

CSI2110 Programming Assignment II

Brent Palmer (300193610)

School of Electrical Engineering and Computer Science, University of Ottawa

CSI2110[C/D] Data Structures and Algorithms

Lucia Moura

December 5, 2022

Experiment 1: Validation

Introduction

In the first experiment, the goal is to determine if the new KDtree and NearestNeighborsKD classes were implemented correctly. In order to validate the classes, six different points are tested using the original NearestNeighbors class as well as the NearestNeighborsKD class using the Exp1 class. The neighbors discovered by these algorithms are output to the terminal and manually saved into text files. If NearestNeighborsKD finds the same neighbors as the original NearestNeighbors class, then the new KDtree and NearestNeighborsKD classes were implemented correctly. It is important to note that the order the points are output in are not always expected to be the same.

Results

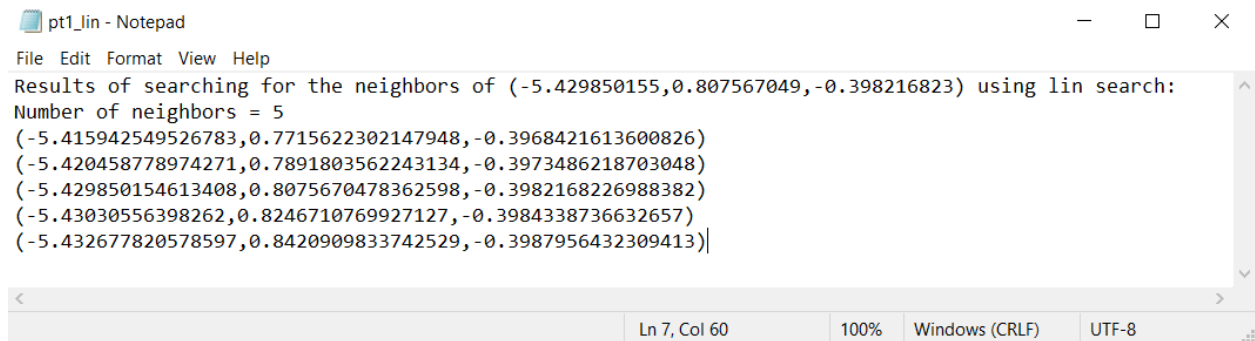
***Screenshots directly from terminal included in Appendix if preferred (displays student number in directory of commands)*

***Discussion of results below*

***View Appendix to see all lines used to execute my versions of Exp1, Exp2, Exp3*

Test Point 1:

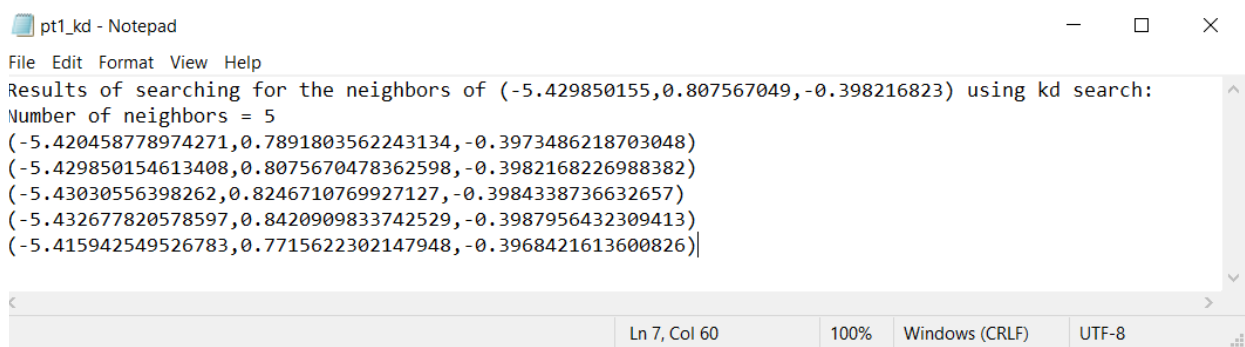
```
java Exp1 lin 0.05 Point_Cloud_1.csv -5.429850155 0.807567049 -0.398216823
```



```
pt1_lin - Notepad
File Edit Format View Help
Results of searching for the neighbors of (-5.429850155,0.807567049,-0.398216823) using lin search:
Number of neighbors = 5
(-5.415942549526783,0.7715622302147948,-0.3968421613600826)
(-5.420458778974271,0.7891803562243134,-0.3973486218703048)
(-5.429850154613408,0.8075670478362598,-0.3982168226988382)
(-5.43030556398262,0.8246710769927127,-0.3984338736632657)
(-5.432677820578597,0.8420909833742529,-0.3987956432309413)|
Ln 7, Col 60 100% Windows (CRLF) UTF-8
```

Figure 1: Neighbors of point 1 using lin search

java Exp1 kd 0.05 Point_Cloud_1.csv -5.429850155 0.807567049 -0.398216823



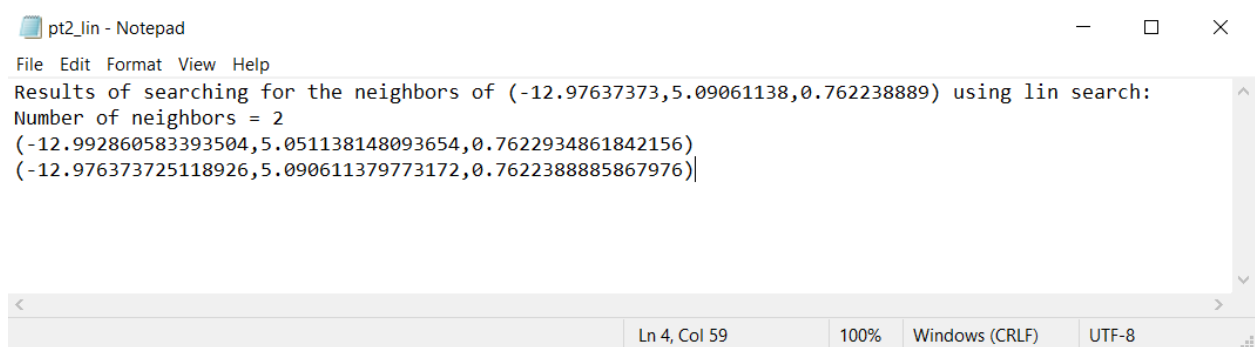
```
pt1_kd - Notepad
File Edit Format View Help
Results of searching for the neighbors of (-5.429850155,0.807567049,-0.398216823) using kd search:
Number of neighbors = 5
(-5.420458778974271,0.7891803562243134,-0.3973486218703048)
(-5.429850154613408,0.8075670478362598,-0.3982168226988382)
(-5.43030556398262,0.8246710769927127,-0.3984338736632657)
(-5.432677820578597,0.8420909833742529,-0.3987956432309413)
(-5.415942549526783,0.7715622302147948,-0.3968421613600826)|
```

