ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Σχεδίαση και Ανάπτυξη Εφαρμογής Συστήματος Συστάσεων Κινηματογραφικών Ταινιών για Κινητά Τηλέφωνα

Ιωάννης Μ. Καδιανάκης

Επιβλέπουσα : **Καράλη Ιζαμπώ,** Επίκουρη Καθηγήτρια

ΑΘΗΝΑ

Ιούλιος 2018

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Σχεδίαση και Ανάπτυξη Εφαρμογής Συστήματος Συστάσεων Κινηματογραφικών Ταινιών για Κινητά Τηλέφωνα

Ιωάννης Μ. Καδιανάκης
Α.Μ.: 1115200900090

ΕΠΙΒΛΕΠΟΥΣΑ: **Καράλη Ιζαμπώ,** Επίκουρη Καθηγήτρια

ΠΕΡΙΛΗΨΗ

Ο 21ος αιώνας έχει χαρακτηριστεί ως ο “Αιώνας της Πληροφορίας”. Καθημερινά ο άνθρωπος γίνεται δέκτης αμέτρητων πληροφοριών, ηχητικών και οπτικών ερεθισμάτων που προέρχονται από εργαλεία που χρησιμοποιεί σε καθημερινή βάση. Η τηλεόραση, το ραδιόφωνο και το διαδίκτυο είναι μερικά από αυτά, με το διαδίκτυο να αποτελεί το κορυφαίο μέσο ψυχαγωγίας, ενημέρωσης, κοινωνικής συναναστροφής κ.α.

Είναι μεγάλης σημασίας το φιλτράρισμα των μηνυμάτων που δεχόμαστε για να βρει κάποιος ειδήσεις που τον ενδιαφέρουν, ταινίες που θα ήθελε να δει, μουσική που ταιριάζει με την τρέχουσα διάθεσή του – ικανοποιώντας έτσι τις επιθυμίες του. Συστήματα σκοπός των οποίων είναι το φιλτράρισμα των πληροφοριών αυτών, είναι τα Συστήματα Συστάσεων, τα οποία έχουν εφαρμογή σε πληθώρα προγραμμάτων.

Η εργασία αυτή περιγράφει τη μελέτη των Συστημάτων Συστάσεων και την υλοποίηση ενός τετοιου συστήματος στη μορφή εφαρμογής για κινητό Android. Το Σύστημα Σύστασης που πραγματοποιήσαμε αποτελείται από έναν αλγόριθμο Συνεργατικού Φιλτραρίσματος που προτείνει στο χρήστη ταινίες, βάσει βαθμολογιών που έχει δώσει.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Φιλτράρισμα Πληροφορίας

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Συστήματα Σύστασεων, Συνεργατικό Φιλτράρισμα, Εφαρμογές Κινητών, Κινηματογραφικές Ταινίες

ABSTRACT

The 21st century can be described as the “Age of Information”. On a daily basis human is a recipient of a great amount of data, sound and visual stimuli produced by his everyday tools. Television, radio and internet are some of them, with the internet being the first choice of entertainment, news, social interaction etc.

Filtering is of great importance for someone trying to find news that concern them, movies that they would like to watch, music fitting their current mood – so as to fulfill their wishes. Systems that aim to filter this information, are Recommendation Systems, which can be used in many applications.

This thesis describes the process of developing an Android Application. Through a Collaborative Filtering algorithm, the application recommends movies to a user, taking his/her past ratings into account.

SUBJECT AREA: Information Filtering

KEYWORDS: Recommendation Systems, Collaborative Filtering, Android Applications, Movies

ΕΥΧΑΡΙΣΤΙΕΣ

Για τη διεκπεραίωση της παρούσας Πτυχιακής Εργασίας, θα ήθέλα να ευχαριστήσω την επιβλέπουσα, επίκουρη καθηγήτρια Ιζαμπώ Καράλη για τη συνεργασία, την υπομονή της και την πολύτιμη συμβολή της στην ολοκλήρωση της. Επίσης θα ήθελα να ευχαριστήσω τους κοντινούς μου ανθρώπους για τη στήριξη τους καθ' όλη τη διάρκεια αυτής.

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

ΠΡΟΛΟΓΟΣ

Η εργασία ξεκίνησε να γράφεται το Νοέμβριο του 2017, με την ολοκλήρωσή της να γίνεται τον Ιούλιο του 2018.

Η παρούσα Πτυχιακή Εργασία εκπονήθηκε στα πλαίσια του προπτυχιακού προγράμματος σπουδών του Τμήματος Πληροφορικής και Τηλεπικοινωνιών του Εθνικού και Καποδιστριακού Πανεπιστημίου Αθηνών, κατά το ακαδημαϊκό έτος 2017-2018, υπό την επίβλεψη της επίκουρης καθηγήτριας Ιζαμπώ Καράλη.

1. ΕΙΣΑΓΩΓΗ

Είναι συχνό φαινόμενο σε αγορές που πρόκειται να κάνουμε ή σε ταινιές που πάμε να δούμε, να αναζητούμε προτάσεις από ανθρώπους που εμπιστευόμαστε ή ανθρώπους που γνωρίζουμε ότι έχουν εμπειρία. Στο διαδικτυακό κόσμο συχνά οι χρήστες βλέπουν καρτέλες με τίτλο: “προϊόντα που ίσως σας ενδιαφέρουν”, χωρίς να έχουν προβεί οι ίδιοι σε κάποια κίνηση. Αυτή η πρόταση γίνεται από τα Συστήματα Συστάσεων (Recommendation Systems)(ΣΣ), δηλαδή εφαρμογές που υπολογίζουν το πόσο μπορεί να αρέσει ένα αντικείμενο σε κάποιο χρήστη.

Οι υλοποιήσεις τέτοιων συστημάτων μπορούν να εφαρμοστούν σε πωλήσεις βιβλίων, ειδήσεις, διαδικτυακές αναζητήσεις, προτάσεις ταινιών και εστιατορίων. Εταιρείες όπως το Twitter, το Netflix, η Google, το Amazon [18, 19, 20, 21] προσπαθούν να δώσουν λύση στους χρήστες– εκπληρώνοντας και προβλέποντας τις επιθυμίες τους. Είναι ασταμάτητος ο αγώνας για την πιο εύστοχη συμβουλή σε κάποιον αγοραστή ή κάποια φαν τηλεοπτικής σειράς, που νιώθει ότι τα “θέλω” της συμβαδίζουν με τα προϊόντα που προτείνονται. Ένας ικανοποιημένος πελάτης είναι σίγουρο ότι θα ξαναεμπιστευτεί τη συγκεκριμένη εταιρεία για την διασκέδασή του ή αγορές του, αυξάνοντας έτσι τα κέρδη για την εταιρεία [22]. Γι' αυτό και πληθώρα οργανισμών επιστρατεύει εξελιγμένους αλγόριθμους για να συνδυάσουν τεράστιες ποσότητες δεδομένων προερχόμενες από εκατομμύρια χρήστες, που θα κάνουν ευτυχισμένους τους πελάτες τους.

Παραδείγματος χάριν, στην πλατφόρμα του Netflix, τα 2/3 των ταινιών που παρακολουθούνται, είναι λόγω προτάσεων που έγιναν στους χρήστες. Η Google παρατήρησε αύξηση του αριθμού των χρηστών που ακολουθούν συνδέσμους προτεινόμενων ειδήσεων, κατά 38%. Στο διαδικτυακό κατάστημα της Amazon ένα ποσοστό πωλήσεων, της τάξεως του 35%, πραγματοποιείται επειδή η λίστα προτεινόμενων προϊόντων προσέλκυσε το ενδιαφέρον των πελατών [4].

Τα Συστήματα Συστάσεων είναι “πολύτιμα μέσα για τους δικτυακούς χρήστες στην διαχείριση της υπερβολικής ποσότητας πληροφοριών και τους βοηθούν να κάνουν καλύτερες επιλογές” [8]. Για την αναγκαιότητα των ΣΣ, θα αναφερθώ και στο κεφάλαιο 2.1.2, συγκεκριμένα στο φαινόμενο της μακριάς ουράς (“Long tail phenomenon”). Στο 2.1.3 γίνεται αναφορά στο βραβείο Netflix (Netflix Prize) σαν ένα καλό παράδειγμα “συνεργασίας πολλών ομάδων ανθρώπων για την επίλυση ενός προβλήματος” [3], το οποίο “συγκέντρωσε τα βλέμματα στο πεδίο των ΣΣ και της αξίας της αναπαραγωγής προσωπικών συστάσεων από δεδομένα χρηστών” [8].

Στην παρούσα εργασία, υλοποιώ μια εφαρμογή σε Android πλατφορμα που προτείνει ταινίες σε χρήστες βάσει ενός Collaborative Filtering αλγορίθμου.

2. ΣΥΣΤΗΜΑΤΑ ΣΥΣΤΑΣΕΩΝ

2.1.1 Ιστορική αναδρομή

Στα μέσα του 1990 αναπτύχθηκε το πρώτο ΣΣ στο ερευνητικό κέντρο της Xerox στο Palo Alto. Από μια ομάδα τεσσάρων ερευνητών, έγινε προσπάθεια να φιλτράρουν την μεγάλη ποσότητα εισερχόμενων μηνυμάτων με το πρόγραμμα Tapestry [2].

Αν και σε πειραματικό επίπεδο, το Tapestry ενσωμάτωνε ένα υβριδικό μοντέλο φιλτραρίσματος πληροφοριών, φιλτράροντάς ως προς το περιεχόμενό τους ή και ως προς την απήχησή τους.

Οι χρήστες μπορούσαν να αναζητήσουν με τις κατάλληλες λέξεις-κλειδιά τα περιεχόμενα μηνυμάτων αλλων χρηστών, ενώ παράλληλα, υπήρχε η δυνατότητα από ένα χρήστη να σηματοδοτήσει ένα άρθρο ως “ενδιαφέρον”, φέρνοντας το στην επιφάνεια για κάποια που αναζητεί “ενδιαφέροντα” άρθρα [2].

Εικόνα 1: Η ατμόσφαιρα στο ερευνητικό κέντρο της Xerox συμβάδιζε με την χίπικη κουλτούρα στην Καλιφόρνια του 1970.

Εικόνα 2: Μια τυπική συνάντηση στο συνεδριακό χώρο του ερευνητικού κέντρου.

2.1.2 Long Tail Phenomenon

*"Τα περισσότερα λεφτά είναι στις μικρότερες πωλήσεις" [Kevin Law, φορέας κεφαλαίων επιχειρηματικού κινδύνου (v*enture capitalist) και πρώην σύμβουλος μουσικής βιομηχανίας*]*

Το "Φαινόμενο της Μακριάς Ουράς" (Long Tail Phenomenon) αναφέρεται στην κατανομή στοιχείων με χαρακτηριστική την συγκέντρωση ενος μικρού αριθμού αυτών στο "κεφάλι" της κατανομής και το μεγαλύτερο στην "ουρά". Παρατηρώντας το παρακάτω διάγραμμα, για τη συγκεκριμένη περίπτωση που έχει να κάνει με δημοτικότητα προιόντων και το πλήθος τους, είναι διακριτές δυο περιοχές - την κόκκινη με τα λίγα προιόντα υψηλής δημοτικότητας και την κίτρινη με τα περισσότερα σε πλήθος προιόντα αλλά όχι υψηλά σε αγοραστική ζήτηση. Είναι σημαντικό να επισημανθεί ότι οι δυο αυτές περιοχές έχουν ίδιο εμβαδό [32].

Εικόνα 3: Διάγραμμα Long Tail Phenomenon

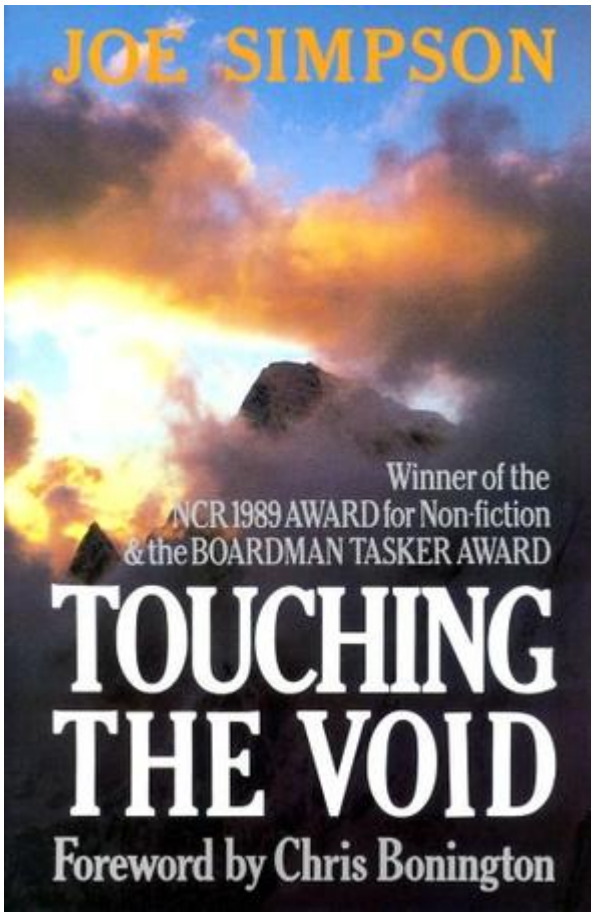
Στα καταστήματα λιανικής πώλησης, αναμένεται η επίτευξη κέρδους από την πώληση των πιο δημοφιλών προιοντων της "κεφαλής", σε αντίθεση με τη διαδικτυακή αγορά που το κέρδος επιτυγχάνεται διαφορετικά λόγω των Συστήματων Συστάσεων.

Σε ένα κατάστημα πώλησης δίσκων μουσικης, προυπόθεση για να μείνει στα ράφια του ένα άλμπουμ είναι να πουλάει τουλάχιστον 2 αντίτυπά το χρόνο, καθότι έτσι "πληρώνεται" το ενοίκιο για αυτά τα ελάχιστα εκατοστά χώρου που καταλαμβανει. Μια αίθουσα σινεμά πρέπει να προσελκύσει το λιγότερο 1500 άτομα σε διάρκεια προβολών 14 ημέρων, για την κάλυψη του κόστους ενοικίασης της αίθουσας. Κάτι παρόμοιο ισχύει και για τα βιβλιοπωλεία, τα σημεία πώλησης τύπου και τα καταστήματα ενοικίασης ταινιών [34].

Κάνοντας χρήση ενός ΣΣ και χάριν του Long Tail Phenomenon, οι εταιρείες ηλεκτρονικού εμπορίου, όπως η Amazon, συγκεντρώνουν πολλά έσοδα από την "ουρά" των προιόντων [33]. Ενδεικτικό παράδειγμα είναι αυτό της μεγάλης αμερικάνικης αλυσίδας βιβλιοπωλείων Barnes & Noble. Κατά προσέγγιση, το σύνολο των τίτλων που διαθέτει φτάνει 130.000. Τα

μισά από τα κέρδη βιβλίων για το Amazon γίνονται εκτός των κορυφαίων 130.000 τίτλων που παρέχονται [34].

Πρέπει ακόμη να αναφερθεί η περίπτωση των δύο βιβλίων “Touching the Void.” και “Into Thin Air”. Το βιβλίο της προσωπικής περιπέτειας του Joe Simpson - “Touching the Void”- αν και πήρε καλές κριτικές την χρονιά του 1988 που πρωτοεκδόθηκε, αποσύρθηκε από τα ράφια για μια δεκαετία περίπου, μεχρις ότου ο αλγόριθμος του Amazon να “αναλάβει δράση” [35]. Το ανερχόμενο στις λίστες με τα δημοφιλή βιβλία του 1997, “Into Thin Air” του Jon Krakauer συγκέντρωσε κριτικές αναγνωστών που ανέφεραν την ομοιότητά του με το δημιούργημα του J. Simpson. Το Σύστημα Συστάσεων του Amazon, αναγνωρίζοντας αυτό το συμβάν, συμπεριέλαβε στην προτεινόμενη λίστα του best-seller “Into Thin Air” το “Touching The Void”. Έτσι, το ξεχασμένο “Touching The Void”, κατέληξε να επανεκδοθεί και να γνωρίσει μεγάλης ζήτησης από το αναγνωστικό κοινό [35, 33].



Εικόνα 4: Touching the Void, εξώφυλλο της έκδοσης του 1988

2.1.3 Netflix Prize

Τον Οκτώμβριο του 2006, η διαδικτυακή εταιρεία ενοικίασης ταινιών Netflix, ανακοίνωσε ενα διαγωνισμό στον οποίο ο νικητής θα κέρδιζε το βραβείο του ενός εκατομμυρίου δολαρίων. Περίπου 4500 ομάδες από 182 χώρες συμμετείχαν στο διαγωνισμό που σκόπευε στην βελτίωση του υπάρχοντος αλγορίθμου για συστάσεις Cinematch εως 10% [1]. Επίσης, η κορυφαία στην κατάταξη ομάδα ετησίως, θα αμοιβόταν με 50.000 δολάρια [3]. Ο διαγωνισμός επιτέλεσε θετικά στην έρευνα πάνω στους Content Filtering αλγορίθμους [6]. Παρείχε ένα ξεκάθαρο πρόβλημα προς επίλυση το οποιό ενεργοποίησε πολλές ομάδες στο να συγκεντρωθούν για τη βελτίωση μονάχα μιας μέτρησης. Ενω μπορεί να θεωρηθεί ότι έγινε μια υπεραπλούστευση ενός προβλήματος συστάσεων, η όλη διαδικασία παρήγαγε σημαντικά αποτελέσματα [8].

Οι αρχικοί αλγόριθμοι που υλοποιήθηκαν ήταν είτε User- είτε Item-based, προσφέροντας μικρά ποσοστά βελτίωσης [1]. Μετά το πρώτο έτος διεξαγωγής του διαγωνισμού, η ομάδα KorBell κέρδισε το ετήσιο βραβείο με βελτίωση 8.43%. Ανέφεραν παραπάνω από 2000 ώρες δουλειάς ώστε με τον τελικό συνδυασμό 107 αλγορίθμων, να ανέβουν στην κορυφή της κατάταξης [8]. Ο διαγωνισμός έληξε το 2009 με τη συλλογική προσπάθεια τεσσάρων διαφορετικών διαγωνιζομένων [6]. Η βελτίωση επιτέυχθηκε λαμβάνοντας υπόψιν όχι μόνο παραμέτρους που αφορούσαν τις ταινίες και τους χρήστες αλλά το χρόνο, το συνδυασμό και τον υπολογισμό διάφορων προγνωστικών δεικτών, νέες τεχνικές όπως η παραγοντοποίηση πίνακα λαμβάνοντας υπόψιν τους γείτονες (neighborhood-aware matrix factorization) και την αυτόματη ρύθμιση παραμέτρων [37].

