ΙΟΥΛΙΟΣ
2015

ΧΑΡΟΚΟΠΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΜΑΤΙΚΗΣ

Π.Μ.Σ.: «ΠΛΗΡΟΦΟΡΙΚΗ & ΤΗΛΕΜΑΤΙΚΗ»

Κατεύθυνση: Τεχνολογίες και Εφαρμογές Ιστού

Δημιουργία εμπλουτισμένου γράφου σε Συστήματα Συστάσεων βασισμένα σε Ετικέτες.

Επιβλέπων Καθηγητής:
Βαρλάμης Ηρακλής, Επίκουρος Καθηγητής

Μέλη Τριμελούς Επιτροπής:
Μιχαήλ Δημήτριος, Επίκουρος Καθηγητής
Τσερπές Κωνσταντίνος, Λέκτορας

Σαμαρτζόπουλος Νίκος

Α.Μ: 12406

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα της διπλωματικής μου εργασίας και Επίκουρο Καθηγητή του Τμήματος Πληροφορικής & Τηλεματικής του Χαροκοπείου Πανεπιστημίου κ.Βαρλάμη Ηρακλή για την πολύτιμη καθοδήγηση και βοήθεια, καθώς και τον υποψήφιο διδάκτορα του τμήματος κ.Σαρδιανό Χρήστο για την πολύτιμη συμβολή του.

Τέλος, οφείλω να ευχαριστήσω θερμά την οικογένεια μου για τη στήριξη, τη συμπαράσταση και την κατανόησή τους σε όλη την διάρκεια των μεταπτυχιακών μου σπουδών.

Περίληψη

Στις μέρες μας, τα συστήματα συστάσεων έχουν αποδειχθεί ως ένα πολύτιμο εργαλείο για να διαχειρίζονται οι χρήστες την υπερφόρτωση των πληροφοριών. Η έρευνα στον συγκεκριμένο τομέα, η οποία κρατάει πάνω από δυο δεκαετίες, έχει βοηθήσει σημαντικά στην εύρεση νέων αλγορίθμων και τεχνικών, βελτιώνοντας συνεχώς την διαδικασία παραγωγής συστάσεων. Από την έρευνα που έχει πραγματοποιηθεί προκύπτει ότι δεν υπάρχει μια μοναδική λύση που να καλύπτει όλους τους τομείς-περιπτώσεις. Διαφορετικές τεχνικές πρέπει συνεχώς να παραμετροποιούνται και να βελτιώνονται (M. McNee, et al., 2002).

Ένας τομέας που κερδίζει όλο και περισσότερο ενδιαφέρον είναι τα κοινωνικά δίκτυα βασισμένα σε ετικέτες, καθώς οι εφαρμογές τους είναι αρκετά χρήσιμες για τα συστήματα συστάσεων. Οι ετικέτες μπορεί να παρέχουν αποσπασματικές πληροφορίες όχι μόνο για το περιεχόμενο των χρηστών, αλλά επίσης και για τις προτιμήσεις των χρηστών και ως εκ τούτου να συνεισφέρουν στην βελτίωση των συστάσεων.

Σκοπός της παρούσας διπλωματικής διατριβής είναι η παρουσίαση μιας προτεινόμενης προσέγγισης για την δημιουργία ενός εμπλουτισμένου γράφου, ο οποίος εμπεριέχει πληροφορίες από τον γράφο αξιολόγησης των χρηστών, από τον κοινωνικό γράφο και από τον γράφο προσθήκης ετικετών. Εν συνεχεία γίνεται η αξιολόγηση με τεχνικές συνεργατικού φιλτραρίσματος και παραγοντοποίησης πινάκων στον εμπλουτισμένο γράφο, σε έναν μικρότερο που παρουσιάζει συστάσεις για χρήστες, αντικείμενα και ετικέτες και στο κλασικό γράφο σύστασης αντικειμένων.

Λέξεις-κλειδιά: Συστάσεις, Συστήματα Συστάσεων, Συνεργατικό Φιλτράρισμα, Κοινωνικά δίκτυα βασισμένα σε ετικέτες, UserUser, ItemItem, FunkSVD, Δημιουργία εμπλουτισμένου γράφου, Lenskit, Παραγαντοποίηση Πινάκων.

Abstract

Nowadays, recommendation systems have been proved as a valuable tool for users to manage information overload. Research in this field, which holds more than two decades, has helped greatly in finding new algorithms and techniques, constantly improving the production process of recommendations. From the research that has been made, it shows that there isn't any unique solution that covers all areas-situations. Different techniques must constantly be parameterized and improved (M. McNee, et al., 2002).

An area that is gaining more and more interest are social networks based on tags, as their applications are quite useful for recommender systems. Tags may provide partial information, not only, for the content of users, but also for the user preferences and thus contributing to the improvement of the recommendations.

The aim of this diploma thesis is to present a proposed approach for the creation of an extended graph, which contains information from the evaluation graph of users, from the social graph and from the tagging graph. Additionally, an evaluation by collaborative filtering techniques and matrix factorization is taking place on the extended graph, on a smaller one which presents recommendations for users, objects and labels and on the classic recommendation objects graph.

Keywords: Recommendations, Recommender Systems, Collaborative Filtering, Social networks based on tags, UserUser, ItemItem, FunkSVD, Richgraph, Lenskit, Matrix Factorization

Περιεχόμενα

1. Εισαγωγή 8
   1.1 Συστήματα συστάσεων 8
   1.2 Σκοπός 9
   1.3 Δομή 10
2. Θεωρητικό Υπόβαθρο 11
   2.1 Προκλήσεις των Συστημάτων Σύστασης 11
   2.2 Τεχνικές συστάσεων 12
      2.2.1 Συνεργατικό φιλτράρισμα (Collaborative Filtering) 13
      2.2.2 Φιλτράρισμα βασισμένο στο περιεχόμενο (Content-Based Filtering) 15
   2.3 Υβριδικά συστήματα συστάσεων 16
   2.4 Matrix Factorization – singular value decomposition 17
   2.5 Μετρικές αξιολόγησης για την ορθότητα των συστάσεων 21
3. Συστήματα Συστάσεων βασισμένα σε ετικέτες 24
   3.1 Συναφείς εργασίες 24
4. Προτεινόμενη προσέγγιση για την δημιουργία εμπλουτισμένου γράφου. 27
   4.1 Περιγραφή 27
   4.2 Παράδειγμα 34
   4.3 Συμπεράσματα προτεινόμενης προσέγγισης 38
5. Εργαλεία για την δημιουργία και την αξιολόγηση των συστάσεων 39
   5.1 Lenskit 39
6. Πειραματική αξιολόγηση 45
   6.1 Σύνολα Δεδομένων 46
   6.2 Διαδικασία πειραμάτων και αποτελέσματα 48
   6.3 Συμπεράσματα πειραματικής διαδικασίας 64
7. Επίλογος και Μελλοντικές Επεκτάσεις 65
8. Παραρτήματα 67
   8.1 pom.xml 67
   8.2 Διάγραμμα επικοινωνίας των components για τον FunkSVD 69
9. Βιβλιογραφία 70

Εικόνες

Εικόνα 1.1 πίνακας βαθμολογίας χρηστών, όπου κάθε κελί αντιστοιχεί στην βαθμολογία που έδωσε ο χρήστης u στο αντικείμενο i 9
Εικόνα 2.1 Singular Valued Decomposition (SVD) 17
Εικόνα 2.2 κατάταξη των k τιμών 18
Εικόνα 2.3 παράδειγμα εφαρμογής SVD 19
Εικόνα 2.4 αλγόριθμος FunkSVD 21
Εικόνα 4.1 αναπαράσταση εμπλουτισμένου πίνακα 27
Εικόνα 4.2 κλάσεις του εργαλείου 28
Εικόνα 4.3 δομή του εργαλείου 29
Εικόνα 4.4 δημιουργία εμπλουτισμένου γράφου από την εισαγωγή τριών ξεχωριστών γράφων 30
Εικόνα 4.5 παράδειγμα Config αρχείου για το lastfm 33
Εικόνα 4.6 αλγόριθμος δημιουργίας nWeight 33
Εικόνα 4.7 παράδειγμα εμπλουτισμένου γράφου 34
Εικόνα 4.8 παράδειγμα user-user γράφου 35
Εικόνα 4.9 παράδειγμα user-user γράφου με χρονοσφραγίδα 35
Εικόνα 4.10 παράδειγμα user-item γράφου 36
Εικόνα 4.11 παράδειγμα user-item γράφου με χρονοσφραγίδα 36
Εικόνα 4.12 παράδειγμα user-item-tag γράφου 37
Εικόνα 5.1 trainTest block, eval.groovy 40
Εικόνα 5.2 CROSSFOLD 41
Εικόνα 6.1 μορφή των παραγόμενων csv αρχείων 48
Εικόνα 6.2 train-test richgraph 49
Εικόνα 6.3 train - test evaluation, eval.groovy 50

Πίνακες

Πίνακας 6.1 σύνολα δεδομένων 47
Πίνακας 6.2 πυκνότητα δεδομένων για τους πίνακες user-item 48

Διαγράμματα

Διάγραμμα 6.1 nWeights user-item, lastfm 51
Διάγραμμα 6.2 nWeights users, lastfm 51
Διάγραμμα 6.3 nWeights richgraph, lastfm 52
Διάγραμμα 6.4 οι μέσοι χρόνοι για την εκπαίδευση και την αξιολόγηση των 3 αλγορίθμων στον γράφο user-item του lastfm 52
Διάγραμμα 6.5 οι μέσοι χρόνοι για την εκπαίδευση και την αξιολόγηση των 3 αλγορίθμων στον γράφο users του lastfm 53

Διάγραμμα 6.6 αποτελέσματα μετρικών για τους τρείς αλγορίθμους στον γράφο user-item του lastfm 53
Διάγραμμα 6.7 αποτελέσματα μετρικών για τους τρείς αλγορίθμους στον γράφο users του lastfm 54
Διάγραμμα 6.8 nWeights user-item alternative, lastfm 54
Διάγραμμα 6.9 nWeights richgrpaph (with user item alternative), lastfm 55
Διάγραμμα 6.10 αποτελέσματα μετρικών για τον αλγόριθμο FunkSVD στους γράφους user-item και richgraph του lastfm 55
Διάγραμμα 6.19 nWeights user item alternative, delicious 56
Διάγραμμα 6.20 nWeights users, delicious 56
Διάγραμμα 6.21 nWeights richgraph, delicious 57
Διάγραμμα 6.22 οι μέσοι χρόνοι για την εκπαίδευση και την αξιολόγηση των 3 αλγορίθμων στον γράφο user-item alternative του delicious 57
Διάγραμμα 6.23 οι μέσοι χρόνοι για την εκπαίδευση και την αξιολόγηση των 3 αλγορίθμων στον γράφο users του delicious 58
Διάγραμμα 6.24 αποτελέσματα μετρικών για τους τρείς αλγορίθμους στον γράφο useritem alternative του delicious 58
Διάγραμμα 6.25 αποτελέσματα μετρικών για τους τρείς αλγορίθμους στον γράφο users του delicious 59
Διάγραμμα 6.26 αποτελέσματα μετρικών για τον αλγόριθμο FunkSVD στους γράφους user-item και richgraph, του delicious 59
Διάγραμμα 6.11 nWeights user - item movielense 60
Διάγραμμα 6.12 nWeights users movielens 60
Διάγραμμα 6.13 nWeights richgraph movielens 61
Διάγραμμα 6.14 οι μέσοι χρόνοι για την εκπαίδευση και την αξιολόγηση των 3 αλγορίθμων στον γράφο user-item του movielens 61
Διάγραμμα 6.15 οι μέσοι χρόνοι για την εκπαίδευση και την αξιολόγηση των 3 αλγορίθμων στον γράφο users του movielens 62
Διάγραμμα 6.16 αποτελέσματα μετρικών για τους τρείς αλγορίθμους στον γράφο user-item του movielens 62
Διάγραμμα 6.17 αποτελέσματα μετρικών για τους τρείς αλγορίθμους στον γράφο users του movielens 63
Διάγραμμα 6.18 αποτελέσματα μετρικών για τον αλγόριθμο FunkSVD στους γράφους user-item και richgraph, του movielens 63

Κενή σελίδα

1. Εισαγωγή

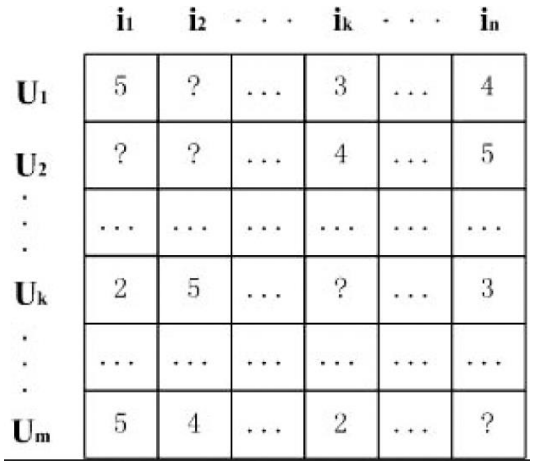
1.1 Συστήματα συστάσεων

Στις μέρες μας, η ποσότητα των πληροφοριών που διατίθεται στο διαδίκτυο αυξάνεται συνεχώς. Πλέον, οι χρήστες δυσκολεύονται να αναζητήσουν και να βρουν εύκολα και γρήγορα αυτό που πραγματικά επιθυμούν. Μια σημαντική λύση δίνουν τα συστήματα συστάσεων (RS, Recommendation System), τα οποία προσπαθούν να φιλτράρουν την πληροφορία που είναι πιθανώς ενδιαφέρουσα για κάθε χρήστη.

Ο ακόλουθοι όροι χρησιμοποιούνται συχνά στα συστήματα συστάσεων (Setten, 2005):

- Αντικείμενο (item): οτιδήποτε αναπαριστά μια πληροφορία, αυτό μπορεί να είναι προϊόντα, υπηρεσίες, άνθρωποι, ηλεκτρονικά έγγραφα, κ.λπ.
- Σύσταση (recommendation): το αποτέλεσμα του συστήματος, μπορεί να είναι ένα αντικείμενο ή μια λίστα από αντικείμενα τα οποία μπορεί να ενδιαφέρουν τον χρήστη.
- Πρόβλεψη (prediction): το αναμενόμενο ενδιαφέρον ενός χρήστη για ένα αντικείμενο. Η πρόβλεψη είναι διαφορετική από την έννοια της σύστασης, καθώς σύμφωνα με την πρώτη γίνεται μια υπόθεση σχετικά με το ενδιαφέρον του χρήστη για ένα αντικείμενο, ενώ η δεύτερη σχετίζεται με την πρόταση των ποιων σχετικών αντικειμένων για έναν χρήστη.
- Βαθμολογία (rating): ένα αντικειμενικό μέτρο το οποίο αναπαριστά το ενδιαφέρον του χρήστη για ένα αντικείμενο. Στα συστήματα συστάσεων έχουμε τις πραγματικές βαθμολογίες, δηλαδή τις υπάρχουσες αξιολογήσεις που έχουν κάνει οι χρήστες και τις προβλεπόμενες, δηλαδή αυτές που παράγει ο αλγόριθμος.
- Ακρίβεια της πρόβλεψης: ένα μέτρο το οποίο δείχνει σε ποιο βαθμό οι προβλεπόμενες βαθμολογίες συμφωνούν με τις υπάρχουσες-πραγματικές. Όσο πιο ακριβείς είναι οι προβλέψεις, τόσο καλύτερη η επίδοση του συστήματος.
- Τεχνική πρόβλεψης: ο συγκεκριμένος αλγόριθμος που χρησιμοποιείται στο σύστημα με σκοπό να υπολογίσει την προβλεπόμενη βαθμολογία για ένα αντικείμενο.

Ένα σύστημα συστάσεων είναι ένα πρόγραμμα, το οποίο παράγει συστάσεις διαφόρων «αντικειμένων» (π.χ. προϊόντα, υπηρεσίες, πρόσωπα) που μπορεί να ενδιαφέρουν ένα χρήστη ή προβλέπει την βαθμολογία που μπορεί να βάλει ένας χρήστης σε ένα αντικείμενο (Εικόνα 1.1). Πολύ γνωστές διαδικτυακές πλατφόρμες, όπως το Amazon.com, το eBay.com, IMDB.com, το Netflix.com, το Last.fm, κ.α. χρησιμοποιούν τέτοια συστήματα με σκοπό να προβλέψουν βαθμολογίες (ratings) αλλά και προτιμήσεις χρηστών.



Εικόνα 1.1 πίνακας βαθμολογίας χρηστών, όπου κάθε κελί αντιστοιχεί στην βαθμολογία που έδωσε ο χρήστης u στο αντικείμενο i

Τα συστήματα συστάσεων παραμένουν ένας ενεργός τομέας έρευνας τα τελευταία χρόνια, εμπλέκοντας διάφορα επιστημονικά πεδία όπως την στατιστική, την μηχανική μάθηση, την εξόρυξη δεδομένων και την ανάκτηση πληροφοριών. Το ενδιαφέρον σε αυτόν τον τομέα εξακολουθεί να παραμένει υψηλό, έχοντας διατυπωθεί πολλές διαφορετικές προσεγγίσεις. Αυτό οφείλεται στην αυξανόμενη ζήτηση για πρακτικές εφαρμογές, οι οποίες είναι σε θέση να παρέχουν εξατομικευμένη σύσταση, αναλύοντας τον τεράστιο όγκο πληροφοριών.

Οι πιο δημοφιλείς μέθοδοι που χρησιμοποιούνται στα συστήματα συστάσεων είναι το φιλτράρισμα με βάση το περιεχόμενο (content-based filtering) και το συνεργατικό φιλτράρισμα (collaborative filtering). Η σύγχρονη τάση ωστόσο είναι τα υβριδικά μοντέλα συστημάτων συστάσεων τα οποία συνδυάζουν χαρακτηριστικά των δυο προηγούμενων μεθόδων.

1.2 Σκοπός

Στόχος της εργασίας είναι η πρόταση ενός εργαλείου παραγωγής εμπλουτισμένου γράφου, το οποίο θα λαμβάνει υπόψη του τις βαθμολογίες των χρηστών, τις σχέσεις των χρηστών (δηλαδή ποιος είναι φίλος με ποιον) και τις ετικέτες που προσθέτουν οι χρήστες στα αντικείμενα. Οι ετικέτες στα κοινωνικά δίκτυα βοηθούν σημαντικά στην οργάνωση των δεδομένων. Τα συστήματα συστάσεων βασισμένα σε ετικέτες μπορούν να παρέχουν πληροφορίες σχετικά με το περιεχόμενο των αντικειμένων και των προτιμήσεων του χρήστη και συνεπώς βοηθούν ώστε να παραχθούν καλύτερες εξατομικευμένες συστάσεις. Σε αυτά τα συστήματα υπάρχουν τρείς σημαντικοί τύποι αντικειμένων: οι χρήστες (users), οι ετικέτες (tags) και οι πόροι-αντικείμενα (resources-items).

Στην συνέχεια, πραγματοποιείται η αξιολόγηση του γράφου, η οποία περιλαμβάνει δυο στάδια.

- Στην αρχή εκτελούνται οι αλγόριθμοι UserUser, ItemItem και FunkSVD στους γράφους:

    - χρήστες - αντικείμενα (user-item), όπου απεικονίζονται οι αξιολογήσεις των χρηστών για τα αντικείμενα και
    - στους χρήστες (users), το οποίο είναι ένα κομμάτι του εμπλουτισμένου γράφου και περιλαμβάνει χρήστες - χρήστες, χρήστες - αντικείμενα και χρήστες - ετικέτες. Ουσιαστικά, αν μπορούσαμε να το αναπαραστήσουμε ως ένα πίνακα, τότε ο πίνακας στην Εικόνα 1.1 θα περιείχε εκτός από αντικείμενα, και χρήστες και ετικέτες. Συνεπώς, θα μπορούσαν να γίνουν συστάσεις στους χρήστες για αντικείμενα, για χρήστες αλλά και για ετικέτες.
- Στο δεύτερο στάδιο εφαρμόζεται ο αλγόριθμος FunkSVD όπου εκπαιδεύεται από τα δεδομένα του εμπλουτισμένου γράφου (richmatrix), ο οποίος περιλαμβάνει υπονοούμενες ακμές που παράγονται από επεξεργασία σχετικών γράφων (π.χ. γράφοι μεταξύ χρηστών) και αξιολογείται στα δεδομένα του γράφου user-item. Ουσιαστικά, εξετάζουμε κατά πόσο μια τεχνική παραγοντοποίησης πινάκων, όπου χρησιμοποιεί και ο συγκεκριμένος αλγόριθμος, μπορεί να παράγει καλύτερες συστάσεις για αντικείμενα, αξιοποιώντας τον εμπλουτισμένο γράφο.

1.3 Δομή

Η παρούσα διπλωματική εργασία απαρτίζεται από επτά κεφάλαια. Στο επόμενο κεφάλαιο παρουσιάζεται η θεωρητική επισκόπηση των τεχνικών σύστασης και των μετρικών αξιολόγησης των συστημάτων σύστασης. Στο κεφάλαιο τρία γίνεται μια αναφορά στα συστήματα συστάσεων βασισμένα σε ετικέτες, καθώς και σε ορισμένες συναφείς εργασίες από την βιβλιογραφία. Στο κεφάλαιο τέσσερα παρουσιάζεται η προτεινομένη προσέγγιση για την δημιουργία του εμπλουτισμένου γράφου, στο κεφάλαιο πέντε περιγράφεται το εργαλείο Lenskit και στην συνέχεια στο έκτο κεφάλαιο αναλύονται τα αποτελέσματα από τις αξιολογήσεις των αλγορίθμων. Τέλος στο κεφάλαιο επτά γίνεται μια σύνοψη της εργασίας και επισημαίνονται ορισμένες μελλοντικές επεκτάσεις.

2. Θεωρητικό Υπόβαθρο

Ο όρος «συνεργατικό φιλτράρισμα» πρωτοεμφανίστηκε στο πλαίσιο του πρώτου εμπορικού συστήματος συστάσεων με την ονομασία Tapestry (Goldberg, Nichols, M. Oki, & Terry, 1992), το οποίο σχεδιάστηκε με σκοπό να προτείνει έγγραφα σε χρήστες. Το κίνητρο ήταν να αξιοποιηθεί η κοινωνική συνεργασία προκειμένου να αποτρέψει τους χρήστες από το να κατακλύζονται από ένα μεγάλο όγκο εγγράφων. Το συνεργατικό φιλτράρισμα, το οποίο αναλύει τα δεδομένα από όλους τους χρηστές προκειμένου να ταιριάξει σωστά τα ζεύγη χρηστών-αντικειμένων, έκτοτε αντιπαρατίθενται με την παλαιότερη μέθοδο, αυτή του φιλτραρίσματος με βάση το περιεχόμενο, η οποία έχει τις αρχικές ρίζες της στην ανάκτηση πληροφοριών.

Όπως σημειώνουν και οι Billsus & Pazzani (1998) οι αρχικές διατυπώσεις για τα συστήματα συστάσεων βασίστηκαν στην στατιστική συσχέτιση και στην προγνωστική μοντελοποίηση, χωρίς να υιοθετούν πρακτικές από τα πεδία της στατιστικής και της μηχανικής μάθησης. Το πρόβλημα του συνεργατικού φιλτραρίσματος συνδέθηκε με την τεχνική της κατηγοριοποίησης (classification), επιτρέποντας στις τεχνικές μείωσης των διαστάσεων (SVD) να εμφανιστούν στο προσκήνιο με σκοπό την βελτίωση, σε μεγάλο βαθμό, της ποιότητας των αποτελεσμάτων. Παράλληλα, αρκετές προσπάθειες έχουν καταβληθεί οι οποίες συνδυάζουν τεχνικές συνεργατικού φλιταρίσματος και φλιταρίσματος με βάση το περιεχόμενο.

Όλο και περισσότεροι ερευνητές ωθούνται προς το πεδίο των συστημάτων συστάσεων, κυρίως λόγω της δημοσιοποίησης των συνόλων δεδομένων στο διαδίκτυο και της άμεσης συσχέτισης του με το Ηλεκτρονικό Εμπόριο (Sharma & Gera, 2013). Παράδειγμα αποτελεί η διαδικτυακή πλατφόρμα Netflix[1] η οποία κυκλοφόρησε ένα σύνολο δεδομένων μεγάλης κλίμακας που περιείχε εκατομμύρια βαθμολογίες χρηστών σε χιλιάδες τίτλους ταινιών και ανακοίνωσε ανοικτό διαγωνισμό για τον καλύτερο αλγόριθμο συνεργατικού φιλτραρίσματος. Τεχνικές παραγοντοποίησης πινάκων με βάση την γραμμική άλγεβρα και την στατιστική ανάλυση αναδείχθηκαν καινοτόμες (Robillard, Maalej, Walker, & Zimmermann, 2014).

2.1 Προκλήσεις των Συστημάτων Σύστασης

Οι προκλήσεις για τους αλγόριθμους σύστασης επεκτείνονται σε τρεις βασικές διαστάσεις: τη σποραδικότητα των δεδομένων (sparsity), την επεκτασιμότητα (scalability) και τη δημιουργία συστάσεων για νέους χρήστες (cold-start) (Sharma & Gera, 2013).

- **Το πρόβλημα της σποραδικότητας** των δεδομένων είναι ένα από τα σημαντικότερα προβλήματα στο πεδίο των RS, καθώς επηρεάζει την ποιότητα των συστάσεων.

  Ο υπολογισμός της σποραδικότητας για ένα σύνολο δεδομένων δίνεται από τον παρακάτω τύπο:

[1] https://www.netflix.com/

$$1 - \frac{\text{μη μηδενικές καταχωρήσεις}}{\text{συνολικές καταχωρήσεις}},$$

συνεπώς σε έναν πίνακα χρηστών-αντικειμένων με βαθμολογίες είναι

$$1 - \frac{\text{βαθμολογίες}}{\text{χρήστες} * \text{αντικείμενα}}$$

Αν η σποραδικότητα είναι υψηλή αυτό οφείλεται στο γεγονός ότι οι περισσότεροι χρήστες βαθμολογούν λίγα αντικείμενα και συνεπώς οι διαθέσιμες βαθμολογίες είναι αραιές. Πολλές έρευνες ασχολούνται με αυτό το θέμα, παρόλα αυτά απαιτείται ακόμη περισσότερη προσπάθεια.

- **Το πρόβλημα δημιουργίας συστάσεων για νέους χρήστες** αναφέρεται στην κατάσταση κατά την οποία ένα νέος χρήστης ή ένα νέο αντικείμενο εισάγεται στο σύστημα. Σε αυτήν την περίπτωση είναι πολύ δύσκολο να παραχθούν συστάσεις, καθώς υπάρχει λίγη πληροφορία για αυτόν τον χρήστη και για το νέο αντικείμενο δεν υπάρχουν διαθέσιμες βαθμολογίες. Συνεπώς, το συνεργατικό φιλτράρισμα δεν μπορεί να βοηθήσει σε αυτές τις περιπτώσεις. Παρόλα αυτά, σύμφωνα με τις έρευνες (Burke, 2007), τα υβριδικά συστήματα μπορούν να συνεισφέρουν στην επίλυση αυτού του ζητήματος.

- **Η επεκτασιμότητα** χαρακτηρίζει το βαθμό στον οποίο ένα σύστημα μπορεί να ανταπεξέλθει στην αυξανομένη πληροφορία. Με την τεράστια αύξηση της πληροφορίας, τα συστήματα συστάσεων πρέπει να επεξεργαστούν πολλά δεδομένα. Σε αρκετές περιπτώσεις με την αύξηση του αριθμού των αντικειμένων και των χρηστών, οι υπολογισμοί αυξάνονται με γεωμετρική πρόοδο και μερικές φορές οδηγούνται σε ανακριβή αποτελέσματα. Μέθοδοι όπως η παραγοντοποίηση των πινάκων μπορούν να αντιμετωπίσουν το συγκεκριμένο πρόβλημα.

2.2 Τεχνικές συστάσεων

Οι τεχνικές συστάσεων κατηγοριοποιούνται ως εξής:

- Συνεργατικό φιλτράρισμα (Collaborative filtering): οι συστάσεις βασίζονται στις προηγούμενες βαθμολογίες όλων των χρηστών συλλογικά.
- Φιλτράρισμα βασισμένο στο περιεχόμενο (Content-based filtering): οι συστάσεις βασίζονται στις προηγούμενες προτιμήσεις του χρήστη και στην ομοιότητα που μπορεί να υπάρχει στο περιεχόμενο (περιγραφές, τίτλοι κλπ.) αυτών που έχει ήδη επιλέξει και των μελλοντικών προτάσεων.
- Υβριδική προσέγγιση: μέθοδοι που συνδυάζουν τις προηγούμενες τεχνικές.

2.2.1 Συνεργατικό φιλτράρισμα (Collaborative Filtering)

Το συνεργατικό φιλτράρισμα (CF) είναι η διαδικασία της αξιολόγησης των πληροφοριών με τη χρήση της γνώμης των άλλων χρηστών. Συνήθως, οι προβλέψεις σχετικά με τα ενδιαφέροντα των χρηστών γίνονται με τη συλλογή πληροφοριών από πολλούς άλλους παρόμοιους χρήστες (Schafer, Frankowski, Herlocker, &Sen, 2007).

Συχνά τα συστήματα CF πρέπει να επεξεργαστούν ένα τεράστιο όγκο δεδομένων. Μέσα στην τελευταία δεκαετία οι CF τεχνικές βελτιώνονται συνεχώς και είναι από τις πιο σημαντικές τεχνικές εξατομίκευσης στον τομέα των συστημάτων σύστασης.

Αρχικά οι CF τεχνικές εμπνεύστηκαν από τη φύση του ανθρώπου, και το γεγονός ότι οι άνθρωποι μοιράζονται απόψεις με άλλους ανθρώπους. Για παράδειγμα, όταν αρκετοί από τους καλούς φίλους σας, δείχνουν ιδιαίτερη προτίμηση για μια ταινία, μπορεί να αποφασίσετε και εσείς να πάτε να την δείτε. «Αυτό οφείλεται στο γεγονός ότι η γεύση είναι μια κοινωνιολογική έννοια και όχι μόνο ένα προσωπικό θέμα» (Schafer, Frankowski, Herlocker, &Sen, 2007).

Σήμερα, οι υπολογιστές και το Διαδίκτυο μας επιτρέπουν να εξετάσουμε τις απόψεις μεγάλων διασυνδεδεμένων κοινοτήτων, οι οποίες αποτελούνται από χιλιάδες μέλη. Τα άτομα μπορούν να επωφεληθούν από μια κοινότητα, καθώς έχουν πρόσβαση στη γνώση και στην εμπειρία άλλων χρηστών σχετικά με ποικίλα αντικείμενα. Επιπλέον, αυτή η πληροφορία μπορεί να βοηθήσει τα άτομα να αναπτύξουν την δική τους προσωπική άποψη ή να λάβουν μια απόφαση σχετικά με τα αντικείμενα τα οποία έχουν βαθμολογηθεί. Συνεπώς, οι χρήστες μέσω των CF τεχνικών μπορούν να πάρουν συστάσεις για συγκεκριμένα αντικείμενα και να συνδεθούν με άλλους χρήστες με τα ίδια ενδιαφέροντα.

