ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΔΥΤΙΚΗΣ ΕΛΛΑΔΑΣ

ΣΧΟΛΗ ΔΙΟΙΚΗΣΗΣ & ΟΙΚΟΝΟΜΙΑΣ

ΤΜΗΜΑ ΔΙΟΙΚΗΣΗΣ ΕΠΙΧΕΙΡΗΣΕΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΚΑΝΟΝΕΣ ΣΥΣΧΕΤΙΣΗΣ ΓΙΑ ΤΗΝ ΕΞΟΡΥΞΗ ΔΕΔΟΜΕΝΩΝ ΕΠΙΧΕΙΡΗΣΕΩΝ

Κολώνια Αγγελική

Στείρου Αθηνά

Επιβλέπων Καθηγητής: Δρ. Χαλκιόπουλος Κωνσταντίνος

ΠΑΤΡΑ, ΙΟΥΝΙΟΣ 2016

ΠΕΡΙΛΗΨΗ

Ο σκοπός αυτής της διπλωματικής εργασίας είναι η ανάδειξη της ωφελιμότητας της **εξόρυξης γνώσης** από Βάσεις Δεδομένων επιχειρήσεων. Μέσα από μία εισαγωγική παράθεση διάφορων μεθόδων εξόρυξης γνώσης, οδηγούμαστε στην επιλογή των **Κανόνων Συσχέτισης** ως τη μέθοδο εξόρυξης γνώσης που θα αναλύσουμε και που πάνω σε αυτήν θα στηρίξουμε την έρευνά μας.

Για να κατανοήσουμε τον τρόπο με τον οποίο παράγεται η γνώση μέσω των Κανόνων Συσχέτισης, επιλέξαμε αλγόριθμους εύρεσης κανόνων όπως ο **Apriori** και ο **FP-Growth**. Αρχικά παραθέτουμε τον τρόπο με τον οποίο ο κάθε αλγόριθμος λειτουργεί, παίρνοντας το χαρακτηριστικό παράδειγμα των "**καλαθιών αγοράς**". Μέσα από την περιγραφή λειτουργίας του κάθε αλγόριθμου ανακαλύπτουμε τη σημασία θεμελιωδών μεγεθών που αφορούν τους Κανόνες Συσχέτισης, όπως η Εμπιστοσύνη και η Υποστήριξη.

Στη συνέχεια περνάμε από την θεωρία στην πράξη. Χρησιμοποιώντας δειγματοληπτικές Βάσεις Δεδομένων άλλοτε με λίγες εγγραφές και άλλοτε πιο μεγάλες, πειραματιζόμαστε με τους αλγόριθμους Apriori και FP-Growth, ώστε να ανακαλύψουμε την εξόρυξη γνώσης που πετυχαίνουν μέσα από τις πληροφορίες που δίνουμε από τις Βάσεις Δεδομένων. Αφενός για να κερδίσουμε χρόνο και αφετέρου για να προσδώσουμε αξιοπιστία στα αποτελέσματα των αλγορίθμων, επιλέξαμε το **WEKA**, ένα λογισμικό εξόρυξης γνώσης που, μεταξύ άλλων, παρέχει στο χρήστη τη δυνατότητα εξαγωγής Κανόνων Συσχέτισης και μάλιστα με τη χρήση των αλγορίθμων Apriori και FP-Growth. Μέσα από τα αποτελέσματα του WEKA οδηγούμαστε σε αξιολόγηση των Κανόνων Συσχέτισης όσον αφορά την εγκυρότητα, την ουσία και την ωφελιμότητά τους σε μία πιθανή λήξη απόφασης για μία επιχείρηση. Επιπροσθέτως, προχωρήσαμε και στην σύγκριση μεταξύ των δύο αλγορίθμων για να κατανοήσουμε τις περιπτώσεις που θα πρέπει να προτιμάται ο ένας αντί του άλλου.

Τέλος, εξαγάγαμε και συγκρίναμε τους χρόνους που απαιτούν οι αλγόριθμοι Apriori και FP-Growth για την εξαγωγή Κανόνων Συσχέτισης από μία Βάση Δεδομένων. Η διαδικασία αυτή έγινε για διαφορετικές τιμές των μέτρων σπουδαιότητας των αλγορίθμων προκειμένου να βρεθεί ποιος εκ των δύο είναι πιο αποδοτικός.

Συνεπώς, η διπλωματική μας εργασία εστιάζει μεν στην εξόρυξη γνώσης από Βάσεις Δεδομένων επιχειρήσεων, αλλά με τέτοιο τρόπο ώστε αυτοί οι κανόνες να είναι όσο το δυνατόν πιο ουσιαστικοί για εγκυρότερα συμπεράσματα, αλλά και με σκοπό να αποκόπτονται οι περιττοί κανόνες ώστε η επιχείρηση να κερδίζει χρόνο και χρήμα.

ABSTRACT

The purpose of this project is to promote the significance of **Data Mining** as a tool of **extracting knowledge** from Databases of companies. Through an introductory quote of various mining methods, we were led to the choice of **Association Rules** as the mining method that will be analyzed and with which we will support our research.

To understand the way in which knowledge is produced through Association Rules we have chosen rule-finding algorithms such as **Apriori** and **FP-Growth**. Initially we mention the manner in which each algorithm works by taking the example of the "**market basket**". Through the functional description of each algorithm we discover the importance of fundamentals on the Association Rules, such as **Trust** and **Support**.

Then we go from theory to practice. Using Sample Databases sometimes with a few records and sometimes larger Databases, we practice with the algorithms Apriori and FP-Growth, in order to discover their data mining abilities. Both to save time and also to lend credibility to the results of the algorithms, we chose **WEKA**, a data mining software which, among others, provides the user with the ability to export Association Rules including the use of algorithms Apriori and FP-Growth. Through the results of WEKA we are led to evaluate Association Rules regarding the validity, the substance and usefulness to a possible vital decision for a company. In addition, we proceeded to a comparison between the two algorithms to understand the situations that should be preferred instead of each other.

Finally, we found and compared the run times required by the algorithms Apriori and FP-Growth for the Association Rules mining from a known Database. This procedure occurred for different values of measure of interest of the algorithms in order to find which of the two is most efficient.

Therefore, our thesis focuses on extracting knowledge from companies' databases from Association Rules, but in such a way that these rules will be as possible essential for authoritative conclusions, and in order to cut out unnecessary rules so that the companies can save time and money.

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΙΣΑΓΩΓΗ

Κάθε επιχείρηση ή οργανισμός διατηρεί αρχεία τα οποία φυλάσσονται στις αντίστοιχες Βάσεις Δεδομένων των οργανισμών και των επιχειρήσεων. Ο όγκος αυτών των δεδομένων, με την πάροδο του χρόνου, αυξάνεται με έναν εκπληκτικό ρυθμό. Από μία τόση μεγάλη γκάμα πληροφορίας, πολλές φορές, οι χρήστες αυτών των δεδομένων αναζητούν μέσω αυτών **πιο εξειδικευμένες πληροφορίες**.

Για παράδειγμα, ένας διευθυντής πωλήσεων δεν είναι πια ικανοποιημένος με μία απλή λίστα από στοιχεία πελατών, αλλά θέλει λεπτομερείς πληροφορίες σχετικά με τις προηγούμενες αγορές των πελατών, καθώς επίσης και με τις προβλέψεις για τις μελλοντικές αγορές τους. Οι **απλές ερωτήσεις** που μπορούν να υποβληθούν σε μία δομημένη γλώσσα ερωτήσεων (SQL), **δεν είναι αρκετές** για να υποστηρίξουν αυτές τις αυξανόμενες απαιτήσεις για πληροφορίες.

Σε αυτό το σημείο, έρχεται να παρέμβει η **Εξόρυξη Γνώσης** από δεδομένα (**Data Mining**) προκειμένου να ικανοποιήσει αυτού του είδους τις ανάγκες. Η εξόρυξη γνώσης από δεδομένα συχνά ορίζεται σαν η εύρεση πληροφοριών που είναι κρυμμένες σε μία Βάση Δεδομένων. Με τον όρο "**κρυμμένες**", εννοούμε πως, ο άνθρωπος, θα χρειαζόταν πάρα πολύ από το χρόνο του για να μελετήσει έναν τόσο μεγάλο όγκο δεδομένων, να καταγράψει παρατηρήσεις και, ακολούθως, να βγάλει συμπεράσματα μέσα από αυτές. Με την χρήση, όμως, των κατάλληλων **αλγορίθμων** εξόρυξης γνώσης, ο χρόνος αυτός μειώνεται δραστικά και ο άνθρωπος που υπηρετεί τα συμφέρονται της επιχείρησης στην οποία ανήκει, επωφελείται από τα χρήσιμα συμπεράσματα που εξάγονται από αυτούς τους αλγόριθμους.

Μία, λοιπόν, από τις σημαντικότερες και νεότερες τεχνικές Εξόρυξης Γνώσης, είναι οι **Κανόνες Συσχέτισης**. Με βάση την τεχνική των Κανόνων Συσχέτισης, οι πληροφορίες που συγκεντρώνονται από μία Βάση Δεδομένων παράγουν ενδιαφέρουσες συσχετίσεις και πρότυπα που βρίσκουν εφαρμογή από τους τομείς της ζωής και της ενασχόλησης του ανθρώπου, μέχρι τα τηλεπικοινωνιακά δίκτυα και την αγορά και τη διαχείριση ρίσκου. Αυτό που έδωσε, όμως, μεγάλη ώθηση στους κανόνες συσχέτισης ήταν η ανάγκη κατανόησης και ανάλυσης του καλαθιού αγοράς.

Σε αυτήν την διπλωματική εργασία, αφού αποδώσουμε την έννοια και την σημαντικότητα της "**εξόρυξης γνώσης**" και παραθέσουμε τις διάφορες κατηγορίες - μεθόδους που ακολουθούνται μέχρι σήμερα, θα αναλύσουμε μέσω παραδειγμάτων ειδικότερα την εξόρυξη γνώσης με χρήση Κανόνων Συσχέτισης. Σκοπός μας είναι, μέσα από αυτά τα **παραδείγματα**, να αναδειχθεί η **ωφελιμότητα** της χρήσης των κανόνων, αναδεικνύοντας την σημαντικότητα της γνώσης που προκύπτει. Την σημαντικότητα αυτής της γνώσης θα την εξετάσουμε από την σκοπιά της βελτίωσης της λειτουργίας των **επιχειρήσεων** και του καταλυτικού της ρόλου στη **λήψη αποφάσεων** των στελεχών της.

1. Εξόρυξη Γνώσης

1.1 Εισαγωγή

Η **Εξόρυξη Γνώσης** είναι η εξεύρεση μίας πληροφορίας ή προτύπων από μεγάλες Βάσεις Δεδομένων με χρήση αλγορίθμων ομαδοποίησης ή κατηγοριοποίησης και των αρχών της Στατιστικής, της Τεχνητής Νοημοσύνης, της Μηχανικής Μάθησης και των συστημάτων Βάσεων Δεδομένων. Στόχος της, είναι η πληροφορία που θα εξαχθεί και τα πρότυπα που θα προκύψουν, να έχουν δομή κατανοητή προς τον άνθρωπο ούτως ώστε να τον βοηθήσουν να πάρει τις κατάλληλες αποφάσεις.

Η Εξόρυξη Γνώσης από δεδομένα περιλαμβάνει πολλούς διαφορετικούς αλγόριθμους για να πραγματοποιηθούν διαφορετικές εργασίες. Όλοι αυτοί οι αλγόριθμοι επιχειρούν να ταιριάξουν ένα μοντέλο στα δεδομένα. Οι αλγόριθμοι εξετάζουν τα δεδομένα και καθορίζουν ένα μοντέλο που να είναι το πλησιέστερο στα χαρακτηριστικά των δεδομένων που εξετάζονται. Οι αλγόριθμοι εξόρυξης γνώσης μπορεί να θεωρηθεί ότι αποτελούνται από τρία μέρη:

- **Μοντέλο**: Ο σκοπός του αλγορίθμου είναι να ταιριάζει το μοντέλο στα δεδομένα
- **Προτίμηση**: Πρέπει να χρησιμοποιούνται κάποια κριτήρια για να ταιριάξει ένα μοντέλο έναντι ενός άλλου
- **Αναζήτηση**: Όλοι οι αλγόριθμοι απαιτούν μία τεχνική για να κάνουν αναζήτηση στα δεδομένα

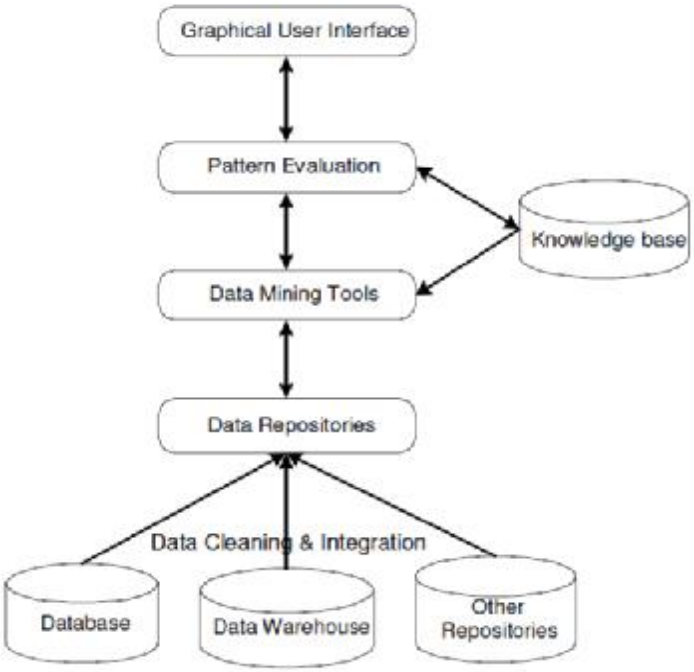
Μελετώντας μερικές από τις μεθόδους της εξόρυξης γνώσης, αναφέρουμε τις παρακάτω:

