Rapport de DBScan partie 2

Ichrak El Hatimi
300217634

Experiment 1 results:

Point3D	Linear method neighbors	KD Tree method neighbors
1	-5.415942549526783,0.7715622302147948,-0.39 68421613600826, -5.420458778974271,0.7891803562243134,-0.39 73486218703048, -5.429850154613408,0.8075670478362598,-0.39 82168226988382, -5.43030556398262,0.8246710769927127,-0.398 4338736632657, -5.432677820578597,0.8420909833742529,-0.39 87956432309413]	[-5.420458778974271,0.7891803562243134,-0.39734862187 03048, -5.429850154613408,0.8075670478362598,-0.398216822698 8382, -5.43030556398262,0.8246710769927127,-0.3984338736632 657, -5.432677820578597,0.8420909833742529,-0.398795643230 9413, -5.415942549526783,0.7715622302147948,-0.396842161360 0826]
2	[-12.992860583393504,5.051138148093654,0.76 22293486184 2156, -12.976373725118926,5.090611379773172,0.762 2388885867976]	[-12.992860583393504,5.051138148093654,0.762293486184 2156, -12.976373725118926,5.090611379773172,0.762238888586 7976]
3	[-36.10818686248445,14.241618397722052,4.29 3473761897471]	[-36.10818686248445,14.241618397722052,4.293473761897 471]
4	[3.120500565609992,0.0626817334571532,0.461 0553779823442, 3.1325477563686954,0.0533098761028178,0.462 8090065256689, 3.135663991597881,0.0437404978846847,0.4632 473987090112, 3.124626229458665,0.071407284130788,0.43085 57147843563, 3.101060042170619,0.0613504651643957,0.4275 782041593502, 3.102224266499409,0.0518528106206318,0.4277 147594345744, 3.127132326652067,0.0426729987558792,0.4311 288249443483, 3.1074370070938127,0.0328693350571258,0.428 3975616247083, 3.110494894440637,0.0233580135464159,0.4288	[3.106996584743287,0.0607965822112821,0.398201900732 0806, 3.1121272286089696,0.0513459811348819,0.398837376432 1324, 3.123179921500925,0.041944387155322,0.40023549170399 07, 3.1233786019830654,0.0227801237087123,0.400235492320 7727, 3.1274271451816897,0.0036195964459265,0.400743890135 2219, 3.1431306038656457,0.032567946655484,0.4027774803812 885, 3.124626229458665,0.071407284130788,0.43085571478435 63, 3.101060042170619,0.0613504651643957,0.4275782041593 502, 3.102224266499409,0.0518528106206318,0.4277147594345

	072530996791, 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]	744, 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]
5	[11.597053489523276,3.032865894391464,1.8696242228185609, 11.580473933555549,2.9906018684790574,1.8654633424019456]	(11.597053489523276,3.032865894391464,1.8696242228185609) (11.580473933555549,2.9906018684790574,1.8654633424019456)
6	[14.159820885717384,4.680702456874969,-0.1337915844837233, 14.180766680737111,4.639415392714257,-0.1338543779659772]	[14.159820885717384,4.680702456874969,-0.1337915844837233, 14.180766680737111,4.639415392714257,-0.1338543779659772]

We notice that we have the same results regardless of which method was used, and that the only difference is the order of the points.

<u>Experiment 2 results:</u>

After running Exp2 we have noticed that rangeQuery using the KDTree method is a lot faster than rangeQuery using the linear method, which is exactly what we expected.

File	Running time in ms using KDTree and step parameter = 10	Running time in ms using linear method and step parameter = 10
Point_Cloud_1.csv	160	265
Point_Cloud_2.csv	272	719
Point_Cloud_3.csv	159	579

Experiment 3 results:

After running the DBScanKD class (which is basically just the DBScan that we implemented in part 1 except I used NearestNeighborsKD instead of the normal NearestNeighbors we used in part 1) and the normal DBScan we noticed that the running time for each Point Cloud file is :

File	Running time in ms using KDTree	Running time in ms using linear method
Point_Cloud_1.csv	1160	4079
Point_Cloud_2.csv	3168	12211
Point_Cloud_3.csv	1719	9273