Assignment #1

by: Sirinian Aram Emmanouil AM: 2537

Ερώτηση 1η

1) Ο αλγόριθμος ξεκινώντας αρχηκοποιεί μια δομή, μεγέθους K το οποίο ο χρήστης το καθορίζει αρχικά. Ο αλγόριθμος επίσης κρατάει έναν μετρητή, ο οποίος αρχικά είναι μηδενισμένος αλλά μετά την ανάγνωση ενός αντικειμένου από την είσοδο τον αυξάνει κατά ένα. Για κάθε αντικείμενο που διαβάζει από την είσοδο μπαίνει σε ένα βρόχο. Ο βρόχος εκτελείται K φορές, η κάθε εκτέλεση του αντιστοιχίζεται με μια θέση της δομής και έπειτα εκτελείται ο αλγόριθμος Reservoir Sampling. Τα αποτελέσματα των εκτελέσεων του αλγορίθμου Reservoir Sampling αποθηκεύονται στις αντίστοιχες θέσεις μνήμης της δομής.

2)

Ρεμφάνισης(xi) : Η πιθανότητα εμφάνισης του I-στου στοιχείου στο δείγμα.

Ρεμφάνισης στο j(xi) : Η πιθανότητα εμφάνισης του I-στου στοιχείου στο δείγμα στην θέση j της δομής.

$$Ρεμφάνισης(xi) = \sum_{j=0}^{j=K-1} Ρεμφάνισης\ στο\ j(xi) = \frac{K}{N}$$

Reservoir sampling
•Algorithm: With probability 1/k select the k-th item of the stream and replace the previous choice.

Ερώτηση 3η

1) Η σχέση μεταξύ της υποστήριξης του X και του X΄ είναι: $S(X') \geq S(X)$

2)

Όλα τα συχνά στοιχειοσύνολα που αποτελούνται από στοιχεία υψηλότερου επιπέδου στην ταξινομία είναι: {12}, {13}, {14}, {15}, {12, 14}, {12, 15}, {14, 15}, {12, 14, 15}

3)

Ο αλγόριθμος αρχικά δημιουργεί μιά δομή όπου αποθηκεύονται οι πρόγονοι όλων των φύλλων του δέντρου. Με την χρήση αυτής της δομής μπορούμε να βρίσκουμαι όλα τα συχνά στοιχειοσύνολα με τα φύλλα ή χωρίς συμπεριλαμβάνοντας τους προγόνους μιας και τους έχουμε αντιστοιχισμένους με τα φύλλα στην δομή. Επείτα εκτελούμε τον αλγόριθμο Apriori

στην νέα μας δομή.

Apriori algorithm

•Assumption: The items in an itemset are ordered

Ck = candidate itemsets of size k
Lk = frequent itemsets of size k

1.k =1,C1 = all items
2. While Ck not empty
    3.Scan the database to find which itemsets in Ck are frequent and put them into Lk
    4. Generate the candidate itemsets Ck+1 of size k+1 using Lk
    5. k =k+1

Generate Candidates Ck+1
•We have all frequent k-itemsets Lk
•Step 1: self-join Lk
    •Create set Ck+1 by joining frequent k-itemsets that share the first k-1 items
•Step 2:prune
    •Remove from Ck+1 the itemsets that contain a subset k-itemset that is not frequent

Computing Frequent Itemsets
•Given the set of candidateitemsets Ck, we need to compute the support and find the frequent itemsets Lk.
•Scan the data, and use a hash structure to keep a counter for each candidate itemset that appears in the data.

Assignment #1

by: Sirinian Aram Emmanouil AM: 2537

Ερώτηση 2η (report)

Για να μπορέσουμε να εξάγουμε τις συναρτήσεις που συνδέουν τα δεδομένα των στήλων B, C με την στήλη A φορτώνουμε τα δεδομένα αυτά σε ένα DataFrame. Η απεικόνιση των δεδομένω σε γραφήματα και ποιο συγκεκριμένα, γραφήματα των B και C ως προς το A είναι ο καλύτερος τρόπος για να παρατηρήσουμε διάφορες συμπεριφορές μεταξύ τους. Η απλές αποικονίσεις δεν αποκαλύπουν κάποια σχέση μεταξύ τους κατί που μπορεί κανείς να διακρίνει κάνοντας απλές προσπάθειες. Παρόλο αυτά η απεικόνιση των δεδομένων σε λογαριθμικές κλίμακες τις φανερώνουν.

