Θέμα 2 στην Τεχνητή Νοημοσύνη

Λεωνίδας Αβδελάς | ΑΜ: 03113182

Εισαγωγή

Στο θέμα αυτό επεκτείνουμε το πρόγραμμα του πρώτου θέματος, προσθέτοντας περισσότερες πληροφορίες στον τρόπο που θα γίνεται η επιλογή των ταξί. Πιο συγκεκριμένα, με χρήση της prolog, δημιουργούμε κάποιους κανόνες για την συμβατότητα μεταξύ πελάτη και ταξί, καθώς και ανασχεδιάζουμε τον τρόπο που υπολογίζεται η ευριστική της διαδρομής του ταξί.

Δεδομένα

Τα δεδομένα που είχαμε ήταν ένα υπερσύνολο των δεδομένων της προηγούμενης φοράς. Συγκεκριμένα είχαμε πληροφορίες σχετικά με την κίνηση κάθε δρόμου στο αρχείο traffic.csv, πληροφορίες σχετικά με τον κάθε δρόμο (το είδος του, λωρίδες, αλλά και πολλά άλλα) στο αρχείο lines.csv. Επιπλέον, το αρχείο taxis.csv περιείχε πληροφορίες, όπως χωρητικήτητα, γλώσσα κλπ. Παρομοίως επεκτάθηκε και το αρχείο client.csv.

Για την διαχείριση των δεδομένων, αποφασίστηκε να αποθηκευτούν όλα στην Java αρχικά.

Συγκεκριμένα επεκτείναμε τις κλάσεις customer, taxi, node, ώστε να περιέχονται τα καινούρια δεδομένα και δημιουργήσαμε τις κλάσεις line, traffic και TimeAndAmound η οποία συνδέεται με την traffic για να παρέχει με εύκολο τρόπο τις αντιστοιχίες ώρες-κίνηση.

Πέρα από την αποθήκευση, υπήρξε και μια μικρή μορφοποίηση των δεδομένων σε κατάλληλους τύπους (int, boolean, double etc), με σκοπό την ευκολότερη χρήση των δεδομένων και μέσα στην java, αλλα και στην prolog.

Λογική

Όπως προείπαμε, η prolog χρησιμοποιήθηκε σε δύο σημεία. Στην απόφαση σχετικά με το ο πελάτης και το ταξί “ταιριάζουν”, λαμβάνοντας υπόψιν την γλώσσα, την χωρητικότητα και ο πελάτης χρειάζεται ταξί για να διανύσει μεγάλες αποστάσεις (λάβαμε υπόψιν την συνολική απόσταση πελάτη – ταξί, δηλαδή την απόσταση από το σημείο που είναι το ταξί μέχρι τον πελάτη και μετά απο τον πελάτη μέχρι τον προορισμό του).

Το δεύτερο μέρος που χρησιμοποιήθηκε η prolog ήταν στην εύρεση της ευριστικής συνάρτησης κάθε κόμβου, εκτός από τον πρώτο. Τα στοιχεία που χρησιμοποιήθηκαν εδώ ήταν: Αν ο δρόμος ήταν της κατηγορίας highway ή όχι. Αν δεν ήταν η prolog επέστρεφε σαν ευριστική false, το οποίο το ερμηνεύαμε σαν την μέγιστη τιμή ενός double αριθμού. Δεν έγινε κάποιος έλεγχος για τις υποκατηγορίες του highway. Έπειτα γινόταν έλεγχος για την κατεύθυνση του δρόμου. Σε αυτό το σημείο σαν έλεγχο για την κατεύθυνση χρησιμοποιήσαμε την τιμή που έχουν στον άξονα X τα σημεία που ήταν ο κόμβος που βρισκόταν το ταξί και ο επόμενος κόμβος στον οποίο ήθελε να ταξιδέψει. Γνωρίζοντας ότι τα δεδομένα των κόμβων μας δίνονται ταξινομημένα από τον άξονα X και ότι η διεύθυνση είναι σχετική με την σειρά που μας δίνονται τα δεδομένα (-1 αν η διεύθυνση είναι αντίθετη από την σειρά που μας δίνονται τα δεδομένα, 1 το ανάποδο), αναπτύξαμε τον τρόπο να βρίσκουμε την διεύθυνση του δρόμου. Τέλος, σαν τιμή της ευριστικής χρησιμοποιήθηκε πάλι η Ευκλείδια απόσταση. Προσπαθήσαμε να προσθέσουμε και την κίνηση μέσα στην ευριστική,

χρησιμοποιώντας έναν πολλαπλασιστή. Συγκεκριμένα αν ο δρόμος που θα ταξιδεύαμε είχε μεσαία ή υψηλή κίνηση πολλαπλασιάζαμε την ευριστική με 1.5 ή 2 αντίστοιχα.