- **Κατηγοριοποίηση**: Απεικονίζει τα δεδομένα σε προκαθορισμένες κατηγορίες - κλάσεις. Αναφέρεται συχνά σαν εποπτευόμενη μάθηση, επειδή οι κατηγορίες - κλάσεις καθορίζονται πριν ακόμη εξεταστούν τα δεδομένα (π.χ. καθορισμός του αν θα δοθεί ένα τραπεζικό δάνειο).
- **Παλινδρόμηση**: Χρησιμοποιείται για να απεικονιστεί ένα στοιχειώδες δεδομένο σε μία πραγματική μεταβλητή πρόβλεψης. Στην πραγματικότητα περιλαμβάνει την εκμάθηση της συνάρτησης που κάνει αυτή την απεικόνιση. Η Παλινδρόμηση προϋποθέτει ότι τα σχετικά δεδομένα ταιριάζουν με μερικά γνωστά είδη συνάρτησης (π.χ. γραμμική, λογαριθμική κλπ) και μετά καθορίζει την καλύτερη συνάρτηση αυτού του είδους που μοντελοποιεί τα δεδομένα που έχουν δοθεί.
- **Ανάλυση Χρονοσειρών**: Εδώ, μελετάται η τιμή ενός γνωρίσματος καθώς μεταβάλλεται στο χρόνο. Οι τιμές συνήθως λαμβάνονται σε ίσα χρονικά διαστήματα. Υπάρχουν τρεις βασικές λειτουργίες που πραγματοποιούνται στην ανάλυση συμβολοσειρών: Χρησιμοποιούνται μονάδες μέτρησης

απόστασης για να καθορίσουν την ομοιότητα ανάμεσα σε διαφορετικές χρονοσειρές / Εξετάζεται η δομή της χρονοσειράς για να καθορίσει τη συμπεριφορά της / Χρησιμοποιούνται διαγράμματα χρονοσειρών για την πρόβλεψη μελλοντικών τιμών.

- **Πρόβλεψη**: Πολλές από τις πρακτικές εξόρυξης γνώσης μπορούν να θεωρηθούν σαν πρόβλεψη μελλοντικών καταστάσεων με γνώση των προηγούμενων και των σημερινών δεδομένων. Η πρόβλεψη μπορεί να θεωρηθεί σαν ένα είδος κατηγοριοποίησης. Η διαφορά είναι πως, ως πρόβλεψη, θεωρείται περισσότερο το να δίνεται τιμή σε μία μελλοντική κατάσταση, παρά σε μία τρέχουσα. Οι εφαρμογές πρόβλεψης περιλαμβάνουν πρόγνωση πλημμύρων, αναγνώριση ομιλίας, μηχανική μάθηση και αναγνώριση προτύπου.
- **Συσταδοποίηση**: Είναι παρόμοια με την κατηγοριοποίηση εκτός από το ότι οι συστάδες - ομάδες δεδομένων - δεν είναι προκαθορισμένες, αλλά ορίζονται κυρίως από τα ίδια τα δεδομένα. Η συσταδοποίηση αναφέρεται εναλλακτικά και σαν μη εποπτευόμενη μάθηση ή τμηματοποίηση. Μπορεί να θεωρηθεί σαν μία διαμέριση των δεδομένων σε ομάδες που μπορεί να είναι ή να μην είναι διακριτές μεταξύ τους. Η συσταδοποίηση, συνήθως, επιτυγχάνεται με τον καθορισμό της ομοιότητας, ως προς προκαθορισμένα γνωρίσματα ανάμεσα στα δεδομένα. Τα πιο σχετικά δεδομένα ομαδοποιούνται στις ίδιες ομάδες.
- **Παρουσίαση Συνόψεων**: Απεικονίζει τα δεδομένα σε υποσύνολά τους με συνοδευτικές απλές περιγραφές. Η σύνοψη των δεδομένων ονομάζεται επίσης και χαρακτηρισμός ή γενίκευση. Εξάγει ή παράγει αντιπροσωπευτικές πληροφορίες σχετικά με τις Βάσεις Δεδομένων, Αυτό, στην πραγματικότητα, γίνεται ανακτώντας τμήματα από τα δεδομένα. Σε γενικές γραμμές, η παρουσίαση συνόψεων χαρακτηρίζει τα περιεχόμενα της Βάσης Δεδομένων.
- **Κανόνες Συσχέτισης**: Η ανάλυση συνδέσμων (link analysis), που εναλλακτικά αναφέρεται και σαν ανάλυση συγγένειας (association), αναφέρεται στη διαδικασία εκείνη της Εξόρυξης Γνώσης που αποκαλύπτει συσχετίσεις μεταξύ των δεδομένων. Το καλύτερο παράδειγμα αυτού του είδους εφαρμογής είναι ο προσδιορισμός κανόνων συσχετίσεων. Ένας Κανόνας Συσχέτισης (**Association Rule**) είναι ένα μοντέλο που αναγνωρίζει ειδικούς τύπους συσχέτισης μεταξύ δεδομένων. Αυτές οι συσχετίσεις συχνά χρησιμοποιούνται στις λιανικές πωλήσεις για να αναγνωριστούν προϊόντα που συχνά αγοράζονται μαζί.

Σε γενικές γραμμές, η δημιουργία ενός συστήματος εξόρυξης γνώσης θα λέγαμε πως βασίζεται στην παρακάτω δενδρική δομή:



Εικόνα 1: Data Mining Processing Tree

1.2 Τεχνικές Εξόρυξης Γνώσης

1.2.1 Η Μηχανική Μάθηση - Ορισμοί

Ο τρόπος με τον οποίο προσπαθεί ο άνθρωπος να ερμηνεύσει το περιβάλλον μέσα στο οποίο ζει, πηγάζει μέσα από την παρατήρηση. Συνέχεια της παρατήρησης αποτελεί μία ανάγκη που μας χαρακτηρίζει να δημιουργούμε **μοντέλα**, δηλαδή απλοποιημένες εκδοχές των όσων παρατηρούμε. Η διαδικασία δημιουργίας αυτών των μοντέλων ονομάζεται **επαγωγική μάθηση** (inductive learning). Επιπροσθέτως, μια ιδιότητα που χαρακτηρίζει τον άνθρωπο είναι το να συσχετίζει και να οργανώνει τις εμπειρίες, τις μνήμες, τα γεγονότα και τις παραστάσεις του, καλλιεργώντας νέες δομές που ονομάζονται **Πρότυπα** (patterns). Μέσω τον Μοντέλων και των Προτύπων οδηγούμαστε στην ανακάλυψη γνώσης, στη λήψη αποφάσεων και στην εξαγωγή συμπερασμάτων.

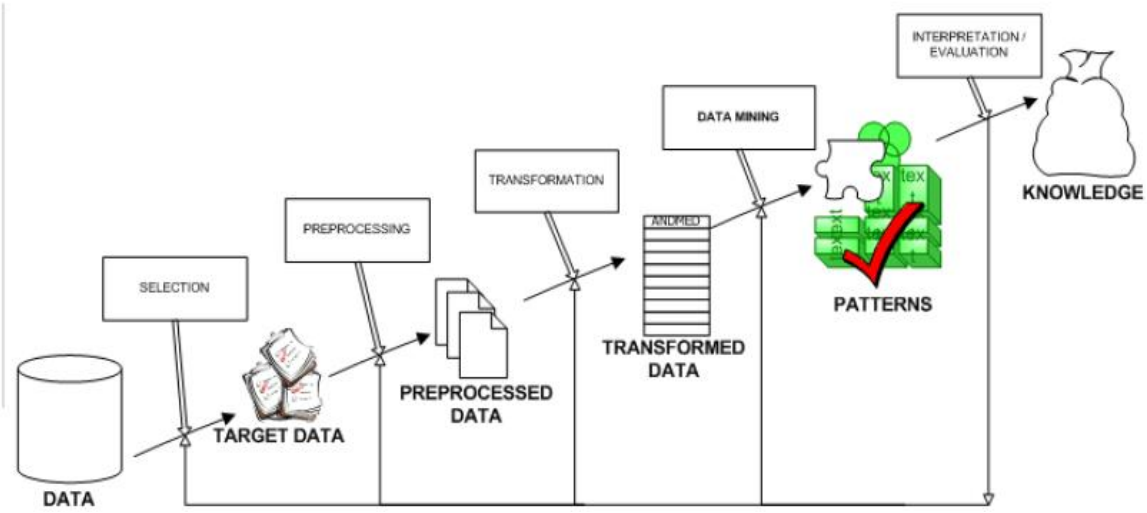
Όταν όλη η παραπάνω διαδικασία γίνεται από ένα υπολογιστικό σύστημα, τότε η διαδικασία αυτή ονομάζεται **μηχανική μάθηση** (machine learning). Έχουν αναπτυχθεί πολλές τεχνικές μηχανικής μάθησης (θα αναφερθούμε σε επόμενες ενότητες) οι οποίες αξιοποιούνται και επιλέγονται η κάθε μία ανάλογα με τη φύση του προβλήματος. Σε αυτό το σημείο κάνουμε μία πρώτη κατηγοριοποίηση στα είδη μηχανικής μάθησης και, ουσιαστικά, τις διακρίνουμε σε:

- **Μάθηση με Επίβλεψη (supervised learning)**
- **Μάθηση χωρίς Επίβλεψη (unsupervised learning)**

Στη **Μάθηση με Επίβλεψη** το υπολογιστικό σύστημα καλείται να οδηγηθεί στην ανακάλυψη γνώσης "μαθαίνοντας" μια έννοια ή συνάρτηση από ένα σύνολο δεδομένων, η οποία αποτελεί περιγραφή ενός μοντέλου. Το όνομά της προκύπτει από το γεγονός πως υπάρχει κάποιος "**επιβλέπων**" (άνθρωπος), οποίος παρέχει την κατάλληλη τιμή εξόδου της συνάρτησης, για τα δεδομένα μέσα των οποίων θέλουμε κάθε φορά να γίνει η εξόρυξη της γνώσης.

Αντιθέτως, στη **Μάθηση χωρίς Επίβλεψη** το υπολογιστικό σύστημα επιβάλλεται μόνο του να ανακαλύψει γνώση, πραγματοποιώντας **συσχετίσεις**, ομαδοποιώντας δεδομένα, δημιουργώντας πρότυπα, χωρίς, όμως, να γνωρίζει εκ των προτέρων πόσα ή ποιά είναι αυτά τα πρότυπα [1].

Κοινός παρονομαστής και για τις δύο κατηγορίες μάθησης, που αποσκοπούν στην ανακάλυψη γνώσης είναι το διαδικαστικό μοντέλο που παραθέτουμε στην παρακάτω εικόνα:



Εικόνα 2: Από τα δεδομένα στη Γνώση

Το μοντέλο αυτό αποτελείται από μια σειρά από τα ακόλουθα βήματα [2]:

1. Την *ανάπτυξη και κατανόηση* της περιοχής της εφαρμογής, της σχετικά προγενέστερης γνώσης του προς εξέταση τομέα και τους στόχους του τελικού χρήστη.
2. Την *ολοκλήρωση των δεδομένων.* Υπάρχουν διαφορετικά είδη αποθηκευτικών πληροφοριών που μπορούν να χρησιμοποιηθούν στη διαδικασία εξόρυξης γνώσης. Κατά συνέπεια, οι πολλαπλές πηγές δεδομένων μπορούν να συνδυαστούν καθορίζοντας το σύνολο στο οποίο τελικά η διαδικασία εξόρυξης πρόκειται να εφαρμοστεί.

3. Τη *δημιουργία του στόχου-συνόλου δεδομένων.* Επιλογή του συνόλου δεδομένων (δηλαδή μεταβλητές, δείγματα δεδομένων) στο οποίο η διαδικασία εξόρυξης πρόκειται να εκτελεσθεί.
4. Τον *καθορισμό και την προ-επεξεργασία δεδομένων.* Αυτό το βήμα περιλαμβάνει βασικές διαδικασίες όπως η αφαίρεση του θορύβου ή των outliers, η συλλογή των απαραίτητων πληροφοριών για τη διαμόρφωση ή τη μέτρηση του θορύβου, η απόφαση σχετικά με τις στρατηγικές διαχείρισης των ελλειπόντων πεδίων δεδομένων.
5. Την *επιλογή των στόχων και των αλγορίθμων εξόρυξης δεδομένων.* Σε αυτό το βήμα αποφασίζουμε το στόχο της διαδικασίας ανακάλυψης γνώσης, επιλέγοντας του στόχους εξόρυξης δεδομένων που θέλουμε να επιτύχουμε. Επίσης, επιλέγονται οι μέθοδοι που θα χρησιμοποιηθούν. Αυτό περιλαμβάνει την επιλογή του κατάλληλου μοντέλου και παραμέτρων (π.χ. κατηγορικό ή αριθμητικό μοντέλο δεδομένων). Επίσης η μέθοδος εξόρυξης δεδομένων πρέπει να αντιστοιχηθεί με τις απαιτήσεις και τα γενικά κριτήρια της διαδικασίας εξόρυξης γνώσης.
6. Την *εξόρυξη γνώσης.* Εφαρμόζοντας ευφυείς μεθόδους, ψάχνουμε για ενδιαφέροντα πρότυπα γνώσης. Τα πρότυπα θα μπορούσαν να είναι μιας συγκεκριμένης αντιπροσωπευτικής μορφής ή ενός συνόλου τέτοιων αντιπροσωπεύσεων, όπως κανόνες κατηγοριοποίησης (classification rules), δέντρα, παλινδρόμηση, συσταδοποίηση (clustering) κλπ. Η απόδοση και τα αποτελέσματα της μεθόδου εξόρυξης δεδομένων εξαρτώνται από τα προηγούμενα βήματα.
7. Την *αξιολόγηση των προτύπων.* Τα εξαγόμενα πρότυπα αξιολογούνται με κάποια μέτρα, προκειμένου να προσδιοριστούν τα πρότυπα τα οποία αντιπροσωπεύουν τη γνώση, δηλαδή τα αληθινά ενδιαφέροντα πρότυπα.
8. Την *σταθεροποίηση και παρουσίαση της γνώσης.* Σε αυτό το βήμα, η εξορυγμένη γνώση ενσωματώνεται στο σύστημα ή απλά την απεικόνισή μας και κάποιες τεχνικές αντιπροσώπευσης γνώσης χρησιμοποιούνται για να παρουσιάσουν την εξορυγμένη γνώση στο χρήστη. Επίσης, ελέγχουμε για επίλυση τυχών συγκρούσεων με προηγούμενη εξορυγμένη γνώση.