Επιπρόσθετα, ο Brandyn Webb (ψευδώνυμο Simon Funk), ένας ανεξάρτητος προγραμματιστής από τη Νέα Ζηλανδία, με μια καταχώρησή του στον προσωπικό του ιστοχώρο, εισήγαγε έναν επαυξητικό, επαναληπτικό και προσεγγιστικό τρόπο για την παραγοντοποίηση πίνακα (αναφερόμενη και ως Singular Value Decomposition - SVD) [38]. Αυτή η μέθοδος παρέχει έναν πρακτικό τρόπο για να εφαρμοστούν βαθμωτές μέθοδοι παραγοντοποίησης πινάκων σε μεγάλα σετ δεδομένων. Κάτι ακόμη άξιο να αναφερθεί για το συγκεκριμένο διαγωνιζόμενο, είναι πως ήταν από τους μοναδικούς από τους 10 κορυφαίους διαγωνιζόμενους που μοιράστηκε ανοιχτά τον κώδικα και τις ιδέες του, οι οποίες αποτέλεσαν βάση για τους υπόλοιπους του “κυνηγούς” του βραβείου[8, 36] !

Μια ακόμη “ανακάλυψη” του Netflix Prize ήταν ο μεγάλος όγκος “θορύβου” στις βαθμολογίες που παρείχαν οι χρήστες [8]. Κατά Herlocker [39], γίνεται η εισαγωγή του όρου “μαγικό εμπόδιο” (“magic barrier”), που αναφέρεται στην περιορισμένη ακρίβεια ενός Συστήματος Συστάσεων λόγω της φυσικής βαθμολογικής πολυμορφίας.

Ο διαγωνισμός του Netflix έδωσε στοιχεία που μπορούν να χρησιμοποιηθούν πέρα από την “εύρεση πιο ταιριαστής ταινίας”. “Με τους ερευνητές παγκοσμίως να μεταχειρίζονται ένα τεράστιο σετ δεδομένων και την πρόκληση της μεγάλης κλίμακας μοντελοποίησης, οι τεχνικές μπορούν να εφαρμοστούν στο πεδίο της επιστήμης, εμπορίου και πολιτικής” [40].

2.1.4 Κατηγοριοποίηση

Μια κατηγοριοποίηση, όπως προτείνεται κατά [11], δημιουργεί έξι διαφορετικές ομάδες Συστημάτων Προτάσεων:

Βάσει Περιεχομένου (Content-Based) (CB)

Το σύστημα προτείνει προϊόντα που είναι παρόμοια με άλλα στα οποία είχε δείξει προτίμηση ο χρήστης. Βάσει των ιδιοτήτων του εκάστοτε προϊόντος υπολογίζεται η εγγύτητα στη φύση των δύο προϊόντων [8]. Θεωρώντας ότι υπάρχει ένας θεατής που έχει βαθμολογήσει υψηλά μια κωμωδία, τότε το σύστημα θα “μάθει” να του προτείνει άλλες ταινίες από το ίδιο είδος. Βέβαια υπάρχουν προβλήματα που σε αυτού του είδους τα συστήματα, καθότι μια κατεξοχήν content-based προσέγγιση έχει θέματα περιορισμένης ανάλυσης περιεχομένου (limited content analysis) και υπερειδίκευσης (overspecialization) [41]. Limited content analysis συμβαίνει όταν ένα σύστημα έχει λιγοστή ποσότητα πληροφορίας είτε για τους χρήστες είτε για τα προιόντα του. Το παραπάνω μπορεί να εμφανιστεί σε εφαρμογές μουσικών προτάσεων, διότι “η συλλογή λεπτομερειών που αφορούν τα κομμάτια είναι μη προσοδοφόρα διαδικασία” [41]. Από την άλλη, το φαινόμενο της υπερειδίκευσης παρατηρείται σε περιπτώσεις που ένα σύστημα πρότασης ταινιών εμφανίσει ως επιθυμητή μια ταινία ίδιου είδους και με τους ίδιους ηθοποιούς που έχει ήδη δει ο χρήστης. Δε τίθεται δυνατή η πρόταση μιας ταινίας διαφορετικού περιεχομένου ή και με διαφορετικούς ηθοποιούς που μπορεί να ενδιαφέρει το χρήστη [41].

Παραδείγματα συστημάτων που υλοποιούν το μοντέλο αυτό είναι το Pandora Radio, το Rotten Tomatoes και το IMDb [42].

Συνεργατικού Φιλτραρίσματος (Collaborative Filtering) (CF)

Η συγκεκριμένη κατηγορία συστημάτων θεωρείται εκτενώς μελετημένη, η πιο ευρέως υλοποιημένη και η απλούστερη στην οικογένεια των ΣΣ [2, 6]. Η βασική ιδέα είναι ότι χρήστες που το ιστορικό των προτιμήσεών τους συμβαδίζει, τείνουν να έχουν ίδιο γούστο. Ας πούμε ότι ο χρήστης Β έχει βαθμολογήσει ως “πολύ καλή” μια ταινία Χ, ενώ το ίδιο έχει κάνει και ο Α για την ίδια ταινία. Το CF σύστημα την επόμενη φορά που ο Β θα βαθμολογήσει μια ταινία Υ ως “πολύ καλή”, στις προτεινόμενες ταινίες για τον Α θα εμφανιστεί η ταινία Υ. Περισσότερες λεπτομέρειες σχετικά με αυτού του είδους τα ΣΣ αναφέρονται στο κεφάλαιο 2.2. Διασημότερα δείγματα εταιρειών που κάνουν χρήση ΣΣ Συνεργατικού Φιλτραρίσματος, είναι το Amazon, το Facebook και το LinkedIn [42].

Βάσει Γνώσης (Knowledge-Based)

Οι δύο προαναφερθείσες μέθοδοι φιλτραρίσματος, δεν μπορούν να εφαρμοστούν σε αγορές που γίνονται σπανιότερα και μπορεί να αφορούν πακέτα διακοπών, αγορές ακίνητης περιουσίας, αυτοκίνητα [9]. Το γεγονός ότι δεν υπάρχει επαρκής αριθμός βαθμολογιών από προηγούμενους αγοραστές τα κάνει μη αποτελεσματικά για προτάσεις σε τέτοια προιόντα. Τη λύση έρχονται να δώσουν συστήματα που το προτεινόμενο προϊόν καταλήγει να είναι μια συνάρτηση των αναγκών του χρήστη και των δυνατοτήτων προσφερόμενων από το ίδιο το προϊόν [8]. Ο πυρήνας ενός γνωσιακού ΣΣ βρίσκεται στον “έλεγχο που έχει ο χρήστης για την καθοδήγηση της διαδικασίας που θα εξάγει την τελική πρόταση σε αυτόν” [9].

Πίνακας 1: Διαφορές τριών Συστημάτων Συστάσεων

Προσέγγιση	Στόχος	Είσοδος
Content-Based	Η σύσταση προκύπτει ως συνάρτηση των χαρακτηριστικών (γνωρισμάτων) που φάνηκε ότι υπάρχει προτίμηση στο παρελθόν και στη δραστηριότητα των χρηστών	Βαθμολογία χρήστη + χαρακτηριστικά προιόντος
Collaborative Filtering	Η σύσταση προκύπτει ως συνάρτηση των βαθμολογίων και της δραστηριότητας των χρηστών	Βαθμολογία χρήστη + βαθμολογία γειτόνων
Knowledge-Based	Η σύσταση προκύπτει ως συνάρτηση των αναγκών του χρήστη και των χαρακτηριστών που επιζητά	Ανάγκες χρήστη + χαρακτηριστικά προιόντος + πεδίο γνώσης

Παρατηρώντας τον παραπάνω πίνακα γίνεται πιο κατανοητή η διαφορά των τριών ΣΣ που μέχρι τώρα έχουν αναφερθεί και των στοιχείων που τα διαφοροποιούν.

Υποκατηγορία του παρόντος συστήματος είναι τα **Συστήματα βασιζόμενα σε Περιορισμούς (Constraint-Based)**, τα οποία προτρέπουν τη χρήστη να θέσει κανόνες και όρια (εύρος τιμής, χρόνια εγγύησης, χρώμα, κ.α.) στο προϊόν, τα οποία θα συνθέσουν το φίλτρο για την εξαγωγή της τελικής λίστας προτεινόμενων προϊόντων [6].

Παραδείγματα διαδικτυακών χώρων που χρησιμοποιούν φιλτράρισμα Βάσει Γνώσης είναι το car.gr, το skroutz.gr και ο ιστότοπος των αγγελιών της χρυσής ευκαιρίας (xe.gr).

Δημογραφικό (Demographic)

Ο χρήστης προτείνεται αντικείμενα που συμφωνούν με δημογραφικά του στοιχεία όπως η ηλικία, η χώρα προέλευσης, η μητρική του γλώσσα κλπ. Η πρόταση ενός χρήστη από την ίδια χώρα θεωρείται μεγαλύτερης βαρύτητας από χρήστη διαφορετικής προέλευσης [8]. “Ιδιαίτερη απήχηση γνωρίζουν τέτοια συστήματα στο χώρο της αγοραλογίας (marketing), σε αντίθεση με την επιστημονική κοινότητα που δεν έχει διερευνύσει εις βάθος το είδος αυτό” [8].

Βάσει κοινότητας (Community-Based)

Ο φιλικός περίγυρος του χρήστη, καθορίζει τις προτάσεις που του γίνονται από αυτό το σύστημα. Είναι αποδεδειγμένο ότι πιο εύκολα κάποια δέχεται προτάσεις από ανθρώπους που γνωρίζει παρά από ανώνυμους χρήστες (ακόμη και αν μοιράζονται ίδιες προτιμήσεις) [16]. Η παραπάνω διαπίστευση σε συνδυασμό με την “αυξημένη απήχηση των μέσω κοινωνικής δικτύωσης, ενισχύει το ενδιαφέρον για Community-Based συστήματα ή αλλίως Κοινωνικά Συστήματα Συστάσεων (Social Recommender Systems)” [15]. Το Facebook χρησιμοποιεί μεθόδους για προτεινόμενα δρώμενα που φίλοι χρηστών έχουν αναφέρει την αρέσκειά τους σε αυτά. Επίσης στην αρχική σελίδα του χρήστη εμφανίζονται οι σελίδες στις οποίες οι φίλοι του έχουν κάνει “like” . Με αυτούς τους τρόπους ο χρήστης μπορεί να ανακαλύψει ποιά από όλες τις φιλικά προσκείμενες σε αυτόν δραστηριότητες του ταιριάζουν και να δηλώσει την προτίμησή του ανάλογα.

Υβριδικό (Hybrid)

Για να ξεπεραστούν τα προβλήματα και οι περιορισμοί καποιου ΣΣ, μπορεί να γίνει συνδυασμός δύο ή και περισσότερων συστημάτων. Κατά Melville [23], προτάθηκε ένα “γενικό πλαίσιο (framework) για Content-Boosted Collaborative Filtering, όπου προβλέψεις Βάσει Περιεχομένου χρησιμοποιούνται για την μετατροπή ενός αραιού πίνακα βαθμολογιών σε πλήρη, ενώ μετέπειτα με CF μέθοδο, δημιουργούνται προτάσεις προς τη χρήστη”. Τα Υβριδικά ΣΣ έχουν στενή σχέση με το επιστημονικό πεδίο της Ανάλυσης Συνόλων (Ensemble Analysis) στο οποίο τα πλεονεκτήματα πολλαπλών τύπων αλγορίθμων Νοημοσύνης Μηχανών (Machine Learning algorithms) συνδυάζονται για καλύτερα αποτελέσματα [9]. Οι δύο κορυφαίες μέθοδοι (Bellkor’s Pragmatic Chaos [43], The Ensemble [44]) για το βραβείο Netflix είχαν βάσεις παρμένες από αυτό το επιστημονικό πεδίο [9]. Δεν είναι τυχαίο που και η ίδια πλατφόρμα του Netflix, κάνει χρήση υβριδικών μεθόδων φιλτραρίσματος. Οι προτάσεις προς το χρήστη γίνονται λαμβάνοντας υπ' όψιν τις προηγούμενες ταινίες που έχει δει αλλά και τις αναζητήσεις που έχει κάνει(Collaborative Filtering), ενώ προσφέρει νέες ταινίες παρόμοιων χαρακτηριστικών με ταινίες που έχει βαθμολογήσει υψηλά(Content-Based Filtering).

2.1.5 Αξιολόγηση Συστηματων Συστάσεων

Από τα προηγούμενα, έγινε ξεκάθαρο ότι υπάρχουν ποικίλοι τρόποι για να προσεγγίσει κάποιος την σχεδίαση ενός Συστήματος Συστάσεων. Ποιό από όλα τα Συστήματα Συστάσεων μπορούμε να πούμε ότι συστήνεται ανεπιφύλακτα; Η απάντηση δεν είναι ξεκάθαρη και γι' αυτό υπάρχουν κλίμακες που κάνουν αποτίμηση ενός RS και σε συνδυασμό με τις ανάγκες του σχεδιασμού, οδηγούν στην τελική επιλογή. Πολλοί μπορεί να παραθέσουν τους διάφορους περιορισμούς των μεθόδων αξιολόγησης, ενώ άλλοι να συμπεράνουν ότι “η ποιότητα ενός ΣΣ δε μπορεί να μετρηθεί με ακρίβεια λόγω της ύπαρξης πολλών διαφορετικών αντικειμενικών παραγόντων “[6].

Παρ' όλα αυτά, η μεγάλη συμβολή των ΣΣ σε ευρεία γκάμα τομέων, “επιτάσσει την καλλιέργεια μεθόδων για ρεαλιστική και ακριβή αξιολόγηση της απόδοσης, αποτελεσματικότητας και επιρροής οποιουδήποτε RS” [6].

Σχεδιασμός μοντέλων αξιολόγησης – Evaluation Design

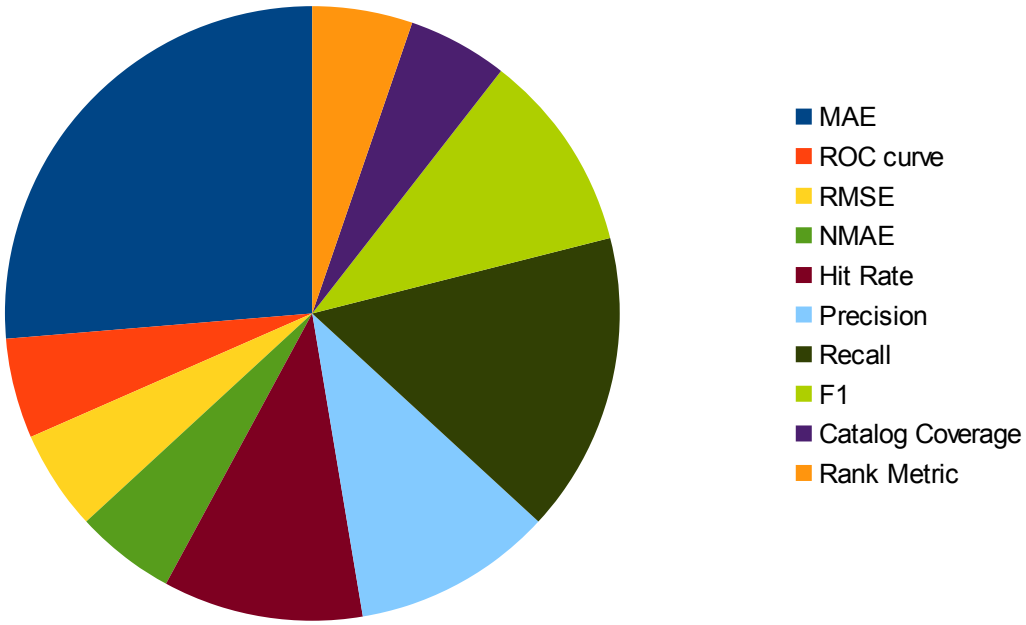
Στην βιβλιογραφική αναφορά κατά Herlocker [7], γίνεται μια εκτενής αναφορά σε μοντέλα που μετρούν την ακρίβεια μαζί με εναλλακτικά κριτήρια αξιολόγησης ΣΣ.

Αυτά κυρίως αφορούν μετρικές που έχουν να κάνουν με :

- την Ακρίβεια Προβλεπόμενων Βαθμολογιών – Ratings Prediction Accuracy
- την Ακρίβεια Προβλεπόμενης Χρήσης – Usage Prediction Accuracy
- την Ακρίβεια Κατάταξης των Προτεινόμενων Αντικειμένων – Classification Accuracy

Ακρίβεια Προβλεπόμενων Βαθμολογιών – Ratings Prediction Accuracy

Κάνοντας χρήση των στοιχείων που παρέχονται στο Introduction to Recommender Systems [6], γίνεται μια σκιαγράφηση των μετρικών που χρησιμοποιήθηκαν από επιστημονικές μελέτες πάνω στο θέμα των ΣΣ. Εκδόθηκαν κατά την περίοδο πέντε ετών (2004-2008) στην διαδικτυακή πλατφόρμα του ACM Transactions on Information Systems (ACM TOIS), κοινότητα ερευνητικών δημοσιεύσεων που αφορούν την ανάκτηση πληροφοριών (information retrieval). Είναι άξιοσημείωτο ότι πολλές ήταν οι μελέτες που έκαναν χρήση περισσότερων από μια μεθόδων για την αξιολόγηση των Συστημάτων Συστάσεών τους.