Ln 7, Col 60 100% Windows (CRLF) UTF-8

Figure 2: Neighbors of point 1 using kd search

Test Point 2:

java Exp1 lin 0.05 Point_Cloud_1.csv -12.97637373 5.09061138 0.762238889

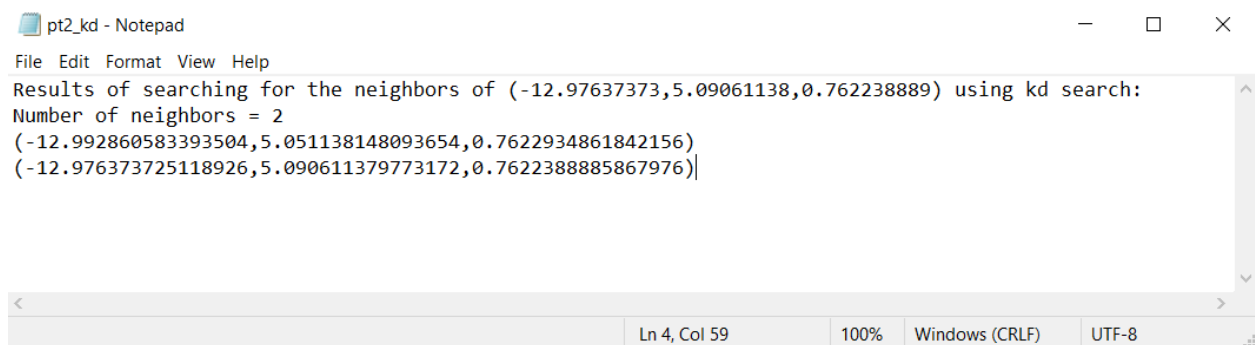


```
pt2_lin - Notepad
File Edit Format View Help
Results of searching for the neighbors of (-12.97637373,5.09061138,0.762238889) using lin search:
Number of neighbors = 2
(-12.992860583393504,5.051138148093654,0.7622934861842156)
(-12.976373725118926,5.090611379773172,0.7622388885867976)|
```

Ln 4, Col 59 100% Windows (CRLF) UTF-8

Figure 3: Neighbors of point 2 using lin search

java Exp1 kd 0.05 Point_Cloud_1.csv -12.97637373 5.09061138 0.762238889



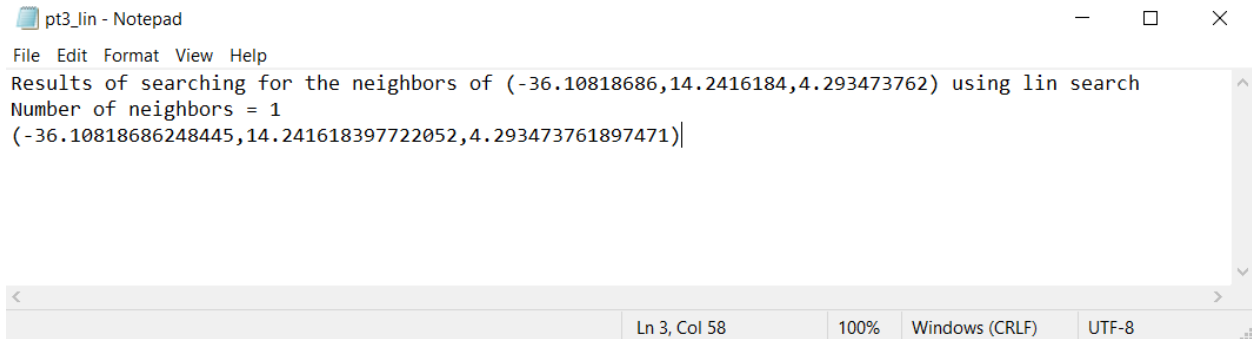
```
pt2_kd - Notepad
File Edit Format View Help
Results of searching for the neighbors of (-12.97637373,5.09061138,0.762238889) using kd search:
Number of neighbors = 2
(-12.992860583393504,5.051138148093654,0.7622934861842156)
(-12.976373725118926,5.090611379773172,0.7622388885867976)|
```

Ln 4, Col 59 100% Windows (CRLF) UTF-8

Figure 4: Neighbors of point 2 using kd search

Test Point 3:

```
java Exp1 lin 0.05 Point_Cloud_1.csv -36.10818686 14.2416184 4.293473762
```



pt3_lin - Notepad

File Edit Format View Help

Results of searching for the neighbors of (-36.10818686,14.2416184,4.293473762) using lin search

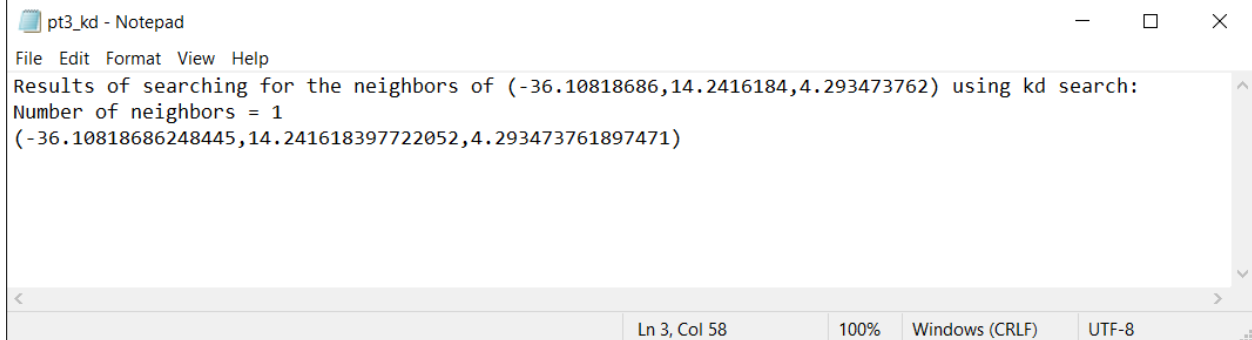
Number of neighbors = 1

(-36.10818686248445,14.241618397722052,4.293473761897471)|

Ln 3, Col 58 100% Windows (CRLF) UTF-8

Figure 5: Neighbors of point 3 using lin search

```
java Exp1 kd 0.05 Point_Cloud_1.csv -36.10818686 14.2416184 4.293473762
```



pt3_kd - Notepad

File Edit Format View Help

Results of searching for the neighbors of (-36.10818686,14.2416184,4.293473762) using kd search:

Number of neighbors = 1

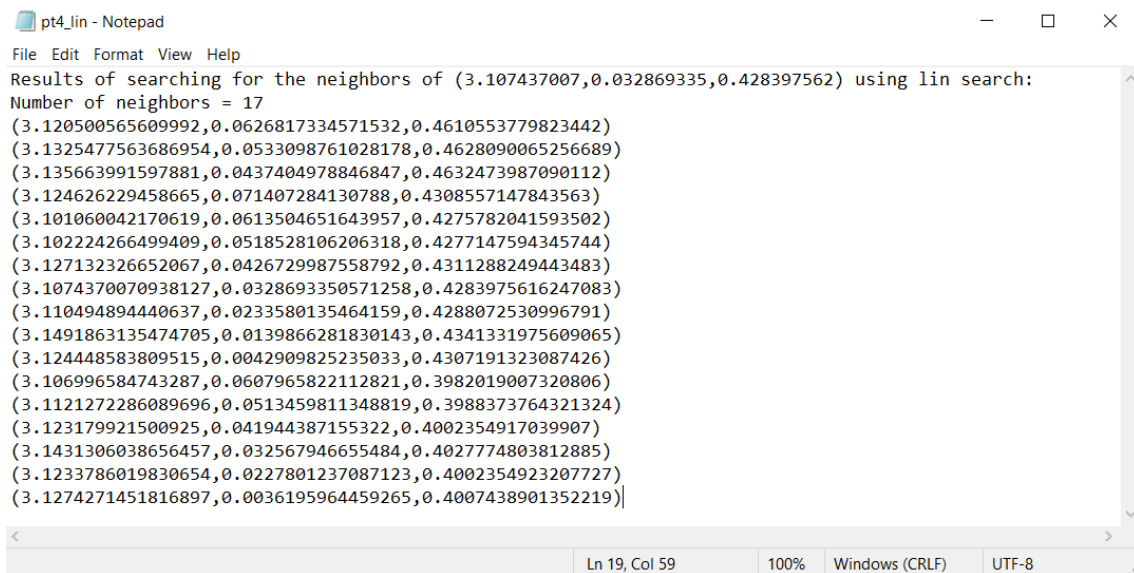
(-36.10818686248445,14.241618397722052,4.293473761897471)

Ln 3, Col 58 100% Windows (CRLF) UTF-8

Figure 6: Neighbors of point 3 using kd search

Test Point 4:

```
java Exp1 lin 0.05 Point_Cloud_1.csv 3.107437007 0.032869335 0.428397562
```



pt4_lin - Notepad

File Edit Format View Help

Results of searching for the neighbors of (3.107437007,0.032869335,0.428397562) using lin search:

Number of neighbors = 17

(3.120500565609992,0.0626817334571532,0.4610553779823442)

(3.1325477563686954,0.0533098761028178,0.4628090065256689)

(3.135663991597881,0.0437404978846847,0.4632473987090112)

(3.124626229458665,0.071407284130788,0.4308557147843563)

(3.101060042170619,0.0613504651643957,0.4275782041593502)

(3.102224266499409,0.0518528106206318,0.4277147594345744)

(3.127132326652067,0.0426729987558792,0.4311288249443483)

(3.1074370070938127,0.0328693350571258,0.4283975616247083)

(3.110494894440637,0.0233580135464159,0.4288072530996791)

(3.1491863135474705,0.0139866281830143,0.4341331975609065)

(3.124448583809515,0.0042909825235033,0.4307191323087426)

(3.106996584743287,0.0607965822112821,0.3982019007320806)

(3.1121272286089696,0.0513459811348819,0.3988373764321324)

(3.123179921500925,0.041944387155322,0.4002354917039907)

(3.1431306038656457,0.032567946655484,0.4027774803812885)

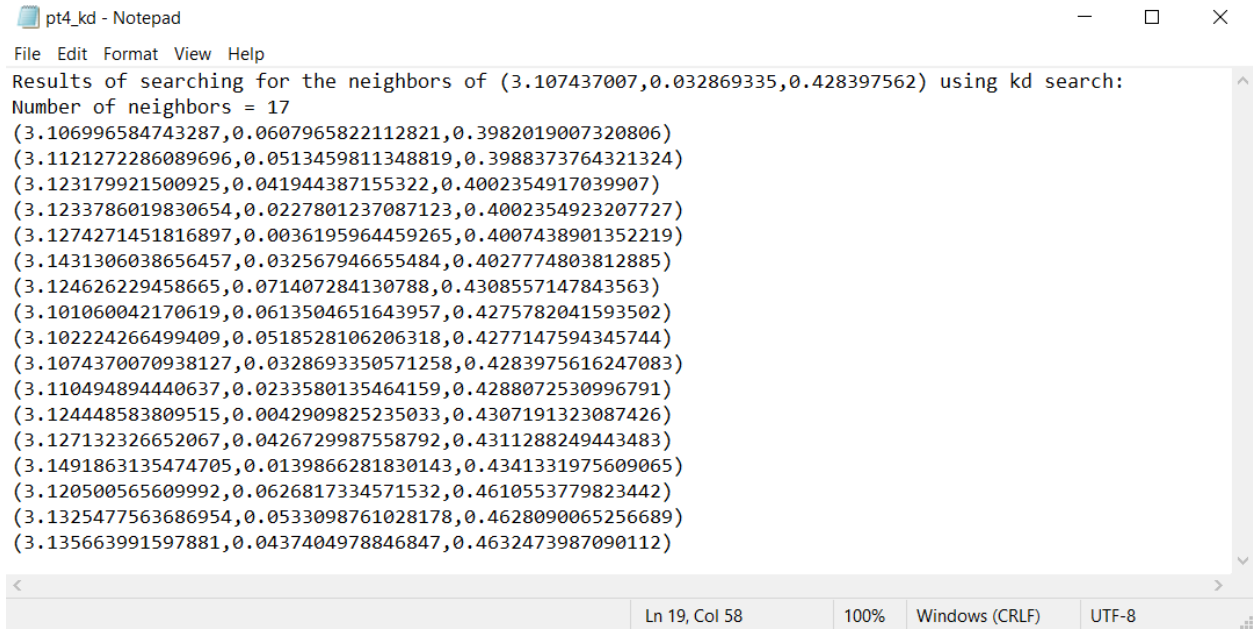
(3.1233786019830654,0.0227801237087123,0.4002354923207727)

