Nearest neighbor search in high dimensional spaces

Ilya Fedorov

March 2020

Content

- Why do we need to optimize KNN?
- Data structures
- Complexity analysis
- Difficulties in high dimensional spaces
- Approximate NN search: Locality Sensitive Hashing
- Approximate NN search: FlyHash
- Approximate NN search: Hierarchical Navigable Small World

Why do we need to optimize KNN?

- Very large datasets
- Very frequent queries
- Duplicates search
- It seems that brute force algorithm doens't use information gained from calculating previous distances: if $d(a, b)$ is big and $d(a, c)$ is small does it mean that $d(c, b)$ is big?

Data structures

- KD-Tree
- Ball-tree
- Ball*-tree
- R-Tree
- etc...

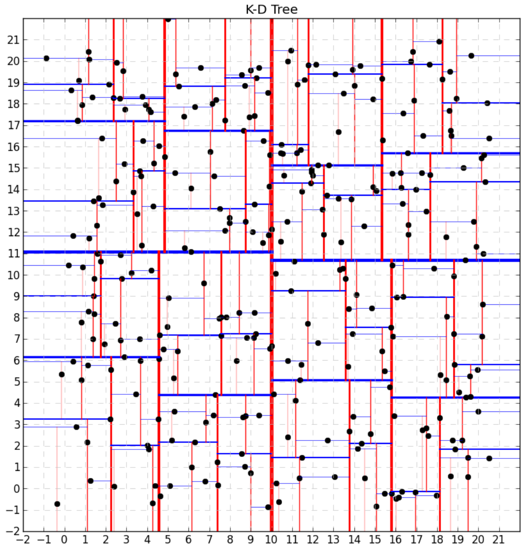
algorithm : *{'auto', 'ball_tree', 'kd_tree', 'brute'}, optional*

Algorithm used to compute the nearest neighbors:

- 'ball_tree' will use `BallTree`
- 'kd_tree' will use `KDTree`
- 'brute' will use a brute-force search.
- 'auto' will attempt to decide the most appropriate algorithm based on the values passed to `fit` method.

Note: fitting on sparse input will override the setting of this parameter, using brute force.

Data structures

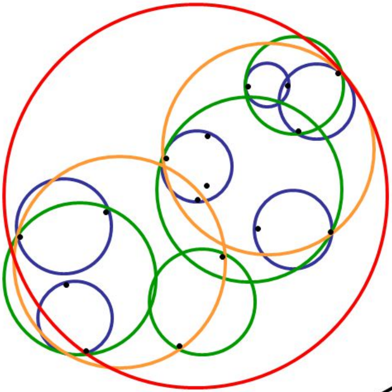


Data structures

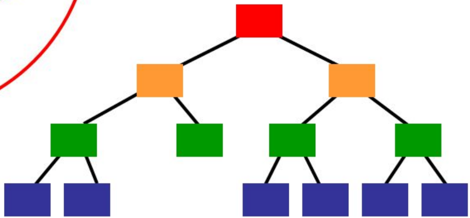
- Tree construction - by $build\_node(\{(x_1, y_1), ...(x_N, y_N)\})$:

```
build_node(Ω):  # omega-objects in the node
   if |Ω| < n_min:
      node.objects = Ω
   else:
      find feature with maximal spread in Ω:
         x^i = arg max_{x^i} σ(x^i)
      find median Ω: μ = median{x^i} # yields balanced tree
      node.feature = i
      node.threshold = μ
      node.left child =
            build_node({x_k ∈ Ω : x_k^i < μ})
      node.right child =
            build_node({x_k ∈ Ω : x_k^i ≥ μ})
   return node
```

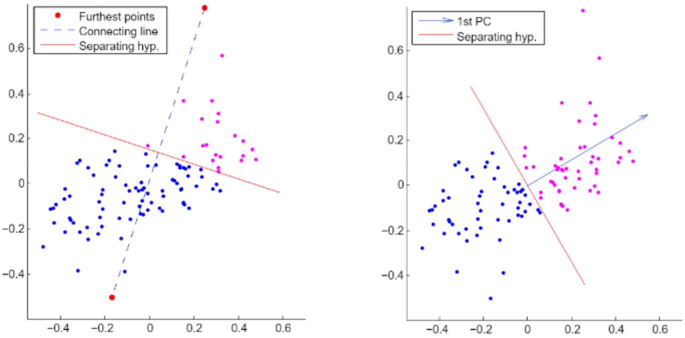
Data structures



A ball-tree: level 4



Data structures



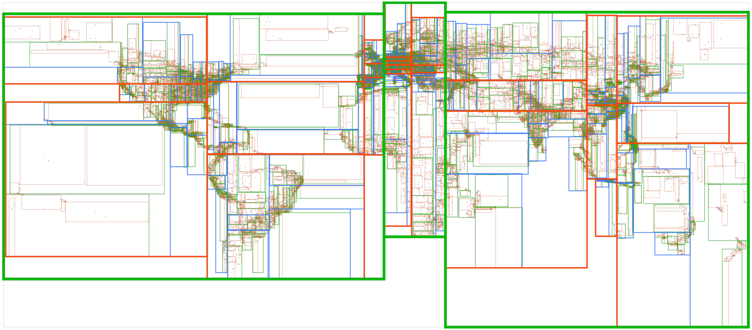
(a) Ball-tree (b) Ball*-tree

Fig. 2: Comparison of the splitting algorithms in ball-tree and ball*-tree

Dolatshah, Mohamad Hadian, Ali Minaei, Behrouz. (2015). Ball*-tree: Efficient spatial indexing for constrained nearest-neighbor search in metric spaces.

Data structures

R-tree



Complexity analysis