Τα συστήματα CF χωρίζονται σε δύο κατηγορίες:

- με **βάση τη μνήμη (memory-based)** και
- με **βάση το μοντέλο (model-based)**.

Οι memory-based αλγόριθμοι χρησιμοποιούν ένα δισδιάστατο πίνακα χρηστών-αντικειμένων στον οποίο αποθηκεύονται οι βαθμολογίες που δίνει κάθε χρήστης για κάθε αντικείμενο (βλ. Εικόνα 1.1). Η πιο δημοφιλής μη-πιθανοτική προσέγγιση που χρησιμοποιείται σε αυτούς τους αλγορίθμους είναι ο αλγόριθμος k κοντινότερων γειτόνων (k-nearest neighbors). Σε γενικές γραμμές υπάρχουν δύο βασικές τεχνικές: με **βάση το χρήστη (User-Based)** και με **βάση το αντικείμενο (Item-Based)**.

Σύμφωνα με την πρώτη δίνεται σημασία στην εύρεση των χρηστών που ακολουθούν το ίδιο μοτίβο βαθμολόγησης με τον εξεταζόμενο χρήστη. Για να παραχθούν πιο ακριβείς προβλέψεις, ανατίθονται στις βαθμολογίες των γειτόνων βάρη με βάση την ομοιότητα που έχουν με τον εξεταζόμενο χρήστη. Για τον υπολογισμό της ομοιότητας μεταξύ δυο χρηστών u, u΄ , ορίζεται μια συνάρτηση ομοιότητας $\mathrm{Sim}_{(u,u')}$.

Για την πρόβλεψη της βαθμολογίας του χρήστη u στο αντικείμενο i χρησιμοποιείται ο σταθμισμένος μέσος όρος που ορίζεται ως εξής:

$$p_{(u,i)} = \frac{\Sigma_{u' \in N} \, Sim_{(u,u')} * r_{(u',i)}}{\Sigma_{u' \in N} \left| Sim_{(u,u')} \right|} \quad \text{ή} \quad = \bar{r}_u + \frac{\Sigma_{u' \in N} \, Sim_{(u,u')} * (r_{(u',i)} - \bar{r}_{u'})}{\Sigma_{u' \in N} \left| Sim_{(u,u')} \right|}$$

όπου N είναι το σύνολο των γειτόνων του u, $\bar{r}_u$ η μέση βαθμολογία του χρήστη u για όλα τα αντικείμενα που έχει βαθμολογήσει και $r_{(u',i)}$ η βαθμολογία του u΄ για το αντικείμενο i.

Γενικά, μια κρίσιμη απόφαση για την υλοποίηση ενός user-user αλγορίθμου είναι η επιλογή της συνάρτησης ομοιότητας χρήστη. Μια γνωστή μέθοδος με πολύ καλά αποτελέσματα (Breese, Heckerman, & Kadie, 1998), (Herlocker, Konstan, & Riedl, 2002) είναι η χρήση του **συντελεστή συσχέτισης Pearson (Pearson's Correlation Coefficient)**, η οποία συγκρίνει τις αξιολογήσεις για τα αντικείμενα τα οποία βαθμολογούνται από τον εξεταζόμενο χρήστη και τους γείτονες του. Ο τύπος δίνεται παρακάτω:

$$sim_{(u,u')} = \frac{\sum_{i=1}^{n} (r_{u,i} - \bar{r}_u)(r_{u',i} - \bar{r}_{u'})}{\sqrt{\sum_{i=1}^{n} (r_{u,i} - \bar{r}_u)^2} \sqrt{\sum_{i=1}^{n} (r_{u',i} - \bar{r}_{u'})^2}}$$

Όπου $r_{u,i}$ η βαθμολογία του χρήστη u για το αντικείμενο i και $r_{u',i}$ η βαθμολογία του χρήστη u΄ για το αντικείμενο i και $\bar{r}_u$, $\bar{r}_{u'}$ η μέση τιμή των βαθμολογιών για τους χρήστες u και u΄.

Μια άλλη μέθοδος είναι η χρήση της **Ομοιότητας Συνημιτόνου (Cosine Similarity)**. Αυτό το μοντέλο είναι διαφορετικό από το προηγούμενο, καθώς βασίζεται στην λογική του διανυσματικού χώρου και όχι στην στατιστική. Η ομοιότητά των χρηστών υπολογίζεται από το συνημίτονο της γωνίας που σχηματίζουν τα δύο διανύσματα. Η εξίσωση είναι η εξής:

$$sim_{(u,u')} = \frac{r_u \cdot r_{u'}}{\|r_u\| \|r_{u'}\|} = \frac{\sum_{i=1}^{n} r_{u,i} \, r_{u',i}}{\sqrt{\sum_{i=1}^{n} r_{u,i}^2} \sqrt{\sum_{i=1}^{n} r_{u',i}^2}}$$

όπου το « · » δηλώνει το εσωτερικό γινόμενο των δυο διανυσμάτων. Αν το αποτέλεσμα είναι 1 τότε έχουμε την απόλυτη ταύτιση αν είναι 0 τότε τα διανύσματα είναι ανεξάρτητα μεταξύ τους.

Άλλες γνωστές μετρικές είναι η Constrained Pearson Correlation, όπου χρησιμοποιεί μια τιμή ως κεντρικό σημείο αντί για την μέση βαθμολογία $\bar{r}_u$ και η Spearman Rank Correlation, όπου χρησιμοποιούνται οι κατατάξεις των αντικειμένων αντί για τις βαθμολογίες

Αντίθετα, σύμφωνα με την **Item-based τεχνική**, η οποία περιεγράφηκε πρώτα στην βιβλιογραφία από τους (Sarwar B. , Karypis, Konstan, & Riedl, 2001) και (Karypis, 2001) παράγονται προβλέψεις βασιζόμενες στις ομοιότητες μεταξύ των αντικειμένων. Για ένα ζευγάρι χρήστη-αντικειμένου <u, i> η πρόβλεψη υπολογίζεται από το σταθμισμένο άθροισμα των βαθμολογιών εκείνων των αντικειμένων που είναι πιο σχετικές με το αντικείμενο i. Για την πρόβλεψη της βαθμολογίας του χρήστη u στο αντικείμενο i δίνεται η παρακάτω εξίσωση:

$$p_{(u,i)} = \frac{\Sigma_{i' \in I} \, sim_{(u,i')} * r_{(u,i')}}{\Sigma_{i' \in I} \left| sim_{(i,i')} \right|}$$

Όπου I το σύνολο των αντικειμένων τα οποία είναι όμοια με το i και $r_{(u,i')}$ η βαθμολογία του χρήστη u στο αντικείμενο i΄.

Όπως και στον user-user αλγόριθμο, έτσι και στον item-item υπάρχουν διάφοροι μέθοδοι για τον υπολογισμό τις ομοιότητας των αντικειμένων.

Η ομοιότητα συνημίτονου ανάμεσα σε διανύσματα αντικειμένων έχει την καλύτερη εφαρμογή καθώς είναι απλή, γρήγορη και παράγει καλύτερη ακρίβεια στις προβλέψεις. Η μέθοδος Pearson Correlation έχει επίσης προταθεί για item–item συστάσεις, αλλά δεν δουλεύει τόσο καλά όπως η Cosine Similarity (Sarwar B. , Karypis, Konstan, & Riedl, 2001). Επίσης, στην βιβλιογραφία (Karypis, 2001) αναφέρεται και η Conditional Probability, κυρίως σε πεδία με μοναδιαίες («like it») βαθμολογίες, όπως σε ιστοσελίδες αγορών.

Τέλος, οι **model-based αλγόριθμοι** συμπεριλαμβάνουν την εξαγωγή κάποιας πληροφορίας από το σύνολο των δεδομένων των βαθμολογιών, την οποία και χρησιμοποιούν σαν μοντέλο για να παράγουν συστάσεις, χωρίς να χρειάζεται να χρησιμοποιήσουν ολόκληρο το σύνολο δεδομένων κάθε φορά. Ουσιαστικά, αυτοί οι αλγόριθμοι εκπαιδεύονται χρησιμοποιώντας τα διαθέσιμα δεδομένα και στη συνέχεια εφαρμόζονται για να προβλέψουν τις βαθμολογίες των χρηστών σε καινούρια αντικείμενα. Το κύριο πλεονέκτημα σε αυτούς τους αλγορίθμους είναι ότι μπορούν να αντιμετωπίσουν το πρόβλημα της επεκτασιμότητας, και επίσης είναι σχετικά πιο γρήγοροι από τους memory-based (Breese, Heckerman, & Kadie, 1998) . Παρόλο που η βασική ιδέα πίσω από τα model-based συστήματα συστάσεων είναι η ίδια, στη βιβλιογραφία υπάρχουν διάφορες model-based προσεγγίσεις για την σύσταση αντικειμένων όπως τα bayesian δίκτυα, τα νευρωνικά δίκτυα, η μέγιστη εντροπία, οι μηχανές Boltzmann, οι μηχανές διανυσμάτων υποστήριξης (Support Vector Machines) ή η παραγοντοποίηση ιδιαζουσών τιμών (Singular Value Decomposition).

2.2.2 Φιλτράρισμα βασισμένο στο περιεχόμενο (Content-Based Filtering)

Οι CF προσεγγίσεις μεταχειρίζονται όλους τους χρήστες και τα αντικείμενα σαν ολότητες, όπου οι προβλέψεις γίνονται χωρίς να λαμβάνονται υπόψη οι ιδιαιτερότητες των μεμονωμένων χρηστών ή των αντικειμένων. Τα Content-based συστήματα συστάσεων (CBF) βασίζονται στο περιεχόμενο των αντικειμένων και στα ενδιαφέροντα του χρήστη (Pazzani&Billsus, 2007). Συνήθως, τα εξατομικευμένα προφίλ δημιουργούνται αυτόματα μέσα από την ανατροφοδότηση των χρηστών (user feedback), και περιγράφουν το είδος των αντικειμένων που αρέσουν στον χρήστη. Προκειμένου να προσδιοριστούν τα αντικείμενα που θα συσταθούν, συγκρίνονται όλες οι πληροφορίες του χρήστη με τα χαρακτηριστικά των αντικειμένων που είναι προς εξέταση.

Συνήθως τα αντικείμενα αποθηκεύονται στις γραμμές ενός πίνακα και τα χαρακτηριστικά των αντικειμένων στις στήλες. Αν για παράδειγμα έχουμε ένα πίνακα με τραγούδια (γραμμές) και ετικέτες (στήλες), τότε θα μπορούσαμε να γεμίσουμε με τιμές 0,1 (1 αν έχει δοθεί η συγκεκριμένη ετικέτα σε αυτό το τραγούδι, 0 αν όχι) ή με αριθμούς που δείχνουν πόσες φορές έχει δοθεί η συγκεκριμένη ετικέτα σε ένα τραγούδι.

Τις περισσότερες φορές, το προφίλ ενός χρήστη περιέχει περιγραφές για τα αντικείμενα, τα οποία του αρέσουν περισσότερο ή πληροφορίες σχετικά με την αλληλεπίδραση των χρηστών με το σύστημα σύστασης. Οι αλληλεπιδράσεις αυτές μπορεί να περιλαμβάνουν την αποθήκευση των ερωτήματα των χρηστών, καθώς και την καταγραφή των δραστηριοτήτων των χρηστών που σχετίζονται με συγκεκριμένα αντικείμενα. Το ιστορικό των αλληλεπιδράσεων ενός χρήστη εκτός

του ότι τον διευκολύνει στην πλοήγηση του μέσα στον ιστότοπο, μπορεί επιπλέον να χρησιμεύσει ως σύνολο δεδομένων εκπαίδευσης για τους αλγόριθμους μηχανικής μάθησης, προκειμένου να δημιουργήσουν ένα πιο εξελιγμένο μοντέλο συστάσεων.

Κάποιες τεχνικές που χρησιμοποιούνται στα Content-based συστήματα συστάσεων είναι οι Μπεϋζιανοί Ταξινομητές (Bayesian classifiers), τα δέντρα απόφασης και τα τεχνητά νευρωνικά δίκτυα.

2.3 Υβριδικά συστήματα συστάσεων

Γενικά, τα υβριδικά συστήματα συστάσεων είναι συστήματα που συνδυάζουν διάφορες τεχνικές συστάσεων μεταξύ τους επιδιώκοντας ακόμη καλύτερα αποτελέσματα.

Ανάλογα με τα χαρακτηριστικά του τομέα και των δεδομένων, διάφοροι τύποι συνδυασμών θα μπορούσαν να παράγουν ανόμοια αποτελέσματα.

Σύμφωνα με τον Burke (2007) η ακόλουθη λίστα περιγράφει διάφορες τεχνικές που θα μπορούσαν να ληφθούν υπόψη κατά την συγχώνευση CF και CBF συστημάτων συστάσεων.

- Weighted: συνδυασμός των διαφόρων τεχνικών σύστασης, όπου το τελικό αποτέλεσμα είναι ένας γραμμικός συνδυασμός των ενδιάμεσων αποτελεσμάτων.
- Mixed: παρουσιάζονται ταυτόχρονα στον χρήστη συστάσεις από διαφορετικά συστήματα συστάσεων.
- Switching: γίνεται εναλλαγή των διαφόρων τεχνικών, έτσι ώστε εάν η πρώτη δεν μπορεί να παράγει μια σύσταση με υψηλή εμπιστοσύνη, τότε χρησιμοποιείται η επόμενη, και ούτω καθεξής.
- Feature Combination: Τα χαρακτηριστικά από διαφορετικές τεχνικές σύστασης συνδυάζονται και χρησιμοποιούνται μαζί σε έναν ενιαίο αλγόριθμο σύστασης.
- Αύξηση χαρακτηριστικών (Feature Augmentation): Μια τεχνική σύστασης χρησιμοποιείται για να υπολογιστεί ένα χαρακτηριστικό (ή ένα σύνολο χαρακτηριστικών) το οποίο στη συνέχεια χρησιμοποιείται ως είσοδος σε μια άλλη τεχνική.
- Διαδοχική υβριδοποίηση (Cascade): Η τεχνική αυτή περιλαμβάνει μια σταδιακή διαδικασία κατά την οποία πρώτα χρησιμοποιείται μια τεχνική σύστασης για να παραχθεί μια κατάταξη υποψηφίων συστάσεων και έπειτα μια δεύτερη τεχνική βελτιώνει τις συστάσεις που δίνονται από την πρώτη τεχνική.
- Υβριδοποίηση μετα-επιπέδου (Meta-Level): παράγεται ένα μοντέλο το οποίο έπειτα χρησιμοποιείται στο αρχικό σύστημα συστάσεων.

2.4 Matrix Factorization – singular value decomposition

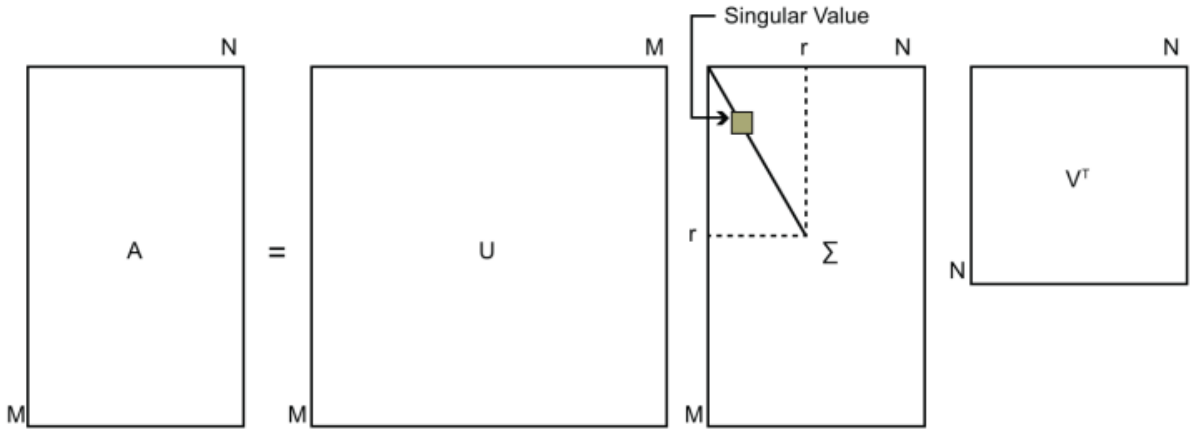
Τα συστήματα συστάσεων επεξεργάζονται τεράστια ποσότητα πληροφορίας, και πρέπει να παράγουν ακριβείς συστάσεις σε μικρό χρόνο. Το συνεργατικό φιλτράρισμα εφαρμόζει τον nearest neighborhood (NNH) αλγόριθμο σε συνδυασμό με μια συνάρτηση ομοιότητας για να υπολογίσει προβλέψεις αξιολόγησης. Παρόλα αυτά, λόγω της σποραδικότητας της πληροφορίας οι NNH αλγόριθμοι συνήθως αντιμετωπίζουν δυσκολίες στην εξεύρεση σωστών αντιστοιχιών και κατά συνέπεια παράγουν συστάσεις με χαμηλή ακρίβεια. Επιπλέον, η υπολογιστική πολυπλοκότητα των αλγορίθμων NNH τείνει να μεγαλώνει καθώς αυξάνονται τα δεδομένα, προκαλώντας προβλήματα κλιμάκωσης.

Για να ξεπεραστούν αυτές οι αδυναμίες εφαρμόζονται τεχνικές παραγοντοποίησης (Matrix factorization). Μια καθιερωμένη τεχνική παραγοντοποίησης πινάκων με πολλές εφαρμογές στην επεξεργασία σήματος και στην στατιστική (Sarwar, Karypis, Konstan, &Riedl , 2000) είναι η Singular Valued Decomposition (SVD).

Σύμφωνα με την τεχνική, υποθέτοντας ότι έχουμε ένα πίνακα A (mxn) με βαθμό r, τότε μπορούμε να διασπάσουμε αυτόν τον πίνακα σε τρεις πίνακες:

$$A=U \cdot \Sigma \cdot V^T$$

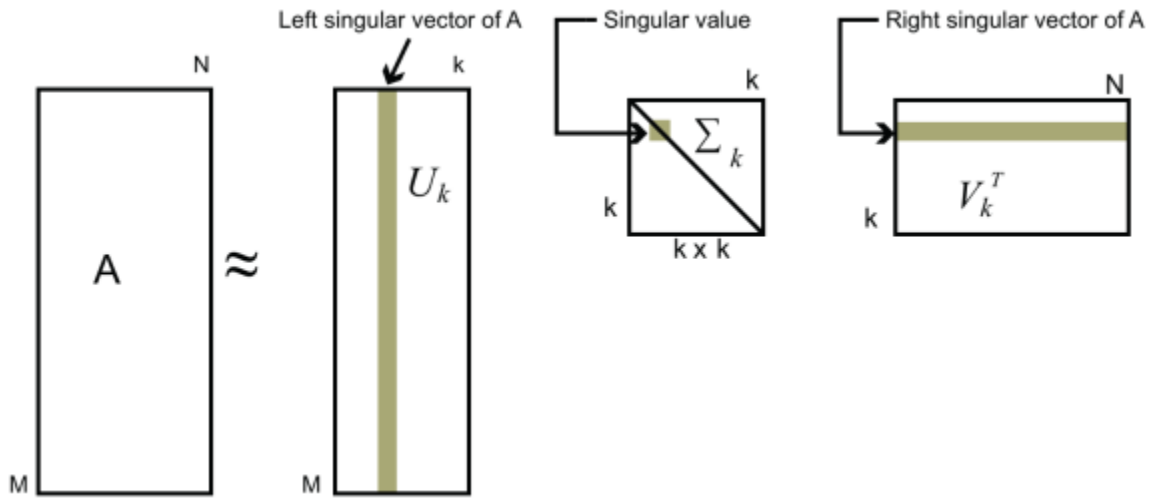
Οι πίνακες U και V είναι ορθογώνιοι πίνακες με μέγεθος mxr και nxr αντίστοιχα και ο πίνακας S είναι ένας διαγώνιος πίνακας (rxr), του οποίου τα διαγώνια στοιχεία είναι οι ιδιοτιμές του πίνακα A.



Εικόνα 2.1 Singular Valued Decomposition (SVD), πηγή: Barbieri, Manco, & Ritacco, (2014)

Για να μειώσουμε τις διαστάσεις, επιλέγουμε τις k μεγαλύτερες ιδιοτιμές και τα αντίστοιχα αριστερά και δεξιά ιδιοδιανύσματα και παράγουμε τον πίνακα $A_\kappa(mxn)$.

$$A \approx A_k = U_k \cdot \Sigma_k \cdot V_k^T$$



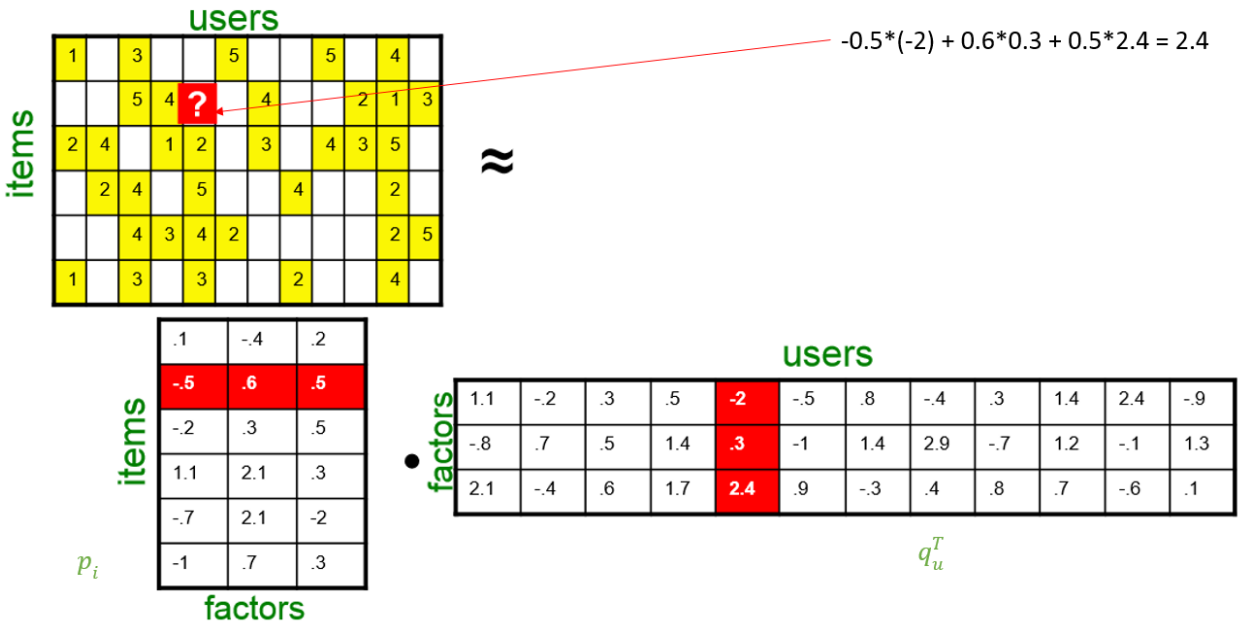
Εικόνα 2.2 κατάταξη των k τιμών, πηγή: Barbieri, Manco, & Ritacco, (2014)

Η εφαρμογή της τεχνικής αυτής στο πεδίο του συνεργατικού φιλτραρίσματος, αποτελεί την παραγοντοποίηση του πίνακα «χρήστη - αντικειμένων», δημιουργώντας μια χαμηλών - διαστάσεων αναπαράσταση του αρχικού πίνακα (υψηλών διαστάσεων) βαθμολογιών. Με αυτόν τον τρόπο αποκαλύπτονται οι κρυφές σχέσεις μεταξύ των χρηστών και των αντικειμένων που θα μπορούσαν να χρησιμοποιηθούν για να συναχθεί η προτίμηση ενός χρήστη για ένα εξεταζόμενο αντικείμενο. Σύμφωνα με την τεχνική, γίνεται αντιστοίχηση των χρηστών και των αντικειμένων σε ένα κοινό χώρο λανθάνουσων τιμών, διάστασης f, έτσι ώστε οι αλληλεπιδράσεις χρηστών - αντικειμένων να παρουσιάζονται ως εσωτερικά δεδομένα σε αυτόν τον χώρο. Κάθε αντικείμενο i απεικονίζεται με ένα διάνυσμα q_i και κάθε χρήστης u με ένα διάνυσμα p_u.

Το γινόμενο $q_u^T p_i$ (οι τιμές του διαγώνιου πίνακα Σ_K είναι ενσωματωμένες) δείχνει την σχέση ανάμεσα στον χρήστη u και στο αντικείμενο i, αναπαριστώντας το συνολικό ενδιαφέρον – βαθμολογία του χρήστη στα χαρακτηριστικά του αντικειμένου.

Ακολουθεί ένα απλοποιημένο παράδειγμα εφαρμογής του SVD, το οποίο απεικονίζει τις βαθμολογίες που έχουν βάλει οι χρήστες στα αντικείμενα. Για κ = 3 ιδιάζουσες τιμές (ορίζεται πειραματικά), τότε κάθε αντικείμενο περιγράφεται από 3 χαρακτηριστικά, δείχνοντας σε ποιο βαθμό αποτελείται από αυτά τα χαρακτηριστικά και κάθε χρήστης περιγράφεται από 3 χαρακτηριστικά, δείχνοντας σε ποιο βαθμό προτιμάει κάθε χαρακτηριστικό.

Στο παρακάτω παράδειγμα παρουσιάζεται ο υπολογισμός της βαθμολογίας r_{25}



Εικόνα 2.3 παράδειγμα εφαρμογής SVD

Γενικά, σημειώνονται όμως μερικά προβλήματα. Οι αλγόριθμοι που υπολογίζουν SVD είναι αργοί, ιδιαίτερα σε μεγάλους πίνακες, αργούν πάρα πολύ να τρέξουν και θέλουν ιδιαίτερη προσοχή στο πως θα διαχειριστούν τα ελλείποντα δεδομένα. Για παράδειγμα ας πάρουμε το σύνολο δεδομένων του movielens, το οποίο απεικονίζει τις βαθμολογίες που έχουν βάλει οι χρήστες στις ταινίες (βλ. Πίνακας 6.1). Έστω R ο πίνακας ο οποίος έχει 2113 (χρήστες) * 10197 (ταινίες) = 21,546,261 κελιά. Κάθε μη-κενό κελί r_{ij} αναπαριστά μια γνωστή βαθμολογία που έχει δώσει ο χρήστης i στην ταινία j. Στο παράδειγμα μας ο πίνακας R έχει 21,546,261 – 855598 = 20,690,663 κενά κελία. Συνεπώς η μέθοδος SVD δεν θα είναι τόσο αποτελεσματική.

Μια μέθοδος που χρησιμοποιείται και στον FunkSVD είναι η μέθοδος του αλγορίθμου καθοδικής κλίσης (Gradient Descent Algorithm), η οποία έχει δύο σημαντικά πλεονεκτήματα. Πρώτον, είναι πολύ πιο γρήγορη και δεύτερον δουλεύει πολύ καλά με τα ελλείποντα δεδομένα. Αυτή η μέθοδος χρησιμοποιεί την κλίση της αντικειμενικής συνάρτησης έτσι ώστε να εντοπίζει την ελάχιστη τιμή. Η βασική ιδέα χρήση της στα συστήματα συστάσεων είναι ότι έχοντας ένα μεγάλο εύρος τιμών προσπαθεί να βρει το χαρακτηριστικό χρήστη και το χαρακτηριστικό αντικειμένου που ελαχιστοποιεί το τετραγωνικό σφάλμα των βαθμολογιών. Με αυτόν τον τρόπο μπορεί να βελτιωθεί η πρόβλεψη.

Ο αλγόριθμος FunkSVD[2] (Funk, 2006), χρησιμοποιεί μια Stochastic Gradient Descent τεχνική, καθώς ο αλγόριθμος τρέχει για όλες τις βαθμολογίες του training του συνόλου δεδομένων. Σε κάθε

[2] Ο αλγόριθμος του Simon Funk έλαβε την τρίτη θέση στον διαγωνισμό του Netflix, στον οποίο προτάθηκαν διάφοροι αλγόριθμοι πρόβλεψης βαθμολογίας των χρηστών για τις ταινίες.

επανάληψη το σύστημα προβλέπει την βαθμολογία, υπολογίζει το σχετικό σφάλμα πρόβλεψης και ανανεώνει τις τιμές των χαρακτηριστικών του χρήστη και των αντικειμένων.

Πιο συγκεκριμένα, όπως φαίνεται και στην Εικόνα 2.4, ο αλγόριθμος FunkSVD περιγράφεται ως εξής:

- Στην αρχή γίνεται η αρχικοποίηση των πινάκων U και V (συνήθως με 0.1, ή με τυχαίες μικρές τιμές μη-μηδενικές). Όπου U ← m*k και V ←n*k, m: χρήστες, n: αντικείμενα, k: χαρακτηριστικά , και $R \approx U \cdot V^T$ (οι τιμές του πίνακα Σ είναι ενσωματωμένες στους U, V)

- Στην συνέχεια εκπαιδεύει κάθε χαρακτηριστικό f και επαναλαμβάνοντας μέχρι κάποιο αριθμό (ο οποίος μπορεί να είτε προκαθορισμένος είτε όχι), υπολογίζει για κάθε βαθμολογία $\mathbf{r_{ai}}$:
  - την πρόβλεψη $\mathbf{p_{ai}}$ η οποία είναι ίση με την στατιστική απόκλιση (bias) που αφορά την βαθμολογία r_{ai} συν το γινόμενο των διανυσμάτων $\vec{U}_a$ και $\vec{V}_i$, δηλαδή

    $$\mathbf{p_{ai} = \mu + b_a + b_i + \vec{U}_a * \vec{V}_i},$$

    όπου μ: μέση βαθμολογία όλων των αντικειμένων και b_a, b_i: παρατηρούμενες αποκλίσεις του χρήστη a και του αντικειμένου i (π.χ. έστω ότι ο χρήστης a βαθμολογεί με 0.5 παραπάνω από τον μέσο όρο και έστω ότι η ταινία i βαθμολογείται με 1 παραπάνω από τον μέσω όρο των βαθμολογιών, ο οποίος είναι 3.1, τότε το $bias_{ai}$ = 3.1 + 0.5 + 1= 4.6),
  - το σφάλμα πρόβλεψης $\mathbf{e_{ai} = r_{ai} - p_{ai}}$
  - και τέλος ανανεώνει τις τιμές των χαρακτηριστικών χρήστη $U_{\alpha f}$ και αντικειμένων V_{if}, υπολογίζοντας την μεταβολή:

    $$\mathbf{\Delta_{U_{\alpha f}} = \lambda * (e_{ai} * V_{if} + \gamma * U_{\alpha f})} \text{ και}$$

    $$\mathbf{\Delta_{V_{if}} = \lambda * (e_{ai} * U_{\alpha f} - \gamma * V_{if}),}$$

    όπου λ: ο ρυθμός μάθησης, συνήθως 0.001 και γ: ο ρυθμιστικός όρος (regularization term) με τιμές 0.1 - 0.2.

