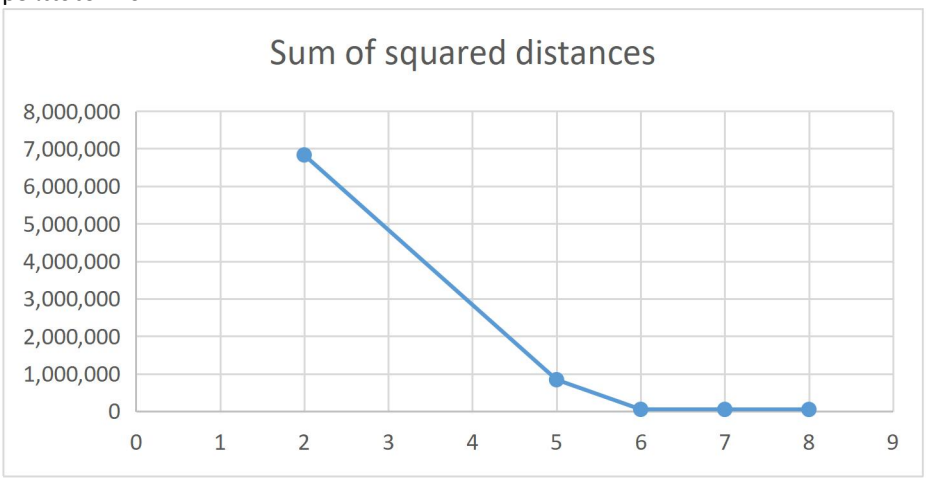
Στυλιανή Καλαϊτζάκη dai19081
Χρήστος Κεχαγιάς dai18213

Πρόγραμμα: kMeans
Γλώσσα: python

Εντολή εκτέλεσης προγράμματος :
~/spark/bin/spark-submit --master spark://master:7077 kMeans.py
Το πρόγραμμα δέχεται προαιρετική παράμετρο 0 ή 1 (για random ή kmeans|| αρχικοποίηση αντίστοιχα). Το default είναι 1 (kmeans||).

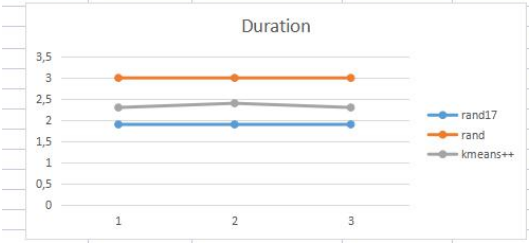
Για τον υπολογισμό των αριθμών των συστάδων έγιναν δοκιμές με διάφορα k.
k-means|| αρχικοποίηση
k=2:
a. Sum of squared distances : 6830876.747450786
b. Silhouette Score: 0.8251592918850534
c. Cluster sizes: [3000000, 3000000]
k=5:
a. Sum of squared distances : 840784.783834335
b. Silhouette Score: 0.8953972803828181
c. Cluster sizes: [1000000, 1000000, 2000000, 1000000, 1000000]
k=6:
a. Sum of squared distances : 50353.377577733045
b. Silhouette Score: 0.9897982912876943
c. Cluster sizes: [1000000, 1000000, 1000000, 1000000, 1000000, 1000000]
k=7:
a. Sum of squared distances : 49000.22335941545
b. Silhouette Score: 0.8664627526292571
c. Cluster sizes: [499660, 1000000, 1000000, 1000000, 1000000, 1000000, 500340]
k=8:
a. Sum of squared distances : 47643.41178083219
b. Silhouette Score: 0.7432622451862656
c. Cluster sizes: [499698, 1000000, 1000000, 1000000, 500154, 1000000, 500302, 499846]

