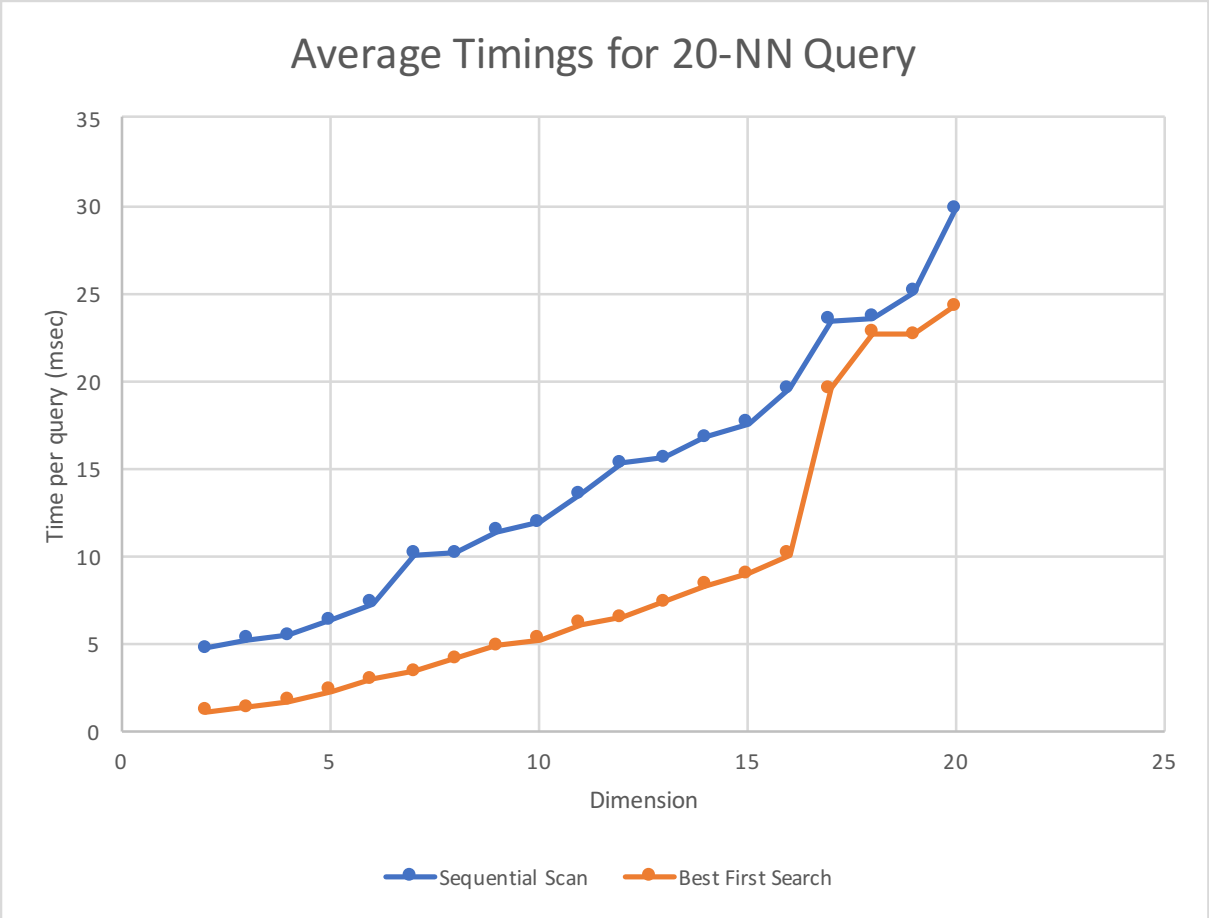
COL362 – Introduction to Database Management Systems

