Σπυρος Μεσσηνης ΑΜ:4421

1ο Σετ Ασκήσεων: Συχνά Σύνολα Αντικειμένων & Ανίχνευση Κοινοτήτων

Υλοποιηση εργασιας

Ο στόχος της εργασίας ήταν ο εντοπισμός συχνά εμφανιζομένων συνόλων αντικειμένων, σε ένα σύνολο δεδομένων (Movielens), που αφορά ένα συστημα αξιολογησης ταινιων απο διαφορους χρηστες.

Το σύνολο αυτό είναι ένα αρχείο τύπου csv, όπου σε αυτό περιέχεται ένα id που αποδίδεται σε κάθε χρήστη, οι ταινίες που έχει ο καθένας, πάλι ονοματισμένες με ενα id, και η αξιολόγηση της συγκεκριμένης ταινίας από τον χρήστη.

Ξεκίνησα την εργασία δημιουργώντας μια συνάρτηση ReadRatings, όπου διαβάζει το συγκεκριμένο αρχείο και το αποτυπώνει σε ένα datafame της βιβλιοθήκης Pandas, με τα εξής ονόματα στηλών:

- 'userId' , ο κωδικός του καθε χρηστη
- 'movieId' , ο κωδικος της καθε ταινιας
- 'rating' , η αξιολογηση της ταινιας απο τον χρηστη
- 'timestamp' , τα δευτερολεπτα που περασαν απο την 1η Ιανουαριου 1970

Επίσης δημιούργησα μια άλλη συνάρτηση ReadMovies, η οποία διαβάζει ενα αρχείο με όνομα movies.csv όπου κάθε γραμμή περιέχει το id της κάθε ταινίας, το όνομα και το genre της. Πάλι η συνάρτηση επιστρέφει ένα datafame της βιβλιοθήκης Pandas, με τα εξής ονόματα στηλών:

- 'movieId' , id της καθε ταινιας
- 'title' ,τιτλος της καθε ταινιας
- 'genres' , κατηγορια ταινιας

Έπειτα υλοποίησα την συνάρτηση CreateUserBaskets που δημιουργεί μια συλλογή καλαθιών, δηλαδή χρηστών, με τις ταινίες που έχει δει ο καθένας. Επιπλέον η συνάρτηση δημιουργεί και επιστρέφει ενα dictionary of frozensets το οποίο κάθε κλειδί του dictionary είναι ένας χρήστης και κάθε value αυτού είναι ένα frozenset που περιέχει τις ταινίες που έχει δει ο καθένας. Αποφάσισα να χρησιμοποιήσω ενα dictionary of frozensets αντί για ένα ένα dictionary of dictionaries, γιατί δεν χρησιμοποιούμε πουθενά τα ratings που έδωσαν οι χρήστες σε κάθε ταινία που είδαν. Επισης μετατρεπω το dictionary of frozensets σε ένα dataframe και το αποθηκεύω σε ένα αρχείο my_userBaskets.csv.

Επίσης υλοποίησα την συνάρτηση CreateMovieBaskets η οποία δημιουργεί αντίστοιχα μια συλλογή καλαθιών, τώρα όμως ταινιών, με θεατές που έχουν δει την συγκεκριμένη ταινία. Επιπλέον η συνάρτηση, ομοίως πάλι με την CreateUserBaskets δημιουργεί και επιστρέφει ένα dictionary of frozensets το οποίο κάθε κλειδί του dictionary είναι μια ταινία και κάθε value αυτού είναι ένα frozenset που περιέχει τους χρήστες που έχουν δει την συγκεκριμένη ταινία. Επίσης πάλι μετατρέπω το dictionary of frozensets σε ένα dataframe και το αποθηκεύω σε ένα αρχείο my_movieBaskets.csv.

Όσο για την υλοποίηση του αλγορίθμου Apriori, δημιούργησα τις συναρτήσεις find_frequent_itemsets, myApriori. Η συνάρτηση myApriori δέχεται ως είσοδο 3 ορίσματα, ένα καλάθι (itemBasket), το minSupport το οποίο είναι ένα ελάχιστο κατώφλι που πρέπει να ξεπερνούν τα αντικείμενα κάθε καλαθιού και το maxLength, το οποίο είναι το μέγιστο μέγεθος των itemsets που ψάχνουμε να βρούμε, με τυπική τιμή maxLength = 5. Ξεκίνησα προσθέτοντας στο dataframe που δημιούργησα παραπάνω στην συνάρτηση ReadRatings μια παραπάνω στήλη με το όνομα counter όπου μετράει ποιοι χρήστες έχουν δει μια ταινία ή ποιες ταινίες έχει δει ένας χρήστης. Έπειτα για κάθε καλάθι που το περιεχόμενο του ξεπερνάει το κατώφλι στηρίγματος, minSupport προσθέτω το μονοσύνολο αυτό σε ένα dictionary που έχει ως key ένα frozenset με το id της ταινίας και ως value πόσες φορές εμφανίζεται στο καλάθι.