Σχήμα 1:Κατανομή μεθόδων αξιολόγησης RS

Από το παραπάνω διάγραμμα, παρατηρείται ότι η μέθοδος του Μέσου Απόλυτου Σφάλματος(MAE – Mean Absolute Error), χρησιμοποιήθηκε επί το πλείστον, με τον τύπο της να δίνεται παρακάτω:

$$MAE = \frac{\left(\sum_{u \in U} \sum_{i \in testset_u} \left| rec(u,i) - r_{(u,i)} \right|\right)}{\left(\sum_{u \in U} \left| (testset_u) \right|\right)}$$

Έτσι υπολογίζεται η μέση απόκλιση μεταξύ των υπολογισμένων προτεινόμενων βαθμολογιών $rec(u,i)$ και των πραγματικών τιμών βαθμολογίας $r_{(u,i)}$ για όλους τους χρήστες $u \in U$ και των αντικειμένων που ανήκουν στο συνολο $testset_u$.

Κατά Sarwar [54], παρουσιάστηκε το RMSE (Root Mean Square Error) για να καλυφθούν οι μεγάλες αποκλίσεις ενώ ακόλουθα κατά Goldberg [55], έγινε εισαγωγή του όρου του κανονικοποιημένου MAE (Normalized MAE – NMAE) για την κάλυψη του βαθμολογικού εύρους [6].

Ακρίβεια Προβλεπόμενης Χρήσης – Usage Prediction

Υπάρχουν εφαρμογές που αντί να προβλέπουν την βαθμολογία του χρήστη για αντικείμενα, π.χ. βαθμολογία ταινιών, αλλά προτείνουν αντικείμενα που ο χρήστης είναι

διατεθημένος να χρησιμοποιήσει. Ένα καλό παράδειγμα είναι η πλαφόρμα του Netflix που δοθείσας μιας ταινίας στη ουρά των ταινιών προς παρακολούθηση, προτείνει ταινίες που ο χρήστης μπορεί να βρει ενδιαφέρουσες [8].

Η μετρική της Ακρίβειας - Precision (P), συσχετίζει το αριθμό των ορθά προτεινόμενων αντικειμένων για τον χρήστη u - $hits_u$ με το συνολικό αριθμό των προτεινόμενων αντικειμένων $recset_u$.

$$P_u = \frac{|(hits_u)|}{|(recset_u)|}$$

Αντιθέτως, η Recall (R) , δίνει την αναλογία των $hits_u$ με το θεωρητικό μέγιστο αριθμό επιτυχιών – hits βάσει του πειραματικού συνόλου $testset_u$.

$$R_u = \frac{|(hits_u)|}{|(testset_u)|}$$

Η F1 κατά [6] ή F-measure κατά [8] κάνει χρήση των δύο προαναφερόμενων μεταβλητών και ο τύπος της :

$$F = \frac{2 \cdot P \cdot R}{(P + R)}$$

ισοσταθμίζει τις R και P, δίνοντας περισσότερη βαρύτητα στη μικρότερη τιμή [6].

Άλλες μονάδες ειναι το ποσοστό των Ψευδοθετικών (False–Positives) (False Positive Rate), Receiver Operating Characteristic Curve (ROC Curve) και Area Under the ROC Curve.

Ακρίβεια Κατάταξης των Προτεινόμενων Αντικειμένων – Classification Accuracy

Ο στόχος της κατάταξης στο πεδίο των ΣΣ είναι για η επιλογή των N πιο σχετικών αντικειμένων δοθέντος του χρήστη [6]. Αυτή μπορεί να γίνει είτε με Βαθμολογημένη Αναφορά - Reference Ranking είτε με Βαθμολογημένη Χρησιμότητα - Utility Based Ranking [8].

- Reference Ranking

Η Normalized Distance based Performance Measure (NDPM) [65] χρησιμοποιείται συχνά για τη Συγκέντρωση Πληροφοριών (Information Retrieval), ενώ ακόλουθα οι μεταβλητές Spearman's Rank Correlation Coefficient, που αποτελεί βάση του αλγορίθμου για την εφαρμογή που υλοποιήθηκε στην εργασία αυτή (βλ. Κεφάλαιο 2.2.3) και η Kendall Rank Correlation Coefficient. Οι δύο τελευταίες μάλιστα, “συσχετίζονται αρκετά στην πράξη” [57].

- Utility Based ranking

Για μια μέθοδο που βασίζεται σε αξιολόγηση βάσει της χρησιμότητας, είναι σημαντικό να κατανοηθεί από τον αλγόριθμο το “πόσο χρήσιμο βρίσκει ο χρήστης το προτεινόμενο αντικείμενο” [9].

Η χρησιμότητα - utility μιας λίστας προτάσεων θεωρείται το άθροισμα των τιμών της χρησιμότητας των ξεχωριστών αντικειμένων [8]. Στην εκάστοτε τιμή της χρησιμότητας αφαιρείται ένας παράγοντας που αφορά τη θέση του προτεινόμενου αντικειμένου στη λίστα. Όπως αναφέρεται κατά [8], καθώς ο χρήστης εξετάζει τη λίστα προτάσεών του σειριακά - από την αρχή εώς το τέλος- η τιμή της χρησιμότητας των αντικειμένων να εμφανίζει φθίνουσα πορεία. Σε εφαρμογές που η χρήστης αναμενεται ότι θα εξετάσει μεγάλη λίστα αντικειμένων, γίνεται χρήση του Normalized Discounted Cumulative Gain (NDCG) [58], με την τιμή της χρησιμότητας να μειώνεται μεν αλλά με “λογαριθμικό ρυθμό” δε [8]. Υποθέτοντας ότι κάθε χρήστης $u \in N$ έχει “κέρδος” (“gain”) $g_{(u,i)}$ με το προτεινόμενο αντικείμενο i, το μέσο Discounted Cumulative Gain (DCG) για μία λίστα J αντικειμένων, να ορίζεται ώς:

$$DCG = \frac{1}{N} \sum_{u=1}^{N} \sum_{j=1}^{J} \left(\frac{g_{(u,i_j)}}{(\log(j+1))} \right)$$

όπου i_j ειναι το αντικείμενο i στη θέση j της λίστας και N το σύνολο των χρηστών. Η βάση του λογαριθμου μπορεί να πάρει τιμές στο [2,10]. “Προτιμάται η τιμή του 2 καθότι έτσι εγγυάται ότι όλες οι θέσεις θα υποστούν μείωση” [8].

NDCG είναι η κανονικοποιημένη μορφή της DCG που δίνεται από τον τύπο:

$$NDCG = \frac{DCG}{DCG^b}$$

όπου DCG^b η βελτιστη τιμή της DCG.

2.1.6 Προβληματισμοί - προκλήσεις

Cold Start Problem

Τα νεοεισαχθέντα αντικείμενα και νέοι χρήστες αποτελούν τη μεγαλύτερη πρόκληση για τα ΣΣ. Οι δύο παράγοντες αυτοί, μαζί αποτελούν το cold start problem [48]. Τα νέα προιόντα κυρίως επηρρεάζουν Collaborative Filtering Συστήματα, όπου δε μπορεί να γίνει πρόταση στο χρήστη δίχως να έχουν δεχθεί ως είσοδο βαθμολογίες. Επιπρόσθετα, το παραπάνω θέμα προκύπτει και σε όχι δημοφιλή αντικείμενα που όμως μπορεί να αποτελούν καθοριστικά για χρήστες με εκλεπτυσμένες προτιμήσεις [10].

Από την άλλη μεριά, Content-Based προσεγγίσεις [49], επειδή δε βασίζονται στις βαθμολογίες χρηστών, χρησιμοποιούνται για να δώσουν συστάσεις σε νέα μη βαθμολογημένα αντικείμενα [50]. Μάλιστα, Content-Based προβλέψεις γειτονικών χρηστών μπορούν να βελτιώσουν τις προβλέψεις για τον ενεργό χρήστη [23]. Από την άλλη μεριά, το θέμα των καινούργιων χρηστών είναι δύσκολο να ξεπεραστεί καθώς χωρίς προηγούμενες προτιμήσεις ενός χρήστη είναι δύσκολο να βρεθούν άλλοι με παρόμοια γούστα [10].

Κατά το σχεδιασμό του ΣΣ μπορεί να χρησιμοποιηθεί ένα κατώφλι για την απόφαση του πλήθους των αντικειμένων που πρέπει κατ'ελάχιστο να βαθμολογηθούν από το χρήστη [8]. Επίσης τα λεγόμενα ως cold items είναι αντικείμενα χωρίς βαθμολογίες [48] ή αντικείμενα που υπάρχουν στο σύστημα για κάποιο συγκεκριμένο χρονικό διάστημα (π.χ. μια μέρα) ή έχουν λιγότερο από τον προκαθορισμένο αριθμό βαθμολογιών (π.χ. κάτω από 10 βαθμολογήσεις χρηστών). Μπορεί ένας ακόμη γενικότερος τρόπος για να θεωρηθεί ένα αντικείμενο ως cold είναι το ποσοστό των πληροφοριών γύρω από αυτό. Επομένως το ΣΣ μπορεί να βελτιωθεί προτείνοντας cold items και όχι τόσα σε ζήτηση (hot items). Βέβαια το σύστημα για χάριν των προτάσεων cold αντικειμένων να χάνει σε θέμα ακρίβειας αντικειμένων με απήχηση.

Fraud - Νωθεία

Η ολοένα και αυξανόμενη χρήση των Συστημάτων Συστάσεων από τους εμπορικούς ιστόχωρους, έχει συμβάλλει στη συγκομιδή κερδών των πωλητών. Απόρροια αυτού είναι ορισμένοι "ασυνείδητοι πωλητές να νωθεύουν τα αποτελέσματα των ΣΣ για δικό τους όφελος" [10]. Ένας συνηθισμένος τρόπος είναι να αυξάνουν την δημοτικότητα προιόντων που προωθούν αυτοί (push attacks) ή να ρίχνουν τις βαθμολογίες σε αντικείμενα που πωλούνται από ανταγωνιστές (nuke attacks)[10]. Αυτού του είδους οι μέθοδοι, έχουν μελετηθεί ως επιθέσεις shiling [46] ή ως επιθέσεις έγχυσης προφίλ (profile injection attacks) [47]. Στην τελευταία περίπτωση, ο κακόβουλος χρήστης δημιουργεί ψευδές προφίλ και συγκεντρώνει πληροφορίες για το σύστημα.

Για παράδειγμα, μια κοινή επίθεση [46] στοχεύει σε ακραίες τιμές βαθμολογίας, που θα οδηγήσει στην αποσταθεροποίηση των προβλεπόμενων βαθμολογιών. Είναι άξιο να σημειωθεί ότι ένα Item-Based CF Συστήμα, τείνει να παραμείνει ανεπηρρέαστο από τέτοιες επιθέσεις[46]. Επίσης Content-Based μέθοδοι που μόνο βασίζονται σε παρελθοντικές βαθμολογίες μένουν αναλλοίωτες από profile injection επιθέσεις.

Sparsity – Αραιότητα

Πληθώρα χρηστών δεν βαθμολογούν τα περισσότερα αντικείμενα και αποτέλεσμα αυτού είναι ο πίνακας χρηστών-βαθμολογιών R να είναι αραιός. Συμβαίνει συχνά στα Collaborative Filtering Συστήματα Συστάσεων και ελαχιστοποιεί τις πιθανότητες να βρεθούν γειτονικοί χρήστες για την εκάστοτε ενεργή χρήστη [10]. Συναντάται σε Συστήματα με υψηλή αναλογία προιόντων - χρηστών (για το ebay: 1 δισεκατομμύριο προιόντα / 250 εκατομμυρία ενεργοί χρήστες, αναλογία 4 : 1 [53]) ή συστήματα που μόλις έχουν ξεκινήσει να λειτουργούν. Κατά [31], ένας τρόπος αντιμετώπισης του παραπάνω είναι η χρήση Item-Based Αλγόριθμου, του οποίου οι παράμετροι βελτιστοποιούνται με Αναπτυγμένη Λογιστική Παλινδρόμηση (Extended Logistic Regression- ELR) [79].

Privacy - Ιδιωτικότητα

Ίσως μια από τις μεγαλύτερες πρόσφατες συζητήσεις που προέκυψαν πάνω στο θέμα της ιδιωτικότητας δεδομένων χρηστών, έγινε το Μάρτιο του 2018. Μαζικά μέσα ενημέρωσης έκαναν λόγο για εκμετάλλευση προσωπικών δεδομένων εκατομμυρίων χρηστών του Facebook από την συμβουλευτική εταιρεία πολιτικής Cambridge Analytica Ltd (CA) [59] [60]. Παρ' όλη την έκθεση του θέματος και τις μετέπειτα ενέργειες για την προστασία των χρηστών [61], κατά [8] , “οι άνθρωποι είναι διατεθημένοι να προσφέρουν προσωπικές πληροφορίες ως ανταλλαγή για προσωπικό όφελος”.

Τα Συστήματα Συστάσεων κάνουν χρήση μιας μεγάλης γκάμας πληροφοριών, δίνοντας πίσω στους χρήστες “καλύτερες υπηρεσίες και προιόντα” [62,63,64,65,66] . Τέτοιες πληροφορίες μπορεί να αφορούν το ποιά προιόντα επιλέγουν να προβάλουν, ποιούς συνδέσμους κλικάρουν, ποιός είναι ο κοινωνικός τους περίγυρος, τι δουλειά κάνουν, τι ηλικία έχουν [67]. Για ελεγχόμενη και υπεύθυνη συλλογή πληροφοριών, ο OECD [68] έχει συντάξει μια σειρά από Θεμιτές Πρακτικές σχετικά με τις Πληροφορίες (Fair Information Practices). Είναι δύσκολη αποστολή η τήρηση των κανόνων αυτών με την παράλληλη ανάγκη για συλλογή τεράστιου όγκου δεδομένων και για συγκέντρωση προσωπικών πληροφοριών χρηστών που θα συνδράμουν στο βέλτιστο προτεινομενο αντικείμενο [8]. Έχει γίνει συζήτηση πάνω στο θέμα της ιδιωτικότητας από την επιστήμη Ανάκτησης Πληροφοριών (Data Mining)[69] και έχουν γίνει υλοποιήσεις αλγορίθμων ΣΣ που πραγματεύονται τη διαφύλαξη του απορρήτου του χρήστη [70, 71, 72]. Έχει προταθεί η κωδικοποίηση των δεδομένων αλλά “μειονέκτημα αυτής είναι η μείωση της απόδοσης των Data Mining αλγορίθμων” [9].

Ανακάλυψη Νέων Αντικειμένων - Novelty

Από τη λίστα προτάσεων στο χρήστη ενός RS, υπάρχουν τα αντικείμενα που ο χρήστης ήδη γνώριζε την ύπαρξή τους και αυτά που είναι καινουργια και άγνωστα. Έτσι, υπάρχει αυτή η “διάκριση μεταξύ του ενθουσιασμού του αγνώστου και της απαίτησης για υψηλής ποιότητας πρότασεων” [25]. Από τη μια πλευρά η ποιότητα των προτεινόμενων αντικειμένων [54] συσχετίζεται με την αρέσκειά τους από το χρήστη. Από την άλλη πλευρά, νέες προτάσεις έχουν σχέση με την “παλαιότητα” αυτών [25]. 'Οσο περισσότερο χρόνο το αντικείμενο υπάρχει μέσα στο ΣΣ τόσο μεγαλύτερη η “παλαιότητα“ του. “Παλιά” αντικείμενα θα πρέπει να αντικαθίστανται από νεότερα, άγνωστα μεχρι πρότεινος αντικείμενα. Βέβαια,

είναι απόρροια αυτού ο χρήστης να νιώσει χαμένος μέσα στις νέες προτάσεις που έχουν εμφανιστεί, άρα να εκλάβει ότι η ποιότητα συστάσεων έχει μειωθεί [25].

Δυνατότητα επέκτασης/ Ευελιξία - Scalability

Μια “κατάρα” του τεράστιου όγκου δεδομένων, κατά τα λεγόμενα του Amazon [21], είναι η καταπόνηση που δέχονται οι αλγόριθμοι των ΣΣ. Όντας σχεδιασμένοι να δουλεύουν με “υποτριπλάσιου μεγέθους δεδομένα, η ανάγκη για τον υπολογσιμό συστάσεων σε πάνω από 29 εκατομμύρια πελάτες, δυσκολεύει τα συστήματα αυτά”. Ένας αλγόριθμος, ενώ μπορεί να έχει “υψηλή αποδοτικότητα σε μικρής κλίμακας δεδομένα, μια αύξησή τους ίσως εισάγει χρονική καθυστέρηση – ρίχνοντας την απόδοσή του” [25]. Επομένως είναι σημαντικό προσόν για ένα RS να μπορεί να δουλεψει με μεγάλου ποσού πληροφορίες και να μεταχειριστεί μελλοντική αύξησή τους [74].

2.2 ΣΥΣΤΗΜΑΤΑ ΣΥΝΕΡΓΑΤΙΚΟΥ ΦΙΛΤΡΑΡΙΣΜΑΤΟΣ

Τα CF συστήματα πρέπει να μαζέψουν πληροφορίες για τις προτιμήσεις έτσι ώστε να γεμίσει η λίστα με τα “προϊόντα που ίσως σας αρέσουν”, είτε έμμεσα είτε άμεσα. Στην εφαρμογή του IMDb, ο χρήστης αφότου βαθμολογήσει την ταινία που έχει δει, το σύστημα καταγράφει αυτήν του την βαθμολογία για μετέπειτα χρήση της πληροφορίας [17]. Αυτός είναι ο άμεσος τρόπος. Με έμμεσο τρόπο, θα μπορούσε ο εξυπηρετητής (server) του Amazon, για παράδειγμα, να “ανταλλάσει πληροφορία με το τερματικό του χρήστη και να κρατάει δεδομένα που αφορούν τη θέση του ή το είδος φυλλομετρητή (browser) που χρησιμοποιεί” [5].

2.2.1 Περιγραφή

Αφότου ένας πίνακας χρηστών-βαθμολογιών αντικειμένων ληφθεί ως είσοδος, γίνεται η επεξεργασία του από έναν CF αλόριθμο και παράγονται ως έξοδος τα εξής [6,27] :

- Μια πρόγνωση (prediction). Είναι μια αριθμητική τιμή που δείχνει το βαθμό που ένας συγκεκριμένος χρήστης προτιμά ή όχι ένα συγκεκριμένο αντικείμενο. Η τιμή της ανήκει στο ίδιο διάστημα που είναι και οι βαθμολογίες των αντικειμένων.
- Μια λίστα N προτεινόμενων αντικειμένων. Η λίστα αυτή περιλαμβάνει αντικείμενα που δεν έχουν αγοραστεί από το χρήστη και είναι υποσύνολο των αντικειμένων που προσφέρονται [27].

