# Intro. to Machine Learning Project2

KDTree KNN Classifier and PCA Algorithm

0416324 An-Fong Hwu

## Build environment (Note, this report is written in md-like format)

\* Ubuntu 16.04 LTS 64bit

\* python 3.5.2

\* Intel Core i5-7500 3.4GHz 4C4T

\* DDR4 -2400 16GB dual channel

## What is KD-Tree?

\* A BST like structure ,which now implement in the high dimensional structure for fast querying the nearest point in high dimension, i.e. K-Nearest Neighbors algorithm can be finished faster

See more at: <https://en.wikipedia.org/wiki/K-d_tree>

\* Algorithm time complexity analysis.

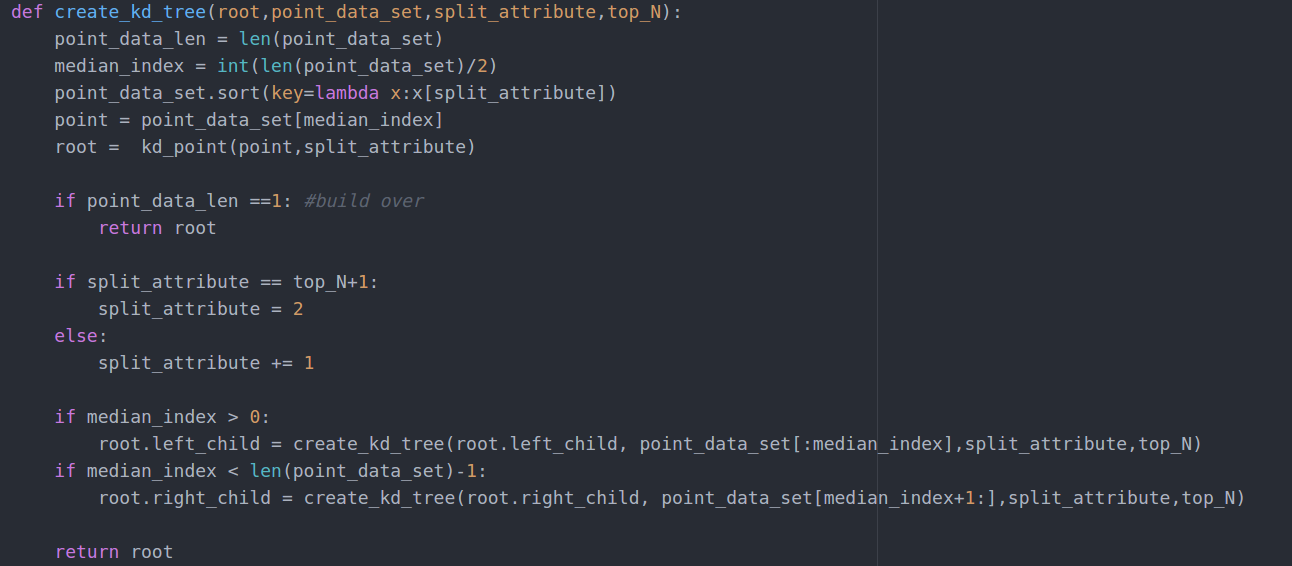
Brute force: O(N^N) vs Using the KD-Tree structure O(NlogN)



## How KD-Tree is built

It is just same as the BST, but now extended into K-Dimensions.

We just change the axis of comparison each time we build one more depth in the tree, recording the current splitting axis and split the data in half (data that < medium and the other, according to the splitting axis)



## How to find K Nearest Neighbors

[Pseudo code]

current\_knn how many k I’ve done

all\_knn of k nearest neighbors

Initialize current\_point =root min\_distance = INF traversed\_point = None

while current\_point is not leaf

do binary\_search in KD-Tree

traversed\_point.push(current)

update the min\_dustance if the current\_point has not been marked as KNN\_traversed before

end while

while traversed\_point is not empty

current\_point = traversed\_point.pop()

if current\_point is leaf

if distance(query\_point,current\_point) < min\_distance

NN = current\_point

min\_distance = distance(query\_point,current\_point)

end if

end if

else

if distance(query\_point[current\_split\_axis],current\_point[current\_split\_axis]) < min\_dist or current\_knn < all\_knn) #distance b/w current point of splitting axis with query point is closer than the min\_dist shows that there might be a closer neighbor at other side

if query\_point[current\_split\_axis]<current\_point[current\_split\_axis]

go to its right\_child(opposite)

end if

else

go to its left\_child

end else

while current\_point is not leaf

do binary\_search from opposite point

traversed\_point.push(current)

update the min\_dustance if the current\_point has not been marked as KNN\_traversed before

end else

end while

return NN and mark the NN.KNN\_traversed as true for found the kth nearest neighbor

## What is PCA algorithm?

\* A statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components.

\* Procedure

1.Find the mean\_value\_matrix according to each column’s average (minus the average for shifting to see how much does the current data differs from the group’s average, in order to check the data dependency )

2.Find the covariance matrix according to the mean value matrix to see the data dependency and the degree of distribution

3.Find the eigen value and eigen vectors according to covariance matrix

4.The top L eigen value will be in the dimension of N\*L where the dimension of original mean

(the eigen value will be listed in column vector) value matrix is M\*N --→ thus the oprimized data structed can be reduced to L where L<<N, hence dimensional reduction is completed