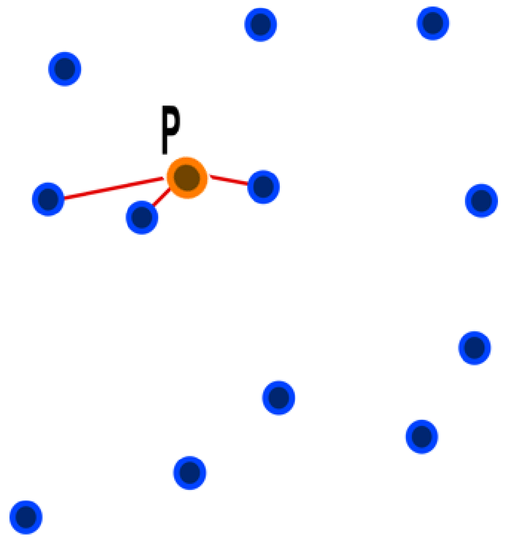
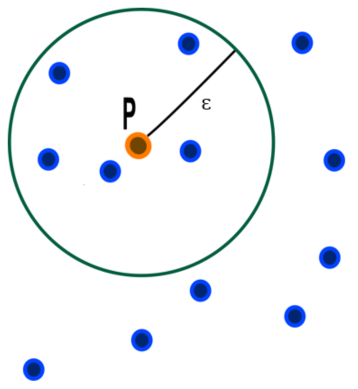
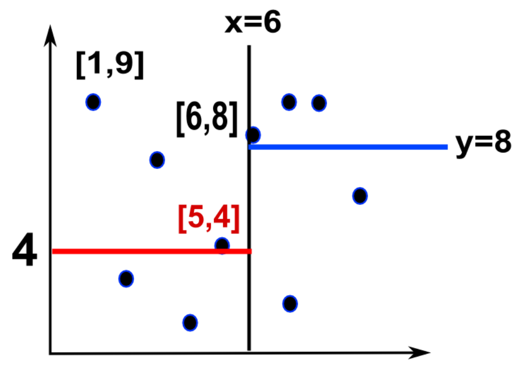




Neighborhood Graphs



Mustafa Hajij

Finding Nearest neighbors

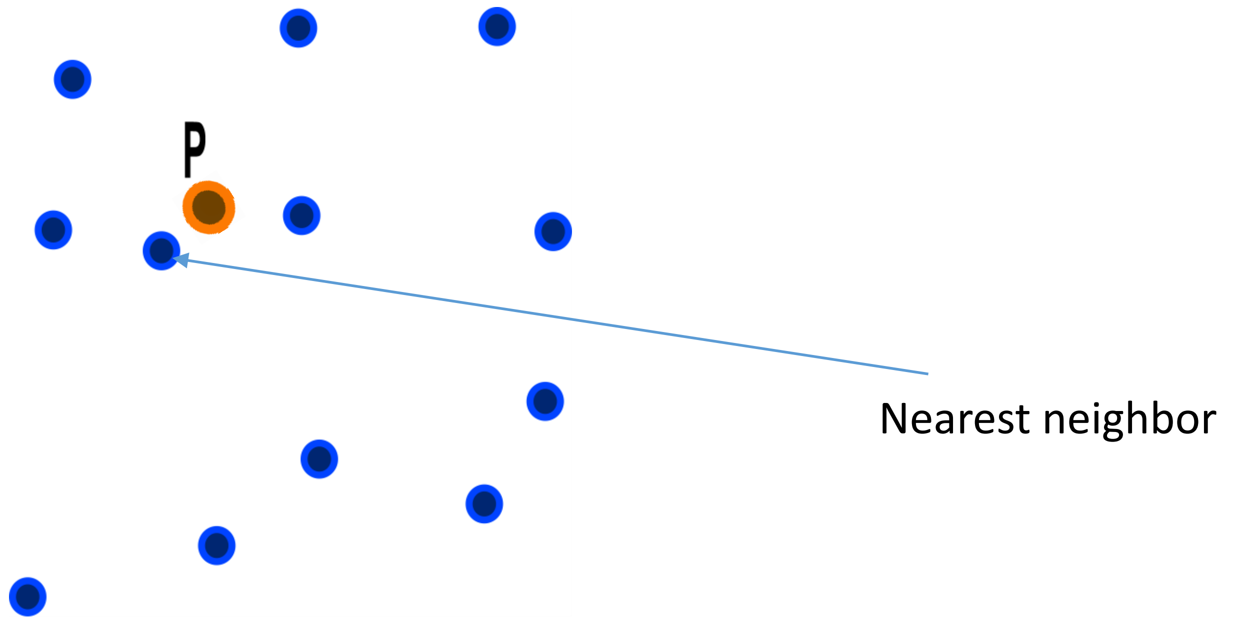
Given: a set X of n points in R^d and distance function d. We are looking for efficient algorithms to for the following :

- Nearest Neighbor: for any query p, returns a point $x \in X$ minimizing $d(p, x)$.

- ε -Near Neighbor: for any query p, returns all points $x \in X$ s.t. $d(p, x) \leq$ ε (if they exist)

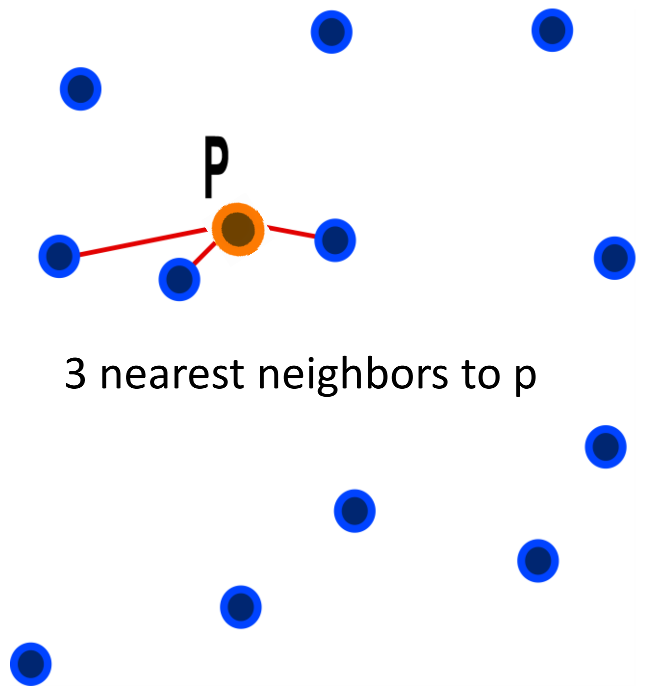
Nearest neighbor

Nearest Neighbor: for any query p, returns a point $x \in X$ minimizing $d(p, x)$



Write a brute-force code for this algorithm

K-Nearest neighbors



3 nearest neighbors to p

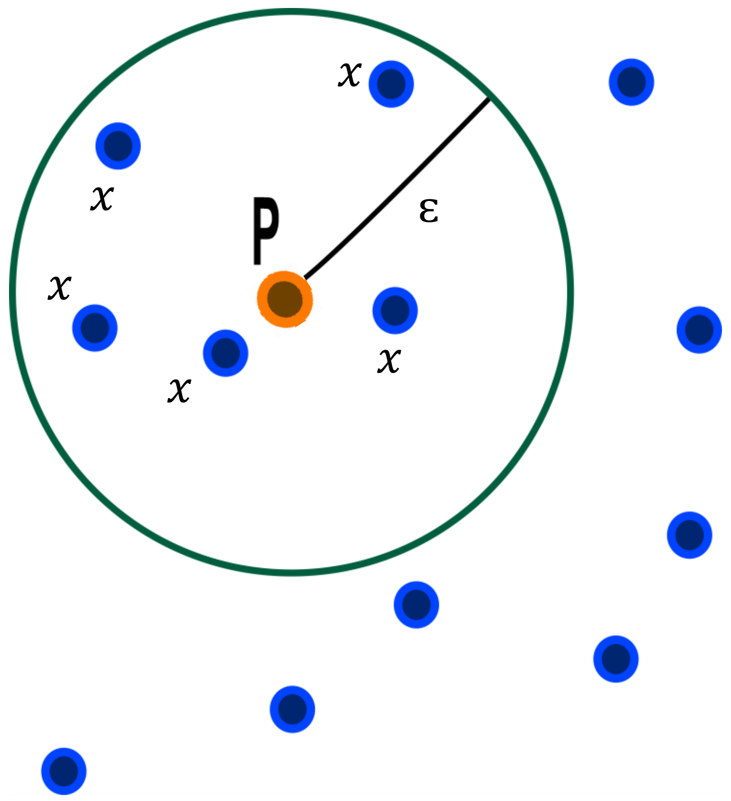
Application : if p is an document, then the
The set of k nearest neighbors of p represent
The k most similar articles to p

Write a brute-force code for this algorithm

Sklearn example

Finding ε –Nearest neighbors

Finding all points x that that satisfy $\mathrm{d}(p, x) \leq \varepsilon$



Write a brute-force code for this algorithm

Sklearn example

Algorithms?

2d tree

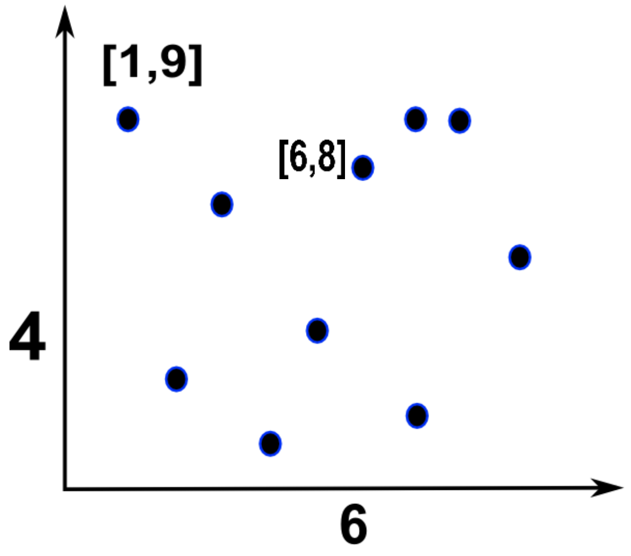
Suppose that we are given a data such as the data X=[[1,9],[2,3],[4,1],[3,7],[5,4],[6,8],[7,2],[8,8],[7,9],[9,6]].

KD-Tree organizes a data set such as the data X as a tree for the propose of fast data inquiry.

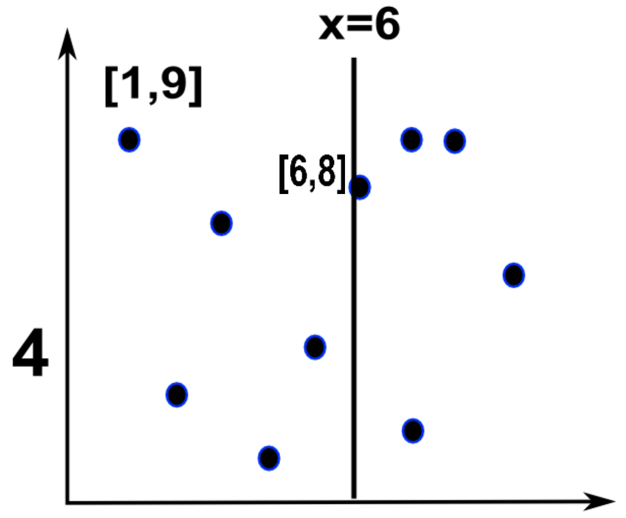
2d-tree: Example

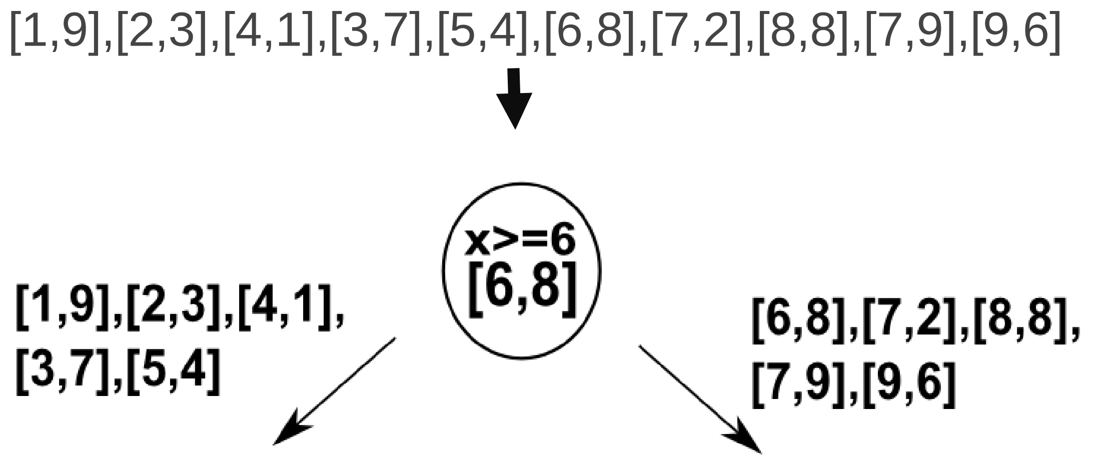
2D-Tree organizes a data set such as the data X as a tree for the propose of fast data inquiry.

[1,9],[2,3],[4,1],[3,7],[5,4],[6,8],[7,2],[8,8],[7,9],[9,6]