Το $U = u_{1,} u_{2,} \dots, u_n$ είναι το σύνολο των χρηστών , με $P = p_{1,} p_{2,} \dots, p_m$ να αναπαριστά το σύνολο προϊόντων και R είναι ο πίνακας βαθμολογιών $n \times m$ με κάθε βαθμολογία $r_{(i,j)}$, όπου $i \in n$ και $j \in m$ [6]. Ο χρήστης για τον οποίο το RS θα βγάλει τη λίστα προτάσεων, ονομάζεται ενεργός χρήστης (active user) [27] .

Κάθε βαθμολογία παίρνει τιμές από ένα εύρος διακεκριμένων τιμών. Αν κάποιος $\chi\rho\acute{\eta}\sigma\tau\eta\varsigma_i$ δεν έχει βαθμολογήσει ένα $\pi\rho οιόν_j$ τότε το πεδίο βαθμολογίας $r_{(i,j)}$ παραμένει κενό ή 0 [6, 27] .

Ο παρακάτω πίνακας δείχνει μια βάση δεδομένων βαθμολογίας του ενεργού χρήστη A και χρηστών u_i .

Πίνακας 2: Βαθμολογίες R(i,j)

	$\alpha\nu\tau\iota\kappa\epsilon\acute{\iota}\mu\epsilon\nu o_1$	$\alpha\nu\tau\iota\kappa\epsilon\acute{\iota}\mu\epsilon\nu o_2$	…	$\alpha\nu\tau\iota\kappa\epsilon\acute{\iota}\mu\epsilon\nu o_m$
A	$r_{(a,1)}$	$r_{(a,2)}$		$r_{(a,m)}$
u_1	$r_{(1,1)}$	$r_{(1,2)}$		$r_{(1,m)}$
u_2	$r_{(2,1)}$	$r_{(2,2)}$		$r_{(2,m)}$
…				
u_n	$r_{(n,1)}$	$r_{(n,2)}$		$r_{(n,m)}$

Δύο κύριες κατηγορίες CF Αλγορίθμων έχουν εντοπιστεί, όπως αναφέρεται κατά [26] :

- Βάσει Μνήμης ή βασιζόμενο στο Χρήστη (Memory Based/ User-based)
- Βάσει μοντέλου ή βασιζόμενο στο Αντικείμενο (Model Based/ Item-based)

Ένας αλγόριθμος που βασίζεται στο Χρήστη (User-Based), χρησιμοποιεί στην πληρότητά του τον πίνακα βαθμολογιών χρηστών για να υπολογίσει την πρόγνωση [27]. Μέσω στατιστικών μεθόδων, κατασκευάζεται το σύνολο χρηστών, αναφερόμενων και ως γειτόνων, που έχουν ιστορικό βαθμολογίας παρόμοιο με τον ενεργό χρήστη.

Το επόμενο βήμα είναι παραχθεί η λίστα των N αντικειμένων με τιμές που αφορούν την πρόγνωση για το αν θα αρέσουν στον προτεινόμενο χρήστη. Υπάρχει εκτενής ποικιλία μεθόδων για υπολογισμό της ομοιότητας των χρηστών όπως η ομοιότητα του συνημιτόνου (cosine similarity), οι μέσες διαφορές τετραγώνων (mean squared differences), η ομοιότητα μετασχηματισμένου συνημιτόνου (adjusted cosine similarity) κ.α. [7, 31].

Item-Based αλγόριθμοι, χρησιμοποιούν τα αποθηκευμένα δεδομένα για να “μάθουν ένα μοντέλο πρόγνωσης. Τα σημαντικότερα χαρακτηριστικά των χρηστών και των αντικειμένων καταγράφονται σε ένα σετ μοντέλου παραμέτρων, παράμετροι οι οποίες αντλούνται από δεδομένα εκμάθησης και χρησιμοποιούνται αργότερα για να προγνώσουν νέες βαθμολογίες” [8]. Τέτοια παραδείγματα αλγορίθμων είναι η Ομαδοποίηση Bayesian (Bayesian Clustering)[28], η Λανθάνουσα Σημειολογική Ανάλυση (Latent Semantic Analysis) [29], η Λανθάνουσα Κατανομή Dirichlet (Latent Dirichlet Allocation) [30] κ.α.

2.2.2 Επιλογή CF Αλγορίθμου

Η επιλογή του αλγορίθμου για την εφαρμογή βασίστηκε περισσότερο σε προσωπική επιλογή λόγω της απλότητας στην κατανόησή της. Είναι χαρακτηριστικό αυτών των συστημάτων, τόσο “η απλότητα” όσο και “η διαισθητικη προσέγγιση” τους [9].

Ακόμη το γεγονός ότι, συμφωνα με την πηγή [6], Content-Based μέθοδοι “μελετώνται εκτενώς τα τελευταία 15 χρόνια και εφαρμόζονται κατά κόρον”, έκανε πιο βέβαιο ότι οι πληροφορίες για αυτα τα συστήματα θα είναι επαρκής και οι αλγόριθμοί τους επιβεβαιωμένοι, ελεγμένοι και αξιολογημένοι από την επιστημονική κοινότητα. Περαιτέρω περιγραφή που αφορά την αποτελεσματικότητα, συγκριτικά με άλλες τεχνικές, γίνεται στο κεφάλαιο 2.2.4 .

2.2.3 Ένα παράδειγμα CF αλγορίθμου

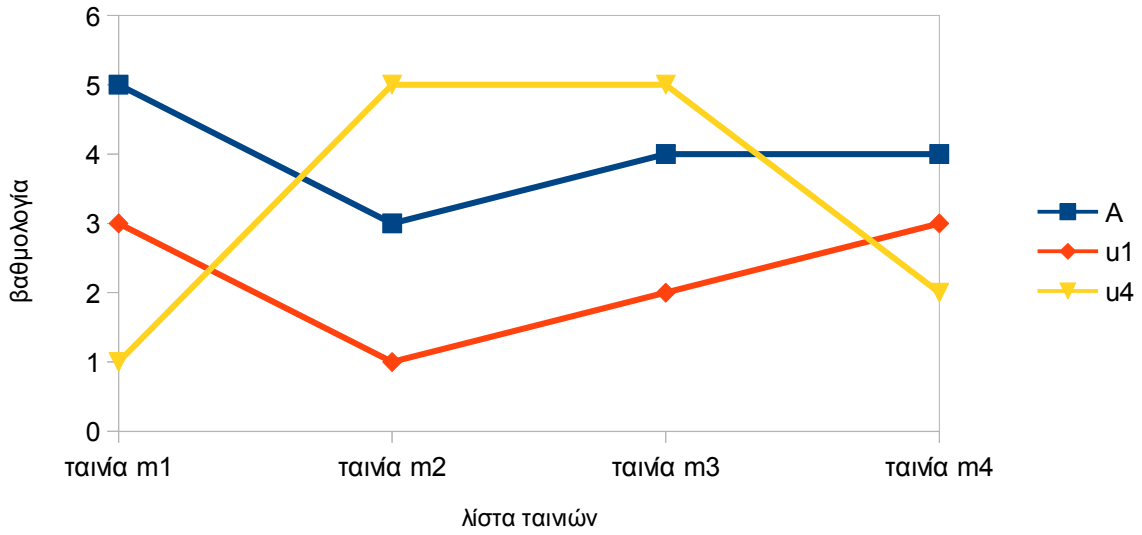
Για την εφαρμογή της πτυχιακής, γίνεται χρήση του παρακάτω πίνακα βαθμολογιών R, με την κάθε σειρά να έχει τις βαθμολογίες ενός ^{χρήστη_i} , ενώ η κάθε στήλη έχει τις βαθμολογίες μιας ^{ταινίας_j} .

Πίνακας 3: Βαθμολογίες Ταινιών Εφαρμογής

R(i,j)	Ταινία m_1 Godfather	Ταινία m_2 Pulp Fiction	Ταινία m_3 Your name.	Ταινία m_4 The Shawshank Redemption	Ταινία m_5 Life Is Beautiful
Ενεργός χρήστης A bloper@gmail.com	5	3	4	4	?
Χρήστης u_1 bibou@hotmail.com	3	1	2	3	3
Χρήστης u_2 zaze@gmail.com	4	3	4	3	5
Χρήστης u_3 tinton@hotmail.gr	3	3	1	5	4
Χρήστης u_4 ezziz@yahoo.net	1	5	5	2	1

Χρησιμοποιώντας το συντελεστή συσχέτισης Pearson (Pearson's correlation coefficient), υπολογίζεται την ομοιότητα (similarity) δύο χρηστών a και b, δοθέντος του πίνακα R και συνόλου προιόντων $^{p \in P}$. Ο συντελεστής παίρνει τιμές στο [-1, +1].

$$similarity(a,b) = \frac{\left(\sum_{p \in P} \left(r_{(a,p)} - \bar{r}_a\right)\left(r_{(b,p)} - \bar{r}_b\right)\right)}{\left(\sqrt{\sum_{p \in P} \left(r_{(a,p)} - \bar{r}_a\right)^2} \sqrt{\sum_{p \in P} \left(r_{(b,p)} - \bar{r}_b\right)^2}\right)}$$



Σχήμα 2: Βαθμολογίες ενεργού χρήστη Α και χρηστών 1, 4

Η ομοιότητα του Α με τον u_1 υπολογίζεται ως εξής: (θεωρώντας $(\bar{r}_a)$ = 4 , $(\bar{r}_1)$ = 2.25)

$$similarity(A, u_1) = \frac{((r_{(A,1)} - (\bar{r}_A)).(r_{(1,1)} - (\bar{r}_1)) + (r_{(A,2)} - (\bar{r}_A)).(r_{(1,2)} - (\bar{r}_1)) + (r_{(A,3)} - (\bar{r}_A)).(r_{(1,3)} - (\bar{r}_1)) + (r_{(A,4)} - (\bar{r}_A)).(r_{(1,4)} - (\bar{r}_1)))}{(\sqrt{((r_{(A,1)} - (\bar{r}_A))^2 + (r_{(A,2)} - (\bar{r}_A))^2 + (r_{(A,3)} - (\bar{r}_A))^2 + (r_{(A,4)} - (\bar{r}_A))^2}).\sqrt{((r_{(1,1)} - (\bar{r}_1))^2 + (r_{(1,2)} - (\bar{r}_1))^2 + (r_{(1,3)} - (\bar{r}_1))^2 + (r_{(1,4)} - (\bar{r}_1))^2}))}$$

Για τους $u_{2,} u_{3,} u_4$, οι μέσες τιμές είναι $(\bar{r}_2)$ = 3.5, $(\bar{r}_3)$ = 3, $(\bar{r}_4)$ = 3.25 ενώ οι συντελεστές Pearson είναι $similarity(A, u_2)$ = 0.7, $similarity(A, u_3)$ = 0.0 και $similarity(A, u_4)$ = -0.79 αντίστοιχα.

Αυτά τα νούμερα δείχνουν ότι ο u_1 και u_2 έχουν κοντινή συσχέτιση με τον Α στην παρελθοντική τους συμπεριφορά. Ο συγκεκριμένος συντελεστής λαμβάνει υπ' όψιν το πόσο διαφορετικά ο κάθε χρήστης εκλαμβάνει τη βαθμολογική σκάλα. Δηλαδή, κάποιοι τείνουν για μόνο ψηλές βαθμολογίες ενώ άλλοι δεν βαθμολογούν με 5 κανέναν αντικείμενο. Έτσι κοιτώντας το γράφημα, παρατηρείται πως ενώ η γραμμή που ακουλουθούν οι βαθμολογίες των Α και u_1 , είναι διαφορετικές, υπάρχει μια γραμμική ομοιότητα σε αυτές [6].

Για το επόμενο βήμα, την πρόβλεψη προτίμησης του ενεργού χρήστη Α για την ταινία m_5 , πρέπει να αποφασίστεί ποιού χρήστη θα ληφθεί υπόψιν τη βαθμολογία και με πόση βαρύτητα θα την εκτιμήσουμε. Για το παρόν παράδειγμα και τον αλγόριθμο που υλοποιήθηκε, χρήστης b με κοντινές προτιμήσεις θεωρείται αυτός που η συνάρτηση

$similarity(A, b)$ δίνει αποτέλεσμα μεγαλύτερο του 0.5. Έτσι επιλέχθηκαν οι u_1 και u_2 , ως χρήστες με κοντινές προτιμήσεις .

Ο τύπος εκτίμησης του Α για κάποια ταινία m που προτείνεται κατά [6], είναι ο εξής :

$$prediction(A, m) = (\bar{r}_A) + \frac{(\sum_{b \in N} similarity(a, b).(r_{(b,m)} - (\bar{r'}_b)))}{(\sum_{b \in N} similarity(a, b))}$$

όπου Ν το σύνολο των χρήστών με τις κοντινότερες προτιμήσεις και $(\bar{r}_A)$ η μέση βαθμολογία του Α και $(\bar{r'}_1)$ = 2.4 , $(\bar{r'}_2)$ = 3.8, οι μέσοι όροι των χρηστών 1 και 2 που συνυπολογίζουν όχι μόνο τις κοινές ταινίες με τον χρήστη Α αλλά και την εκάστοτε ταινία που εξεταζεται, στην προκειμένη περίπτωση η ταινία 5.

Για τον χρήστη Α δίνεται:

$$prediction(A, m_5) = (\bar{r}_A) + \frac{(similarity(a, u_1).(r_{(1,5)} - (\bar{r'}_1)) + similarity(a, u_2).(r_{(2,5)} - (\bar{r'}_2)))}{(similarity(a, u_1) + similarity(a, u_2))} = 4.87$$

Με παρόμοιο τρόπο υπολογίζονται οι προβλεπόμενες βαθμολογίες του Α για όλες τις ταινίες που δεν έχει ακόμη βαθμολογήσει. Έπειτα γεμίζει η λίστα προτάσεων με τις Ν ταινίες υψηλότερης βαθμολογίας. Για την υλοποιημένη εφαρμογή αποθηκεύονται οι βαθμολογίες που ξεπερνούν το 3. Άρα εφόσον η ταινία 5 έδωσε $prediction(A, m_5)$ = 4.87, είναι μια καλή επιλογή για τη λίστα και υπάρχουν μεγάλες πιθανότητες η Α να νιώσει ότι αυτή η ταινία ταιριάζει στα γούστα του.

2.2.4 Αποτελεσματικότητα

Στο παρόν παράδειγμα χρησιμοποιήθηκε Pearson Correlation Coefficient για τη μέτρηση της ομοιότητας μεταξύ χρηστών. Φυσικά και έχουν μελετηθεί και άλλες μέθοδοι όπως η Adjusted Cosine Similarity, η Spearman's rank correlation coefficient και η Mean Squared Difference Measure, οι οποίες παρομοίως μετρούν την ομοιότητα. “Εμπειρικά όμως συμπεραίνεται πως για User-Based Recommendation Συστήματα – η Pearson Coefficient υπερισχύει” [7]. Το παραπάνω ενισχύεται από τα λεγόμενα των [45] και [26], καθώς γίνεται αναφορά σε “ακριβέστερα αποτελέσματα” με τη συγκεκριμένη τεχνική.

Παρ' ολη την αποτελεσματικότητα που φαίνεται να κατέχει η μέθοδος, τα πράγματα γίνονται δύσκολα για αντικείμενα τα οποία είναι αρεστά από την πλειοψηφία των χρηστών [6]. Για παράδειγμα, κοντινές βαθμολογίες δύο χρηστών για μια ταινία μικρής απόδοχής από την κινηματογραφική κοινότητα είναι μεγαλύτερης “αξίας” από ότι σε μια ταινία που θεωρητικά είναι αρεστή σε όλους.

3. ΥΛΟΠΟΙΗΜΕΝΟ ΣΥΣΤΗΜΑ ΣΥΣΤΑΣΕΩΝ

3.1 Τεχνική Εμβάθυνση

3.1.1. Περιβάλλον υλοποίησης

Η εφαρμογή υλοποιήθηκε σε λειτουργικό σύστημα Windows 7 64-bit στο προγραμματιστικό περιβάλλον του Android Studio.

3.1.2. Τεχνολογίες που χρησιμοποιήθηκαν

Οι ακόλουθες τεχνολογίες ενσωματώθηκαν στην εφαρμογή:

Butterknife – Annotation Processing (Επεργασία Σχόλιων) είναι μια τεχνική που χρησιμοποιείται από βιβλιοθήκες του Android για τη παραγωγή κώδικα κατά τη διάρκεια συγγραφής του. Κατά τη διάρκεια προγραμματισμού μιας εφαρμογής, κάνοντας χρήση συγκεκριμένων συναρτήσεων που παρέχει η εκάστοτε Annotation Processing βιβλιοθήκη, μειώνονται τα επαναλαμβανόμενα κομμάτια κώδικα. Η βιβλιοθήκη του Butterknife, αντιστοιχίζει κλάσεις που έχουν να κάνουν με την εμφάνιση της εφαρμογής – Views – και βοηθά στον ορισμό αυτών των κλάσεων μέσα σε λίγες γραμμές κώδικα [80, 81].

RecyclerView – Android View για την εμφάνιση των ταινιών σε μορφή λίστας. Αποτέλεσε την τεχνολογία που αντικατέστησε το ListView, καθώς μέσα από ένα RecyclerView γίνεται πιο αποτελεσματική χρήση των πόρων του Κινητού ενώ παρέχεται μεγαλύτερη ευκολία στη διαχείριση μεγάλου όγκου αντικειμένων [82, 83].

JSON – Αποτελεί ένα πρότυπο αρχείων που χρησιμοποιώντας ευνόητο κείμενο με τα δεδομένα να έχουν τη μορφή ζευγαριών <ιδιότητα-τιμή>. Χρησιμοποιείται για τη μετάδοση πληροφορίας μέσω του Δικτύου και δρα ανεξαρτήτως γλώσσας προγραμματισμού. Εξελίχθηκε με βάση της τη γλώσσα της Javascript αλλά από το 2017 και μετά, πολλές γλώσσες εμπεριέχουν κώδικα για την αναπαραγωγή και τη μετάδοση δεδομενων με JSON μορφή (JSON-format data) [87].

