ΠΑΝΤΕΛΙΔΗΣ ΝΙΚΟΣ
ΑΜ 2787

ΑΛΓΟΡΙΘΜΟΙ ΓΙΑ ΔΕΔΟΜΕΝΑ ΕΥΡΟΙΑΣ ΚΛΙΜΑΚΑΣ

2η ΑΝΑΘΕΣΗ ΓΙΑ ΤΟ ΣΠΙΤΙ

Τεχνικές Μέτρησης Ζευγαριών

Κάτω Τριγωνικό Μητρώο Μέτρησης Ζευγαριών

Η υλοποίηση του κάτω τριγωνικού μητρώου μέτρησης ζευγαριών βρίσκεται στο αρχείο pair_finder.py, στην ρουτίνα TriangularMatrixOfPairCounters(line: 17). Αρχικά φτιάχνω μία λίστα μεγέθους n*(n-1)/2 (n=πλήθος ταινιών της συλλογή) η οποία είναι μία γραμμική αναπαράσταση του κάτω τριγωνικού πίνακα. Στην συνέχεια διατρέχω τα καλάθια των χρηστών και δημιουργώ μία λίστα με όλα τα πιθανά ζευγάρια που μπορούν να παραχθούν για το συγκεκριμένο καλάθι. Διατρέχω την κάθε λίστα με τα ζεύγη και υπολογίζω τα indexes της κάθε ταινίας με βάση το movieID (Πχ η λίστα με τα ζευγάρια για ένα καλάθι 'χ' είναι [[3,2], [6,5]]. Οι δύο τιμές του κάθε ζευγαριού είναι movieIDs εμείς όμως για να μπορέσουμε να βρούμε την θέση που αντιστοιχεί στον μητρώο χρειαζόμαστε το index της κάθε ταινίας) και τα εισάγω στον τύπο που διδαχθήκαμε στο μάθημα για να βρω την θέση του μετρητή του ζευγαριού στο κάτω τριγωνικό μητρώο.

Λεξικό Μέτρησης Ζευγαριών

Η υλοποίηση του λεξικού μέτρησης ζευγαριών βρίσκεται στο αρχείο pair_finder.py, στην ρουτίνα HashCountersOfPairs(line: 50). Η ρουτίνα δουλεύει ως εξής. Διατρέχω ένα ένα τα καλάθια των χρηστών. Για κάθε καλάθι δημιουργώ όλα τα πιθανά ζευγάρια και τα επιστρέφω ως λίστα. Στην συνέχει διατρέχω την λίστα με τα ζευγάρια και ελέγχω αν υπάρχει ήδη το ζευγάρι στο λεξικό. Αν υπάρχει απλά αυξάνω τον μετρητή κατά ένα, αλλιώς αρχικοποιώ νέο μετρητή στο λεξικό με κλειδί το τρέχον ζευγάρι.

Σύγκριση Τεχνικών

Στο αρχείο hashed_vs_triangular.py έχω φτιάξει ένα ντέμο των δύο τεχνικών για τις συγκρίνω. Είναι ξεκάθαρο από τα αποτελέσματα που φαίνονται παρακάτω ότι το λεξικό είναι πιο αποδοτικό.

```
TRIANGULAR MATRIX PAIR GENERATOR ---> 719.8360168933868 sec
TM PAIRS: 601223
HASH PAIR GENERATOR ---> 2.9459574222564697 sec
HASH PAIRS: 601223
```

**Η σύγκριση έγινε για κριτικές με MinScore = 4 στο αρχείο με τους 100 χρήστες.

Ο τριγωνικός πίνακας είναι πολύ αργός για τους παρακάτω λόγους:

1. Ο υπολογισμός της θέσης του ζευγαριού στη λίστα σειριακή αναπαράστασης του κάτω τριγωνικού πίνακα περιέχει ακριβές πράξεις (πολλαπλασιασμός, διαίρεση).
2. Πρέπει να υπολογίζουμε κάθε φορά τις θέσεις των δύο ταινιών στο movie dataframe. (Αυτή καθυστέρηση θα μπορούσε να αντιμετωπιστεί αν περνούσαμε τα δεδομένα μας από προεργασία ώστε το movieId να είναι ένας αύξων αριθμός από το 0 έως το v-1.)
3. Ο πίνακας δεσμεύει χώρο ακόμα και για τα ζευγάρια που δεν συναντάμε.