Κάποιες γνώσεις για τον κόσμο δεν τις αξιοποιήσαμε καθόλου, πχ λωρίδες, κλίση του δρόμου κλπ.

Τα γεγονότα αυτά μεταφέρονταν από το πρόγραμμα της Java σε αυτό της Prolog μέσω assertions που κάναμε δημιουργώντας γεγονότα, όπως belongsIn(nodeId, lineId), language(taxiId, language), traffic(lineID, startingHour, endingHour, Value), oneway(lineId, booleanOneway, booleanOnewayDirection) (booleanDirection false = wrong way), highway(lineId, boolean isHighway).

Αρα δεν υπάρχουν αρχεία με γεγονότα, αφού αυτά παράγονται κάθε φορά που τρέχουμε το πρόγραμμα.

Έξοδος

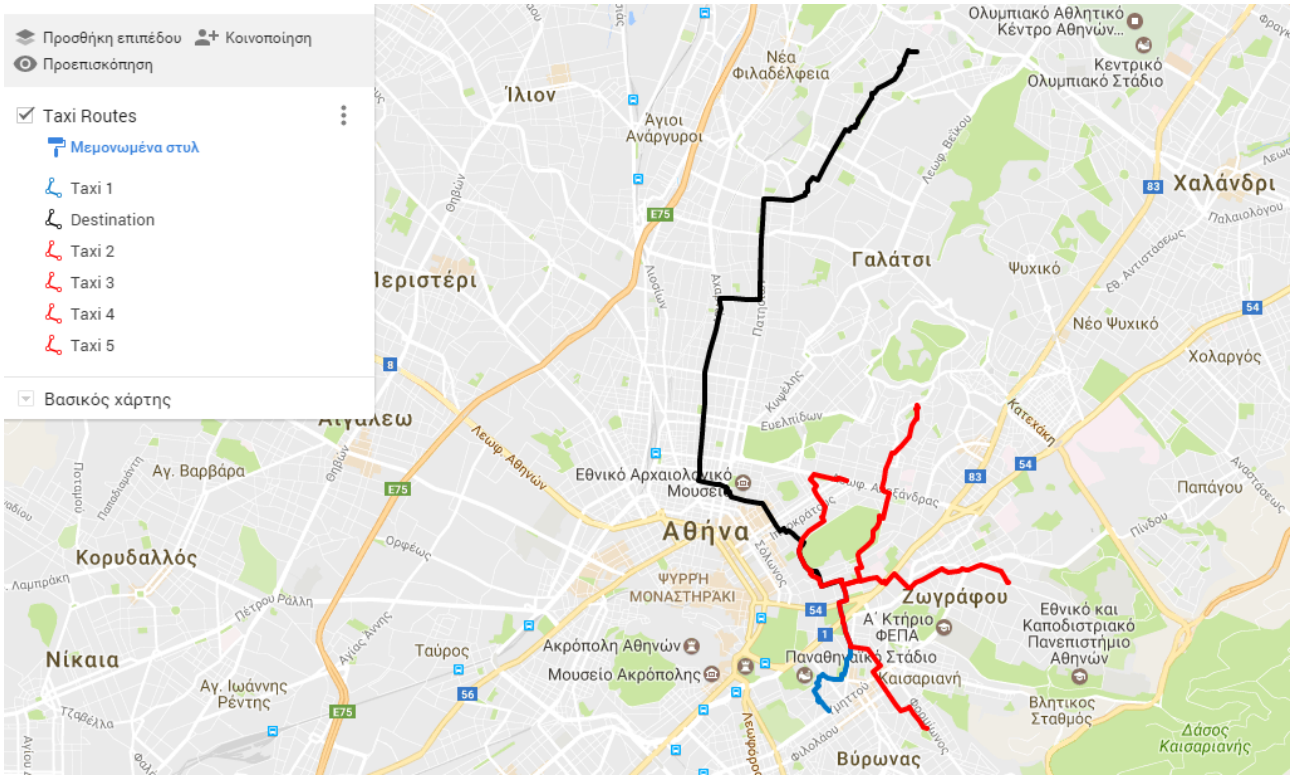
η έξοδος μας αποτελείται αυτή την φορά από 2 KML αρχεία, καθώς και δυο απάντησεις στο command line. Το πρώτο αρχείο και η πρώτη απάντηση μας ενμερώνουν για το βέλτιστο ταξί βάσει απόστασης από τον πελάτη και μας δίνουν μια κατάταξη 5 ταξί, δίνοντας μας και την απόσταση κάθε ταξί. Στο δεύτερο αρχείο, αλλά και απάντηση, μας δίνονται τα ίδια ταξί, αλλά αυτή τη φορά καταταγμένα βάσει του rating του καθενός. Και στις δύο περιπτώσεις, η διαδρομή του επκρατέστερου ταξί τυπώνεται με μπλε, ενώ οι υπόλοιπες με κόκκινο. Τέλος η διαδρομή από τον πελάτη μέχρι τον προορισμό του τυπώνεται με μαύρο.