GSON – αναπτύχθηκε για τις ανάγκες της Google (ονομάζεται αλλιώς και Google Gson), με σκοπό να μετατρέπει αντικείμενα της Java σε JSON και τούμπαλιν [92].

API – Είναι το ακρωνύμιο για το Application Programming Interface. Μια τέτοια διεπαφή αποτελείται από ένα σύνολο συναρτήσεων, πρωτόκολλων επικοινωνίας και εργαλείων για την ανάπτυξη εφαρμογών. Είναι το κομμάτι του server που χειρίζεται τις συνδέσεις τερματικών σε αυτό. Στην παρούσα εργασία, κάνοντας χρηση του theMovieDatabase API, μου δόθηκε η δυνατότητα, χρησιμοποιώντας τις κατάλληλες συναρτήσεις να έχω πρόσβαση και να αναζητήσω σχετικές ταινίες στην πλατφόρμα αυτή [89, 90].

Retrofit 2 – Το Πρωτόκολλο HTTP είναι από τα κυριότερα πρωτόκολλα μεταφοράς που χρησιμοποιούνται από διαδικτυακές εφαρμογές. Καθορίζει τον τρόπο μεταφοράς δεδομένων μεταξύ δύο συσκευών που ανήκουν σε ένα δίκτυο υπολογιστών, μιας συσκευής που έχει το ρόλο του HTTP Server και μιας άλλης με αυτό του HTTP Client. Τα μηνύματα που στέλνονται από τον HTTP Client στον HTTP Server ονομάζονται HTTP Requests [84]. Το Representational State Transfer (REST) είναι ένα σύνολο κανόνων αρχιτεκτονικής που εφαρμόζονται κατά τη δημιουργία Διαδικτυακών Υπηρεσιών. Χαρακτηριστικό των συστημάτων που ενσωματώνουν το μοντέλο REST είναι η γρήγορη απόδοσή, η αξιοπιστία και η προσαρμοστικότητά σε αλλαγές των παραμέτρων τους [85]. Το Retrofit 2, είναι ένα REST πρόγραμμα για το χειρισμό συναρτήσεων που γίνονται στο theMovieDataBaseAPI [86, 88]. Επιλέχθηκε καθότι, κατά [75] , θεωρείται πιο γρήγορο σα τεχνολογία και επίσης από προσωπική εμπειρία, φαίνεται να προτιμάται από επιχειρήσεις που εργάζονται σε Android Development.

Picasso – Βιβλιοθήκη για το φόρτωμα των εικόνων που αφορούν τις αφίσες των ταινιών [93]. Ενσωματώνει λειτουργίες που μόνο με μια γραμμή κώδικα, μπορούμε να κατεβάσουμε εικόνες που παρέχονται από το server του theMovieDatabase.

Logger – Εμφανίζει οργανωμένα μηνύματα που έχει ορίζει ο χρήστης. Στο περιβάλλον του Android Studio, υπάρχει το εργαλείο Logcat, που καταγράφονται μηνύματα του λογισμικού Android. Χάριν της βιβλιοθήκης Logger, ο χρηστης μπορεί να κάνει ευκολότερο το φιλτράρισμα αυτών και να παραμετροποιήει τον τρόπο που του εμφανίζονται τα μηνύματα [94].Χρησιμοποιήθηκε για αποσφαλμάτωση και για επιβεβάιωση σωστής εισαγωγής των δεδομένων είτε στη ΒΔ είτε στις συναρτήσεις επικοινωνίας με το API του theMovieDatabase.

Logging Interceptor – Χρησιμοποιήθηκε για τη βοήθεια της αποσφαλμάτωσης των API συναρτήσεων. Επιπλέον πληροφορίες μεταξύ των HTTP Requests και Responses, εμφανίζονται στο παράθυρο μηνύματων του Android Studio για να βεβαιωθούμε ότι τα μεταφερόμενα πακέτα δεδομένων, έχουν το σωστό μέγεθος και δεν περιέχουν σφάλματα.

Για το theMovieDataBaseAPI

Το API που χρησιμοποιήθηκε για τις πληροφοριές των ταινιών είναι το The Movie Database (TMDb). Είναι κατασκευασμένο από την κοινότητα χρηστών που το απαρτίζουν και λειτουργεί από το 2008, προσφέροντας δωρεάν πληροφορίες για ταινίες και σειρές. Οποιοσδήποτε προγραμματιστής μπορεί να αιτηθεί μοναδικό κλειδάριθμο για να ενσωματώσει τη λειτουργικότητα της online αυτής βάσης στην εφαρμογή του.

Για τη Βάση Δεδομένων Realm

Η Realm είναι ένα framework βάσης δεδομένων, βασισμένο σε ανοιχτό κώδικα. Χρησιμοποιείται κυρίως σε κινητά Android / iOS και είναι γραμμένη σε C++. Εφαρμογές εταιρειών όπως το Amazon, η Google, Ebay και η IBM, την ενσωματώνουν [76]. Χωρίς να βασίζεται στην SQLite ή σε κάποια άλλη SQL βάση δεδομένων, η Realm γράφει τα δεδομένα με την ελάχιστη δυνατή μετατροπή αυτών [76].

Φάνηκε καλή επιλογή για το μοντέλο της Βάσης Δεδομένων καθώς είναι πιο κοντά στον τρόπο σκέψης του Αντικειμενοστραφή Προγραμματισμού παρά στην καθαρή συγγραφή SQL κώδικα. Είναι μια “Βάση Δεδομένων Αντικειμένων” [78].

Στην πορεία υλοποίησης της εφαρμογής ήρθαν στην επιφάνεια θέματα του API που ήταν δύκολο να κατανοηθεί η πηγή των προβλημάτων. Τελικά ήταν θέμα ασυμφωνίας ενός RealmObject – συγκεκριμένα της κλάσης Movie.java – και του Retrofit API. Ήταν δύσκολη η αποσφαλμάτωση και ψάχνοντάς το έφτασα σε άρθρο [77], το οποίο αξιολογεί και συγκρίνει τεχνολογίες βάσεων δεδομένων για Android Εφαρμογές.

Φάνηκε λίγο ξένη η ιδέα μιας ΒΔ να μη χρησιμοποιεί αυτομάτως αυξανόμενα πρωτεύοντα κλειδιά (Auto-incrementing Primary Keys) και είναι προβληματική η διαχείριση από το framework αυτό η διαχείριση ήδη υπάρχοντων Primary Keys. Επίσης δεν επιτρέπεται από το Realm framework, η χρήση 2 foreign keys στα αντικείμενα - πίνακες UserRatesMovie.java και MovieRecommendeForUser.java . Έτσι μόνο ο πίνακας User έχει πρωτεύον κλειδί PK .

3.1.3. Σχεδιαστικες επιλογες

Το σύστημα δέχεται ως είσοδο βαθμολογίες από το χρήστη αλλά και λέξεις για να κάνει ο χρήστης αναζήτηση ταινιών. Οι παραπάνω πληροφορίες σε συνδυασμό με τις πληροφορίες που παρέχει το the Movie Database API, εμφανίζουν ως εξοδο τις ταινίες που πρόσφατα η χρήστης βαθμολόγησε αλλά και προτεινόμενες γι αυτήν ταινίες.

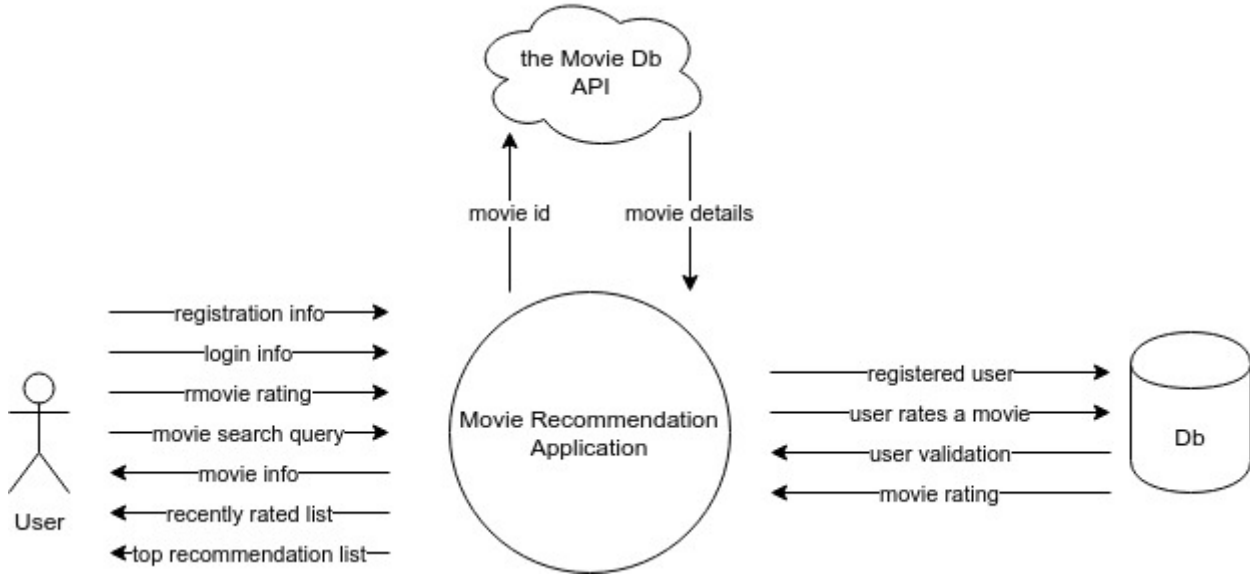
Πρόσθετα, ενσωματώθηκε λειτουργία, κατά την οποία ο χρήστης μπορεί να αναζητήσει ταινίες βάσει χρονολογίας, σκηνοθέτη, ηθοποιών, είδους. Με την επιλογή κάποια ταινίας από τη λίστα εμφανίζεται η περίληψη αυτής, η ημερομηνία πρώτης προβολής στις αίθουσες, ο τίτλος της και η βαθμολογία που έχει δώσει σε αυτήν ο χρήστης. Κάθε χρήστης μπορεί να βαθμολογήσει μόνο μια φορά την κάθε ταινία και δε μπορεί να την αλλάξει αφότου τις έχει δώσει βαθμό.

Για την εφαρμογή , δε χρησιμοποιήθηκε κάποιο συγκεκριμένο μοντέλο αρχιτεκτονικής για την οργάνωση του κώδικα. Εφαρμόστηκε ένα μοντέλο που διαχωρίζει τα Activities, τους Adapters, τα αρχεία κλάσεων του Model (Movie , User) ενώ ξεχωριστά δημιουργήθηκε η MovieRecommendationApp.java κλάση, που περιέχει μεθόδους κοινές ανάμεσα στις java κλάσεις και επίσης αρχικοποιεί συναρτήσεις του theMovieDatabaseApi και της Realm βάσης δεδομένων. Ακόμη περιέχονται βοηθητικές συναρτήσεις που τρέχουν και γεμίζουν τους πίνακες της ΒΔ με τα δεδομένα που χρειάζονται για την παρουσίαση του παραδείγματος του κεφ. 2.2.3 .

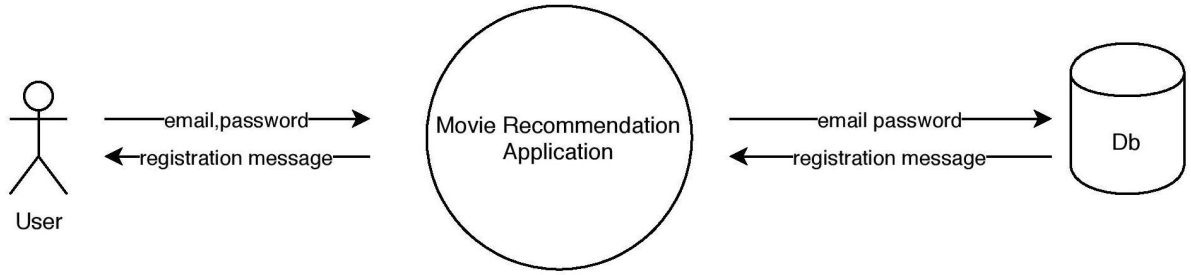
Η Βάση Δεδομένων είναι αποθηκευμένη τοπικά στην εκάστοτε συσκευή που τρέχει την εφαρμογή. Ο αλγόριθμος εύρεσης προτεινόμενων ταινιών (βλ. Παράρτημα Ι) εκτελείται στην Android συσκευή.

3.2 Data Flow Diagrams

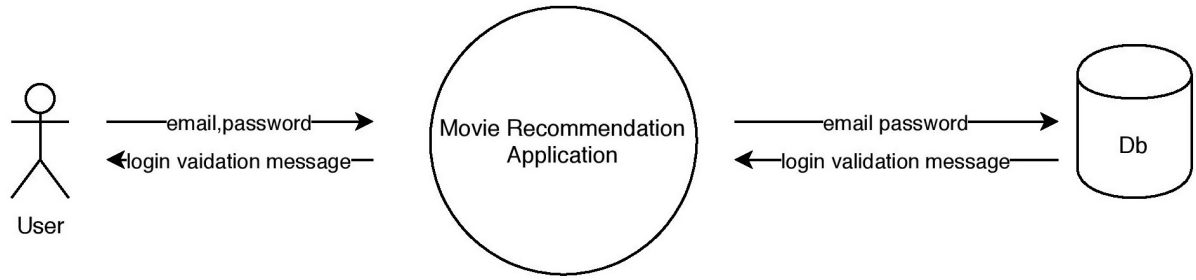
Τα Data Flow Diagrams είναι μια οπτική αναπαράσταση της ροής δεδομένων μέσα από τη διάδραση του χρήστη και του συστήματος που μελετάται. Άρχισαν να χρησιμοποιούνται τη δεκαετία του '70 από επιστήμονες του χώρουν Ανάπτυξης Εφαρμογών. Πρώτη αναφορά σε αυτά έχουμε στο κείμενο του Constantie και Yourdon [91].Θα βοηθήσει για την κατανόηση του σκελετού της εφαρμογής και των λειτουργιών που εμπεριέχονται σε αυτήν. Ένα τυπικό DFD αποτελείται από επίπεδα ξεκινώντας από το Επίπεδο 0 (Level 0) – το λεγόμενο και ως Context Diagram. Είναι μια συνολική αναπαράσταση του συνόλου του Συστήματος. Τα επόμενα επίπεδα, συνήθως κάνουν πιο λεπτομερή την αναπαράσταση των λειτουργιών και τις διασπούν σε μεμονομένα κομμάτια.