1.2.2 Μάθηση με Επίβλεψη

Σε αυτή την κατηγορία μάθησης, διακρίνουμε δύο είδη προβλημάτων: (α) τα προβλήματα *ταξινόμησης* και (β) τα προβλήματα *παρεμβολής*. Η ταξινόμηση (classification) έχει να κάνει με τη δημιουργία πρόβλεψης κλάσεων / κατηγοριών, όπως για παράδειγμα η ομάδα αίματος. Αντίστοιχα, η παρεμβολή έχει να κάνει με τη δημιουργία πρόβλεψης αριθμητικών τιμών.

Οι κυριότερες τεχνικές εξόρυξης γνώσης αναφορικά με την μηχανική μάθηση με επίβλεψη είναι:

- Μάθηση Εννοιών (Concept Learning)
- Δέντρα Απόφασης (Decision Trees)
- Μάθηση Κανόνων (Rule Learning)
- Μάθηση Κατά Περίπτωση (Instance Based Learning)
- Μάθηση κατά Bayes
- Γραμμική Παρεμβολή (Linear Regression)
- Νευρωνικά Δίκτυα (Neural Networks)
- Μηχανές Διανυσμάτων Υποστήριξης (Support Vector Machines, SVMs)

Παρακάτω παραθέτουμε μία περιγραφή των πιο σημαντικών τεχνικών εξόρυξης γνώσης, χρησιμοποιώντας Μηχανική Μάθηση με επίβλεψη:

Μάθηση Εννοιών

Η Μάθηση Εννοιών είναι ένα κλασικό παράδειγμα επαγωγικής μάθησης, σύμφωνα με το οποίο το υπολογιστικό σύστημα τροφοδοτείται με θετικά ή αρνητικά παραδείγματα. Όπου "*θετικό*", χαρακτηρίζουμε το παράδειγμα που ανήκει στην εκάστοτε έννοια, ενώ, αντίστοιχα, "*αρνητικό*", αυτό που δεν ανήκει. Στη συνέχεια, το σύστημα καλείται να παράγει μία γενικευμένη περιγραφή αυτής της έννοιας, κοινώς να δημιουργήσει ένα μοντέλο. Ο λόγος που το κάνει αυτό, είναι για να είναι εν συνεχεία σε θέση να αποφασίσει αν μία άγνωστη περίπτωση ανήκει - ή όχι - σε αυτήν την έννοια. Ένα χαρακτηριστικό παράδειγμα είναι ο **αλγόριθμος απαλοιφής υποψηφίων**.

Ο αλγόριθμος απαλοιφής υποψηφίων (candidate elimination algorithm) επιτελεί γενικεύσεις και εξειδικεύσεις σε κάποιες αρχικές υποθέσεις (έννοιες) με βάση τα δεδομένα εκπαίδευσης. Ο λόγος που το κάνει ο αλγόριθμος αυτό, είναι για να περιορίσει το χώρο αναζήτησης. Ο αλγόριθμος διατηρεί δύο σύνολα, G και S, που από κοινού περιγράφουν όλο το χώρο αναζήτησης και ορίζονται ως εξής:

G: το σύνολο των πιο γενικών υποψήφιων υποθέσεων

S: το σύνολο των πιο εξειδικευμένων υποψήφιων υποθέσεων

Με βάση τα θετικά ή αρνητικά παραδείγματα, ο αλγόριθμος περιορίζει το σύνολο G κάνοντας εξειδικεύσεις και επεκτείνει το σύνολο S κάνοντας γενικεύσεις. Παραθέτουμε των ψευδοκώδικα παρακάτω:

```
Αρχικοποίησε:
    Το G στο σύνολο όλων των υποθέσεων.
    Το S στο κενό σύνολο.
Για κάθε δεδομένο εκπαίδευσης x:
    Αν το x είναι θετικό:
      1. Διέγραψε τα μέλη του G που δεν ικανοποιούν το x.
      2. Για κάθε υπόθεση s∈S που δεν ικανοποιεί το x:
```

   a. **Διέγραψε** την s από το S.
   b. **Πρόσθεσε** στο S όλες τις ελάχιστες γενικεύσεις h της s, έτσι ώστε κάθε υπόθεση h να ικανοποιεί το x και να υπάρχει κάποια υπόθεση του G που να είναι πιο γενική από κάποια υπόθεση του G που να είναι πιο γενική από κάποια άλλη υπόθεση του S.
   c. **Διέγραψε** από το S όποια υπόθεση είναι πιο γενική από κάποια άλλη υπόθεση του S.

Αν το x είναι *αρνητικό*:

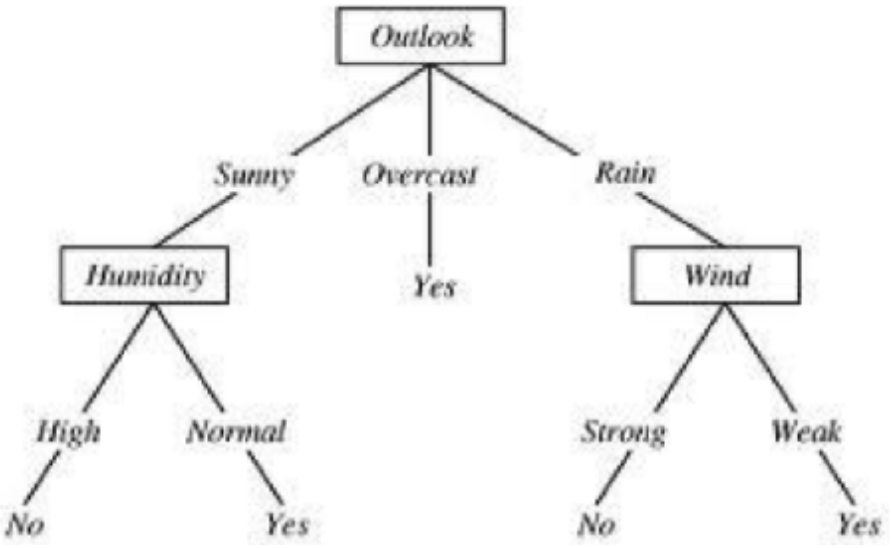
1. **Διέγραψε** τα μέλη του S που δεν ικανοποιούν το x.
2. **Για κάθε** υπόθεση g∈G που δεν ικανοποιεί το x:
   a. **Διέγραψε** την g από το G.
   b. **Πρόσθεσε** στο G όλες τις ελάχιστες ειδικεύσεις h της g, έτσι ώστε κάθε υπόθεση h να ικανοποιεί το x και να υπάρχει κάποια υπόθεση του S που να είναι πιο ειδική από κάποια άλλη υπόθεση του G.

Δέντρα Απόφασης

Οι αλγόριθμοι επαγωγής Δέντρων Απόφασης είναι οι πιο δημοφιλείς, ενώ έχουν εφαρμοστεί με πολύ ικανοποιητικά αποτελέσματα σε διάφορους τομείς. Χαρακτηριστικά παραδείγματα αποτελούν τα έμπειρα συστήματα διάγνωσης στον κλάδο της ιατρικής, όπου οι αλγόριθμοι που περιλαμβάνουν, περιέχουν κανόνες που προκύπτουν από Δέντρα Απόφασης. Επιπροσθέτως, άλλο παράδειγμα, αποτελούν οι αλγόριθμοι απογείωσης - προσγείωσης - πτήσης των αεροπλάνων.

Πρόκειται για μία μέθοδο προσέγγισης συναρτήσεων που έχουν ως στόχο διακριτές τιμές. Τα αποτελέσματά τους είναι μία **δενδροειδής δομή** που με απλό και κατανοητό τρόπο περιγράφει τα δεδομένα από τα οποία εξάγονται οι κανόνες. Κάθε κόμβος στο δέντρο ορίζει μία συνθήκη ελέγχου της τιμής κάποιου χαρακτηριστικού (attribute) των περιπτώσεων (instances) και κάθε κλαδί που φεύγει από τον κόμβο αυτό, αντιστοιχεί σε μια διαφορετική διακριτή τιμή του χαρακτηριστικού αυτού.

Η αναπαράσταση των Δέντρων Απόφασης είναι μία **διάζευξη** που αποτελείται από συζεύξεις περιορισμών στις τιμές των χαρακτηριστικών. Κάθε **μονοπάτι** από τη ρίζα προς ένα **φύλλο** αντιστοιχεί σε συζεύξεις περιορισμών στις τιμές των χαρακτηριστικών. Το δέντρο συνολικά εκφράζει τη διάζευξη αυτών των συζεύξεων, αφού απαρτίζεται από όλα τα πιθανά - εναλλακτικά μονοπάτια. Το σημαντικότερο πλεονέκτημα των Δέντρων Απόφασης είναι η ευκολία με την οποία ερμηνεύονται. Παρακάτω παρατίθεται ένα δείγμα Δέντρου Απόφασης, σχετικά με την απόφαση που πρέπει να παρθεί ή όχι για το αν θα πάμε περίπατο.



Εικόνα 3: Παράδειγμα Δέντρου Απόφασης

Μάθηση κατά Bayes

Στη μάθηση κατά Bayes (Bayesian learning), κάθε παράδειγμα εκπαίδευσης του υπολογιστικού συστήματος, μπορεί (σταδιακά πάντα) να αυξήσει ή να μειώσει την πιθανότητα να είναι σωστή μία υπόθεση. Κάτι τέτοιο δίνει ιδιαίτερα μεγάλη ευελιξία στους σχετικούς αλγόριθμους αφού δεν απορρίπτουν αμέσως μία υπόθεση σε περίπτωση που προκύψει ότι δεν είναι σε πλήρη συμφωνία με τα παραδείγματα εκπαίδευσης. Επιπροσθέτως, ήδη υπάρχουσα γνώση μπορεί να συνδυαστεί με τα δεδομένα εκπαίδευσης με τη μορφή αρχικών τιμών πιθανότητας για τις "υπό εξέταση" υποθέσεις. Εκτός, όμως, από τη φάση της εκπαίδευσης, η μάθηση κατά Bayes δίνει ευελιξία και στο στάδιο της εφαρμογής της γνώσης που προκύπτει.

Μία δυσκολία, σε πρακτικό επίπεδο, κατά την εφαρμογή αυτής της τεχνικής εξόρυξης γνώσης, αποτελεί η *απαίτηση για τη γνώση πολλών τιμών πιθανοτήτων*. Συγκεκριμένα, όταν αυτές οι τιμές δεν είναι δυνατό να υπολογιστούν με ακρίβεια, τότε υπολογίζονται κατ' εκτίμηση με βάση παλαιότερες υποθέσεις, εμπειρική γνώση κ.α.

Νευρωνικά Δίκτυα

Τα Νευρωνικά Δίκτυα (neural networks) παρέχουν έναν πρακτικό, πολύ απλό τρόπο για την εκμάθηση αριθμητικών και διανυσματικών συναρτήσεων σε διακριτά ή συνεχή μεγέθη. Χρησιμοποιούνται τόσο για παρεμβολή (γραμμική και μη γραμμική), όσο και για ταξινόμηση. Μεγάλο τους πλεονέκτημα αποτελεί η ανοχή που παρουσιάζουν σε δεδομένα εκπαίδευσης με θόρυβο, δηλαδή δεδομένα που περιστασιακά έχουν λανθασμένες τιμές, όπως για παράδειγμα λάθη καταχώρησης. Το μειονέκτημά τους έγκειται στο ότι αδυνατούν να ερμηνεύσουν ποιοτικά την γνώση την οποία μοντελοποιούν.

Μηχανές Διανυσμάτων Υποστήριξης

Οι Μηχανες Διανυσμάτων Υποστήριξης ή ΜΔΥ (Support Vector Machines, SVMs) προτάθηκαν από τον Vladimir Vapnik και τους συνεργάτες του το 1992 ως μία νέα μέθοδος μάθησης. Τα τελευταία χρόνια έχουν καθιερωθεί ως μία από τις πιο διαδεδομένες μεθόδους παρεμβολής (γραμμικής και μη) και ταξινόμησης, αποτελώντας συνήθως τη βέλτιστη επιλογή για εφαρμογές όπως η αναγνώριση γραφής (handwriting recognition), η ταξινόμηση κειμένων (text categorization) και η ταξινόμηση δεδομένων έκφρασης γονιδίων (gene expression data) [1].