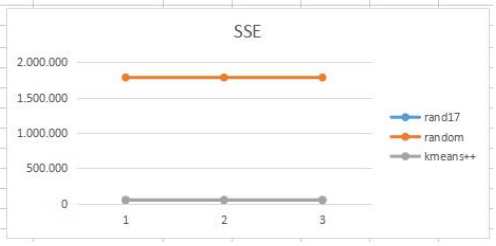
Συγκίρνοντας τις παραπάνω τιμές βλέπουμε πως ο συντελεστής περιγράμματος είναι πιο κοντά στο 1 όταν k=6. Αυτό σημαίνει ότι οι συστάδες είναι συμπαγής και μακριά η μία από την άλλη, και άρα είναι καλά διαχωρισμένες. Το συνολικό άθροισμα των τετραγώνων των αποστάσεων πέφτει απότομα στα k=5,6 και μετά συνεχίζει να μειώνεται με χαμηλό ρυθμό. Από τη μέθοδο Elbow για βέλτιστο k προκύπτει ότι το k είναι 5 ή 6. Συνδυάζοντας αυτές τις δύο τιμές και το μέγεθος των συστάδων που δείχνει πιο ομοιόμορφη κατανομή στην περίπτωση του k=6, συμπεραίνουμε ότι βέλτιστο k=6.



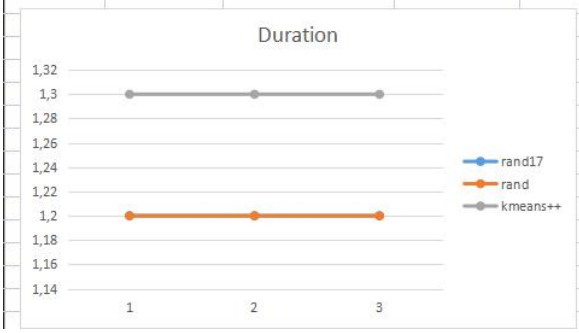
Με τυχαία αρχικοποίηση στον kMeans τα αποτελέσματα είναι ίδια όταν το πρόγραμμα τρέχει σε δύο slaves. Όταν τρέχει σε έναν slave ο συντελεστής περιγράμματος στο k=6 δεν πλησίαζε στο 1 και το άθροισμα των τετραγώνων των αποστάσεων ήταν διαφορετικό. Αυτό υποθέσαμε ότι συμβαίνει λόγω τη φύσης του αλγορίθμου με αυτή την αρχικοποίηση και τα επιλεγμένα centroids δεν είναι καλά τοποθετημένα στο σύνολο των δεδομένων, με αποτέλεσμα να μην αρκεί ένας slave. Μετά από πειραματισμούς με την τιμή του seed καταλήξαμε στο ότι θέτοντας seed=17 τα αποτελέσματα είναι ίδια με παραπάνω.

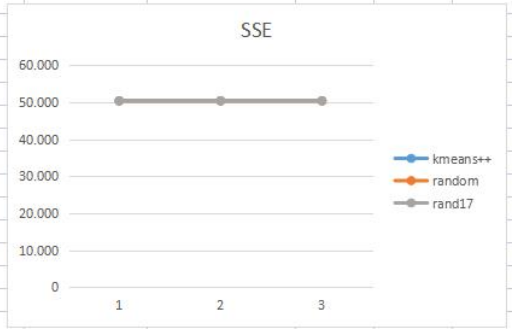
1 slave						
	Αρχικοποίηση με Kmeans||		Τυχαία αρχικοποίηση		Τυχαία αρχικοποίηση με seed=17	
Execution	Duration	SSE	Duration	SSE	Duration min	SSE
1	2,4 min	50.353	3,0 min	1.782.761	1,9 min	50.353
2	2,3 min	50.353	3,0 min	1.782.761	1,9 min	50.353
3	2,3 min	50.353	3,0 min	1.782.761	1,9 min	50.353
Min	2,3	50.353	3	1.782.761	1,9	50.353
Max	2,4	50.353	3	1.782.761	1,9	50.353
Avarage	2,3	50.353	3	1.782.761	1,9	50.353