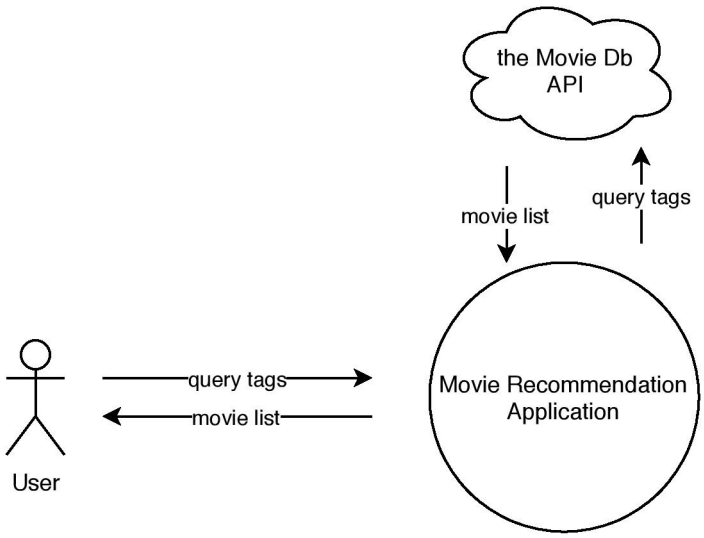
Σχήμα 3: Level 0 - Context Diagram



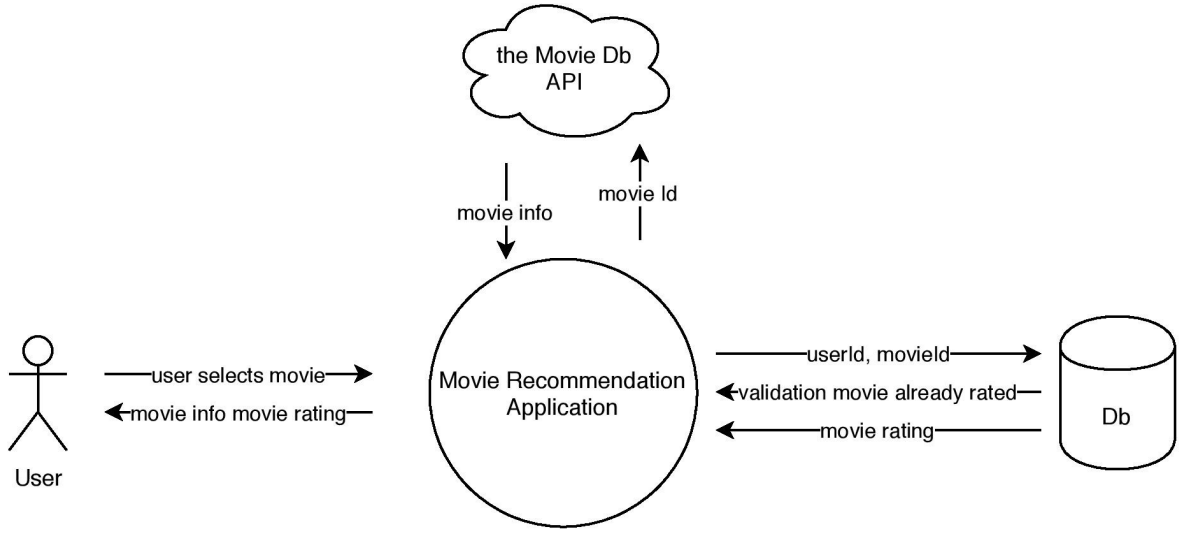
Σχήμα 4: Level 1a - Registration



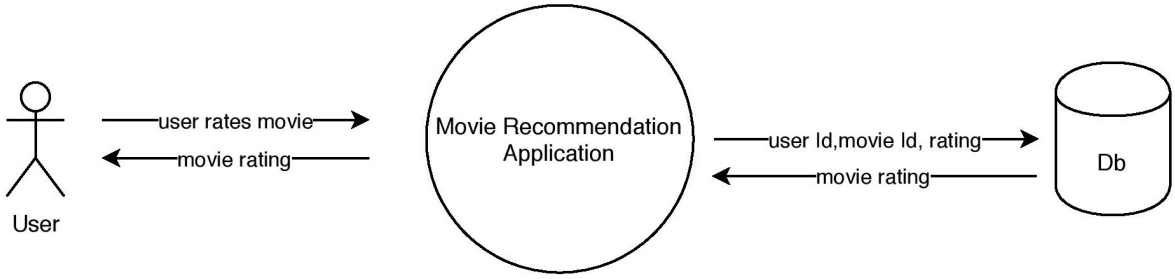
Σχήμα 5: Level 1b - Login



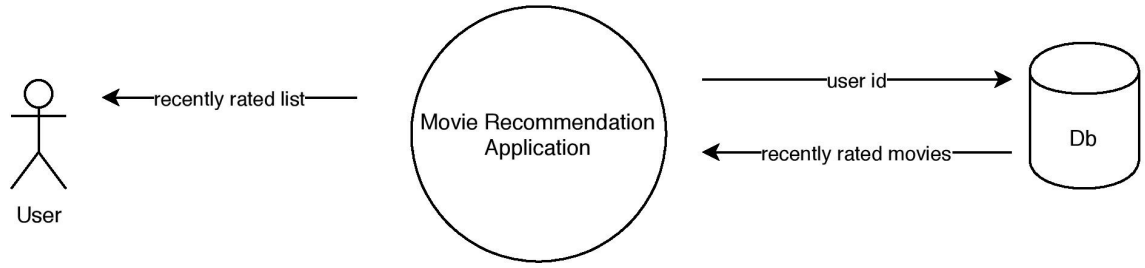
Σχήμα 6: Level 1c - Searching movies



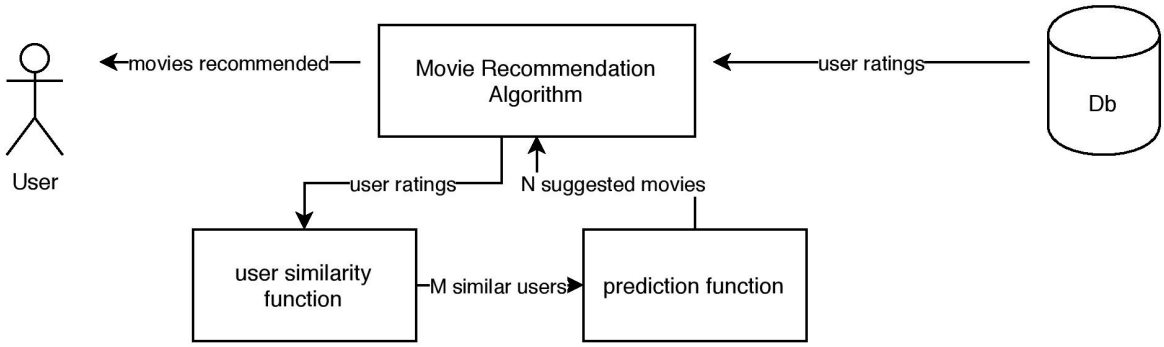
Σχήμα 7: Level 1d – Selecting a Movie



Σχήμα 8: Level 1e – Rating a Movie



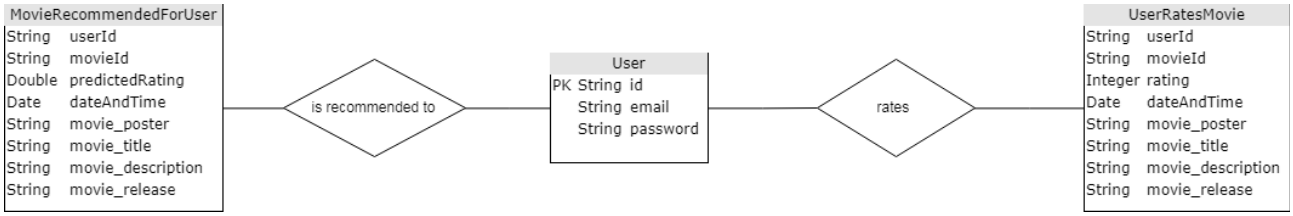
Σχήμα 9: Level 1f - Recently Rated Movie List



Σχήμα 10: Level 1g – Recommended Movie List

3.3 Βάση Δεδομένων

3.3.1. Σχεδιασμός Βάσης



Σχήμα 11: Σχεδιάγραμμα Βάσης Δεδομένων

Κλάση που περιέχει τα στοιχεία της αποθηκευμένης προτεινόμενης ταινίας MovieRecommendedForUser

```
private String userId;
private String movieId;
private Double predictedRating;
private Date dateAndTime;
private String movie_poster;
private String movie_title;
private String movie_description;
private String movie_release;
```

Κλάση που περιέχει τα στοιχεία των εγγεγραμένων χρηστών

User

```
@PrimaryKey
private String id;
private String username;
private String email;
```

```
private String password;
```

Κλάση που περιέχει τις βαθμολογημένες ταινίες από το χρήστη UserRatesMovie

```
private String userId;
private String movieId;
private Integer rating;
private Date dateAndTime;
private String movie_poster;
private String movie_title;
private String movie_description;
private String movie_release;
```

3.3.2. Queries

Για την επιβεβαίωση έγκυρης επιλογής μοναδικού username κατά την εγγραφή του χρήστη στην υπηρεσία:

```
RealmQuery<User> query = realm.where(User.class)
        .equalTo("email", s);
RealmResults<User> result = query.findAll();
```

Για την εγγραφή του χρήστη στην υπηρεσία :

```
realm.beginTransaction();
final User user = realm.createObject(User.class,
        UUID.randomUUID().toString());
user.setEmail(signupEmail.getText().toString());
user.setPassword(signupPassword.getText().toString());
realm.commitTransaction();
```

Για την επιβεβαίωση τον έγκυρων στοιχείων του χρήστη κατά την είσοδό του στην υπηρεσία:

```
User userCheck = realm.where(User.class)
        .equalTo("email", email).findFirst();
User passwordCheck = realm.where(User.class)
        .equalTo("password", password).findFirst();
if (userCheck == null) {

MovieRecommendationApp.getInstance().lastActivity.showErrorDialog("
No such username exists");
    return;
}
if (passwordCheck == null) {

MovieRecommendationApp.getInstance().lastActivity.showErrorDialog("
Wrong password :S");
    return;
```

```
}
```

Για την εμφάνιση της λίστας Ν των προτεινόμενων ταινιών με φθίνουσα σειρά προβλεπόμενης βαθμολογίας από το ΣΣ :

```
RealmQuery<MovieRecommendedForUser> queryRecommendation = realm

        .where(MovieRecommendedForUser.class)

        .equalTo("userId",MovieRecommendationApp.getInstance().getL
oggedInUserId())

        .sort("predictedRating", Sort.DESCENDING);

RealmResults<MovieRecommendedForUser>    movieRecommendations    =
queryRecommendation.findAll();
```

Για την εμφάνιση της λίστας των πιο πρόσφατα βαθμολογημένων ταινιών :

```
RealmQuery<UserRatesMovie> queryRecentlyRated = realm.

        where(UserRatesMovie.class)

        .equalTo("userId",MovieRecommendationApp.getInstance().getL
oggedInUserId())

        .sort("dateAndTime", Sort.DESCENDING);

RealmResults<UserRatesMovie>          moviesRecentlyRated          =
queryRecentlyRated.findAll();
```

Ταινίες που δεν είναι στη λίστα των προτεινόμενων ταινιών

Για την συνθήκη αλήθειας που αφορά το αν μια ταινία είναι ήδη βαθμολογημένη ή όχι από το χρήστη :

```
    Realm realm = Realm.getDefaultInstance();

    RealmResults<UserRatesMovie> result;

    RealmQuery<UserRatesMovie> query =
realm.where(UserRatesMovie.class)

        .equalTo("userId", currentUserId)

        .equalTo("movieId", movieID);

result = query.findAll();

return result.size() != 0;
```

Για την καταχώριση βαθμολογίας στη βάση δεδομένων:

Realm realm = Realm.*getDefaultInstance*();

```
try {
    realm.beginTransaction();
    UserRatesMovie urm = realm.createObject(UserRatesMovie.class);
    urm.setUserId(userid);
    urm.setMovieId(movieid);
    urm.setUserId(userid);
    urm.setDateAndTime(new Date());
    urm.setRating(Math.round(rating));
    urm.setMovie_poster(poster);
    urm.setMovie_title(title);
    urm.setMovie_description(description);
    urm.setMovie_release(release);
    realm.commitTransaction();
} finally {
    realm.close();
```

Για την ανάκτηση από τη βάση δεδομένων, της βαθμολογίας μιας ταινίας από το χρήστη:

```
Realm realm = Realm.getDefaultInstance();

RealmResults<UserRatesMovie> movieRatingResult;
try {
                        RealmQuery<UserRatesMovie>      query      =
realm.where(UserRatesMovie.class)
            .equalTo("userId", currentUserId)
            .equalTo("movieId", movieID);
    UserRatesMovie userRatesMovie = query.findFirst();
          Logger.w("User  has  rated  this  movie  with  a  "  +
userRatesMovie.getRating());
    mRatingBar.setRating(userRatesMovie.getRating());
} finally {
    realm.close();
```

```
}
```

Ταινίες που είναι στη λίστα των προτεινόμενων ταινιών

Για την καταχώριση της στη βάση δεδομένων:

```
    Realm realm = Realm.getDefaultInstance();
try {
    realm.beginTransaction();
    UserRatesMovie urm = realm.createObject(UserRatesMovie.class);
    urm.setUserId(userid);
    urm.setMovieId(movieid);
    urm.setUserId(userid);
    urm.setDateAndTime(new Date());
    urm.setRating(Math.round(rating));
    urm.setMovie_poster(poster);
    urm.setMovie_title(title);
    urm.setMovie_description(description);
    urm.setMovie_release(release);
    realm.commitTransaction();
} finally {
    realm.close();
}
```

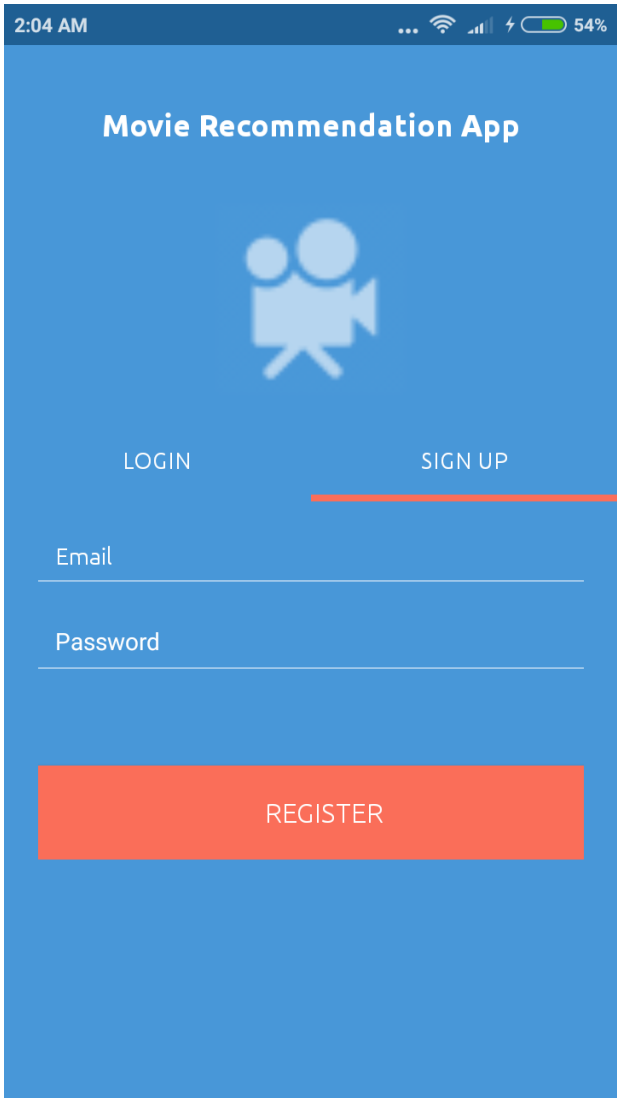
Για τη διαγραφή της ταινίας από τον πίνακα της ΒΔ που αφορά τις προτεινόμενες και την εγγραφή της σε αυτόν με τις βαθμολογημένες:

```
    RealmQuery<MovieRecommendedForUser> query =
realm.where(MovieRecommendedForUser.class)
        .equalTo("userId", activeUserId)
        .and()
        .equalTo("movieId", movieId);
```

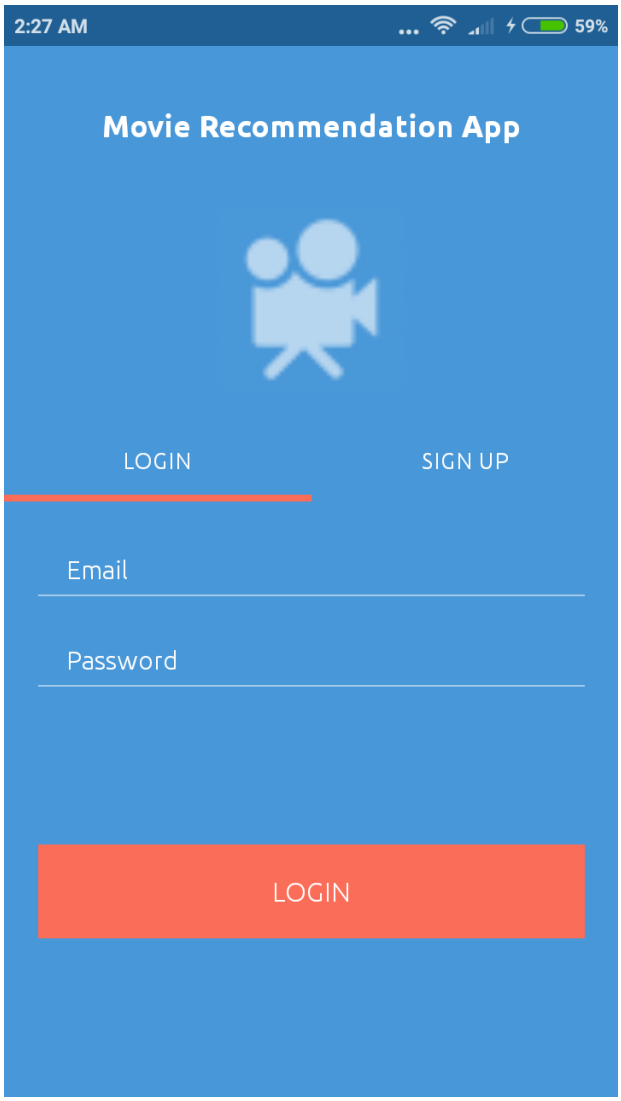
```
final RealmResults<MovieRecommendedForUser> result = query.findAll();
Logger.w("delete recommened movie results before delete" + result.size());
// All changes to data must happen in a transaction
realm.executeTransaction(new Realm.Transaction() {
    @Override
    public void execute(Realm realm) {
        // Delete all matches
        result.deleteAllFromRealm();
    }
});
```

3.4 Εγχειριδιο χρήσης

Κάθε νέος χρήστης για τη χρήση της εφαρμογής θα πρέπει να κατοχυρώσει ένα μοναδικό username και έναν κωδικό. Αφότου έχει συμπληρώσει τα πεδία και πατήσει Register τότε μπορεί να κάνει login στην εφαρμογή από την αντίστοιχη καρτέλα.

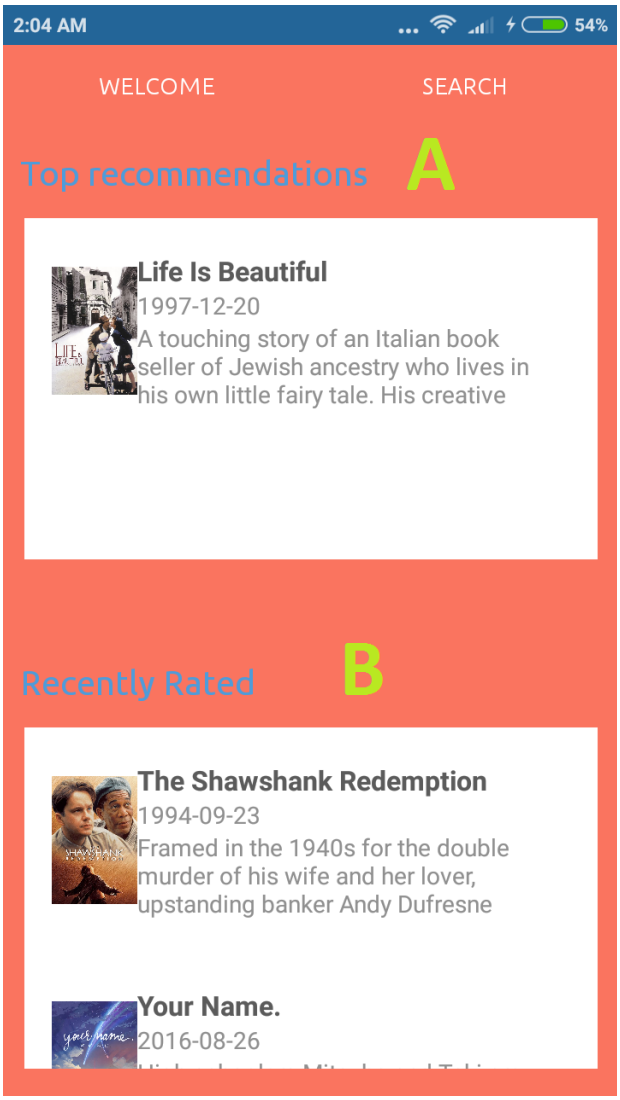


Εικόνα 5: Screenshot - Register Page



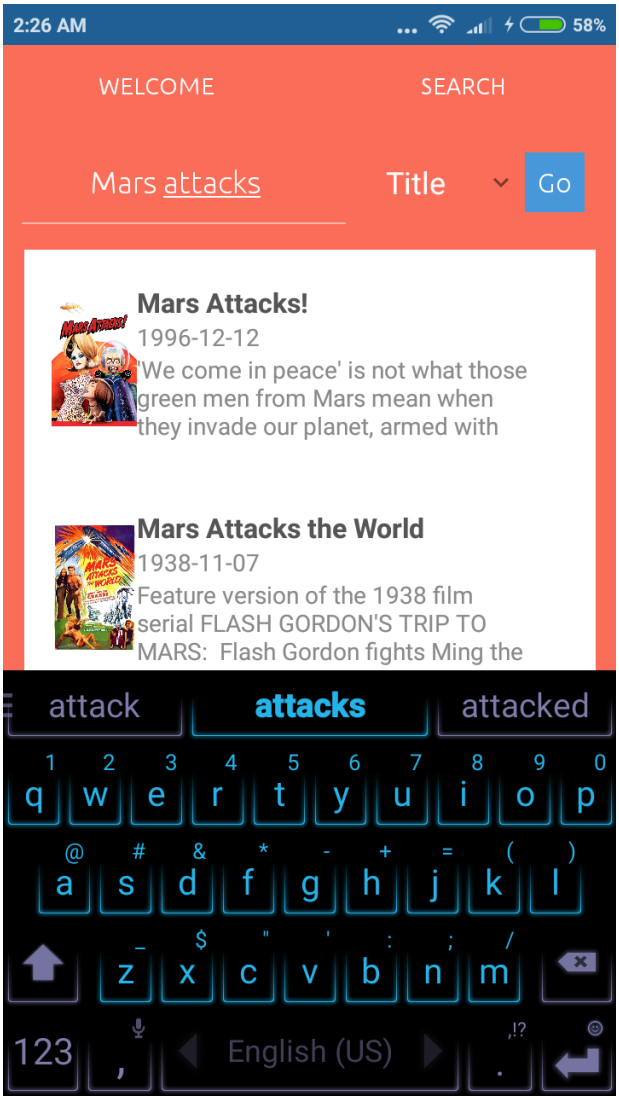
Εικόνα 6: Screenshot - Login Page

Με την είσοδό του σε αυτήν, η πρώτη καρτέλα δείχνει στο πάνω μέρος (A) την ταξινομημένη λίστα με τις προτεινόμενες ταινίες , ενώ στο κάτω (B) φαίνονται οι πιο πρόσφατα βαθμολογημένες ταινίες από το χρήστη.