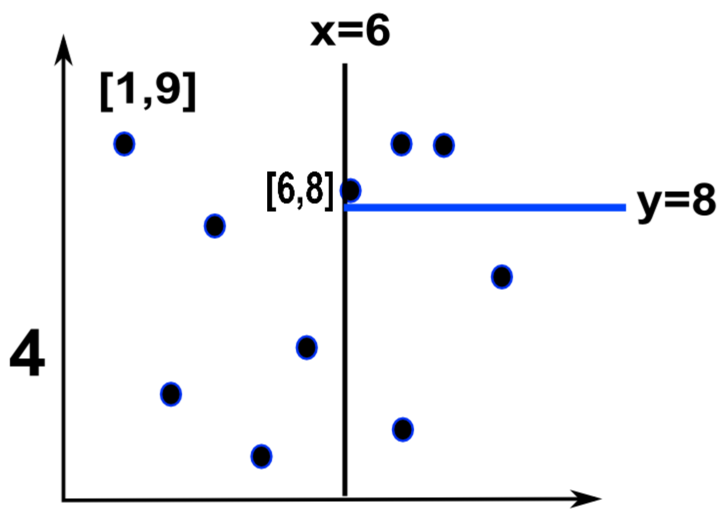


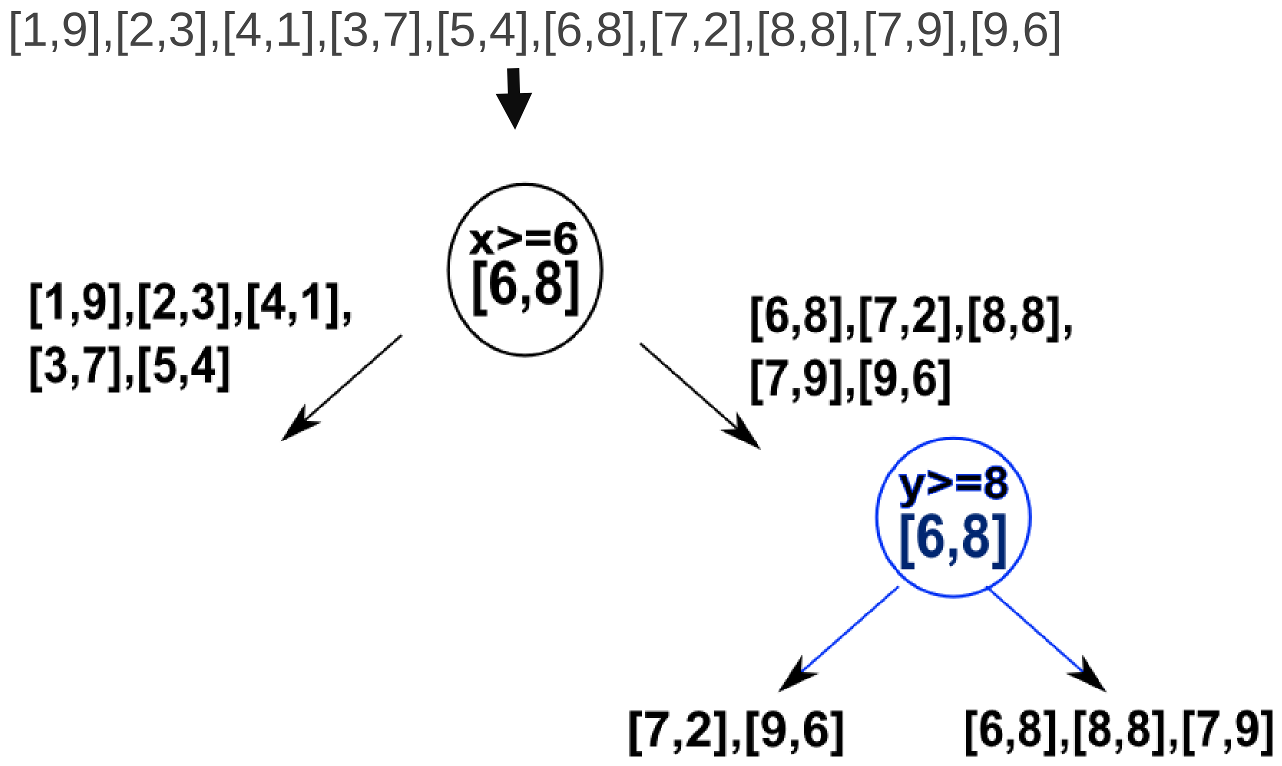
2d-tree: Example



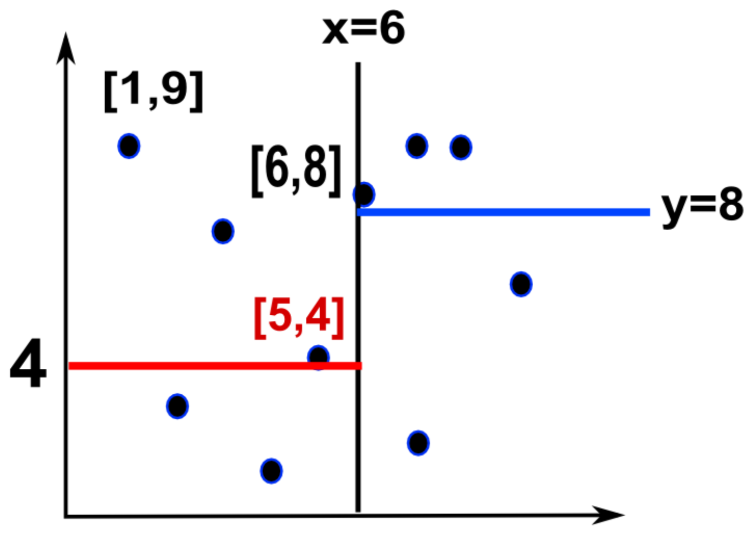


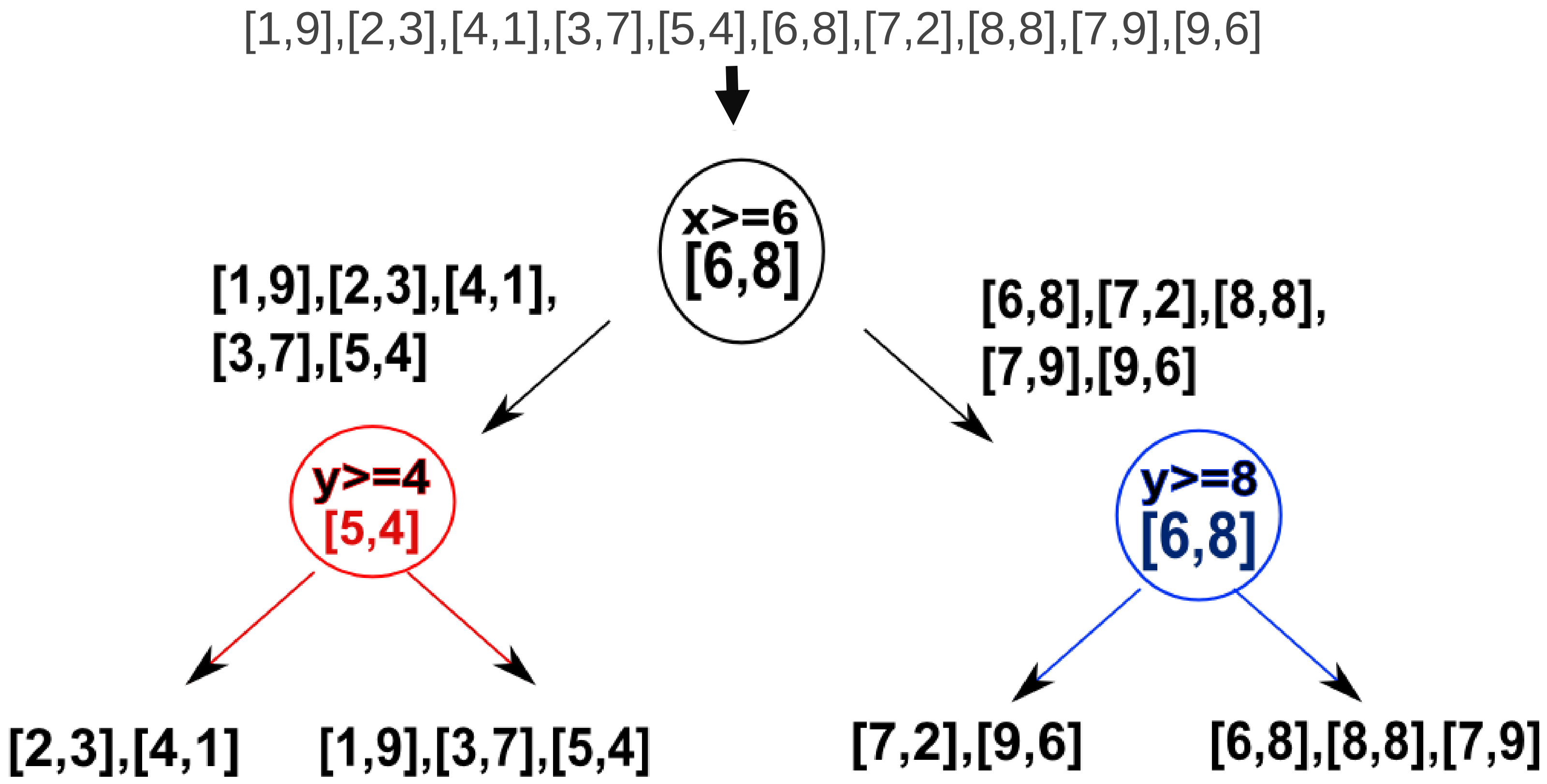
2d-tree: Example





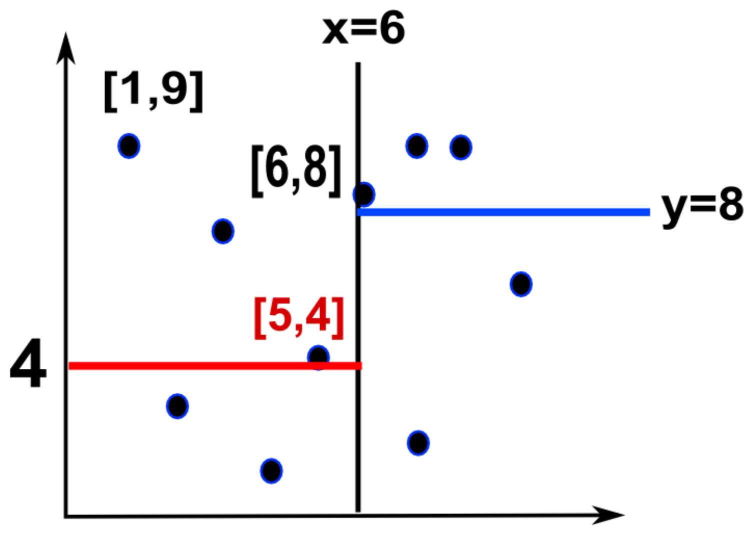
2d-tree: Example

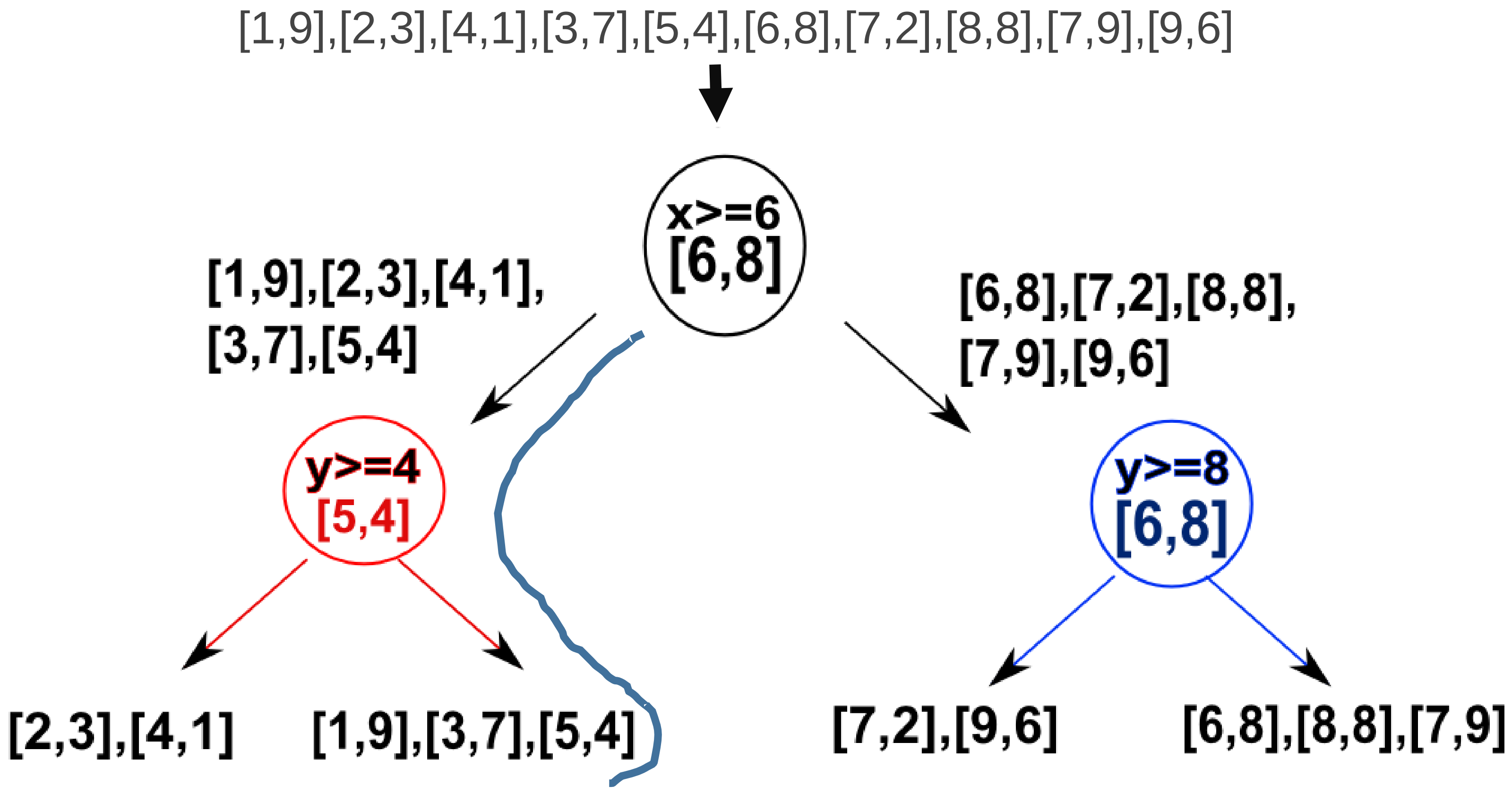




sklearn

2d-tree: Example



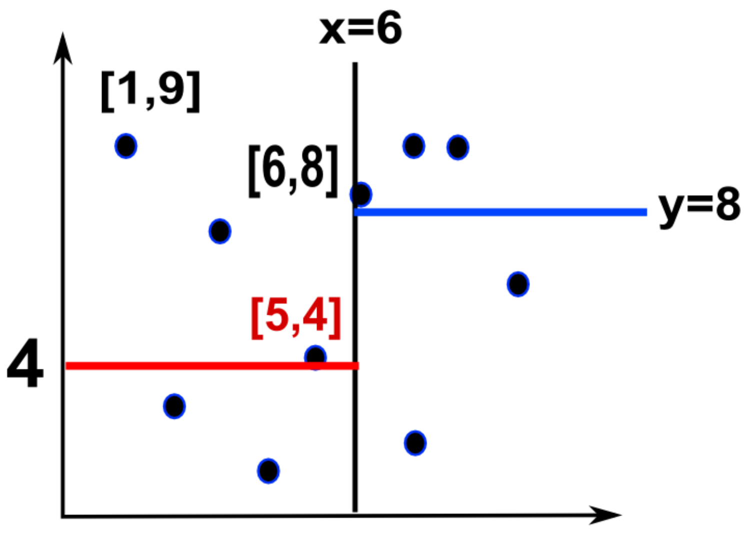


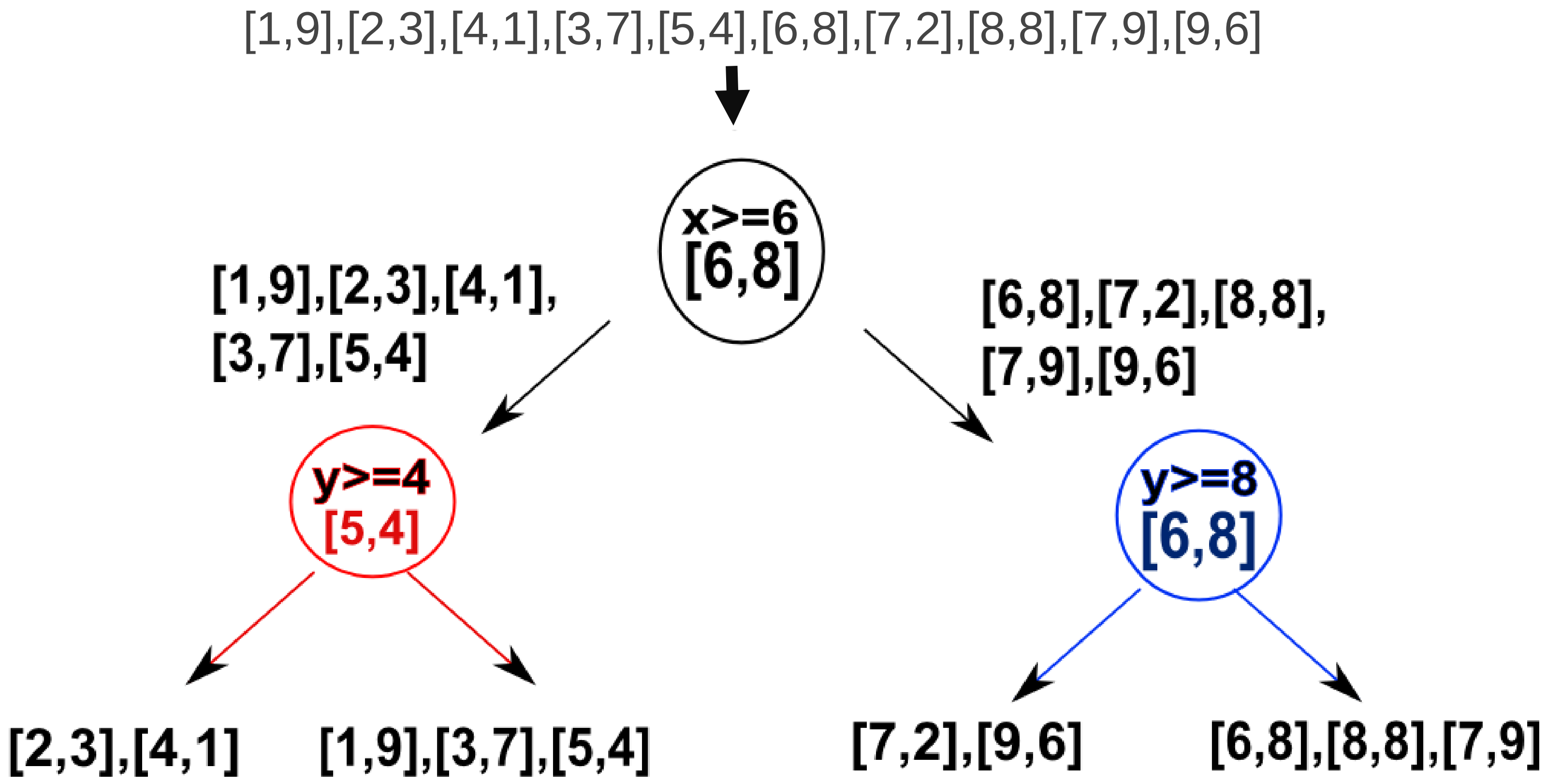
Points here satisfy all the conditions as we travel the tree from top to the leaf

sklearn

2d-tree: Example

What information do we store at the each node in the tree ?





