**1η εργασία - Αναζήτηση και συσταδοποίηση διανυσμάτων στη C/C++**

**Χριστίνα Ιακωβίδου sdi19-**

**Ροδάνθη Αλεξίου sdi1900007**

**Ερώτημα Α**

**Εισαγωγή**

Ο κώδικας σχετίζεται με τον Αναδρομικό Συναρτησιακό Κατακερματισμό (LSH) για αναζήτηση κοντινότερων γειτόνων, με την χρήση Ευκλείδιας απόστασης.

**Συναρτήσεις και Κλάσεις**

**Κλάση LSHImplementation**

Μέλη Δεδομένων

* **Euclidian\_Amplified\_Functions**: Ένας πίνακας που αποθηκεύει ανακλημένες συναρτήσεις για τον Ευκλείδιο LSH.
* **Euclidian\_Hash\_Tables**: Πίνακες κατακερματισμού για την αποθήκευση σημείων δεδομένων.
* **v**: Ένας διάνυσμα διανυσμάτων που χρησιμοποιείται για υπολογισμούς συναρτήσεων κατακερματισμού.
* **t**: Ένα διάνυσμα σταθερών που χρησιμοποιείται στους υπολογισμούς συναρτήσεων κατακερματισμού.
* **w**, **k**, **L**: Παράμετροι για το LSH.

Constructors

* **LSHImplementation(int L, int k, int w)**: Κατασκευαστής για την αρχικοποίηση των παραμέτρων LSH.

Μέθοδοι Μελών

* **getEuclidianHashTables()**: Επιστρέφει μια σταθερή αναφορά στους πίνακες κατακερματισμού του Ευκλείδιου.
* **getEuclidianAmplifiedFunctions()**: Επιστρέφει μια αναφορά στις ανακλημένες συναρτήσεις.
* **getAmplifiedFunction(int j, int k)**: Επιστρέφει μια ανακλημένη συνάρτηση για δεδομένους δείκτες.
* **getL()**, **getK()**, **getW()**: Συναρτήσεις προσπέλασης για τις παραμέτρους L, k και w.
* **normal\_distribution\_generator()**: Δημιουργεί τυχαίους αριθμούς από κανονική κατανομή.
* **modulo(long long int a, long long int b)**: Υπολογίζει τη λειτουργία modulo.
* **euclidian\_hash\_query(...)**: Υπολογίζει τη θέση στον πίνακα κατακερματισμού για ένα διάνυσμα ερωτήματος.
* **euclidian\_hash\_centroid(...)**: Υπολογίζει τη θέση στον πίνακα κατακερματισμού για ένα κεντροειδές διάνυσμα.
* **calcute\_euclidian\_distance(...)**: Υπολογίζει την Ευκλείδια απόσταση μεταξύ δύο διανυσμάτων.
* **Euclidian\_put\_hash(...)**: Εισάγει ένα διάνυσμα σε όλους τους πίνακες κατακερματισμού.
* **insertCentroidIntoLSH(...)**: Εισάγει ένα κεντροειδές διάνυσμα στους πίνακες κατακερματισμού.
* **read\_mnist\_images(...)**: Διαβάζει δεδομένα εικόνας MNIST.
* **getImageLine(...)**: Επιστρέφει μια γραμμή εικόνας δεδομένου ενός δείκτη.
* **initializeVectors(...)**: Αρχικοποιεί διανύσματα και συναρτήσεις κατακερματισμού.
* **initializeAmplifiedFunctions(...)**: Αρχικοποιεί ανακλημένες συναρτήσεις κατακερματισμού.
* **initializeT(...)**: Αρχικοποιεί τις σταθερές **t** για τις συναρτήσεις κατακερματισμού.
* **cleanupHashTables(...)**: Καθαρίζει τους πίνακες κατακερματισμού αφαιρώντας κενές θέσεις.
* **initializeLSH(...)**: Αρχικοποιεί τις παραμέτρους LSH, τα διανύσματα και τους πίνακες κατακερματισμού.
* **FindNearestNeighbor(...)**: Βρίσκει τον κοντινότερο γείτονα για ένα σημείο ερώτησης.
* **FindNearestNeighborCentroid(...)**: Βρίσκει τον κοντινότερο γείτονα για ένα κεντροειδές.
* **Euclidian\_Full\_Search\_NN(...)**: Εκτελεί πλήρη αναζήτηση για τους κοντινότερους γείτονες.
* **Euclidian\_Full\_Search\_Range(...)**: Εκτελεί πλήρη αναζήτηση για τα σημεία δεδομένων εντός δοθείσας ακτίνας.

**Χρήση**

Στην mnist.cpp υπάρχει η υλοποίηση για την σωστή ανάγνωση των arguments και μια ενδεικτική εκτέλεση είναι:

**./mnist -d train-images.idx3-ubyte -q t10k-images.idx3-ubyte -ο out.txt**

**Ερώτημα Β**

**Εισαγωγή**

Υλοποίηση αλγορίθμων για τη συσταδοποίηση διανυσμάτων στον χώρο R.

**Συναρτήσεις**

* **std::vector<unsigned char\*> kmeans\_plusplus\_init(const fileData& data, int K)** - Συνάρτηση για την αρχικοποίηση του K-Means++.
* **double euclideanDistance(const unsigned char\* point1, const unsigned char\* point2, int size)** - Υπολογισμός της Ευκλείδιας απόστασης μεταξύ δύο σημείων δεδομένων.
* **std::vector<int> assignToNearestCentroids(const fileData& data, const std::vector<unsigned char\*>& centroids)** - Ανάθεση σημείων δεδομένων στο πλησιέστερο κέντρο με τον αλγόριθμο Lloyd
* **int assignToNearestCentroidLSH(LSHImplementation& lsh, fileData& data, int L, int k, std::vector<double>& t, std::vector<unsigned char\*>& centroids, int queryImage)** - Ανάθεση ενός σημείου ερωτήματος στο πλησιέστερο κέντρο χρησιμοποιώντας το LSH
* **void updateCentroidsMacQueen(const fileData& data, const std::vector<int>& assignments, int K, std::vector<unsigned char\*>& new\_centroids)** - Ενημέρωση των κέντρων χρησιμοποιώντας τη μέθοδο του MacQueen.

**Υλοποίηση**

Στο στάδιο που ενημερώνουμε τα κέντρα μετά το βήμα ανάθεσης που έχουμε επιλέξει, τα ενημερωμένα κέντρα εμφανίζονται όλα 0 και δεν καταφέραμε να τυπώσουμε τα σωστά σημεία (από το debugging εντοπίσαμε ένα θέμα στο πως θα ορίσουμε τον τύπο των κέντρων και τις πράξεις που θα κάνουμε σε αυτά μέσω του McQueen αλγορίθμου)