1.2.3 Μάθηση χωρίς Επίβλεψη

Σε αυτήν την κατηγορία "Μάθησης", το υπολογιστικό σύστημα **καλείται να ανακαλύψει συσχετίσεις και ομάδες από τα δεδομένα**, βασιζόμενο αποκλειστικά και μόνο στις ιδιότητές τους. Αυτό έχει ως αποτέλεσμα να προκύψουν πρότυπα (περιγραφές), όπου το κάθε πρότυπο περιγράφει ένα μέρος από τα δεδομένα.

Τέτοια παραδείγματα προτύπων πληροφόρησης είναι και οι **Κανόνες Συσχέτισης** (Association Rules) που αποτελεί τον βασικό πυλώνα της θεματολογίας αυτής της διπλωματικής εργασίας. Άλλο παράδειγμα τέτοιου τύπου προτύπων είναι οι ομάδες (clusters), οι οποίες προκύπτουν από τη διαδικασία της ομαδοποίησης (clustering).

Παραθέτουμε παρακάτω μία συνοπτική περιγραφή της μεθόδου **clustering** και των αντίστοιχων αλγορίθμων, μια και με τους Κανόνες Συσχέτισης θα ασχοληθούμε αναλυτικότερα σε όλα τα υπόλοιπα Κεφάλαια:

Ομάδες (Clusters)

Πρόκειται για πρότυπα πληροφόρησης τα οποία προκύπτουν από την διαδικασία της ομαδοποίησης (clustering). Με τον όρο "**ομαδοποίηση**", εδώ, αναφερόμαστε στο διαχωρισμό ενός συνόλου δεδομένων (συνήθως πολυδιάστατων) σε ομάδες κατά τέτοιον τρόπο ώστε τα σημεία που ανήκουν στην ίδια ομάδα να "μοιάζουν" όσο το δυνατόν περισσότερο, ενώ, αντίστοιχα, τα σημεία που ανήκουν σε διαφορετικές ομάδες να διαφέρουν όσο το δυνατόν περισσότερο. Υπάρχουν τρεις γενικές κατηγορίες αλγόριθμων ομαδοποίησης:

- Οι αλγόριθμοι που είναι βασισμένοι σε *διαχωρισμούς* (partition based). Αυτοί οι αλγόριθμοι προσπαθούν να βρουν τον καλύτερο δυνατό διαχωρισμό ενός συνόλου δεδομένων από ένα συγκεκριμένο αριθμό ομάδων (π.χ. αλγόριθμος K-means)
- Οι *ιεραρχικοί* αλγόριθμοι (hierarchical). Αυτοί οι αλγόριθμοι προσπαθούν, με ιεραρχικό πάντα τρόπο, να ανακαλύψουν τον αριθμό και τη δομή των ομάδων (π.χ. Απόσταση Μανχάταν)

- Οι *πιθανοκρατικοί* αλγόριθμοι (probabilistic). Αυτοί οι αλγόριθμοι βασίζονται σε μοντέλα πιθανοτήτων [1].

1.2.4 Άλλα είδη Μάθησης

Εκτός από τις μεθόδους εξόρυξης γνώσης μέσα από τη Μηχανική Μάθηση (με ή χωρίς επίβλεψη) που είδη παραθέσαμε, υπάρχουν και άλλες προσεγγίσεις. Δύο από αυτές είναι η **Ενισχυτική Μάθηση** και οι **Γενετικοί Αλγόριθμοι**.

Ενισχυτική Μάθηση

Η ενισχυτική μάθηση (reinforcement learning) είναι ένας γενικός όρος ο οποίος έχει αποδοθεί σε μία οικογένεια τεχνικών βάσει των οποίων το υπολογιστικό σύστημα μάθησης προσπαθεί να "μάθει" μέσα από την άμεση **αλληλεπίδραση** με το περιβάλλον. Πεδία εφαρμογής αποτελούν ο έλεγχος κίνησης των ρομπότ, η βελτιστοποίηση εργασιών σε εργοστάσια, η μάθηση επιτραπέζιων παιχνιδιών κτλ.

Η ενισχυτική μάθηση διαφοροποιείται από τη μάθηση με επίβλεψη η οποία πραγματοποιείται με παραδείγματα για τα οποία, όπως ήδη έχουμε αναφέρει, για κάθε είσοδο είναι γνωστή η επιθυμητή έξοδος. Μπορεί, μεν, αυτό το είδος μάθησης να είναι πολύ σημαντικό, όμως από μόνο του δεν επαρκεί για αλληλεπιδραστική μάθηση. Σε διαδραστικά παραδείγματα είναι συνήθως δύσκολο να εξαχθούν παραδείγματα επιθυμητής συμπεριφοράς τα οποία να είναι κατάλληλα και να αναπαριστούν όλες τις καταστάσεις στις οποίες το σύστημα μάθησης θα πρέπει να δράσει.

Γενετικοί Αλγόριθμοι

Πρόκειται για μία μέθοδο μάθησης που βασίζεται στην προσομοίωση του φυσικού φαινομένου της **εξέλιξης** (evolution). Η αναπαράσταση των υποθέσεων γίνεται συνήθως από ακολουθίες bit (bit strings), αν και υπάρχουν περιπτώσεις στις οποίες χρησιμοποιούνται συμβολικές αναπαραστάσεις. Η αναζήτηση της κατάλληλης υπόθεσης ξεκινά τυχαία με έναν πληθυσμό αρχικών υποθέσεων, τα μέλη του οποίου παράγουν τη "**νέα γενιά**" μέσω διαδικασιών αναπαραγωγής όπως η διασταύρωση (crossover) και η τυχαία μετάλλαξη (random mutation) που είναι αντίστοιχες των πραγματικών διαδικασιών στη βιολογική εξέλιξη. Σε κάθε βήμα, οι υποθέσεις του τρέχοντος πληθυσμού αξιολογούνται με βάση μια προκαθορισμένη συνάρτηση καταλληλότητας (fitness function) και με βάση αυτήν την αξιολόγηση που δίνει η συνάρτηση, επιλέγονται ή απορρίπτονται αντίστοιχα από την επόμενη γενιά [1].

2. Κανόνες Συσχέτισης

2.1 Κανόνες Συσχέτισης - Ορισμοί

Η ανακάλυψη ή εξόρυξη κανόνων συσχέτισης (association rule mining) προτάθηκε στις αρχές της δεκαετίας του '90 από τον Rakesh Agrawal ως τεχνική ανάλυσης **καλαθιού αγορών** (market basket analysis), όπου το ζητούμενο είναι η ανακάλυψη συσχετίσεων ανάμεσα στα αντικείμενα μίας Βάσης Δεδομένων.

Οι κανόνες συσχέτισης εκφράζουν το αποτέλεσμα της ανάλυσης χιλιάδων καλαθιών αγοράς πελατών. Ένας τέτοιος κανόνας είναι ο εξής: «Οι πελάτες που αγοράζουν γάλα, αγοράζουν παράλληλα και ψωμί σε ποσοστό 60%». Ο παραπάνω κανόνας γράφεται σύντομα ως «γάλα g ψωμί». Η πρόταση αυτή παρουσιάζει ένα αίτιο, αγορά γάλακτος, και το συνδέει με ένα αποτέλεσμα, αγορά ψωμιού. Επίσης παρέχει μια ένδειξη για το πόσο πιθανό είναι να συμβαίνει μια τέτοια σχέση αιτίας-αιτιατού μέσω του ποσοστού που δίνεται. Οι κανόνες συσχέτισης επομένως, όπως υποδηλώνει το όνομά τους, είναι κανόνες «if-then» που συσχετίζουν αντικείμενα μεταξύ τους.

Έστω $I=\{i_1,i_2, \ldots, i_m\}$ ένα σύνολο από διακριτά κατηγορήματα που αποκαλούμε items (αντικείμενα). Έστω επίσης D ένα σύνολο από δοσοληψίες (transactions), όπου κάθε δοσοληψία (transaction) T είναι ένα σύνολο από αντικείμενα, το οποίο καλείται itemset, και για το οποίο ισχύει $T \subseteq I$. Κάθε δοσοληψία χαρακτηρίζεται από ένα μοναδικό αναγνωριστικό που καλείται TID. Θα λέγαμε ότι μία δοσοληψία T περιέχει το X, ένα σύνολο από κάποια αντικείμενα του I, αν ισχύει $X \subseteq T$.

Κανόνας Συσχέτισης (Association rule), σύμφωνα με το βιβλίο των Μ. Χαλκίδης, Μ. Βαζιργιάννης (2005) *"Εξόρυξη από Βάσεις Δεδομένων και τον Παγκόσμιο Ιστό",* είναι μια συσχέτιση της μορφής $X \Rightarrow Y$, όπου $X \subseteq I$, $Y \subseteq I$ και $X \cap Y = \emptyset$. Το πρώτο μέλος του κανόνα θα το αποκαλούμε *υπόθεση* ενώ το δεύτερο *συμπέρασμα.* Ο $X \Rightarrow Y$ ισχύει στο σύνολο των δοσοληψιών D με εμπιστοσύνη (confidence) c, αν το c% των δοσοληψιών στο D περιέχουν το X περιέχουν επίσης και το Y. Ο κανόνας $X \Rightarrow Y$ έχει *υποστήριξη (support)* s, αν το s% των δοσοληψιών στο D περιέχουν το $X \cup Y$.

Ο ορισμός που δόθηκε για την υποστήριξη μπορεί να γενικευτεί και ένα itemset (σύνολο από items). Επομένως ένα itemset X θα λέμε ότι έχει υποστήριξη s και θα γράφουμε sup(X) = s, αν το s% των δοσοληψιών στο D περιέχουν το X. Ακόμα, ένα itemset X θα λέμε ότι έχει μήκος k και θα το αποκαλούμε k-itemset όταν αυτό αποτελείται από k το πλήθος items, δηλαδή $|X|= k$.

Με βάση τον παραπάνω ορισμό για την υποστήριξη ενός itemset μπορούμε να δώσουμε έναν ισοδύναμο ορισμό για την υποστήριξη και εμπιστοσύνη ενός κανόνα συσχέτισης.

Ο κανόνας X⇒Y έχει υποστήριξη s, όταν sup(X⇒Y) = sup(X∪Y) και εμπιστοσύνη c, όταν conf(X⇒Y) = $\frac{\sup(X \cup Y)}{\sup(X)}$= c.

Αφού η υποστήριξη s ενός itemset X έχει οριστεί ως ποσοστό με το οποίο συναντάται το συγκεκριμένο itemset στο σύνολο των δοσοληψιών, μπορούμε να αντιστοιχήσουμε την υποστήριξη στην πιθανότητα με την οποία μια δοσοληψία δύναται να περιέχει το itemset αυτό. Έστω P(X) η πιθανότητα με την οποία το itemset X περιέχεται σε μια οποιαδήποτε δοσοληψία, τότε προφανώς θα ισχύει sup(X)=P(X).

Αντίστοιχα για έναν κανόνα συσχέτισης X⇒Y η υποστήριξη μπορεί θεωρηθεί ως η πιθανότητα μια δοσοληψία να περιέχει το X∪Y, δηλαδή sup(X⇒Y) = P(X∪Y). Η εμπιστοσύνη του κανόνα είναι η δεσμευμένη πιθανότητα με την οποία μια δοσοληψία που περιέχει το X να περιέχει και το Y, δηλαδή conf(X⇒Y) = P(X|Y).

Από τους ορισμούς που δώσαμε για την υποστήριξη και την εμπιστοσύνη, γίνεται αντιληπτό ότι το πρώτο μέτρο μας δείχνει πόσο συχνά εμφανίζονται τα itemsets του κανόνα, ενώ το δεύτερο μέτρο μας δίνει την ισχύ συνεπαγωγής του κανόνα. Είναι προφανές ότι οι κανόνες X⇒Y και Y⇒X έχουν την ίδια υποστήριξη, αλλά ο κανόνας με τη μεγαλύτερη εμπιστοσύνη είναι και ο πιο χρήσιμος καθώς η σχέση μεταξύ αιτίας και αποτελέσματος είναι πιο ισχυρή [2].

Έστω το παρακάτω πρόβλημα, όπου υπάρχει ένας μεγάλος αριθμός αντικειμένων (items), για παράδειγμα ψωμί, γάλα, κτλ. Οι πελάτες γεμίζουν τα καλάθια τους με κάποιο υποσύνολο αυτών των αντικειμένων και το ζητούμενο είναι να βρεθεί ποια από αυτά τα αντικείμενα αγοράζονται μαζί, χωρίς να μας ενδιαφέρει ποιός είναι ο αγοραστής.

Οι κανόνες συσχέτισης αποτελούν προτάσεις της μορφής {X_1,..., X_n} g Y, που σημαίνει ότι αν βρεθούν όλα τα X_1,..., X_n στο καλάθι, τότε είναι πιθανόν να βρεθεί και το Y. Για παράδειγμα, ένας τέτοιος κανόνας θα μπορούσε να λέει:

"όποιος αγοράζει καφέ (X_1) και ζάχαρη (X_2) αγοράζει και αναψυκτικά (Y)"

Βέβαια, μια απλή αναφορά ενός τέτοιου κανόνα δεν έχει και μεγάλη αξία αν δεν συνοδεύεται από κάποια ποσοτικά μεγέθη που μετρούν την ποιότητα των ευρεθέντων κανόνων συσχέτισης. Τέτοια μεγέθη είναι η υποστήριξη (support) και η εμπιστοσύνη (confidence) που ορίζονται ως εξής:

- **Υποστήριξη** (support): εκφράζει την πιθανότητα να βρεθεί το καλάθι {X_1...X_n, Y} στη Βάση Δεδομένων και ισούται με τον λόγο των εγγραφών που περιλαμβάνουν το {X_1...X_n, Y} προς το σύνολο των εγγραφών.
- **Εμπιστοσύνη** (confidence): εκφράζει την πιθανότητα να βρεθεί το Y σε ένα καλάθι που περιέχει τα {X_1,...X_n} και ισούται με το λόγο των εγγραφών που περιλαμβάνουν το {X_1...X_n, Y} προς το σύνολο των εγγραφών που περιλαμβάνουν τα X_i.