Δομή του αλγορίθμου FunkSvd

Αρχικοποίηση των πινάκων U,V

Για f από 1 μέχρι k

  Μέχρι κάποιον αριθμό επανέλαβε

    Για ∀ r_{ai}

      Υπολόγισε την πρόβλεψη p_{ai}

      Υπολόγισε το σφάλμα πρόβλεψης e_{ai}

      Άλλαξε το $U_{\alpha f}$

      Άλλαξε το V_{if}

Εικόνα 2.4 αλγόριθμος FunkSVD

Γενικά, η χρήση των αλγορίθμων SVD στα συστήματα συστάσεων σκοπό έχει να αποκαλύψει τις κρυμμένες σχέσεις ανάμεσα σε χρήστες και αντικείμενα που επιτρέπουν να υπολογιστεί η προβλεπόμενη βαθμολογία που θα βάλει ο χρήστης σε ένα αντικείμενο και επίσης, να ελαττώσει τη σποραδικότητα του αρχικού πίνακα χρηστών-αντικειμένων και συνεπώς να βελτιώσει την ακρίβεια πρόβλεψης του συνολικού συστήματος συστάσεων (Hu, Meng, & Wang, 2011).

2.5 Μετρικές αξιολόγησης για την ορθότητα των συστάσεων

Σύμφωνα με τις έρευνες (Gunawardana & Shani, 2009), έχουν χρησιμοποιηθεί διάφοροι τύποι μετρικών για την αξιολόγηση της ποιότητας των συστημάτων συστάσεων. Αναλόγως τον τύπο των συστάσεων που παράγει ένα σύστημα, μπορούν να χρησιμοποιηθούν διαφορετικές μετρικές για την αξιολόγηση. Ένα RS μπορεί να προβλέψει πως οι χρήστες βαθμολογούν ένα αντικείμενο, την σειρά εμφάνισης των βαθμολογιών (από τα πιο ενδιαφέροντα αντικείμενα στα λιγότερα) ή ποιο/α αντικείμενο/α είναι πιο ενδιαφέροντα για κάθε χρήστη. Οι μετρικές θα μπορούσαν να κατηγοριοποιηθούν ως εξής:

- Ακρίβεια πρόβλεψης: πόσο καλά μπορεί ένα σύστημα συστάσεων να προβλέψει τις προτιμήσεις ενός χρήστη. Οι μετρικές που χρησιμοποιούνται πιο συχνά είναι το **Μέσο Τετραγωνικό Σφάλμα (Root Mean Square Error - RMSE)** και το **Μέσο Απόλυτο Σφάλμα (Mean Absolute Error – MAE)**.

Για να υπολογιστεί η RMSE, πρέπει να προσδιοριστεί η διαφορά μεταξύ της πραγματικής βαθμολογίας του χρήστη και της πρόβλεψης. Έστω r_{ui} η πραγματική βαθμολογία του χρήστη u για το αντικειμενο i και $\hat{r}_{ui}$ η πρόβλεψη, τότε η απόκλιση είναι $\hat{r}_{ui}$ - r_{ui}. Αναλογως των προβλέψεων η απόκλιση μπορεί να είναι θετική η αρνητική. Ο τύπος της RMSE δίνεται παρακάτω:

$$\text{RMSE} = \sqrt{\frac{\Sigma_{(u,i)\in T}\ (\hat{r}_{ui} - r_{ui})^2}{N}}$$

Ενώ η MAE υπολογίζεται ως εξής:

$$\text{MAE} = \sqrt{\frac{\Sigma_{(u,i)\in T}\ |\hat{r}_{ui} - r_{ui}|}{N}}$$

Όπου N ο αριθμός όλων των βαθμολογιών και T το τεστ σύνολο δεδομένων.

Στην MAE όλες οι αποκλίσεις υπολογίζονται ισοδύναμα, ενώ στην RMSE τα μεγάλα λάθη «τιμωρούνται» περισσότερο από τα μικρά. Συνεπώς, η MAE χρησιμοποιείται σε περιπτώσεις όπου ο χρήστης δεν ενδιαφέρεται ιδιαίτερα για λάθη τα οποία παρουσιάζονται σε αντικείμενα που σημειώνουν ή που θα έπρεπε να σημειώνουν υψηλές βαθμολογίες. Οι Hijikata et al. (2009) αναφέρουν ότι η μετρική MAE σε σχέση με την RMSE είναι πιο απλή και πιο εύκολη στο να κατανοηθεί και δεύτερον είναι πιο αξιόπιστη τεχνική για την εξέταση της διαφοράς των μέσων απολύτων λαθών ανάμεσα σε δύο συστήματα.

Γενικά, η RMSE είναι μεγαλύτερη η ίση με την MAE. Αν είναι ίση, τότε όλα τα λάθη έχουν το ίδιο μέγεθος. Επίσης μπορεί να γίνει κανονικοποίηση και στις δυο μετρικές σύμφωνα με το εύρος των βαθμολογιών (π.χ. 1 έως 5).

$$\text{normalized RMSE} = \frac{\text{RMSE}}{r_{max} - r_{min}}$$

$$\text{normalized MAE} = \frac{\text{MAE}}{r_{max} - r_{min}}$$

Ανάλογα με την αξιολόγηση, οι μετρικές μπορούν να υπολογισθούν ανά χρήστη ή ανά βαθμολογία

- <u>Κατάταξη αντικειμένων:</u> τα συστήματα συστάσεων κατατάσσουν τις προβλέψεις σύμφωνα με τις προτιμήσεις των χρηστών. Στην αρχή της λίστας παρουσιάζονται τα πιο σημαντικά – σχετικά αντικείμενα και στο τέλος της τα λιγότερο σημαντικά-σχετικά.

Μια πολύ συχνή μετρική για την μέτρηση της σωστής κατάταξης των αντικειμένων είναι το Κανονικοποιημένο Αθροιστικό Κέρδος (normalized Discounted Cumulative Gain – nDCG). Υπολογίζεται πρώτα το Discounted Cumulative Gain (DCG), το οποίο συγκρίνεται με την ιδανική κατάταξη (IDCG). Ο τύπος δίνεται παρακάτω:

Έστω κ το πλήθος των συστάσεων και $\mathrm{rel_i}$ η ομοιότητα μεταξύ της πραγματικής και της προβλεπόμενης βαθμολογίας για την πρόταση i, τότε:

$$\mathrm{DCG_K} = \sum_{\mathrm{I=1}}^{\mathrm{K}} \frac{2^{\mathrm{rel_{i-1}}}}{\log_2(\mathrm{i+1})}, \qquad \mathrm{nDCG_K} = \frac{\mathrm{DCG_K}}{\mathrm{IDCG_K}}$$

Η τιμές που μπορεί να πάρει η συγκεκριμένη μετρική είναι από ένα 1 έως 0, με το 1 να υποδεικνύει την καλύτερη κατάταξη.

- Τέλος, υπάρχουν και οι μετρικές κατηγοριοποίησης: για την αξιολόγηση στο κατά πόσο ένα σύστημα συστάσεων μπορεί να συστήνει εκείνα τα αντικείμενα τα οποία είναι τα πιο ενδιαφέροντα για τον χρήστη. Οι πιο γνωστές μετρικές που χρησιμοποιούνται είναι οι precision, recall, accuracy και false positive rate.

Στην παρούσα αξιολόγηση χρησιμοποιήθηκαν οι μετρικές RMSE, MAE (για κάθε χρήστη και για κάθε βαθμολογία), nDCG και Top-N nDCG, η οποία είναι αντίστοιχη της nDCG με τη διαφορά ότι δίνεται ο αριθμός (N) των συστάσεων για την λίστα στην οποία θα εφαρμοστεί.

3. Συστήματα Συστάσεων βασισμένα σε ετικέτες.

Σήμερα οι χρήστες του Διαδικτύου δεν είναι μόνο καταναλωτές των πληροφοριών, αλλά συμμετέχουν ενεργά στα κοινωνικά δίκτυα και αλληλοεπιδρούν με αυτά. Οι ιστοσελίδες κοινωνικής δικτύωσης βασισμένες σε ετικέτες επιτρέπουν στους χρήστες όχι μόνο να δημοσιεύουν και να επεξεργάζονται πόρους - αντικείμενα (π.χ. τραγούδια, ταινίες, σελιδοδείκτες), αλλά επίσης να δημιουργούν και να προσθέτουν μεταδεδομένα σε μορφή επιλεγμένων λέξεων-κλειδιών που ονομάζονται ετικέτες. Πολύ γνωστά παραδείγματα τέτοιων ιστοσελίδων είναι το Delicious, το BibSonomy, το Flickr και το Last.fm. Αυτές οι ιστοσελίδες είναι εύκολες στη χρήση και δωρεάν σε οποιονδήποτε είναι διατεθειμένος να συμμετάσχει. Μόλις ένας χρήστης είναι συνδεδεμένος, μπορεί να προσθέσει έναν πόρο στο σύστημα, καθώς και να εκχωρήσει αυθαίρετες ετικέτες σε αυτό.

Τα συστήματα συστάσεων είναι τα κατάλληλα εργαλεία για να παρέχουν χρήσιμες και κατάλληλες συστάσεις στα κοινωνικά δίκτυα. Οι ετικέτες στα κοινωνικά δίκτυα βοηθούν σημαντικά στην οργάνωση των δεδομένων. Τα συστήματα συστάσεων βασισμένα σε ετικέτες μπορούν να παρέχουν πληροφορίες σχετικά με το περιεχόμενο των αντικειμένων και των προτιμήσεων του χρήστη και συνεπώς βοηθούν ώστε να παραχθούν καλύτερες εξατομικευμένες συστάσεις (Rawashdeh, Kim, & Mohamad Alj, 2013). Σε αυτά τα συστήματα υπάρχουν τρείς σημαντικοί τύποι: οι χρήστες (users), οι ετικέτες (tags) και οι πόροι-αντικείμενα (resources-items).

Οι περισσότερες έρευνες που γίνονται σε αυτόν τον τομέα, επικεντρώνονται κυρίως στην αντιμετώπιση του προβλήματος σύστασης για νέους χρήστες (Cold-start problem: όταν υπάρχουν πολύ λίγες πληροφορίες για έναν χρήστη, το σύστημα αδυνατεί να εξάγει συμπεράσματα, συνεπώς δεν γίνεται σωστή σύσταση αντικειμένων σε αυτούς), την σποραδικότητα των δεδομένων (παραγωγή μη αξιόπιστων δεδομένων, λόγω αραιού πίνακα χρηστών-αντικειμένων) και την βελτίωση της ακρίβειας (δηλαδή κατά πόσο αξιόπιστες είναι οι συστάσεις που παράγονται από το σύστημα).

3.1 Συναφείς εργασίες

Στο άρθρο τους οι Lee et al., (2011) παρουσιάζουν μια τεχνική για τη σύσταση αντικειμένων μέσα σε κοινωνικά δίκτυα που να ταιριάζει με τα ενδιαφέροντα των χρηστών και της ομάδας με την πάροδο του χρόνου. Σε αυτό το σύστημα οι χρήστες και οι ετικέτες, που συνδέονται με ένα αντικείμενο, αντιπροσωπεύονται και ομαδοποιούνται σε θεματικά σύνολα με χρονολογική σειρά. Με το μοντέλο τους, το οποίο συνδυάζει δυο παραλλαγές του προτύπου της λανθάνουσας κατανομής του Dirichlet (Latent Dirichlet Allocation, LDA), πετύχαν μεγαλύτερη ακρίβεια στις συστάσεις. Πιο συγκεκριμένα, εφαρμόζοντας το σύστημα τους σε ένα σετ δεδομένων από το Flickr προέκυψε ότι η μέση ακρίβεια ήταν αρκετά υψηλή (περίπου 70%).

Χρησιμοποιώντας την Latent Dirichlet Allocation (LDA) προσέγγιση οι Zhang et al., (2012) δημιουργήσανε ένα μοντέλο ($(CRS)^2$) το οποίο συνδυάζει το περιεχόμενο και την σχέση των χρηστών, των ετικετών και των αντικειμένων. Στο πείραμα που πραγματοποίησαν οι συγγραφείς του άρθρου συγκρίθηκε το προτεινόμενο μοντέλο ($(CRS)^2$)) με τους HOSVD και RLDA (τροποποιημένη έκδοση του ($(CRS)^2$) που αγνοεί το περιεχόμενο των αντικειμένων). Τα δεδομένα του πειράματος αντλήθηκαν από το CiteULike και τα αποτελέσματα ήταν πολύ θετικά για τον ($(CRS)^2$). Η επίδοση του ήταν καλύτερη σε σχέση με τους άλλους αλγορίθμους και μπόρεσε να αντιμετωπίσει το πρόβλημα της σποραδικότητας των δεδομένων κυρίως στην σύσταση αντικειμένων και ετικετών.

Οι Wang, et al., (2010) σχεδιάσανε δυο αλγόριθμους οι οποίοι βασίζονται στο Random Walk with Restarts framework (RWR). Σύμφωνα με το πρώτο (User-ResourceTag-basedAlgorithm | RWUR), οι ετικέτες θεωρούνται ως μια άλλη μορφή βαθμολογίας (ratings), ενώ στον δεύτερο (User-User Tag-basedAlgorithm | RWUU) οι ομοιότητες μεταξύ των χρηστών βασίζονται στην πληροφορία των ετικετών τους. Δοκίμασαν τους αλγορίθμους τους σε ένα πραγματικό σετ δεδομένων «ταινίες-βαθμολογίες» (MovieLens) και κατέληξαν ότι ο πρώτος (RWUR) είχε μικρότερη επίδοση από τον βασικό αλγόριθμο (RWR), ενώ ο δεύτερος (RWUU) είχε μεγαλύτερη επίδοση και δεν παρουσίασε κανένα πρόβλημα σε αραιά σύνολα δεδομένων.

Οι Durao & Dolog, (2012) στο μοντέλο τους χρησιμοποιούν την Ομοιότητα Συνημιτόνου (Cosine Similarity) για να υπολογίσουν την ομοιότητα των ετικετών σε συνδυασμό με την δημοτικότητα της ετικέτας (δηλαδή το βάρος της ετικέτας), την αντιπροσωπευτικότητα της ετικέτας (δηλαδή την συχνότητα εμφάνισης της | term frequency) και την σχέση ετικέτας-χρήστη (δηλαδή πόσο συχνά χρησιμοποιείται μια ετικέτα από έναν χρήστη) με σκοπό να παράγουν πιο εξατομικευμένες συστάσεις στους χρήστες. Τα δεδομένα στα οποία εξετάστηκε το μοντέλο τους είχαν συλλεχθεί από το Delicious. Σύμφωνα με τα αποτελέσματα, το 60% των συστάσεων ήταν επιτυχημένο, αλλά δυστυχώς δεν μπόρεσαν επιλύσουν τα προβλήματα της ασάφειας των ετικετών και της σποραδικότητας των δεδομένων.

Οι Yuan et al., (2011) προτείνουν έναν νέο collaborative filtering αλγόριθμο (Social Tag-based CF) που βασίζεται στη χρήση ετικετών για τα βιβλία και στην ανίχνευση των χρηστών που έχουν παρόμοιο προφίλ με τον υποψήφιο, με σκοπό να αντιμετωπιστούν αποτελεσματικά τα προβλήματα που είναι σχετικά με τη σύσταση νέων χρηστών και τη σποραδικότητα των δεδομένων. Πιο συγκεκριμένα, αρχικά γίνεται ανίχνευση των χρηστών που έχουν παρόμοιο προφίλ με τον υποψήφιο, από τις κοινότητες των κοινωνικών δικτύων με τη βοήθεια του Q-based (modularity) GN (Girvan and Newton) αλγορίθμου. Στη συνέχεια δημιουργείται ένα τριπλό μοντέλο συσχετίσεων μεταξύ βιβλίων – ετικετών – χρηστών από το οποίο προκύπτουν οι καλύτερες υποψήφιες ετικέτες. Τέλος, εφαρμόζεται ο αλγόριθμος κατηγοριοποίησης Naïve Bayes για την επιλογή των προτεινόμενων βιβλίων, που απορρέουν από τις υποψήφιες ετικέτες. Χρησιμοποιώντας ένα σετ δεδομένων από το douban.com, συγκρίνουν τον συγκεκριμένο αλγόριθμο με τους content based recommendation, item-based CF και user-based CF αλγορίθμους. Σύμφωνα με τα αποτελέσματα, ο προτεινόμενος αλγόριθμος αποδίδει καλύτερα από τους υπόλοιπους αλγορίθμους αλλά υστερεί στην απόκριση των αποτελεσμάτων.

Τα κοινωνικά δίκτυα μπορούν να διαδραματίσουν σημαντικό ρόλο στα συστήματα συστάσεων. Σύμφωνα με τους ερευνητές (Pan, et al., 2012) οι ετικέτες μπορούν να αποτελέσουν χρήσιμα κλειδιά για τον εντοπισμό των προτιμήσεων των χρηστών. Πρώτον, οι ετικέτες οι οποίες προστίθενται σε ένα συγκεκριμένο αντικείμενο μπορεί να θεωρηθούν ως ένα αφηρημένο περιεχόμενο αυτού του στοιχείου. Δεύτερον, ακόμη και για το ίδιο αντικείμενο, διαφορετικός χρήστης μπορεί να αναθέσει εντελώς διαφορετικές ετικέτες, συνεπώς, οι προσωπικές προτιμήσεις είναι φυσικά ενσωματωμένες στις διάφορες χρήσεις των ετικετών.

Κλείνοντας, όπως φαίνεται και από τις παραπάνω έρευνες, η αξιοποίηση των ετικετών βοηθούν τα μοντέλα συστημάτων συστάσεων να γίνουν πιο αποδοτικά, παρόλα αυτά το πρόβλημα σύστασης για νέους χρήστες και η μείωση της σποραδικότητας των δεδομένων αποτελούν μια μεγάλη πρόκληση για την ερευνητική κοινότητα.

4. Προτεινόμενη προσέγγιση για την δημιουργία εμπλουτισμένου γράφου.

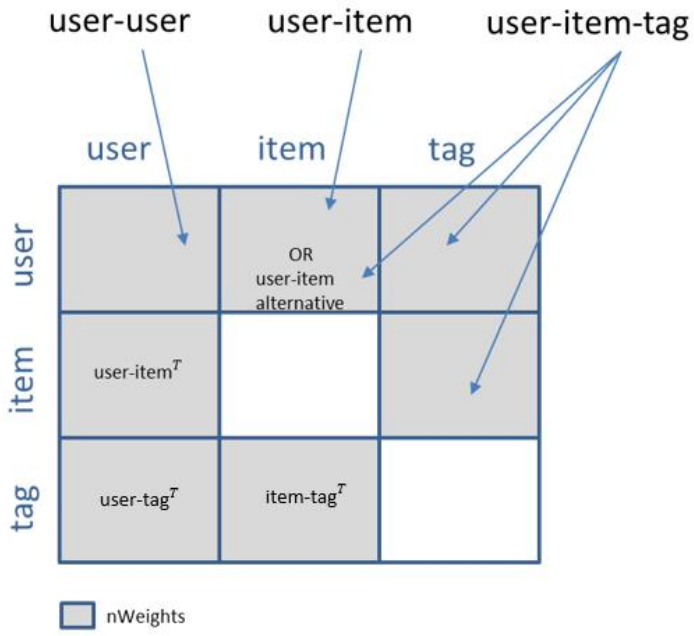
4.1 Περιγραφή

Ο σκοπός του εργαλείου είναι να δημιουργηθεί ένας εμπλουτισμένος γράφος από τρία είδη γράφων. Δηλαδή, να αξιοποιηθεί η πληροφορία που εμπεριέχεται στους γράφους και να δημιουργηθούν νέες σχέσεις που θα αποτυπώνονται στον εμπλουτισμένο γράφο. Στο εργαλείο μπορούν να δοθούν μέχρι τρία είδη γράφων «αφετηρίας»:

- ενός μονομερή (unipartite), που θα περιέχει τις ακμές φιλίας μεταξύ των χρηστών | user-user,
- ενός διμερή (bipartite), με τις ακμές αξιολόγησης των αντικειμένων από τους χρήστες | user-item και
- ενός τριμερή (tripartite), με την πληροφορία ακμών μεταξύ των τριών ανεξάρτητων συνόλων χρήστες, αντικείμενα, ετικέτες | user-item-tag.

Παρόλα αυτά, για να επιτευχθεί η δημιουργία του τελικού γράφου θα πρέπει να δίνεται από τον χρήστη τουλάχιστον ένας τριμερής γράφος.

Αν μπορούσαμε να το αναπαραστήσουμε σαν έναν εμπλουτισμένο πίνακα, θα είχε την παρακάτω μορφή:



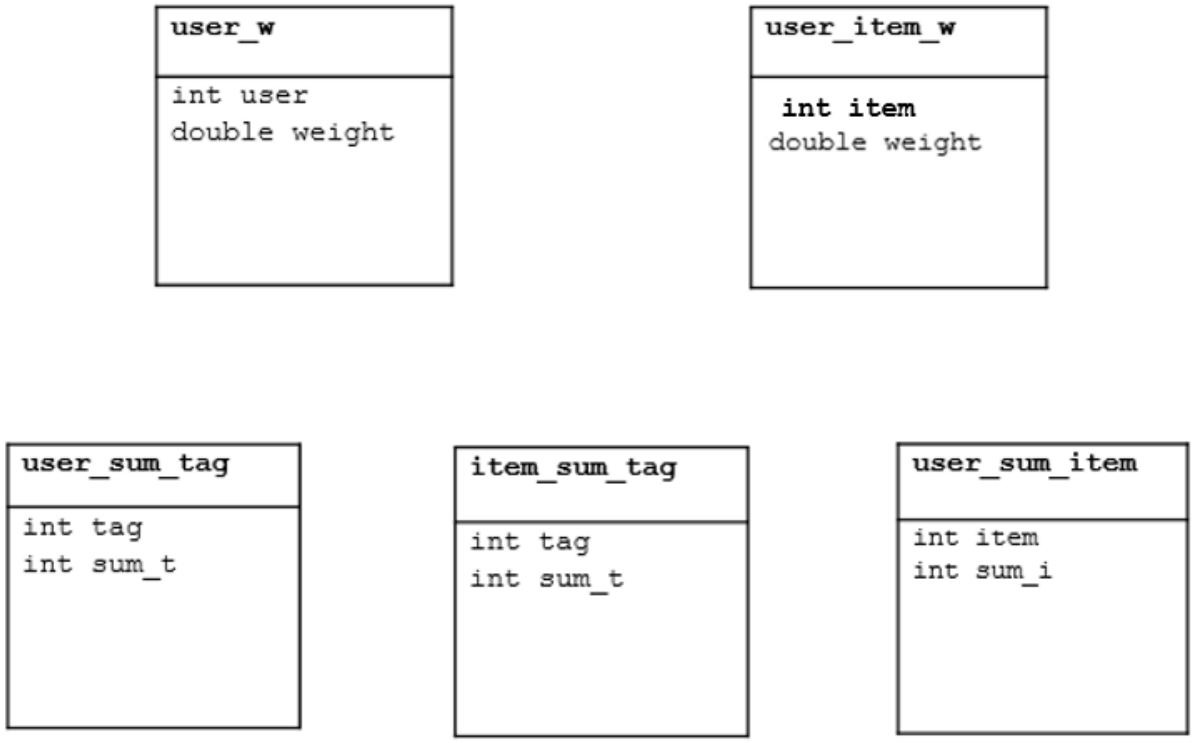
Εικόνα 4.1 αναπαράσταση εμπλουτισμένου πίνακα

Ουσιαστικά, εκμεταλλευόμαστε την πληροφορία που δίνεται από τους τρεις γράφους και δημιουργούμε βάρη (nWeights, normalized weights) για τις υπάρχουσες και τις καινούργιες σχέσεις.

Οι περιπτώσεις που λαμβάνει υπόψη του το σύστημα είναι οι εξής:

- Αν υπάρχει ο user-user γράφος τότε δημιουργείται και ο αντίστοιχος user-user
- Αν υπάρχει ο user-item γράφος τότε δημιουργείται και ο αντίστοιχος user-item (η διαφορά με τον αρχικό είναι ότι τα ratings είναι κανονικοποιημένα), αν δεν δίνεται τότε δημιουργείται ο user-item alternative από τον user-item-tag
- Τέλος, από τον user-item-tag δημιουργείται ο user-tag, ο item-tag και οι αντίστροφοι τους tag-user και tag-item.

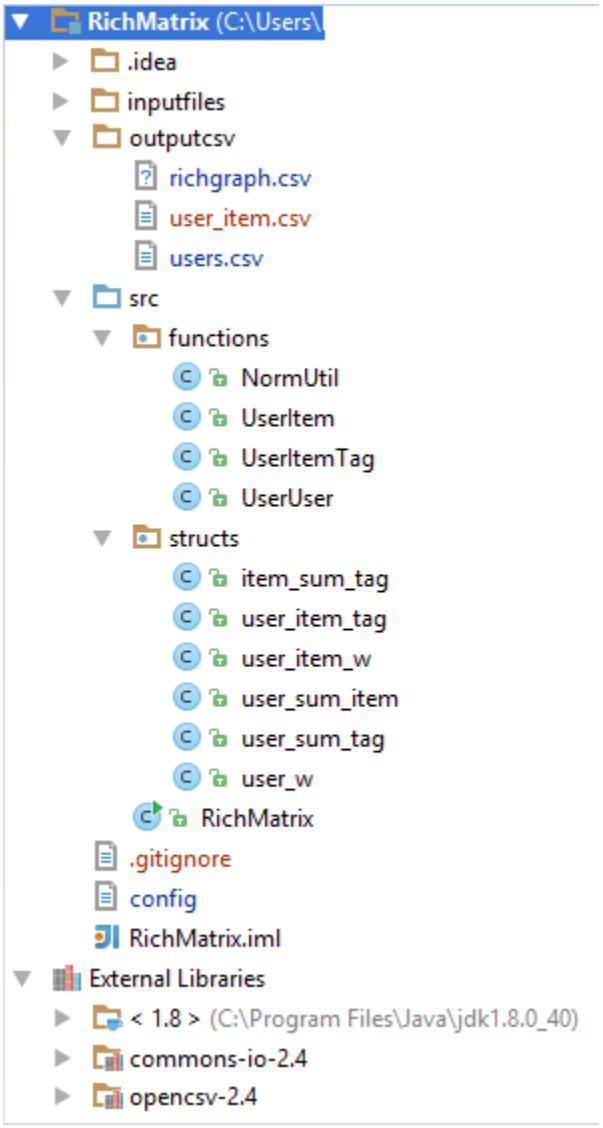
Ο κώδικας του εργαλείου είναι γραμμένος σε Java και χρησιμοποιήθηκε το προγραμματιστικό περιβάλλον (IDE) InteliJ[3] (Community Edition). Για την υλοποίηση του εργαλείου χρησιμοποιήθηκε η δομή treeMap <K,V>[4], όπου K (integer): τα ids των κόμβων (userid ή itemid ή tagid) και V: τα arraylists από αντικείμενα με παραμέτρους από τις αντίστοιχες κλάσεις:



Εικόνα 4.2 κλάσεις του εργαλείου

[3] https://www.jetbrains.com/idea/
[4] https://docs.oracle.com/javase/7/docs/api/java/util/TreeMap.html

Η δομή του προγράμματος όπως φαίνεται από το IDE:



Εικόνα 4.3 δομή του εργαλείου

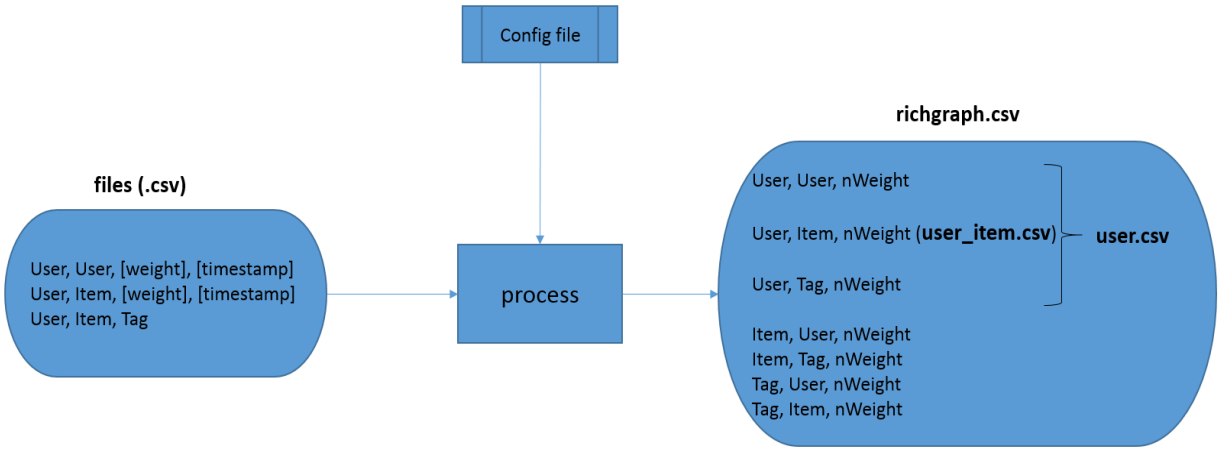
Στην κλάση UserUser γίνεται η ανάγνωση του αρχείου (.csv ή .dat) που περιέχει τον γράφο user-user, ελέγχει αν δίνεται το βάρος ή η χρονοσφραγίδα και δημιουργείται το treemap με κλειδί το id του user και value ένα arraylist από αντικείμενα της κλάσης user_w. Αντιστοίχως και η UserItem, όπου το arraylist περιέχει αντικείμενα της κλάσης user_item_w. Τέλος, στην κλάση UserItemTag διαβάζεται το αρχείο που περιέχει τον γράφο user-item-tag και δημιουργεί τρία treemaps: το UserTag, το οποίο αναπαριστά τον γράφο user-tag με βάρη το άθροισμα των ετικετών για κάθε χρήστη, το ItemTag, το οποίο αναπαριστά τον γράφο item-tag με βάρη το άθροισμα των ετικετών για κάθε αντικείμενο και το UserItemAlter, το οποίο αναπαριστά τον γράφο user-item alternative με βάρη το άθροισμα των αντικειμένων, στα οποία έχει προσθέσει ετικέτες ο κάθε χρήστης.

