INTRODUCTION

I have implemented 3 processes of KMeans.

1. Sequential KMeans
2. Map Reduce version of KMeans
3. Spark KMeans

I have implement all the processes on **kddcup.data_10_percent** dataset.

Sequential KMeans

K = 2

- Time took to process data 30716 ms
- Running my KMeans algorithm Time took 7764 ms

0	normal : 1185	probe : 1272	dos : 280926		
1	r2l : 1126	normal : 96091	probe : 2835	dos : 110532	u2r : 52

- Running Weka KMeans Algorithm Time took 39589 ms

0	r2l : 74	normal : 958	u2r : 6	probe : 2702	dos : 107298
1	r2l : 1052	normal : 96320	probe : 1405	dos : 284160	u2r : 46

K = 5

- Time took to process data 31850 ms
- Running my KMeans algorithm Time took 52922 ms

0	r2l : 8	normal : 16	probe : 235	dos : 86758	
1	r2l : 1032	normal : 81360	dos : 2197	probe : 4	u2r : 46
2	r2l : 49	normal : 5351	dos : 20464	probe : 2396	
3	normal : 67	dos : 280699			
4	r2l : 37	normal : 10481	u2r : 6	probe : 1472	dos : 1338

- Running Weka KMeans Algorithm Time took 66487 ms

0	r2l : 8	normal : 16	probe : 303	dos : 86759	
1	r2l : 1031	normal : 80850	dos : 2196	probe : 3	u2r : 45
2	normal : 79	dos : 280719			
3	r2l : 49	normal : 5329	dos : 20464	probe : 2323	
4	r2l : 38	normal : 11004	u2r : 7	probe : 1478	dos : 1320

HADOOP

K = 5

- Time took to process data 35390 ms
- Time took to make clusters 181358 ms

3	r2l : 1	normal : 19252	probe : 311	u2r : 17	dos : 89239
2	normal : 892	u2r : 2	probe : 117	dos : 23788	
1	r2l : 1024	normal : 33415	probe : 2747	dos : 244553	u2r : 28
0	r2l : 101	normal : 43703	probe : 932	dos : 33878	u2r : 5

- Time took to process output 6594 ms

k=2

- Time took to process data 27371 ms
- Time took to train 184488 ms

1	r2l : 1024	normal : 36550	u2r : 43	probe : 2298	dos : 343848
0	r2l : 102	normal : 60720	probe : 1809	dos : 47610	u2r : 9

- Time took to process output 4896 ms

SPARK

K = 5

Process 32

- Time Took to process data 27434 ms
- Time Took to create RDD 16316 ms
- Time Took to train data 182640 ms

0	r2l : 8	normal : 16	probe : 298	dos : 86778	
1	normal : 111	probe : 12	dos : 280769		
2	normal : 15	dos : 20465	probe : 2299		
3	r2l : 1067	normal : 91751	probe : 1460	dos : 3436	u2r : 52
4	r2l : 51	normal : 5385	dos : 10	probe : 38	

- Time Took to process output 30665 ms

Process 16

- Time Took to process data 26359 ms
- Time Took to create RDD 16843 ms
- Time Took to train data 427878 ms

0	r2l : 49	normal : 5325	dos : 20464	probe : 2325	
1	normal : 31	dos : 280649			
2	r2l : 1038	normal : 69947	dos : 2196	probe : 4	u2r : 46
3	r2l : 8	normal : 16	probe : 303	dos : 86763	
4	r2l : 31	normal : 21959	u2r : 6	probe : 1475	dos : 1386

- Time Took to process output 29883 ms

Process 8

- Time Took to process data 25069 ms
- Time Took to create RDD 16905 ms
- Time Took to train data 517758 ms

0	r2l : 8	normal : 16	probe : 298	dos : 86778	
1	normal : 111	probe : 12	dos : 280769		
2	r2l : 1067	normal : 91751	probe : 1460	dos : 3436	u2r : 52
3	normal : 15	dos : 20465	probe : 2299		
4	r2l : 51	normal : 5385	dos : 10	probe : 38	

- Time Took to process output 35344 ms

Process 4

- Fails

Process 2

- Fails

Process 1

- Fails

K = 2

Process 32

- Time Took to process data 10274 ms
- Time Took to create RDD 16272 ms
- Time Took to train data 178117 ms

0	r2l : 1126	normal : 96082	probe : 2835	dos : 110532	u2r : 52
1	normal : 1196	probe : 1272	dos : 280926		

- Time Took to process output 13266 ms

Process 16

- Time Took to process data 10310 ms
- Time Took to create RDD 17001 ms
- Time Took to train data 208231 ms

0	r2l : 71	normal : 982	u2r : 6	probe : 2702	dos : 107299
1	r2l : 1055	normal : 96296	probe : 1405	dos : 284159	u2r : 46

- Time Took to process output 13481 ms

Process 8

- Time Took to process data 10957 ms
- Time Took to create RDD 16136 ms
- Time Took to train data 236001 ms

0	r2l : 71	normal : 982	u2r : 6	probe : 2702	dos : 107299
1	r2l : 1055	normal : 96296	probe : 1405	dos : 284159	u2r : 46

- Time Took to process output 13997 ms

Process 4

- Fails

Process 2

- Fails

Process 1

- Fails

ANALYSIS

- Sequential version of KMeans is faster than any of the implementations.
- Map Reduce version is faster than Spark implementation.
- If we see the clusters in any version, those are not fully distinguishable. If we can reduce the input features to important ones then the cluster result might be more distinguishable.
- I couldn’t tune the parameters very well to get the best and fastest output on Spark.
- My program doesn’t work when I reduce the executor cores below 8 (ex. 4,2,1).
- In each program, initially I loaded all the data in memory. My sequential KMeans algorithm can load all the data of kddcup.data dataset. But, I couldn’t load the data for Map reduce or spark version. I think it is better to process data line by line rather loading all data to memory. But, for

KMeans I think we can process data line by line as we have to normalize the data before we can use it.

CONCLUSION

In Map Reduce version we are using File to save the intermediate result in each iteration step. And in spark we are using memory. So, spark version of KMeans should be faster. I don’t get why the running time of Spark is just as the same as Map Reduce version. The reason might be I couldn’t tune the parameters to get the best output. But, I think if the workload is not very high enough we should not use distributed system. We should try to solve a problem is sequential or threaded version at first without considering distributed system.