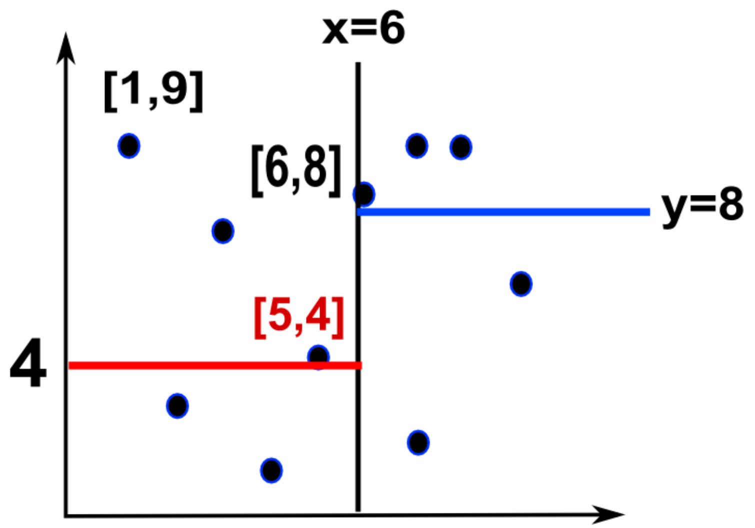
sklearn

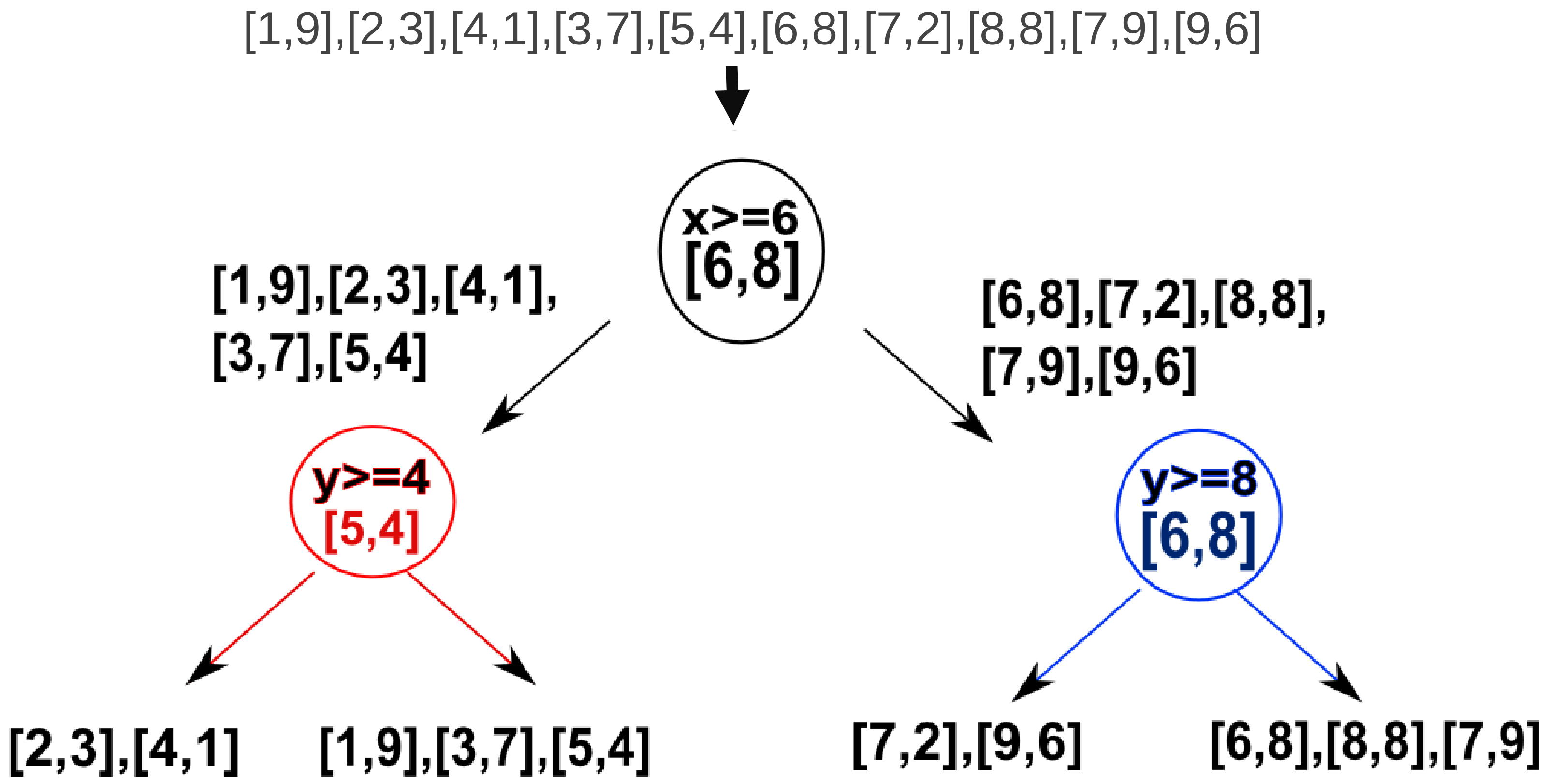
2d-tree: Example

What information do we store at the each node in the tree ?

At each node we store the following data :

1- The dimension we split on. In the 2d example below we store if it is the first coordinate or the second one.

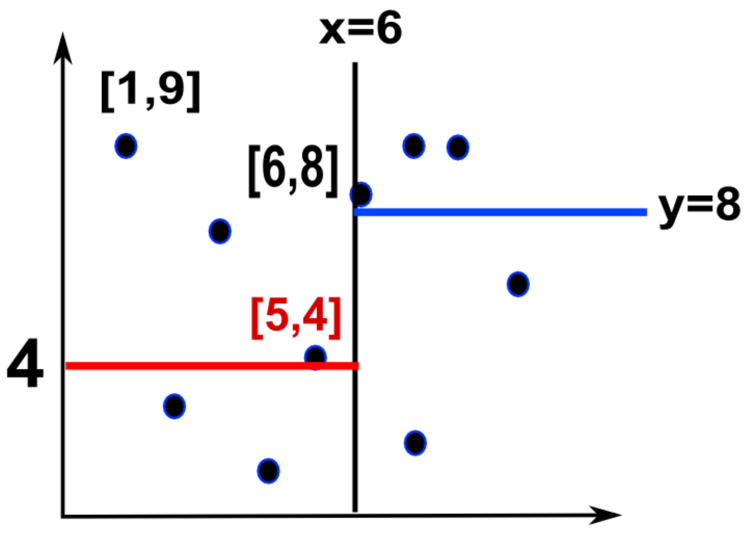


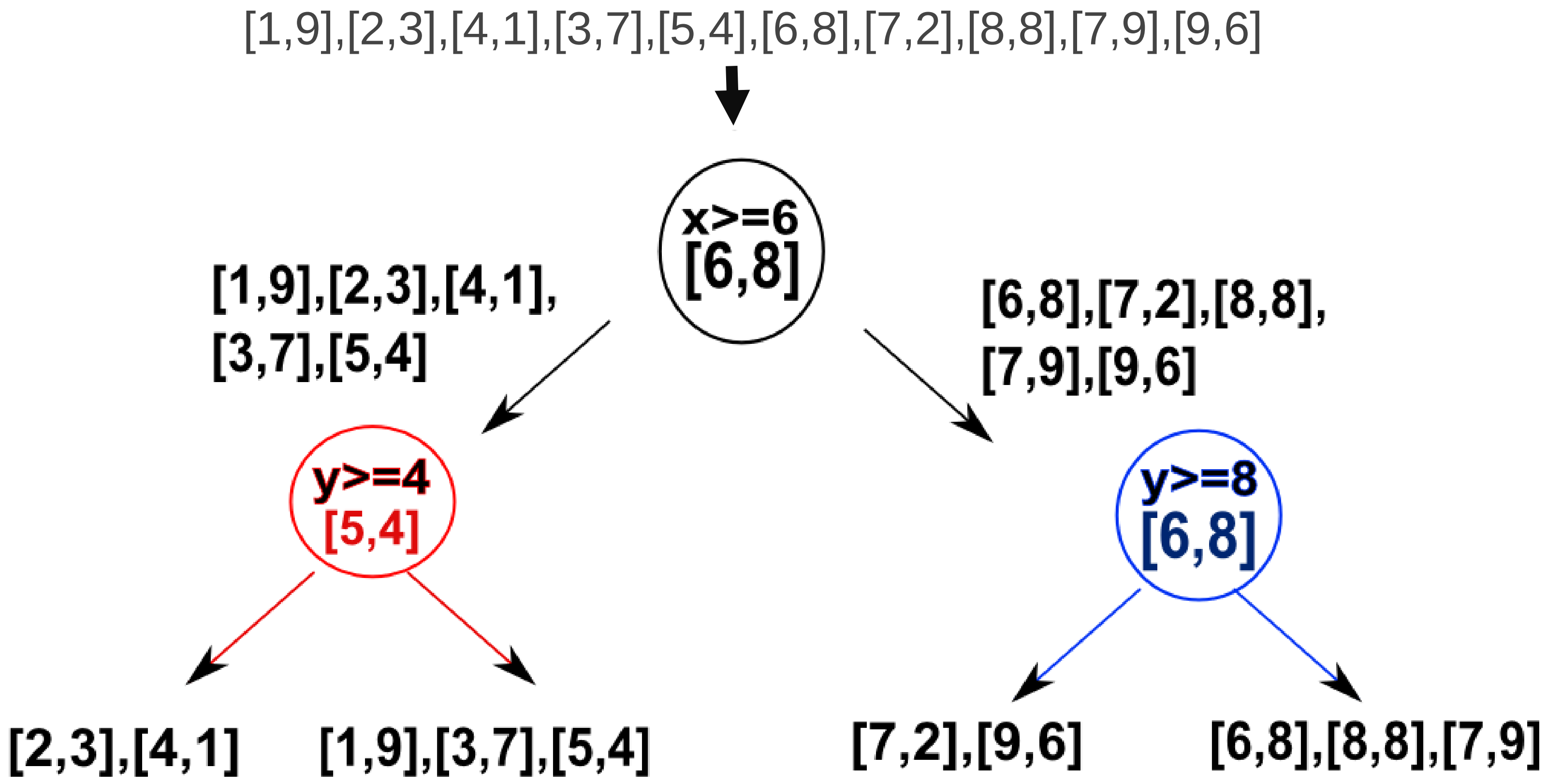


sklearn

2d-tree: Example

What information do we store at the each node in the tree ?
At each node we store the following data :
1- The dimension we split on. In the 2d example below we store if it is the first coordinate or the second one.
2- the split value (the value of the coordinate)





sklearn

2d-tree: Example

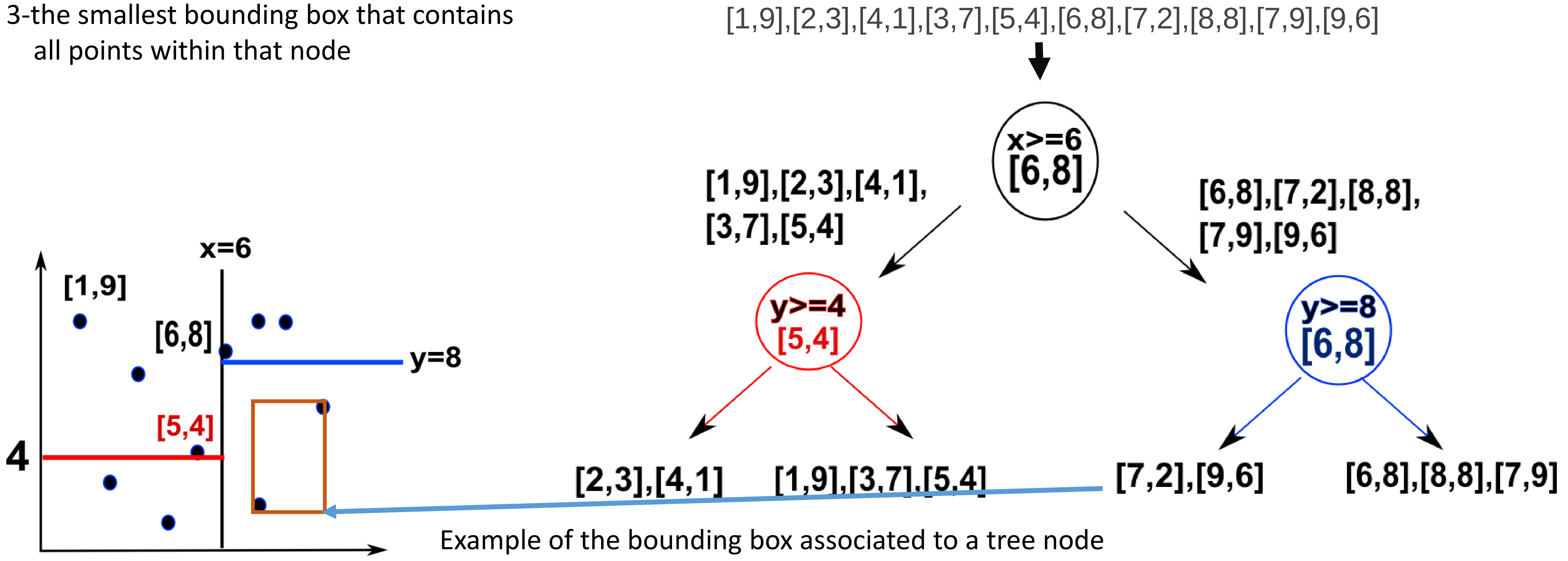
What information do we store at the each node in the tree ?

At each node we store the following data :

1- The dimension we split on. In the 2d example below we store if it is the first coordinate or the second one.

2- the split value (the value of the coordinate)

3-the smallest bounding box that contains all points within that node

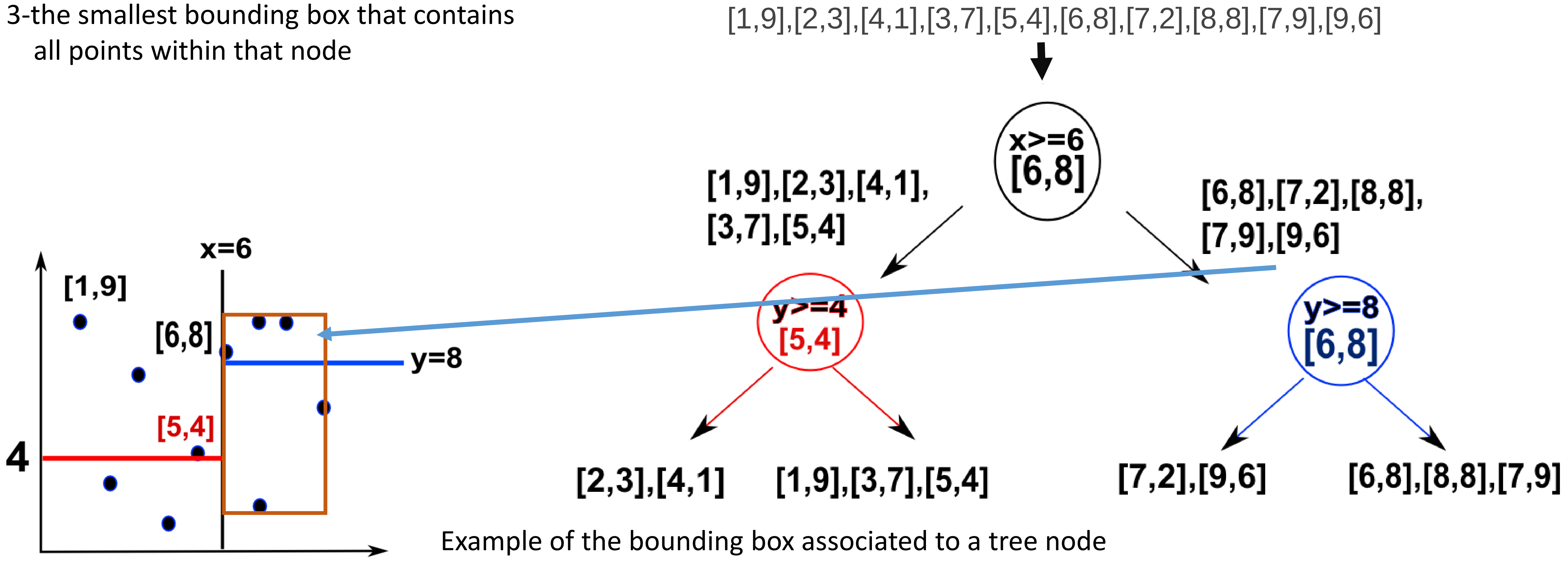


Example of the bounding box associated to a tree node

sklearn

2d-tree: Example

What information do we store at the each node in the tree ?
At each node we store the following data :
1- The dimension we split on. In the 2d example below we store if it is the first coordinate or the second one.
2- the split value (the value of the coordinate)
3-the smallest bounding box that contains
  all points within that node