(3.1274271451816897,0.0036195964459265,0.4007438901352219)|

Ln 19, Col 59 100% Windows (CRLF) UTF-8

Figure 7: Neighbors of point 4 using lin search

java Exp1 kd 0.05 Point_Cloud_1.csv 3.107437007 0.032869335 0.428397562

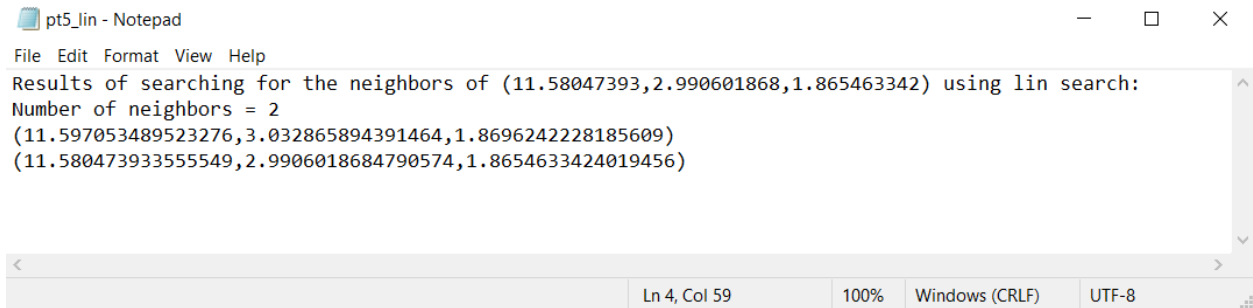


```
pt4_kd - Notepad
File Edit Format View Help
Results of searching for the neighbors of (3.107437007,0.032869335,0.428397562) using kd search:
Number of neighbors = 17
(3.106996584743287,0.0607965822112821,0.3982019007320806)
(3.1121272286089696,0.0513459811348819,0.3988373764321324)
(3.123179921500925,0.041944387155322,0.4002354917039907)
(3.1233786019830654,0.0227801237087123,0.4002354923207727)
(3.1274271451816897,0.0036195964459265,0.4007438901352219)
(3.1431306038656457,0.032567946655484,0.4027774803812885)
(3.124626229458665,0.071407284130788,0.4308557147843563)
(3.101060042170619,0.0613504651643957,0.4275782041593502)
(3.102224266499409,0.0518528106206318,0.4277147594345744)
(3.1074370070938127,0.0328693350571258,0.4283975616247083)
(3.110494894440637,0.0233580135464159,0.4288072530996791)
(3.124448583809515,0.0042909825235033,0.4307191323087426)
(3.127132326652067,0.0426729987558792,0.4311288249443483)
(3.1491863135474705,0.0139866281830143,0.4341331975609065)
(3.120500565609992,0.0626817334571532,0.4610553779823442)
(3.1325477563686954,0.0533098761028178,0.4628090065256689)
(3.135663991597881,0.0437404978846847,0.4632473987090112)
```

Figure 8: Neighbors of point 4 using kd search

Test Point 5:

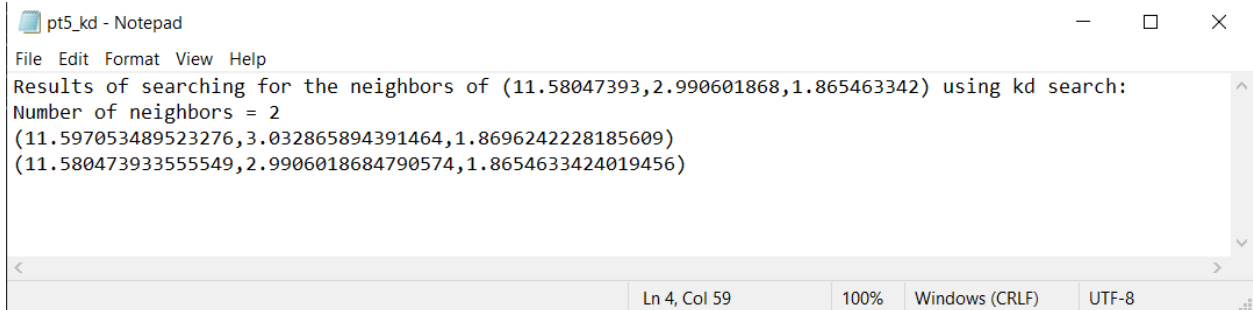
java Exp1 lin 0.05 Point_Cloud_1.csv 11.58047393 2.990601868 1.865463342



```
pt5_lin - Notepad
File Edit Format View Help
Results of searching for the neighbors of (11.58047393,2.990601868,1.865463342) using lin search:
Number of neighbors = 2
(11.597053489523276,3.032865894391464,1.8696242228185609)
(11.580473933555549,2.9906018684790574,1.8654633424019456)
```

Figure 9: Neighbors of point 5 using lin search

java Exp1 kd 0.05 Point_Cloud_1.csv 11.58047393 2.990601868 1.865463342

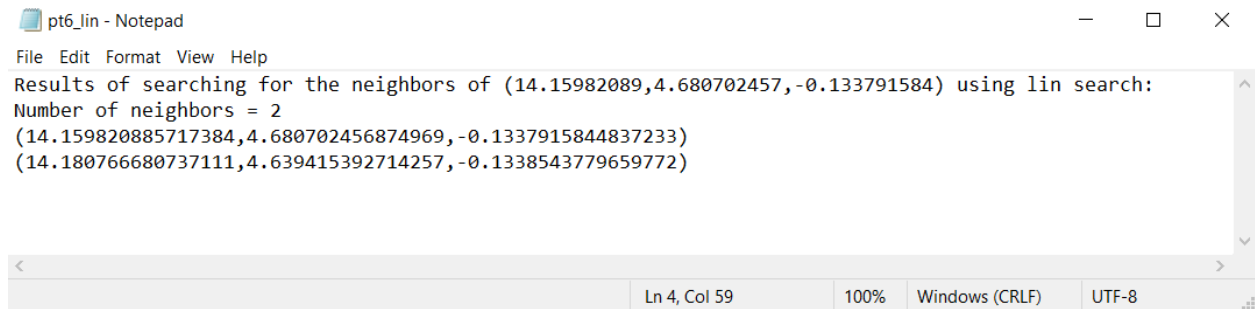


```
pt5_kd - Notepad
File Edit Format View Help
Results of searching for the neighbors of (11.58047393,2.990601868,1.865463342) using kd search:
Number of neighbors = 2
(11.597053489523276,3.032865894391464,1.8696242228185609)
(11.580473933555549,2.9906018684790574,1.8654633424019456)
```

Figure 10: Neighbors of point 5 using kd search

Test Point 6:

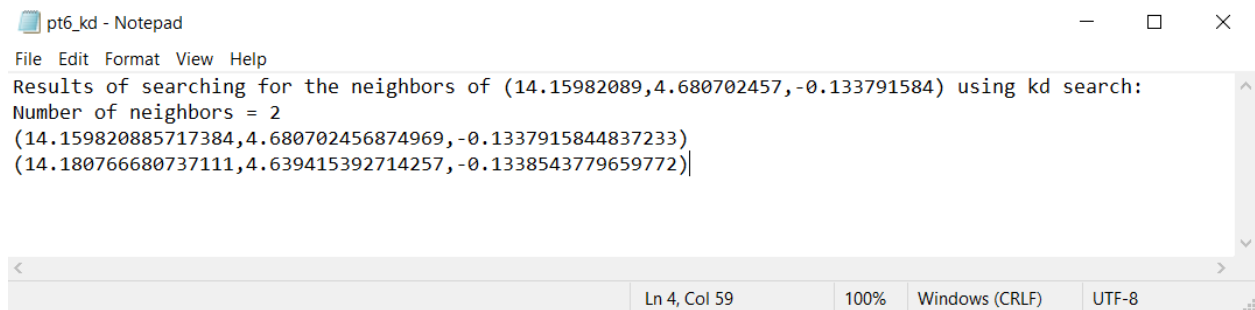
```
java Exp1 lin 0.05 Point_Cloud_1.csv 14.15982089 4.680702457 -0.133791584
```



```
pt6_lin - Notepad
File Edit Format View Help
Results of searching for the neighbors of (14.15982089,4.680702457,-0.133791584) using lin search:
Number of neighbors = 2
(14.159820885717384,4.680702456874969,-0.1337915844837233)
(14.180766680737111,4.639415392714257,-0.1338543779659772)
Ln 4, Col 59 100% Windows (CRLF) UTF-8
```