Καταλήγουμε λοιπόν ότι ο πίνακας κατακερματισμού (λεξικό) είναι σημαντικά πιο αποδοτικός από άποψη χρόνου αλλά και από άποψη χώρου.

Τεχνικές Εύρεσης Κανόνων Συσχέτισης

Αλγόριθμος A Priori (Χωρίς δειγματοληψία)

Στην υλοποίηση μου(pair_finder.py l:63 -> myApriori) ο a priori υπολογίζει τα συχνά bags of movies τα οποία στην συνέχει θα μπουν ως είσοδο στην ρουτίνα που υπολογίζει τους κανόνες συσχέτισης. Ο mApriori κάνει χρήση τριών βοηθητικών συναρτήσεων:

1. Calculate_frequencies (line: 89)

   Αυτή η συνάρτηση παίρνει ως είσοδο μία λίστα με λίστες με frozenset και έναν ακέραιο αριθμό N. Πιο συγκεκριμένα το πρώτο όρισμα είναι μία λίστα με μία λίστα για κάθε χρήστη η οποία περιέχει όλους τους πιθανούς συνδυασμούς bags of movies για τον συγκεκριμένο χρήστη σε μορφή frozensets. Η ρουτίνα υπολογίζει τις συχνότητες κάθε bag of movies που συναντάμε, τα αποθηκεύει σε ένα dictionary με κλειδιά τα bag of movies και το επιστρέφει.

2. Frequency_filter(line: 111)

   Αυτή η συνάρτηση παίρνει ως είσοδο μία τιμή min_frequency που θα χρησιμοποιηθεί ως κάτω φράγμα συχνοτήτων για τα bag of movies και ένα λεξικό με τα bag of movies και τις συχνότητες τους. Η συνάρτηση φιλτράρει το λεξικό από τους συνδυασμούς με συχνότητα χαμηλότερη από min_frequency.

3. Get_combos(line:118)

   Η συνάρτηση παίρνει ως είσοδο το τα καλάθια των χρηστών, τους έως τώρα συνδυασμούς και έναν ακέραιο που συμβολίζει το μέγεθος των συνδυασμών που θα παράγει η συνάρτηση. Ο αλγόριθμος δουλεύει ως εξής:

   - Διατρέχει τα καλάθια των χρηστών
   - Ελέγχουμε αν το καλάθι έχει μέγεθος τουλάχιστον k.(Ειδάλλως δεν μπορεί να παράγει συνδυασμούς μεγέθους k)
   - Για κάθε καλάθι διατρέχουμε τα στοιχεία του.

- Ελέγχουμε αν η τρέχουσα ταινία είναι συχνή μέσω του λεξικού που κρατάει τις συχνές ταινίες και συνδυασμούς.
- Διατρέχουμε όλους τους συνδυασμούς μήκους k-1.
- Ελέγχουμε αν ο τρέχον συνδυασμός μήκους k-1 παράγεται από το τρέχον καλάθι και αν η τρέχουσα ταινία δεν βρίσκεται στον συνδυασμό.
- Αν ισχύει η παραπάνω συνθήκη σχηματίζω την ένωση της τρέχουσας ταινίας, τους συνδυασμού και την εισάγω στην λίστα επιστροφής.

Η συνάρτηση επιστρέφει μία λίστα με μία λίστα για κάθε χρήστη η οποία περιέχει τους παραχθέντες συνδυασμούς σε μορφή frozenset.

Ο a apriori δουλεύει ως εξής. Σχηματίζει τους συνδυασμούς για το κάθε καλάθι (για τα μονοσύνολα παραλείπεται αυτό το βήμα, για τα ζευγάρια χρησιμοποιείται η ρουτίνα get_pairs, ενώ για όλες τις υπόλοιπες περιπτώσεις γίνεται χρήση της ρουτίνας get_combos), μετράει τις συχνότητες των συνδυασμών από τα καλάθια (calculate_frequencies) και φιλτράρει τους συνδυασμούς που δεν περνούν το φράγμα συχνοτήτων min_frequency (frequency filter). Αυτά τα βήματα επαναλαμβάνονται μέχρι max_length. Η ρουτίνα επιστρέφει ένα λεξικό με κλειδιά το μήκος των συνδυασμών και τιμές λεξικά με κλειδί τους συνδυασμούς του συγκεκριμένου μήκους και τιμή την συχνότητα του συνδυασμού.