Η μεγαλύτερη δυσκολία στην ανακάλυψη κανόνων συσχέτισης είναι ο μεγάλος αριθμός τέτοιων κανόνων που θεωρητικά υφίστανται σε μία Βάση Δεδομένων και η επιλογή εκείνων που έχουν πρακτική αξία. Αυτό συνήθως γίνεται θέτοντας κάποιο κάτω όριο στις τιμές των μεγεθών εμπιστοσύνη και υποστήριξη.

Παράδειγμα:

Έστω τρία σύνολα αντικειμένων (item sets) ή καλάθια αγορών τα οποία αγοράστηκαν από τρεις πελάτες [1]. Οι αριθμοί εκφράζουν κάποιο συγκεκριμένο είδος (π.χ. 1=καφές, 2=ζάχαρη κτλ).

Καλάθι	**Προϊόντα**			
#1	1	2	3	4
#2	1	2	4	5
#3	1	2	5	

Πίνακας 1: **Πίνακας Καλαθιών Αγορών**

Από τα δεδομένα του παραπάνω πίνακα προκύπτουν κανόνες όπως ο:

{1} g {2} με 100% υποστήριξη και 100% εμπιστοσύνη

που ερμηνεύεται ως "όποιος αγοράζει το είδος 1 (καφέ) αγοράζει και το είδος 2 (ζάχαρη). Επιπλέον κανόνες που μπορεί να προκύψουν, είναι οι:

{3} g {4} με 33% υποστήριξη και 100% εμπιστοσύνη

{2} g {3} με 33% υποστήριξη και 33% εμπιστοσύνη

{1,2} g {4} με 66% υποστήριξη και 66% εμπιστοσύνη

{1} g {2,4} με 66% υποστήριξη και 66% εμπιστοσύνη

κτλ.

Πλεονεκτήματα των Κανόνων Συσχέτισης ως τεχνική Μάθησης:

- Η διαδικασία ανακάλυψης / μάθησης των κανόνων συσχέτισης είναι "εξερευνητική", αφού δεν ελέγχεται μία προκαθορισμένη σχέση / υπόθεση για την εγκυρότητά της, αλλά επιχειρείται η ανακάλυψη νέων πρωτότυπων συσχετίσεων
- Οι κανόνες που προκύπτουν είναι της μορφής "if - then" (αν αγοράσει ο πελάτης το Χ τότε θα αγοράσει και το Υ) κάτι που τους κάνει ιδιαίτερα κατανοητούς σε σχέση με άλλες μορφές αναπαράστασης γνώσης
- Λόγω της προέλευσής τους, οι κανόνες συσχέτισης προσπαθούν να ανταπεξέλθουν σε πραγματικά προβλήματα, όπως ο θόρυβος και το μεγάλο μέγεθος των δεδομένων, σε αντίθεση με παραδοσιακές τεχνικές μάθησης, όπου η εκπαίδευση είναι σημαντικό να γίνεται σε "καθαρά" (χωρίς θόρυβο) και προσεκτικά επιλεγμένα δεδομένα

- Υπάρχει μία ποικιλία αλγορίθμων μάθησης κανόνων συσχέτισης που, ανάλογα με την εφαρμογή, επιτυγχάνουν υψηλή απόδοση σε τομείς όπως η ταχύτητα σε πολύ μεγάλο όγκο δεδομένων, η επεκτασιμότητα, η αυξητικότητα, ο χειρισμός σύνθετων τύπων δεδομένων, όπως χωρικά, χρονικά, πολυμεσικά κτλ.

Το βασικό μειονέκτημα των κανόνων συσχέτισης είναι η δυσκολία στην επιλογή εκείνων το κανόνων που παρουσιάζουν το μεγαλύτερο "ενδιαφέρον", από ένα πολύ μεγαλύτερο σύνολο κανόνων. Συχνά, οι κανόνες περιγράφουν συσχετίσεις ήδη γνωστές, όπως για παράδειγμα:

if περιστατικό = έγκυος *then* φύλο = γυναίκα

ή παράλογες, όπως για παράδειγμα:

if όνομα = Κώστας *then* εισόδημα = χαμηλό

Ως εκ τούτου, οι μετρικές της υποστήριξης και της εμπιστοσύνης δεν είναι συνήθως αρκετές ώστε να ελαττωθεί ικανοποιητικά το σύνολο των κανόνων συσχέτισης, αφήνοντας μόνο τους έγκυρους, πρωτότυπους και ενδιαφέροντες κανόνες, χωρίς τον κίνδυνο να χαθούν κάποιοι σημαντικοί ή να προκύψει ένα τεράστιο σύνολο ασήμαντων κανόνων.

Παρόλο που έχουν προταθεί κάποιες ιδέες για να ξεπεραστεί το πρόβλημα αυτό, παραμένει σημαντικός ο ρόλος του ανθρώπινου παράγοντα, που πρέπει να ορίσει ο άνθρωπος κατάλληλα τα μεγέθη της υποστήριξης και της εμπιστοσύνης και τελικά να επιλέξει ποιοι κανόνες είναι πραγματικά χρήσιμοι.

Για την ανακάλυψη κανόνων συσχέτισης χρησιμοποιείται η ιδιότητα της μονοτονίας (monotonicity property) ή αλλιώς ιδιότητα **a priori** σύμφωνα με την οποία: "*Αν ένα σύνολο αντικειμένων S είναι συχνό, τότε όλα τα υποσύνολα του S είναι επίσης συχνά*". Συχνό είναι ένα σύνολο αντικειμένων όταν εμφανίζεται σε ποσοστό καλαθιών ίσο ή μεγαλύτερο από ένα όριο που συνήθως ορίζει ο χρήστης.

Σε έναν αλγόριθμο εύρεσης κανόνων συσχέτισης μας ενδιαφέρει κυρίως ο αριθμός των περασμάτων στα δεδομένα που απαιτείται κατά την εκτέλεσή του. Θεωρώντας ότι τα δεδομένα δεν χωράνε στην κύρια μνήμη, το σημαντικότερο κόστος είναι ο αριθμός των προσπελάσεων στο δίσκο. Υπάρχουν δύο βασικές μέθοδοι (οικογένειες αλγορίθμων) για την ανακάλυψη κανόνων συσχέτισης:

- Στην πρώτη μέθοδο, βρίσκονται αρχικά όλα τα συχνά αντικείμενα, κατόπιν τα συχνά ζεύγη, οι συχνές τριάδες κτλ., μέχρι να βρεθούν τα μέγιστα συχνά σύνολα αντικειμένων (maximal frequent itemsets) δηλαδή τα σύνολα S εκείνα των οποίων κανένα υπερσύνολο δεν είναι συχνό. Η εύρεση συχνών ζευγαριών είναι σημαντικό στάδιο του αλγορίθμου γιατί σε πολλά σύνολα δεδομένων αποτελεί το δυσκολότερο κομμάτι, ενώ στη συνέχεια η ανακάλυψη

τριάδων, τετράδων κ.ο.κ. απαιτεί λιγότερο χρόνο. Επιπλέον, σε πολλά προβλήματα το ζητούμενο είναι η εύρεση μόνο των συχνών δυάδων, δηλαδή συσχετίσεων ανάμεσα σε δύο αντικείμενα. Αντιπροσωπευτικό παράδειγμα, που θα αναλύσουμε εκτενέστερα στη συνέχεια της διπλωματικής εργασίας, είναι ο αλγόριθμος **Apriori**.

- Στη δεύτερη μέθοδο, βρίσκονται όλα τα μέγιστα συχνά σύνολα αντικειμένων, με ένα ή ελάχιστα περάσματα. Αυτές οι μέθοδοι έχουν συνήθως μεγάλες απαιτήσεις σε μνήμη, λόγω του ότι βασίζονται σε πολύπλοκες (συνήθως δενδροειδείς) δομές δεδομένων για να αποθηκεύσουν πληροφορίες για τα δεδομένα. Αντιπροσωπευτικό παράδειγμα είναι ο αλγόριθμος **FP – Growth** [3].

2.2 Κανόνες Συσχέτισης - "Το Καλάθι Αγοράς"

Τι είναι, όμως, ένα **καλάθι αγοράς** και ποιο το νόημα να το αναλύσουμε; Για να απαντήσουμε σε αυτό το ερώτημα θα χρησιμοποιήσουμε ένα παράδειγμα: Έστω ότι έχουμε αναλάβει την διοίκηση μίας επιχείρησης σούπερ μάρκετ. Θέλοντας να κατανοήσουμε την αγοραστική συμπεριφορά των πελατών μας, καταγράφουμε σε μία λίστα, την οποία μετέπειτα θα αποκαλούμε "Βάση Γνώσης", το προϊόντα που ψωνίζει ο εκάστοτε πελάτης σε κάθε του επίσκεψη για έναν πεπερασμένο αριθμό επισκέψεων. Έστω ο παρακάτω πίνακας, ο οποίος απεικονίζει τα αγορασμένα προϊόντα ενός πελάτη μας μέσα σε ένα χρονικό διάστημα πέντε ημερών:

Μέρα	Αγορασμένα Προϊόντα
1η	Ψωμί, Γάλα
2η	Ψωμί, Ζάχαρη, Μπύρα, Αυγά
3η	Γάλα, Ζάχαρη, Μπύρα, Κόκα Κόλα
4η	Ψωμί, Γάλα, Ζάχαρη, Μπύρα
5η	Ψωμί, Γάλα, Ζάχαρη, Κόκα Κόλα

Πίνακας 2: Αγορές πελάτη πέντε ημερών

Σκοπός μας είναι ο σχεδιασμός ενός συστήματος, που, με βάση την παραπάνω γνώση θα μας οδηγήσει στα παρακάτω επωφελή συμπεράσματα:

1. Ποια είναι η **αγοραστική συμπεριφορά** του πελάτη. Εδώ εντάσσεται η καταγραφή των προϊόντων που ψωνίζει πιο συχνά, η συχνότητα με την οποία τα ψωνίζει, ο τρόπος με τον οποίο συνδυάζει την αγορά ενός προϊόντος με κάποιο άλλο.
2. Σε ποια προϊόντα θα πρέπει να δοθεί μεγαλύτερη έμφαση ως προς την **προώθησή** τους.
3. Διαχείριση **αποθεμάτων** και αποθήκης. Όταν μπορούμε να προσεγγίσουμε σε ικανοποιητικό βαθμό την συχνότητα με την οποία αγοράζεται ένα προϊόν,

μπορούμε με μεγαλύτερη ασφάλεια και λιγότερη ζημιά, να εκτιμήσουμε την ποσότητά του που πρέπει να έχουμε ως απόθεμα.

4. Διαχείριση των **πελατειακών σχέσεων**. Ο συνδυασμός αγοράς ενός προϊόντος με την αγορά κάποιου άλλου, μας επιτρέπει, εκτιμώντας την επαναληψιμότητα αυτού του συνδυασμού, να βάλουμε, ενδεχομένως σε μια προσφορά τον συνδυασμό αγοράς τους, βελτιώνοντας, έτσι, την σχέση μας με τους πελάτες, δεδομένου ότι έτσι αισθάνονται ότι κατανοούμε καλύτερα τις ανάγκες τους.

Ο παραπάνω πίνακας, δεν είναι παρά ένα πάρα πολύ μικρό δείγμα ενός πολύ μεγαλύτερου πίνακα ο οποίος περιλαμβάνει χιλιάδες πελάτες για πολύ μεγαλύτερο διάστημα ημερών. Και αν αυτός ο πολύ μικρός πίνακας, θέλει αρκετό χρόνο για να αναλυθεί και να οδηγηθούμε σε συμπεράσματα, φανταστείτε πόσο χρόνο απαιτεί ο τεράστιος πίνακας όπου έχουν αποθηκευτεί όλοι οι πελάτες του σούπερ μάρκετ για ένα χρονικό διάστημα 30 ημερών π.χ. και όχι 5! Η ανάλυση του δεύτερου πίνακα από τον άνθρωπο, απαιτεί πάρα πολύ χρόνο και ο χρόνος στον επιχειρηματικό τομέα συνεπάγεται χρήμα. Για τον λόγο αυτό, οδηγούμαστε στην ανάγκη δημιουργίας συστημάτων που, ταχύτητα και αξιόπιστα θα οδηγούνται στην εξόρυξη αυτής της γνώσης, θα μπορούν να δημιουργούν κανόνες συσχέτισης, λήψης αποφάσεων, αλλά και εξαγωγής συμπερασμάτων [1][3].

2.3 Αγοραστική Συμπεριφορά των Πελατών (Χρήση της a priori ιδιότητας)

Υποθέτουμε πως σε δέκα καλάθια αγορών από το super market τα προϊόντα που συναντάμε ποιο συχνά είναι τα εξής:

- Ψωμί
- Καφές
- Γάλα
- Ζάχαρη

Το πρόβλημα που έχουμε να αντιμετωπίσουμε, λοιπόν, έχει να κάνει αφενός με τη **συχνότητα** που συναντάμε το κάθε προϊόν στο σύνολο των δέκα καλαθιών που εξετάσαμε και αφετέρου στον τρόπο που μπορεί να **συνδυάζονται** αυτά τα προϊόντα. Για παράδειγμα, θέσαμε υποθετικά ερωτήματα του τύπου "Άραγε όταν ένας πελάτης αγοράσει καφέ, αγοράζει και ζάχαρη;", "Αν, ναι, πόσο συχνά παρατηρείται αυτό το φαινόμενο;". Απαντήσεις σε τέτοιου είδους ερωτήματα, μας οδηγούν σε ιδιαιτέρως επωφελή συμπεράσματα σχετικά με την αγοραστική συμπεριφορά των πελατών για τους παρακάτω λόγους:

- Όσο πιο **συχνά** αγοράζεται ένα προϊόν, τόσο μεγαλύτερη - φροντίζει η επιχείρηση - να είναι η διαθεσιμότητά του.