Figure 11: Neighbors of point 6 using lin search

```
java Exp1 kd 0.05 Point_Cloud_1.csv 14.15982089 4.680702457 -0.133791584
```



```
pt6_kd - Notepad
File Edit Format View Help
Results of searching for the neighbors of (14.15982089,4.680702457,-0.133791584) using kd search:
Number of neighbors = 2
(14.159820885717384,4.680702456874969,-0.1337915844837233)
(14.180766680737111,4.639415392714257,-0.1338543779659772)
Ln 4, Col 59 100% Windows (CRLF) UTF-8
```

Figure 12: Neighbors of point 6 using kd search

Tests to Confirm Both Methods Give Same Results

The results for each of these 6 points admittedly do not return many points. It was most efficient for me to simply eyeball each of the results. I compared the results in the terminal (screenshots included below), as I would run the lin and kd for each point consecutively. The results would be side by side, so I simply went point by point from the lin output and made sure the point existed in the kd output. It was a very brief process for all points (except point 4), and ultimately both methods gave the same results for each method. This confirms that KDtree and NearestNeighborsKD were implemented correctly. The 12 output files pictured above are included in the exp1 folder.

Experiment 2: Computational Time

Summary

In the second experiment, the goal is to determine the time required to find the neighbors of every 10 points in the given Point Cloud files. An eps of 0.5 is used for these computations. In order to execute this experiment, a loop must iterate over every 10 points from the list of points (implemented with $i+=\text{step}$ where step is 10). In each iteration, the time before and after the rangeQuery of the i^{th} point is measured to calculate the time the rangeQuery takes. The difference is stored into an accumulator that accumulates the total running time of all the rangeQueries. An additional counter is incremented that tracks the number of points that have been processed. Afterwards, the total accumulated time across all range queries is divided by the number of points processed to determine the average processing of each rangeQuery.

Results

Point Cloud 1:

```
C:\CSI2110\ProgrammingAssignment2_300193610>java Exp2 lin 0.5 Point_Cloud_1.csv 10
The running total is 666882535ns
The number of points processed is 2964
Average time to find the neighbors of every 10 points in file Point_Cloud_1.csv using lin search: 0.224994ms

C:\CSI2110\ProgrammingAssignment2_300193610>java Exp2 kd 0.5 Point_Cloud_1.csv 10
The running total is 100948393ns
The number of points processed is 2964
Average time to find the neighbors of every 10 points in file Point_Cloud_1.csv using kd search: 0.034058ms
```

Figure 13: Average time of rangeQuery running Point_Cloud_1

Point Cloud 2:

```
C:\CSI2110\ProgrammingAssignment2_300193610>java Exp2 lin 0.5 Point_Cloud_2.csv 10
The running total is 2012057801ns
The number of points processed is 5083
Average time to find the neighbors of every 10 points in file Point_Cloud_2.csv using lin search: 0.39584ms

C:\CSI2110\ProgrammingAssignment2_300193610>java Exp2 kd 0.5 Point_Cloud_2.csv 10
The running total is 284758009ns
The number of points processed is 5083
Average time to find the neighbors of every 10 points in file Point_Cloud_2.csv using kd search: 0.056021ms
```

Figure 14: Average time of rangeQuery running Point_Cloud_2

Point Cloud 3:

```
C:\CSI2110\ProgrammingAssignment2_300193610>java Exp2 lin 0.5 Point_Cloud_3.csv 10
The running total is 1901363673ns
The number of points processed is 4515
Average time to find the neighbors of every 10 points in file Point_Cloud_3.csv using lin search: 0.421121ms

C:\CSI2110\ProgrammingAssignment2_300193610>java Exp2 kd 0.5 Point_Cloud_3.csv 10
The running total is 156823041ns
The number of points processed is 4515
Average time to find the neighbors of every 10 points in file Point_Cloud_3.csv using kd search: 0.034733ms
```

Figure 15: Average time of rangeQuery running Point_Cloud_3

Table 1: Results average rangeQuery times using different searching algorithms in tabular format

	Linear Search (ms)	K-D Search (ms)
Point_Cloud_1	0.224994	0.034058
Point_Cloud_2	0.39583	0.056021
Point_Cloud_3	0.421121	0.034733
Average	0.347315	0.041604

Discussion of the Results

The compute times are exactly as expected. The rangeQuery that uses linear search, as implied by the name, runs in linear time. This has an algorithm complexity of $O(n)$. The rangeQuery that uses k-d search, however, uses a tree that typically has a depth around $O(\log(n))$. This means that the k-d search should be running with an algorithm complexity of around $O(\log(n))$. In other words, the expected complexity is $O(\log(n))$, but a perfectly unbalanced tree could theoretically give a complexity of $O(n^2)$. After observing the results, the relative times measured between the linear search and the k-d search seem to reflect the expected complexities of $O(n)$ and $O(\log(n))$. Between the three point clouds, the linear search takes 0.347315ms on average to run each rangeQuery. The k-d search, however, only takes 0.041604ms on average to run each rangeQuery. The k-d search runs rangeQuery much faster (a factor of roughly 8.35x faster) relative to the linear search, and this relativity is the expectation when comparing an algorithm with complexity of $O(\log(n))$ to another algorithm that has complexity of $O(n)$. Ultimately, the results of measuring the average time of each rangeQuery using linear and k-d searches give results that agree with the expected algorithm complexities of $O(n)$ and $O(\log(n))$ respectively.

Experiment 3: Integration to DBScan

Summary

The final set of experiments puts everything together. The goal is to compare the runtime of DBScan using NearestNeighbors, which uses the linear search, to DBScan using NearestNeighborsKD, which uses the k-d search. Conducting the experiment is quite simple. The time is measured before and after running the main of DBScan, and the difference is the length it took to run DBScan using the selected algorithm. To remain consistent between all tests, I used consistent eps and minPts values throughout all the tests, being set to 1.2 and 10 respectively.