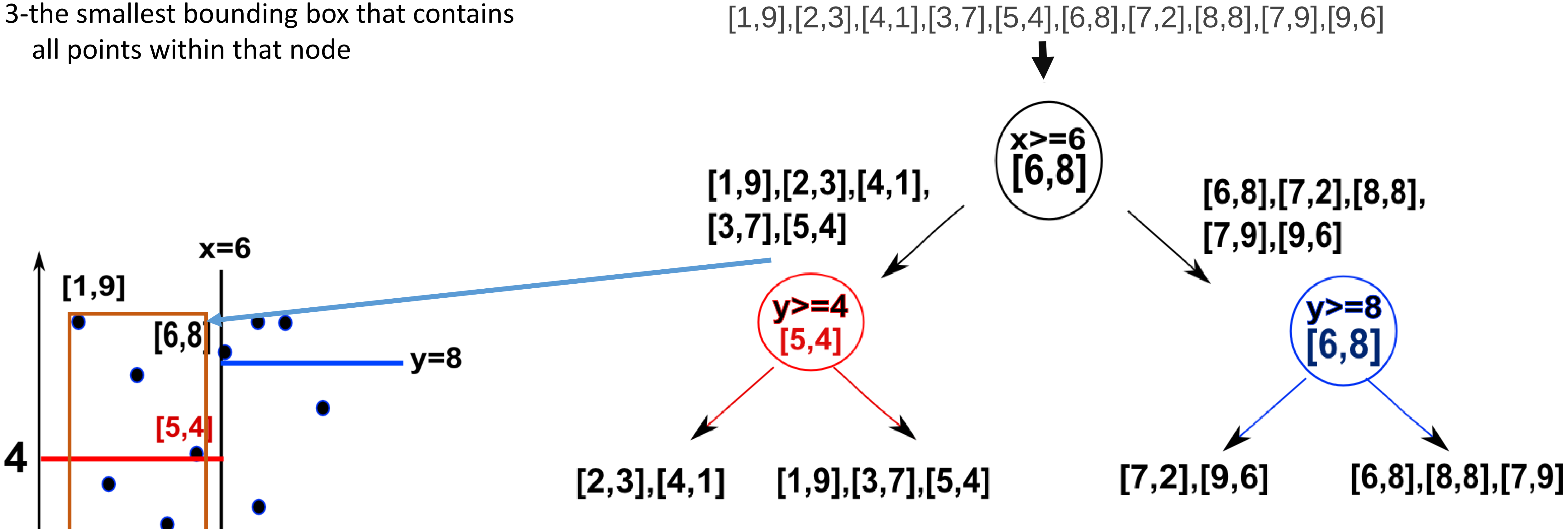


Example of the bounding box associated to a tree node

sklearn

2d-tree: Example

What information do we store at the each node in the tree ?
At each node we store the following data :
1- The dimension we split on. In the 2d example below we store if it is the first coordinate or the second one.
2- the split value (the value of the coordinate)
3-the smallest bounding box that contains
 all points within that node

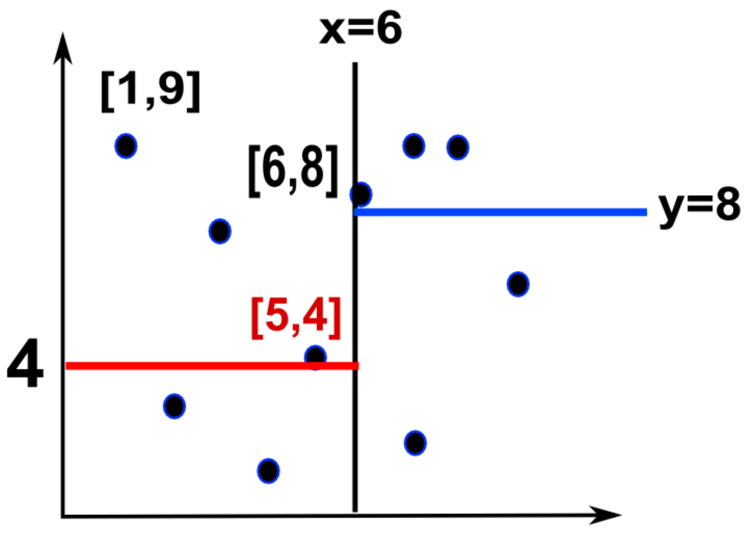


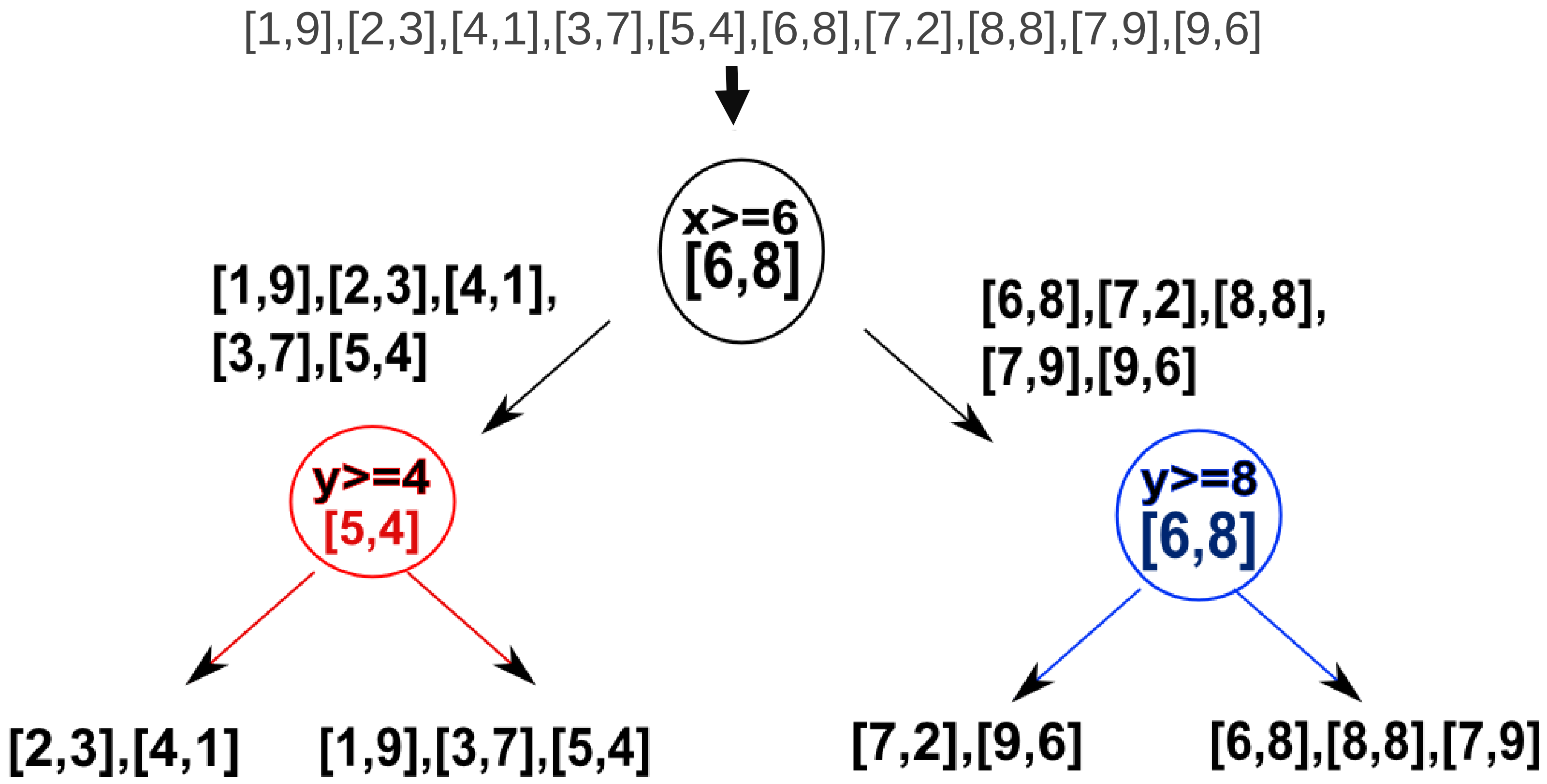
Example of the bounding box associated to a tree node

sklearn

2d-tree: Example

Now suppose that we want to find the nearest neighbor from the data X to a given point .
In that case we navigate in the KD-tree to the region that “hopefully” contains the nearest neighbor

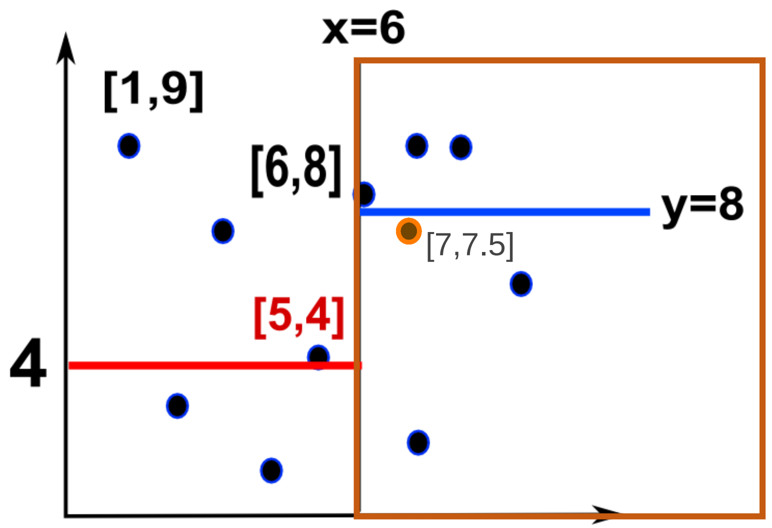


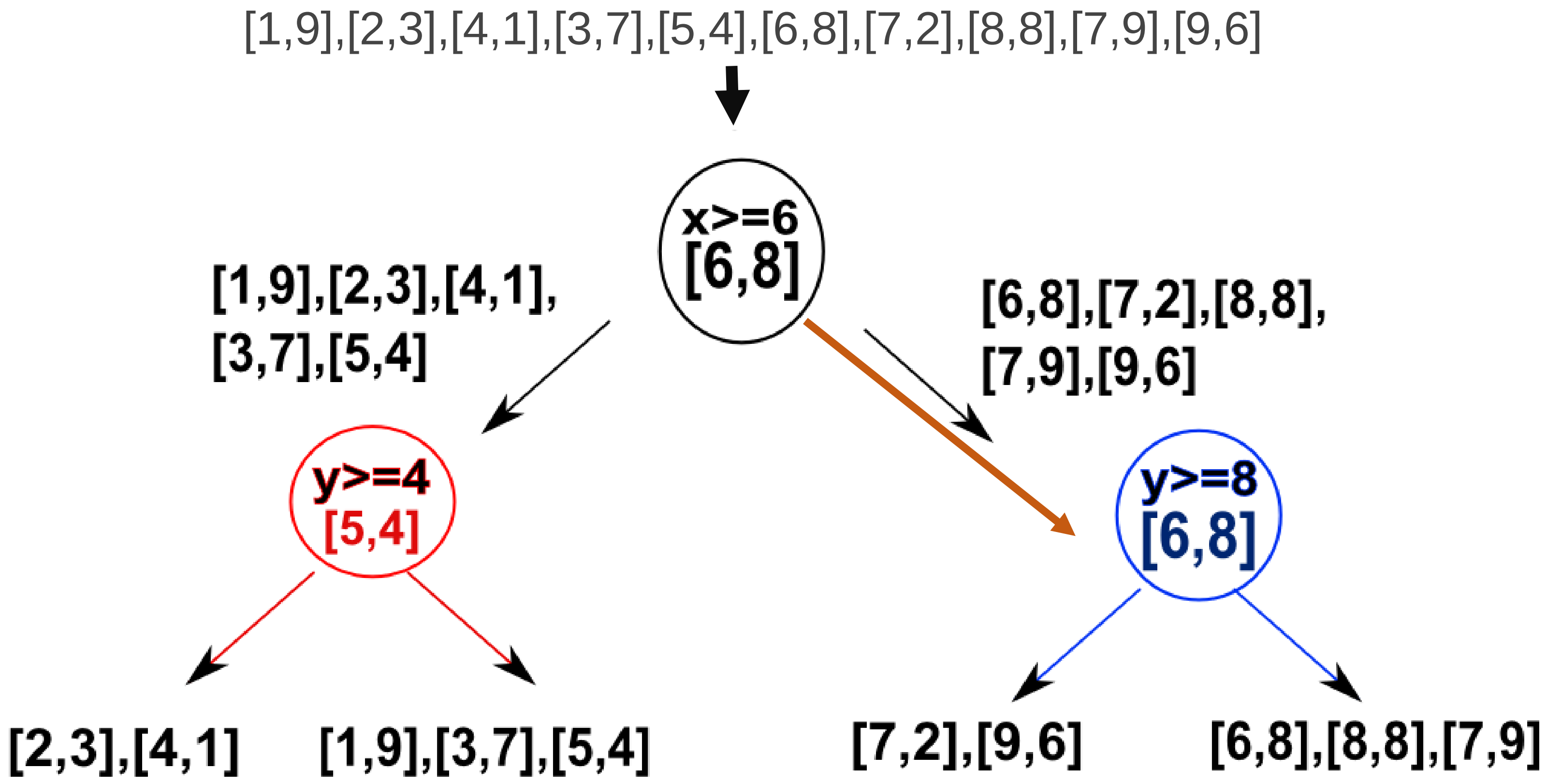


sklearn

2d-tree: Example

Consider the point p=[7.7.5]. To find the closest point from the set X to p. We navigate down the tree following the conditions on the nodes

