ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΠΡΟΧΩΡΗΜΕΝΑ ΘΕΜΑΤΑ ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ

Παπαδόπουλλος Μιχάλης

el14702@central.ntua.gr

031-14702

Εισαγωγή:

Δημιουργούμε δύο (2) εικονικές μηχανές στην πλατφόρμα **~okeanos** με χαρακτηριστικά:

- 2 CPU
- 2048MB RAM
- 30GB HDD
- Ubuntu Server LTS

Η γλώσσα υλοποίησης είναι η Python

Machine Learning - Ομαδοποίηση δεδομένων με εκτέλεση του K-means αλγόριθμου

Δεδομένα: Τα δεδομένα που θα χρησιμοποιήσετε είναι πραγματικά και αφορούν διαδρομές taxi στην Νέα Υόρκη. Οι δοθείσες διαδρομές των taxi έγιναν από τον Ιανουάριο εώς το Ιούνιο του 2015 και υπάρχουν διαθέσιμες online εδώ.
Λόγω των περιορισμένων πόρων που έχουμε στη διάθεσή μας, θα επεξεργαστούμε μόνο ένα υποσύνολο μεγέθους ~2 GB. Τα δεδομένα αυτά περιέχουν 13 εκατομμύρια διαδρομές, που πραγματοποιήθηκαν το Μάρτιο του 2015 και υπάρχουν διαθέσιμα εδώ.

Στο αρχείο αυτό, περιλαμβάνονται δύο αρχεία κειμένου (.csv) που ονομάζονται: **yellow_tripdata_1m.csv** και **yellow_tripvendors_1m.csv**. Το πρώτο αρχείο περιλαμβάνει όλη την απαραίτητη πληροφορία για μια διαδρομή.

Το αρχείο των **TripData** έχει την εξής μορφή:

```
369367789289,2015-03-27 18:29:39,2015-03-27
19:08:28,-73.975051879882813,40.760562896728516,-73.84790039062
5,40.7326850
89111328,34.8
```

- Το **πρώτο** πεδίο αποτελεί το **μοναδικό id μιας διαδρομής**.
- Το **δεύτερο** και **τρίτο** πεδίο την **ημερομηνία και ώρα έναρξης και λήξης** της διαδρομής.
- Το **τέταρτο** και **πέμπτο** πεδίο το **γεωγραφικό μήκος και πλάτος του σημείου επιβίβασης**.
- Το **έκτο** και **έβδομο** πεδίο περιλαμβάνουν **το γεωγραφικό μήκος και πλάτος του σημείου αποβίβασης**.
- Το **όγδοο** πεδίο δείχνει το **συνολικό κόστος της διαδρομής**.

Το αρχείο **TripVendors** περιέχει πληροφορία για τις εταιρείες taxi και έχει την εξής μορφή:

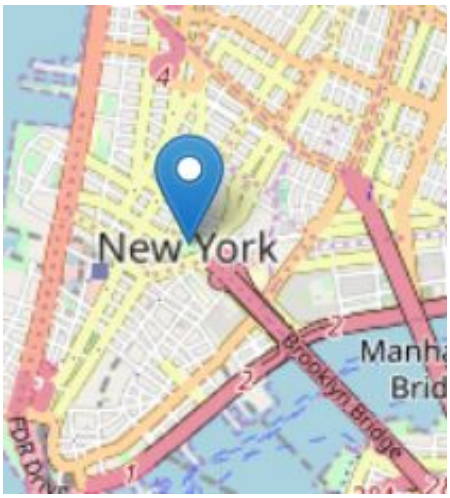
```
369367789289,1
```

- Το **πρώτο** πεδίο αποτελεί το **μοναδικό id μιας διαδρομής**
- Το **δεύτερο** πεδίο το **μοναδικό αναγνωριστικό μιας εταιρείας taxi** (vendor).