Η κλάση RichMatrix περιέχει την main μέθοδο η οποία καλεί τις κλάσεις του πακέτου function και δημιουργεί τα εξής τρία αρχεία (.csv) με delimeter (;), τα οποία αποθηκεύονται στον φάκελο outputcsv:

user_item.csv: η πρώτη στήλη περιέχει τα ids των χρηστών, η δεύτερη τα ids των αντικειμένων και η τρίτη τα βάρη.

users.csv: η πρώτη στήλη περιέχει τα ids των χρηστών, η δεύτερη περιέχει, με την εξής σειρά τα ids των χρηστών (friends ids), τα ids των αντικειμένων (item ids) και τα ids των ετικετών (tag ids) και η τρίτη τα αντίστοιχα βάρη.

richgraph.csv: αρχίζει με τους χρήστες και εμπεριέχει το users.csv, δηλαδή user-user, user-item, user-tag και συνεχίζει με τα αντικείμενα, δηλαδή item-user (transpose user-item) και item-tag και τελειώνει με τις ετικέτες, δηλαδή tag-user ((transpose user-tag)) και tag-item (transpose item-tag).



Εικόνα 4.4 δημιουργία εμπλουτισμένου γράφου από την εισαγωγή τριών ξεχωριστών γράφων

Στα τρία .csv αρχεία που προκύπτουν, εκτελούνται οι δυο CF αλγόριθμοι και ο FunkSVD και αξιολογούνται τα αποτελέσματα (βλ. κεφ. 6.Πειραματική αξιολόγηση).

Το εργαλείο μπορεί να δεχθεί csv ή dat αρχεία με συγκεκριμένη όμως δομή.

Η δομή που πρέπει να έχουν τα αρχεία των γράφων, είναι η εξής:

Unipartite: one-edge-per-line (from, to, [weight], [timestamp]) όπου τα from & to είναι υποχρεωτικά και τα weight & timestamp προαιρετικά.

Bipartite: one-edge-per-line (user, item, [weight], [timestamp]) όπου τα user & item είναι υποχρεωτικά και τα weight & timestamp προαιρετικά.

Tripartite: one-edge-per-line (user, item, tag)

Για την προ επεξεργασία των γράφων, οι παράμετροι του processing διαβάζονται από ένα configuration αρχείο, στο οποίο ξεχωρίζουν οι παράμετροι του κάθε γράφου. Ουσιαστικά θα

πρέπει ο χρήστης να συμπληρώνει στο configuration αρχείο μια σειρά παραμέτρων οι οποίες περιγράφουν την δομή του γράφου και του preprocessing (Εικόνα 4.4).

Συγκεκριμένα, για τον unipartite γράφο και τον bipartite γράφο θα πρέπει να ορίζονται τα εξής:

- Αν υπάρχει το αρχείο.
- Η διαδρομή του αρχείου.
- Το delimeter (π.χ. \t, ;, κ.λπ.).
- Ορισμός στηλών, δηλαδή τι βρίσκεται στην κάθε στήλη (π.χ. αν στην πρώτη στήλη βρίσκονται τα id των users, τότε userIDcolumn=0).
- Αν δίνεται timestamp και αν ναι θα πρέπει να εισαχθεί το μεγαλύτερο.
- Αν δίνεται βάρος.

Ενώ για τον τριμερή γράφο θα πρέπει να ορίζονται τα εξής:

- Η διαδρομή του αρχείου.
- Το delimeter.
- Ορισμός στηλών.

Τέλος, θα πρέπει να ορίζεται ο αριθμός από τον οποίο θα ξεκινάνε τα ids των αντικειμένων και των ετικετών. Επίσης, όσο αναφορά την κανονικοποίηση θα πρέπει να ορίζεται το εύρος και σε ποιο δεκαδικό θα γίνεται η στρογγυλοποίηση.

Ακολουθεί ένα παράδειγμα του config αρχείου για το σύνολο δεδομένων του lastfm, το οποίο περιλαμβάνει και τους τρεις γράφους (βλ. Πίνακας 6.1). Συνεπώς, το file1exist, file2exist θα πάρουν την τιμή 1, το timestamp και το givenweight δεν θα πάρουν τιμή, καθώς για τον γράφο user-user δεν εμπεριέχεται ούτε βάρος, ούτε χρονοσφραγίδα. Ενώ για τον user-item, στον οποίο εμπεριέχονται αυτά τα δυο πεδία, το timestamp2 και το givenweight2 θα πάρουν και θα δοθεί και η τελευταία χρονοσφραγίδα στο last_timestamp2. Όσο αναφορά την κανονικοποίηση, το εύρος των παραγόμενων τιμών θα είναι από 1 έως 5 με στρογγυλοποίηση στο πρώτο δεκαδικό (#.#). Τέλος, τα id των αντικειμένων θα ξεκινούν από 100000, ενώ των ετικετών από 1000000. Αυτό χρειάζεται για να μπορούμε να τα ξεχωρίζουμε.

```
################################ config file ######################

################ user-user ##################
#put 1: if user-user file exist
file1exist=1

#pathfile1
path1=inputfiles/lastfm_user_friends.dat

#delimeter
split=\t

#0,1,2,3 columns
userIDcolumn=0
friendIDcolumn=1
```

```
weightcolumn=2
timestampcolumn=

#put 1: if timestamp exists
timestamp=
last_timestamp=

#put 1: if weight exist
givenweight=


################ user-item ##################
#put 1: if user-item file exist
file2exist=1


path2=inputfiles/lastfm_user_artists.dat

#delimeter
split2=\t

#0,1,2,3 columns
user2IDcolumn=0
itemIDcolumn=1
weight2column=2
timestamp2column=3

#put 1: if timestamp exists
timestamp2=1
last_timestamp2=1231129940000


#put 1: if weight exist
givenweight2=1


################# user-item-tag ##############

path3=inputfiles/lastfm_user_taggedartists.dat

#delimeter
split3=\t

#0,1,2 columns
user3IDcolumn=0
item2IDcolumn=1
tagIDcolumn=2


############normalization & rounding | positive numbers##########
normalizedLow=1
normalizedHigh=5
rounded=#.#

########## richmatrix############
#from where to start the ids, for lastfm and movielens: items=100000,
tags=1000000 | for delicious: items=1000000, tags=10000000
```

```
number_of_items=100000
number_of_tags =1000000

#############################################################################
```

Εικόνα 4.5 παράδειγμα Config αρχείου για το lastfm

Ο αλγόριθμος για την δημιουργία του νέου βάρους έχει ως εξής:

Αλγόριθμος δημιουργίας nWeight

Input: from, to , [weight], [timestamp] (user-user or user-item files)

Output: from ; to ; nWeight

if (timestamp)

  if (weight)

    then nWeight = $\frac{\text{timestamp}}{\text{last timestamp}}$ * weight

    else nWeight = $\frac{\text{timestamp}}{\text{last timestamp}}$

else

  if (weight)

    then nWeight = weight

    else nWeight = 1

Εικόνα 4.6 αλγόριθμος δημιουργίας nWeight

Κανονικοποίηση τιμών

Αφού δημιουργηθούν τα βάρη στη συνέχεια κανονικοποιούνται. Σκοπός της κανονικοποίησης είναι η κατηγοριοποίηση των δεδομένων με σκοπό την ένταξη τους σε συγκεκριμένο εύρος τιμών. Για την συγκεκριμένη προσέγγιση χρησιμοποιήθηκε η min-max κανονικοποίηση, η οποία εφαρμόζει γραμμικό μετασχηματισμό στα δεδομένα. Έστω ότι minX και maxX είναι η ελάχιστη και μέγιστη τιμή του γνωρίσματος X. Η min-max κανονικοποίηση αντιστοιχίζει την τιμή του γνωρίσματος X με την τιμή ν΄ στο νέο διάστημα [norm.minX, norm.maxX] σύμφωνα με την σχέση:

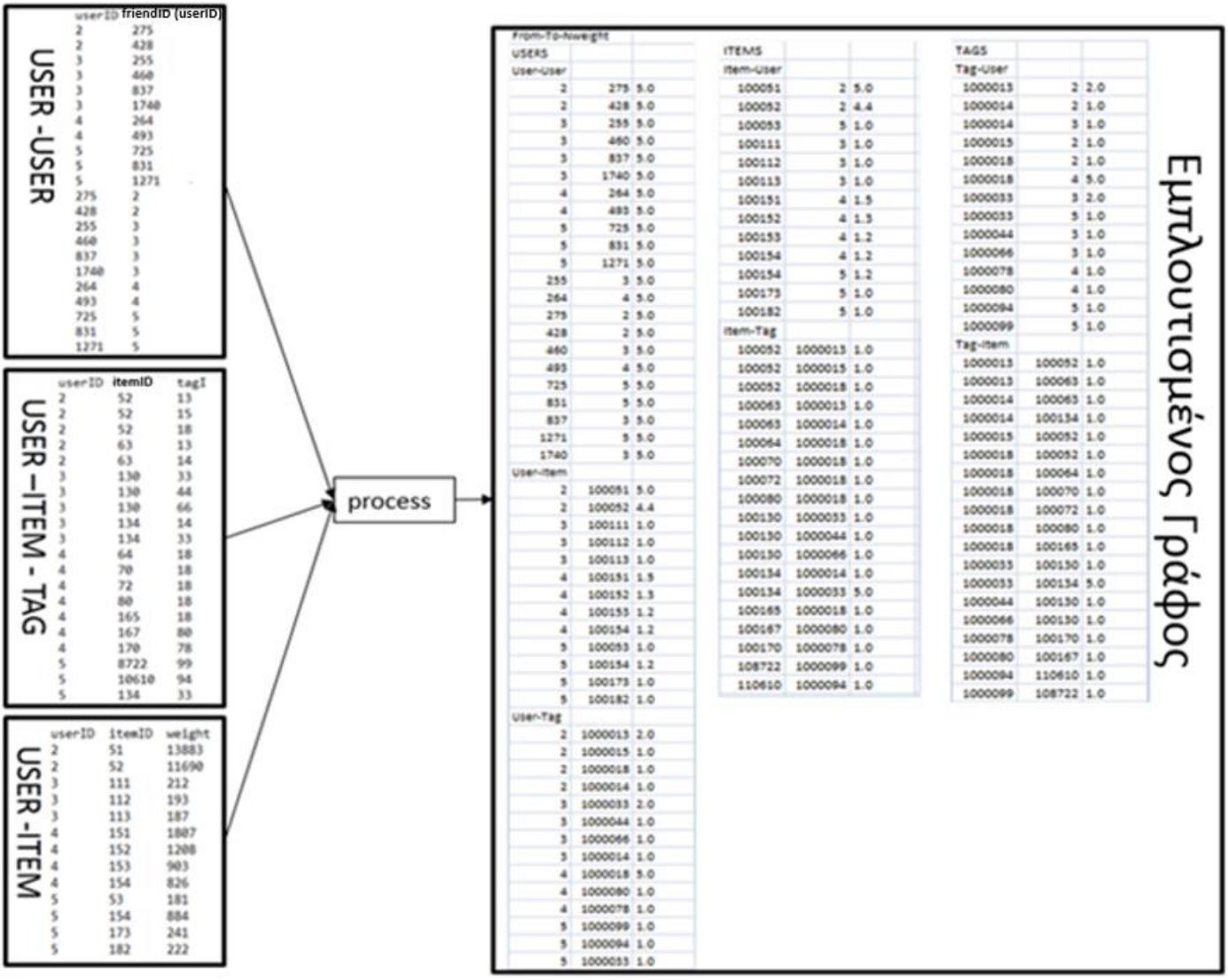
$$\nu' = \frac{(\nu - \text{minX})(\text{norm.maxX} - \text{norm.minX})}{(\text{maxX} - \text{minX})} + \text{norm.minX}$$

Για παράδειγμα, αν ν = 3 στο διάστημα [1, 34], τότε για το διάστημα [1, 5], το ν΄ = 1.24.

Ο χρήστης μπορεί από το config αρχείο να δώσει το εύρος των τιμών που θα γίνει η κανονικοποίηση, καθώς επίσης και σε ποιο δεκαδικό θα γίνει η στρογγυλοποίηση (βλ. rounded=#.#, Εικόνα 4.5).

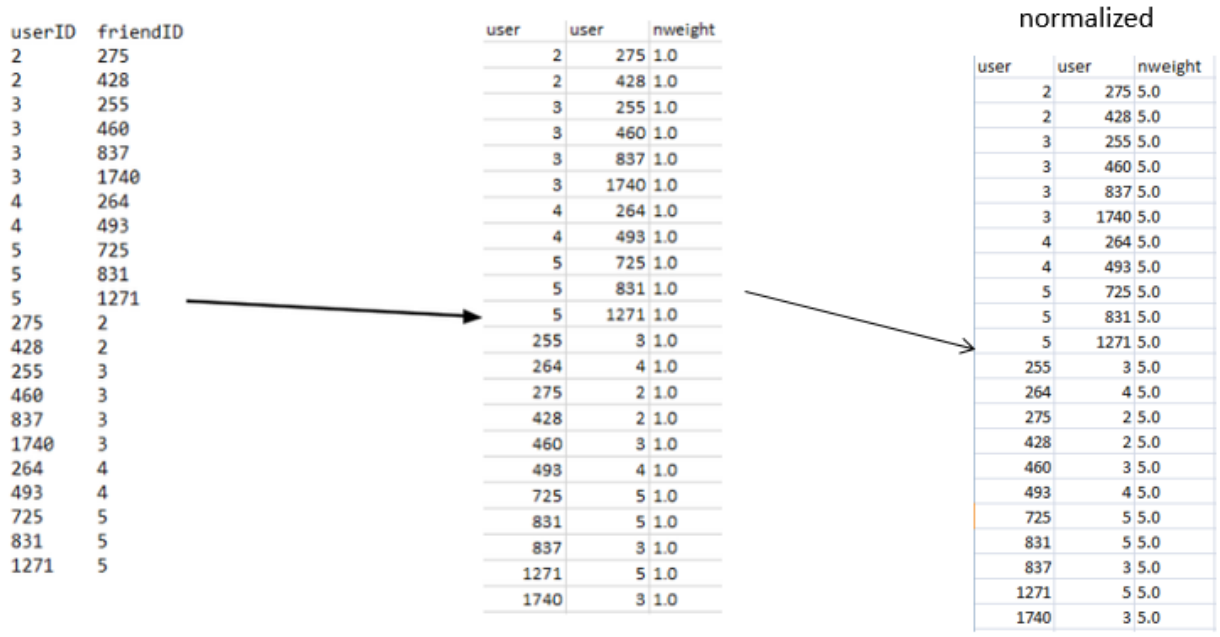
4.2 Παράδειγμα

Στην συνέχεια παρουσιάζεται ένα παράδειγμα δημιουργίας ενός εμπλουτισμένου γράφου. Το παράδειγμα προέρχεται από το σύνολο δεδομένων του lastfm. Όπως φαίνεται και από την εικόνα με την βοήθεια των τριών διαφορετικών γράφων δημιουργείται ο εμπλουτισμένος γράφος.



Εικόνα 4.7 παράδειγμα εμπλουτισμένου γράφου

Πιο συγκεκριμένα, ο γράφος user – user αναπαριστά την φιλία μεταξύ των χρηστών (μη κατευθυνόμενος γράφος). Για παράδειγμα, όπως φαίνεται και από την παρακάτω εικόνα, ο χρήστης 2 είναι φίλος με τον χρήστη 275 και 428, κλπ.. Εφόσον, δεν δίνεται κάποιο βάρος ή χρονοσφραγίδα, το βάρος του χρήστη 2 με τον 275 και τον 428 θα είναι 1, το οποίο με την κανονικοποίηση θα γίνει 5 στον τελικό γράφο.



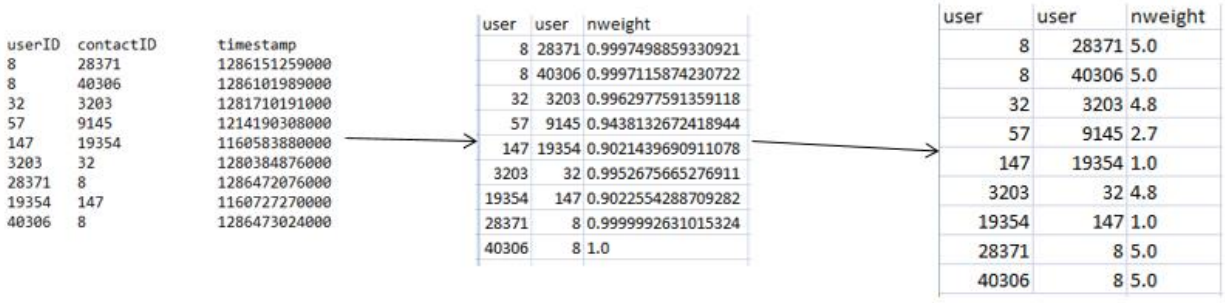
userID	friendID
2	275
2	428
3	255
3	460
3	837
3	1740
4	264
4	493
5	725
5	831
5	1271
275	2
428	2
255	3
460	3
837	3
1740	3
264	4
493	4
725	5
831	5
1271	5

user	user	nweight
2	275	1.0
2	428	1.0
3	255	1.0
3	460	1.0
3	837	1.0
3	1740	1.0
4	264	1.0
4	493	1.0
5	725	1.0
5	831	1.0
5	1271	1.0
255	3	1.0
264	4	1.0
275	2	1.0
428	2	1.0
460	3	1.0
493	4	1.0
725	5	1.0
831	5	1.0
837	3	1.0
1271	5	1.0
1740	3	1.0

user	user	nweight
2	275	5.0
2	428	5.0
3	255	5.0
3	460	5.0
3	837	5.0
3	1740	5.0
4	264	5.0
4	493	5.0
5	725	5.0
5	831	5.0
5	1271	5.0
255	3	5.0
264	4	5.0
275	2	5.0
428	2	5.0
460	3	5.0
493	4	5.0
725	5	5.0
831	5	5.0
837	3	5.0
1271	5	5.0
1740	3	5.0

Εικόνα 4.8 παράδειγμα user-user γράφου

Σε άλλο παράδειγμα, όπως στο delicious, ο γράφος USER – USER, ο οποίος δείχνει πότε ένας χρήσης έκανε follow έναν άλλον χρήστη, περιέχει και χρονοσφραγίδες, συνεπώς, όπως φαίνεται και στο παρακάτω σχήμα, ο χρήστης 8 έκανε follow τον χρήστη 28371 με χρονοσφραγίδα 1286151259000 (unix time) και αντιστρόφως ο χρήστης 28371 με τον 8 με χρονοσφραγίδα 1286472076000. Το βάρος για τους χρήστες προκύπτει διαιρώντας την κάθε χρονοσφραγίδα με την μεγαλύτερη χρονοσφραγίδα. Άρα στο συγκεκριμένο παράδειγμα το βάρος που θα προκύψει μεταξύ αυτών των δυο χρηστών, μετά την κανονικοποίηση θα είναι 5.0.



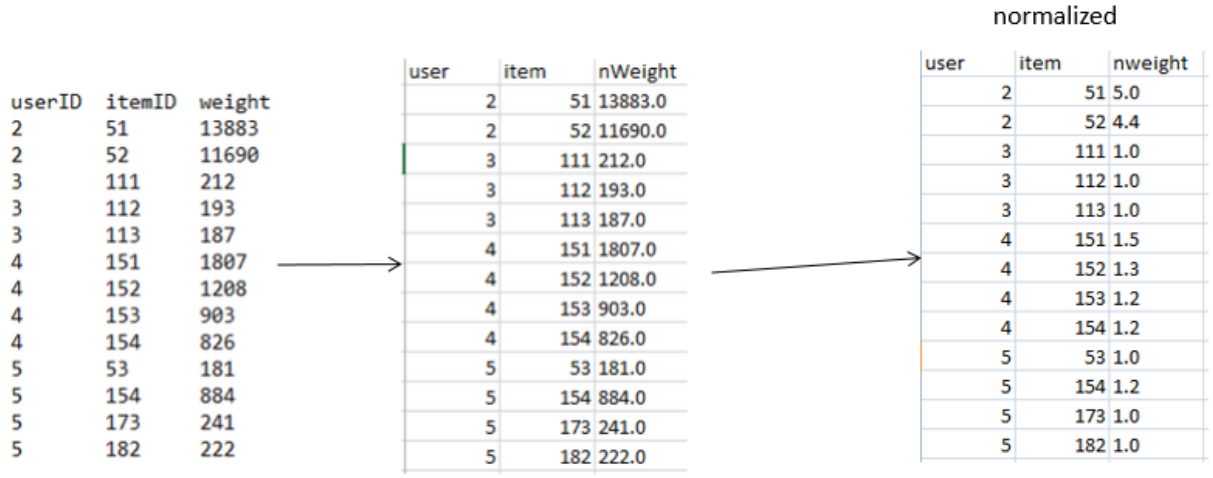
userID	contactID	timestamp
8	28371	1286151259000
8	40306	1286101989000
32	3203	1281710191000
57	9145	1214190308000
147	19354	1160583880000
3203	32	1280384876000
28371	8	1286472076000
19354	147	1160727270000
40306	8	1286473024000

user	user	nweight
8	28371	0.9997498859330921
8	40306	0.9997115874230722
32	3203	0.9962977591359118
57	9145	0.9438132672418944
147	19354	0.9021439690911078
3203	32	0.9952675665276911
19354	147	0.9022554288709282
28371	8	0.9999992631015324
40306	8	1.0

user	user	nweight
8	28371	5.0
8	40306	5.0
32	3203	4.8
57	9145	2.7
147	19354	1.0
3203	32	4.8
19354	147	1.0
28371	8	5.0
40306	8	5.0

Εικόνα 4.9 παράδειγμα user-user γράφου με χρονοσφραγίδα

Στην συνέχεια ο γράφος USER – ITEM αναπαριστά την σχέση μεταξύ των χρηστών και τον αντικειμένων. Το παρακάτω παράδειγμα προέρχεται από το σύνολο δεδομένων του lastfm και δείχνει πόσες φορές ένας χρήστης έχει ακούσει τον συγκεκριμένο καλλιτέχνη. Ο χρήστης 2 άκουσε τον καλλιτέχνη 51 13883 φορές, ενώ τον καλλιτέχνη 52 11690 φορές. Εφόσον δεν υπάρχει χρονοσφραγίδα, το βάρος δεν αλλάζει, συνεπώς μετά την κανονικοποίηση το βάρος του χρήστη 2 με τον καλλιτέχνη 51 θα είναι 5, ενώ με τον 52 θα είναι 4.4.



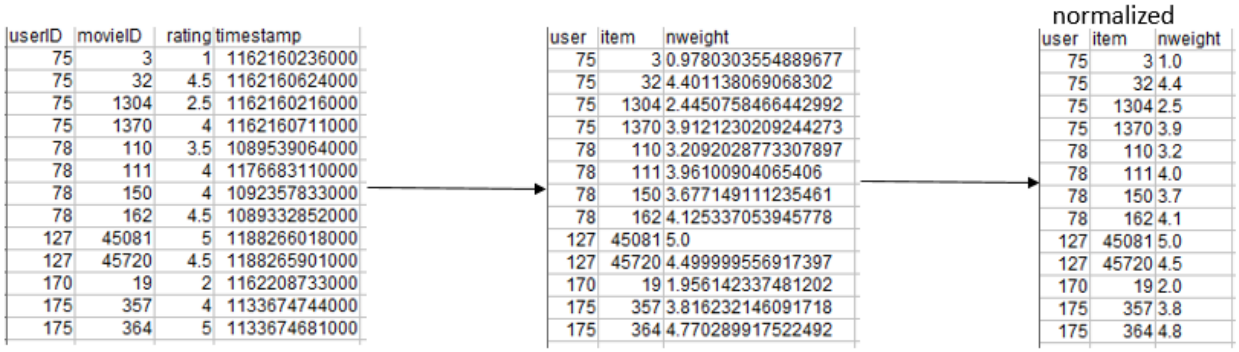
userID	itemID	weight
2	51	13883
2	52	11690
3	111	212
3	112	193
3	113	187
4	151	1807
4	152	1208
4	153	903
4	154	826
5	53	181
5	154	884
5	173	241
5	182	222

user	item	nWeight
2	51	13883.0
2	52	11690.0
3	111	212.0
3	112	193.0
3	113	187.0
4	151	1807.0
4	152	1208.0
4	153	903.0
4	154	826.0
5	53	181.0
5	154	884.0
5	173	241.0
5	182	222.0

user	item	nweight
2	51	5.0
2	52	4.4
3	111	1.0
3	112	1.0
3	113	1.0
4	151	1.5
4	152	1.3
4	153	1.2
4	154	1.2
5	53	1.0
5	154	1.2
5	173	1.0
5	182	1.0

Εικόνα 4.10 παράδειγμα user-item γράφου

Σε ένα αντίστοιχο παράδειγμα με χρονοσφραγίδες (π.χ. movielens), το τελικό βάρος προκύπτει εφόσον πολλαπλασιαστεί η αρχική αξιολόγηση με το βάρος των χρονοσφραγίδων ($\frac{\text{timestamp}}{\text{last\_timestamp}}$)



userID	movieID	rating	timestamp
75	3	1	1162160236000
75	32	4.5	1162160624000
75	1304	2.5	1162160216000
75	1370	4	1162160711000
78	110	3.5	1089539064000
78	111	4	1176683110000
78	150	4	1092357833000
78	162	4.5	1089332852000
127	45081	5	1188266018000
127	45720	4.5	1188265901000
170	19	2	1162208733000
175	357	4	1133674744000
175	364	5	1133674681000

user	item	nweight
75	3	0.9780303554889677
75	32	4.401138069068302
75	1304	2.4450758466442992
75	1370	3.9121230209244273
78	110	3.2092028773307897
78	111	3.96100904065406
78	150	3.677149111235461
78	162	4.125337053945778
127	45081	5.0
127	45720	4.499999556917397
170	19	1.956142337481202
175	357	3.816232146091718
175	364	4.770289917522492

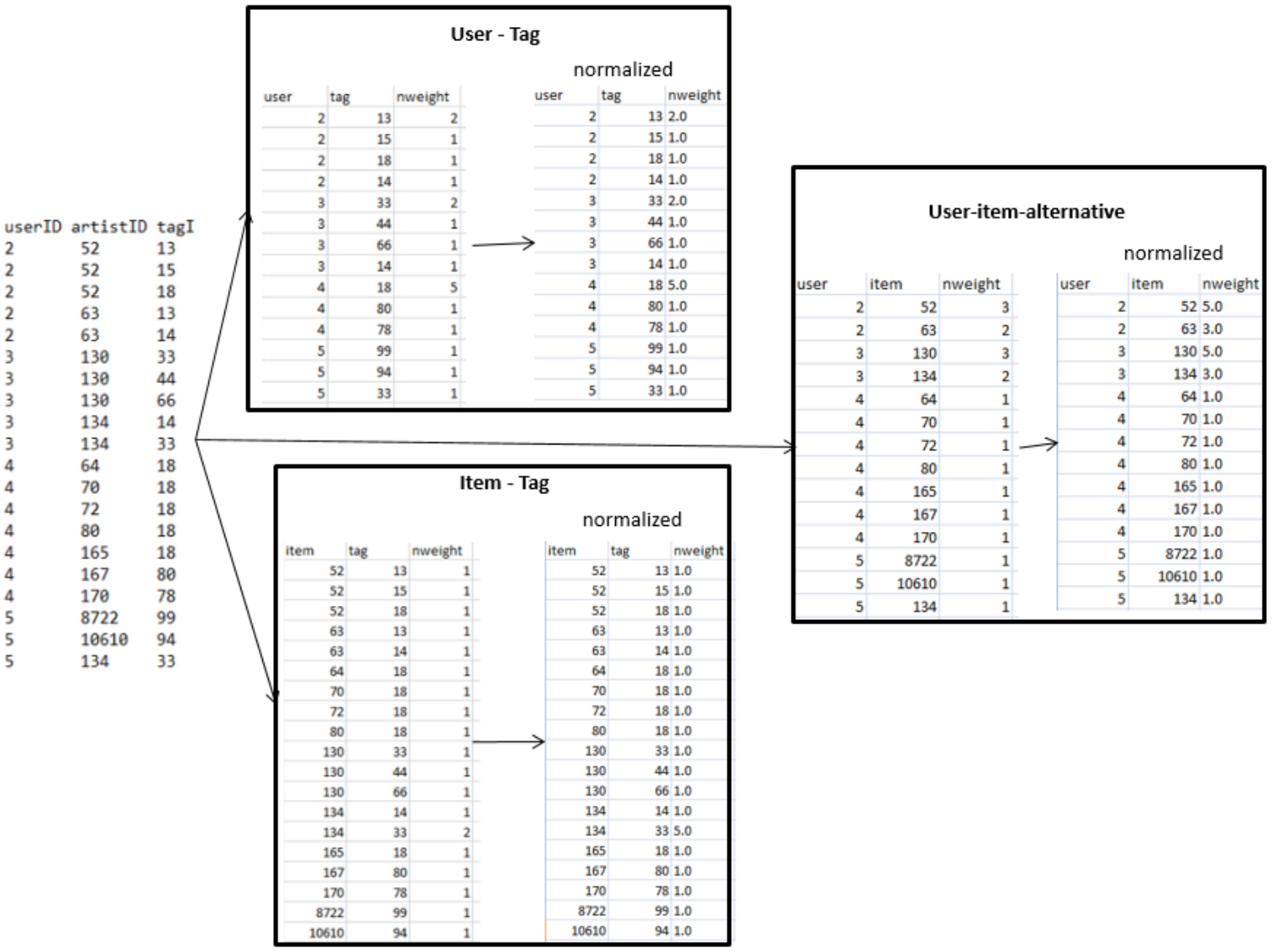
user	item	nweight
75	3	1.0
75	32	4.4
75	1304	2.5
75	1370	3.9
78	110	3.2
78	111	4.0
78	150	3.7
78	162	4.1
127	45081	5.0
127	45720	4.5
170	19	2.0
175	357	3.8
175	364	4.8

Εικόνα 4.11 παράδειγμα user-item γράφου με χρονοσφραγίδα

Τέλος, από τον γράφο USER – ITEM – TAG, ο οποίος για το σύνολο δεδομένων του lastfm δείχνει την ετικέτα που έχει προσθέσει ο κάθε χρήστης σε κάθε καλλιτέχνη, μπορούν να προκύψουν τρεις άλλοι γράφοι

- USER – TAG, ο οποίος δείχνει πόσες φορές ένας χρήστης έχει προσθέσει μια ετικέτα

- ITEM – TAG, ο οποίος δείχνει πόσες φορές μια ετικέτα έχει προστεθεί σε έναν καλλιτέχνη
- USER – ITEM ALTERNATIVE, ο οποίος χρειάζεται στην περίπτωση όπου δεν υπάρχει διμερής γράφος User – Item και αναπαριστά πόσες φορές ένας χρήστης έχει προσθέσει κάποια ετικέτα σε έναν καλλιτέχνη.



Εικόνα 4.12 παράδειγμα user-item-tag γράφου

Για παράδειγμα, από την παραπάνω εικόνα βλέπουμε ότι ο χρήστης 2 έχει προσθέσει στον καλλιτέχνη 52 τρείς ετικέτες την 13, την 15 και την 18. Συνεπώς, στον γράφο User – Tag, ο χρήστης 2 με την ετικέτα 15 και 18 θα έχει βάρος 1 και με την ετικέτα 13 θα έχει βάρος 2, καθώς έχει προσθέσει την ίδια και στον καλλιτέχνη 63. Στον γράφο Item – Tag, ο καλλιτέχνης 52 θα έχει

βάρος 1 με τις ετικέτες 13, 15, 18, καθώς του έχουν προστεθεί από μια φορά η κάθε μια. Τέλος, στον γράφο User – Item Alternative, ο χρήστης 2 με τον καλλιτέχνη 52 θα έχει βάρος 3, καθώς του έχει προσθέσει ετικέτες τρεις φορές.