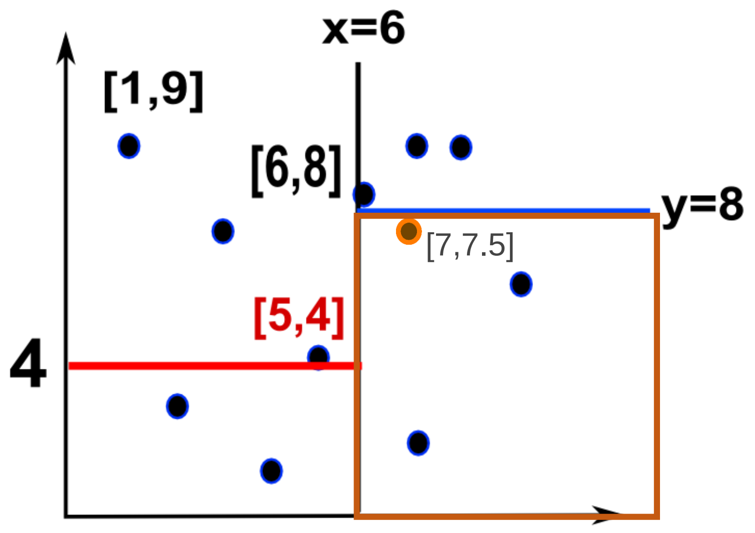


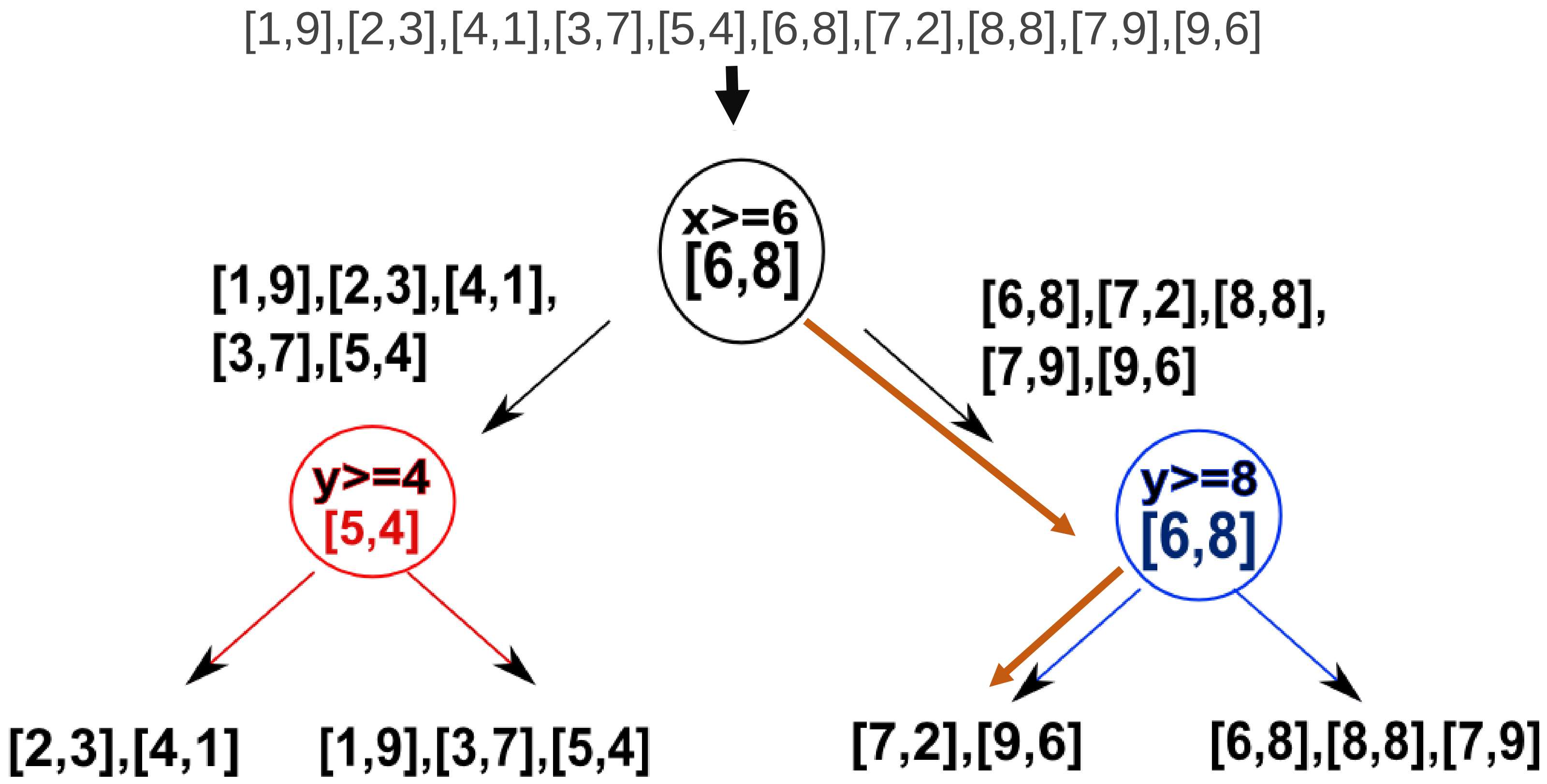


sklearn

2d-tree: Example

Consider the point p=[7.7.5]. To find the closest point from the set X to p. We navigate down the tree following the conditions on the nodes



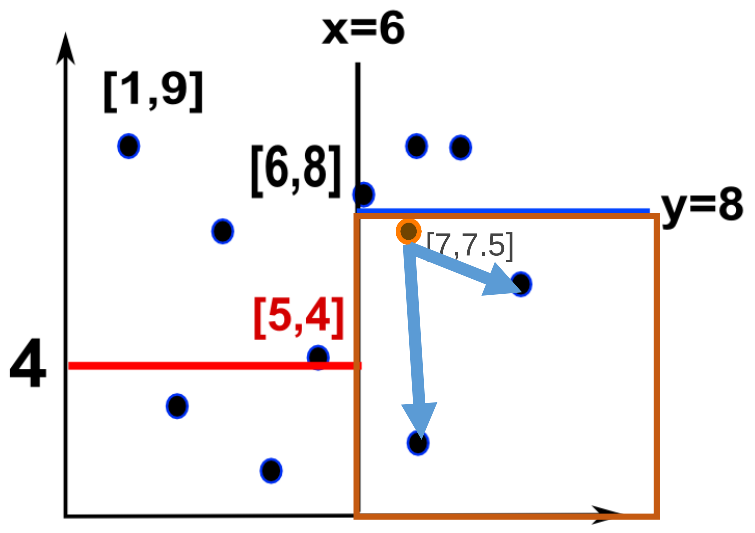


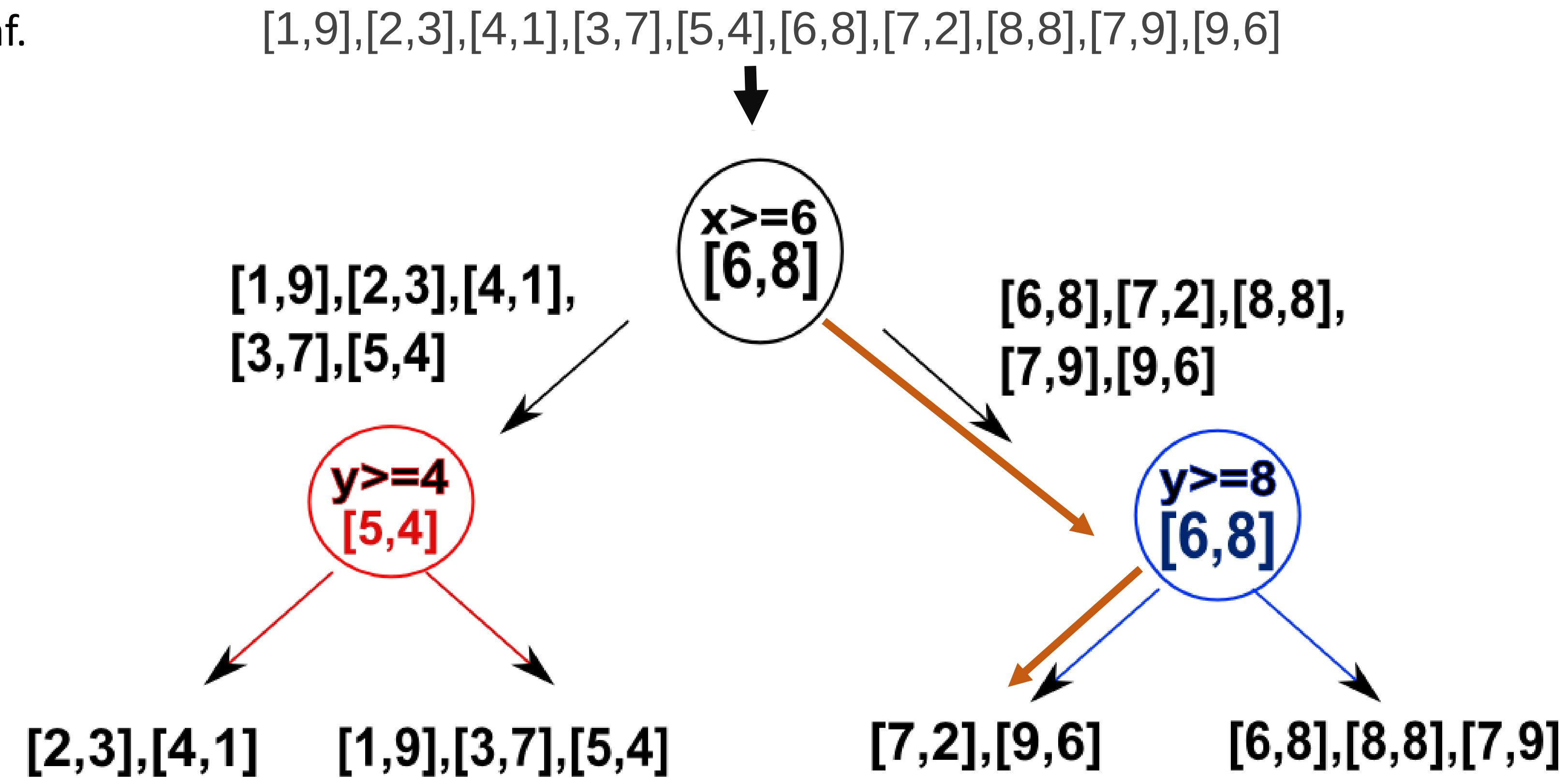
sklearn

2d-tree: Example

Consider the point p=[7.7.5]. To find the closest point from the set X to p. We navigate down the tree following the conditions on the nodes

When we arrive to a leaf we check the distance from the point p to every other point in the leaf.



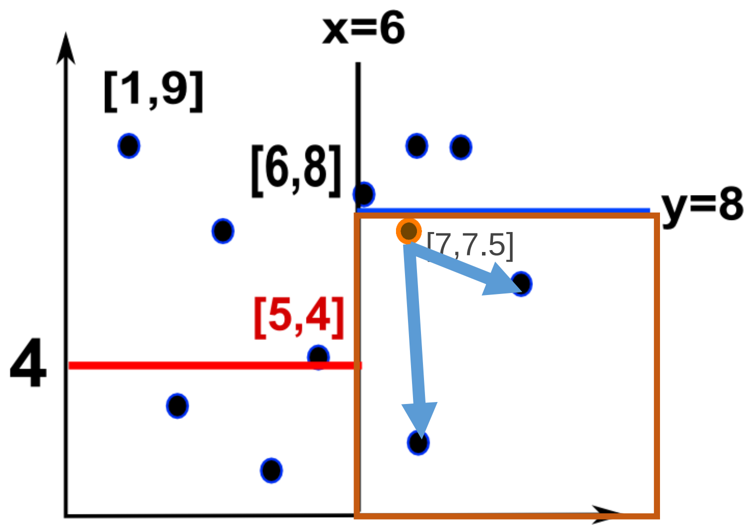


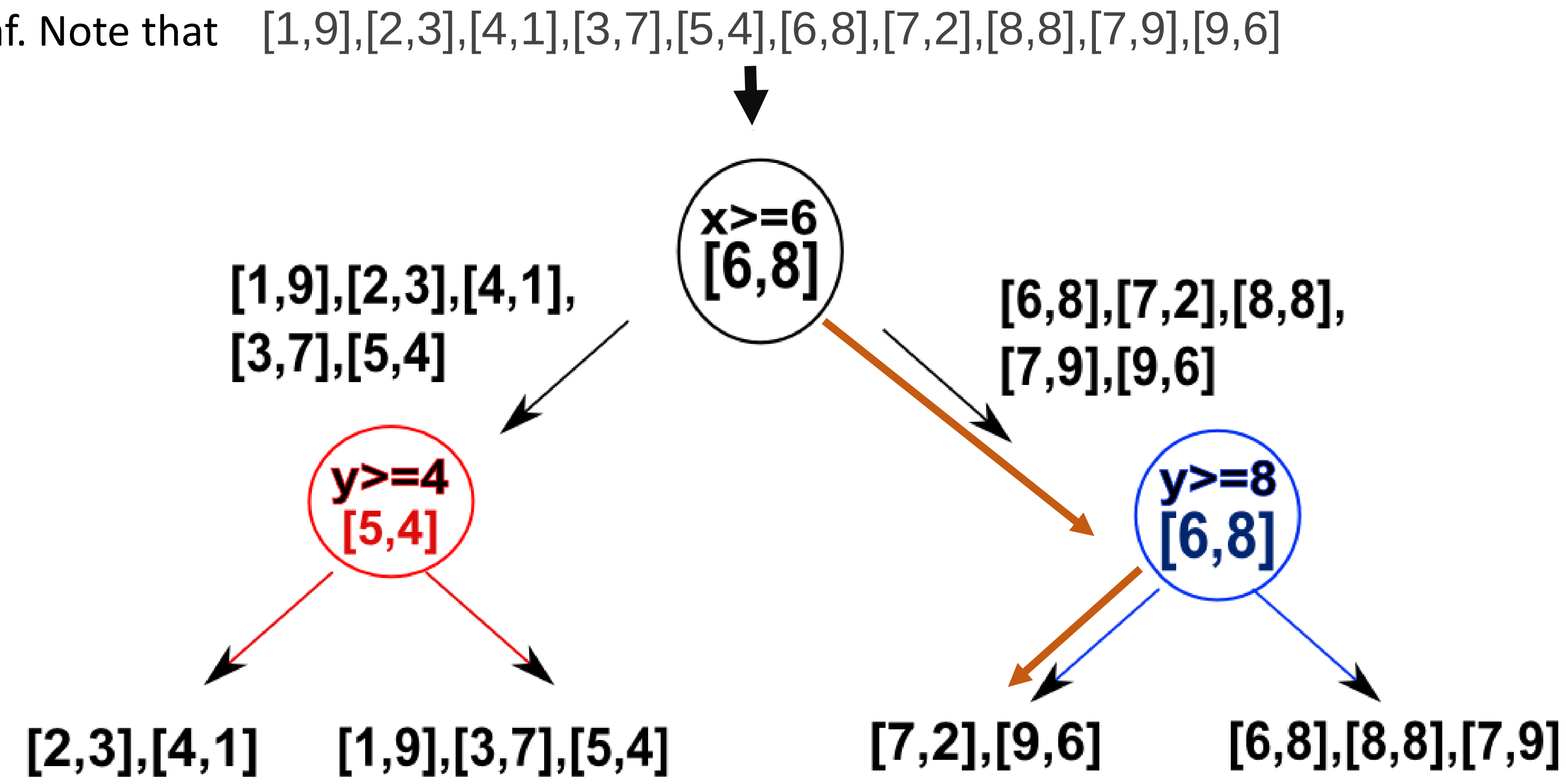
sklearn

2d-tree: Example

Consider the point p=[7.7.5]. To find the closest point from the set X to p. We navigate down the tree following the conditions on the nodes

When we arrive to a leaf we check the distance from the point p to every other point in the leaf. Note that This is the leaf that contains the point.





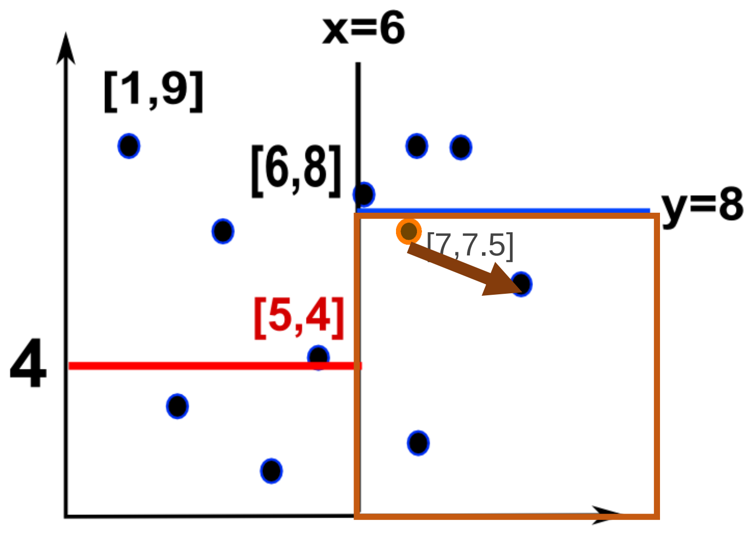
sklearn

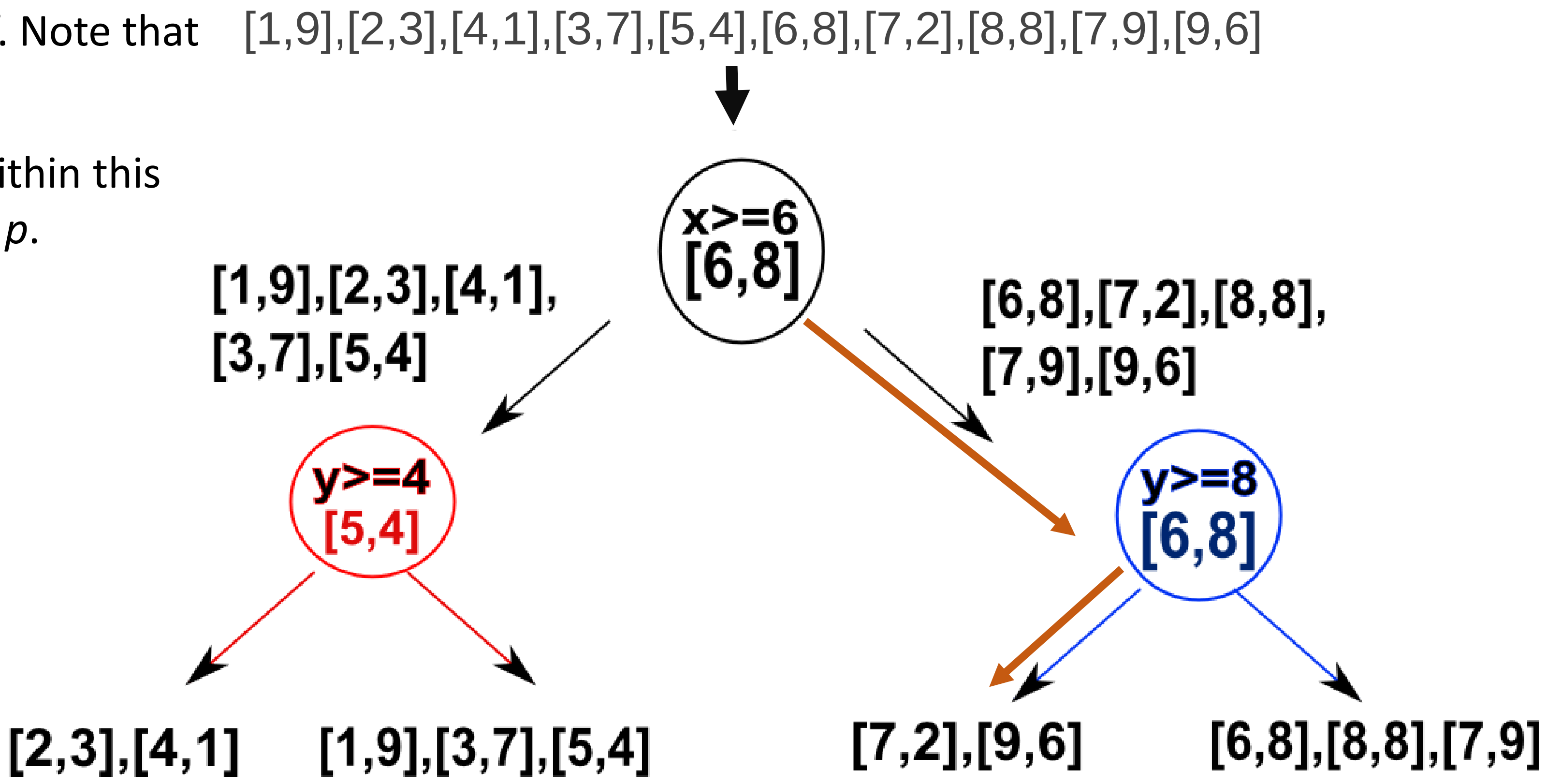
2d-tree: Example

Consider the point *p*=[7.7.5]. To find the closest point from the set X to p. We navigate down the tree following the conditions on the nodes

When we arrive to a leaf we check the distance from the point *p* to every other point in the leaf. Note that This is the leaf that contains the point.

