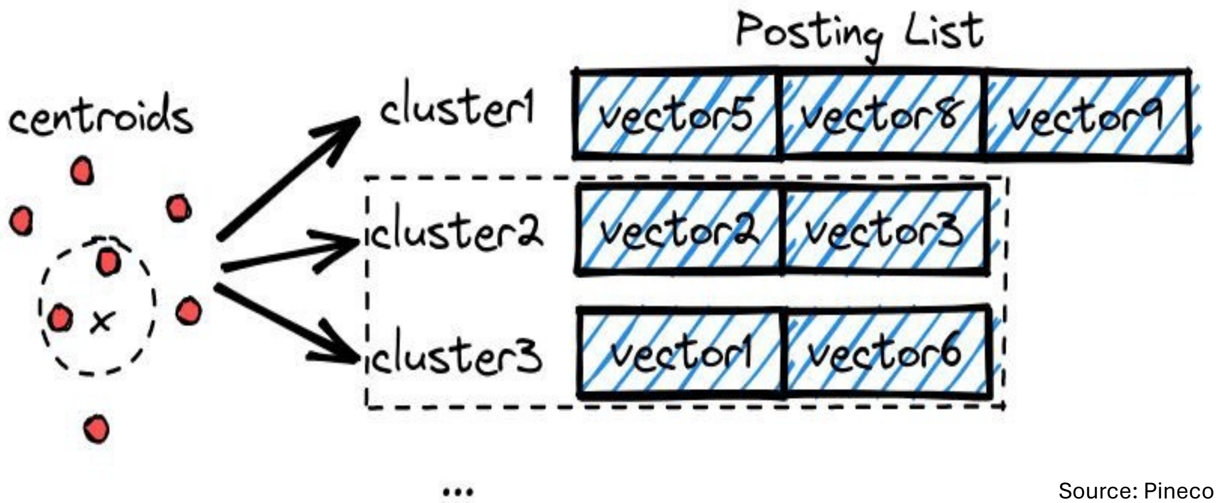
Source: Pinecone

The inverted index is from centroids to the vectors in each cluster



Source: Pinecone

To search, find closest centroids and search in the corresponding clusters



Source: Pinecone

Why IVF?

- Faster to build; the index size is much smaller
- However, search is slower than in HNSW $O(\log N)\ vs\ O(sqrt(N))$
- HNSW has a better recall as well
- A better approach: In IVF, search the nearest centroids using HNSW for better recall!

Product Quantization

A Space and Computation saving technique

Product Quantization

How many vectors of floats are possible in a vector space?

Infinite – each element of the vector can be any of the infinite floats

How can we reduce the number of possible vectors (scope)?

Approximate the floats by a representative finite range of integers

In k-means, what is the representative vector in each cluster?

Centroid – the “mean” for each of the k-clusters

What if each of the element (dimension / feature) of the original vector is mapped to the number of the closest centroid in that dimension?