4.3 Συμπεράσματα προτεινόμενης προσέγγισης

Στο συγκεκριμένο κεφάλαιο παρουσιάστηκε μια υποδομή για την παραμετρική δημιουργία ενός εμπλουτισμένου γράφου, ανάλογα με τη φύση και την ερμηνεία διαφορετικών γράφων. Το προτεινόμενο εργαλείο, στόχο έχει να απλοποιήσει την διαδικασία επιλογής και συνδυασμού διαφορετικών γράφων με σκοπό να δημιουργήσει έναν εμπλουτισμένο γράφο, ο οποίος θα ενσωματώνει πολλαπλές πληροφορίες. Η επιλογή των γράφων, καθώς και ο τρόπος χρήσης της πληροφορίας που είναι διαθέσιμη σε αυτούς, μπορεί να γίνει ευκολά με την τροποποίηση των παραμέτρων του config αρχείου (βλ. Εικόνα 4.5). Επιπλέον, το συγκεκριμένο εργαλείο μπορεί να δημιουργήσει τον εμπλουτισμένο γράφο, μόνο από την εισαγωγή του τριμερή γράφου (user-item-tag), συνεπώς καλύπτει τις περιπτώσεις όπου δεν είναι διαθέσιμοι οι άλλοι γράφοι στα σύνολα δεδομένων. Τέλος, το εργαλείο διαχωρίζει τους τρεις γράφους ({user-item} ∈{users} ∈{richgraph}), με την ίδια δομή, συνεπώς ο χρήστης μπορεί να χρησιμοποιήσει κατευθείαν το κάθε γράφο (τα csv αρχεία), όπως εκείνος επιθυμεί.

5. Εργαλεία για την δημιουργία και την αξιολόγηση των συστάσεων

Υπάρχουν διάφορα εργαλεία τα οποία ενσωματώνουν αλγορίθμους συστημάτων συστάσεων, με σκοπό την δημιουργία και αξιολόγηση των συστάσεων. Οι κύριοι στόχοι, καθώς και οι υπάρχουσες λειτουργίες αυτών των λογισμικών διαφέρουν σημαντικά.

Κάποια από τα πιο γνωστά εργαλεία είναι τα εξής:

Το LensKit[5] και το MyMediaLite[6] δημιουργήθηκαν αρχικά για την υποστήριξη της έρευνας και της εκπαίδευσης πάνω σε αλγορίθμους συστημάτων συστάσεων (Said & Bellogín, 2014). Το πρώτο, το οποίο αναπτύχθηκε από το GroupLens[7], είναι ένα πακέτο εργαλείων βασισμένο σε Java και χρησιμοποιείται για την ανάπτυξη, την έρευνα και την μελέτη RS αλγορίθμων. Ενώ, το MyMediaLite απευθύνεται τόσο σε ακαδημαϊκούς όσο και εμπορικούς χρήστες και καταγράφει την κατάσταση των αλγορίθμων τέχνης RS. Το MyMediaLite υλοποιείται σε C#, αλλά μπορεί εύκολα να κληθεί από άλλες γλώσσες, όπως για παράδειγμα Ruby και Python.

Επίσης, όσο αναφορά την επεκτασιμότητα, το GraphLab[8], το GraphChi[9] και το Apache Mahout[10] υποστηρίζουν μεγάλα σύνολα δεδομένων μέσω κατανεμημένων υπολογιστικών συστημάτων (Fan & Bifet, 2013).

Στην παρούσα εργασία και πιο συγκεκριμένα για την δημιουργία συστάσεων στους διάφορους γράφους και την αξιολόγηση αυτών, χρησιμοποιήθηκε το εργαλείο Lenskit, το οποίο περιγράφεται παρακάτω.

5.1 Lenskit

Το Lenskit (Ekstrand, Ludwig, Konstan, & Riedl, 2011) είναι ένα πακέτο λογισμικού (Java-based) ανοιχτού κώδικα για την κατασκευή, την έρευνα και την εκμάθηση συστημάτων συστάσεων. Ουσιαστικά είναι μια ευέλικτη και ισχυρή πλατφόρμα για πειραματισμό με διάφορες τεχνικές σύστασης.

Πιο συγκεκριμένα προσφέρει:

- Κοινά APIs για εργασίες σύστασης, όπως να προτείνει και να προβλέπει αντικείμενα
- Εφαρμογή τυποποιημένων αλγορίθμων σύστασης και πρόβλεψης αξιολογήσεων, καθιστώντας εύκολη την ενσωμάτωση αυτών σε ερευνητικές εργασίες και εφαρμογές.

[5] http://lenskit.org/
[6] http://www.mymedialite.net/
[7] http://grouplens.org/
[8] https://en.wikipedia.org/wiki/GraphLab
[9] https://github.com/GraphChi
[10] http://mahout.apache.org/

- Ένα σύνολο εργαλείων αξιολόγησης για τη μέτρηση των επιδόσεων των συστημάτων συστάσεων σε κοινά σύνολα δεδομένων, με μια ποικιλία από μετρικές.
- Εκτενή κώδικα υποστήριξης για να επιτρέψει στους προγραμματιστές να δημιουργήσουν νέους αλγόριθμους, μεθοδολογίες αξιολόγησης, και άλλες επεκτάσεις με ελάχιστες εργασίες.

Ο LensKit evaluator επιτρέπει στον χρήστη να εκπαιδεύσει τους αλγορίθμους που επιθυμεί σε διαφορετικά σύνολα δεδομένων, να μετρήσει την επίδοση τους και να επικυρώσει σε μέρη (cross-validate) τα αποτελέσματα, για την επίτευξη της εγκυρότητας της ανάλυσης.

Για να εκτελεστεί οποιοδήποτε πείραμα στον evaluator, χρειάζεται ένα evaluation script, το οποίο είναι γραμμένο σε γλώσσα προγραμματισμού Groovy[11] (eval.groovy). Μέσα στο evaluation script ορίζονται τα εξής:

- Το σύνολο δεδομένων που θα γίνει το evaluation
- Ποιοι αλγόριθμοι θα χρησιμοποιηθούν
- Ποιες μετρικές θα χρησιμοποιηθούν για να μετρήσουν την επίδοση

Στον LensKit, ο train-test evaluator κάνει build και test τους αλγορίθμους που έχουν δηλωθεί, μετράει τα αποτελέσματα με τις μετρικές και γράφει τα αποτελέσματα σε ένα αρχείο. Το trainTest {} δείχνει στον LensKit, ότι θα υλοποιηθεί train-test evaluation.

Στην αρχή του trainTest, όπως και στον παρακάτω κώδικα ορίζεται το σύνολο δεδομένων (source csvfile (**"testing/ useritem_movielens.csv "**)). Στο συγκεκριμένο παράδειγμα έχουμε ένα csv αρχείο, με delimeter « ; » (semicolon) και το domain δείχνει ότι οι αξιολογήσεις που εμπεριέχονται σε αυτό το σύνολο δεδομένων είναι από 1 έως 5, με ακρίβεια βήματος 0.1.

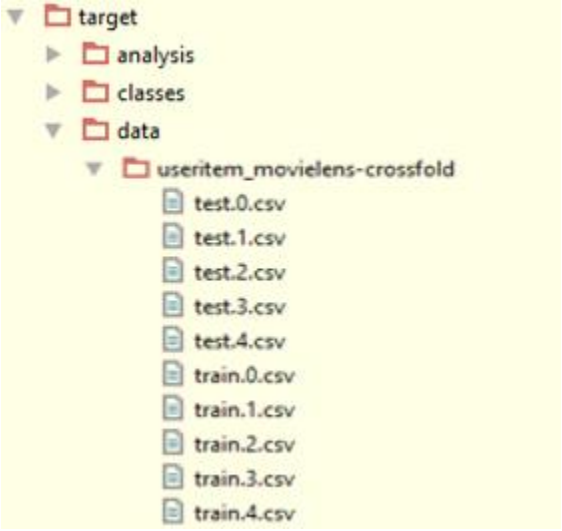
Η εντολή crossfold παίρνει το σύνολο δεδομένων και το τμηματοποιεί για την διαδικασία του crossfold validation. Εξορισμού, δημιουργούνται πέντε υποσύνολα, από τα οποία προκύπτουν και τα ζεύγη train – test.

```
trainTest {
    dataset crossfold("${dataDir}") {
      source csvfile("testing/useritem_movielens.csv") {
       delimiter ";"
        domain {
               minimum 1.0
               maximum 5.0
               precision 0.1
               }
          }
      }
```

Εικόνα 5.1 trainTest block, eval.groovy

[11] http://www.groovy-lang.org/

Τα υποσύνολα train και test του αρχικού υποσυνόλου αποθηκεύονται σε ένα ξεχωριστό folder.



Εικόνα 5.2 CROSSFOLD

Το επόμενο μέρος είναι ο ορισμός των αλγορίθμων. Για την συγκεκριμένη εργασία χρησιμοποιήθηκαν ο UserUser, ο FunkSVD και ο ItemItem, όπως δίνονται από τον Lenskit:

UserUser

```
algorithm("UserUser") {
    // use the user-user rating predictor
    bind ItemScorer to UserUserItemScorer
    bind (BaselineScorer, ItemScorer) to UserMeanItemScorer
    bind (UserMeanBaseline, ItemScorer) to ItemMeanRatingItemScorer
    bind VectorNormalizer to MeanVarianceNormalizer

    // use 30 neighbors for predictions
    set NeighborhoodSize to 30

    // override normalizer within the neighborhood finder
    // this makes it use a different normalizer (subtract user mean) for computing
    // user similarities
    within(NeighborhoodFinder) {
        bind UserVectorNormalizer to BaselineSubtractingUserVectorNormalizer
        // override baseline to use user mean
        bind (UserMeanBaseline, ItemScorer) to GlobalMeanRatingItemScorer
    }

    // and apply some Bayesian damping to the baseline
    within(BaselineScorer, ItemScorer) {
        set MeanDamping to 25.0d
     }
}
```

Ο κώδικας του αλγορίθμου UserUser βρίσκεται στο lenskit-knn module, κάτω από το πακέτο org.grouplens.lenskit.knn.user. Οι προεπιλεγμένες ρυθμίσεις είναι αυτές που φαίνονται

παραπάνω. Ουσιαστικά, χρησιμοποιείται ο βασικός αλγόριθμος UserUser με ομοιότητα συνημίτονου για 30 γείτονες.

ItemItem

```
algorithm("ItemItem") {
    // use the item-item rating predictor with a baseline and normalizer
    bind ItemScorer to ItemItemScorer
    bind (BaselineScorer, ItemScorer) to UserMeanItemScorer
    bind (UserMeanBaseline, ItemScorer) to ItemMeanRatingItemScorer
    bind UserVectorNormalizer to BaselineSubtractingUserVectorNormalizer

    // retain 500 neighbors in the model, use 30 for prediction
    set ModelSize to 500
    set NeighborhoodSize to 30

    // apply some Bayesian smoothing to the mean values
    within(BaselineScorer, ItemScorer) {
        set MeanDamping to 25.0d
    }
}
```

Ο κώδικας του αλγορίθμου ItemItem βρίσκεται και αυτός στο lenskit-knn module, στο org.grouplens.lenskit.knn.item πακέτο. Όπως φαίνεται και παραπάνω χρησιμοποιεί 30 γείτονες για πρόβλεψη.

FunkSVD

```
algorithm("FunkSVD") {
    bind ItemScorer to FunkSVDItemScorer
    bind UserVectorNormalizer to BaselineSubtractingUserVectorNormalizer
    bind(BaselineScorer, ItemScorer) to UserMeanItemScorer
    bind(UserMeanBaseline, ItemScorer) to ItemMeanRatingItemScorer
    set FeatureCount to 40
set LearningRate to 0.002
set IterationCount to 125
}
```

Στον lenskit, ο κώδικας του FunkSVD βρίσκεται στο lenskit-SVD module, κάτω από το πακέτο org.grouplens.lenskit.mf.FunkSVD. Όπως φαίνεται και από το παραπάνω configuration, για τον αλγόριθμο ορίζονται 40 χαρακτηριστικά (FeatureCount), για κάθε 125 επαναλήψεις (IterationCount) που θα εκτελέσει για κάθε πρόβλεψη και με ρυθμό μάθησης (IterationCount) 0.002.

Επόμενο βήμα είναι η δήλωση των μετρικών. Στο συγκεκριμένο παράδειγμα δηλώνονται πέντε μετρικές:

```
metric CoveragePredictMetric
metric RMSEPredictMetric
metric NDCGPredictMetric
metric MAEPredictMetric
metric topNnDCG {
listSize5
}
```

Αυτές οι μετρικές είναι κλάσεις του πακέτου org.grouplens.lenskit.eval.metrics.predict και παράγουν τα εξής:

- CoveragePredictMetric: coverage και διάφορα άλλα στατιστικά όπως:
  - NUsers, ο αριθμός των χρηστών που ελέγχθηκαν
  - NAttempted, ο αριθμός των προβλέψεων που αποπειράθηκαν να γίνουν
  - NGood, ο αριθμός των προβλέψεων που έγιναν
  - Coverage, ο αριθμός των προβλέψεων που αποπειράθηκαν να γίνουν προς τον αριθμό των προβλέψεων που έγιναν.
- RMSEPredictMetric: υπολογίζει το RMSE των προβλέψεων σε σχέση με τις πραγματικές βαθμολογίες. Υπολογίζει και ανά χρήστη (RMSE.ByUser) και ανά βαθμολογία (RMSE.ByRating).
- NDCGPredictMetric: υπολογίζει το nDCG τις λίστας με τις βαθμολογίες των χρηστών.

Τα αποτελέσματα της αξιολόγησης αποθηκεύονται στο αρχείο eval-results.csv. Επίσης, μπορούν να παραχθούν και άλλα δυο αρχεία: το userOutput, το οποίο περιέχει τα αποτελέσματα των μετρικών για κάθε test χρήστη και το predictOutput, το οποίο παρουσιάζει την προβλεπόμενη και την πραγματική βαθμολογία.

```
output "${config.analysisDir}/eval-results.csv"
predictOutput "${config.analysisDir}/eval-preds.csv"
userOutput "${config.analysisDir}/eval-user.csv"
```

Επίσης, δίνεται η δυνατότητα παραγωγής διαγραμμάτων επικοινωνίας των components για οποιοδήποτε αλγόριθμο. Για παράδειγμα του FunkSVD (βλ. παράρτημα 8.2):

```
target('draw') {
    dumpGraph {
        output "${config.analysisDir}/funk.dot"
        algorithm("FunkSVD")
    }

}
```

Τέλος, εμπεριέχεται και ένα script αρχείο γραμμένο σε python (chart.py) το οποίο αναλύει (δημιουργία box plot, κ.α.) τα αποτελέσματα από το παραγόμενο csv αρχείο.

```
target('analyze') {
    requires 'evaluate'
    // Run R. Note that the script is run in the analysis directory; you might
want to
    // copy all R scripts there instead of running them from the source dir.
    ant.exec(executable: 'python', dir: config.analysisDir) {
        arg value: "${config.scriptDir}/chart.py"
    }

}
```

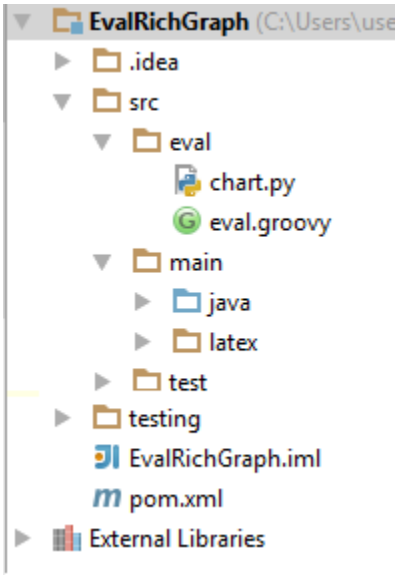
6. Πειραματική αξιολόγηση

Η εκτέλεση των πειραμάτων έγινε χρησιμοποιώντας έναν προσωπικό υπολογιστή με τα εξής χαρακτηριστικά:

- Windows 8 64-bit
- 6GB μνήμης RAM
- SSD σκληρός δίσκος
- Τετραπύρηνος επεξεργαστής Intel Core i7-4702MQ στα 2.20GHz

Μέσω του ολοκληρωμένου περιβάλλοντος διαχείρισης Intelij[12], δημιουργήθηκε ένα Maven Project from Archetype, χρησιμοποιώντας την έκδοση 2.1 του lenskit-archetype-fancy-analysis. Η έκδοση της Java είναι η 1.8. Περισσότερες λεπτομέρειες παρουσιάζονται στο αρχείο pom.xml (βλ. Παράρτημα 8.1).

Η δομή του project έχει ως εξής:



Μέσα στον φάκελο eval εμπεριέχεται το αρχείο eval.groovy, το οποίο περιγράψαμε παραπάνω και το chart.py, ένα script αρχείο για την ανάλυση των αποτελεσμάτων. Στο project δημιουργήθηκε και ένας φάκελος που βρίσκονται όλα τα csv αρχεία προς εξέταση.

Επιπλέον, εκτός από την χρήση του Lenskit, χρησιμοποιήθηκαν και τα προγράμματα TED Notepad[13] για την επεξεργασία των ακμών των γράφων (shuffling και splitting train & test files) και το SPSS Statistics 21.0[14] για την εμφάνιση των συχνοτήτων των nWeights.

[12] Προτιμάτε από τους περισσότερους προγραμματιστές που πρώτο-ασχολούνται με το Lenskit, λόγο της ευκολίας που παρέχει.
[13] http://jsimlo.sk/notepad/
[14] http://www-01.ibm.com/software/analytics/spss/products/statistics/

6.1 Σύνολα Δεδομένων

Τα σύνολα δεδομένων τα οποία χρησιμοποιήθηκαν στο εργαλείο, έχουν παραχθεί από το Information Retrieval Group του πανεπιστημίου Universidad Autónoma de Madrid και είναι διαθέσιμα στην ιστοσελίδα του grouplens[15].

Το πρώτο σύνολο δεδομένων αφορά το Last.fm[16], μια μουσική ιστοσελίδα, η οποία είναι εύκολη στη χρήση και δωρεάν σε οποιονδήποτε είναι διατεθειμένος να συμμετάσχει. Μόλις ένας χρήστης είναι συνδεδεμένος μπορεί να ακούσει το μουσικό κομμάτι που επιθυμεί και να προσθέσει αυθαίρετες ετικέτες σε αυτό. Το συγκεκριμένο σύνολο δεδομένων περιέχει τρεις γράφους:

- ✓ έναν κοινωνικό γράφο με 1892 μοναδικούς χρήστες,
- ✓ έναν γράφο το οποίο περιέχει 17632 καλλιτέχνες και δείχνει πόσες φορές ο χρήστης έχει ακούσει κάποιον καλλιτέχνη και
- ✓ έναν γράφο προσθήκης ετικετών με 11946 μοναδικές ετικέτες.

Το δεύτερο σύνολο δεδομένων αφορά το MovieLens[17], μια διαδικτυακή κοινότητα που συνιστά ταινίες στους χρήστες για να παρακολουθήσουν με βάση τις προτιμήσεις τους (χρησιμοποιώντας το συνεργατικό φιλτράρισμα). Το συγκεκριμένο σύνολο δεδομένων περιέχει δυο γράφους:

- ✓ έναν γράφο αξιολογήσεων με 2113 μοναδικούς χρήστες και 10197 ταινίες,
- ✓ έναν γράφο προσθήκης ετικετών με 13222 μοναδικές ετικέτες.

Τέλος, το τρίτο σύνολο δεδομένων αφορά το Delicious[18], μια διαδικτυακή υπηρεσία, για την αποθήκευση, τον διαμοιρασμό και την ανακάλυψη νέων σελιδοδεικτών. Το συγκεκριμένο σύνολο περιέχει δυο γράφους:

- ✓ έναν κοινωνικό γράφο με 1867 μοναδικούς χρήστες και
- ✓ έναν γράφο προσθήκης ετικετών με 53388 μοναδικές ετικέτες.

Στον παρακάτω πίνακα (Πίνακας 6.1) παρουσιάζονται συνοπτικά τα χαρακτηριστικά των συνόλων δεδομένων που χρησιμοποιήθηκαν. Επιπροσθέτως, για τους γράφους user-user και user-item φαίνονται η χρονοσφραγίδα και τα βάρη (αν δίνονται), ο τύπος του βάρους, το βήμα και η κλίμακα.

Το σύνολο των χρηστών και των αντικειμένων για κάθε γράφο που δίνεται είναι το ίδιο. Δηλαδή, οι χρήστες που εμπεριέχονται στον κοινωνικό γράφο είναι οι ίδιοι και στον γράφο αξιολογήσεων και στον γράφο προσθήκης ετικετών. Αντιστοίχως, και τα αντικείμενα που εμφανίζονται στο γράφο αξιολογήσεων είναι τα ίδια και στον γράφο προσθήκης ετικετών.

[15] http://grouplens.org/datasets/hetrec-2011/
[16] http://www.last.fm/
[17] https://movielens.org/
[18] https://delicious.com/

		Σύνολα Δεδομένων		
Γράφοι	**Χαρακτηριστικά**	**Lastfm**	**Movielens**	**Delicious**
User – User Κοινωνικός Γράφος	Πλήθος Μοναδικών Χρηστών	1892	-	1867
	Πλήθος Ακμών Φιλίας	12717(Directed)/ 25434	-	7668(Directed)/ 15328
	Μέσος Όρος Ακμών Φιλίας ανά Χρήστη	13.44	-	8.24
	Βάρος Ακμών	-	-	-
	Timestamp	-	-	✓
User – Item Γράφος Αξιολογήσεων	Πλήθος Μοναδικών Χρηστών που αξιολογούν	1892	2113	-
	Πλήθος Μοναδικών Αντικειμένων	17632	10197	-
	Πλήθος Βαθμολογιών	92834	855598	-
	Μέσος Όρος Πλήθους Βαθμολογιών ανά Χρήστη	49.07	404.92	-
	Μέσος Όρος Πλήθους Βαθμολογιών ανά Αντικείμενο	5.3	84.64	-
	Βάρος Ακμών	✓	✓	-
	Τύπος βάρους	Listening Count	Ratings	-
	Κλίμακα βαρών	1 - 352698	0.5 - 5	-
	Ακρίβεια/Βήμα	1	0.5	-
	Timestamp	-	✓	-
User – Item – Tag Γράφος Προσθήκης Ετικετών	Πλήθος Μοναδικών Χρηστών που προσθέτουν ετικέτες	1892	2113	1867
	Πλήθος Μοναδικών Αντικειμένων που έχουν προστεθεί Ετικέτες	17632	10197	69226
	Πλήθος Μοναδικών Ετικετών που έχουν προστεθεί από τους Χρήστες	11946	13222	53388
	Πλήθος Ετικετών που έχουν προστεθεί από τους Χρήστες	186479	47957	437593
	Μέσος Όρος Πλήθους Ετικετών ανά Χρήστη	98.56	22.70	234.38
	Μέσος Όρος Πλήθους Ετικετών ανά Αντικείμενο	14.89	8.12	6.32

Πίνακας 6.1 Σύνολα δεδομένων

Η πυκνότητα των δεδομένων (βλ. κεφ. 2.1) για τους πίνακες user-item των συνόλων δεδομένων φαίνεται παρακάτω. Για το σύνολο δεδομένων Delicious, δεν δίνεται ο γράφος αξιολογήσεων, συνεπώς παρουσιάζεται ο user-item alternative:

Lastfm (user-item)	**Movielens** (user-item)	**Delicious** (user-item alternative)
$\frac{92834}{1892*17632} = 0.003$	$\frac{855598}{2113*10197} = 0.04$	$\frac{104799}{1867*69226} = 0.0008$

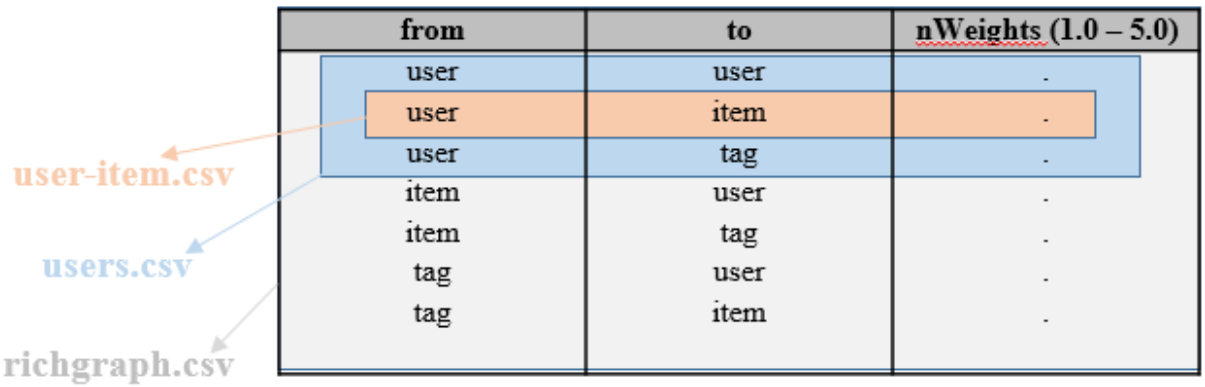
Πίνακας 6.2 πυκνότητα δεδομένων για τους πίνακες user-item

6.2 Διαδικασία πειραμάτων και αποτελέσματα

Όπως αναφέρεται και σε προηγούμενο κεφάλαιο, με την βοήθεια του εργαλείου δημιουργούνται τρεις γράφοι:

- user-item: βαθμολογίες χρηστών σε αντικείμενα (αν δίνεται από το σύνολο δεδομένων), διαφορετικά το user-item alternative (που δημιουργείται από τον user-item-tag)
- users: περιλαμβάνει τον user-user (αν δίνεται), τον user-item και τον user-tag
- richgraph: περιλαμβάνει τον users, τον item-user, τον item-tag, τον tag-user και τον tag-item.

Η μορφή που έχουν και τα τρία αρχεία είναι .csv με delimiter « ; » και περιέχουν τρεις στήλες της δομής from; to; nWeights (normalized weights).



Εικόνα 6.1 μορφή των παραγόμενων csv αρχείων

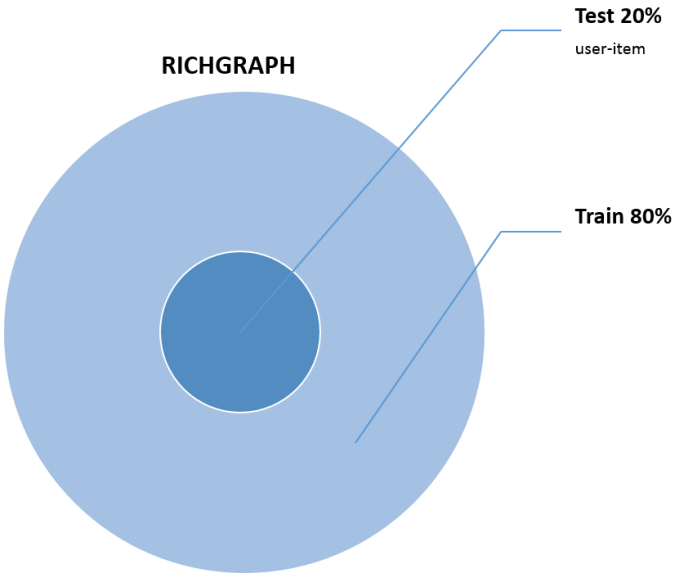
Αυτά τα παραγόμενα αρχεία εισάγονται στο Lenskit (βλ. eval.groovy, Εικόνα 5.1) και επειδή αυτό το πρόγραμμα αντιλαμβάνεται μόνο users και items (δηλαδή το from είναι οι users, το to είναι τα items και το nWeights τα ratings), για το λόγο αυτό ξεχωρίζουμε με διαφορετικούς αριθμούς τα

users, items και tags δίνοντας διαφορετικό αριθμό έναρξης των ids για τα items και τα tags (βλ. Εικόνα 4.5).

Επιπροσθέτως, όπως έχει προαναφερθεί και στο κεφάλαιο 4, προϋπόθεση για την δημιουργία των τριών γράφων – csv αρχείων είναι η ύπαρξη του τριμερή γράφου user-item-tag, από τον οποίο μπορούν να δημιουργηθούν οι user-item alternative, user-tag, item-tag (και φυσικά οι ανάστροφοι αυτών).

Κατά την διαδικασία των πειραμάτων, εκτελέστηκαν πρώτα οι τρεις αλγόριθμοι (UserUser, ItemItem, FunkSVD) στον user-item και στον users, με την διαδικασία του 5-crossfold validation (βλ. eval.groovy, Εικόνα 5.1). Ουσιαστικά μπορούμε να αξιολογήσουμε ποιος αλγόριθμος αποδίδει καλύτερα στον γράφο user-item και στον users. Για τον γράφο users θα μπορούσαμε να πούμε ότι αποτελεί μια «προέκταση» του πρώτου, καθώς για τους ίδιους χρήστες και τα αντικείμενα του user-item, περιέχει επιπλέον τους χρήστες (user-user) και τις ετικέτες (user-tag).

Τέλος, εκτελέστηκε ο FunkSVD στον richgraph, συγκρίνοντας τα αποτελέσματα με αυτά του user-item. Ουσιαστικά, χωρίσαμε τον richgraph σε train (80%) και test (20%) και από το test κρατήσαμε μόνο τις ακμές που αντιπροσωπεύουν την σχέση χρήστη – αντικείμενο (Εικόνα 6.2). Έπειτα χρησιμοποιήσαμε τον FunkSVD για να εκπαιδευτεί από το train αρχείο και να δοκιμαστεί στο test αρχείο. Σε αυτή την διαδικασία δεν χρησιμοποιήθηκε crossfold validation.



Εικόνα 6.2 train-test richgraph

Στόχος, αυτής της δοκιμής είναι να ελέγξουμε κατά πόσο η εκπαίδευση του FunkSVD σε έναν γράφο με περισσότερη πληροφορία μπορεί να επιφέρει καλύτερα αποτελέσματα (μείωση του ποσοστού λάθους για τις προβλέψεις user-item) από τον απλό γράφο user-item. Ουσιαστικά συγκρίνουμε τα αποτελέσματα του FunkSVD για richgraph (train & test) με του FunkSVD για user-item (train & test)

Για να γίνει η συγκεκριμένη αξιολόγηση, τροποποιήθηκε το κομμάτι του dataset στο eval.groovy του Lenskit. Όπως φαίνεται και παρακάτω στο κομμάτι του dataset, ξεχωρίζουμε σε ποιο αρχείο θα γίνει train και σε ποιο test.