Αλγόριθμος A Priori (Με δειγματοληψία)

Ο αλγόριθμος sampledApriori (pair_finder.py line: 144) χρησιμοποιεί δύο βοηθητικές ρουτίνες:

1. Reservoir_sampling (pair_finder.py line: 197)

   Αυτή η ρουτίνα ουσιαστικά υλοποιεί το αλγόριθμο reservoir sampling και επιλέγει αν θα μπει στο δείγμα η συγκεκριμένη κριτική.

2. Run_apriori (pair_finder.py line: 165)

   Αυτή η ρουτίνα διατρέχει το stream με τις κριτικές και καλεί για κάθε κριτική την ρουτίνα reservoir_sampling. Επιπλέον μας δίνει την δυνατότητα να διακόψουμε την δειγματοληψία πατώντας ‘Y’ ή ‘y’ αν έχουμε μαζέψει τουλάχιστον sample_size δείγμα. Μόλις τελειώσει η δειγματοληψία είτε από τον χρήστη είτε επειδή τελείωσε το stream καλεί την ρουτίνα myApriori.

Η ρουτίνα sampledApriori αρχικά τρέχει τον run_apriori με ενεργοποιημένο το κουμπί για διακοπή της δειγματοληψίας. Αν ο χρήστης διακόψει την δειγματοληψία επιστρέφεται το αποτέλεσμα τους run_apriori. Αν τελειώσει η ροή τότε ξανακαλείται η ρουτίνα run apriori με απενεργοποιημένο το κουμπί διακοπής. Χρησιμοποιούμε το αποτέλεσμα του δεύτερου περάσματος για να διαγράψουμε τα false positives από το αποτέλεσμα του πρώτου περάσματος. Η διαδικασία αυτή γίνεται διαγράφοντας τις εγγραφές του πρώτου περάσματος που δεν βρίσκονται στο αποτέλεσμα του δεύτερου περάσματος.

Τέλος επιστρέφει τελικά ότι έμεινε από το πρώτο αποτέλεσμα.

Απόδοση A Priori με δειγματοληψία

Η σύγκριση των δύο μεθόδων γίνεται στο αρχείο sampled_apriori_performance.py. Εδώ έχω φτιάξει ένα ντέμο που χρονομετρώ τις δύο μεθόδους και υπολογίζω precision, recall και f1-score για τον sampledApriori. Παρακάτω φαίνονται 3 διαφορετικές περιπτώσεις που έτρεξα το demo:
(Οι ενδείξεις της μορφής 2 ---→ 121 μεταφράζονται ως: δημιουργήθηκαν και παρέμειναν 121 2-συνολα)

- SampledApriori με διπλό πέρασμα, sample_size = 100, max_length = 4, min_frequency = 0.1, file = ratings.csv, min_rating = 4.
  myApriori με max_length = 4, min_frequency = 0.1, file = ratings.csv, min_rating = 4.

```
1 ----------------------> 121
2 ----------------------> 410
3 ----------------------> 290
4 ----------------------> 42
Apriori Time--- 127.36688661575317 seconds ---
========FIRST EXECUTION OF SAMPLED APRIORI========
1 ----------------------> 140
2 ----------------------> 556
3 ----------------------> 582
4 ----------------------> 229
========SECOND EXECUTION OF SAMPLED APRIORI=======
1 ----------------------> 154
2 ----------------------> 1133
3 ----------------------> 3196
4 ----------------------> 4517
*** 345 false-positives were removed in the second execution.
Sampled Apriori Time--- 65.065265417099 seconds ---
{'tp': 618, 'fp': 544, 'fn': 245, 'precision': 0.53184165232358, 'recall': 0.7161066048667439, 'f1': 0.6103703703703705}
```

