Τμήμα Μηχανικών Η/Υ και Πληροφορικής Πανεπιστημίου Ιωαννίνων
ΜΥΕ047: *Αλγόριθμοι για Δεδομένα Ευρείας Κλίμακας*
Ακαδημαϊκό Έτος 2019-20

Διδάσκων: Σπύρος Κοντογιάννης

2ο Σετ Ασκήσεων: ΚΑΝΟΝΕΣ ΣΥΣΧΕΤΙΣΗΣ & ΡΟΕΣ ΔΕΔΟΜΕΝΩΝ

Ανακοίνωση: Τρίτη, 5 Μαΐου 2020 **Παράδοση:** Παρασκευή, 5 Ιουνίου 2020

Τελευταία Ενημέρωση: Τρίτη, 26 Μαΐου 2020

1. ΠΕΡΙΓΡΑΦΗ ΕΡΓΑΣΙΑΣ

Συνεχίζουμε και σε αυτή την εργασία το σύστημα αξιολόγησης ταινιών που μας απασχόλησε και στην προηγούμενη εργασία. Αυτή τη φορά μελετάται η αναζήτηση κανόνων συσχέτισης μεταξύ των διαφορετικών ταινιών, της μορφής:

«**ΑΝ** κάποιος βαθμολόγησε υψηλά τις ταινίες του Α **ΤΟΤΕ** πιθανότατα θα ήθελε δει και τις ταινίες του Β».

Προκειμένου να το κάνετε αυτό, θα ασχοληθείτε με τη μέτρηση της **συχνότητας εμφάνισης** (δηλαδή, του στηρίγματος με τη μορφή ποσοστού και όχι απόλυτου αριθμού εμφανίσεων) συνόλων αντικειμένων σε καλάθια.

Συγκεκριμένα, θεωρήστε ως καλάθια τους θεατές (κάθε θεατής είναι κι ένα διαφορετικό **userId**), τα περιεχόμενα των οποίων απαρτίζονται από εκείνες τις ταινίες (κάθε ταινία είναι κι ένα διαφορετικό **movieId**) που ψηφίστηκαν από τον συγκεκριμένο θεατή με βαθμό τουλάχιστον **MinScore**, για κάποια τιμή MinScore $\in$ { 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5 } που παρέχεται από τον χρήστη ως είσοδος.

Θα χρησιμοποιήσετε τον αλγόριθμο APRIORI που θα υλοποιήσετε, αρχικά σε ολόκληρη τη συλλογή καλαθιών και στη συνέχεια σε ένα τυχαίο δείγμα σταθερού μήκους από τη συλλογή θα δημιουργείτε (εκλαμβάνοντας τη συλλογή ως ροή καλαθιών), ώστε να υπολογίσετε σύνολα ταινιών που (πιθανώς) έχουν συχνότητα εμφάνισης ΤΟΥΛΑΧΙΣΤΟΝ **MinFrequency**. Στη συνέχεια, δεδομένων των (πιθανά) συχνών συνόλων αντικειμένων που ανακαλύφθηκαν στη συλλογή καλαθιών, θα δημιουργήσετε για καθένα από αυτά όλους τους κανόνες συσχέτισης που ικανοποιούν κάποια κριτήρια σημαντικότητας (ελάχιστη εμπιστοσύνη – κλιμάκωση) που δίνει ο χρήστης. Τελικά, θα αποτυπώνετε στην οθόνη του χρήστη μια οπτικοποίηση των αποτελεσμάτων σας.

2. ΠΛΑΝΟ ΕΚΠΟΝΗΣΗΣ ΕΡΓΑΣΙΑΣ

Θα πρέπει να κάνετε τα εξής στοιχειώδη βήματα υλοποίησης:

(1) Ρουτίνα **CreateMovieBaskets**: Δημιουργία των καλαθιών με ταινίες, από τα δεδομένα εισόδου (αρχείο ratings.csv, ή αρχείο ratings_100user.csv). Υπενθυμίζεται ότι κάθε γραμμή του αρχείου εισόδου, μετά την πρώτη γραμμή που προσδιορίζει τον μορφότυπο των επόμενων γραμμών, αναπαριστά (διαχωρισμένες με κόμματα) τις τιμές του προσδιοριστικού-χρήστη που παρέχει την αξιολόγηση, του προσδιοριστικού-ταινίας που αξιολογείται, και της χρονοσφραγίδας παροχής της αξιολόγησης. Ο μορφότυπος κάθε γραμμής είναι ο εξής (όπως εξηγείται και στην πρώτη γραμμή του αρχείου):

userId, movieId, rating, timestamp

Οι γραμμές (μετά την πρώτη) είναι ταξινομημένες κατ' αύξουσα τιμή των userId και, για τον ίδιο χρήστη, κατ' αύξουσα τιμή των movieId. Οι αξιολογήσεις είναι τιμές από το σύνολο { 0.5, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5 }. Οι χρονοσφραγίδες (πεδίο timestamp) αναπαριστούν τα δευτερόλεπτα από τα μεσάνυχτα της 1ης Ιανουαρίου 1970 (Coordinated Universal Time -- UTC).