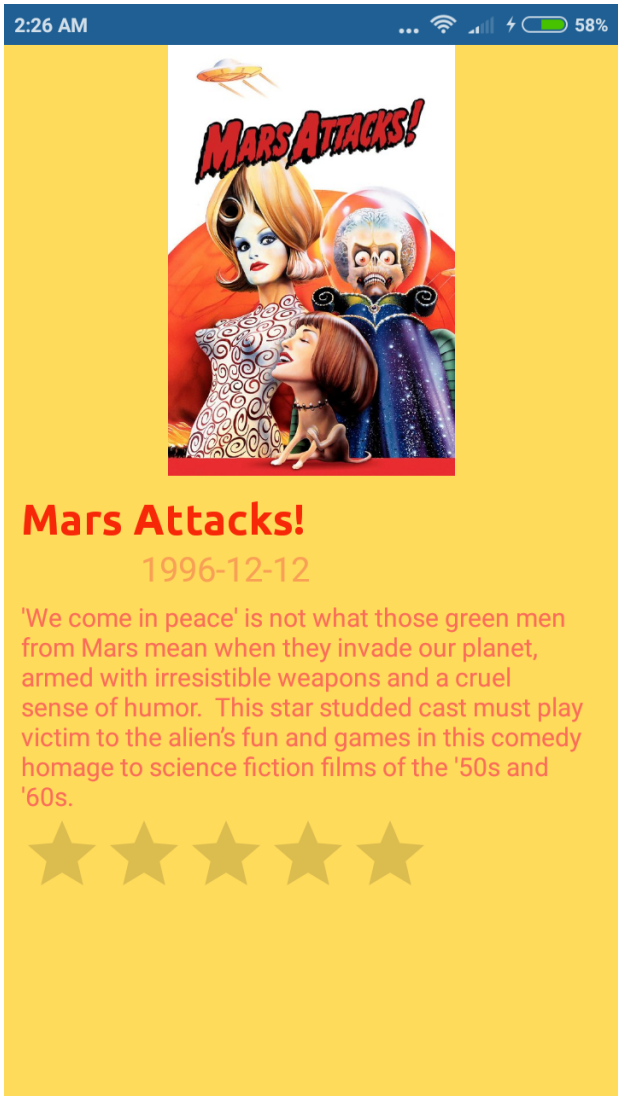


Εικόνα 7: Screenshot - Welcome Page

Στη δεύτερη καρτέλα ο χρήστης μπορεί να αναζητήσει ταινίες και να δει λεπτομέρειες που αφορούν αυτές και έχουν να κάνουν με τη βαθμολογία που ο ίδιος τους έχει βάλει, την ημερομηνία που παρουσιάστηκε στις αίθουσες προβολής,μια μικρή περίληψή της και την αφίσα αυτής. Η αναζήτηση μπορεί να γίνει ανά είδος ταινίας (Drama,Documentary,Horror,...) , χρονιάς, ηθοποιών που παίζουν σε αυτή,σκηνοθετών και τίτλου.



Εικόνα 8: Screenshot - Search Movie by Title



Εικόνα 9: Screenshot - Selecting a Movie

4. ΣΥΜΠΕΡΑΣΜΑΤΑ

Σε μία πρώτη προσέγγιση η υλοποίηση ενός Συστήματος Συστάσεων θα μπορούσε να αποτελείται από μια απλή εισαγωγή των δεδομένων ως είσοδο, στην προκειμένη περίπτωση βαθμολογιών ταινιών, την εφαρμογή κάποιου μαθηματικού τύπου και τέλος την εξαγωγή του τελικού αποτελέσματος, δηλαδή τις καλύτερες επιλογές ταινιών για το χρήστη της εφαρμογής. Ωστόσο, στην πραγματικότητα κάτι τέτοιο γίνεται πιό περίπλοκο καθώς υπάρχουν πολλοί τρόποι για να φτάσει κάποιος στο τελικό αποτέλεσμα. Επιπλέον, υπάρχει και η πιθανότητα αν αλλάξουμε μια παράμετρο, οι προγνώσεις να μην είναι πια έγγυρες – με αποτέλεσμα ο χρήστης να νιώθει ότι το σύστημα αυτό δεν δύναται να αντιληφθεί και να ικανοποιήσει τις προτιμήσεις του .

Επιστήμες όπως η ψυχολογία, η θεωρία παιγνίων, η τεχνητή νοημοσύνη, το marketing, η φυσική μπορούν να ενσωματώσουν τεχνικές Συστημάτων Συστάσεων που θα τις εξελίξει αλλά και να συνεργαστούν στη σχεδίαση του βέλτιστου αλγορίθμου. Ενός αλγορίθμου που θα ξέρει τι ακριβώς επιζητά ο χρήστης και ανά πάσα στιγμή – άμεσα να μπορεί να του κάνει μια συναρπαστική νέα πρόταση. Βέβαια θα πρέπει να γίνει εκτενής μελέτη πάνω στις επιπτώσεις ενός τέτοιου συστήματος στον άνθρωπο αλλά και στους τρόπους θωράκισής του απέναντι σε κακόβουλους χειριστές ή χρήστες αυτού.

Στην εργασία, δόθηκε μεγαλύτερη σημασία σε τεχνολογίες που χρησιμοποιούνται επί το πλείστον από τον κλάδο της Ανάπτυξης Εφαρμογών σε Περιβάλλον Android. Η μορφοποίηση της εφαρμογής έγινε με τέτοιον τρόπο ώστε να είναι ελκυστική στο αγοραστικό κοινό και να μπορεί να υποστηριχθεί από την πλατφόρμα εφαρμογών Android, το Google Play Store.

Ως μελλοντικά θέματα για ενασχόληση προτείνεται η μελέτη των Συστημάτων Συστάσεων σε συνδυσμό με την Τεχνητή Νοημοσύνη. Ακόμη θα μπορούσε να γίνει μια μελέτη για το πόσο πιο γρήγορο θα γινόταν το σύστημα αν οι υπολογισμοί των συναρτήσεων ομοιότητας χρηστών και πρόβλεψης προτίμησης πραγματοποιούνταν σε απομακρυσμένο server και όχι τοπικά, στον client. Μια άλλη ιδέα θα ήταν η υλοποίηση ενός Συστήματος Συστάσεων που θα σύστηνε στο σχεδιαστή ενός τέτοιου Συστήματος τις ιδανικές επιλογές του βάσει παραμέτρων όπως το μέγεθος των δεδομένων εισόδου, τις απαιτήσεις που έχει για τη χρονική απόκριση του Συστήματος, το αν θα ήθελε η μέθοδός του να είναι Item-based ή Content-Based , κλπ.

ΠΙΝΑΚΑΣ ΟΡΟΛΟΓΙΑΣ

Ξενόγλωσσος όρος	**Ελληνικός Όρος**
Recommendation Systems	Συστήματα Συστάσεων
Collaborative Filtering	Συνεργατικό Φιλτράρισμα
Content-Based	Βάσει Περιεχομένου
Limited Content Analysis	Περιορισμένης Ανάλυσης Περιεχομένου
Overspecialization	Υπερειδίκευσης
Knowledge-Based	Βάσει Γνώσης
Constraint-Based	Βάσει Περιορισμών
Marketing	Αγοραλογία
Community-Based	Βάσει Κοινότητας
Social Recommender Systems	Κοινωνικά Συστήματα Συστάσεων
Framework	Πλαίσιο
Ensemble Analysis	Ανάλυση Συνόλων
Machine Learning	Νοημοσύνη Μηχανών
Prediction	Πρόγνωση
Active User	Ενεργός Χρήστης
Memory-Based	Βάσει Μνήμης
Model-Based	Βάσει Μοντέλου
User-Based	Βάσει Χρήστη
Cosine Similarity	Ομοιότητα του Συνημιτόνου
Mean Squared Differences	Μέσες Διαφορές Τετραγώνων
Adjusted Cosine Similarity	Ομοιότητα Μετασχηματισμένου Συνημιτόνου
Item-Based	Βάσει Αντικειμένου
Bayesian Clustering	Ομαδοποίηση Bayesian
Latent Semantic Analysis	Λανθάνουσα Σημειολογική Ανάλυση
Latent Dirichlet Allocation	Λανθάνουσα Κατανομή Dirichlet
Pearson’s Correlation Coefficient	Συντελεστής Συσχέτισης Pearson
Similarity	Ομοιότητα
Information Retrieval	Ανάκτηση Πληροφοριών
Mean Absolute Error	Μέσο Απόλυτο Σφάλμα
Root Mean Square Error	Μέση Τετραγωνική Ρίζα Σφάλματος
Precision	Ακρίβεια
Recall	Ανάκληση
False – Positives	Ψευδοθετικά
Reference Ranking	Βαθμολογημένη Αναφορά
Utility Based Ranking	Βαθμολογημένη Χρησιμότητα
Cold Item	Αντικείμενο όχι σε Ζήτηση
Hot Item	Αντικείμενο σε Ζήτηση
Cold Start	Εκκίνησης εν Ψυχρώ
Fraud	Νωθεία
Profile Injection Attack	Επίθεση Έγχυσης Προφίλ
Privacy	Ιδιωτικότητα
Sparsity	Αραιότητα
Novelty	Καινοτομία
Scalability	Ευελιξία

HyperText Transfer Protocol	Πρωτόκολλο Μεταφοράς Υπερκειμένου
Application Programming Interface	Διεπαφή Προγραμματισμού Εφαρμογής
Debugging	Αποσφαλμάτωση
Operation System	Λειτουργικό Σύστημα
Auto-Incrementing	Αυτομάτως Αυξανόμενα
Database	Βαση Δεδομένων
Data Flow	Ροή Δεδομένων
Gain	Κέρδος
Server	Εξυπηρετητής
Browser	Φυλλομετρητής
Extended Logistic Regression	Αναπτυγμένη Λογιστική Παλινδρόμηση
Fair Information Practices	Θεμιτές Πρακτικές σχετικά με τις Πληροφορίες

ΣΥΝΤΜΗΣΕΙΣ – ΑΡΚΤΙΚΟΛΕΞΑ – ΑΚΡΩΝΥΜΙΑ5

RS	Recommendation System
CF	Collaborative Filtering
ΣΣ	Συστημα Συστάσεων
SVD	Singular Value Decomposition
Db	Database
IMDb	Internet Movie Database
ACM	Association for Computing Machinery
TOIS	Transactions on Information Systems
MAE	Mean Absolute Error
RMSE	Root Mean Square Error
NMAE	Normalized Mean Absolute Error
ROC Curve	Receiver Operating Characteristic Curve
NDPM	Normalized Distance based Performance Measure
NDCG	Normalized Discounted Cumulative Gain
DCG	Discounted Cumulative Gain
AU ROC Curve	Area Under the ROC Curve
HTTP	HyperText Transfer Protocol
API	Application Programming Interface
GSON	Google Serialization Object Notation
JSON	JavaScript Object Notation
SQL	Structured Query Language
IBM	International Business Machines
OS	Operation System
iOS	iPhone OS
PK	Primary Key
FK	Foreign Key
ΒΔ	Βάση Δεδομένων
DFD	Data Flow Diagram
NB	Naive Bayes
TAN	Tree Augmented Naive Bayes
ELR	Extended Logistic Regression

ΠΑΡΑΡΤΗΜΑ Ι

Υλοποίηση αλγορίθμου

```
SIMILARITY_PILLOW = 0.5;
PREDICTION_PILLOW = 3.0;

void MovieRecommendationAlgorithm(){
    String activeuserID = "xxxxxxxxxxx";
    RealmResults users = RealmDatabase.query(SELECT * FROM RealmDatabase.Users WHERE userid != activeuserID);

    //lista IDs geitonwn
    List<String> neighbours = new List();
    for ( user in users) {
        if ( similarity(activeuserID, user.getId() > = SIMILARITY_PILLOW){
            neighbours.add(user.getId());
        }
    }

    Map<String,Double> movieMapActiveUser = new Map();

    RealmResults activeUserResults = RealmDatabase.query(SELECT * FROM RealmDatabase.UserRatesMovie WHERE userid = activeuserID in );
    //apothikevoume sto movieMapActiveUser to zevgos <Movie ID, Movie Rating>
    for ( userRatesMovie in activeUserResults) {

movieMapActiveUser.put(userRatesMovie.getMovieID(),userRatesMovie.getRating());
    }

    //oles oi vathmologhmenes tainies
    RealmResults movieIds = RealmDatabase.query(SELECT * FROM RealmDatabase.UserRatesMovie);
```

```
        String currentMovieId;
        List<String> notRatedMoviesIDs = new List();
        for ( userRatesMovie in movieIDs){
                currentMovieId = userRatesMovie.getMovieID();
                //an den einai mesa sth lista twn ids twn tainiwn pou exei vathmologhsei o active
user
                if (! movieMapActiveUser containsKey(currentMovieId) && (!notRatedMoviesIDs
containsKey(currentMovieId))){
                        notRatedMoviesIDs.add(currentMovieId);
                }
        }

        //gia oles tis tainies pou den exei vatmhologhsei akomh o active user
        for ( movieId in notRatedMoviesIDs){
                //vriskoume thn provlepomenh vathmologia
                prediction(activeuserID,movieId[i],neighbours);
        }

  }


 Double similarity(String activeuserID, string user_i_ID){
        Map<String,Double> helperA = new Map();
        Map<String,Double> helperB = new Map();

        List<String> sameMovieIds = new List();

        Double activeUserAVG = avgRating(activeuserID);
        Double user_i_AVG = 0.0;

        // oles oi tainies pou exei vathmologhsei o active user
        RealmResults activeUserResults = RealmDatabase.query(SELECT * FROM
RealmDatabase.UserRatesMovie WHERE userid = activeuserID);
        //oles oi tainies pou exei vathmologhsei o user i
        RealmResults user_i_Results = RealmDatabase.query(SELECT * FROM
RealmDatabase.UserRatesMovie WHERE userid = user_i_ID);
```

```
//ta ids kai oi vathmologies tou active user apothikevontai ws zevgos sth helperA
for ( user in activeUserResults){
        helperA.put(user.getMovieID(),user.getRating());
}

//ta ids kai oi vathmologies tou user i  apothikevontai ws zevgos sth helperB
//an exoun koino id me aftes sto helperA
for ( user in user_i_Results){
        String currentMovieId = user.getMovieID();
        Double currrentMovieRating  = user.getRating();

        //an vrhkame koinh tainia tou active user kai tou user i
        // tha krathsouem to id thhs kai th vathmologia sth lista
        if helperA.containsKey(currentMovieId){
                helperB.put(currentMovieId,currrentMovieRating);
                user_i_AVG = user_i_AVG + currrentMovieRating;
                //lista pou kratame to ID twn koinwn tainiwn
                sameMovieIds.add(currentMovieId);
        }
}
//ypologismos M.O. aftwn twn tainiwn
user_i_AVG = user_i_AVG / helperB.size();

Double K_sum = 0.0;
Double L_sum = 0.0;
Double M_sum = 0.0;

// gia oles tis koines tainies twn active user kai user i
//pearson correlation coefficient calculation
for ( movieId in sameMovieIds){
        String currentId = movieId[i];
        // h vathmologia ths tainias me to current id gia ton active user A
        Double r_A = helpreA.get(currentId);
```

```
            // h vathmologia ths tainias me to current id gia ton user i
            Double r_i = helpreB.get(currentId);
            K_sum = K_sum + (r_A - activeUserAVG) * (r_i -user_i_AVG);
            L_sum = L_sum + (r_A - activeUserAVG)^2;
            M_sum = M_sum + (r_i - user_i_AVG)^2;
        }

        return ( K_sum / (sqrt( L_sum) * sqrt(M_sum)));
    }

Double prediction(activeuserID, notYetRatedMovieId, neighboursList){
        activeAVG = avgRating(activeuserID);
        Double A = 0.0;
        Double B = 0.0;
        for ( user in neighboursList) {
            avgRating(user[i]);
            A = A + similarity ( activeuserID , user[i]) *
getUser_i_MovieRating(user[i],notYetRatedMovieId) - avgRating(user[i]);
            B = B + similarity(activeuserID,user[i]);
        }

        final Dobule prediction  = activeAvg  + A / B
        if (prediction < PREDICTION_PILLOW )
            return;
        int movieId = notYetRatedMovieId;
        Movie m = TheMovieDatabaseAPI.getMovieDetails(movieId);
        //an h tainia einai hdh stis proteinomeenes gia ton  active user
        //thn diagrafoume apo th database gia na thn antikatasthsoume me neoterh timh ths
        if !(movieIsUnique(notYetRatedMovieId,activeuserID)){
            deleteMovie(notYetRatedMovieId,activeuserID);
        }
        addToDatabase(m);
        RealmDatabaseObject MovieRecommendedForUser movieRecForUser;
        movieRecForUser.setMovieId(notYetRatedMovieId);
        movieRecForUser.setPredictedRating(prediction);
```

```
        movieRecForUser.setUserId(activeUserId);
        movieRecForUser.setMovieDetails(new
Date(),m.getOverview(),m.getPoster(),m.getReleaseDate(),m.getTitle());
        RealmDatabase.save(movieRecForUser);
    }

boolean movieIsUnique(String movieId, String activeuserID){
        size = 0;
        for ( entry in Database ){
                if ((entry[i].getUserId() == activeuserID) && (entry[i].getMovieID() == movieId)){
                        size++;
                }
        }
        return size == 0;
    }

 void deleteMovie(String movieId, String activeuserID){
        for ( entry in Database ){
                if ((entry[i].getUserId() == activeuserID) && (entry[i].getMovieID() == movieId)){
                        RealmDatabase.delete(entry[i])
                }
        }
    }

Double getUser_i_MovieRating(String userId, String movieId){
        Double result;
        for ( entry in Database ){
                if ((entry[i].getUserId() == activeuserID) && (entry[i].getMovieID() == movieId)){
                        result = entry[i].getRating();
                        break;
                }
        }
        return result;
    }
```

```
Double avgRating(String userId){
        Double userAvg = 0.0;
        Database userResults;
        for ( user in userResults) {
                userAVG = userAVG  + user[i].getRating();
        }
        return (userAVG / userResults.size());
  }
```

ΑΝΑΦΟΡΕΣ

[1] J. Huttner, "*From Tapestry to SVD A Survey of the Algorithms That Power Recommender Systems*" , Haverford College Department of Computer Science, 8 May 2009, pp. 1-32.