We then choose the point that is closest within this bucket and record its distance to the point *p*.

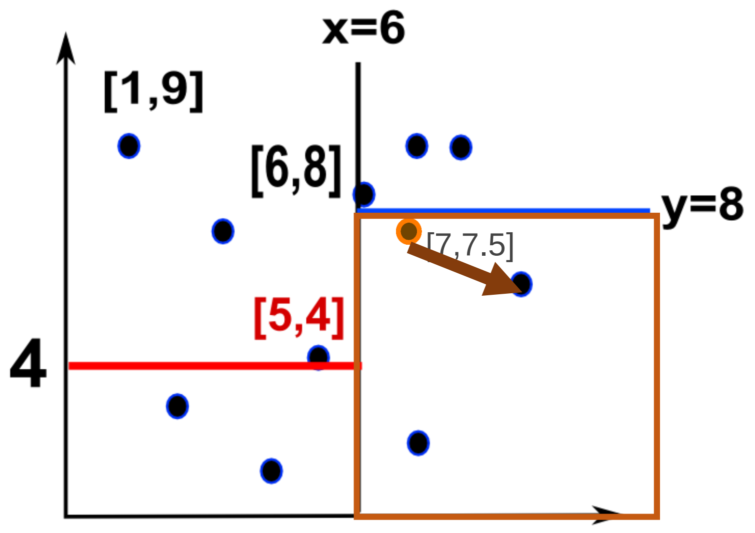


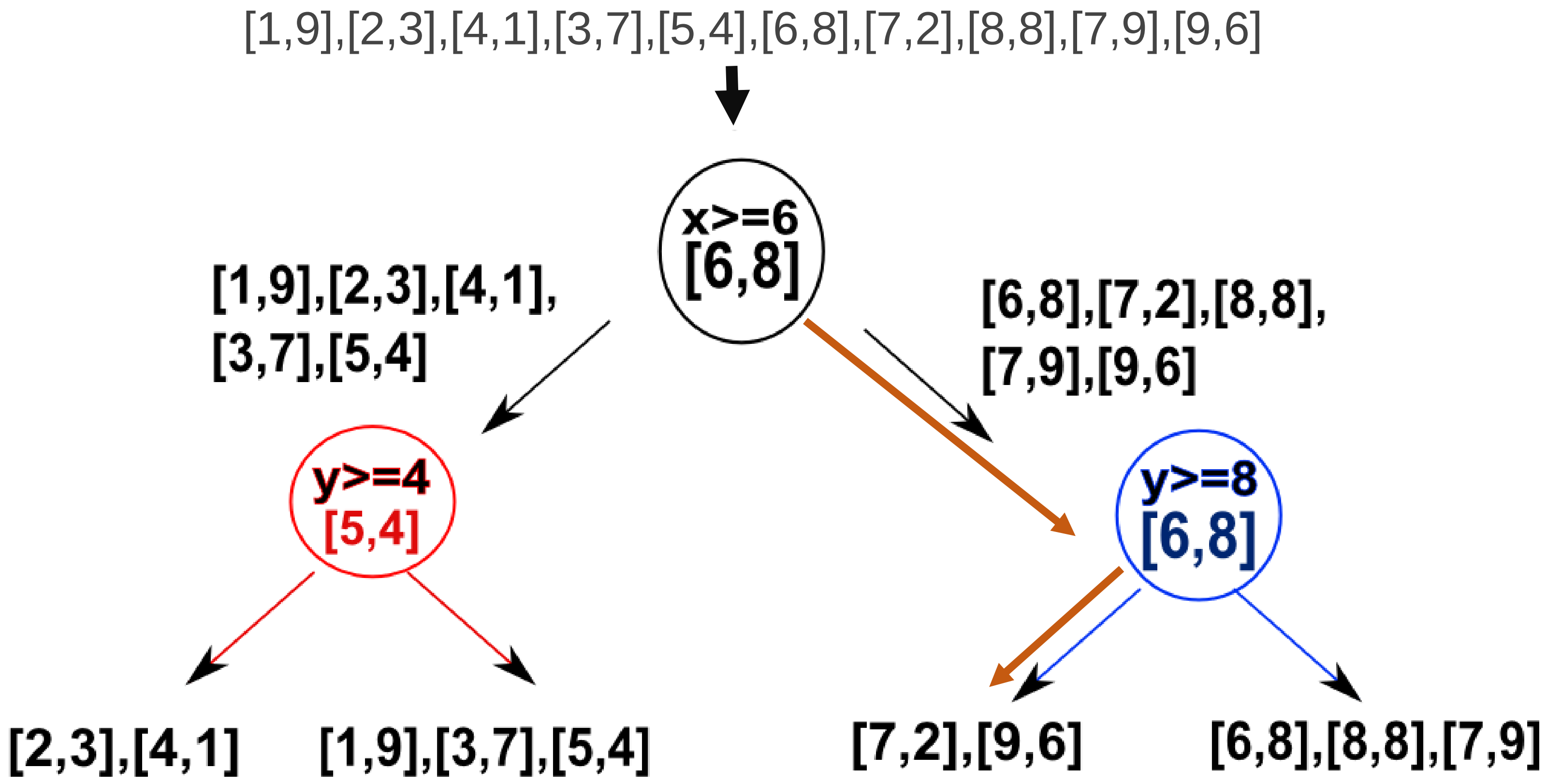


sklearn

2d-tree: Example

What we do from this point is backtracking and try other branches of the tree.



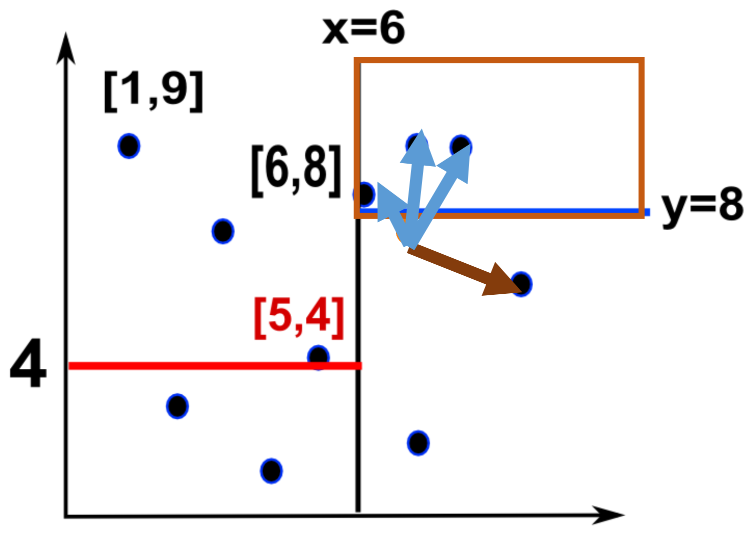


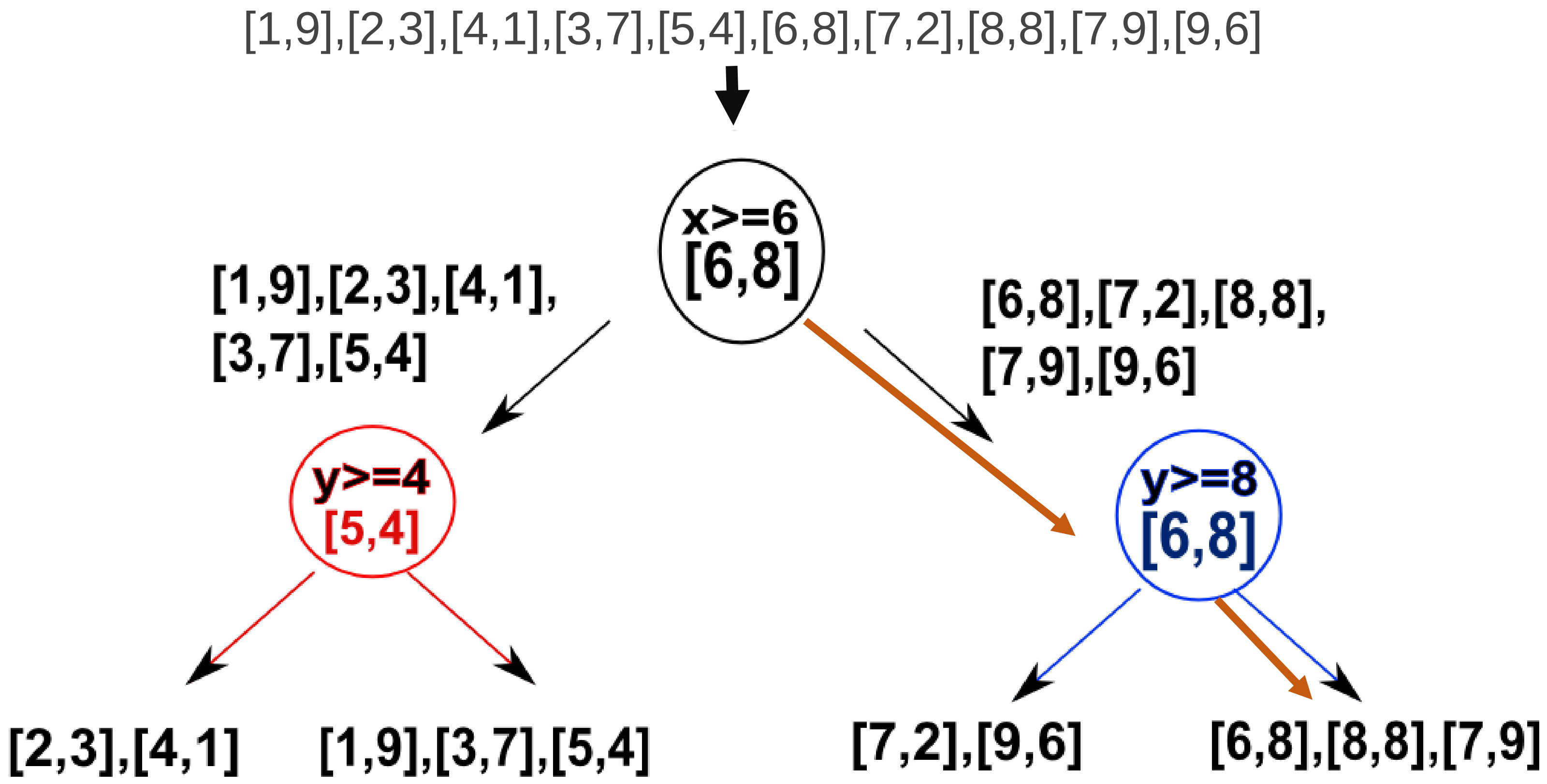
sklearn

2d-tree: Example

What we do from this point is backtracking and try other branches of the tree.

At each leaf we visit, we again measure the distance to every node within that leaf.



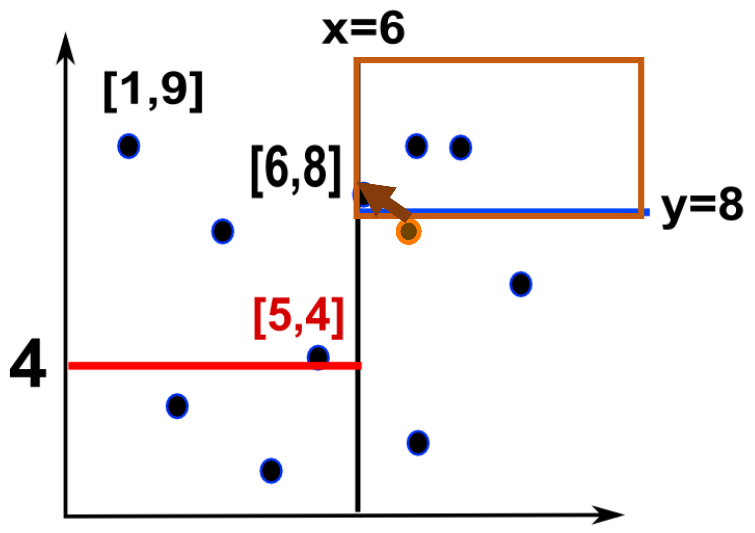


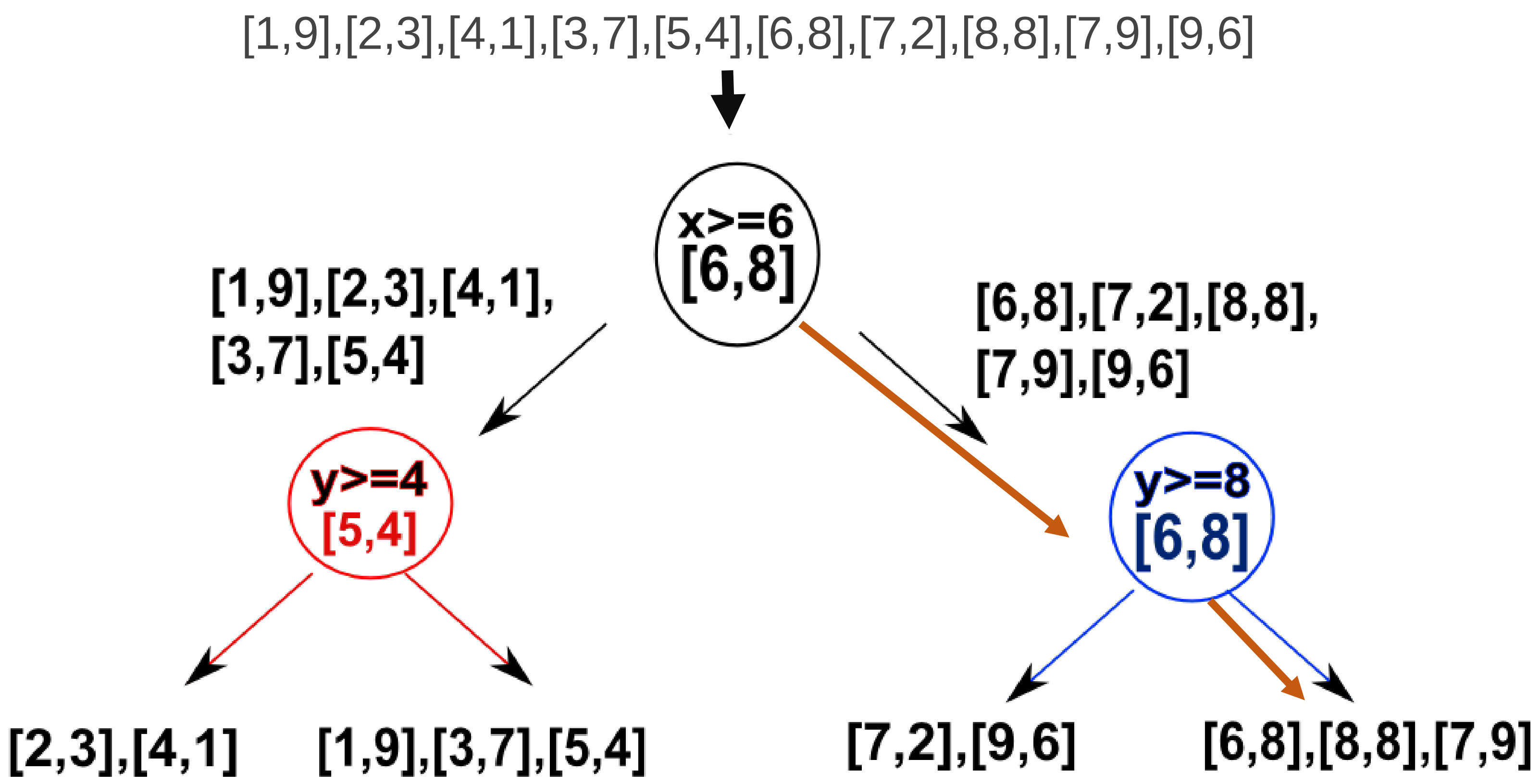
sklearn

2d-tree: Example

What we do from this point is backtracking and try other branches of the tree.