Ως έξοδος της ρουτίνας **CreateMovieBaskets** κατασκευάζεται μια (ανώνυμη) συλλογή των καλαθιών προς επεξεργασία, η οποία είτε αποθηκεύεται σε αρχείο CSV, για μελλοντική χρήση, ή επιστρέφεται ως ένας τύπος δεδομένων list-of-lists με το όνομα **userBaskets**.

ΣΗΜΕΙΩΣΗ: Στα αρχεία με τις βαθμολογίες, οι αξιολογήσεις είναι ταξινομημένες αλφαβητικά, πρώτα ανά χρήστη και κατόπιν ανά ταινία. Έτσι, για κάθε χρήστη, οι ταινίες που έχουν βαθμολογηθεί από αυτόν εμφανίζονται κατά αύξουσα σειρά. Είναι πολύ χρήσιμο να ΔΙΑΤΗΡΗΣΕΤΕ αυτή την ταξινόμηση των ταινιών και σε κάθε καλάθι που δημιουργείτε. Κατόπιν, όταν θα μελετάτε ζεύγη, τριάδες, κ.λπ., φροντίστε πάντοτε τα στοιχεία της πλειάδας να είναι επίσης ταξινομημένα σε αύξουσα σειρά.

(2) Ρουτίνα **ReadMovies**: Δημιουργία ενός DataFrame με ταινίες από τα δεδομένα εισόδου (αρχείο movies.csv). Υπενθυμίζεται ότι κάθε γραμμή του αρχείου εισόδου, μετά την πρώτη γραμμή που προσδιορίζει τον μορφότυπο των επόμενων γραμμών, αναπαριστά (διαχωρισμένες με κόμματα) τις τιμές του προσδιοριστικού-ταινίας, τον τίτλο της και τις κατηγορίες στις οποίες εμπίπτει. Ο μορφότυπος κάθε γραμμής είναι ο εξής (όπως εξηγείται και στην πρώτη γραμμή του αρχείου):

movieId, Title, (bar-separated) Genres

Επιστρέφεται η μεταβλητή movies_df, που είναι τύπου δεδομένων pandas DataFrame, με τα εξής ονόματα στηλών:

- 'movieId': θετικός ακέραιος (αναγνωριστικό ταινίας_
- 'title': αλφαριθμητικό (τίτλος ταινίας)
- 'genres': αλφαριθμητικό (κατηγορίες ταινίας)

Η συγκεκριμένη ρουτίνα δίνει τη δυνατότητα παροχής λεπτομερειών για μια ταινία της οποίας γνωρίζουμε μόνο το αναγνωριστικό.

(3) Δημιουργία δυο διαφορετικών ρουτινών, για την επακριβή μέτρηση όλων των **ζευγών** αντικειμένων, εντός της κύριας μνήμης.

a. Η πρώτη ρουτίνα (έστω **TriangularMatrixOfPairsCounters**) θα οργανώνει σε ένα κάτω-τριγωνικό μητρώο ΟΛΟΥΣ τους K*(K-1)/2 μετρητές εμφανίσεων **για ζεύγη** μεταξύ των K αντικειμένων (K = όλες οι αξιολογημένες οι ταινίες από τουλάχιστον έναν χρήστη) στα N καλάθια (ένα καλάθι για κάθε χρήστη).
b. Η δεύτερη ρουτίνα (έστω **HashedCountersOfPairs**) θα οργανώνει σε έναν πίνακα κατακερματισμού (hash table) μετρητές για ζεύγη αντικειμένων (οι ταινίες) στα N καλάθια (ένα καλάθι για κάθε χρήστη). Η συγκεκριμένη ρουτίνα θα αποθηκεύει, για ζεύγος αντικειμένων (i,j) που εμφανίζεται σε καλάθι τουλάχιστον μία φορά, την τριάδα (i,j,c), ως ζεύγος κλειδιού-τιμής (hash(i,j) , c) με κλειδί το hash(i,j) και τιμή το c που είναι το πλήθος εμφανίσεων του (i,j) στη συλλογή των καλαθιών μας.

(4) Ρουτίνα **myApriori**: Υλοποίηση του αλγορίθμου APRIORI, οποίος θα μετρά διαδοχικά **συχνότητες εμφάνισης** (frequencies) αντικειμένων, ζευγών αντικειμένων, τριάδων αντικειμένων, κ.ο.κ., που ξεπερνούν ένα συγκεκριμένο **κατώφλι (συχνοτήτων) εμφάνισης** min_frequency$\in$(0,1), με τυπική τιμή

min_frequency = 0.1. Επίσης, αναζητάμε συχνά σύνολα που περιλαμβάνουν ΤΟ ΠΟΛΥ max_length ≥ 2 ταινίες. Με άλλα λόγια, ο αλγόριθμος δέχεται ως είσοδο τα εξής πεδία: (itemBaskets, min_frequency, max_length). Η έξοδος του αλγορίθμου είναι μια λίστα λιστών frequent_itemsets, μια λίστα frequent_itemsets[0] = L_1 με συχνά εμφανιζόμενα μονοσύνολα, μια λίστα frequent_itemsets[1] = L_2 με τα συχνά δισύνολα, μια λίστα frequent_itemsets[2] = L_3 με συχνά τρισύνολα, …, έως και μια λίστα frequent_itemsets[K-1] = L_K με τις συχνές Κ-άδες ταινιών, όπου είτε Κ = max_length, ή είμαστε πλέον βέβαιοι ότι (αν και K<maxlength) δεν υπάρχουν μεγαλύτερα συχνά σύνολα ταινιών.

