Project-Erg3

Υλοποίηση αλγορίθμων υπόδειξης κρυπτονομισμάτων (Recommendation)

Author

Μάριος Παπαμιχαλόπουλος 1115201400149

Tools used

- C++11
- Git
- Valgrind 3.13.0
- Ubuntu Gnome 16.04
- Google C++11 Unit Testing Framework

Compilation

```
make
```

Execution

Έχω προσθέσει πέρα από τις παραμέτρους που ζητούνται στην εκφώνηση και μία έξτρα την -t, στην οποία δίνεται το dataset της προηγούμενης εργασίας σαν έξοδος. Για να τρέξει σωστά το πρόγραμμα πρέπει επίσης, να βρίσκονται στο ίδιο φάκελο με τη main τα αρχεία

‘vader_lexicon.csv’ και ‘coins_queries.csv’, γιατί τα paths τους είναι ορισμένα hard coded.

```
./recommendation -t <twitter_dataset> -d <tweets_dataset>
-o <output_file> -validate
```

Φιλοσοφία Προγράμματος & Περιγραφή Αρχείων

Χρήστες για τους οποίους δεν μπορώ να βρω γείτονες στα hash tables ή δεν έχουν γείτονες στη συστάδα εκτυπώνω κατάλληλο μήνυμα στο output αρχείο. Επίσης, εκτυπώνω το ίδιο μήνυμα για τους χρήστες, για τους οποίους δεν μπορώ να βρω εκ των προτέρων γείτονες, λόγω ότι τα tweets τους, περιέχουν μηδέν πληροφορία πριν την κανονικοποίηση.

```
recommendation.cpp: το αρχείο που βρίσκεται η συνάρτηση ma
in
```

```
recommendationLSH.cpp: αρχείο για το recommendation με τη
βοήθεια της δομής LSH. Περιέχει 2 είδη recommendation, ένα
 βρίσκοντας για κάθε χρήστη τους κοντίνοτερους P γείτονες
συγκρίνοντάς τους με - Για το clustering χρησιμοποιώ τον α
πλό k-means αλγόριθμο.
τους υπόλοιπους χρήστες και ένα βρίσκοντας για κάθε χρήστη
 τους κοντινότερους αντιπροσώπους(virtual users).
```

```
recommendationClustering.cpp: παρόμοια λειτουργία με το LS
H, αλλ με χρήση k-means clustering.
```

```
utilities.cpp: βοηθητικές συναρτήσεις για να είναι πιο ευδ
ιάκριτη η main.
```

```
validation.cpp:
```

- Για το clustering χρησιμοποιώ τον απλό k-means αλγόριθμο.

```
clustering.cpp: Αρχείο για το clustering. Χρησιμοποιώ τον
απλό k-means αλγόριθμο. Η συσταδοποίηση κρατάει MAX_LOOPS(
15) ή μέχρι η μεταβολή της συνάρτησης στόχου να γίνει μικρ
ότερη από 1%.
```

```
clustering_utilities.cpp: Αρχείο με βοηθητικές συναρτήσεις
 για το clustering.
```

Αρχεία από 1η εργασία:

```
fi.cpp
hyperplane.cpp
hash_table.
hash_node.h
help_functions.h
```

Παρατηρήσεις

Το dataset που δόθηκε εχει πολλούς χρήστες που δεν αναφέρονται σε κανένα κρυπτονόμισμα, δηλαδή τα tweets τους περιέχουν μη χρήσιμη πληροφορία. **Τέτοια διανύσματα δεν τα λαμβάνω καν υπόψη στη συσταδοποίηση**. Επίσης, μετά από την κανονικοποίηση των διανυσμάτων χρήστη-κρυπτονομίσματα, προκύπτουν και πολλά μηδενικά vectors. Πρέπει να σημειωθεί ότι το dataset δεν βοηθει και πολύ στην εκπόνηση αποτελεσμάτων, γιατί τα διανύσματα των παραπάνω περιπτώσεων αποτελούν περίπου 3000 από τα 3521 του dataset, αν και στη δεύτερη περίπτωση για την ομοιότητα με τους γείτονες χρησιμοποιώ το διάνυσμα πριν αυτό κανονικοποιηθεί.

Τα MAE και οι χρόνοι που παίρνω για τα (α) ερωτήματα της άσκησης είναι:

- **Recommendation MAE: 0.17016**

  Execution time: 6.41197 secs

- **Clustering Recommendation MAE: 0.166797**

  Execution time: 3.57837 secs

Τα MAE και οι χρόνοι που παίρνω για τα (β) ερωτήματα της σκησης είναι:

- **LSH Recommendation MAE: 0.305701**

  Execution time: 1.91381 secs

- **Clustering Recommendation MAE: 0.207658**

  Execution time: 0.794224 secs

Παρατηρώ ότι:

- Τα recommendation με βάση τους virtual users, δεν κάνουν τόσο καλό recommend, γιατί συσταδοποιούμε τα tweets όπως έχουν οριστεί στο χώρο της δεύτερης εργασας και ύστερα παράγουμε τους αντιπροσώπους της κάθε συστάδας. Σε αυτήν την περίπτωση ΔΕΝ τα συσταδοποιούμε με βάση το συναίσθημα του κάθε tweet, γι αυτό και είναι χειρότερα τα νούμερα.
- Το clustering recommendation είναι καλύτερο του LSH recommendation. Όσον αφορά τους χρόνους, επειδή χρησιμοποιώ 3 hash tables, το LSH είναι πιο χρονοβόρα διαδικασία.
- Παρατηρούμε ότι τα recommendation για τα (β) ερωτήματα είναι πιο γρήγορα αλλά απαιτούν παραπάνω προεργασία, οπότε αυτός ο χρόνος είναι λίγο πλασματικός.