2 slaves						
	Αρχικοποίηση με Kmeans||		Τυχαία αρχικοποίηση		Τυχαία αρχικοποίηση με seed=17	
Execution	Duration	SSE	Duration	SSE	Duration	SSE
1	1.3 min	50.353	1.2 min	50.353	1.2 min	50.353
2	1.3 min	50.353	1.2 min	50.353	1.2 min	50.353
3	1.3 min	50.353	1.2 min	50.353	1.2 min	50.353
Min	1,3	50.353	1,2	50.353	1,2	50.353
Max	1,3	50.353	1,2	50.353	1,2	50.353
Avarage	1,3	50.353	1,2	50.353	1,2	50.353





Σχολιασμός των αποτελεσμάτων

1 slave

Αρχικοποίηση με Kmeans:

Παρατηρούμε ότι και στις 3 εκτελέσεις η διάρκεια τους είναι μεταξύ 2.3 και 2.4 min ενώ το άθροισμα τετραγωνικού σφάλματος σε όλες τις εκτελέσεις είναι 50353,377577733.

Τυχαία αρχικοποίηση:

Και στις 3 εκτελέσεις ο χρόνος είναι 3.0 min ενώ το άθροισμα του τετραγωνικού σφάλματος είναι 1782761,37643096.

Τυχαία αρχικοποίηση με seed=17:

Και στις 3 εκτελέσεις ο χρόνος είναι 1.9 min ενώ το άθροισμα του τετραγωνικού σφάλματος είναι 50353,377577733.

Μεταξύ της τυχαίας αρχικοποίησης, της τυχαίας αρχικοποίησης με seed=17 και της αρχικοποίησης με Kmeans:

Παρατηρούμε μια αρκετή μεγάλη διαφορά τόσο στον χρόνο εκτέλεσης όσο και στον υπολογισμό του αθροίσματος του τετραγωνικού σφάλματος. Συγκεριμένα ο χρόνος της τυχαίας αρχικοποίησης είναι

μεγαλύτετρος από την αρχικοποίηση με Kmeans κι αυτό υποθέτουμε πως συμβαίνει εξαιτίας του ότι καθυστερεί στο να βρεί το σωστό αριθμό συστάδων σε αντίθεση με την αρχικοποίηση με Kmeans όπου εμείς εξαρχής δίνουμε το k=6. Όσον αφορά την τυχαία αρχικοποίηση με seed 17 παρατηρούμε μια τρομερή μείωση στον χρόνο εκτέλεσης του αλγορίθμου.

2 slaves

Αρχικοποίηση με Kmeans:

Παρατηρούμε ότι και στις 3 εκτελέσεις η διάρκεια τους είναι μεταξύ 1.3 min ενώ το άθροισμα τετραγωνικού σφάλματος σε όλες τις εκτελέσεις είναι ίδιο με αυτού του ενός slave(αρχικοποίησης με Kmeans), δηλαδή 50353,377577733.

Τυχαία αρχικοποίηση:

Και στις 3 εκτελέσεις ο χρόνος είναι 1.2 min ενώ το άθροισμα του τετραγωνικού σφάλματος είναι 50353,377577733

Τυχαία αρχικοποίηση με seed=17:

Και στις 3 εκτελέσεις ο χρόνος είναι 1.2 min ενώ το άθροισμα του τετραγωνικού σφάλματος είναι 50353,377577733.

Μεταξύ της τυχαίας αρχικοποίησης, της τυχαίας αρχικοποίησης με seed=17 και της αρχικοποίησης με Kmeans:

Στην συγκεριμένες υλοποιήσεις δεν υπάρχει διαφορά τόσο στην διάρκεια εκτέλεσης όσο και στο άθροισμα του τετραγωνικού σφάλματος.

Μεταξύ των υλοποιήσεων με 1 slave και 2 slaves:

Αυτό που πατατηρείται και είναι λογικό είναι ότι η διάρκεια εκτέλεσης έχει δραματική διαφορά μιας και όπως είναι φυσικό η χρήση των 2 slaves παραπέμπουν στον παραλληλισμό του προγράμματος με συνέπεια την μείωση του χρόνου εκτέλεσης. Πιο συγκεκριμένα η εκτέλεση με 1 slave είναι 2,4-3,0min ενώ με χρήση δύο slaves ο χρόνος αυτός μειώνεται στο 1,2-1,3min.