Έτσι επαναληπτικά μέχρι τον maxLength, καλούμε την συνάρτηση find_frequent_itemsets οπού βρίσκει κάθε κ-άδα από το προηγούμενο πιο συχνό σύνολο αντικειμένων ,πχ κάθε δυάδα από τα πιο συχνά μονοσύνολα ή κάθε τριάδα από τις πιο συχνές δυάδες, που ξεπερνούν το minSupport, και τις τοποθετεί σε πάλι σε ένα dictionary που έχει ως key frozensets με τα id's των ταινιών και ως value πόσες φορές εμφανίζονται στο καλάθι, πχ πόσοι χρήστες έχουν δει την ίδια ή τις ίδιες ταινίες. Τέλος στη συνάρτηση myApriori μετατρέπω το dictionary of frozensets σε μια λίστα από λίστες, frequentItemsets που περιέχει όλα τα μονοσύνολα, ζεύγη, τριάδες κλπ. αντικειμένων, δηλαδή όπου

frequentItemsets[0] = $L1$ (συχνά εμφανιζόμενα μονοσύνολα), frequentItemsets [1] = $L2$ (συχνά δι-σύνολα), frequentItemsets [2] = $L3$ (συχνά τρι-σύνολα),…, έως και μια λίστα frequentItemsets [$K - 1$] = LK με όπου είτε K = maxLength.

Πιο συγκεκριμένα η find_frequent_itemsets δέχεται ως ορίσματα το καλάθι που χρησιμοποιούμε εκείνη τη στιγμή, το προηγούμενο σύνολο αντικειμένων, πχ κάθε δυάδα από τα πιο συχνά μονοσύνολα ή κάθε τριάδα από τις πιο συχνές δυάδες, και το minSupport. Ξεκινάμε και βρίσκουμε όλες τις δυάδες, τριάδες κλπ απο το προηγούμενο σύνολο k_1_itemsets, που ξεπερνούν το minSupport και μετράμε πόσες φορές εμφανίζεται κάθε επόμενη όμαδα αντικειμένων. Έπειτα επιστρέφουμε το dictionary of frozensets με αντικείμενα μόνο αυτά που ξεπερνούν το minSupport.

Τώρα για την υλοποίηση της ExactCounting χρησιμοποίησα πάλι την τεχνική που έκανα και με την myApriori, δημιουργώντας μια επιπλέον στήλη στο dataframe με όνομα counter που κάνει ακριβώς την ίδια δουλειά με την παραπάνω. Έπειτα επαναληπτικά καλούμε πάλι την συνάρτηση find_frequent_itemsets οπού πάλι βρίσκει κάθε κ-άδα αντικειμένων, ωστόσο τώρα δεν έχουμε κάποιο minSupport και βρίσκουμε κάθε μονοσύνολο, δυάδα κλπ. που εμφανίζεται τουλάχιστον 1 φορά. Έτσι επιστρέφουμε ένα dictionary of dictionary of frozensets, οπού κάθε υπό - dictionary of frozensets έχει ως key μια κ-άδα αντικειμένων και ως value τις εμφανίσεις του.

Όσο για την ρουτίνα SON την χώρισα σε δύο φάσεις. Την πρώτη φάση που εκτελεί τον APRIORI σε κάθε chunk χωριστά και κοιτάει ποια πλειάδα ξεπερνά το κατώφλι στηρίγματος, (minSupport). Πάλι χρησιμοποιώ την τεχνική που ανέφερα παραπάνω, δημιουργώντας μια επιπλέον στήλη στο dataframe με όνομα counter και δημιουργώ το itemset με τα πιο συχνά μονοσύνολα, δηλαδή αυτά που ξεπερνούν το minSupport. Έπειτα με μια βοηθητική συνάρτηση chunks χωρίζω το συγκεκριμένο dictionary of frozensets σε κομμάτια(chunks) ανάλογα το chunkSize που έχει δώσει ο χρήστης. Και εκτελώ τον Apriori σε καθένα από αυτά. Όποια chunks περνούν το minSupport τα κρατάω σε μια λίστα και «προχωράω» στην επόμενη κ-άδα, δηλαδή στις δυάδες. Και το επαναλαμβάνω μέχρι και το maxLength.