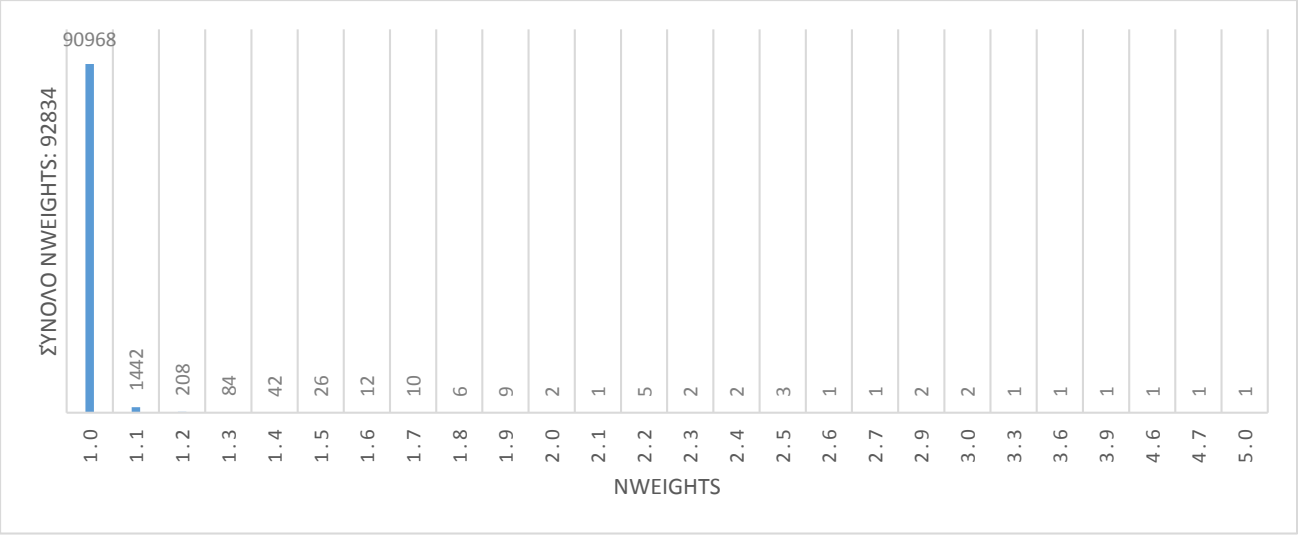
```
dataset {
    train  {
        file "testing/richgraph_movielens80.csv"
        delimiter ";"
        domain {
            minimum 1.0
            maximum 5.0
            precision 0.1
        }


    }
    test {
        file "testing/richgraph_movielens20.csv"
        delimiter ";"
        domain {
            minimum 1.0
            maximum 5.0
            precision 0.1
        }

    }
}
```

Εικόνα 6.3 train - test evaluation, eval.groovy

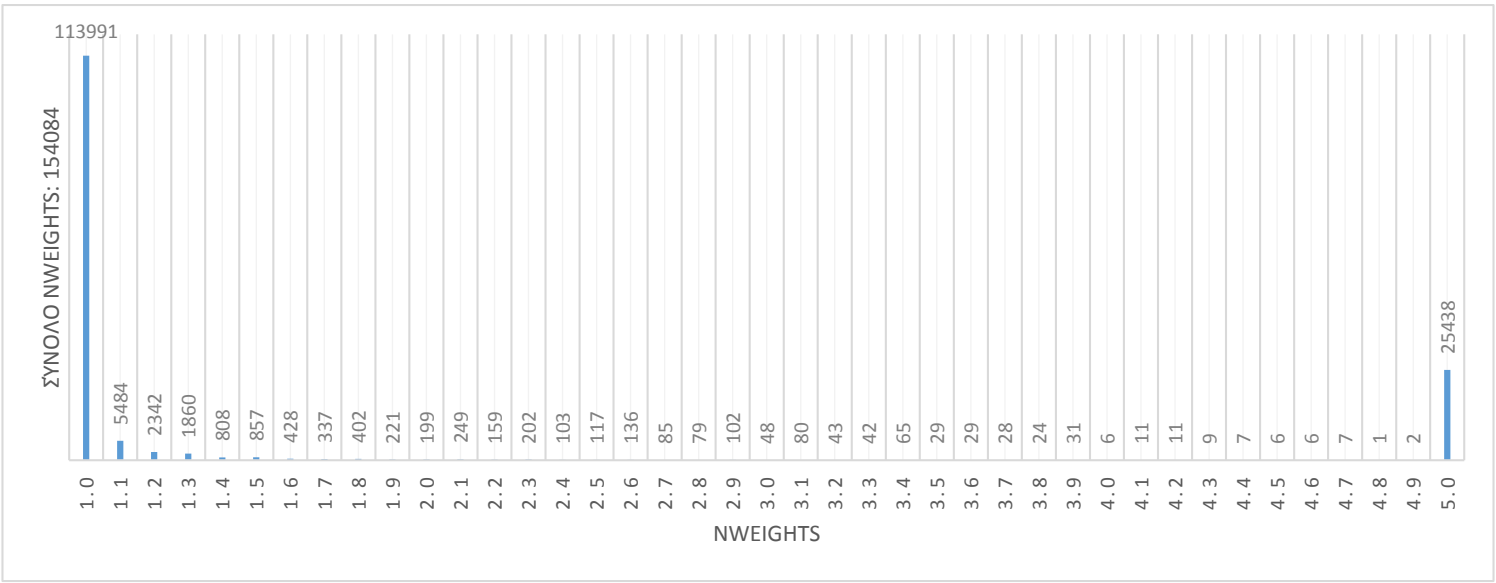
Στην συνέχεια παρουσιάζονται τα αποτελέσματα για κάθε σύνολο δεδομένων

Lastfm

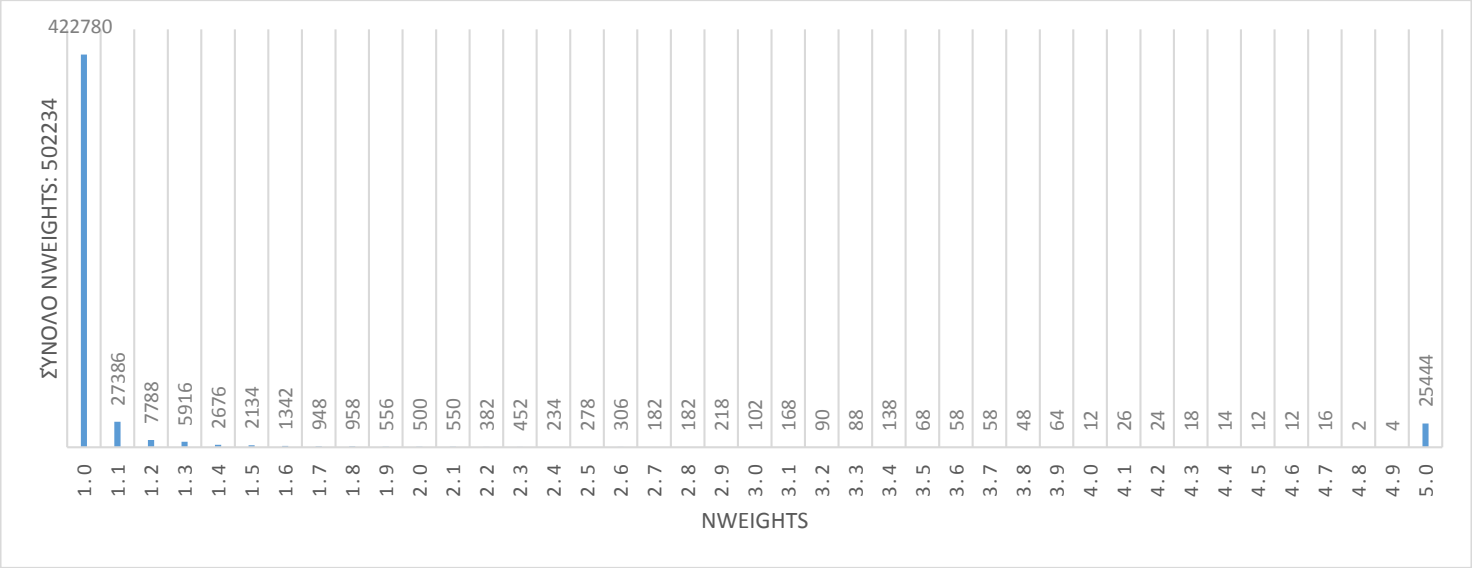
Παρακάτω φαίνονται το σύνολο των nWeights, οι τιμές των nWeights και η συχνότητα εμφάνισης αυτών για κάθε γράφο του lastfm. Όπως φαίνεται και παρακάτω η τιμή 1 εμφανίζεται με ποσοστό πάνω από το 95% για τον γράφο user-item. Αυτό συμβαίνει γιατί η κατανομή των αρχικών βαθμολογιών δεν είναι ομοιόμορφη.



Διάγραμμα 6.1 nWeights user-item, lastfm

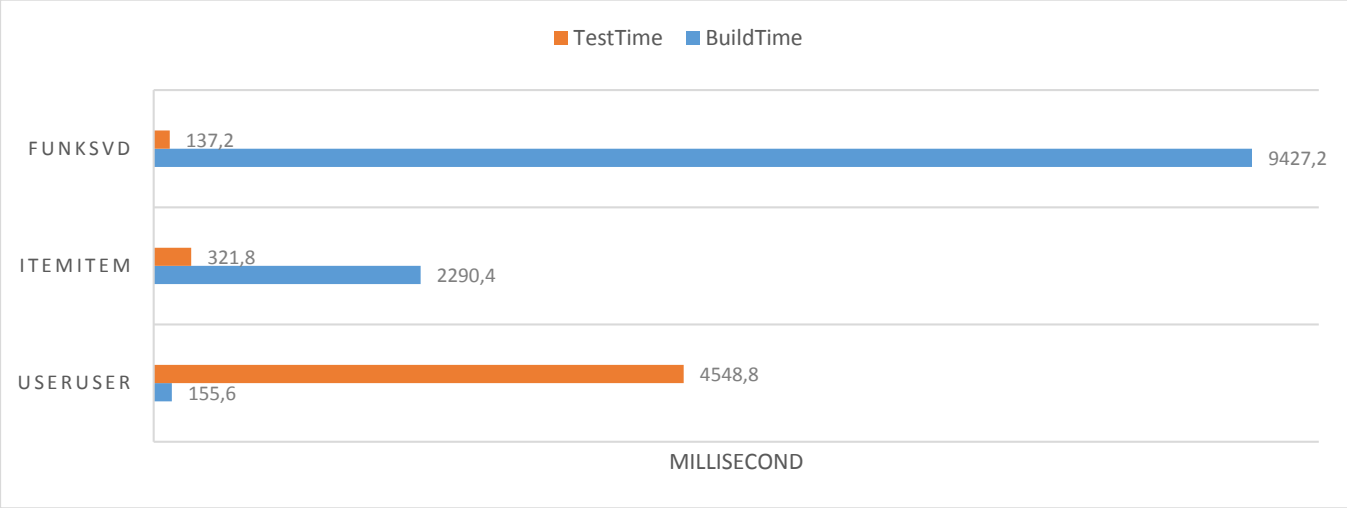


Διάγραμμα 6.2 nWeights users, lastfm

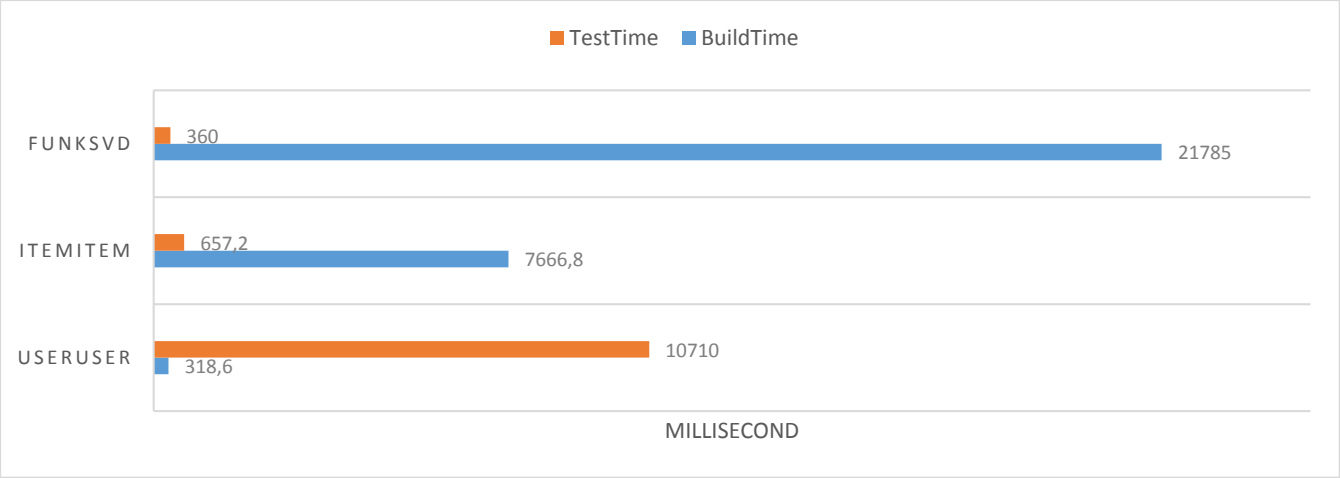


Διάγραμμα 6.3 nWeights richgraph, lastfm

Στην συνέχεια παρουσιάζονται τα αποτελέσματα από την αξιολόγηση των τριών αλγορίθμων για τους γράφους user-item και users. Στα πρώτα διαγράμματα φαίνονται οι μέσοι χρόνοι (build και test time) για τα πέντε τμήματα (5 crossfold validation) του συνόλου των δεδομένων. Ο αλγόριθμος FunkSVD και στους δυο γράφους χρειάζεται περισσότερο χρόνο για εκπαίδευση σε σχέση με τους άλλους δυο.

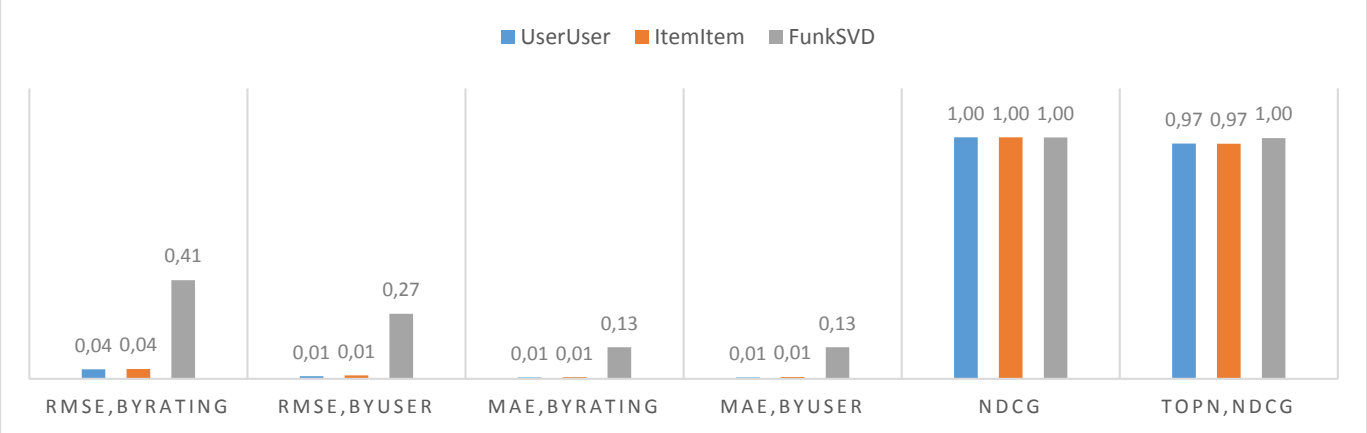


Διάγραμμα 6.4 οι μέσοι χρόνοι για την εκπαίδευση και την αξιολόγηση των 3 αλγορίθμων στον γράφο user-item του lastfm



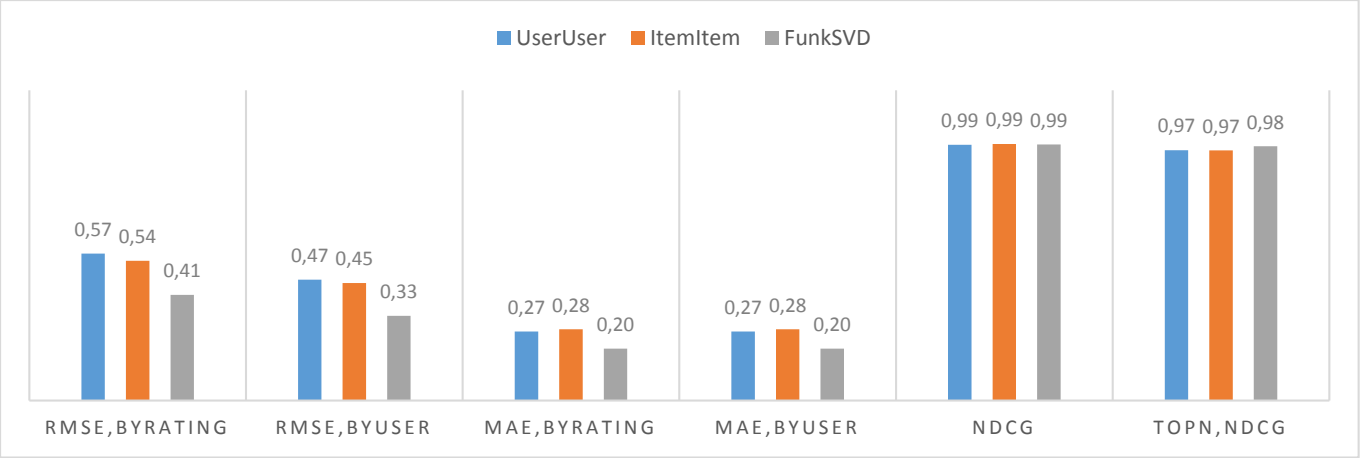
Διάγραμμα 6.5 οι μέσοι χρόνοι για την εκπαίδευση και την αξιολόγηση των 3 αλγορίθμων στον γράφο users του lastfm

Στα επόμενα διαγράμματα παρουσιάζεται ο μέσος όρος των μετρικών για τα πέντε τμήματα του συνόλου δεδομένων. Η ανομοιόμορφη κατανομή στον γράφο user-item επηρέασε τα αποτελέσματα, καθώς το ποσοστό λάθους για τους αλγορίθμους UserUser και ItemItem έφτασε στο μηδέν.



Διάγραμμα 6.6 αποτελέσματα μετρικών για τους τρείς αλγορίθμους στον γράφο user-item του lastfm

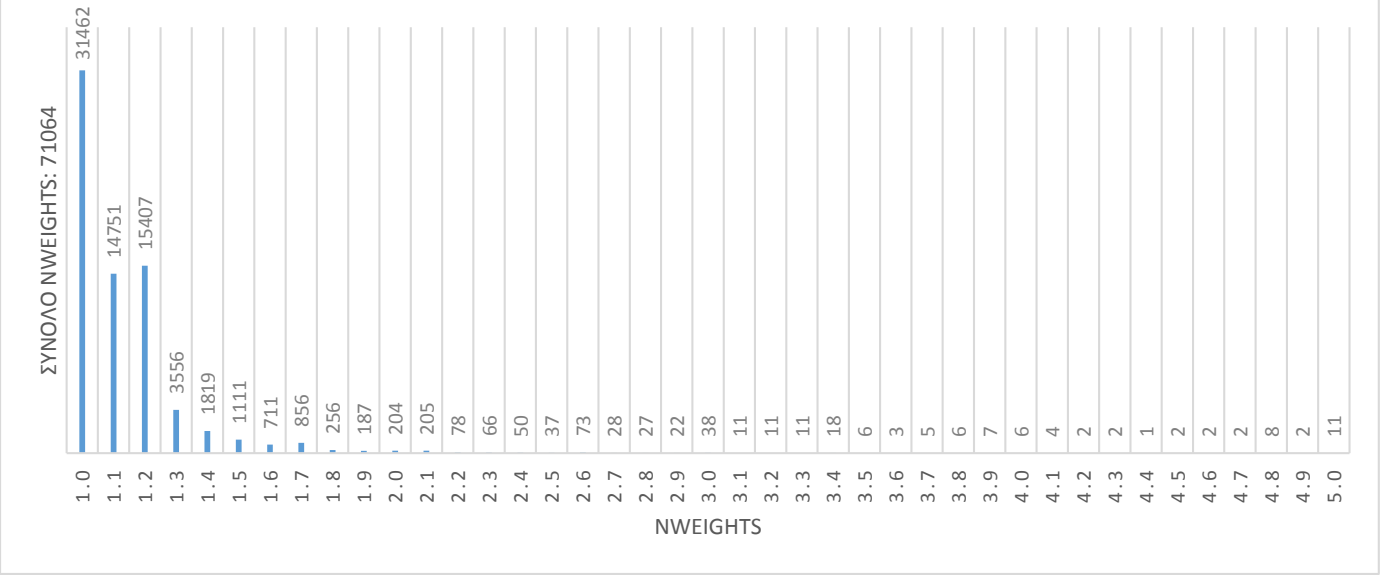
Όπως φαίνεται και στο παρακάτω διάγραμμα, ο αλγόριθμος FunkSVD είχε καλύτερα αποτελέσματα για τον γράφο users, με ελάχιστη όμως διαφορά από τους άλλους δυο.



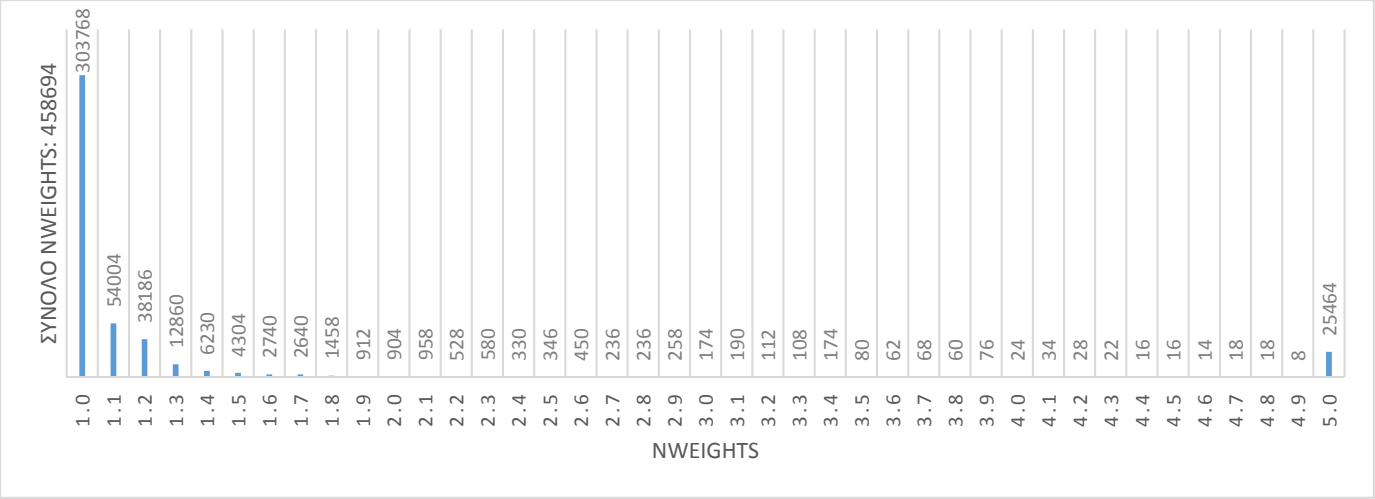
Διάγραμμα 6.7 αποτελέσματα μετρικών για τους τρείς αλγορίθμους στον γράφο users του lastfm

Συνεχίζοντας με την αξιολόγηση του richgraph και επειδή, όπως αναφέρθηκε και παραπάνω, τα κανονικοποιημένα βάρη του γράφου user-item για το σύνολο δεδομένων του lastfm δεν ήταν ομοιόμορφα, δοκιμάστηκε η χρήση του user-item alternative (αφήνοντας την μεταβλητή file2exist κενή στο config αρχείο, βλ. Εικόνα 4.5).

Η συχνότητα εμφάνισης των βαρών για το user-item alternative και του νέου richgraph εμφανίζεται παρακάτω. Η εμφάνιση των τιμών δεν είναι και πάλι ομοιόμορφη, καθώς το μεγαλύτερο ποσοστό καταλαμβάνουν οι τιμές 1, 1.1, 1.2 , αλλά είναι καλύτερη σε σχέση με τον user-item.

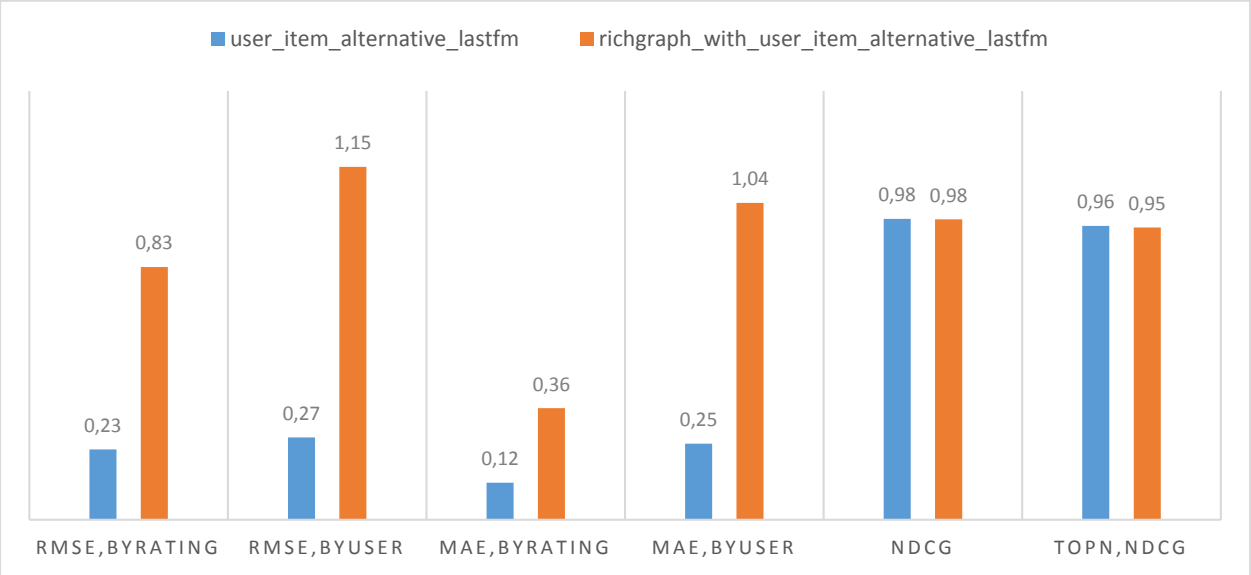


Διάγραμμα 6.8 nWeights user-item alternative, lastfm



Διάγραμμα 6.9 nWeights richgrpaph (with user item alternative), lastfm

Όπως, φαίνεται και από τα παρακάτω αποτελέσματα, η χρήση του FunkSVD στον richgraph δεν επέφερε καλύτερα αποτελέσματα, αντιθέτως η διαφορά των αντίστοιχων είναι πολύ μεγάλη.

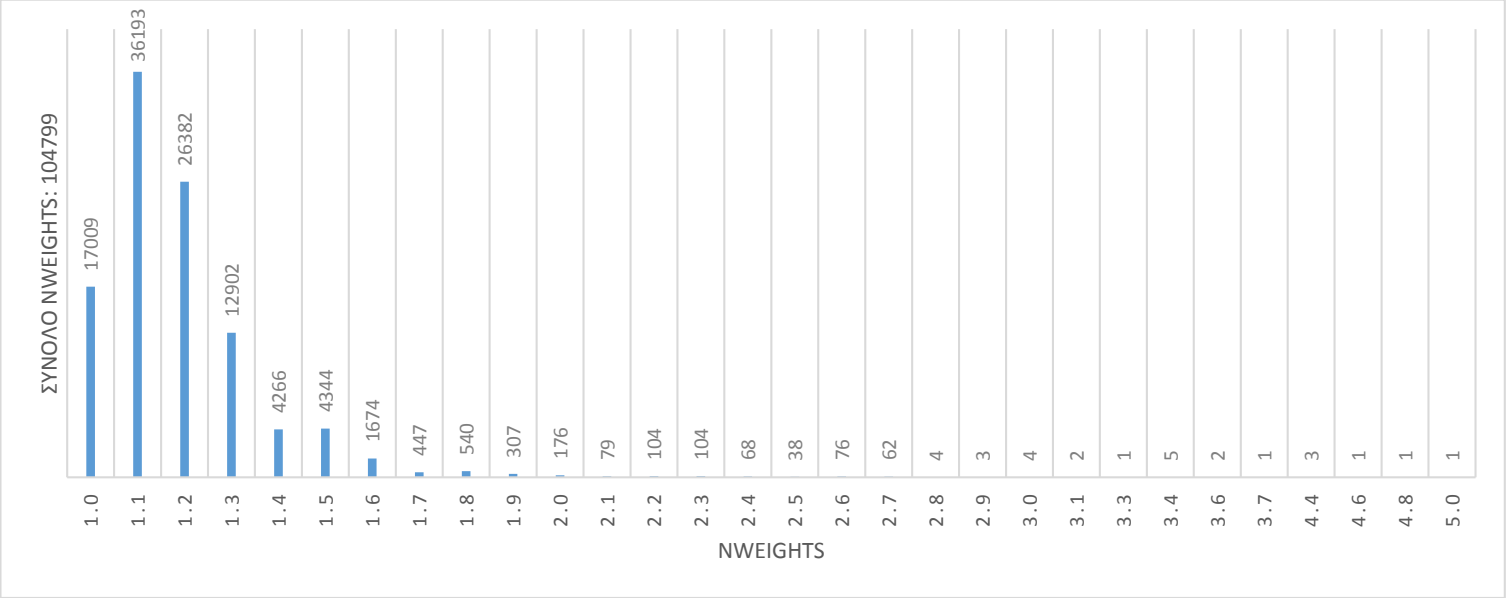


Διάγραμμα 6.10 αποτελέσματα μετρικών για τον αλγόριθμο FunkSVD στους γράφους user-item και richgraph του lastfm

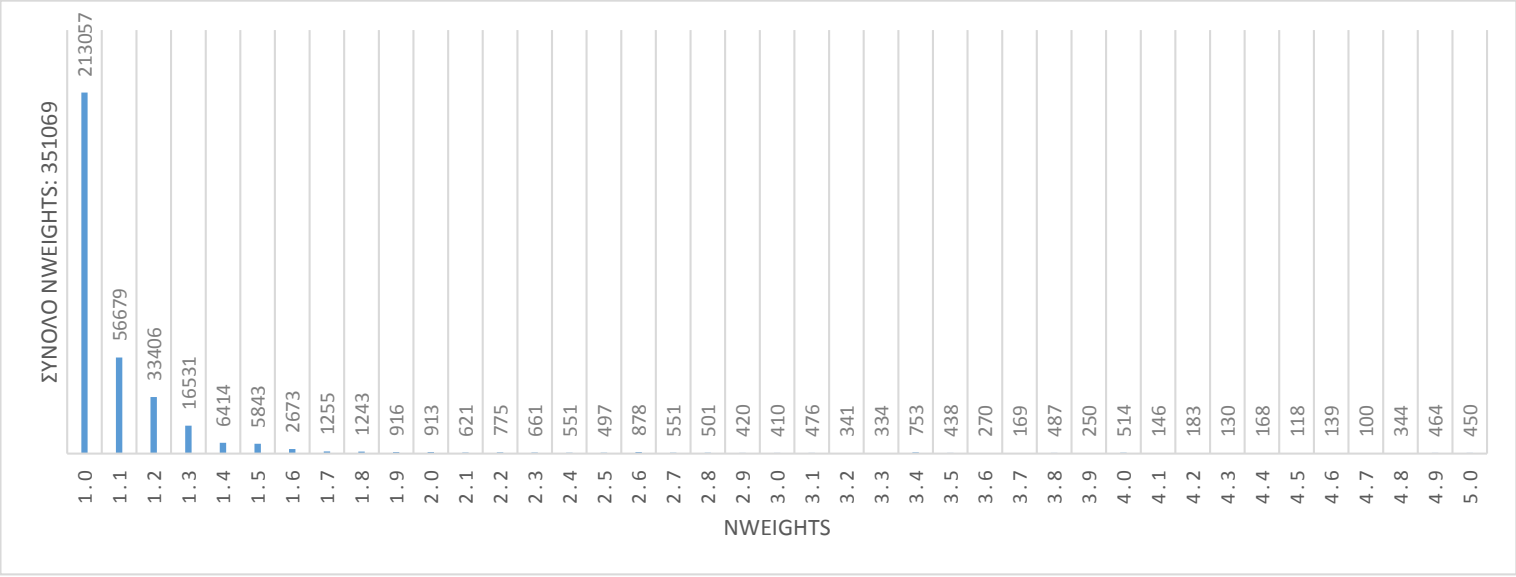
Delicious

Ακολουθώντας την ίδια προσέγγιση, παρουσιάζεται στα επόμενα διαγράμματα το σύνολο των nWeights, οι τιμές των nWeights και η συχνότητα εμφάνισης αυτών, για κάθε γράφο του delicious.