- Ø Η γνώση πως συχνά ένα προϊόν αγοράζεται μαζί με κάποιο άλλο, υποδεικνύει στην επιχείρηση πως επιβάλλεται να εξασφαλίσει ικανοποιητικά **αποθεματικά** και για το δεύτερο προϊόν.
- Ø Η γνώση πως συχνά ένα προϊόν αγοράζεται μαζί με κάποιο άλλο, πιθανόν να οδηγήσει την επιχείρηση σε ένα "**πακέτο προσφοράς**" των δύο προϊόντων και με αυτόν τον τρόπο αποδεικνύει στον πελάτη πως κατανοεί τις ανάγκες του και φροντίζει για αυτές.

Συνεπώς, το πρόβλημά μας σε αυτήν την περίπτωση είναι η εξαγωγή Κανόνων Συσχέτισης μεταξύ αυτών των τεσσάρων προϊόντων που αναφέραμε. Για την προστασία των δεδομένων μέσω του "κλαδέματος" (pruning) κάποιου συνόλου κανόνων, θα ξεκινήσουμε την διαδικασία εξαγωγής των Κανόνων Συσχέτισης, θέτοντας συγκεκριμένα όρια για τα δύο βασικά μεγέθη: (α) **Εμπιστοσύνη** και (β) **Υποστήριξη**. Αυτό, όπως είναι λογικό, θα μας οδηγήσει στην αποδοχή κάποιων κανόνων και στην απόρριψη κάποιων άλλων.

Ιδανικός αλγόριθμος για να ξεκινήσουμε την διαδικασία εξαγωγής των κανόνων, είναι ο αλγόριθμος **Apriori** [1][3].

3. Τεχνικές Εύρεσης Κανόνων Συσχέτισης

3.1 Ο αλγόριθμος Apriori

Όπως συμπεράναμε από το Κεφάλαιο 2, η αγορά ενός προϊόντος όταν αγοράζεται μαζί και ένα άλλο προϊόν, αντιπροσωπεύει έναν κανόνα συσχέτισης. Στο παράδειγμα - πρόβλημα που μελετήσαμε και που θα αναλύσουμε παρακάτω (βλ. σελίδα **32**) σχετικά με την αγοραστική συμπεριφορά των πελατών, θα χρησιμοποιήσουμε τον αλγόριθμο **Apriori** για δύο λόγους. Αρχικά για να επιτύχουμε το προφανές, δηλαδή την εξόρυξη γνώσης μέσω κανόνων συσχέτισης. Μία βασική δυνατότητα δευτερευόντως που μας έδωσε, όμως, αυτός ο αλγόριθμος ήταν η ρύθμιση των μεγεθών της **Εμπιστοσύνης** και της **Υποστήριξης** με σκοπό να **φιλτράρουμε - αποκρύψουμε** κάποιους κανόνες μέσω της **απόρριψής** τους. Σε αυτήν την ενότητα, θα αναλύσουμε περαιτέρω τον Apriori αλγόριθμο, αλλά θα εξετάσουμε και άλλους αλγόριθμους εξόρυξης κανόνων συσχέτισης και στη συνέχεια θα τους συγκρίνουμε μεταξύ τους.

3.1.1. Παραδοχές για τον Αλγόριθμο Apriori

Από τα βασικά χαρακτηριστικά του αλγόριθμου είναι πως χρησιμοποιεί την ιδιότητα **συχνών στοιχειοσυνόλων** που περιλαμβάνει το ότι "*οποιοδήποτε υποσύνολο ενός*

συχνού στοιχειοσυνόλου είναι επίσης συχνό". Τα συχνά στοιχειοσύνολα μπορούμε να τα αποκαλέσουμε και κλειστά προς τα κάτω διότι αν κάποιο στοιχειοσύνολο ικανοποιεί τις απαιτήσεις της ελάχιστης υποστήριξης, το ίδιο συμβαίνει και για όλα τα υποσύνολά του. Αναλογιζόμενοι το αντίστροφο της προηγούμενης πρότασης, αν γνωρίζουμε ότι ένα στοιχειοσύνολο δεν είναι συχνό δεν χρειάζεται να δημιουργήσουμε κανένα υπερσύνολό του, σαν υποψήφιο, επειδή και αυτό αποκλείεται να είναι συχνό.

Σύμφωνα με τους Μ. Χαλκίδης, Μ. Βαζιργιάννης (2005), ο αλγόριθμος Apriori, διαβάζει τον αρχικό πίνακα D διαδοχικές φορές. Συνολικά ο πίνακας θα διαβαστεί το πολύ τόσες φορές όσες είναι το πλήθος των διαφορετικών items στον πίνακα. Στο πρώτο διάβασμα (πέρασμα) του πίνακα μετριέται η υποστήριξη των 1-itemsets και βρίσκεται ποια από αυτά ικανοποιούν την απαίτηση για ελάχιστη υποστήριξη. Σε κάθε επόμενο βήμα χρησιμοποιούνται τα itemsets του προηγούμενου περάσματος για να δημιουργηθούν καινούργια itemsets. Τα itemsets αυτά ονομάζονται υποψήφια (candidate itemsets) καθώς δεν γνωρίζουμε την υποστήριξή τους και κατ' επέκταση αν είναι συχνά (frequent). Για το λόγο αυτόν μετριέται η υποστήριξή τους μέσω ενός περάσματος από τον αρχικό πίνακα. Το κλειδί σε όλη αυτή τη διαδικασία είναι ότι σε κάθε βήμα γίνεται ακριβώς ένα μόνο πέρασμα από τον αρχικό πίνακα. Στο τέλος του κάθε βήματος αποφασίζεται ποια itemsets είναι συχνά ώστε να χρησιμοποιηθούν για το επόμενο βήμα. Αυτός είναι περιγραφικά ο τρόπος με τον οποίο ο αλγόριθμος Apriori παράγει τα frequent itemsets.

Το όνομα του αλγορίθμου οφείλεται στην εξής ιδιότητα: *Κάθε υποσύνολο ενός συχνού itemset είναι επίσης συχνό*. Επίσης ισχύει και η αντιστροφή της παρακάτω ιδιότητας: *Υπάρχει τουλάχιστον ένα υποσύνολο ενός μη συχνού itemset που να είναι επίσης μη συχνό.* Δεδομένων αυτών των ιδιοτήτων μπορούμε να παράγουμε τα υποψήφια itemsets από τα ήδη γνωστά frequent itemsets και μόνο. Απορρίπτουμε έτσι ένα μεγάλο σύνολο από υποψήφια itemsets και δεν υπολογίζουμε την υποστήριξή τους, καθώς είναι γνωστό εκ των προτέρων (a priori) ότι αυτά δεν πρόκειται να είναι συχνά.

Πριν δώσουμε τον ψευδοκώδικα που περιγράφει τον αλγόριθμο είναι χρήσιμο να εξηγήσουμε κάποιες παραδοχές που γίνονται και τους συμβολισμούς που χρησιμοποιούνται. Για όλα τα items που υπάρχουν σε έναν πίνακα θεωρούμε ότι υπάρχει μια διάταξη μεταξύ τους, για παράδειγμα λεξικογραφική. Μπορούμε ακόμα να αντικαταστήσουμε τα items με φυσικούς αριθμούς ώστε η διάταξη να είναι περισσότερο προφανής. Τα items που αποτελούν ένα itemset βρίσκονται αποθηκευμένα με βάση αυτή την διάταξη.

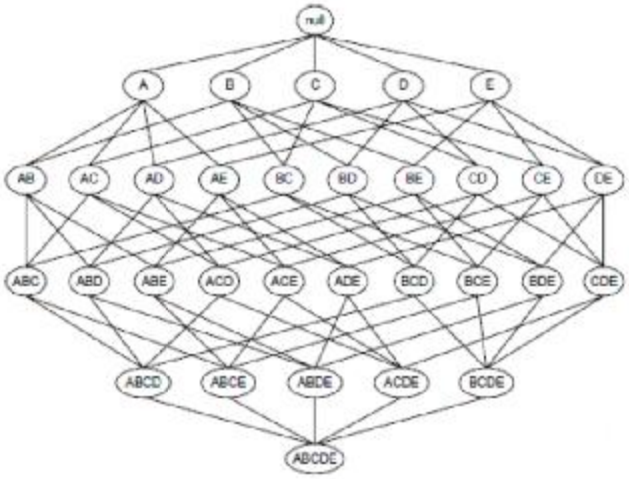
Έστω ένα k-itemset X, τότε χρησιμοποιούμε τον συμβολισμό X[1]·X[2]· … ·X[k] για να δείξουμε ότι το itemset X αποτελείται από τα items X[1], X[2], … ,X[k] για τα οποία ισχύει ότι X[1]< X[2]< … < X[k]. Το σύνολο των συχνών (frequent ή large) k-itemsets θα συμβολίζεται με C_k. Φυσικά ισχύει ότι το σύνολο C_k είναι υπερσύνολο του L_k, δηλαδή $L_k \subseteq C_k$.

Κάθε itemset έχει, εκτός από τη λίστα με τα items που περιέχει, και έναν μετρητή υποστήριξης (support count) που χρησιμοποιείται για να υπολογιστεί η υποστήριξή του. Ο μετρητής αυτός (count) αρχικοποιείται στο 0 και κάθε φορά που συναντάται το συγκεκριμένο itemset σε ένα transaction του πίνακα τότε αυξάνεται κατά 1. Κατά συνέπεια όταν εξεταστούν όλες οι σειρές του πίνακα η τιμή του μετρητή διαιρούμενη με το πλήθος των σειρών του πίνακα δίνει την υποστήριξη του αντίστοιχου itemset. Να σημειωθεί ότι το minsup που χρησιμοποιούμε παρακάτω αντιστοιχεί όχι σε ποσοστά αλλά σε αριθμό transactions που απαιτούνται. Ο αλγόριθμος Apriori δίνεται με μορφή ψευδοκώδικα παρακάτω [2]:

1. L_1 = {large 1-itemsets};
2. **for** (k=2; $L_{k-1}\neq\emptyset$; k++) **do begin**
3. C_k = apriori-gen(L_{k-1}); //δημιουργία υποψηφίων
4. **forall** transactions t∈D **do begin**
5. C_t = subset(C_k, t); //οι υποψήφιοι που περιέχονται στο t
6. **for each** candidates c∈C_t **do**
7. c.count++;
8. end
9. L_k = {c∈C_k | c.count ≥ minsup)
10. End
11. Return $\cup_k L_k$;

Όπως είπαμε και προηγουμένως στο πρώτο πέρασμα από τον πίνακα δοσοληψιών D βρίσκονται τα συχνά 1-itemsets, μετρώντας πόσες φορές εμφανίζονται το κάθε item και απομακρύνοντας αυτά που εμφανίζονται λιγότερο από minsup φορές (εντολή 1). Κάθε επόμενο πέρασμα (εντολή 2), έστω το k πέρασμα, περιλαμβάνει δύο φάσεις. Η πρώτη φάση αφορά στην παραγωγή των υποψηφίων k-itemsets C_k από τα συχνά (k-1)-itemsets L_k που βρέθηκαν στο προηγούμενο πέρασμα. Για το σκοπό αυτό χρησιμοποιείται η συνάρτηση apriori-gen (εντολή 3) που περιγράφετε παρακάτω. Η δεύτερη φάση αφορά στον υπολογισμό του support count για τα υποψήφια itemsets. Για κάθε transaction (εντολή 4) βρίσκονται τα υποψήφια itemsets που περιέχονται σ' αυτό (εντολή 5). Το κρίσιμο σημείο στη δευτερη αυτή φάση είναι ο γρήγορος υπολογισμός του συνόλου C_t, των υποψηφίων δηλαδή που περιέχονται στη δοσοληψία t. Η συνάρτηση subset επιτυγχάνει αυτόν το σκοπό και περιγράφεται αναλυτικά στη συνέχεια. Στο τέλος του περάσματος υπολογίζεται το L_k (εντολή 9) απορρίπτοντας τα itemsets του C_k που δεν είναι συχνά. Ο αλγόριθμος επιστρέφει την ένωση όλων των συχνών itemsets (εντολή 11) λύνοντας έτσι το ζητούμενο πρόβλημα.

Η λογική παραγωγής υποψήφιων στοιχειοσυνόλων παρατίθεται στο παρακάτω πλέγμα της εικόνας:



Εικόνα 4: **Πλέγμα Στοιχειοσυνόλων (Κλειστότητα προς τα κάτω)**

3.1.2. Συνάρτηση Apriori-gen

Η συνάρτηση apriori-gen όπως περιγράψαμε παραπάνω πρέπει να παράγει τα υποψήφια k-itemsets από τα γνωστά συχνά (k-1)-itemsets. Έτσι έχει ως είσοδο το σύνολο L_{k-1} και ως έξοδο το σύνολο C_k, ένα υπερσύνολο δηλαδή του L_k, όπως έχει ήδη εξηγηθεί [2].