  Σε αυτό το πείραμα βλέπουμε ότι ο αλγόριθμος δούλεψε σε κάποιο βαθμό. Επιπλέον μας γλύτωσε αρκετό χρόνο. Δύο πιστεύω πως είναι οι παράγοντες που έπαιξαν τον μεγαλύτερο ρόλο. Αρχικά το min_frequency = 0.1 είναι σημαίνει ότι πρέπει ένας συνδυασμός να βρίσκεται σε τουλάχιστον 61 καλάθια (συνολικοί χρήστες 610). Αυτό έχει ως αποτέλεσμα να μην περνάνε πολλοί συνδυασμοί το φίλτρο. Αντίθετα στον sampledApriori χρειάζεται να συναντήσουμε ένα συνδυασμό μόνο σε 10 καλάθια (sample_size = 100) με αποτέλεσμα να περνάνε πιο εύκολα οι συνδυασμοί και σε συνδυασμό με το δεύτερο πέρασμα που μειώνει αρκετά false-positives ο sampledApriori καταφέρνει να πιάσει αρκετούς από τους συνδυασμούς.

- SampledApriori με διπλό πέρασμα, sample_size = 50, max_length = 4, min_frequency = 0.1, file = ratings_100users_shuffled.csv, min_rating = 4.
  myApriori με max_length = 4, min_frequency = 0.1, file = ratings_100users.csv, min_rating = 4.

```
1 ----------------------> 140
2 ----------------------> 509
3 ----------------------> 629
4 ----------------------> 368
Apriori Time--- 8.407517433166504 seconds ---
========FIRST EXECUTION OF SAMPLED APRIORI========
1 ----------------------> 250
2 ----------------------> 2386
3 ----------------------> 7375
4 ----------------------> 14442
========SECOND EXECUTION OF SAMPLED APRIORI=======
1 ----------------------> 201
2 ----------------------> 1839
3 ----------------------> 6488
4 ----------------------> 13796
*** 4829 false-positives were removed in the second execution.
Sampled Apriori Time--- 64.81883907318115 seconds ---
{'tp': 1573, 'fp': 18051, 'fn': 73, 'precision': 0.08015695067264574, 'recall': 0.9556500607533415, 'f1': 0.14790785143394453}
```

ΠΑΝΤΕΛΙΔΗΣ ΝΙΚΟΣ
ΑΜ 2787

Σε αυτή την περίπτωση χρησιμοποιούμε τα μπερδεμένα δεδομένα των 100 χρηστών. Πάλι έχουμε το πρόβλημα του προηγούμενου παραδείγματος απλώς τώρα το f1 και το precision score μειώθηκαν σε μεγάλο βαθμό ενώ το recall έφτασε το 0.95 (που είναι ιδανική τιμή). Τα αποτελέσματα αυτά οφείλονται στο αυξημένα False Positives, τα οποία προκύπτουν επειδή χρησιμοποιούμε ίδιο min_frequency και στις δύο παραλλαγές της μεθόδου ενώ χρησιμοποιούν dataset διαφορετικού μεγέθους. Το αυξημένο recall οφείλεται στο ίδιο λόγο, δηλαδή την αδυναμία του συστήματος να φιλτράρει τα δεδομένα. (με μόνο τουλάχιστον 5 εμφανίσεις η ταινία περνάει για sample_size=50 σε αντίθεση με το ολόκληρο αρχείο που χρησιμοποιεί ο myApriori που θέλει τουλάχιστον 10 εμφανίσεις για να περάσει μια ταινία αυτός είναι ο διπλάσιος αριθμός).

- SampledApriori με διακοπή δειγματοληψίας, sample_size = 50, max_length = 4, min_frequency = 0.1, file = ratings_100users_shuffled.csv, min_rating = 4.
  myApriori με max_length = 4, min_frequency = 0.1, file = ratings_100users.csv, min_rating = 4.

```
1 ----------------------> 140
2 ----------------------> 509
3 ----------------------> 629
4 ----------------------> 368
Apriori Time--- 7.856880426406686 seconds ---
========FIRST EXECUTION OF SAMPLED APRIORI========
1 ----------------------> 145
2 ----------------------> 490
3 ----------------------> 543
4 ----------------------> 269
Sampled Apriori Time--- 4.7331321239471436 seconds ---
{'tp': 643, 'fp': 804, 'fn': 1003, 'precision': 0.4443676572218383, 'recall': 0.39064398541919804, 'f1': 0.41577756223731005}
```