ΠΡΟΣΟΧΗ: Για οποιοδήποτε σύνολο αντικειμένων A, frequency(A) = support(A) / N είναι η **συχνότητα εμφάνισης** όλων των αντικειμένων του A στα καλάθια, όπου support(A) είναι το **πλήθος** των καλαθιών που περιέχουν όλες τις ταινίες από το A και N είναι το πλήθος των καλαθιών στη συλλογή μας.

Το πλήθος περασμάτων που θα κάνει ο αλγόριθμος στη συλλογή των καλαθιών itemBaskets θα πρέπει να είναι σε άμεση συνάρτηση με τον μέγιστο πληθάριθμο ενός συχνού συνόλου που υπάρχει στη συλλογή και δεν ξεπερνά την παράμετρο ελέγχου max_length. Πχ, αν στη συλλογή υπάρχουν μέχρι και συχνές πεντάδες αλλά είμαστε σίγουροι πως δεν υπάρχουν συχνές εξάδες, ή αν ισχύει ότι max_length ≤ 5, τότε θα πρέπει να γίνουν από τον αλγόριθμο APRIORI το πολύ 5 περάσματα της συλλογής.

(5) Ρουτίνα **sampledApriori**: Πρόκειται για εφαρμογή του αλγορίθμου APRIORI, όχι απαραίτητα σε ολόκληρη τη συλλογή αξιολογήσεων, αλλά σε ένα αρχικό τμήμα της (εφόσον δοθεί από τον χρήστη σήμα διακοπής της ροής, πχ πατώντας κάποιο πλήκτρο), διατηρώντας συνεχώς ένα **σταθερού μήκους δείγμα καλαθιών**. Συγκεκριμένα, αποθηκεύουμε σε μια μεταβλητή ratings_stream τύπου PANDAS dataframe, την επιστροφή της ρουτίνας ανάγνωσης του CSV αρχείου αξιολογήσεων. Στη συνέχεια, χειριζόμαστε το ratings_stream ως μία ροή αξιολογήσεων (κάθε γραμμή του είναι και μια νέα αξιολόγηση). Ένας επαναληπτικός βρόχος σάρωσης των γραμμών του ratings_stream προσομοιώνει την ανάγνωση, μία προς μία, των αξιολογήσεων στη ροή. Η ανάγνωση αυτή συνεχίζεται έως ότου πατηθεί ένα συγκεκριμένο πλήκτρο διακοπής, ή ολοκληρωθεί η ροή των αξιολογήσεων. Δείτε ένα απλό παράδειγμα βρόχου που τερματίζεται πρόωρα (με το πάτημα ενός συγκεκριμένου πλήκτρου) στο παράδειγμα κώδικα repeat_until_keystroke.py.

Το ζητούμενο είναι να διατηρούμε, ανά πάσα στιγμή, ένα τυχαία και ομοιόμορφα επιλεγμένο σταθερό υποσύνολο καλαθιών (π.χ. 50 για το μικρό σύνολο αξιολογήσεων με τους 100 χρήστες, ή 100 για το μεγάλο σύνολο αξιολογήσεων με τους 610 χρήστες, αλλά αυτό θα είναι παράμετρος της εισόδου), μεταξύ των καλαθιών (δηλαδή, των χρηστών) που έχουν εμφανιστεί μέχρι τώρα (από ένα άγνωστο συνολικό πλήθος χρηστών). Αυτό γίνεται ως εξής:

a. Προκειμένου να γνωρίζουμε τους χρήστες που έχουμε συνολικά δει μέχρι στιγμής, συντηρούμε ένα σύνολο SetOfUsers από userId-τιμές για τις αξιολογήσεις της ροής που έχουμε επεξεργαστεί.

b. Για κάθε νέα αξιολόγηση της ροής, έστω current_assessment, κάνουμε το εξής:

```
current_user = current_assessment['userId']

current_movie = current_assessment['movieId']

if current_user not in SetOfUsers:              # new user appeared…

        SetOfUsers.add(current_user)

        SampleOfBaskets = ReservoirSampling(NumbeOfDistinctUsersSoFar,SampleOfBaskets,current_user)
```

```
if current_user in SampleOfBaskets:
        # SampleOfBaskets = Dictionary with (userId, movieId-frozenset)…
        SampleOfBaskets[current_user].union({current_movie})
```

c. Επαναλαμβάνουμε την παραπάνω διαδικασία επεξεργασίας των αξιολογήσεων της ροής, μέχρις ότου στο πληκτρολόγιο πατηθεί το προεπιλεγμένο πλήκτρο διακοπής ανάγνωσης της ροής, ή μέχρι να ολοκληρωθεί η ροή.

d. Αφού θα έχει ολοκληρωθεί (ή διακοπεί πρόωρα) η σάρωση της ροής αξιολογήσεων, εκτελείται ο αλγόριθμος Apriori στα καλάθια του δείγματος. Επιστρέφονται τα σύνολα ταινιών που έχουν συχνότητα τουλάχιστον ίση με το κατώφλι συχνότητας που δίνεται ως είσοδος.