Η συνάρτηση αποτελείται από δύο βήματα, το join-step (ένωση) και το prune-step (κλάδεμα) που περιγράφονται από τις παρακάτω σχέσεις:

$C'_k = \{X \cup Y \mid X, Y \in L_{k-1}, |X \cap Y| = k-2\}$ join-step

$C_k = \{X \in C'_k, |X$ contains members of $L_{k-1}\}$ prune-step

Στο πρώτο βήμα γίνεται η ένωση δύο (k-1)-itemsets που ανήκουν στο L_{k-1} και επιπλέον έχουν ακριβώς (k-2) κοινά items. Έτσι το itemset που θα προκύψει από αυτήν την ένωση θα αποτελείται από τα (k-2) κοινά items συν μη κοινό item από τα δύο (k-1)-itemsets, δηλαδή θα έχει σύνολο k itemsets.

Στο επόμενο βήμα γίνεται χρήση της βασικής αρχής του αλγορίθμου Apriori. Κατά συνέπεια απορρίπτονται εκείνα τα itemsets για τα οποία υπάρχει τουλάχιστον ένα (k-1) υποσύνολό τους που να μην ανήκει στο σύνολο L_{k-1} (πρόταση ισοδύναμη με την απαίτηση να υπάρχουν ακριβώς k μέλη του L_{k-1} στα itemsets), γιατί είναι *a priori* γνωστό ότι δεν είναι συχνά.

Στο πρώτο βήμα μπορούμε να κάνουμε μια βελτίωση εκμεταλλευόμενοι τη διάταξη που έχουμε ορίσει για τα items. Χρησιμοποιώντας ορολογία από τη γλώσσα SQL το βελτιωμένο join-step γράφεται ως εξής:

```
insert into Ck´´
```

```
select X[1], X[2], … , X[k-1], Y[k-1]
from L_{k-1}X, L_{k-1}Y
where X[1] = Y[1], … , X[k-2] = Y[k-2], X[k-1]<Y[k-1]
```

Το σύνολο C_k που υπολογίζεται με το τροποποιημένο πρώτο βήμα είναι πιο μικρό από το αντίστοιχο Ck΄, υπολογίζεται πιο εύκολα λόγω της ύπαρξης της διάταξης και κατά συνέπεια επιταχύνει την όλη διαδικασία.

Σε αυτό το σημείο είναι απαραίτητο ένα μικρό παράδειγμα. Έστω ότι το L_3 περιέχει πέντε 3-itemsets L_3={{1 2 3}, {1 2 4}, {1 3 4}, {1 3 5}, {2 3 4}}. Τότε μετά το join-step θα έχουμε το εξής αποτέλεσμα C_4'' = {{1 2 3 4}, {1 3 4 5}}. Τέλος το prune-step θα διαγράψει το itemset {1 3 4 5} επειδή το υποσύνολό του {1 4 5} δεν βρίσκεται στο L_3. Κατά συνέπεια το τελικό αποτέλεσμα θα είναι C_4 = {{1 2 3 4}}.

3.1.2. Συνάρτηση Subset

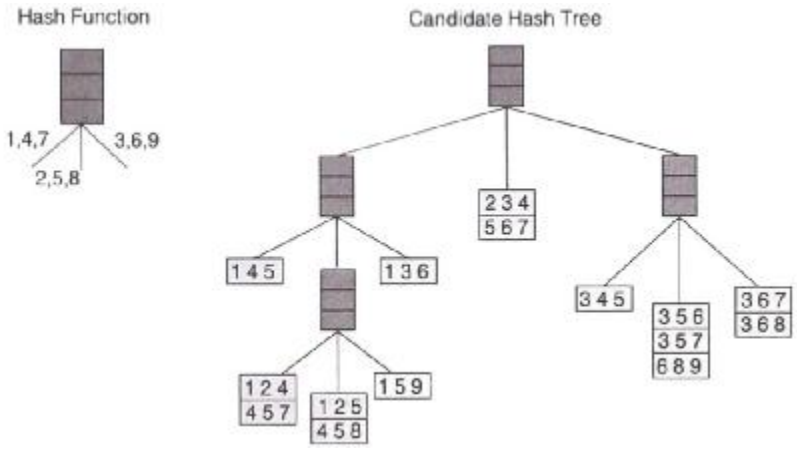
Η συνάρτηση subset έχει αναλάβει το πιο δύσκολο έργο του αλγορίθμου. Πρέπει να υπολογίσει για κάθε transaction ποιο είναι το υποσύνολο εκείνο του C_k το οποίο αποτελείται από όλα itemsets που περιέχονται στο εκάστοτε transaction. Γίνεται αντιληπτό ότι για το σκοπό αυτόν πρέπει τα υποψήφια itemsets να αποθηκεύονται με τέτοιον τρόπο ώστε να επιταχύνεται η όλη διαδικασία [2].

Τα υποψήφια itemsets C_k αποθηκεύονται σε ένα hash-tree (δέντρο κατακερματισμού). Ένας κόμβος του δέντρου αυτού περιέχει είτε μια λίστα από itemsets, αν είναι κόμβος φύλλο, είτε έναν πίνακα κατακερματισμού (hash-tree) αν πρόκειται για εσωτερικό κόμβο. Κάθε κουβάς (bucket) του πίνακα κατακερματισμού ενός εσωτερικού κόμβου δείχνει σε έναν άλλο κόμβο. Θεωρώντας ότι η ρίζα του hash-tree έχει βάθος 1, τότε ένας εσωτερικός κόμβος βάθους d δείχνει σε κόμβους βάθους d+1. Τα itemsets επομένως αποθηκεύονται μόνο στα φύλλα, ενώ οι υπόλοιποι κόμβοι περιέχουν πληροφορία για το πώς θα αναζητηθούν τα itemsets.

Όταν προσθέτουμε ένα καινούργιο itemset, ξεκινάμε από τη ρίζα και συνεχίζουμε διασχίζοντας το δέντρο μέχρι να φτάσουμε σε κάποιο φύλλο. Τότε σε εκείνο το φύλλο προσθέτουμε το itemset στο σύνολο των itemsets που υπάρχουν ήδη αποθηκευμένα. Σε έναν εσωτερικό κόμβο βάθους d αποφασίζουμε ποιο κλαδί του κόμβου θα ακολουθήσουμε εφαρμόζοντας τη συνάρτηση κατακερματισμού στο d-οστό item του itemset. Όλοι οι κόμβοι αρχικά δημιουργούνται ως φύλλα. Όταν ο αριθμός των itemsets για ένα φύλλο ξεπεράσει ένα συγκεκριμένο κατώφλι τότε το φύλλο αυτό μετατρέπετε σε εσωτερικό κόμβο με τόσα φύλλα όσα τα buckets της συνάρτησης κατακερματισμού. Ως συνέπεια του τρόπου δημιουργίας του hash-tree, αν το δέντρο αποθηκεύσει τα υποψήφια k-itemsets, δηλαδή το C_k, τότε θα έχει βάθος

το πολύ k+1 (η μέγιστη διαδρομή θα περιλαμβάνει k hash-tables και θα καταλήγει σε ένα φύλλο).

Ένα hash-tree για το σύνολο C_3 παρουσιάζεται στην εικόνα 5. Ο πίνακας δοσοληψιών έχει 9 διαφορετικά items αριθμημένα με τους φυσικούς 1-9. Η συνάρτηση κατακερματισμού που επιλέχθηκε είναι η modulo 3, δηλαδή το υπόλοιπο της διαίρεσης με το 3, όπως επίσης φαίνεται στο σχήμα. Κάθε εσωτερικός κόμβος κατά συνέπεια δείχνει σε 3 κόμβους.



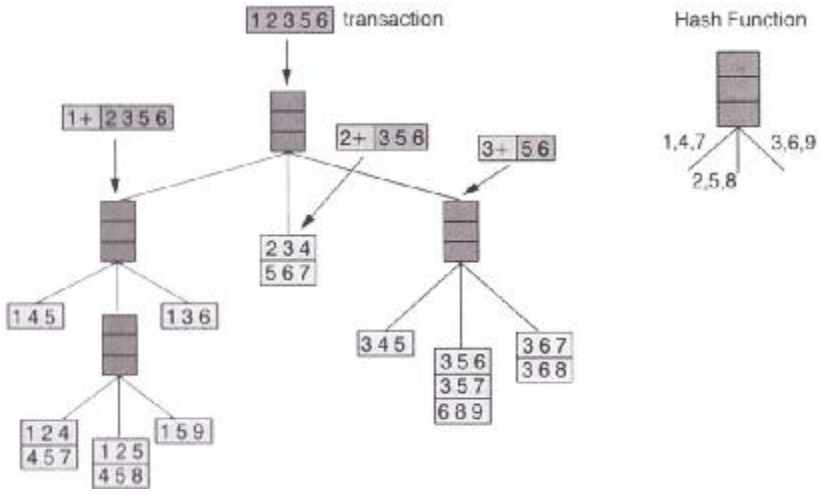
Εικόνα 5: Αποθήκευση itemset {1 5 9}

Ας δούμε πως αποθηκεύεται το itemset {1 5 9} (εικόνα 5). Στη ρίζα (βάθος 1) η συνάρτηση κατακερματισμού για το πρώτο item 1 οδηγεί στο πρώτο κλαδί. Στον αμέσως επόμενο κόμβο το item 5 κατακερματίζεται και η διαδρομή που πρέπει να ακολουθηθεί είναι η μεσαία. Τέλος το itemset 9 οδηγείται μέσω της συνάρτησης κατακερματισμού στο τελευταίο κλαδί του κορμού βάθους 3 και αποθηκεύεται εκεί ως φύλλο.

Με αυτόν τον τρόπο τα itemsets που δημιουργούνται από τη συνάρτηση παραγωγής υποψηφίων apriori-gen αποθηκεύονται στο hash-tree. Ας εξηγήσουμε όμως και τον τρόπο που βρίσκει η συνάρτηση subset τους υποψήφιους που περιέχονται σε μια δοσοληψία t. Αν βρισκόμαστε σε κάποιο φύλλο τότε η διαδικασία τελείωσε καθώς προσθέτουμε στο υπό αναζήτηση σύνολο C_t εκείνα τα itemsets που βρίσκονται αποθηκευμένα στο φύλλο και που περιέχονται στην t. Αν βρισκόμαστε σε κάποιον εσωτερικό κόμβο και φτάσαμε σε αυτό το σημείο εφαρμόζοντας τη συνάρτηση κατακερματισμού για το item i τότε εφαρμόζουμε τη συνάρτηση κατακερματισμού για κάθε item που βρίσκεται μετά το i με βάση τη διάταξη των items που έχουμε ορίσει. Η διαδικασία επαναλαμβάνεται αναδρομικά μέχρι να καταλήξουμε σε κάποιο φύλλο. Στη ρίζα του δέντρου εφαρμόζουμε τη συνάρτηση κατακερματισμού για κάθε item που υπάρχει στην δοσοληψία.

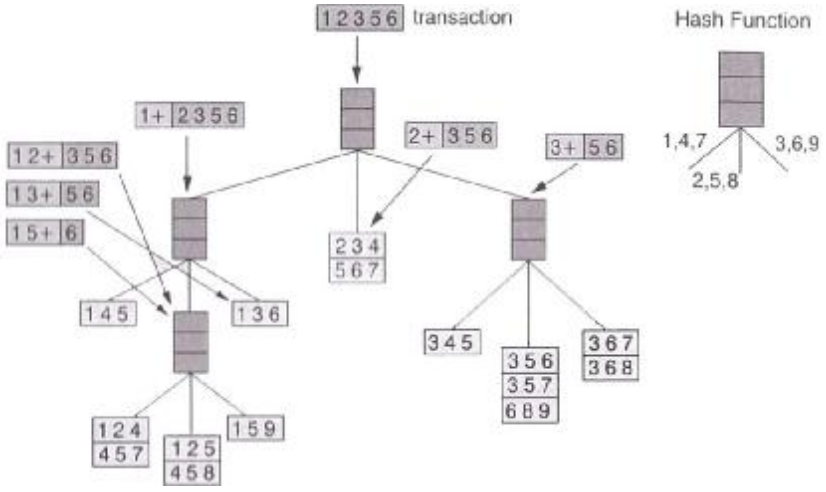
Ας θεωρήσουμε το προηγούμενο παράδειγμα όπου έχουμε αποθηκευμένο το hash-tree για το σύνολο C_3. Έστω επίσης ότι η δοσοληψία t περιέχει τα items {1 2 3 5 6}. Θα παρακολουθήσουμε τι συμβαίνει στη ρίζα του δέντρου (εικόνα 6). Είπαμε ότι πρέπει να εφαρμοστεί η συνάρτηση κατακερματισμού για όλα τα items της

δοσοληψίας ώστε να εξασφαλίσουμε ότι βρίσκουμε τα 3-itemsets. Όπως φαίνεται καλύτερα στο σχήμα για το πρώτο item ακολουθούμε το πρώτο κλαδί, για το δεύτερο το μεσαίο και για το τρίτο το τελευταίο.



Εικόνα 6: **Αναζήτηση** itemset {1 2 3 5 6} (I)

Σε κάθε κόμβο η διαδικασία επαναλαμβάνεται αναδρομικά. Για παράδειγμα ας δούμε τι θα ακολουθήσει στον κόμβο στον οποίο βρεθήκαμε μετά την εφαρμογή της συνάρτησης κατακερματισμού στο item 1 (εικόνα 7). Τώρα πρέπει να κάνουμε hashing σε όλα τα items (μέχρι και το τρίτο από το τέλος item πάντα) που βρίσκεται μετά το 1 με βάση τη διάταξη. Έχουμε επομένως τα items 2, 3 και 5 που το καθένα οδηγεί σε διαφορετικό κλαδί από τον κόμβο που βρισκόμαστε. Τα παραπάνω αποτυπώνονται στο σχήμα που ακολουθεί [2].