  Ο sampledApriori με ένα πέρασμα αντιμετωπίζει το ανάποδο πρόβλημα. Επειδή σταματάμε την δειγματοληψία πριν τελειώσει η ροή είναι σχεδόν σίγουρο ότι πολλά από τα καλάθια του δείγματος μας δεν είναι ολοκληρωμένα. Αυτό έχει ως αποτέλεσμα να μην παράγονται πολλοί συνδυασμοί και επιπλέον αυτοί που περνάνε είναι πιθανό να είναι False Positives. (Σε αυτή την περίπτωση δεν κάνουμε το 2^{ο} πέρασμα του αλγορίθμου που έχει σκοπό την μείωση των False Positives) Εφόσον και οι τρεις δείκτες είναι αρκετά μειωμένοι είναι πολύ πιθανό να υπάρχουν λίγα True Positives.

Σχηματισμός Κανόνων Συσχέτισης

Η ρουτίνα generate_all_rules (pair_finder.py line: 258) χρησιμοποιεί την έξοδο του apriori για να δημιουργήσει τους κανόνες συσχέτισης από το κάθε bag of movies. Αυτό το πετυχαίνει με την χρήση της βοηθητικής ρουτίνας:

- Generate_rules_from_itemset (pair_finder.py line: 290): Ουσιαστικά αυτή η ρουτίνα δέχεται ως είσοδο ένα bags_of_movies ενός συγκεκριμένου μήκους και κάνοντας χρήση αναδρομής παράγει όλους τους κανόνες που πληρούν τις παραμέτρους που δώσαμε.

  Πιο συγκεκριμένα:

  1. Ξεκινάει να σχηματίζει όλους τους κανόνες με την μεγαλύτερη δυνατή υπόθεση (με βάση το μήκος του bag of movies εισόδου)

2. Ελέγχει αν ο κανόνας είναι αποδεκτός.
3. Αν είναι αποδεκτός:
    i. Αποθηκεύουμε τον κανόνα
    ii. Καλούμε την συνάρτηση για υποθέσεις με μία ταινία λιγότερη

4. Αν δεν είναι αποδεκτός
    i. Προχωράμε στον επόμενο κανόνα αυτού του μήκους.

Η συνάρτηση generate_all_rules καλεί την συνάρτηση generate_rules_from_itemset για όλα τα παραχθέντα από τον apriori bags of movies. Τέλος μας επιστρέφει ένα dataframe με όλους τους κανόνες που δημιουργήθηκαν και τα γνωρίσματα τους.

Οδηγίες χρήσης μενού

Στο αρχείο menu.py υλοποιώ το app που μας ζητήσατε. Μόλις τρέξετε το αρχείο θα σας εμφανίσει την οθόνη επιλογής αρχείου. Θα σας ζητηθεί να επιλέξετε μία από τις 3 επιλογές για να διαλέξετε ένα από τα 3 αναγραφόμενα αρχεία. Μόλις επιλέξετε αρχείο θα σας ζητηθεί να δώσετε Min_Score.

**Θα πρέπει να βάλετε τα σωστά paths στις γραμμές 29,30,31 στο αρχείο menu.py

```
==========LOADING OPTIONS==========

(1)   Load ratings.csv

(2)   Load ratings_100users.csv

(3)   Load ratings_100user_shuffled.csv
(This option is used in sampled apriori. It returns a stream of the ratings)

2
Give a Min-Score(0,5):
4
```

Στην συνέχεια θα εμφανιστεί το menu του apriori όπου θα πρέπει να δώσετε τις παραμέτρους που θέλετε.

```
=================APRIORI OPTIONS=================

(1)   Classic Apriori   #,MinFrequency,MaxLength,MinConfidence,MinLift,MaxLift

(2)   Sampled Apriori   #,MinFrequency,MaxLength,MinConfidence,MinLift,MaxLift,SampleSize

MinFrequency,MinConfidence,MinLift,MaxLift -------> (0,1)
You can disable MinLift,MaxLift by giving -1 as input
Example_1: 1,0.1,4,0.5,-1,-1  Example_2: 2,0.5,5,0.5,0.2,-1,100

1,0.1,4,0.5,4,-1
============== Apriori Execution ===============
1 ---------------------> 140
2 ---------------------> 509
3 ---------------------> 629
4 ---------------------> 368
Min Frequency:  0.1
Max Length:  4
Min Confidence:  0.5
Min Lift:  4.0
Max Lift:  -1.0
Generated 15 rules in 8.305092811584473 seconds
```