Για την αξιολόγηση της ρουτίνας **sampledApriori**, να γίνει χρήση των «ανακατεμένων» συλλογών αξιολογήσεων που δίνονται στα εξής αρχεία:

- Αξιολογήσεις που παρέχονται από τους 100 πρώτους χρήστες, αλλά με τυχαία σειρά αξιολογήσεων: http://www.cse.uoi.gr/~kontog/courses/Algorithms-For-Big-Data/locked-stuff/assignments/lab-2/datasets/ratings_100users_shuffled.csv

- Αξιολογήσεις που παρέχονται από 610 χρήστες, αλλά με τυχαία σειρά αξιολογήσεων: http://www.cse.uoi.gr/~kontog/courses/Algorithms-For-Big-Data/locked-stuff/assignments/lab-2/datasets/ratings.csv

ΠΑΡΑΤΗΡΗΣΗ 1: Κάθε φορά που αποθηκεύετε ένα συγκεκριμένο υποσύνολο ταινιών, προτείνεται να είναι τύπου frozenset(), ώστε να μπορεί να δίνεται ως κλειδί σε ένα λεξικό (για παράδειγμα, με τιμές τις συχνότητες των συνόλων ταινιών που θεωρείτε συχνά) δίχως να παίζει ρόλο η σειρά εμφάνισης των στοιχείων του.

ΠΑΡΑΤΗΡΗΣΗ 2: Αν αφήνουμε να ολοκληρωθεί η ανάγνωση της ροής, θα έχουμε κρατήσει στο δείγμα μας το 10% των χρηστών (και ολόκληρων των καλαθιών τους). Έτσι, ένα συγκεκριμένο σύνολο ταινιών Α που εμφανίζεται συχνό (δηλαδή, ξεπερνά τω κατώφλι συχνότητας minfrequency που δίνεται ως είσοδος), αναμένεται (αλλά δεν είναι βέβαια σίγουρο) να είναι το ίδιο συχνό (άρα, να ξεπερνά το κατώφλι συχνότητας minfrequency) και στο δείγμα. Φυσικά, κάποια σύνολα θα συμβεί να έχουν μικρότερη συχνότητα στο δείγμα, κάποια άλλα θα έχουν μεγαλύτερη συχνότητα στο δείγμα. Ένα δεύτερο (πλήρες) πέρασμα της ροής για να μετρηθούν οι πραγματικές συχνότητες των συνόλων που εμφανίστηκαν ως συχνά στο δείγμα, μπορεί να εξαλείψει τα FALSE-POSITIVES.

ΠΑΡΑΤΗΡΗΣΗ 3: Όταν διακόπτεται πρόωρα η ανάγνωση της ροής, τότε τα καλάθια όλων των χρηστών (μεταξύ αυτών και εκείνα που διατηρούνται στο δείγμα τη στιγμή της διακοπής) είναι μικρότερα (γιατί οι αξιολογήσεις εμφανίζονται στη ροή «ανακατεμένες». Αυτό σημαίνει ότι όλες οι συχνότητες συνόλων-ταινιών στο δείγμα θα είναι μικρότερες σε σχέση με τις συχνότητές τους σε ολόκληρη τη ροή, αλλά όχι απαραίτητα αναλόγως μικρότερες. Όσο πιο κοντά φτάνει η διαδικασία ανάγνωσης της ροής, τόσο πιο κοντά στις πραγματικές συχνότητες θα είναι οι συχνότητες που μετρώνται εντός του δείγματος.

(6) Ρουτίνα **AssociationRulesCreation**: Η συγκεκριμένη ρουτίνα θα δέχεται ως είσοδο μια συλλογή των (πιθανά) συχνών συνόλων ταινιών (δηλαδή, την έξοδο είτε του βήματος 3, ή του βήματος 4), καθώς και τις εξής παραμέτρους για την αξιολόγηση των παραγόμενων κανόνων:

- **Κατώφλι εμπιστοσύνης**: min_confidence ∈ (0,1). Τυπική τιμή: min_confidence = 0.5. Υπενθυμίζεται ότι μεταξύ ξένων υποσυνόλων αντικειμένων A και B, η εμπιστοσύνη του κανόνα A –> B ορίζεται ως εξής: Confidence(A–>B) = frequency(A U B) / frequency(A) = Pr(B|A).