Για τα δεδομένα που μας δίνονται έχουμε τα παρακάτω αποτελέσματα:

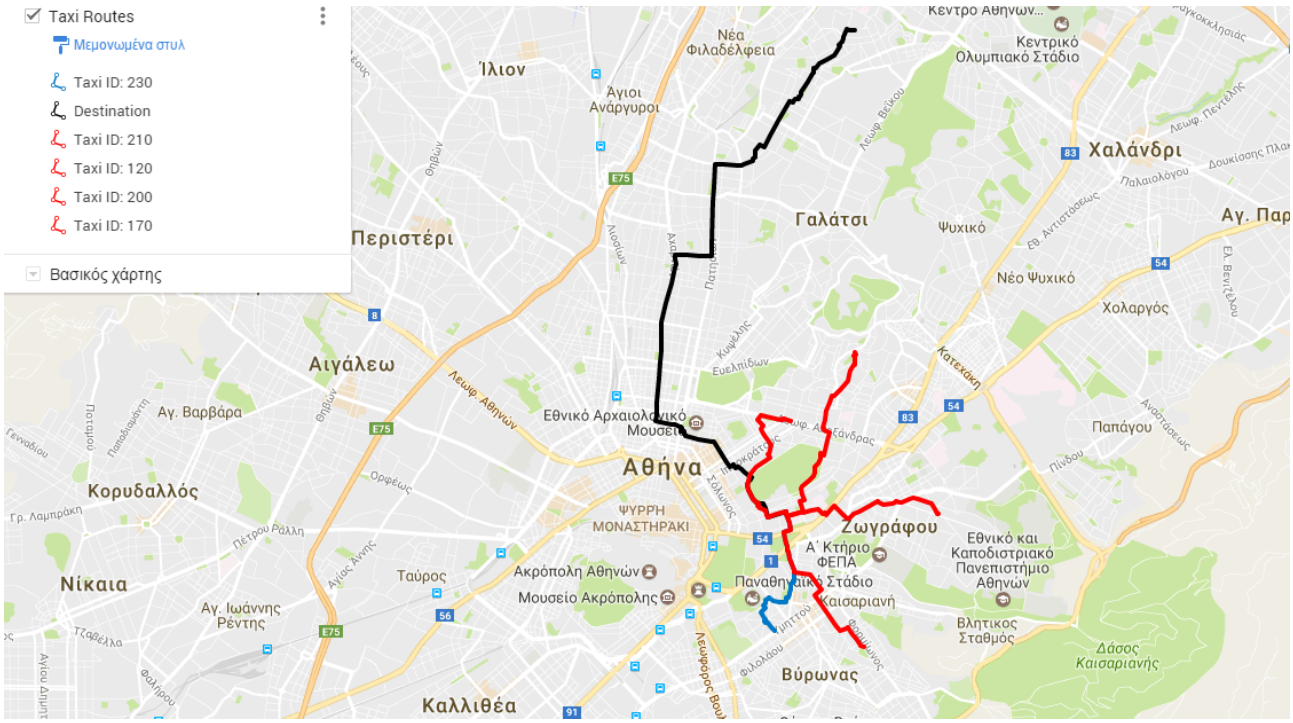
```
The top five best results depending on distance we found were:
(visual representation in map1.kml)
1:    Id: 230 Distance: 0.02202831082311162
2:    Id: 120 Distance: 0.02486894781253231
3:    Id: 210 Distance: 0.0249914084432662
4:    Id: 200 Distance: 0.02584569442605691
5:    Id: 170 Distance: 0.027737490460401564

Top five taxi candidates rearanged by rating:
(visual representation in map2.kml)
1 :   Id: 230 Rating: 9.8
2 :   Id: 210 Rating: 9.2
3 :   Id: 120 Rating: 8.0
4 :   Id: 200 Rating: 8.0
5 :   Id: 170 Rating: 7.1
```

Map1.kml



Map2.kml



Σε αυτή την περίπτωση τυχαίνει να είναι ίδια.

Για την δική μας περίπτωση επιλέξαμε να αλλάξουμε μόνο τις πληροφορίες του πελάτη, ενώ αφήσαμε όλα τα άλλα δεδομένα ίδια.

Έχουμε 1 υποψήφιο ταξί, εφόσον ο πελάτης είναι 5 άτομα:

```
The top 1 results depending on distance we found were:
(visual representation in map1.kml)
1:    Id: 190 Distance: 0.012969357822611424

Top 1 taxi candidates rearanged by rating:
(visual representation in map2.kml)
1 :   Id: 190 Rating: 6.6
```

Map1.kml & Map2.kml