**Για να τρέξετε τον sampledApriori θα πρέπει να διαλέξετε την 3η επιλογή του loading_menu.

Μόλις ολοκληρωθεί η εκτέλεση του αλγορίθμου θα εμφανιστεί το κύριο μενού.

```
================================

(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]

(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|INTEREST>
                                                   [format: s,<c,l,i>]
(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]
```

Στο κύριο μπορείτε να επιλέξετε όποια λειτουργία θέλετε αρκεί να γράψετε σωστά την σύνταξη της εντολής.

Παρακάτω φαίνονται εικόνες από όλες τις λειτουργίες του μενού:

(a) Εμφάνιση όλων των διαθέσιμων κανόνων συσχέτισης

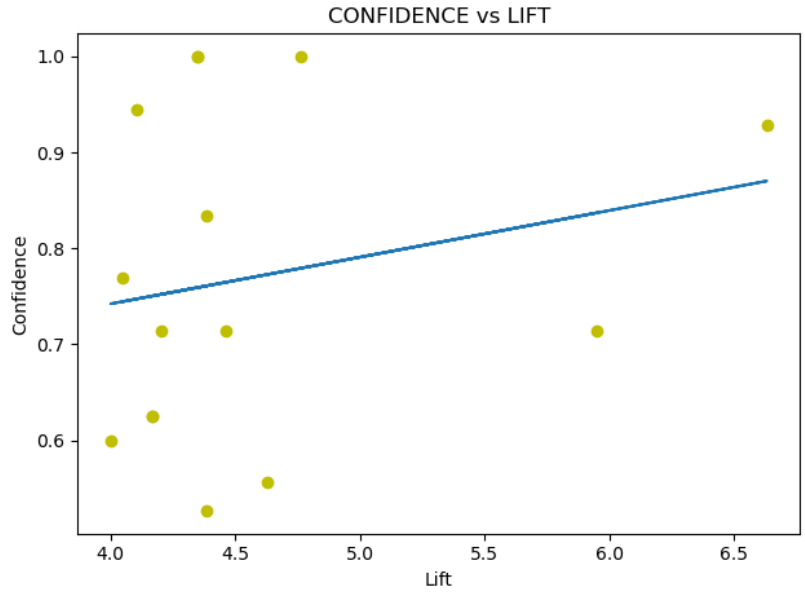
```
a
                itemset                    rule    hypothesis    conclusion  frequency  confidence      lift  interest  rule ID
0          [1136, 1213]          [1213]-->[1136]        [1213]        [1136]       0.10    0.714286  4.464286     -0.06        1
1          [1089, 1214]          [1214]-->[1089]        [1214]        [1089]       0.12    0.600000  4.000000     -0.03        2
2          [4993, 5952]          [5952]-->[4993]        [5952]        [4993]       0.17    0.944444  4.106280     -0.06        3
3          [4886, 6539]          [6539]-->[4886]        [6539]        [4886]       0.10    0.625000  4.166667     -0.05        4
4        [58559, 91529]        [91529]-->[58559]       [91529]       [58559]       0.10    1.000000  4.761905     -0.11        5
5          [6874, 7438]          [7438]-->[6874]        [7438]        [6874]       0.13    0.928571  6.632653     -0.01        6
6          [1221, 1610]          [1610]-->[1221]        [1610]        [1221]       0.10    0.769231  4.048583     -0.09        7
7          [4878, 7438]          [7438]-->[4878]        [7438]        [4878]       0.10    0.714286  5.952381     -0.02        8
8       [21, 50, 296]       [50, 296]-->[21]     [50, 296]          [21]       0.10    0.555556  4.629630     -0.02        9
9   [4993, 5952, 7153]  [5952, 7153]-->[4993]  [5952, 7153]        [4993]       0.15    1.000000  4.347826     -0.08       10
10  [4993, 5952, 7153]  [5952]-->[4993, 7153]        [5952]  [4993, 7153]       0.15    0.833333  4.385965     -0.04       11
11  [4993, 5952, 7153]  [7153]-->[4993, 5952]        [7153]  [4993, 5952]       0.15    0.714286  4.201681     -0.02       12
12  [4993, 6539, 7153]  [6539, 7153]-->[4993]  [6539, 7153]        [4993]       0.10    1.000000  4.347826     -0.13       13
13  [1089, 1221, 2571]  [1221, 2571]-->[1089]  [1221, 2571]        [1089]       0.10    0.625000  4.166667     -0.05       14
14  [1089, 1221, 2571]  [1221]-->[1089, 2571]        [1221]  [1089, 2571]       0.10    0.526316  4.385965     -0.02       15
Would you like to continue?
Press e to exit or anything else to continue:
```