- **Κατώφλι κλιμάκωσης**: MinLift > 1 (ή, τιμή MinLift = -1 που δηλώνει ότι πρέπει να αγνοηθεί η συγκεκριμένη παράμετρος). Η συγκεκριμένη παράμετρος εστιάζει το ενδιαφέρον μόνο σε κανόνες της μορφής A–>B που υποδηλώνουν θετική συσχέτιση, και έχουν πολλαπλασιαστική κλιμάκωση κατά MinLift της πιθανότητας Pr(B|A) σε σχέση με το Pr(B): **Pr(B|A) / Pr(B) > MinLift > 1**.

- **Ανώφλι αποκλιμάκωσης**: 1 > MaxLift > 0 (ή, τιμή MaxLift = -1, που δηλώνει ότι πρέπει να αγνοηθεί η συγκεκριμένη παράμετρος). Η συγκεκριμένη παράμετρος εστιάζει το ενδιαφέρον μόνο σε κανόνες της μορφής A–>B που υποδηλώνουν αρνητική συσχέτιση του B από το A, και έχουν πολλαπλασιαστική αποκλιμάκωση κατά MaxLift της πιθανότητας Pr(B|A) σε σχέση με το Pr(B): **Pr(B|A) / Pr(B) < MaxLift < 1**.

- Για τον κανόνα A–>B μεταξύ ξένων υποσυνόλων αντικειμένων A και B, η **κλιμάκωση** Lift(A–>B) είναι ο λόγος μεταβολής της πιθανότητας Pr(B|A) για εμφάνιση του συνόλου αντικειμένων B, όταν (ξέρουμε ότι) εμφανίστηκε το σύνολο αντικειμένων A $\subseteq$ Ω – B (Ω είναι το σύμπαν όλων των αντικειμένων), ως προς την πιθανότητα Pr(B) να εμφανιστεί στα καλάθια το B ανεξαρτήτως του A. Η κλιμάκωση του κανόνα υπολογίζεται λοιπόν ως εξής:

  Lift(A–>B) = Confidence(A–>B) / frequency(B)
  = frequency(A U B) / [frequency(A) * frequency(B)]

Σημειώνεται ότι ενδέχεται να υπάρχουν κλιμακώσεις μεγαλύτερες της μονάδας, δηλώνοντας **θετική συσχέτιση** του B ως προς το A, αλλά και μικρότερες της μονάδας, υπονοώντας **αρνητική συσχέτιση** του B ως προς το A.

Αν για παράδειγμα δώσουμε MinConfidence = 0.5 και MinLift = 2 και MaxLift = -1, τότε μας ενδιαφέρει να βρούμε κανόνες A–>B με εμπιστοσύνη τουλάχιστον 0.5, που εξασφαλίζουν τουλάχιστον διπλασιασμό της πεποίθησης Pr(B|A) ότι θα εμφανιστεί στο τρέχον καλάθι το B επειδή ξέρουμε ότι εμφανίστηκε το A, σε σχέση με την πρότερη πεποίθηση Pr(B).

Η έξοδος του αλγορίθμου είναι μια μεταβλητή rules_df, τύπου DataFrame, με τα εξής πεδία (ονόματα στηλών): 'itemset','rule', 'hypothesis', 'conclusion','frequency','confidence','lift', 'interest', 'rule ID'.

Παράδειγμα εξόδου που θα πρέπει να επιστρέφει ο αλγόριθμος Association_Rules_Creation:

```
>>> rules_df.head()
        itemset hypothesis conclusion  ...      lift  interest  rule ID
0    [161, 349]      [161]      [349]  ...  4.262620  0.545553        1
1    [349, 454]      [349]      [454]  ...  4.225277  0.486435        2
2  [2683, 1517]     [1517]     [2683]  ...  5.116342  0.593833        3
3  [1968, 2918]     [1968]     [2918]  ...  4.005860  0.497692        4
4  [4896, 5816]     [4896]     [5816]  ...  4.966475  0.565710        5

[5 rows x 9 columns]
>>> rules_df.iloc[0]
itemset             [161, 349]
hypothesis               [161]
conclusion               [349]
rule           [161] --> [349]
frequency             0.109836
confidence            0.712766
lift                   4.26262
interest              0.545553
rule ID                      1
Name: 0, dtype: object
```