Assignment 3

KD-Trees

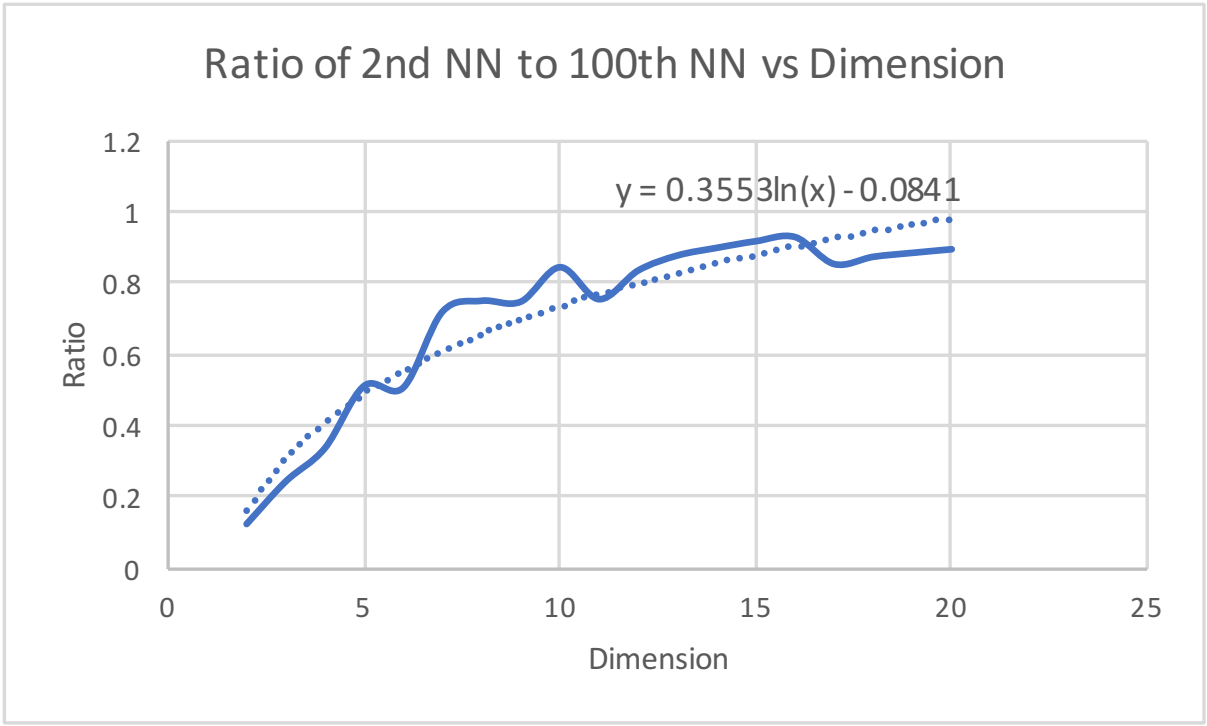
Sagar Goyal 2015CS10253
Mehak Preet Dhaliwal 2015CS10238
Nilaksh Agarwal 2015PH10813



Sequential Scan – O(n.log(k) – Shows an linear curve with d – as shown, due to longer time being taken to compute distances in kd-tree.

Best First Search (using kd Trees) – This is of the order O(k.log(n)). This also has a factor of d, So this is almost linear for d <=16, since the data (10,000 >> 2^16). But, 2^17 > 10,000, so kd-trees is no longer as useful in K-nearest neighbour search. Hence, we see the spike in time.

This is due to time complexity no longer being of the order O(logN). It progresses to O(N) which is the worst-case time complexity.



Here, increasing the dimension is essentially expanding the search space as now, equal number of points have a wider space to spread over.
So, for example if we take 2D Trees and 3D trees, both with 100,000 points, equally spaced, the 2D trees will have (square root of 100,000) number of points between 0 and 1, whereas 3D trees will have (cube root of 100,000) number of points between 0 and 1.
This leads to the points spacing out.
Hence the 2nd closest neighbour gets further away.

However, since now the spread of the points is closer to the query point.
Imagine a sphere of some radius (<1) in both 2D Tree space and 3D tree space.
So, since in 2D case this sphere only interacts with one plane, it can have some points within the space.
In the 3D tree case however, due to a extra-dimension to spread the dataset, there are more points within the sphere and hence, the distance to the 100th Nearest neighbout keeps decreasing.

What this results in is the 100^{th} neighbour coming closer to the query point as the dimensions keep increasing.

This results in an overall increase in the ratio.