Μέσο του γραφήματος B ως προς το A σε λογάριθμο του 10 μορφή. Παρατηρούμε μια συσχέτιση μορφής ευθείας γραμής $\log(y)=k*\log(x)+\log(a), y=a*x^k$. Ποιό συγκεκριμένα παρατηρώντας ότι η ευθεία διαπερνάει από τα σημεία (1,1) και (10, 100) μπορούμε να συμπεραίνουμε ότι η συνάρτηση που περιγράφει τις δύο στήλες είναι $\log(y)=2*\log(x), y=x^2$ (όπου y: A, x: B).

Μέσο του γραφήματος C ως προς το A σε λογάριθμό του 10 μορφή. Παρατηρούμε μια συσχέτιση μορφής ευθείας γραμής $\log(y)=\log(a), y=a$. Ποιό συγκεκριμένα παρατηρώντας δύο τέτοιες ευθείες οι οποίες στο περίπου είναι οι εξείς: $y(\alpha)\approx -0.9, \alpha\in(-\infty, 0)$ & $y(\alpha)\approx 4.06, \alpha\in(0,\infty)$

Ερώτηση 4η (report)

Αρχικά ο αλγόριθμος προσθέτει όλες τις χρήσιμες συναρτήσεις μέσο των βιβλιοθηκών και κατασκευάζει ένα Dataframe χρησιμοποιώντας την αποθηκευμένοι πληροφορία μέσα στο αρχείο business.json. Στην συνέχεια αφερούνται όλες οι εγγραφές του Dataframe που έχουν το πεδίο "categories" ή το "neighborhood" κενό ή όσες εγγραφές έχουν το πεδίο city διάφορα του "Toronto". Έπειτα δημιουργούμε μια νέα λίστα, που θα περιέχει τις εγγραφές του Dataframe με τα πεδία "business_id", "categories" ομαδοποιημένα με βάση το πεδίο "neighborhood". Διαγράφουμε τις γειτονιές που δεν έχουν τουλάχιστον 100 επιχειρήσεις. Τώρα ο αλγόριθμος για να βρεί για κάθε περιοχή τις 10 κατηγορίες που την περιγράφουν καλύτερα χρησιμοποιεί την μέθοδο TF-IDF. Η μέθοδος αυτή είναι πολύ καλή γιατί πέρνει εποχή την συχνότητα μιας κατηγορίας σε μία γειτονιά αλλά και την μοναδικότητα της κατηγορίας σε όλες τις γειτονιές. Οι δύο αυτές μεταβλητές υπολογίζονται με τις μεθόδους TF(w, d) και IDF(w) αντίστοιχα. Το γινόμενο των τιμών που επιστρέφουν οι μέθοδοι αυτοί αποτελούν και το αποτέλεσμα που επιστρέφει η TF-IDF. Στην συνέχεια κατασκευάζετε μια λίστα από όλες τις κατηγορίες όλων των γειτονιών χρησιμοποιώντας την τροποποιημένη μας λίστα, με αυτό δημιουργείτε το set του. Με την χρήση του set από κατηγορίες δημιουργεί ενα dictionary το οποίο παίρνει ως κλειδί μια εγγραφή του set και για value το

IDF της. Τέλος ο αλγόριθμος για κάθε περιοχή υπολογίζει το TF-IDF των κατηγοριών, χρησιμοποιώντας το λεξικό που δημιουργήσαμε προιγουμένος ύστερα υπολογίζει μια λίστα με τα 10 χαρακτηριστικά με τα ποιό μεγάλα TF-IDF και την εισάγει στην λίστα που θα περιλαμβάνει όλα τα τελικά μας αποτελέσματα. Οι κατηγορίες των γειτονιών που μας επιστρέφει η μέθοδος είναι όντως αυτά που χαρακτηρίζουνε την γειτονιά αλλά και δεν εμφανίζονται συχνά και στις άλλες γειτονιές δηλ. όλες οι stop word λέξεις απαλείφονται μέσα στο γινόμενο.