Τώρα για το δεύτερο πέρασμα του SON δημιουργώ την συνάρτηση phase2 και μέσα σε αυτή καλώ την συνάρτηση ExactCounting και το πρώτο πέρασμα της

SON. Το σκεπτικό μου είναι ότι αν ένα chunk του SON δεν περιέχεται μέσα σε ένα itemset του ExactCounting τότε πρέπει να το σβήσω γιατί είναι False-Positive. Έτσι αποφάσισα να κοιτάω αν το κάθε chunk του SON, για όλες τις κ-άδες αντικειμένων, είναι υποσύνολο της αντίστοιχης κ-άδας στο ExactCounting. Αν δεν ειναι τελικά υποσύνολο, σβήνουμε το συγκεκριμένο chunk και επιστρέφουμε το αλλαγμένο dictionary.

Τέλος την presentResults την έχω υλοποίηση ακριβώς όπως λέτε στην εκφώνηση, ωστόσο έχω αφαιρέσει τις επιλογές *Load baskets from csv file* και *Save baskets to csv file*, αφού γίνονται αυτόματα από το τις συναρτήσεις CreateUserBaskets και CreateMovieBaskets.

Για να εκτελέσετε ή τον Apriori ή τον SON πρέπει πρώτα να δημιουργήσετε ενα καλάθι και μετά να καλέσετε τον αντίστοιχο αλγόριθμο και πρέπει να το αλλάξετε απο την συναρτηση presentResults (έχω την SON σε σχόλιο).

Επιπλέον για την συνάρτηση presentResults έφτιαξα κάποιες βοηθητικές συναρτήσεις απλά για να δείχνει κάποια δεδομένα για μια ταινία, επιλογή (M), και για κάποιο χρήστη, επιλογή(U).

Για τις συγκεκριμένες συναρτήσεις απλά ψάχνω στο dataframe που δημιούργησα ReadRatings, αν ψάχνω για users, ή στο ReadMovies αν ψάχνω για ταινίες και τυπώνω τα ανάλογα δεδομένα.

Για την εργασια χρησιμοποιησα τις παρακατω βιβλιοθηκες:

- Την βιβλιοθήκη pandas, για τα dataframes.
- Από την βιβλιοθήκη collections το defaultdict, για ένα dictionary.
- Από την βιβλιοθήκη itertools το islice, για να χωρισω τα chunks.
- Την βιβλιοθήκη matplotlib.pyplot, για την σχεδίαση του ιστογράμματος.
- Την βιβλιοθήκη time, για να μετρήσω πόσο χρόνο κάνει για να εκτελεστεί η κάθε μέθοδος.

Apriori

Δυάδες ταινιών που παρακολουθήθηκαν από τουλάχιστον 50 κοινούς θεατές

Time : 0.47 minutes

Αρχείο αποτελεσμάτων : results_movies_Apriori.txt

Δυάδες θεατών που παρακολούθησαν τουλάχιστον 5 ταινίες από κοινού

Time : 0.2 minutes

Αρχείο αποτελεσμάτων : results_users_Apriori.txt

100 users

Τριάδες ταινιών που παρακολουθήθηκαν από τουλάχιστον 15 κοινούς θεατές

Time : 1.07 minutes

Αρχείο αποτελεσμάτων : results_movies_Apriori_100.txt

Τριάδες θεατών που παρακολούθησαν τουλάχιστον 2 ταινίες από κοινού

Time : 0.08 minutes

Αρχείο αποτελεσμάτων : results_users_Apriori_100.txt

SON

Δυάδες ταινιών που παρακολουθήθηκαν από τουλάχιστον 50 κοινούς θεατές

Time : 0.43 minutes

Αρχείο αποτελεσμάτων : results_movies_SON.txt

Δυάδες θεατών που παρακολούθησαν τουλάχιστον 5 ταινίες από κοινού

Time : 0.18 minutes

Αρχείο αποτελεσμάτων : results_users_SON.txt

100 users

Τριάδες ταινιών που παρακολουθήθηκαν από τουλάχιστον 15 κοινούς θεατές

Time : 0.78 minutes

Αρχείο αποτελεσμάτων : results_movies_SON_100.txt

Τριάδες θεατών που παρακολούθησαν τουλάχιστον 2 ταινίες από κοινού

Time : 0.08 minutes

Αρχείο αποτελεσμάτων : results_users_SON_100.txt

Χρησιμοποίησα και το αρχειο ratings_100users.csv, διότι, οσο χρησιμοποιούσα το ratings.csv μετά απο κάποιο σημείο ο υπολογιστής μου κολλούσε υπερβολίκα και χρειαζόταν restart.