[2] D. Goldberg, D. Nichols, B. M. Oki and T. Douglas, "Using collaborative filtering to weave an information Tapestry," in *Communications of the ACM*, vol. 35, Dec 1992, pp.1-10.

[3] T. Segaran, *"Programming Collective Intelligence - Building Smart Web 2.0 Applications,"* O'Reilly, 2007.

[4] K. Manoj, D.K. Yadav, S. Ankur and G. K. Vijay, "A movie Recommender System : MOVREC" in *International Journal of Computer Applications* (0975 – 8887) Volume 124 – No.3, August 2015, pp. 1-11.

[5] I. Portugal,P. Alencar and D. Cowan, "*The Use of Machine Learning Algorithms in Recommender Systems: A Systematic Review"*, Volume 97, 1 May 2018, pp. 205-227.

[6] D. Jannach, M. Zanker, A. Felfernig and G. Friedrich, *"An Introduction To Recommender Systems,"* Camrbidge University Press, 2011.

[7] J.L. Herlocker, J.A. Konstan, Al Borschers and John Riedl, "*An Algorithmic Framework for Perfoming Collaborative Filtering,"* ACM Press, 1999, pp.230-237.

[8] F. Ricci, L. Rokach and S. Bracha, *"Recommender Systems Handbook",* Springer, 2011.

[9] C. Aggarwal, "*Recommender Systems the Textbook,"* Springer, 2016.

[10] P. Melville and V. Sinhwani, *"Encyclopedia of Machine Learning",* Chapter No: 00338, 22 April 2010, pp. 1-9.

[11] R. Burke, "*Hybrid web recommender systems. In: The Adaptive Web*," Springer Berlin / Heidelberg, 2007, pp. 377–408.

[12] J. Bobadilla, F. Ortega, A. Hernando and A. Gutierrez, "*Recommender systems survey. Knowledge-Based Systems" ,* 2013, pp.109–132.

[13] O. Arazy, N. Kumar and B. Shapira, "*Improving social recommender systems",* IT Professional, 2009, pp.38–44.

[14] D. Ben-Shimon, A. Tsikinovsky, L. Rokach, A. Meisels, G. Shani and L. Naamani, "Recommender system from personal social networks," in *K. Wegrzyn-Wolska, P.S. Szczepaniak (eds.) AWIC ,Advances in Soft Computing*, vol. 43, Springer, 2007, pp. 47–55.

[15] J. Golbeck, "*Generating predictive movie recommendations from trust in social networks,"* Trust Management, 4th International Conference, iTrust 2006, Pisa, Italy, May 16–19, 2006, Proceedings, pp. 93–104.

[16] R.R. Sinha and K. Swearingen, "*Comparing recommendations made by online systems and friends,"* DELOS Workshop: Personalisation and Recommender Systems in Digital Libraries , 2001.

[17] help.imdb.com/article/imdb/discover-watch/recommended-for-you-faqs/GPZ2RSPB3CPVL86Z?ref_=cons_tt_rec_lm [Προσπελάστηκε 28/3/2018]

[18] A. Ahmed, B. Kanagal, S. Pandey, V. Josifovski, L. G. Pueyo and J. Yuan, "Latent fact or models with additive and hierarchically - smoothed user preferences," in *Proceedings of the sixth ACM international conference on Web search and data mining*, February 2013, pp. 385-394.

[19] J. Liu, P. Dolan and E. R. Pedersen, "Personalized news recommendation based on click behavior," in *Proceedings of the 15th international conference on Intelligent user interfaces*, February 2010, pp. 31-40.

[20] H. Steck, "Evaluation of recommendations: rating-prediction and ranking," in *Proceedings of the 7th ACM Conference on Recommender Systems*, 2013 October, pp. 213-220.

[21] G. Linden, B. Smith, and J. York, "*Amazon.com Recommendations Item-to-Item Collaborative Filtering*," Industry Report, 2003 January-February, the IEEE Computer Society.

[22] L.S. Chen, F. H. Hsu, M. C. Chen and Y. C. Hsu, "*Developing recommender systems with the consideration of product profitability for sellers,"* Information Sciences, 2008, pp.1032-1048.

[23] P. Melville, R. J. Mooney and R. Nagarajan, "Content-boosted collaborative filtering for improved recommendations," In *Proceedings of the eighteenth national conference on artificial intelligence* (AAAI-02), Edmonton, Alberta, 2002, pp. 187–192.

[24] *www.medium.com/the-graph/how-recommender-systems-make-their-suggestions-da6658029b76* [Προσπελάστηκε 28/3/2018]

[25] A. Lampropoulos and G. Tsihrintzis, "*Machine Learning Paradigms Applications in Recommender Systems*," Intelligent Systems Reference Library, Volume 92, Springer, 2015.

[26] J.S. Breese, D. Heckerman, C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering", in *Proceedings of Fourteenth Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann, 1998, pp. 43–52.

[27] B. Sarwar, G. Karypis, J. Konstan and John Riedl, "*Item-Based Collaborative Filtering Recommendation Algorithms",* Department of Computer Science and Engineering University of Minnesota, Minneapolis, May 2001.

[28] Z. Cheng, N. Hurley, "Robust collaborative recommendation by least trimmed squares matrix factorization," in *Proceedings of the 2010 22Nd IEEE International Conference on Tools with Artificial Intelligence* - Volume 02, ICTAI '10, IEEE Computer Society, 2010, pp. 105–112.

[29] B. Mobasher, R.D. Burke, J.J. Sandvig, "Model-based collaborative filtering as a defense against profile injection attacks," in *AAAI Press* ,2006.

[30] Z. Cheng, N. Hurley, "Trading robustness for privacy in decentralized recommender systems," in *IAAI, Proceedings of the Twenty-First Conference on Innovative Applications of Artificial Intelligence*, AAAI , 2009.

[31] X. Su, R. Greiner, T.M. Khoshgoftaar and X. Zhu, "Hybrid collaborative filtering algorithms using a mixture of experts," in *Web intelligence*, 2007, pp. 645–649.

[32] en.wikipedia.org/wiki/Long_tail [Προσπελαστηκε 26/4]

[33] C. Anderson, "*The long tail: why the future of business is selling less of more,"* Hyperion, 2006.

[34] www.wired.com/2004/10/tail/ [Προσπελάστηκε 26/4]

[35] www.nytimes.com/2006/08/10/books/10manly.html [Προσπελάστηκε 26/4]

[36] www.kdnuggets.com/news/2007/n08/3i.html [Προσπελάστηκε 27/4]

[37] A. Toscher, M. Jahrer, and R. Legenstein, "Improved neighborhood-based algorithms for large-scale recommender systems," in *Proceedings of the ACM SIGKDD Workshop on Large Scale Recommenders Systems and the Netflix Prize at KDD '08* (Las Vegas), 2008.

[38] S. Funk, "*Netflix update: Try this at home ,"* 2006, URL sifter.org/~simon/journal/20061211.html

[39] J.L. Herlocker, J.A. Konstan, L.G. Terveen and J.T. Riedl, "Evaluating collaborative filtering recommender systems," ACM Trans. Inf. Syst. 22(1), 2004, pp. 5–53.

[40] www.nytimes.com/2009/09/22/technology/internet/22netflix.html?_r=0 [Προσπελάστηκε 30/4]

[41] U. Shardanand and P. Maes, "Social information filtering: Algorithms for automating "word of mouth"," in CHI '95: Proc. of the SIGCHI Conf. on Human factors in Computing Systems,. ACM Press/Addison-Wesley Publishing Co., 1995, pp. 210–217.

[42] en.wikipedia.org/wiki/Recommender_system#Content-based_filtering [Προσπελάστηκε 7/6]

[43] Y. Koren, "The Bellkor solution to the Netflix grand prize.", Netflix prize documentation, 81, 2009. www.netflixprize.com/assets/GrandPrize2009_BPC_BellKor.pdf

[44] www.the-ensemble.com/

[45] J. Herlocker, J. Konstan,, and J. Riedl. "*An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms,"* Information Retrieval, 2002, pp. 287–310.

[46] S. K. Lam, and J. Riedl, "Shilling recommender systems for fun and profit," in *WWW '04: Proceedings of the 13th international conference on World Wide Web*, New York , 2004, pp. 393–402.

[47] R. Burke, B. Mobasher, R. Bhaumik, and C. Williams, "Segment-based injection attacks against collaborative filtering recommender systems.," in *ICDM '05: Proceedings of the fifth IEEE international conference on data mining*, 2005, pp. 577–580.

[48] A.I. Schein, A. Popescul, L. H. Ungar and D. M. Pennock, "Methods and metrics for cold-start recommendations," in: *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, 2002, pp. 253–260.

[49] R. J. Mooney and L. Roy, "Content-based book recommending using learning for text categorization," in *Proceedings of the fifth ACM conference on digital libraries*, June 2000,pp. 195–204.

[50] M. J. Pazzani and D. Billsus, "*Learning and revising user profiles: The identification of interesting web sites,"* 1997, Machine Learning, pp. 313–331.

[51] A.S. Harpale and Y. Yang, "Personalized active learning for collaborative filtering," in *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval*, Singapore, 2008, pp. 91–98.

[52] R. Jin and L. Si, "A Bayesian approach toward active learning for collaborative filtering," in *UAI '04: Proceedings of the 20th conference on uncertainty in artificial intelligence*, Arlington: AUAI Press, 2004, pp. 278–285.

[53] expandedramblings.com/index.php/ebay-stats/ [Προσπελάστηκε 15/7]

[54] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Analysis of recommendation algorithms for e-commerce," in *Proceedings of the 2nd ACM Conference on Electronic Commerce(EC '00)* (Minneapolis, MN), ACM, 2000, pp. 158–167.

[55] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, "*Eigentaste: A constant time collaborative filtering algorithm*," Information Retrieval 4, 2001, no. 2, 133–151.

[56] Y.Y. Yao, "*Measuring retrieval effectiveness based on user preference of documents,"* J. Amer. Soc. Inf. Sys 46(2), 1995, pp. 133–145.

[57] G.A. Fredricks and R.B. Nelsen, "*On the relationship between spearman's rho and kendall's tau for pairs of continuous random variables,"* Journal of Statistical Planning and Inference 137(7), 2007, pp. 2143–2150.

[58] K. Järvelin and J. Kekäläinen, *"Cumulated gain-based evaluation of ir techniques,"* ACM Trans. Inf. Syst. 20(4), 2002, pp. 422–446.

[59] www.theguardian.com/technology/2018/apr/04/facebook-cambridge-analytica-user-data-latest-more-than-thought [Προσπελάστηκε 20/7]

[60] www.nytimes.com/2018/03/17/us/politics/cambridge-analytica-trump-campaign.html [Προσπελάστηκε 20/7]

[61] www.bbc.com/news/topics/c81zyn0888lt/facebook-cambridge-analytica-data-scandal [Προσπελάστηκε 20/7]

[62] N.F. Awad, and M. S. Krishnan, "*The personalization privacy paradox: An empirical evaluation of information transparency and the willingness to be profiled online for personalization,"* MIS Quarterly 30(1), 2006, pp. 13–28.

[63] R. K. Chellappa and R. G. Sin, "*Personalization versus privacy: An empirical examination of the online consumer's dilemma,"* Information Technology and Management 6(2), 2005, pp. 181–202.

[64] L. F. Cranor, 'I didn't buy it for myself': privacy and ecommerce personalization," in: *WPES*, 2003, pp. 111–117.

[65] A. Kobsa, "Privacy-enhanced web personalization," in *P.Brusilovsky, A. Kobsa, W. Nejdl (eds.) The Adaptive Web*, Springer-Verlag, 2007, pp. 628–670.

[66] E. Toch, Y. Wang and L. F. Cranor, *"Personalization and privacy: a survey of privacy risks and remedies in personalization-based system*s," in *User Modeling and User-Adapted Interaction* ,22(1–2), 2012, pp. 203–220.

[67] A.J. Jeckmans, M. Beye, Z. Erkin, P. Hartel, R. L. Lagendijk, and Q. Tang, "Privacy in recommender systems," in *Social Media Retrieval, Computer Communications and Networks*, Springer, 2013, pp. 263–281.

[68] OECD: Recommendation of the council concerning guidelines governing the protection of privacy and transborder flows of personal data. Tech. rep., Organization for Economic Cooperation and Development (1980).

[69] C. Aggarwal and P. Yu, *Privacy-preserving data mining: models and algorithms.* Springer, 2008.

[70] J. Canny, "*Collaborative filtering with privacy via factor analysis,*" ACM SIGR Conference, 2002, pp. 238–245.

[71] H. Polat and W. Du, "*Privacy-preserving collaborative filtering using randomized perturbation techniques,"*+ IEEE International Conference on Data Mining, 2003, pp. 625–628.

[72] I. Gunes, C. Kaleli, A. Bilge, and H. Polat, "Shilling attacks against recommender systems: a comprehensive survey," in *Artificial Intelligence Review,* 42(4), 767–799, 2014.

[73] H. Polat and W. Du, “SVD-based collaborative filtering with privacy,” in *ACM symposium on Applied Computing*, 2005, pp. 791–795.

[74] L. Ungar, D. Foster, E. Andre, S. Wars, F. S. Wars, D. S. Wars and J. H. Whispers, “Clustering methods for collaborative filtering,” in *Proceedings of AAAI Workshop on Recommendation Systems*, 1998, AAAI Press.

[75] B. Sarwar, G. Karypis, J. Konstan and J. Reidl, “Item-based collaborative filtering recommendation algorithms,” in *Proceedings of 10th International Conference on World Wide Web*, 2001, pp. 285–295.

[76] instructure.github.io/blog/2013/12/09/volley-vs-retrofit/ [Προσπελάστηκε 21/7]

[77] academy.realm.io [Προσπελάστηκε 20/3]

[78] kpgalligan.tumblr.com/post/133281929963/my-talk-at-droidcon-uk [Προσπελάστηκε 21/7]

[79] academy.realm.io/posts/realm-primary-keys-tutorial/ [Προσπελάστηκε 21/7]

[80] R. Greiner, X. Su, B. Shen and W. Zhou, *“Structural Extension to Logistic Regression: Discriminative Parameter Learning of Belief Net Classifiers,”* Machine Learning, 2005, 59(3), pp. 297 - 322

[81] medium.com/@robhor/annotation-processing-for-android-b7eda1a41051 [Προσπελάστηκε 26/7]

[82] stablekernel.com/the-10-step-guide-to-annotation-processing-in-android-studio/ [Προσπελάστηκε 26/7]

[83] winneredge.wordpress.com/2015/10/31/what-is-recycler-view-in-android/ [Προσπελάστηκε 26/7]

[84] developer.android.com/guide/topics/ui/layout/recyclerview [Προσπελάστηκε 26/7]

[85] tutorialspoint.com/http/http_requests.htm [Προσπελάστηκε 26/7]

[86] en.wikipedia.org/wiki/Representational_state_transfer [Προσπελάστηκε 26/7]

[87] en.wikipedia.org/wiki/JSON [Προσπελάστηκε 26/7]

[88] guides.codepath.com/android/Consuming-APIs-with-Retrofit [Προσπελάστηκε 26/7]

[89] medium.freecodecamp.org/what-is-an-api-in-english-please-b880a3214a82 [Προσπελάστηκε 26/7]

[90] en.wikipedia.org/wiki/Application_programming_interface [Προσπελάστηκε 26/7]

[91] E. Yourdon and L. Constantine, *Structured Design: Fundamentals of a Discipline of Computer Program and System Design,* Prentice-Hall, 1979.

[92] en.wikipedia.org/wiki/Gson [Προσπελάστηκε 27/7]

[93] stacktips.com/tutorials/android/how-to-use-picasso-library-in-android [Προσπελάστηκε 27/7]

[94] github.com/noveogroup/android-logger [Προσπελάστηκε 27/7]