Results

Point Cloud 1:

```
C:\CSI2110\ProgrammingAssignment2_300193610>java Exp3 lin 1.2 10 Point_Cloud_1.csv
Cluster 25 contains 9800 points
Cluster 29 contains 8033 points
Cluster 3 contains 2484 points
Cluster 8 contains 2029 points
Cluster 24 contains 1730 points
Cluster 1 contains 1445 points
Cluster 4 contains 779 points
Cluster 11 contains 726 points
Cluster 21 contains 657 points
Cluster 20 contains 342 points
Cluster 2 contains 328 points
Cluster 27 contains 268 points
Cluster 16 contains 202 points
Cluster 5 contains 192 points
Cluster 13 contains 138 points
Cluster 23 contains 104 points
Cluster 28 contains 93 points
Cluster 6 contains 56 points
Cluster 12 contains 45 points
Cluster 22 contains 22 points
Cluster 19 contains 22 points
Cluster 9 contains 18 points
Cluster 18 contains 16 points
Cluster 26 contains 15 points
Cluster 15 contains 14 points
Cluster 17 contains 13 points
Cluster 14 contains 13 points
Cluster 10 contains 13 points
Cluster 7 contains 13 points
Noise cluster contains 23 points
The total time to run DBScan on file Point_Cloud_1.csv using lin search is 12307.4926ms
```

Figure 16: Total runtime of DBScan using NearestNeighbors on Point_Cloud_1

```

C:\CSI2110\ProgrammingAssignment2_300193610>java Exp3 kd 1.2 10 Point_Cloud_1.csv
Cluster 25 contains 9800 points
Cluster 29 contains 8033 points
Cluster 3 contains 2484 points
Cluster 8 contains 2029 points
Cluster 24 contains 1730 points
Cluster 1 contains 1445 points
Cluster 4 contains 779 points
Cluster 11 contains 726 points
Cluster 21 contains 657 points
Cluster 20 contains 342 points
Cluster 2 contains 328 points
Cluster 27 contains 268 points
Cluster 16 contains 202 points
Cluster 5 contains 192 points
Cluster 13 contains 138 points
Cluster 23 contains 104 points
Cluster 28 contains 93 points
Cluster 6 contains 56 points
Cluster 12 contains 45 points
Cluster 22 contains 22 points
Cluster 19 contains 22 points
Cluster 9 contains 18 points
Cluster 18 contains 16 points
Cluster 26 contains 15 points
Cluster 15 contains 14 points
Cluster 17 contains 13 points
Cluster 14 contains 13 points
Cluster 10 contains 13 points
Cluster 7 contains 13 points
Noise cluster contains 23 points
The total time to run DBScan on file Point_Cloud_1.csv using kd search is 6605.8307ms

```

Figure 17: Total runtime of DBScan using NearestNeighborsKD on Point_Cloud_1

Point Cloud 2:

```

C:\CSI2110\ProgrammingAssignment2_300193610>java Exp3 lin 1.2 10 Point_Cloud_2.csv
Cluster 26 contains 28960 points
Cluster 1 contains 13695 points
Cluster 10 contains 2949 points
Cluster 23 contains 1691 points
Cluster 12 contains 860 points
Cluster 2 contains 698 points
Cluster 8 contains 672 points
Cluster 7 contains 355 points
Cluster 17 contains 199 points
Cluster 6 contains 125 points
Cluster 15 contains 82 points
Cluster 27 contains 77 points
Cluster 13 contains 57 points
Cluster 5 contains 55 points
Cluster 21 contains 53 points
Cluster 29 contains 30 points
Cluster 20 contains 27 points
Cluster 28 contains 25 points
Cluster 25 contains 24 points
Cluster 24 contains 24 points
Cluster 22 contains 20 points
Cluster 14 contains 19 points
Cluster 11 contains 19 points
Cluster 18 contains 18 points
Cluster 16 contains 17 points
Cluster 4 contains 17 points
Cluster 3 contains 16 points
Cluster 9 contains 15 points
Cluster 19 contains 10 points
Noise cluster contains 12 points
The total time to run DBScan on file Point_Cloud_2.csv using lin search is 34199.2805ms

```

Figure 18: Total runtime of DBScan using NearestNeighbors on Point_Cloud_2

```

C:\CSI2110\ProgrammingAssignment2_300193610>java Exp3 kd 1.2 10 Point_Cloud_2.csv
Cluster 26 contains 28960 points
Cluster 1 contains 13695 points
Cluster 10 contains 2949 points
Cluster 23 contains 1691 points
Cluster 12 contains 860 points
Cluster 2 contains 698 points
Cluster 8 contains 672 points
Cluster 7 contains 355 points
Cluster 17 contains 199 points
Cluster 6 contains 125 points
Cluster 15 contains 82 points
Cluster 27 contains 77 points
Cluster 13 contains 57 points
Cluster 5 contains 55 points
Cluster 21 contains 53 points
Cluster 29 contains 30 points
Cluster 20 contains 27 points
Cluster 28 contains 25 points
Cluster 25 contains 24 points
Cluster 24 contains 24 points
Cluster 22 contains 20 points
Cluster 14 contains 19 points
Cluster 11 contains 19 points
Cluster 18 contains 18 points
Cluster 16 contains 17 points
Cluster 4 contains 17 points
Cluster 3 contains 16 points
Cluster 9 contains 15 points
Cluster 19 contains 10 points
Noise cluster contains 12 points
The total time to run DBScan on file Point_Cloud_2.csv using kd search is 16709.3152ms

```

Figure 19: Total runtime of DBScan using NearestNeighborsKD on Point_Cloud_2

Point Cloud 3:

```

C:\CSI2110\ProgrammingAssignment2_300193610>java Exp3 lin 1.2 10 Point_Cloud_3.csv
Cluster 8 contains 13401 points
Cluster 32 contains 10525 points
Cluster 35 contains 8086 points
Cluster 3 contains 6354 points
Cluster 6 contains 3170 points
Cluster 33 contains 862 points
Cluster 2 contains 363 points
Cluster 10 contains 312 points
Cluster 34 contains 297 points
Cluster 26 contains 280 points
Cluster 37 contains 239 points
Cluster 30 contains 209 points
Cluster 29 contains 137 points
Cluster 1 contains 93 points
Cluster 5 contains 92 points
Cluster 13 contains 75 points
Cluster 18 contains 65 points
Cluster 36 contains 62 points
Cluster 38 contains 61 points
Cluster 27 contains 39 points
Cluster 23 contains 39 points
Cluster 31 contains 32 points
Cluster 25 contains 31 points
Cluster 21 contains 28 points
Cluster 15 contains 28 points
Cluster 39 contains 23 points
Cluster 16 contains 23 points
Cluster 4 contains 22 points
Cluster 19 contains 20 points
Cluster 12 contains 20 points
Cluster 40 contains 19 points
Cluster 22 contains 17 points
Cluster 20 contains 17 points
Cluster 24 contains 16 points
Cluster 11 contains 15 points
Cluster 17 contains 14 points
Cluster 28 contains 13 points
Cluster 9 contains 12 points
Cluster 7 contains 12 points
Cluster 14 contains 11 points
Noise cluster contains 7 points
The total time to run DBScan on file Point_Cloud_3.csv using lin search is 23060.0681ms