(b) Εμφάνιση όλων των κανόνων που περιέχουν μία λίστα από ταινίες στο επιλεγμένο πεδίο

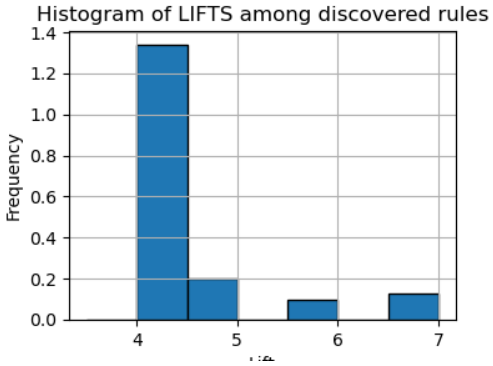
```
b,c,7153,4993
                itemset                    rule hypothesis    conclusion  frequency  confidence      lift  interest  rule ID
10  [4993, 5952, 7153]  [5952]-->[4993, 7153]     [5952]  [4993, 7153]       0.15    0.833333  4.385965     -0.04       11
Would you like to continue?
Press e to exit or anything else to continue:
```

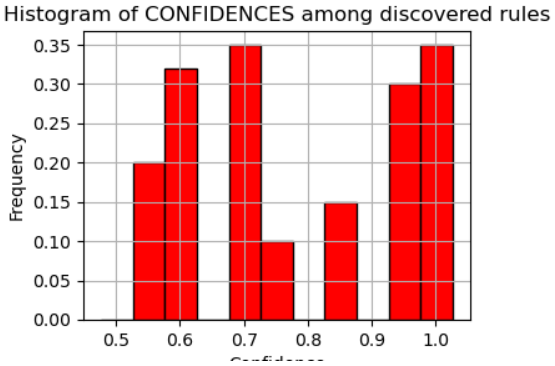
```
b,i,4993,7153
                itemset                    rule    hypothesis    conclusion  frequency  confidence      lift  interest  rule ID
9   [4993, 5952, 7153]  [5952, 7153]-->[4993]  [5952, 7153]        [4993]       0.15    1.000000  4.347826     -0.08       10
10  [4993, 5952, 7153]  [5952]-->[4993, 7153]        [5952]  [4993, 7153]       0.15    0.833333  4.385965     -0.04       11
11  [4993, 5952, 7153]  [7153]-->[4993, 5952]        [7153]  [4993, 5952]       0.15    0.714286  4.201681     -0.02       12
12  [4993, 6539, 7153]  [6539, 7153]-->[4993]  [6539, 7153]        [4993]       0.10    1.000000  4.347826     -0.13       13
Would you like to continue?
Press e to exit or anything else to continue:
```

(c) Σύγκριση confidence-lift κανόνων σε scatter plot



(h) Εμφάνιση ιστογραμμάτων confidence, lift





(m) Εμφάνιση λεπτομερειών συγκεκριμένης ταινίας

```
m,7153
      movieId                                              title                        genres
4800     7153  Lord of the Rings: The Return of the King, The...  Action|Adventure|Drama|Fantasy
```