Ζητούμενα: Χρησιμοποιώντας τα δεδομένα του αρχείου **TripData**, θέλουμε να βρούμε τις κεντρικές συντεταγμένες των top 5 περιοχών επιβίβασης πελατών. Επομένως θα χρησιμοποιήσουμε μόνο το πρώτο αρχείο.

Περιγραφή:
Διαβάζουμε το αρχείο και κρατάμε μόνο τις απαραίτητες για την άσκηση πληροφορίες. Δηλαδή το **τέταρτο** και **πέμπτο** πεδίο κάθε γραμμής του αρχείου που αντιστοιχεί στο **γεωγραφικό μήκος** και **πλάτος του σημείου επιβίβασης**.
Κατα την επεξεργασία, φιλτράρουμε ταυτόχρονα όσα δεδομένα είναι ελλιπή (μια ίσως καλύτερη λύση είναι να αγνοούμε όσες συντεταγμένες είναι εκτός των ορίων ενδιαφέροντος).
Αρχικά επιλέγουμε τα πέντε (5) πρώτα σημεία του dataset μας ώς κέντρα και εκτελούμε τον αλγόριθμό για τρεις (3) επαναλήψεις προτού εξάγουμε τα αποτελέσματα.

#	Longitude	Latitude
1	-74.01403589	40.71059112
2	-73.94046141	40.76362689
3	-73.99570466	40.72206538
4	-74.00370484	40.7387053
5	-73.98400183	40.75275303



http://master:50070/webhdfs/v1/output/173925/part-00000?op=OPEN

Χρησιμοποιούμε ώς μετρική την απόσταση **Haversine** η οποία λαμβάνει υπόψη ότι η γη είναι σφαιρική και υπολογίζει την απόσταση μεταξύ δύο (2) συντεταγμένων της.

Στην συνέχεια παραθέτω ψευδοκώδικα για την υλοποίηση της άσκησης και συνοπτική περιγραφή για την κάθε λειτουργία.

Ψευδοκώδικας:

Όπως έχω περιγράψει και πιο πάνω, κάνουμε κάποιου είδους προ-επεξεργασία στα δεδομένα ώστε να εξασφαλίσουμε πως δεν περιέχονται λανθασμένες τιμές στο working dataset.

```
// preprocessing & filtering
def map(key, value):
    x, y = line.split(' ')[3:5]
    if x != 0.0 and y != 0.0:
        emit(x, y)
```

Για κάθε σημείο βρίσκουμε το index του κέντρου που απέχει την μικρότερη απόσταση. Επιστρέφεται το index του κέντρου που απέχει την μικρότερη απόσταση μαζί με το σημείο και τον αριθμό ένα (1) ώστε να ομαδοποιηθούν μετά από την συνάρτηση reduce και να υπολογιστεί ο μέσος όρος. Η συνάρτηση αυτή βοηθά στην σύγκλιση των κεντρικών σημείων.

```
// min-index
def map(key, value):
    ix = 0
    mdist = float("+inf")
    for i in range(NO_CENTROIDS):
        dist = haversine(value, centroid[i])
        if mdist > dist:
            mdist = dist
    emit(ix, (value, 1))
```

Η reduce αποτελεί την τελευταία συνάρτηση στη διαδικασία υπολογισμού των κεντρικών σημείων. Ομαδοποιεί τα σημεία με βάση το index του κέντρου που απέχουν τη μικρότερη απόσταση. Στη συνέχεια προστίθενται κατα μέρη οι τιμές των συντεταγμένων και υπολογίζεται ο συνολικός

αριθμός των σημείων με την πρόσθεση των άσσων που προέκυψαν από την προηγούμενη διαδικασία. Τέλος επιστρέφονται οι καινούριες συντεταγμένες που προκύπτουν απο τη διαίρεση των τιμών x/sum και y/sum - δηλαδή ο μέσος όρος των σημείων αυτών.

```
// mean values
def reduce(key, values):
  x = y = sum = 0
  for value in values:
      x += value[0][0]
      y += value[0][1]
      sum += value[1]
  emit(key, (x / sum, y / sum))
```