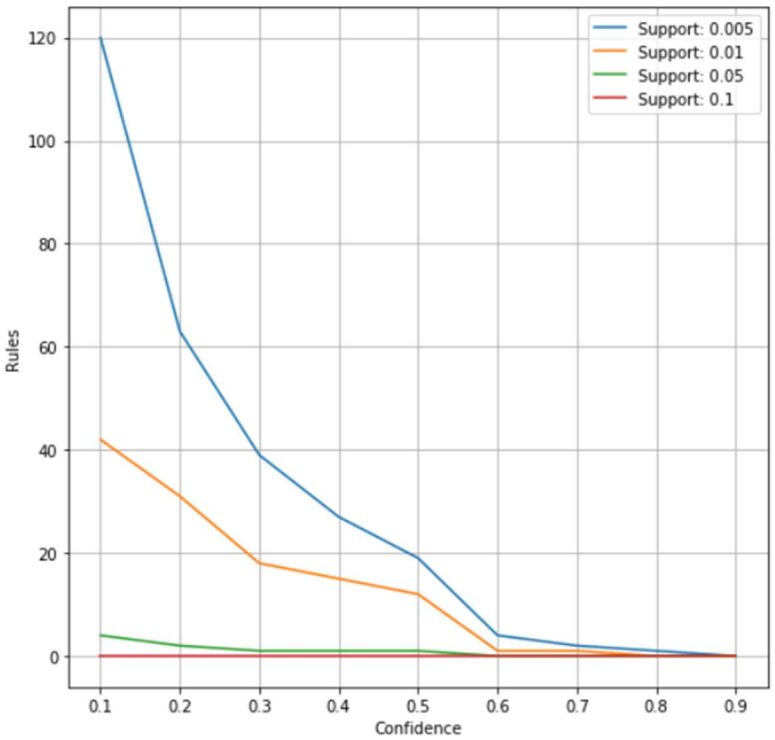
Για την αναπαράσταση συνόλων ταινιών (όπως στις στήλες itemset, hypothesis, conclusion) μπορείτε να κάνετε χρήση είτε (διατεταγμένων) λιστών ή «παγωμένων συνόλων» (frozensets). Το πεδίο rule είναι αλφαριθμητικό, ενώ τα υπόλοιπα πεδία είναι float αριθμοί.

(7) Ρουτίνα **presentResults**: Δέχεται ως είσοδο το rules_df που παράγεται από τους αλγορίθμους συσχέτισης, και δίνει τις εξής δυνατότητες παρουσίασης:

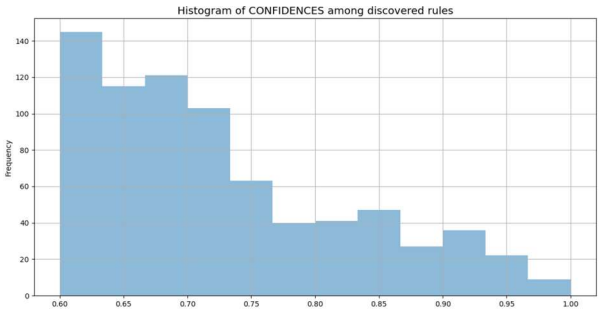
```
================================================================================
(a)   List ALL discovered rules                          [format: a]
(b)   List all rules containing a BAG of movies          [format:
in their <ITEMSET|HYPOTHESIS|CONCLUSION>                  b,<i,h,c>,<comma-sep. movie IDs>]
(c)   COMPARE rules with <CONFIDENCE,LIFT>               [format: c]
(h)   Print the HISTOGRAM of <CONFIDENCE|LIFT >          [format: h,<c,l >]
(m)   Show details of a MOVIE                            [format: m,<movie ID>]
(r)   Show a particular RULE                             [format: r,<rule ID>]
(s)   SORT rules by increasing <CONFIDENCE|LIFT >        [format: s,<c,l >]
(v)   VISUALIZATION of association rules                 [format: v,<draw_choice:
      (sorted by lift)                                     [c(ircular),r(andom),s(pring)]>,
                                                           <num of rules to show>]
(e)   EXIT                                               [format: e]
================================================================================
```

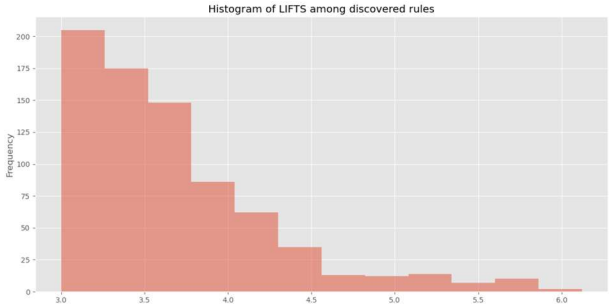
Σχετικά με την παρουσίαση των στατιστικών στοιχείων για τους κανόνες, βλ. περιπτώσεις (c) και (h), αναμένεται να γίνουν τα εξής:

- Αποτύπωση κανόνων ως προς εμπιστοσύνη – κλιμάκωση. Για παράδειγμα, να γίνει σύγκριση της εμπιστοσύνης (και της κλιμάκωσης) ως συνάρτηση του ελάχιστου μήκους (=πληθάριθμος itemset), όπως στο ακόλουθο σχήμα:

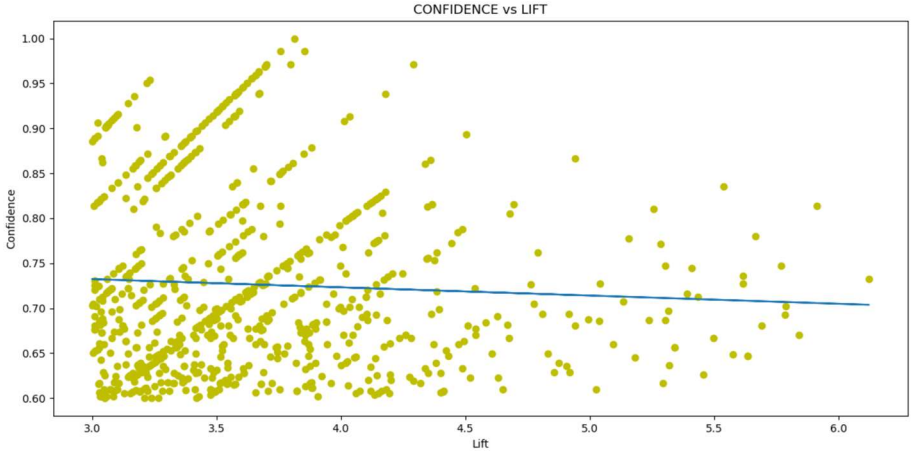


- Δημιουργία ιστογραμμάτων για εμπιστοσύνη – κλιμάκωση, ως συνάρτηση των στηριγμάτων των συνόλων. Για παράδειγμα:

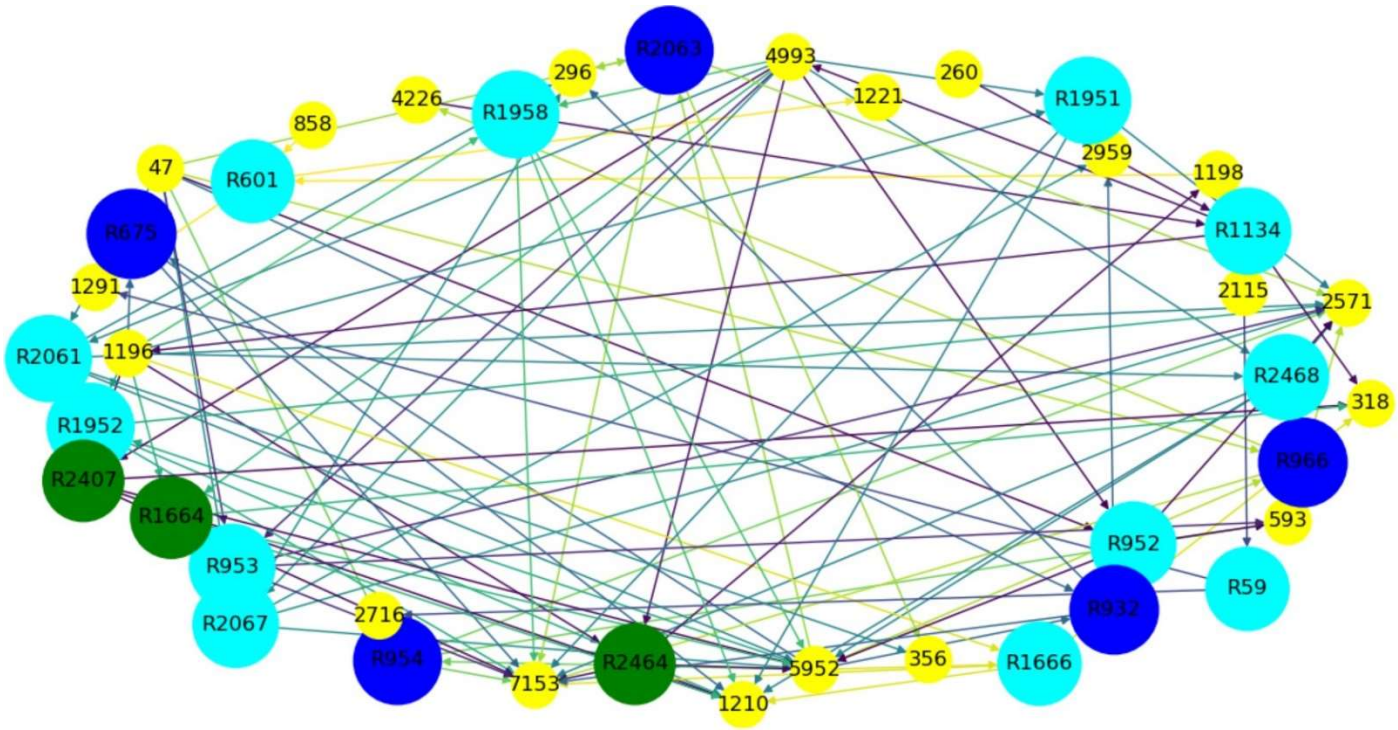




- Συσχέτιση εμπιστοσύνης – κλιμάκωσης:



Οπτικοποίηση αποτελεσμάτων (περίπτωση v): Δημιουργία και αναπαράσταση γραφήματος για ένα μικρό πλήθος (πχ, τους 10) κορυφαίους κανόνες ως προς τη μετρική της κλιμάκωσης (lift): Στο γράφημα που θα δημιουργήσετε, λαμβάνονται υπόψη όλες οι εμπλεκόμενες ταινίες σε αυτούς τους κορυφαίους κανόνες. Οι κίτρινοι κόμβοι αναπαριστούν ταινίες, οι υπόλοιποι κόμβοι αναπαριστούν κανόνες. Το μέγεθος κάθε κόμβου-κανόνα σχετίζεται με την τιμή της κλιμάκωσης, ενώ το χρώμα του σχετίζεται με την τιμή της εμπιστοσύνης του. Τα τόξα υποδεικνύουν τη σχέση μεταξύ των ταινιών στους κανόνες. Για παράδειγμα, ο κανόνας R: {A,B} --> {C,D} θα δημιουργήσει έναν κόμβο R, στον οποίο εισέρχονται δυο τόξα από τους (κίτρινους) κόμβους A και B, ενώ εξέρχονται και δυο τόξα προς τους (κίτρινους) κόμβους C,D. Μπορείτε να αξιοποιήσετε τη ρουτίνα αναπαράστασης draw_rules_graph.py που θα βρείτε στο συνοδευτικό υλικό της 2ης ανάθεσης (δείτε το ecourse του μαθήματος), που αξιοποιεί τη βιβλιοθήκη networkx της python. Η συγκεκριμένη ρουτίνα δέχεται ως είσοδο ένα dataframe με τους προς οπτικοποίηση κανόνες, και παράγει στην έξοδο το κατάλληλο γράφημα, όπως αυτό που φαίνεται στο ακόλουθο σχήμα:



3. ΠΕΙΡΑΜΑΤΙΚΗ ΑΞΙΟΛΟΓΗΣΗ

Αρχικά μπορείτε να χρησιμοποιήσετε για την πειραματική αξιολόγηση των προγραμμάτων σας το εξής αρχείο:

- http://www.cse.uoi.gr/~kontog/courses/Algorithms-For-Big-Data/locked-stuff/assignments/lab-1/datasets/ratings_100users.csv

που περιλαμβάνει τις αξιολογήσεις ταινιών από 100 διαφορετικούς χρήστες. Ειδικά για την sampledApriori, θα πρέπει να κάνετε χρήση της «ανακατεμένης» συλλογής των αξιολογήσεων από τους 100 πρώτους χρήστες, που παρέχεται στο εξής αρχείο:

- http://www.cse.uoi.gr/~kontog/courses/Algorithms-For-Big-Data/locked-stuff/assignments/lab-2/datasets/ratings_100users_shuffled.csv

Για την τελική αναφορά σας όμως, θα πρέπει να χρησιμοποιήσετε ως είσοδο για τον myApriori το αρχείο:

- http://www.cse.uoi.gr/~kontog/courses/Algorithms-For-Big-Data/locked-stuff/assignments/lab-1/datasets/ratings.csv

και για τον sampledApriori το αρχείο:

- http://www.cse.uoi.gr/~kontog/courses/Algorithms-For-Big-Data/locked-stuff/assignments/lab-2/datasets/ratings_shuffled.csv

που περιλαμβάνουν τις αξιολογήσεις ταινιών από 610 διαφορετικούς χρήστες.

Το ακόλουθο αρχείο:

- http://www.cse.uoi.gr/~kontog/courses/Algorithms-For-Big-Data/locked-stuff/assignments/lab-2/datasets/movies.csv

περιλαμβάνει πληροφορίες για όλες τις ταινίες που εμπλέκονται στα αρχεία αξιολόγησης.

Για τον αλγόριθμο APRIORI θα πρέπει να δημιουργήσετε όλα τα συχνά σύνολα αντικειμένων. Για τον αλγόριθμο SampledAPRIORI θα πρέπει να εκτιμήσετε την ακρίβεια της προσέγγισης (ως προς το ground-truth που παράγει ο APRIORI) σύμφωνα με τις μετρικές που χρησιμοποιήθηκαν και στην πρώτη εργασία σας (precision,recall,f1-score).

Σε κάθε στατιστική επεξεργασία των αποτελεσμάτων σας θα πρέπει να επιχειρήσετε να δώσετε κάποια εκτενή εξήγηση των μετρήσεών σας.

4. ΠΑΡΑΔΟΣΗ ΕΡΓΑΣΙΑΣ

Θα πρέπει να αναρτήσετε στο eCourse, το αργότερα μέχρι την **Παρασκευή 05/06/2020**, ένα ZIP αρχείο με όνομα της μορφής **2019-20_CSE-UOI_MYE047-ABD_< EPWNYMO >-< ONOMA >_< AM >_ASSIGNMENT-2.ZIP**, το οποίο περιλαμβάνει τα εξής:

(ι) Φάκελο **SOURCES** με όλα τα προγράμματά σας σε Python.

(ιι) Φάκελο **EXPERIMENTS** με τα μητρώα και λεξικά που παράγετε, και snapshots από κάθε παράδειγμα εκτέλεσης.

(ιιι) Αναλυτική αναφορά (σε μορφή MS WORD ή LaTeX), η οποία θα περιγράφει την υλοποίησή σας, και θα παρουσιάζει τα αποτελέσματα (και συνοπτική ερμηνεία τους) των παραδειγμάτων εκτέλεσης και της πειραματικής αξιολόγησης των προγραμμάτων σας.