```

Figure 20: Total runtime of DBScan using NearestNeighbors on Point_Cloud_3

```

C:\CSI2110\ProgrammingAssignment2_300193610>java Exp3 kd 1.2 10 Point_Cloud_3.csv
Cluster 8 contains 13401 points
Cluster 32 contains 10525 points
Cluster 35 contains 8086 points
Cluster 3 contains 6354 points
Cluster 6 contains 3170 points
Cluster 33 contains 862 points
Cluster 2 contains 363 points
Cluster 10 contains 312 points
Cluster 34 contains 297 points
Cluster 26 contains 280 points
Cluster 37 contains 239 points
Cluster 30 contains 209 points
Cluster 29 contains 137 points
Cluster 1 contains 93 points
Cluster 5 contains 92 points
Cluster 13 contains 75 points
Cluster 18 contains 65 points
Cluster 36 contains 62 points
Cluster 38 contains 61 points
Cluster 27 contains 39 points
Cluster 23 contains 39 points
Cluster 31 contains 32 points
Cluster 25 contains 31 points
Cluster 21 contains 28 points
Cluster 15 contains 28 points
Cluster 39 contains 23 points
Cluster 16 contains 23 points
Cluster 4 contains 22 points
Cluster 19 contains 20 points
Cluster 12 contains 20 points
Cluster 40 contains 19 points
Cluster 22 contains 17 points
Cluster 20 contains 17 points
Cluster 24 contains 16 points
Cluster 11 contains 15 points
Cluster 17 contains 14 points
Cluster 28 contains 13 points
Cluster 9 contains 12 points
Cluster 7 contains 12 points
Cluster 14 contains 11 points
Noise cluster contains 7 points
The total time to run DBScan on file Point_Cloud_3.csv using kd search is 10134.8802ms

```

Figure 21: Total runtime of DBScan using NearestNeighborsKD on Point_Cloud_3

Table 2: Results average DBScan times using different searching algorithms in tabular format

	Linear Search (ms)	K-D Search (ms)
Point_Cloud_1	12307.4926	6605.8307
Point_Cloud_2	34199.2805	16709.3152
Point_Cloud_3	23060.0681	10134.8802
Average	23188.94707	11150.0087

Note: NearestNeighbors uses linear search and NearestNeighborsKD uses k-d search

Discussion of the Results

The runtime results of running DBScan with NearestNeighbors and NearestNeighborsKD is exactly what was expected. The time complexity of using DBScan with NearestNeighbors is expected to be $O(n^2)$, as all n points need to be visited and the neighborhood of each point must be searched. Since NearestNeighbors uses a linear search, that is n points using a searching algorithm of complexity $O(n)$. Multiplying $n*n$ yields an algorithm complexity of $O(n^2)$. The time complexity of using DBScan with NearestNeighborsKD is expected to be around $O(n*\log(n))$, as all n points need to be visited and the neighborhood of each point must be searched. NearestNeighborsKD, however, uses a k-d tree that should have a depth of $O(\log(n))$. Therefore this search is expected to have complexity of $O(\log(n))$. It is worth noting it could have a complexity of $O(n)$ with a perfectly unbalanced tree. Since NearestNeighborsKD uses k-d search, that is n points using a searching algorithm of expected complexity $O(\log(n))$. Multiplying $n*\log(n)$ yields an algorithm complexity of $O(n*\log(n))$ in the expected case. The worst case could potentially be $O(n^2)$.

The expectations have been set. Analyzing the results reveal that running DBScan with NearestNeighbors and NearestNeighborsKD matches the anticipated results. Running Point_Cloud_1, Point_Cloud_2, and Point_Cloud_3, DBScan is consistently significantly faster running NearestNeighborsKD relative to NearestNeighbors. In each of the tests, DBScan runs about twice as fast running NearestNeighborsKD. If an average was computed between the three test clouds, NearestNeighborsKD ran a factor of roughly 2.08x faster than NearestNeighbors. It makes sense that NearestNeighborsKD, the algorithm that is expected to run with a complexity of $O(n*\log(n))$, runs much faster than NearestNeighbors, the algorithm that is expected to run with a complexity of $O(n^2)$.

Appendix

List of All Test Case Terminal Inputs

Experiment 1

Point 1

```
java Exp1 lin 0.05 Point_Cloud_1.csv -5.429850155 0.807567049 -0.398216823
```

```
java Exp1 kd 0.05 Point_Cloud_1.csv -5.429850155 0.807567049 -0.398216823
```

Point 2

```
java Exp1 lin 0.05 Point_Cloud_1.csv -12.97637373 5.09061138 0.762238889
```

```
java Exp1 kd 0.05 Point_Cloud_1.csv -12.97637373 5.09061138 0.762238889
```

Point 3

```
java Exp1 lin 0.05 Point_Cloud_1.csv -36.10818686 14.2416184 4.293473762
```

```
java Exp1 kd 0.05 Point_Cloud_1.csv -36.10818686 14.2416184 4.293473762
```

Point 4

```
java Exp1 lin 0.05 Point_Cloud_1.csv 3.107437007 0.032869335 0.428397562
```

```
java Exp1 kd 0.05 Point_Cloud_1.csv 3.107437007 0.032869335 0.428397562
```

Point 5

```
java Exp1 lin 0.05 Point_Cloud_1.csv 11.58047393 2.990601868 1.865463342
```

```
java Exp1 kd 0.05 Point_Cloud_1.csv 11.58047393 2.990601868 1.865463342
```

Point 6

```
java Exp1 lin 0.05 Point_Cloud_1.csv 14.15982089 4.680702457 -0.133791584
```

```
java Exp1 kd 0.05 Point_Cloud_1.csv 14.15982089 4.680702457 -0.133791584
```

Experiment 2

Point_Cloud_1:

```
java Exp2 lin 0.5 Point_Cloud_1.csv 10
```

```
java Exp2 kd 0.5 Point_Cloud_1.csv 10
```

Point_Cloud_2:

```
java Exp2 lin 0.5 Point_Cloud_2.csv 10
```

```
java Exp2 kd 0.5 Point_Cloud_2.csv 10
```

Point_Cloud_3:

```
java Exp2 lin 0.5 Point_Cloud_3.csv 10
```

```
java Exp2 kd 0.5 Point_Cloud_3.csv 10
```

Experiment 3:

Point_Cloud_1:

```
java Exp3 lin 1.2 10 Point_Cloud_1.csv
```

```
java Exp3 kd 1.2 10 Point_Cloud_1.csv
```

Point_Cloud_2:

```
java Exp3 lin 1.2 10 Point_Cloud_2.csv
```

```
java Exp3 kd 1.2 10 Point_Cloud_2.csv
```

Point_Cloud_3:

```
java Exp3 lin 1.2 10 Point_Cloud_3.csv
```

```
java Exp3 kd 1.2 10 Point_Cloud_3.csv
```