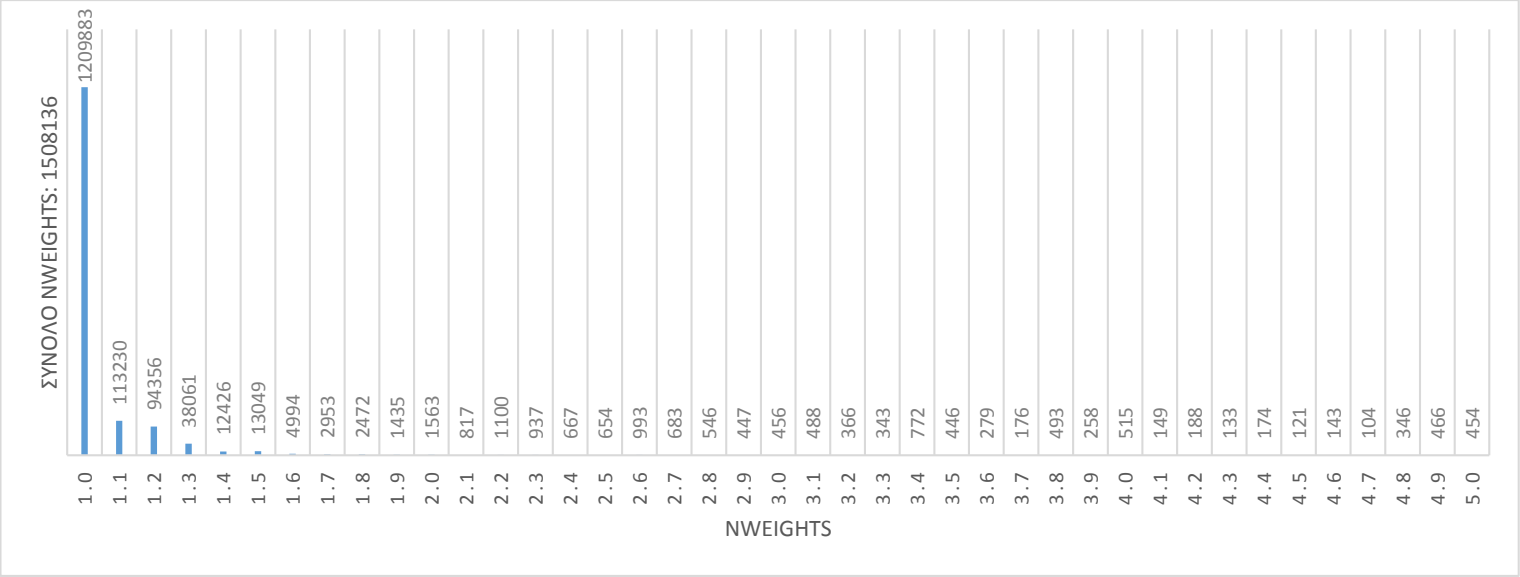
Η συχνότητα εμφάνισης των nWeights είναι και σε αυτή την περίπτωση ανομοιόμορφη με τις τιμές 1, 1.1, 1.2, 1.3 να έχουν το μεγαλύτερο ποσοστό εμφάνισης και στους τρεις γράφους.



Διάγραμμα 6.11 nWeights user item alternative, delicious

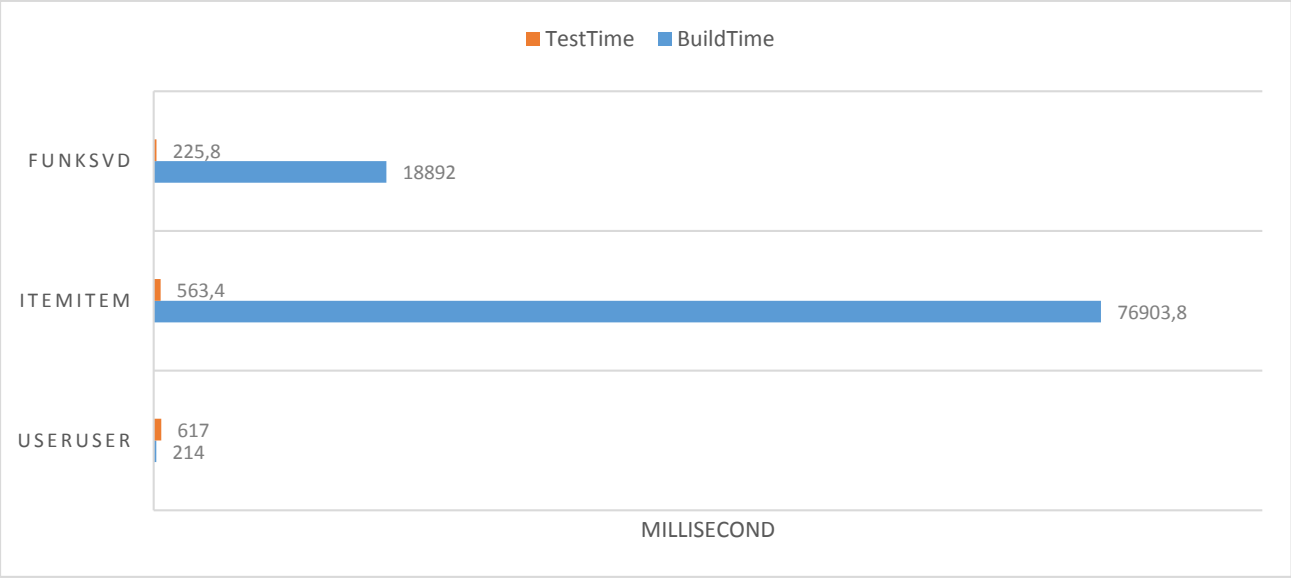


Διάγραμμα 6.12 nWeights users, delicious

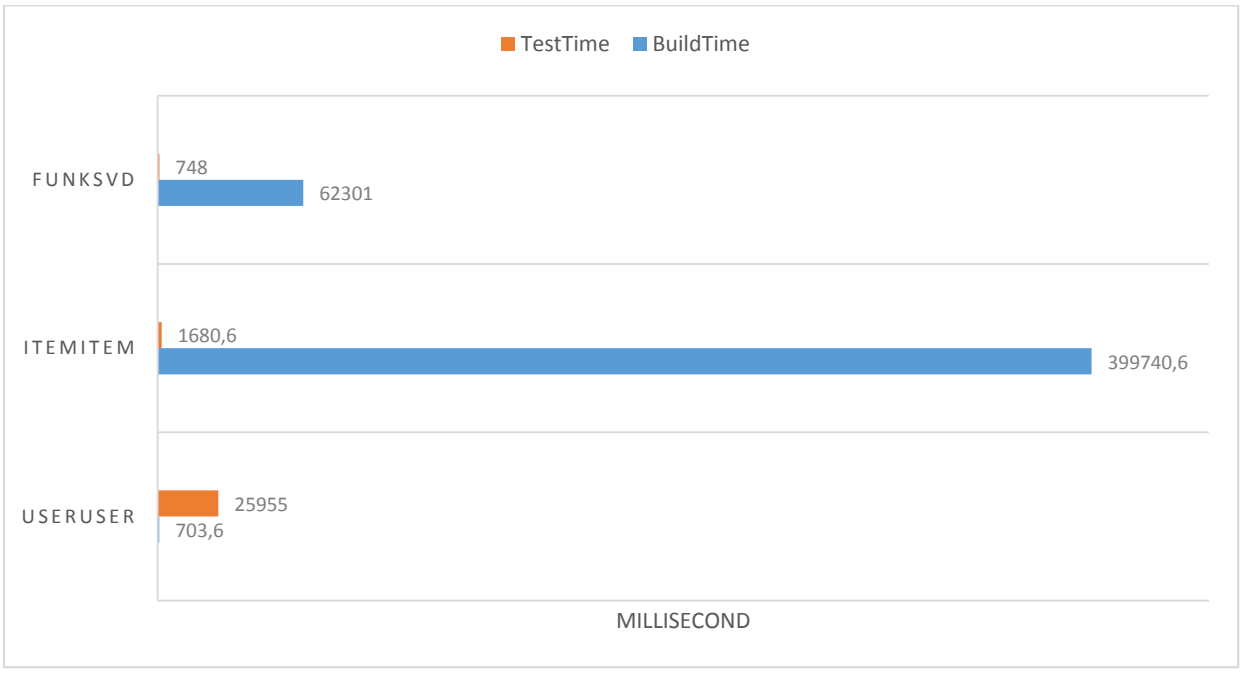


Διάγραμμα 6.13 nWeights richgraph, delicious

Στην συνέχεια παρουσιάζονται τα αποτελέσματα από την αξιολόγηση των τριών αλγορίθμων για τους γράφους user-item και users του delicious. Στο πρώτο διάγραμμα φαίνονται οι μέσοι χρόνοι (build και test time) για τα πέντε τμήματα του συνόλου των δεδομένων. Ο αλγόριθμος ItemItem και στους δυο γράφους χρειάζεται περισσότερο χρόνο για εκπαίδευση σε σχέση με τους άλλους δυο.

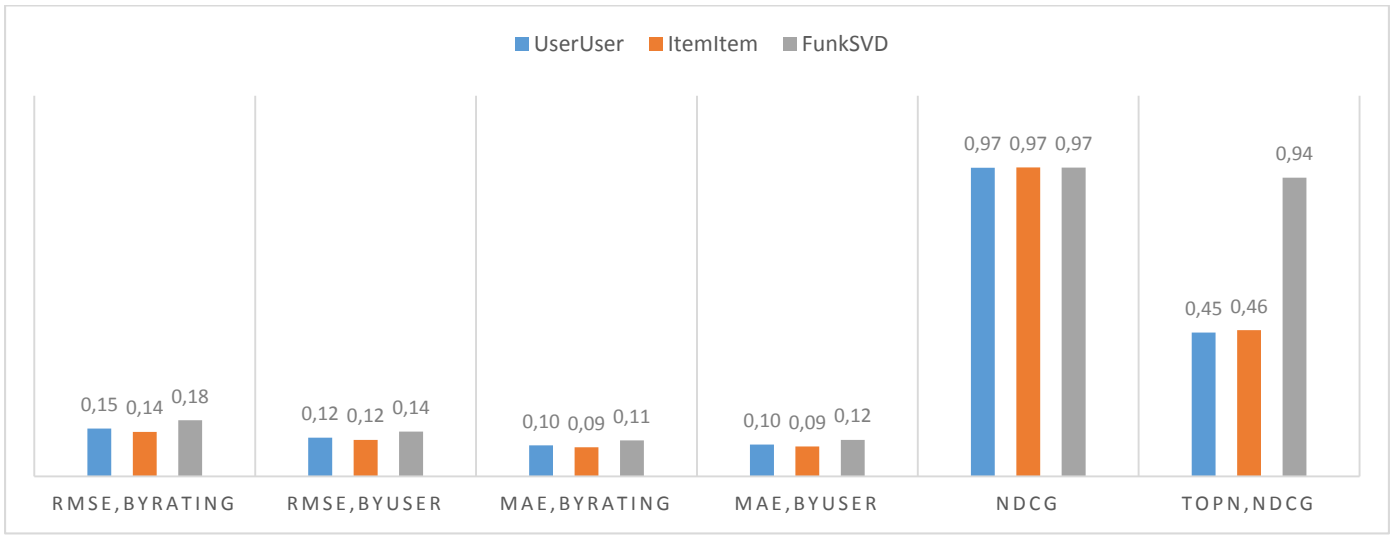


Διάγραμμα 6.14 οι μέσοι χρόνοι για την εκπαίδευση και την αξιολόγηση των 3 αλγορίθμων στον γράφο user-item alternative του delicious

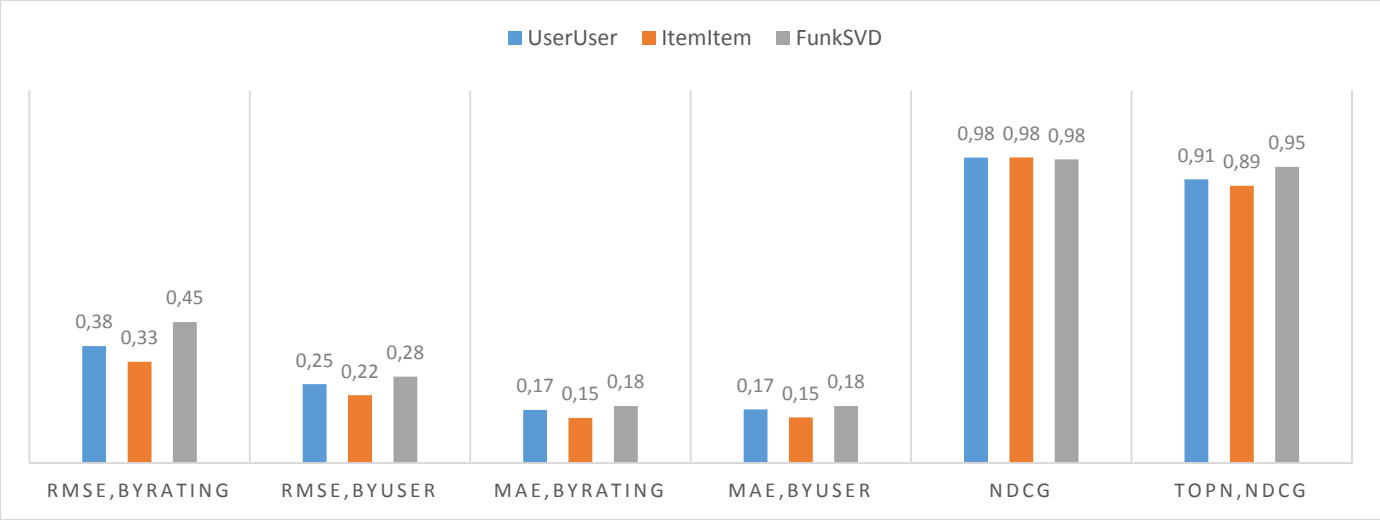


Διάγραμμα 6.15 οι μέσοι χρόνοι για την εκπαίδευση και την αξιολόγηση των 3 αλγορίθμων στον γράφο users του delicious

Από τα παρακάτω αποτελέσματα των μετρικών δεν φαίνεται να υπάρχει μεγάλη διαφορά μεταξύ των αλγορίθμων και για τους δυο γράφους.

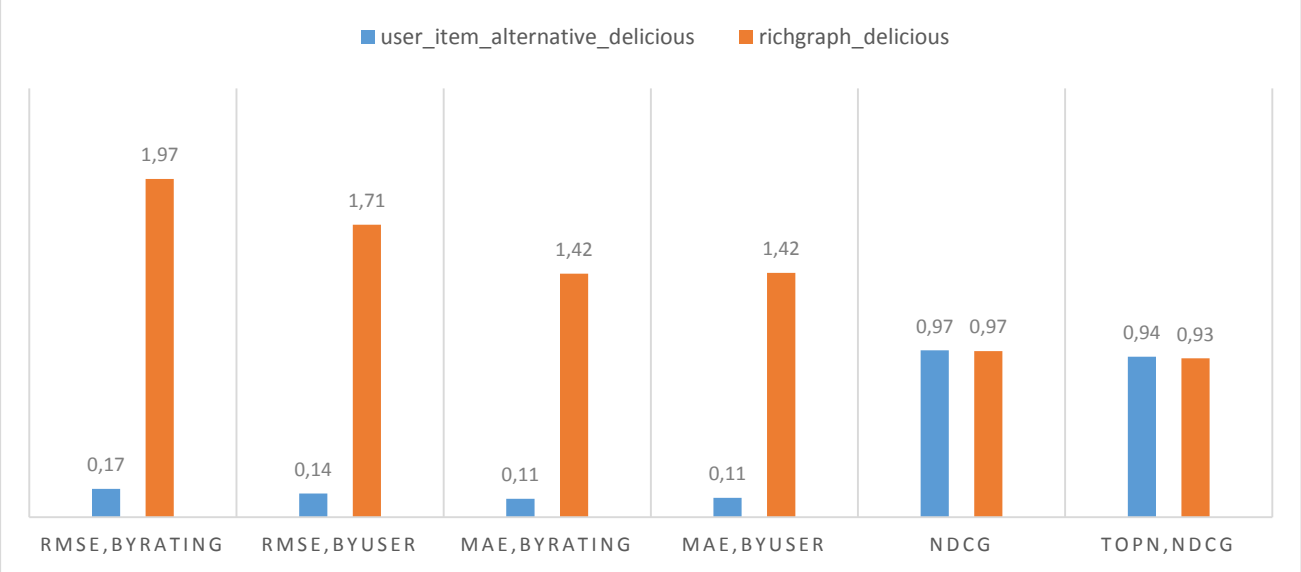


Διάγραμμα 6.16 αποτελέσματα μετρικών για τους τρείς αλγορίθμους στον γράφο useritem alternative του delicious



Διάγραμμα 6.17 αποτελέσματα μετρικών για τους τρείς αλγορίθμους στον γράφο users του delicious

Όσο αναφορά την αξιολόγηση του richgraph σε σχέση με τον user-item, η διαφορά μοιάζει με αυτή του προηγούμενου πειράματος, δείχνοντας μεγάλα ποσοστά λάθους όταν εκτελείται ο αλγόριθμος στον γράφο richgraph.

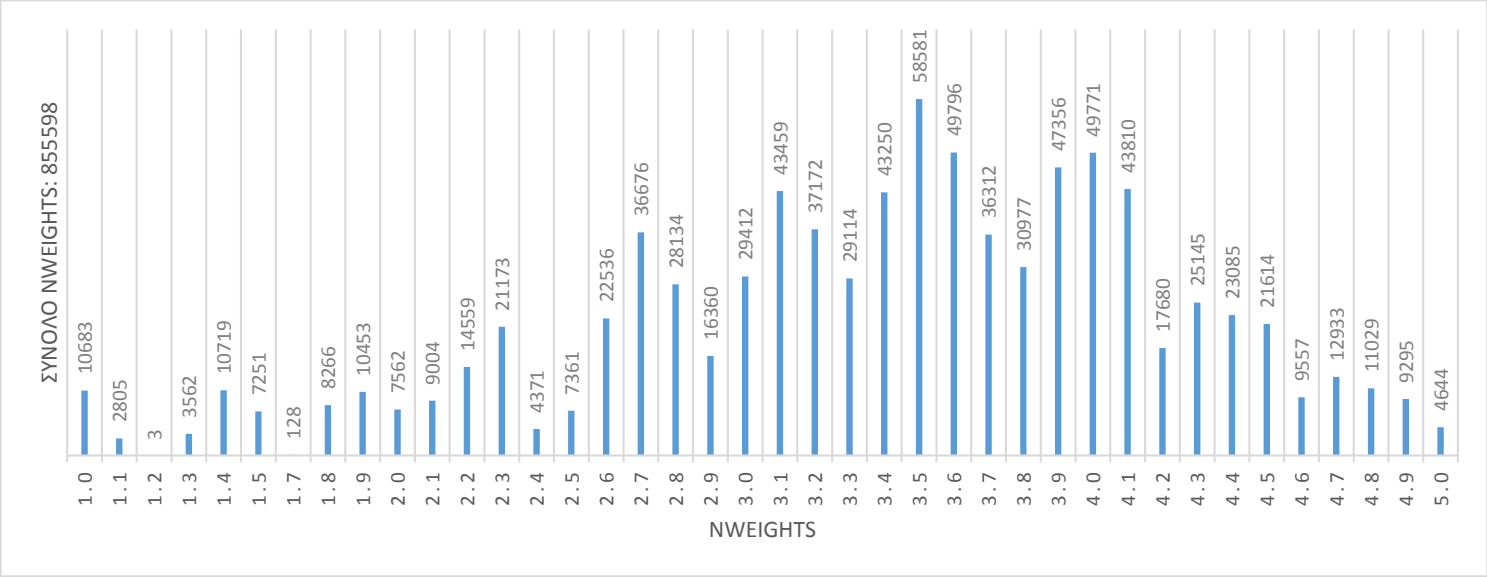


Διάγραμμα 6.18 αποτελέσματα μετρικών για τον αλγόριθμο FunkSVD στους γράφους user-item και richgraph, του delicious

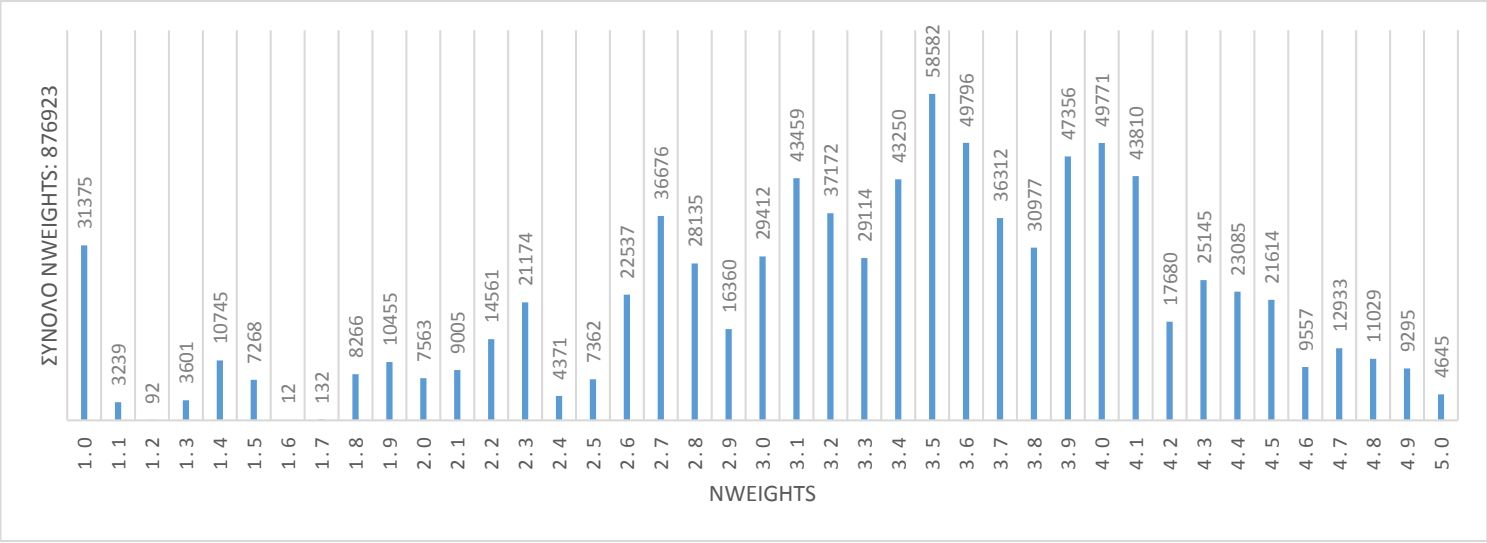
Movielens

Τέλος, παρακάτω φαίνεται το σύνολο των nWeights, οι τιμές των nWeights και η συχνότητα εμφάνισης αυτών, για κάθε γράφο του movielens.

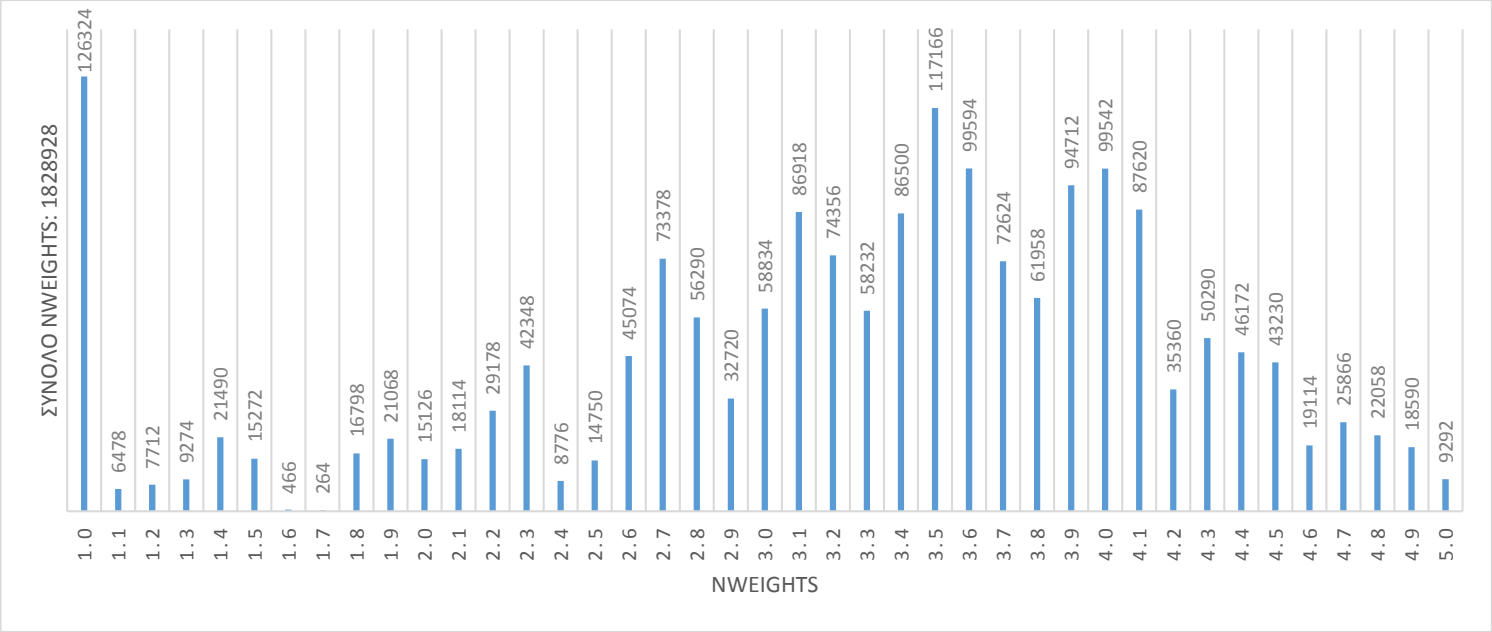
Σε αντίθεση με τα προηγούμενα σύνολα δεδομένων, η κατανομή των nWeights και στους τρεις γράφους του movielens είναι ομοιόμορφή. Επίσης, να υπενθυμίσουμε ότι ο γράφος users του movielens δεν περιλαμβάνει τον κοινωνικό γράφο user-user (βλ. Πίνακας 6.1).



Διάγραμμα 6.19 nWeights user - item movielense

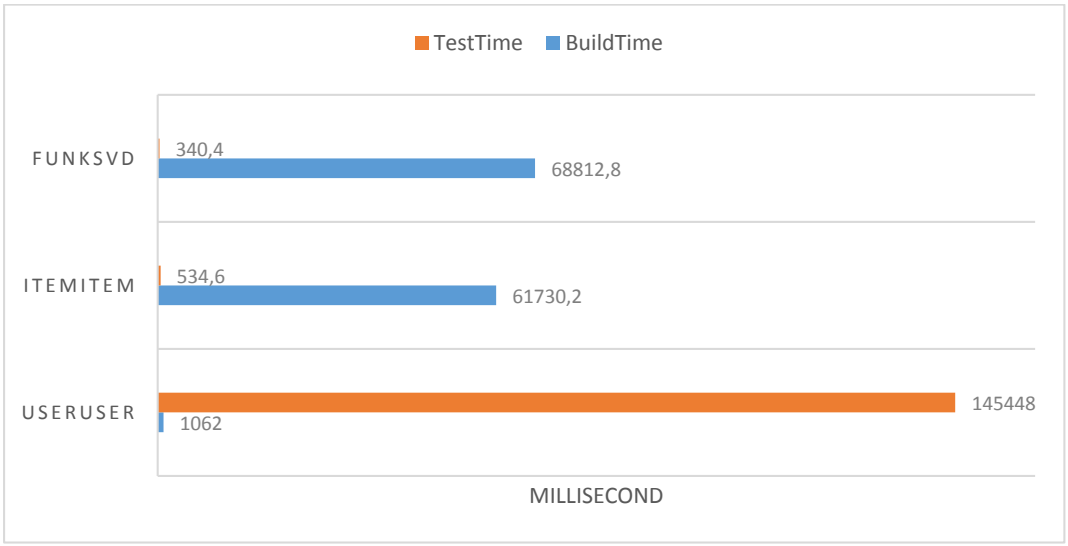


Διάγραμμα 6.20 nWeights users movielens

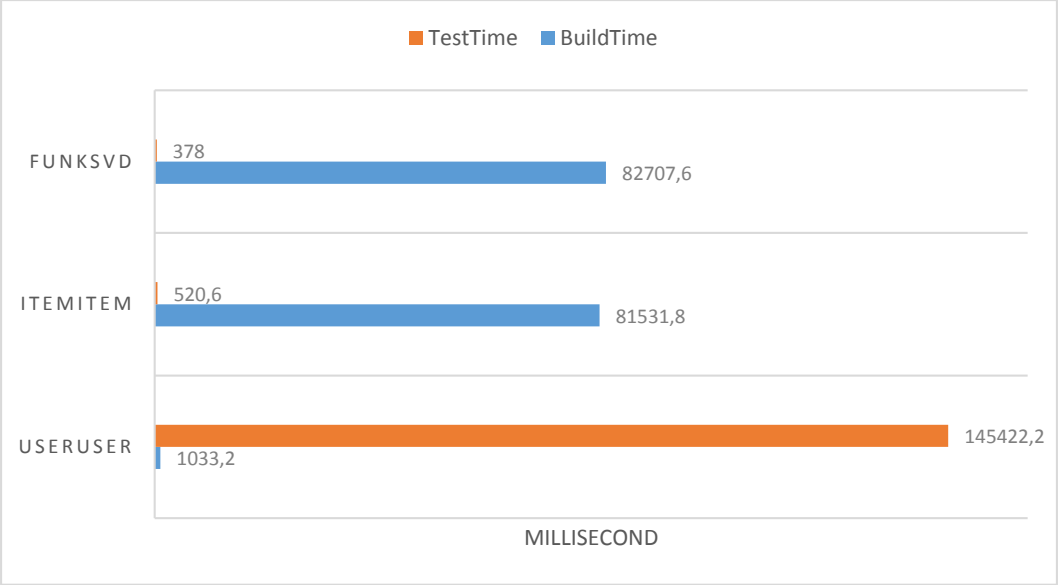


Διάγραμμα 6.21 nWeights richgraph movielens

Στην συνέχεια παρουσιάζονται τα αποτελέσματα από την αξιολόγηση των τριών αλγορίθμων για τους γράφους user-item και users του movielens. Στο πρώτο διάγραμμα φαίνονται οι μέσοι χρόνοι (build και test time) για τα πέντε τμήματα του συνόλου των δεδομένων. Ο αλγόριθμος UserUser και στους δυο γράφους χρειάζεται περισσότερο χρόνο για αξιολόγηση σε σχέση με τους άλλους δυο.