At each leaf we visit, we again measure the distance to every node within that leaf.
If we find a shortest distance we record this distance along with the point that gave us that.



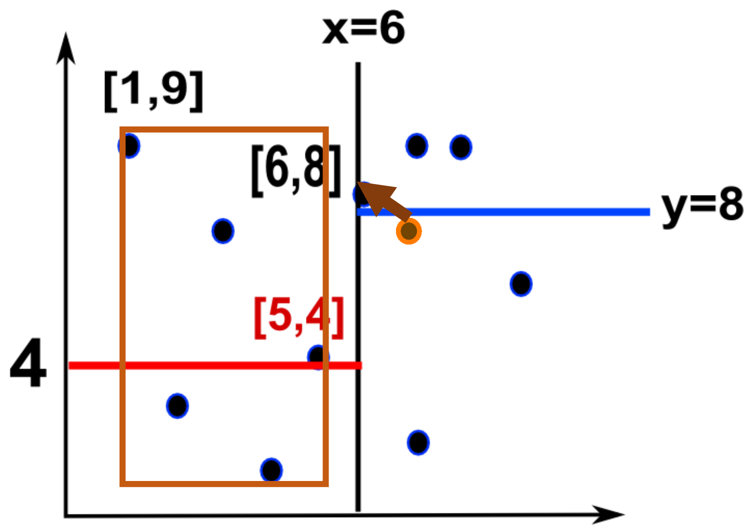


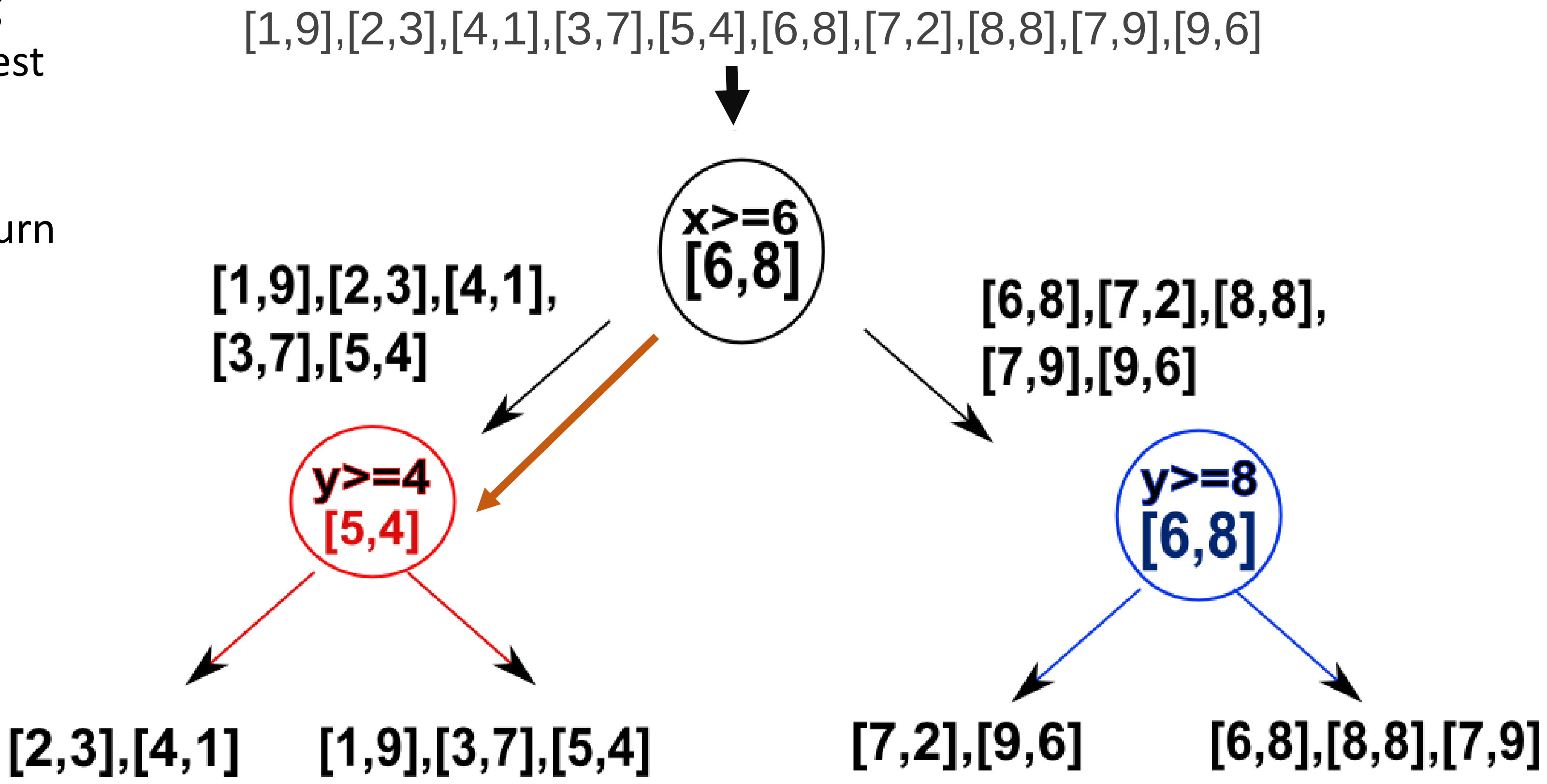
[sklearn](#)

2d-tree: Example

What we do from this point is backtracking and try other branches of the tree.

Now when we go to check the other branch we notice that the distance to the bounding box is greater than the distance to the nearest neighbor found so far, so we ignore that branch from the search. Since there are no more branches to search in this case we return the point that we have found.



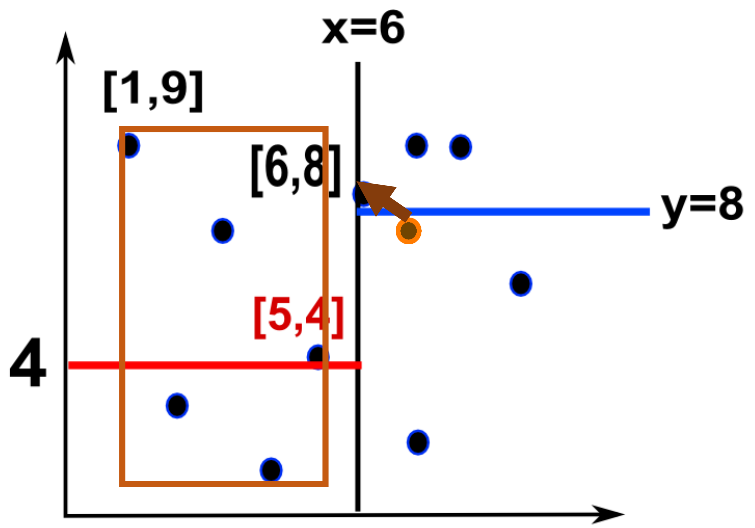


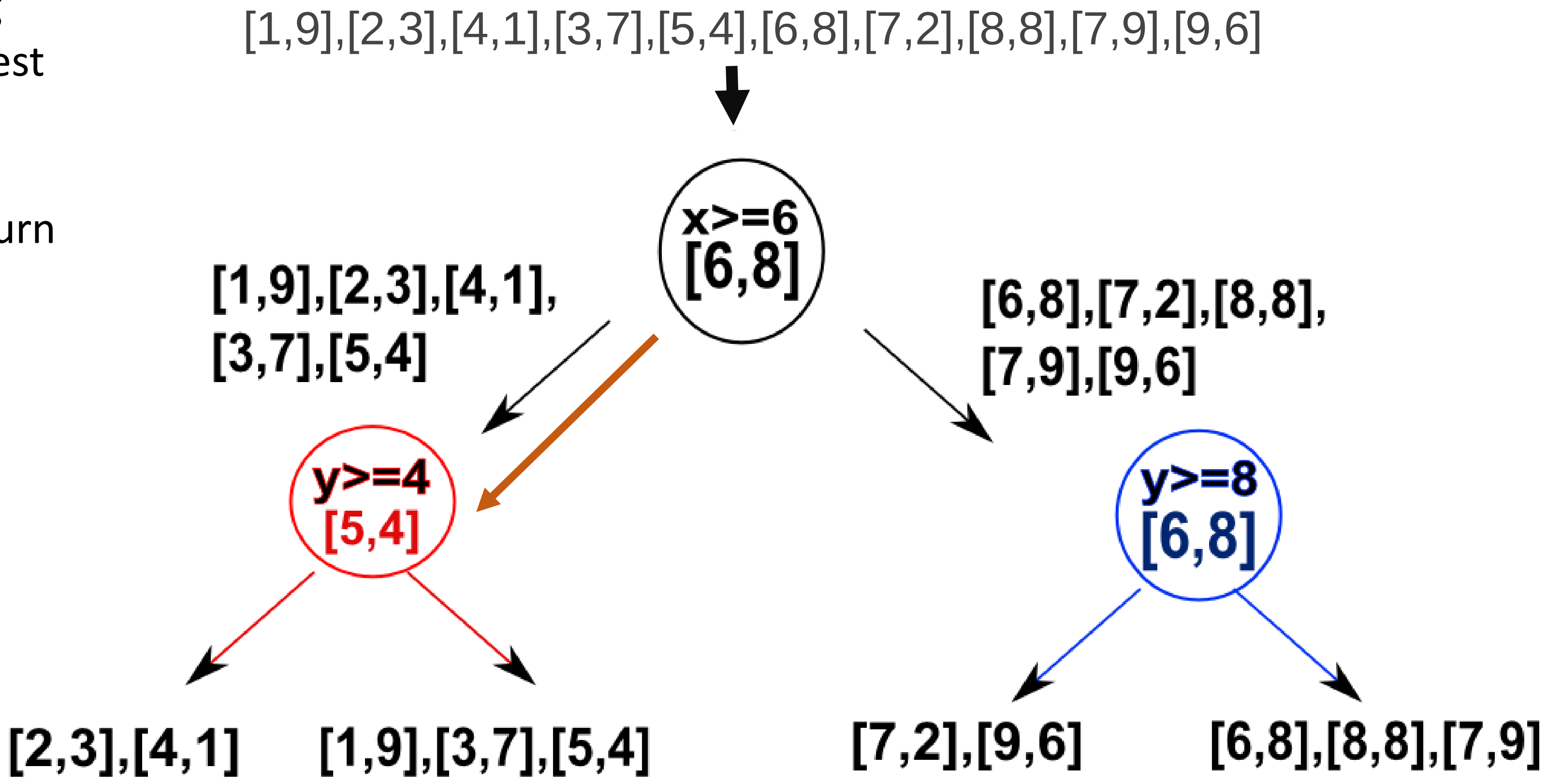
sklearn

2d-tree: Example

What we do from this point is backtracking and try other branches of the tree.

Now when we go to check the other branch we notice that the distance to the bounding box is greater than the distance to the nearest neighbor found so far, so we ignore that branch from the search. Since there are no more branches to search in this case we return the point that we have found.

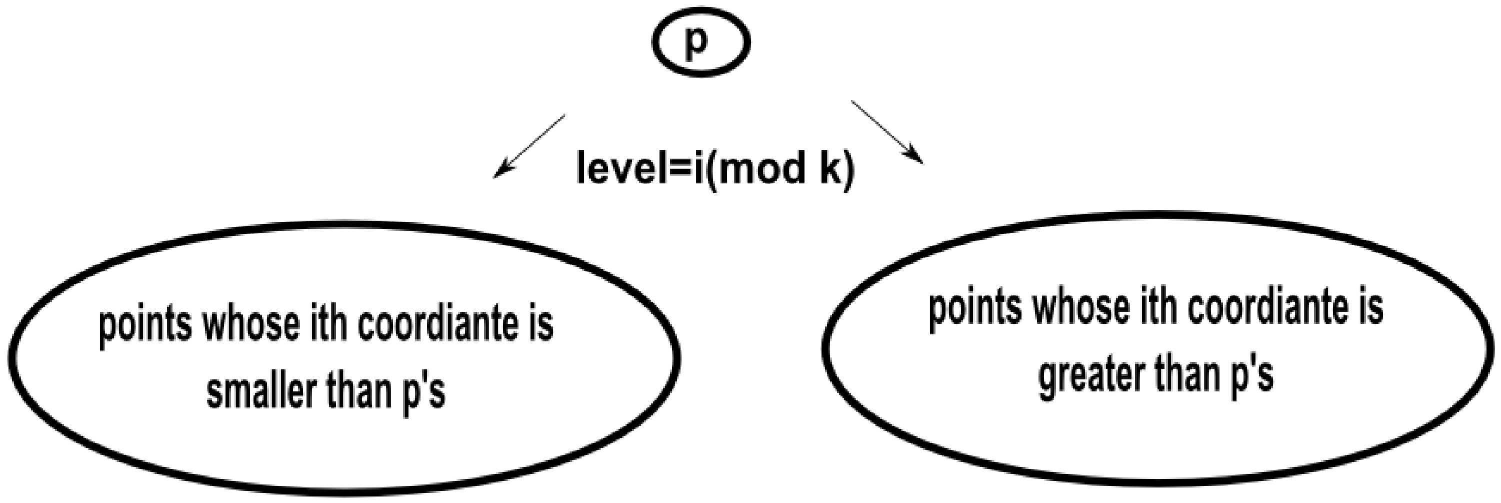




sklearn

See another example

Kd-tree: Example


p
level=i(mod k)
points whose ith coordiante is smaller than p's
points whose ith coordiante is greater than p's

we keep recursively divising the k-space in this manner

KD-tree is only efficient for small d

Sklearn Implementations

1-Brute force : Query here costs $O(n * d)$ where n is the size of the data and d is the dimension of the space (# of features)
2-Ball tree : This is approximately $O(d \log(n))$
3-Kd-tree : Depends largely on d. Efficient for $d < 20$ (around $O\ (d * log(n)\ )$)

Link for a complete description

Neighborhood graphs and their relatives

K-Nearest Neighbor Graph (KNN Graph)

Suppose that we are given a set of points $X = \{p_1, p_2, \dots, p_n\}$ in R^d with a distance function d defined one them.

Sklearn implementation

K-Nearest Neighbor Graph (KNN Graph)

Suppose that we are given a set of points $X = \{p_1, p_2, \dots, p_n\}$ in R^d with a distance function d defined one them.

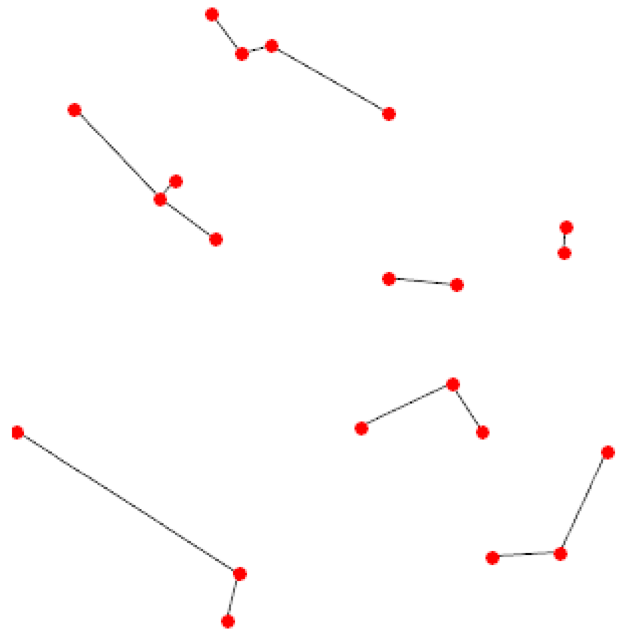
For a fixed integer k, connect the points x, y in X if either $d(x, y) \le d(x, x_k)$ or $d(x, y) \le d(y, y_k)$ where x_k, y_k are the k^{th} nearest neighbors of x, y respectively. Doing do for all points we obtain a graph $G(X, E)$ where E is the set of edges connect the points in X as described above.