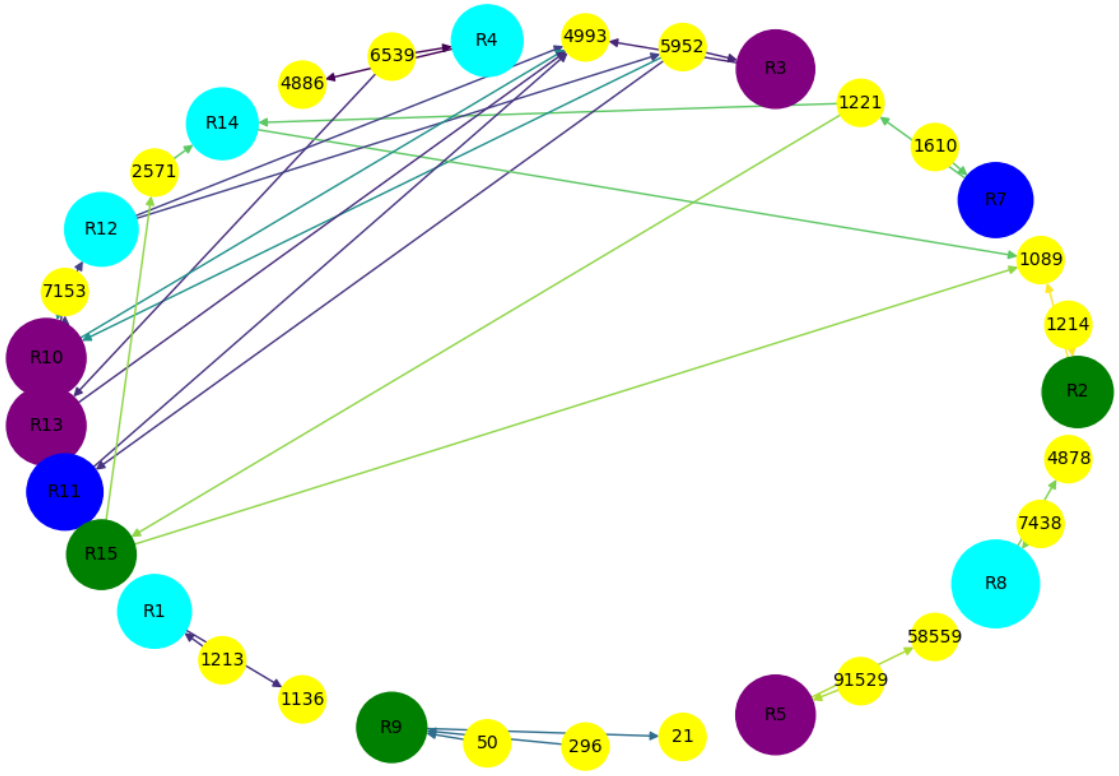
(r) Εμφάνιση λεπτομερειών συγκεκριμένου κανόνα

```
r,10
           itemset                rule    hypothesis conclusion  frequency  confidence      lift  interest  rule ID
9  [4993, 5952, 7153]  [5952, 7153]-->[4993]  [5952, 7153]     [4993]       0.15         1.0  4.347826     -0.08       10
Would you like to continue?
Press e to exit or anything else to continue:
```

(s) Εμφάνιση κανόνων σε ταξινομημένη σύμφωνα με την επιλεγμένη στήλη μορφή

ΠΑΝΤΕΛΙΔΗΣ ΝΙΚΟΣ
ΑΜ 2787

(v) Εμφάνιση κανόνων σε μορφή γραφήματος



(e) Έξοδος από το πρόγραμμα

Experiments

Στο φάκελο αυτό έχω τα εξής αρχεία:

- **Hash_counter_100.json:** περιέχει το λεξικό με τους μετρητές ζευγαριών που δημιουργήθηκαν με hashing
- **Triangular_matrix_100.txt:** περιέχει μια λίστα που είναι ο κάτω τριγωνικός πίνακας σε σειριακή μορφή και έχει τους μετρητές των ζευγαριών
- **Myapriori_rules_100.csv:** περιέχει το dataframe με τους κανόνες που δημιουργήθηκαν από τον απλό apriori για min_score: 4, min_frequency: 0.1, max_length:4, min_confidence:0.5, min_lift=max_lift=-1 στο αρχείο ratings_100users.csv
- **Sampledapriori_rules_100.csv:** περιέχει το dataframe με τους κανόνες που δημιουργήθηκαν από τον sampled apriori για min_score: 4, min_frequency: 0.1, max_length:4, min_confidence:0.5, min_lift=max_lift=-1 στο αρχείο ratings_100users_shuffled.csv