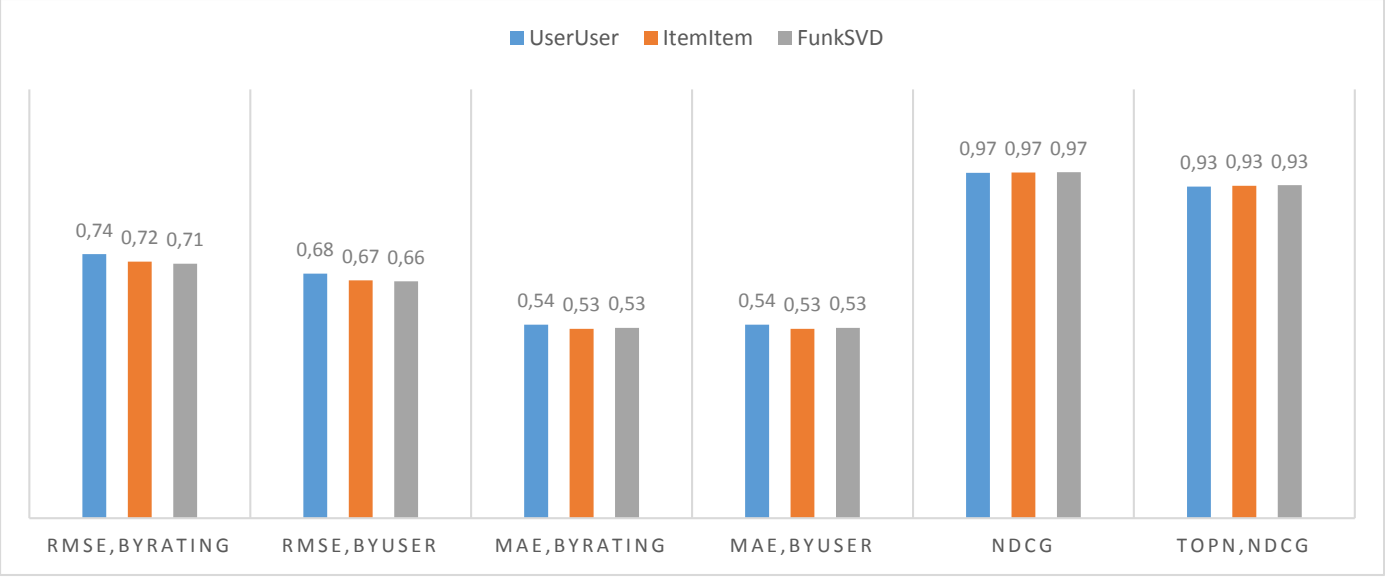


Διάγραμμα 6.22 οι μέσοι χρόνοι για την εκπαίδευση και την αξιολόγηση των 3 αλγορίθμων στον γράφο user-item του movielens

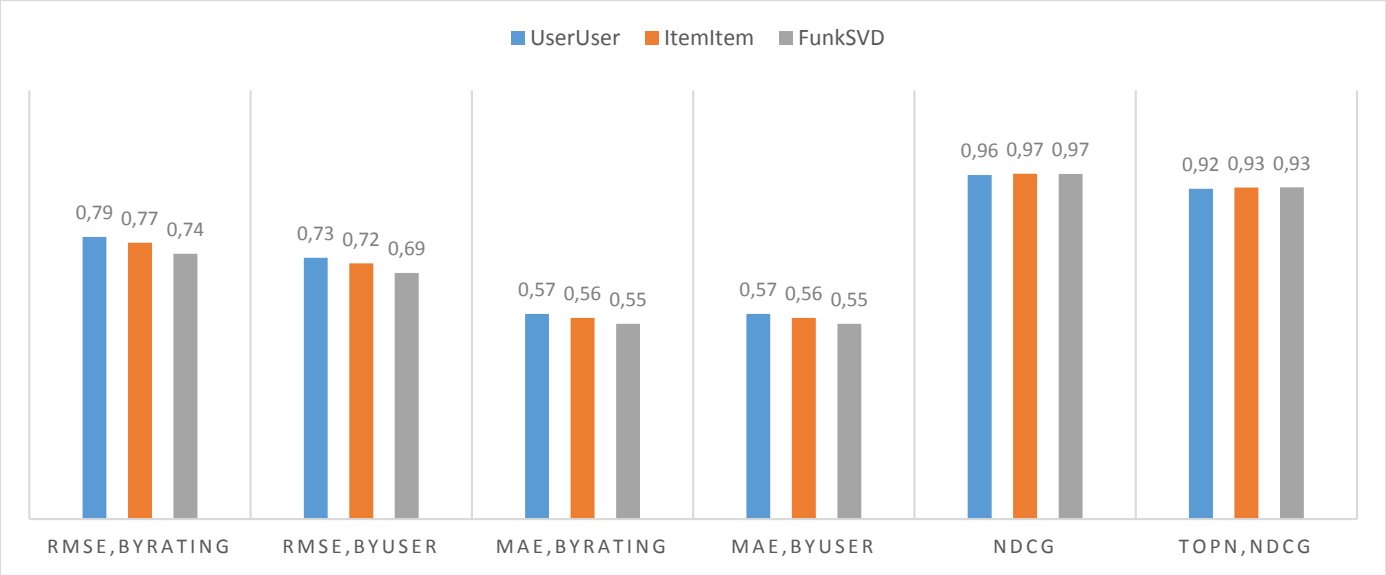


Διάγραμμα 6.23 οι μέσοι χρόνοι για την εκπαίδευση και την αξιολόγηση των 3 αλγορίθμων στον γράφο users του movielens

Στα επόμενα δυο διαγράμματα παρουσιάζεται η αξιολόγηση των αλγορίθμων για τους γράφους user-item και users. Από τα αποτελέσματα φαίνεται ότι δεν υπάρχει ουσιαστική διαφορά ανάμεσα τους.

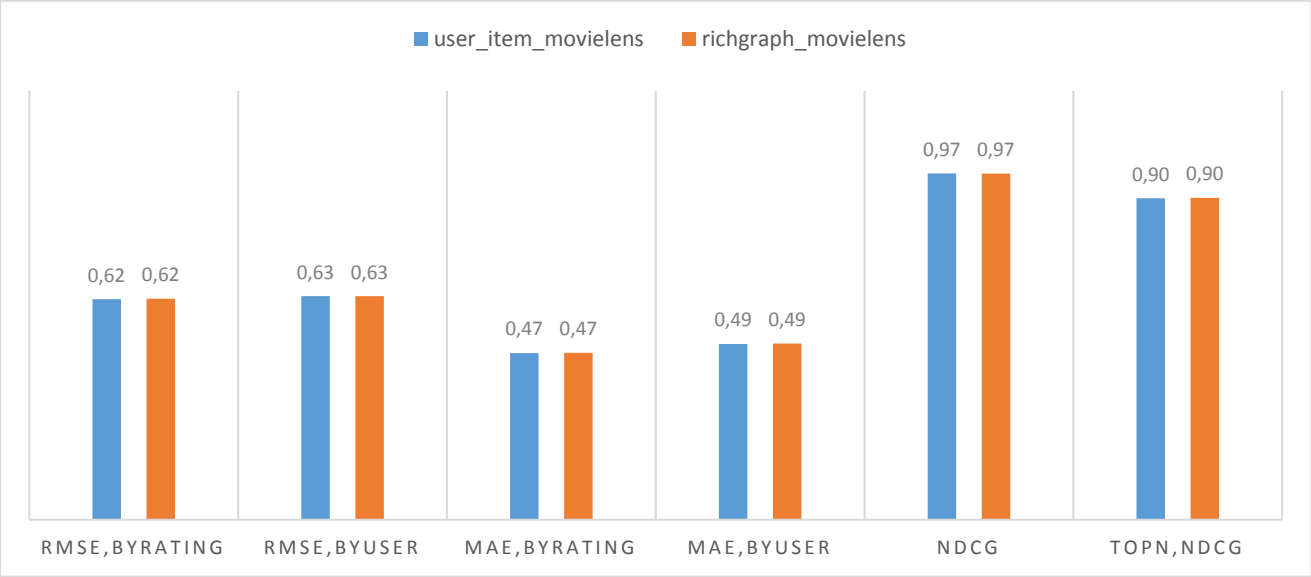


Διάγραμμα 6.24 αποτελέσματα μετρικών για τους τρείς αλγορίθμους στον γράφο user-item του movielens



Διάγραμμα 6.25 αποτελέσματα μετρικών για τους τρείς αλγορίθμους στον γράφο users του movielens

Το ίδιο ισχύει και στην αξιολόγηση του richgraph σε σχέση με τον user-item, όπου τα αποτελέσματα είναι ακριβώς τα ίδια.



Διάγραμμα 6.26 αποτελέσματα μετρικών για τον αλγόριθμο FunkSVD στους γράφους user-item και richgraph, του movielens

6.3 Συμπεράσματα πειραματικής διαδικασίας

Το κομμάτι της πειραματικής διαδικασίας απλοποιείται σημαντικά με την βοήθεια του εργαλείου Lenskit. Το config αρχείο (eval.groovy) δίνει την δυνατότητα στον χρήστη να εκτελέσει πολλά παραδείγματα, χωρίς να επέμβει στον κώδικα του προγράμματος. Αυτό βοήθησε και στην δικιά μας περίπτωση, καθώς δεν χρειάστηκε κάποια άλλη διαδικασία για να αξιολογηθούν οι παραγόμενοι γράφοι από το προτεινόμενο εργαλείο, παρά μόνο να γίνει η εισαγωγή τους στο Lenskit.

Από τα παραπάνω πειράματα, το σύνολο δεδομένων του movielens ήταν το μόνο αξιόλογο προς εξέταση παράδειγμα, με ομοιόμορφη κατανομή βαρών και με πυκνότητα δεδομένων ίση με 0.4 στον πίνακα user-item. Παρόλα αυτά, δεν περιείχε κοινωνικό γράφο, συνεπώς στον γράφο users είχαν προστεθεί επιπλέον μόνο οι ετικέτες σε σχέση με τον user-item. Τα αποτελέσματα των μετρικών για τους γράφους users και user-item ήταν σχεδόν τα ίδια, δείχνοντας ότι μπορούν να γίνουν ταυτόχρονα προβλέψεις και για αντικείμενα αλλά και για ετικέτες. Όσα αναφορά την σύγκριση του richgraph με τον user-item, ο FunkSVD έδωσε τα ίδια αποτελέσματα.

Αντιθέτως, στα άλλα σύνολα δεδομένων με ανομοιόμορφη κατανομή βαρών και με μηδενική σχεδόν πυκνότητα δεδομένων στον πίνακα user-item, τα αποτελέσματα ήταν απογοητευτικά. Τα αποτελέσματα από τον FunkSVD για τον richgraph σε σύγκριση με τον user-item έδειξαν μεγάλες αποκλίσεις με μεγάλα ποσοστά λάθους (rmse>1). Παρόλα αυτά, ο γράφος users έδειξε και σε αυτήν την περίπτωση ότι δουλεύει, καθώς τα ποσοστά λάθους για όλους τους αλγορίθμους ήταν σε λογικά πλαίσια.

Συνοψίζοντας, μπορεί τα αποτελέσματα από τις αξιολογήσεις να μην ήταν ιδανικά, παρόλα αυτά το προτεινόμενο εργαλείο δημιουργίας εμπλουτισμένου γράφου είναι μια αρχή για επιπλέον πειράματα.

7. Επίλογος και Μελλοντικές Επεκτάσεις

Στα πλαίσια της παρούσας εργασίας προτάθηκε ένας τρόπος για την δημιουργία ενός εμπλουτισμένου γράφου από την ύπαρξη τριών διαφορετικών γράφων: ενός κοινωνικού γράφου-μονομερή (user-user), ενός γράφου αξιολόγησης - διμερή (user-item) και ενός γράφου προσθήκης ετικετών-τριμερή γράφου (user-item-tag). Η πειραματική διαδικασία έδειξε ότι στον γράφο users, οι αλγόριθμοι συστάσεων (UserUser, ItemItem, FunkSVD) μπορούν να τρέξουν εξίσου σωστά, όπως σε έναν απλό user-item γράφο. Ο εμπλουτισμένος γράφος (richmatrix), στον οποίο εκπαιδεύτηκε ο FunkSVD και στην συνέχεια αξιολογήθηκε στον user-item, δεν έφερε σωστά αποτελέσματα για το σύνολο δεδομένων του lastfm και του delicious. Παρόλα αυτά, στο σύνολο δεδομένων του movielens όπου τα βάρη ήταν ομοιόμορφα κατανεμημένα και η πυκνότητα των δεδομένων ήταν αρκετά καλύτερη σε σχέση με τα άλλα δυο παραδείγματα, τα αποτελέσματα ήταν ακριβώς τα ίδια, δείχνοντας ότι υπάρχει λόγος για περαιτέρω έρευνα.

Υπάρχουν όμως και ορισμένες ελλείψεις σχετικά με το προτεινόμενο εργαλείο (βλ. κεφ. 4). Κατά αρχήν δεν έχει ληφθεί υπόψη εάν ο κοινωνικός γράφος περιέχει κατευθυνόμενες ακμές, που ουσιαστικά θα πρέπει να μετατραπούν σε μη κατευθυνόμενες. Επίσης, δεν έχει περιληφθεί η περίπτωση να υπάρχουν ίδιες ακμές στον κοινωνικό γράφο ή στον γράφο αξιολόγησης. Για παράδειγμα, ένας χρήστης ήταν φίλος με κάποιον και σε επόμενη χρονική στιγμή δεν είναι ή όταν ένας χρήστης αλλάζει την αξιολόγηση του για ένα αντικείμενο. Μια πιθανή λύση για το τελευταίο θα μπορούσε να είναι η εύρεση του μέσου όρου.

Πολύ σημαντική είναι όμως και η εύρεση της σχέσης user-user, όταν δεν δίνεται από το σύνολο δεδομένων (π.χ. movielens) και των σχέσεων item-item και tag-tag, που στο συγκεκριμένο εργαλείο είναι κενές (βλ. Εικόνα 4.1). Μια πιθανή λύση θα ήταν η χρήση μιας συνάρτησης ομοιότητας για τις ετικέτες και αντίστοιχα και για τα αντικείμενα.

Επιπροσθέτως, ορισμένες αξιοσημείωτες αλλαγές που μπορούν να γίνουν στο συγκεκριμένο εργαλείο είναι η πρόταση διαφορετικών μεθόδων υπολογισμού της κανονικοποίησης των βαθμολογιών-βαρών, καθώς στην παρούσα προσέγγιση χρησιμοποιήθηκε η min-max κανονικοποίηση, η οποία δεν βοηθάει στις περιπτώσεις όπου το max είναι πολύ μεγάλο και το μεγαλύτερο ποσοστό των τιμών βρίσκεται κοντά στο min. Επιπλέον, θα μπορούσε να προταθεί εναλλακτικός τρόπος παραγωγής των βαρών για τους γράφους user-user και user-item σε σχέση με τον υπάρχοντα (βλ. Εικόνα 4.6). Καθώς επίσης και για τον τρόπο παραγωγής βαρών για τους γράφους user-tag και item-tag, όπου στην παρούσα κατάσταση πραγματοποιείται μια συνάθροιση των ετικετών ανά χρήστη και ανά αντικείμενο για να παραχθούν οι συγκεκριμένοι γράφοι. Εκτός αυτών, πολύ σημαντική είναι και η περαιτέρω αξιοποίηση του τριμερή γράφου, όπως για παράδειγμα η χρήση των χρονοσφραγίδων που εμπεριέχονται συνήθως σε αυτόν.

Όσο αναφορά την προγραμματιστική πλατφόρμα Lenskit (βλ. κεφ. 5.1), χρησιμοποιήθηκαν οι προεπιλεγμένες ρυθμίσεις και για τους τρείς αλγορίθμους. Πιθανότατα η δοκιμή διαφορετικών ρυθμίσεων για κάθε γράφο των συνόλων δεδομένων, θα μπορούσε να επηρεάσει τα αποτελέσματα. Επιπλέον, ένα άλλο πείραμα το οποίο θα είχε σημασία να πραγματοποιηθεί είναι

η εφαρμογή του αλγορίθμου UserUser στον γράφο users και η αξιολόγηση του στον γράφο user-item, κάτι αντίστοιχο δηλαδή που έγινε με τον αλγόριθμο FunkSVD και τον γράφο richgraph.

Τέλος, πολύ ενδιαφέρουσα θα ήταν η χρήση και η επέκταση του προτεινόμενου εργαλείου σε διαφορετικού είδους τριμερής γράφους, όπως για παράδειγμα «ηθοποιός – ταινία – ρόλος ταινίας» ή «χρήστης – τοποθεσία – δραστηριότητα», κ.λπ. Ίσως, μελλοντικά να προταθεί και ένα ενιαίο μοντέλο, όπου για οποιονδήποτε τριμερή γράφο θα μπορεί να δημιουργήσει εμπλουτισμένο γράφο, στον οποίο θα αποτυπώνεται όλη η πληροφορία (φανερή και μη) με σκοπό την βελτίωση των συστάσεων.

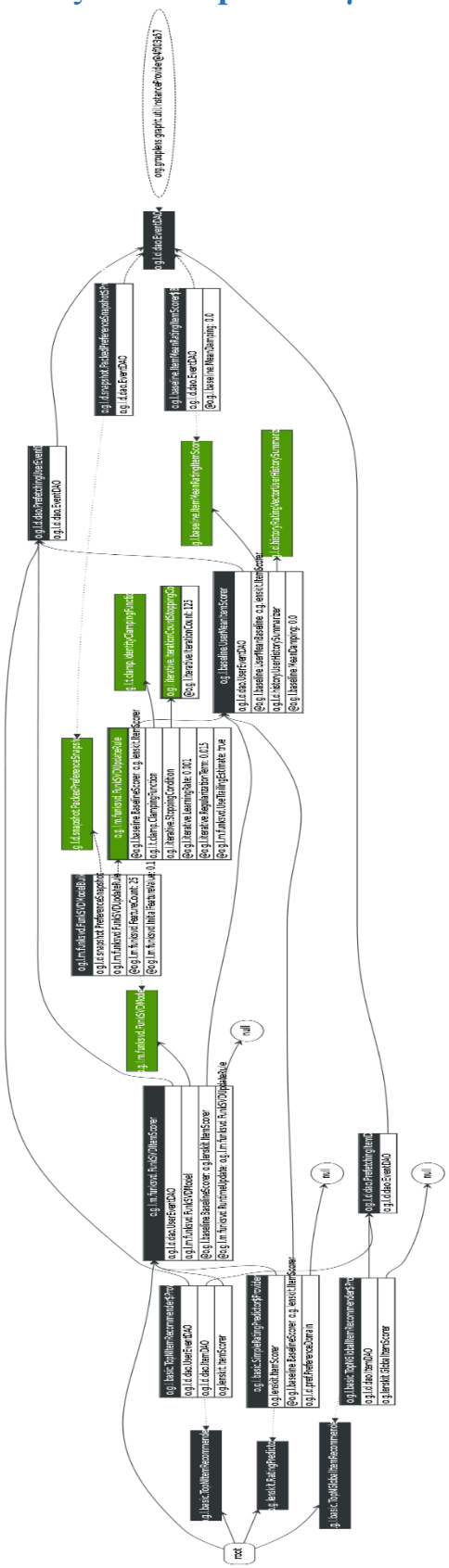
8. Παραρτήματα

8.1 pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-
v4_0_0.xsd">

  <modelVersion>4.0.0</modelVersion>
  <groupId>hua.richgraph.lenskit</groupId>
  <artifactId>1</artifactId>
  <version>1.0-SNAPSHOT</version>
  <packaging>jar</packaging>

  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <lenskit.eval.scriptDir>${basedir}/src/eval/</lenskit.eval.scriptDir>
    <lenskit.eval.dataDir>${basedir}/target/data/</lenskit.eval.dataDir>
    <lenskit.eval.analysisDir>${basedir}/target/analysis/</lenskit.eval.analysisDir>
    <lenskit.eval.publishDir>${basedir}/src/paper/</lenskit.eval.publishDir>

    <!-- The version of LensKit for your project -->
    <!-- Note that we use project.version to pick up the version from the
         source tree this archetype is being built from. -->
    <lenskit.version>2.1-M2</lenskit.version>

    <!-- The location of the Rscript executable. The default is to search
         for Rscript in your PATH. If you need to override this, it should probably
         be done in ~/.m2/settings.xml rather than here. -->
    <rscript.executable>Rscript</rscript.executable>

    <!-- Uncomment and change below to 'yes' if you accept the MovieLens
         data terms of service. -->
    <!-- <grouplens.mldata.acknowledge>no</grouplens.mldata.acknowledge> -->
  </properties>

  <dependencies>

    <!-- You will automatically get a version of LensKit for your project,
    pulled in transitively by the lenskit-eval-maven-plugin. The version you
    get is based on the version of the archetype you used to create this project. -->

    <dependency>
      <groupId>org.grouplens.lenskit</groupId>
      <artifactId>lenskit-all</artifactId>
      <version>${lenskit.version}</version>
    </dependency>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.11</version>
      <scope>test</scope>
    </dependency>
    <dependency>
      <groupId>org.hamcrest</groupId>
      <artifactId>hamcrest-library</artifactId>
      <version>1.3</version>
      <scope>test</scope>
    </dependency>

  </dependencies>

  <!-- Now we specify the build that will actually execute to make our project
       do something cool. The build is a collection of plugins that we execute to
       do the work. Note that we run the evaluations using the lenskit-lifecycle,
       because we use the lenskit-eval-maven-plugin. The phases of this lifecycle
       are lenskit-pre-eval, lenskit-eval, lenskit-post-eval, and lenskit-analyze,
```

```
       in that order. -->

  <build>
    <plugins>

      <!-- LensKit requires at least Java 1.6 -->
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-compiler-plugin</artifactId>
        <version>2.0.2</version>
        <configuration>
          <source>1.8</source>
          <target>1.8</target>
        </configuration>
      </plugin>

      <!-- Run the lenskit eval plugin that does the work of evaluating the
      recommenders you are comparing, by running your Groovy script. -->
      <plugin>
        <artifactId>lenskit-eval-maven-plugin</artifactId>
        <groupId>org.grouplens.lenskit</groupId>
        <version>${lenskit.version}</version>
        <extensions>true</extensions>

    <!-- The default execution for the lenskit-eval-maven-plugin
         will run the run-eval goal in the lenskit-eval phase to
         run your Groovy script to evaluate the recommenders you
         are comparing.  The configuration here configures the
         plugin, including all of its MOJOs.  -->

        <configuration>
          <script>${lenskit.eval.scriptDir}/eval.groovy</script>
          <targets>
            <target>draw</target>
            <target>analyze</target>
          </targets>
        </configuration>
      </plugin>

      <!-- This plugin runs LaTeX to produce the PDF version of our paper.
           We tie this plugin to the appropriate phase of the lenskit lifecycle. -->

      <plugin>
        <groupId>org.codehaus.mojo</groupId>
        <artifactId>latex-maven-plugin</artifactId>
        <version>1.1</version>
        <executions>
          <execution>
            <id>do-latex</id>
            <phase>lenskit-publish</phase>
            <goals>
              <goal>latex</goal>
            </goals>
          </execution>
        </executions>
      </plugin>

    </plugins>
  </build>
</project>
```

8.2 Διάγραμμα επικοινωνίας των components για τον FunkSVD



9. Βιβλιογραφία

Breese, J., Heckerman, D., & Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence* (pp. 43-52). Morgan Kaufmann Publishers Inc.

Barbieri, N., Manco, G., & Ritacco, E. (2014). Probabilistic Approaches to Recommendations. In *Synthesis Lectures on Data Mining and Knowledge Discovery* (pp. 1-197).

Billsus, D., & Pazzani, M. J. (1998). Learning Collaborative Information Filters. In *ICML* (pp. 46-54).

Burke, R. (2007). Hybrid Web Recommender Systems. In *The Adaptive Web* (pp. 377-408). Springer Berlin Heidelberg.

Durao, F., & Dolog, P. (2012). A Personalized Tag-Based Recommendation in Social Web Systems. *arXiv preprint arXiv:1203.0332*.

Ekstrand, M., Ludwig, M., Konstan, J., & Riedl, J. (2011). Rethinking the recommender research ecosystem: reproducibility, openness, and LensKit. In *Proceedings of the fifth ACM conference on Recommender systems* (pp. 133-140). ACM.

Fan, W., & Bifet, A. (2013). Mining big data: current status, and forecast to the future. *ACM SIGKDD Explorations Newsletter*, *14*(2), pp. 1-5.

Funk, S. (2006). *Netflix Update: Try This at Home*. Retrieved from http://sifter.org/~simon/journal/20061211.html

Goldberg, D., Nichols, D., M. Oki, B., & Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, *35*(12), pp. 61-70.

Gunawardana, A., & Shani, G. (2009). A Survey of Accuracy Evaluation Metrics of Recommendation Tasks. *The Journal of Machine Learning Research*, *10*, pp. 2935-2962.

Herlocker, J., Konstan, J., & Riedl, J. (2002). An Empirical Analysis of Design Choices in Neighborhood-Based Collaborative Filtering Algorithms. *Information Retrieval*, *5*(4), pp. 287-310.

Hijikata , Y., Shimizu, T., & Nishida, S. (2009). Discovery-oriented collaborative filtering for improving user satisfaction. In ACM (Ed.), *Proceedings of the 14th international conference on Intelligent user interfaces* (pp. 67-76). ACM.

Hu, X., Meng, X., & Wang, L. (2011). SVD-based group recommendation approaches: an experimental study of Moviepilot. In *Proceedings of the 2nd Challenge on Context-Aware Movie Recommendation* (pp. 23-28). ACM.

Karypis, G. (2001). Evaluation of Item-Based Top-N Recommendation Algorithms. In *Proceedings of the tenth international conference on Information and knowledge management* (pp. 247-254). ACM.

Lee, S., Chung, T., & McLeod, D. (2011). Dynamic Item Recommendation by Topic Modeling for Social Networks. In *Information Technology: New Generations (ITNG), 2011 Eighth International Conference on* (pp. 884-889). IEEE.

M. McNee, S., Albert, I., Cosley, D., Gopalkrishnan , P., Lam, S., Rashid, A., . . . Riedl, J. (2002). On the recommending of citations for research papers. In *Proceedings of the 2002 ACM conference on Computer supported cooperative work* (pp. 116-125). ACM.

Pan, R., Xu, G., Fu, B., Dolog, P., Wang, Z., & Leginus, M. (2012). Improving Recommendations by the Clustering of Tag Neighbours. *Journal of Convergence, 3*(1), pp. 13-20.

Pazzani, M., & Billsus, D. (2007). Content-Based Recommendation Systems. In P. Brusilovsky, A. Kobsa, & W. Nejdl (Eds.), *The Adaptive Web* (Vol. 4321, pp. 325-341). Springer Berlin Heidelberg.

Rawashdeh, M., Kim, H.-N., & Mohamad Alj, J. (2013). Folksonomy link prediction based on a tripartite graph for tag recommendation. *Journal of Intelligent Information Systems, 40*(2), pp. 307-325.

Robillard, M., Maalej, W., Walker, R., & Zimmermann, T. (2014). *Recommendation Systems in Software Engineering* (1 ed.). Springer-Verlag Berlin Heidelberg.

Said, A., & Bellogín, A. (2014). Comparative recommender system evaluation: benchmarking recommendation frameworks. In *Proceedings of the 8th ACM Conference on Recommender systems* (pp. 129-136). ACM.

Sarwar, B., Karypis, G., Konstan, J., & Riedl , J. (2000). *Application of Dimensionality Reduction in Recommender System - A Case Study*. Minnesota Univ Minneapolis, Computer Science.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web* (pp. 285-295). ACM.

Schafer, J., Frankowski, D., Herlocker, J., & Sen, S. (2007). Collaborative Filtering Recommender Systems. In *The Adaptive Web* (pp. 291-324). Springer Berlin Heidelberg.

Setten, M. v. (2005). *Supporting people in finding information: hybrid recommender systems and goal-based structuring.* PhD thesis, Telematica Instituut.

Sharma, L., & Gera, A. (2013). A Survey of Recommendation System: Research Challenges. *International Journal of Engineering Trends and Technology (IJETT), 4*(5), pp. 1989-1992.

Wang, Z., Tan, Y., & Zhang, M. (2010). Graph-based Recommendation on Social Networks. In *Web Conference (APWEB), 2010 12th International Asia-Pacific* (pp. 116-122).

Yuan, Z., Yu, T., & Zhang, J. (2011). A Social Tagging Based Collaborative Filtering Recommendation Algorithm for Digital Library. In *Digital Libraries: For Cultural Heritage, Knowledge Dissemination, and Future Creation* (pp. 192-201). Springer Berlin Heidelberg.

Zhang, Y., Zhang, B., Gao, K., Guo, P., & Sun, D. (2012). Combining content and relation analysis for recommendation in social tagging systems. *Physica A: Statistical Mechanics and its Applications, 391*(22), 5759–5768.