Sklearn implementation

K-Nearest Neighbor Graph (KNN Graph)

Suppose that we are given a set of points $X = \{p_1, p_2, \dots, p_n\}$ in R^d with a distance function d defined one them.

For a fixed integer k, connect the points x, y in X if either $d(x, y) \leq d(x, x_k)$ or $d(x, y) \leq d(y, y_k)$ where x_k, y_k are the k^{th} nearest neighbors of x, y respectively. Doing do for all points we obtain a graph $G(X, E)$ where E is the set of edges connect the points in X as described above.



Example of 1-NN graph

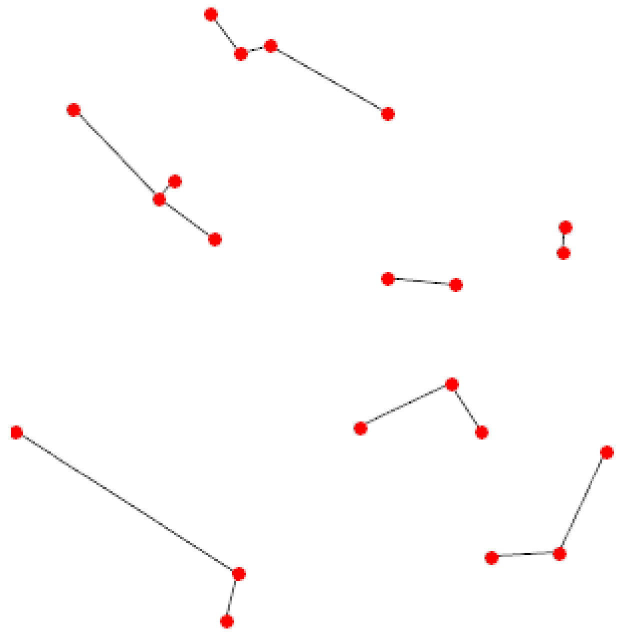
Sklearn implementation

The graph that we obtain is called the K-nearest neighbor graph or K-NN graph.

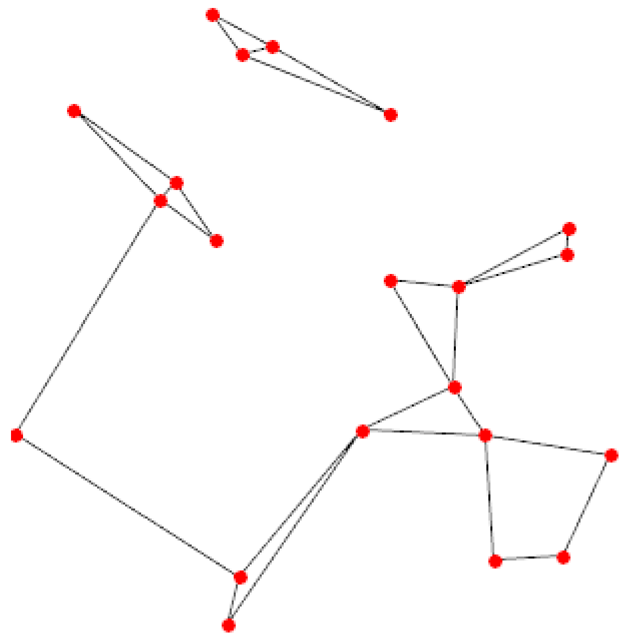
K-Nearest Neighbor Graph (KNN Graph)

Suppose that we are given a set of points $X = \{p_1, p_2, \dots, p_n\}$ in R^d with a distance function d defined one them.

For a fixed integer k, connect the points x, y in X if either $d(x, y) \leq d(x, x_k)$ or $d(x, y) \leq d(y, y_k)$ where x_k, y_k are the k^{th} nearest neighbors of x, y respectively. Doing do for all points we obtain a graph $G(X, E)$ where E is the set of edges connect the points in X as described above.



Example of 1-NN graph



Example of 2-NN graph

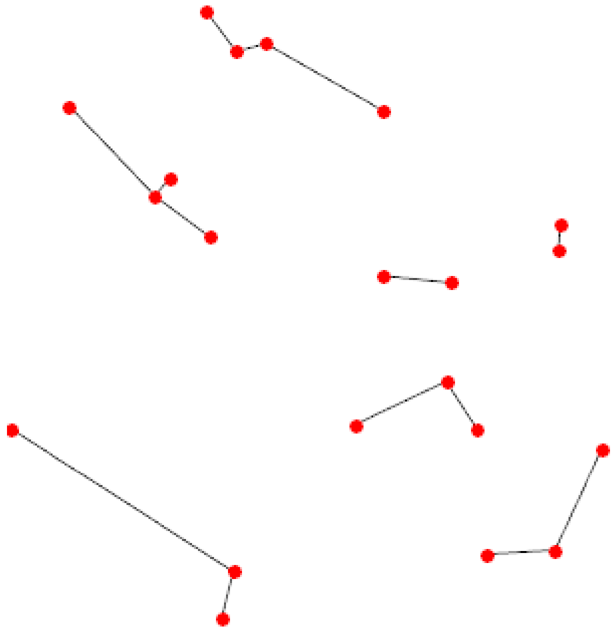
Sklearn implementation

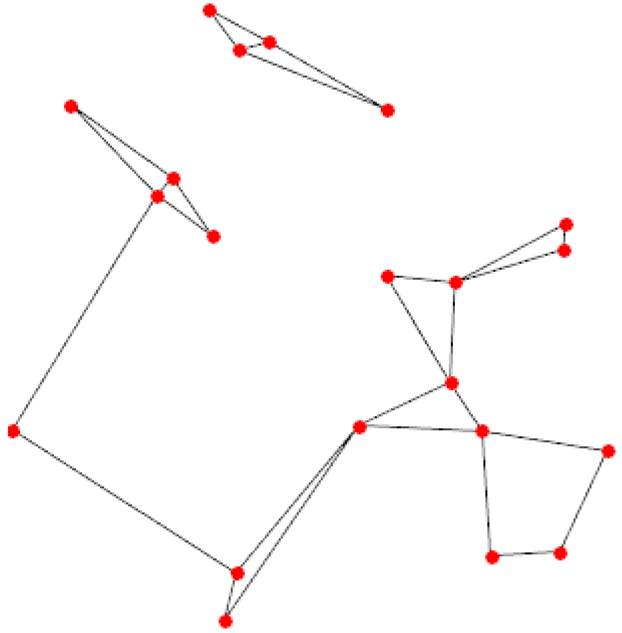
The graph that we obtain is called the K-nearest neighbor graph or K-NN graph.

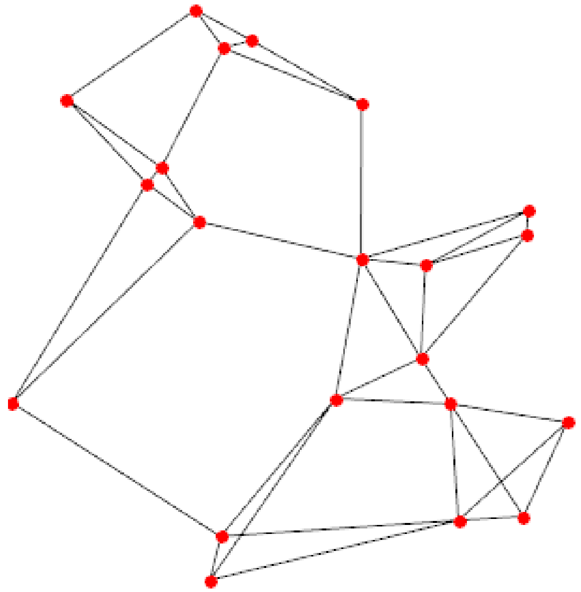
K-Nearest Neighbor Graph (KNN Graph)

Suppose that we are given a set of points $X = \{p_1, p_2, \dots, p_n\}$ in R^d with a distance function d defined one them.

For a fixed integer k, connect the points x, y in X if either $d(x, y) \leq d(x, x_k)$ or $d(x, y) \leq d(y, y_k)$ where x_k, y_k are the k^{th} nearest neighbors of x, y respectively. Doing do for all points we obtain a graph $G(X, E)$ where E is the set of edges connect the points in X as described above.


Example of 1-NN graph


Example of 2-NN graph


Example of 3-NN graph

Sklearn implementation

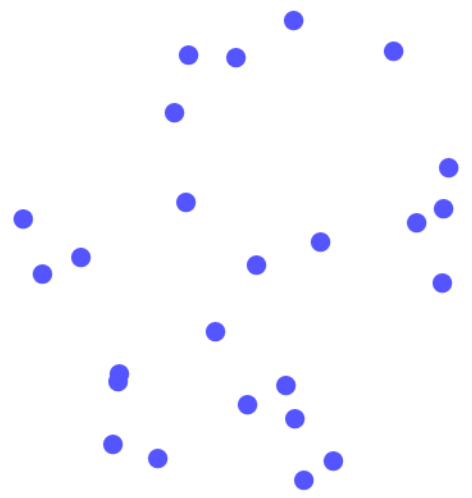
The graph that we obtain is called the K-nearest neighbor graph or K-NN graph.

ε- neighborhood graph

Suppose that we are given a set of points $X = \{p_1, p_2, \ldots, p_n\}$ in R^d with a distance function d defined one them.

For a fixed ε, connect the points x, y if $d(x, y) \leq$ ε.

Doing do for all points we obtain a graph $G(X, E)$ where E is the set of edges connect the points in X as described above.



Sklearn implementation

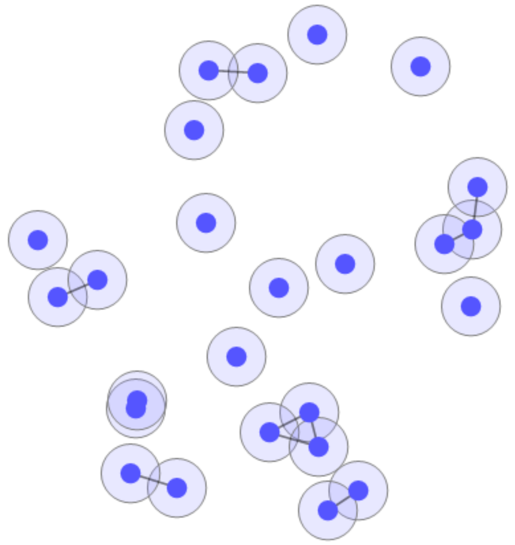
ε- neighborhood graph

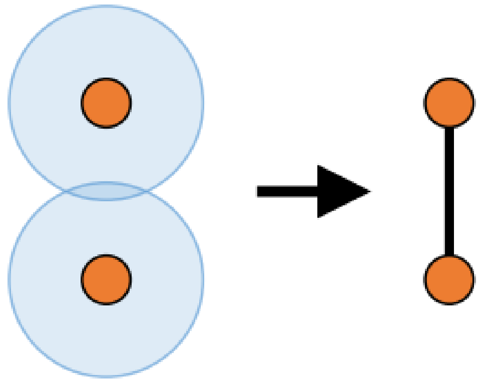
Suppose that we are given a set of points $X = \{p_1, p_2, \dots, p_n\}$ in R^d with a distance function d defined one them.

For a fixed ε, connect the points x, y if $d(x, y) \leq$ ε.

Doing do for all points we obtain a graph $G(X, E)$ where E is the set of edges connect the points in X as described above.

This is *equivalent* to :
Insert an edge if the disks around the points intersect each other.





Sklearn implementation

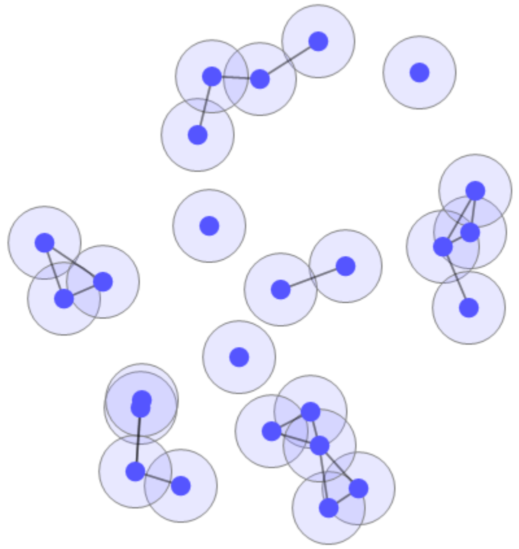
Note that $d(x,y) \leq \frac{\varepsilon}{2}$ if and only if the two balls surrounding x and y intersect

ε- neighborhood graph

Suppose that we are given a set of points $X = \{p_1, p_2, \ldots, p_n\}$ in R^d with a distance function d defined one them.

For a fixed ε, connect the points x, y if $d(x, y) \leq$ ε.

Doing do for all points we obtain a graph $G(X, E)$ where E is the set of edges connect the points in X as described above.



Trying different ε

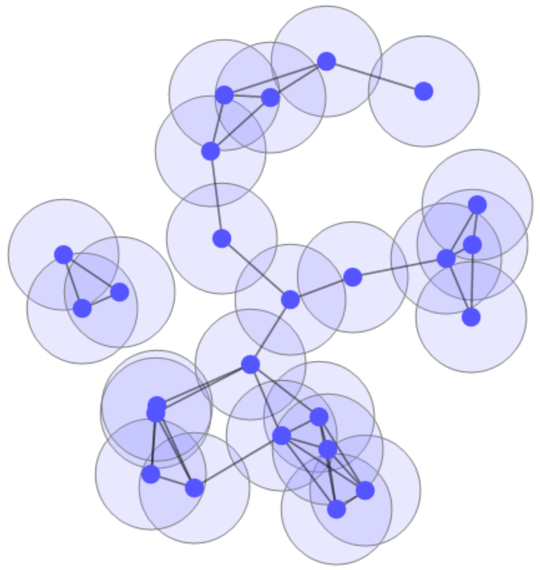
Sklearn implementation

ε- neighborhood graph

Suppose that we are given a set of points $X = \{p_1, p_2, \dots, p_n\}$ in R^d with a distance function d defined one them.

For a fixed ε, connect the points x, y if $d(x, y) \leq$ ε.

Doing do for all points we obtain a graph $G(X, E)$ where E is the set of edges connect the points in X as described above.



Trying different ε

Sklearn implementation

Euclidian Minimal Spanning Tree (EMST)

Suppose that we are given a set of points $X = \{p_1, p_2, \ldots, p_n\}$ in R^d with a distance function d defined one them.

The EMST of X is a minimal spanning tree where the weight of the edge between each pair of points is the distance between those two points.

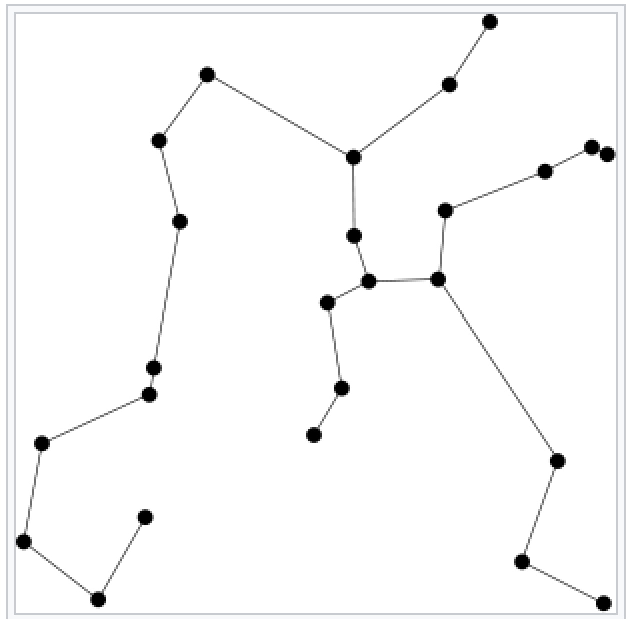


Image source-Wikipedia article