Outputs directly from the terminal for experiment 1:

```
C:\CSI2110\ProgrammingAssignment2_300193610>java Exp1 lin 0.05 Point_Cloud_1.csv -5.429850155 0.807567049 -0.398216823
Results of searching for the neighbors of (-5.429850155,0.807567049,-0.398216823) using lin search:
Number of neighbors = 5
(-5.415942549526783,0.7715622302147948,-0.3968421613600826)
(-5.420458778974271,0.7891803562243134,-0.3973486218703048)
(-5.429850154613408,0.8075670478362598,-0.3982168226988382)
(-5.43030556398262,0.8246710769927127,-0.3984338736632657)
(-5.432677820578597,0.8420909833742529,-0.3987956432309413)

C:\CSI2110\ProgrammingAssignment2_300193610>java Exp1 kd 0.05 Point_Cloud_1.csv -5.429850155 0.807567049 -0.398216823
Results of searching for the neighbors of (-5.429850155,0.807567049,-0.398216823) using kd search:
Number of neighbors = 5
(-5.420458778974271,0.7891803562243134,-0.3973486218703048)
(-5.429850154613408,0.8075670478362598,-0.3982168226988382)
(-5.43030556398262,0.8246710769927127,-0.3984338736632657)
(-5.432677820578597,0.8420909833742529,-0.3987956432309413)
(-5.415942549526783,0.7715622302147948,-0.3968421613600826)

C:\CSI2110\ProgrammingAssignment2_300193610>java Exp1 lin 0.05 Point_Cloud_1.csv -12.97637373 5.09061138 0.762238889
Results of searching for the neighbors of (-12.97637373,5.09061138,0.762238889) using lin search:
Number of neighbors = 2
(-12.992860583393504,5.051138148093654,0.7622934861842156)
(-12.976373725118926,5.090611379773172,0.7622388885867976)

C:\CSI2110\ProgrammingAssignment2_300193610>java Exp1 kd 0.05 Point_Cloud_1.csv -12.97637373 5.09061138 0.762238889
Results of searching for the neighbors of (-12.97637373,5.09061138,0.762238889) using kd search:
Number of neighbors = 2
(-12.992860583393504,5.051138148093654,0.7622934861842156)
(-12.976373725118926,5.090611379773172,0.7622388885867976)
```

```
C:\CSI2110\ProgrammingAssignment2_300193610>java Exp1 lin 0.05 Point_Cloud_1.csv -36.10818686 14.2416184 4.293473762
Results of searching for the neighbors of (-36.10818686,14.2416184,4.293473762) using lin search:
Number of neighbors = 1
(-36.10818686248445,14.241618397722052,4.293473761897471)
```

```
C:\CSI2110\ProgrammingAssignment2_300193610>java Exp1 kd 0.05 Point_Cloud_1.csv -36.10818686 14.2416184 4.293473762
Results of searching for the neighbors of (-36.10818686,14.2416184,4.293473762) using kd search:
Number of neighbors = 1
(-36.10818686248445,14.241618397722052,4.293473761897471)
```

```
C:\CSI2110\ProgrammingAssignment2_300193610>java Exp1 lin 0.05 Point_Cloud_1.csv 3.107437007 0.032869335 0.428397562
Results of searching for the neighbors of (3.107437007,0.032869335,0.428397562) using lin search:
Number of neighbors = 17
```

```
(3.120500565609992,0.0626817334571532,0.4610553779823442)
(3.1325477563686954,0.0533098761028178,0.4628090065256689)
(3.135663991597881,0.0437404978846847,0.4632473987090112)
(3.124626229458665,0.071407284130788,0.4308557147843563)
(3.101060042170619,0.0613504651643957,0.4275782041593502)
(3.102224266499409,0.0518528106206318,0.4277147594345744)
(3.127132326652067,0.0426729987558792,0.4311288249443483)
(3.1074370070938127,0.0328693350571258,0.4283975616247083)
(3.110494894440637,0.0233580135464159,0.4288072530996791)
(3.1491863135474705,0.0139866281830143,0.4341331975609065)
(3.124448583809515,0.0042909825235033,0.4307191323087426)
(3.106996584743287,0.0607965822112821,0.3982019007320806)
(3.1121272286089696,0.0513459811348819,0.3988373764321324)
(3.123179921500925,0.041944387155322,0.4002354917039907)
(3.1431306038656457,0.032567946655484,0.4027774803812885)
(3.1233786019830654,0.0227801237087123,0.4002354923207727)
(3.1274271451816897,0.0036195964459265,0.4007438901352219)
```

```
C:\CSI2110\ProgrammingAssignment2_300193610>java Exp1 kd 0.05 Point_Cloud_1.csv 3.107437007 0.032869335 0.428397562
Results of searching for the neighbors of (3.107437007,0.032869335,0.428397562) using kd search:
Number of neighbors = 17
```

```
(3.106996584743287,0.0607965822112821,0.3982019007320806)
(3.1121272286089696,0.0513459811348819,0.3988373764321324)
(3.123179921500925,0.041944387155322,0.4002354917039907)
(3.1233786019830654,0.0227801237087123,0.4002354923207727)
(3.1274271451816897,0.0036195964459265,0.4007438901352219)
(3.1431306038656457,0.032567946655484,0.4027774803812885)
(3.124626229458665,0.071407284130788,0.4308557147843563)
(3.101060042170619,0.0613504651643957,0.4275782041593502)
(3.102224266499409,0.0518528106206318,0.4277147594345744)
(3.1074370070938127,0.0328693350571258,0.4283975616247083)
(3.110494894440637,0.0233580135464159,0.4288072530996791)
(3.124448583809515,0.0042909825235033,0.4307191323087426)
(3.127132326652067,0.0426729987558792,0.4311288249443483)
(3.1491863135474705,0.0139866281830143,0.4341331975609065)
(3.120500565609992,0.0626817334571532,0.4610553779823442)
(3.1325477563686954,0.0533098761028178,0.4628090065256689)
(3.135663991597881,0.0437404978846847,0.4632473987090112)
```



```
C:\CSI2110\ProgrammingAssignment2_300193610>java Exp1 lin 0.05 Point_Cloud_1.csv 11.58047393 2.990601868 1.865463342
Results of searching for the neighbors of (11.58047393,2.990601868,1.865463342) using lin search:
Number of neighbors = 2
(11.597053489523276,3.032865894391464,1.8696242228185609)
(11.580473933555549,2.9906018684790574,1.8654633424019456)
```

```
C:\CSI2110\ProgrammingAssignment2_300193610>java Exp1 kd 0.05 Point_Cloud_1.csv 11.58047393 2.990601868 1.865463342
Results of searching for the neighbors of (11.58047393,2.990601868,1.865463342) using kd search:
Number of neighbors = 2
(11.597053489523276,3.032865894391464,1.8696242228185609)
(11.580473933555549,2.9906018684790574,1.8654633424019456)
```

```
C:\CSI2110\ProgrammingAssignment2_300193610>java Exp1 lin 0.05 Point_Cloud_1.csv 14.15982089 4.680702457 -0.133791584
Results of searching for the neighbors of (14.15982089,4.680702457,-0.133791584) using lin search:
Number of neighbors = 2
(14.159820885717384,4.680702456874969,-0.1337915844837233)
(14.180766680737111,4.639415392714257,-0.1338543779659772)
```

```
C:\CSI2110\ProgrammingAssignment2_300193610>java Exp1 kd 0.05 Point_Cloud_1.csv 14.15982089 4.680702457 -0.133791584
Results of searching for the neighbors of (14.15982089,4.680702457,-0.133791584) using kd search:
Number of neighbors = 2
(14.159820885717384,4.680702456874969,-0.1337915844837233)
(14.180766680737111,4.639415392714257,-0.1338543779659772)
```