Εικόνα 7: **Αναζήτηση** itemset {1 2 3 5 6} (II)

Έστω, λοιπόν, ένα σύνολο δεδομένων που αντιστοιχούν σε 10 διαφορετικά καλάθια αγορών από ένα σούπερ μάρκετ [1]. Κάθε καλάθι περιλαμβάνει ένα υποσύνολο των προϊόντων του σούπερ μάρκετ. Για τις ανάγκες του αλγορίθμου Apriori θα πραγματοποιήσουμε **δυαδική αναπαράσταση** του πίνακα με τα δεδομένα καλαθιών αγορών και ο πίνακας αυτός θα είναι ο ακόλουθος:

Καλάθι	Ψωμί	Καφές	Γάλα	Ζάχαρη
#1	1	0	1	0
#2	0	1	0	0
#3	1	0	1	1
#4	0	1	0	1
#5	1	0	1	1
#6	1	1	1	0
#7	1	0	0	1
#8	1	1	1	1
#9	0	0	1	1
#10	1	1	0	1

Πίνακας 3: Πίνακας Καλαθιών Αγορών (δυαδική αναπαράσταση)

Στα πλαίσια της δυαδικής αναπαράστασης του πίνακα, όπου "1" αντιστοιχεί σε παρουσία του εκάστοτε προϊόντος στο καλάθι αγοράς και, αντίστοιχα "0" στην απουσία του προϊόντος από το καλάθι. Τα όρια υποστήριξης (sup) και εμπιστοσύνης (conf) είναι 40% και 80% αντίστοιχα.

Ο λόγος για τον οποίο θέσαμε τα συγκεκριμένα όρια ήταν, αφενός γιατί θέλαμε σε τουλάχιστον τέσσερις από τους δέκα πελάτες να παρατηρούνται στο καλάθι αγοράς τους αυτά τα προϊόντα και αφετέρου γιατί θέλαμε μία ισχυρή δυναμική στο συνδυασμό των προϊόντων, κάτι το οποίο προσδίδει αξιοπιστία σε ένα συμπέρασμα που προκύπτει από έναν κανόνα συσχέτισης.

Στο πρώτο βήμα (βήμα 1) ο αλγόριθμος δημιουργεί το σύνολο L_1 το οποίο αποτελείται από όλα τα στοιχειοσύνολα (itemsets) μεγέθους 1. Συνεπώς L_1 = {{ψωμί}, {καφές}, {γάλα}, {ζάχαρη}}

Στο βήμα 2 εκτελείται ο βρόγχος επανάληψης *for,* το σώμα του οποίου αποτελείται από την συνάρτηση *apriori-gen*(L_{k-1}) και έναν εσωτερικό βρόγχο επανάληψης *forall*. Η συνάρτηση παράγει όλους τους πιθανούς συνδυασμούς των αντικειμένων, με σκοπό να δημιουργηθεί το σύνολο των υποψηφίων ζευγών αντικειμένων δηλαδή το C_2 = {{ψωμί, καφές}, {ψωμί, γάλα}, {ψωμί, ζάχαρη}, {καφές, γάλα}, {καφές, ζάχαρη}, {γάλα, ζάχαρη}}.

Στο επόμενο βήμα (βήμα 4) ο εσωτερικός βρόγχος *forall* αποτελείται από την συνάρτηση *subset*(C_k,*t*) και έναν τρίτο βρόγχο επανάληψης τον *for each* στο εσωτερικό του οποίου υπάρχει ένας μετρητής. Αυτός αυξάνει κάθε φορά που κάποιο από τα ζεύγη αντικειμένων βρίσκεται στην εκάστοτε συναλλαγή. Για παράδειγμα ο συνδυασμός αντικειμένων {ψωμί, γάλα} εμφανίζεται στις συναλλαγές #1, #3, #5, #6 και #8 οπότε ο μετρητής θα πάρει την τιμή 5.

Κατόπιν, στο βήμα 9 υπολογίζεται η υποστήριξη των μελών του C_2 και απορρίπτονται εκείνα που δεν ξεπερνούν το όριο ελάχιστης υποστήριξης, ώστε να δημιουργηθεί το σύνολο συχνών ζευγών L_2.

{ψωμί, καφές} = 3/10 = 30% < sup (απορρίπτεται)

{ψωμί, γάλα} = 5/10 = 50% ≥ sup

κτλ.

Τελικά: L_2 = { {ψωμί, γάλα}, {Ψωμί, Ζάχαρη}, {Γάλα, Ζάχαρη} }

Στην επόμενη επανάληψη του βήματος 2 ο βρόγχος θα έχει ως εξής, `for(k=3; L2≠Ø; k++)` και η συνάρτηση *apriori-gen* θα λάβει ως όρισμα για την δημιουργία υποψηφίων το σύνολο L_2. Τα βήματα για την δημιουργία υποψηφίων τριάδων (βάσει των σχέσεων στην σελίδα 29) διαμορφώνεται ως εξής:

Βήμα συνένωσης (join – step): C'_3 = { {ψωμί, γάλα}∪{ψωμί, ζάχαρη}| {ψωμί, γάλα}, {ψωμί, ζάχαρη} ∈ L_2, |{ψωμί, γάλα}∩{ψωμί, ζάχαρη}| =1} ⇒ C'_3 = { {ψωμί, γάλα, ζάχαρη} }

Βήμα κλαδέματος (prune – step): Οι επιμέρους δυάδες (ζεύγη) του {Ψωμί, Γάλα, Ζάχαρη} ανήκουν όλες στο L_2, άρα C_3 = { {ψωμί, γάλα, ζάχαρη} }

Από το βήμα 5 προκύπτει ότι ο συνδυασμός αντικειμένων {Ψωμί, Γάλα, Ζάχαρη} εμφανίζεται στις συναλλαγές #1, #5 και #8 και επομένως απορρίπτεται γιατί το support είναι μικρότερο του ορίου, άρα L_3 = {Ø}.

Ο **αλγόριθμος εύρεσης συχνών συνόλων** σταματά εδώ μιας και δεν ικανοποιείται πλέον η συνθήκη τερματισμού της εντολής 2. Συνεπώς το μέγιστο συχνό σύνολο αντικειμένων είναι το L_2.

Το επόμενο βήμα είναι η εξαγωγή των κανόνων από τα συχνά σύνολα (στο συγκεκριμένο παράδειγμα μόνο το L_2), βάσει της εμπιστοσύνης τους.

L_2 = { {Ψωμί, Γάλα}, {Ψωμί, Ζάχαρη}, {Γάλα, Ζάχαρη} }

Ελέγχεται η Εμπιστοσύνη όλων των πιθανών κανόνων που μπορεί να προκύψουν από το L_2:

{Ψωμί, Γάλα}

Ψωμί g Γάλα: εμπιστοσύνη = 5/7 = 71% < conf (απορρίπτεται)
Γάλα g Ψωμί: εμπιστοσύνη = 5/6 = 83% > conf (εγκρίνεται)

{Ψωμί, Ζάχαρη}

Ψωμί g Ζάχαρη: εμπιστοσύνη = 5/7 = 71% < conf (απορρίπτεται)
Ζάχαρη g Ψωμί: εμπιστοσύνη = 5/7 = 71% < conf (απορρίπτεται)

{Γάλα, Ζάχαρη}

Γάλα g Ζάχαρη: εμπιστοσύνη = 4/6 = 66% < conf (απορρίπτεται)
Ζάχαρη g Γάλα: εμπιστοσύνη = 4/7 = 57% < conf (απορρίπτεται)

Τελικά παράγεται μόνον ο κανόνας: Γάλα g Ψωμί, δηλαδή όποιος αγοράζει Γάλα, αγοράζει και Ψωμί. Αν, για παράδειγμα, ελαττώναμε τη ζητούμενη εμπιστοσύνη στο 70%, τότε θα είχαν παραχθεί τέσσερις κανόνες αντί για ένας.

Χρονική πολυπλοκότητα του αλγορίθμου:

Έστω, *N* ο αριθμός των συναλλαγών, *M* το κατώτατο όριο υποστήριξης και *R* ο αριθμός των διακριτών στοιχείων στην Βάση. Η πολυπλοκότητα για την δημιουργία ενός συνόλου *i* είναι O(R^i) και ο χρόνος υπολογισμού της υποστήριξης για κάθε σύνολο ολοκληρώνετε σε O(n). Ο συνολικός χρόνος θα είναι: **O[(R + N) + (R^2 + N) + (R^3 + N) …] = O[MN + (R^1+R^2+ … R^M)] = O(MN+ (1-R^M)/(1-R))**

3.2 Ο Αλγόριθμος FP-Growth

Ο αλγόριθμος Apriori επιτυγχάνει πολύ καλή απόδοση μειώνοντας μεγάλο αριθμό υποψήφιων προτύπων. Παρόλα αυτά, σε περιπτώσεις μεγάλου αριθμού συχνών προτύπων ή προτύπων τα οποία έχουν μεγάλο μήκος ή χρήση κατωφλίων υποστήριξης αρκετά χαμηλών υπάρχει το εξής πρόβλημα: Είναι πολύ κοστοβόρο να διαχειριστούμε μεγάλο αριθμό από συχνά πρότυπα. Το παραπάνω πρόβλημα είναι αυτό που έδωσε λαβή για περαιτέρω μελέτη με αντικειμενικό σκοπό να αποφύγουμε την παραγωγή και τεστάρισμα μεγάλου αριθμού υποψηφίων προτύπων [4][8].

Στα πλαίσια αυτά λοιπόν αναπτύχθηκαν τρεις μέθοδοι, σύμφωνα με τους Jiawei Han, Jian Pei et. (2004), ΙΠ Γουρδουλής (2008):

1. Το πρώτο βήμα είναι η κατασκευή μιας **συμπαγής δομής δεδομένων** (compact data structure) που ονομάζεται frequent pattern ή συντομογραφικά Fp-tree. Πρόκειται για ένα δέντρο που περιέχει πληροφορίες για τα συχνά πρότυπα με προθεματικό τρόπο. Πιο συγκεκριμένα κόμβους στο δέντρο έχουν μόνο τα συχνά πρότυπα μεγέθους 1, ενώ οι κόμβοι είναι διατεταγμένοι με τρόπο ώστε οι πιο συχνοί να είναι πιο εύκολα προσβάσιμοι. Αυτού του είδους το δέντρο χρησιμοποιείται έναντι της βάσης δεδομένων γιατί αφενός είναι μικρότερης έκτασης αφετέρου περιέχει πολύ πιο ποιοτική πληροφορία.

2. Ως δεύτερο βήμα περνάμε σε μία **μέθοδο σταδιακής εύρεσης των συχνών προτύπων** (pattern-fragment growth mining), βασιζόμενοι στο Fp-tree και ξεκινώντας από τα συχνά πρότυπα μεγέθους 1. Τα πρότυπα αυτά θα αποτελέσουν την κατάληξη επόμενων συχνών προτύπων. Πως συμβαίνει αυτό; Απλά ξεχωρίζουμε τα στοιχεία από την βάση με τις συναλλαγές που καταλήγουν στα συχνά πρότυπα μεγέθους 1. Αυτό μας δίνει μια βάση, υποσύνολο της αρχικής, και για την οποία κατασκευάζουμε το αντίστοιχο Fp-tree (conditional). Η διαδικασία αυτή γίνεται επαναληπτικά επικολλώντας τα αντικείμενα ακριβώς πριν την κατάληξη. Η ύπαρξη των συχνών προτύπων μέσα στις συναλλαγές σημαίνει αυτόματα ότι θα απεικονίζονται οπωσδήποτε και στο Fp-tree, μέσα από κάποιο

διακριτό μονοπάτι, πράγμα που σημαίνει ότι η εύρεση των συχνών προτύπων γίνεται με ολοκληρωμένο τρόπο.

3. Ο μηχανισμός που μόλις περιγράψαμε κινείται πάνω στην λογική του *«διαίρει και βασίλευε»*, σε αντίθεση με τον Apriori που σε κάθε βήμα διογκώνονταν το μέγεθος και η πολυπλοκότητα του προβλήματος. Εδώ κάθε φορά το conditional Fp-tree είναι αρκετά μικρότερο από τον προκάτοχό του και το μόνο που κάνουμε στο τέλος είναι να επικολλήσουμε την κατάληξη. Έτσι η ανίχνευση συχνών προτύπων ολοένα και μεγαλύτερου μεγέθους οδηγεί σε δομές δεδομένων ολοένα και πιο μικρές.

3.2.1 Σχεδιασμός και κατασκευή του Fp-tree

Έστω I = { a_1,a_2,a_3..,a_m} ένα σύνολο από διακριτά αντικείμενα (**items**).

Έστω επίσης D = < T_1,T_2,T_3..,T_m > ένα σύνολο από συναλλαγές (**transactions**) όπου κάθε συναλλαγή T είναι ένα υποσύνολο αντικειμένων του.

Η **υποστήριξη** (ή αλλιώς συχνότητα εμφάνισης) ενός προτύπου A – όπου A είναι ένα σύνολο από αντικείμενα – **είναι ο αριθμός των συναλλαγών που περιέχουν το A.**

Θα λέμε ότι ένα **πρότυπο A είναι συχνό όταν η υποστήριξή του είναι μεγαλύτερη από κάποιο προκαθορισμένο ελάχιστο κατώφλι κ.**

Έτσι λοιπόν με βάση τα παραπάνω το **πρόβλημα εύρεσης συχνών προτύπων** που αντιμετωπίζουμε έχει ως εξής [4][8]:

*«Δεδομένης μιας βάσης συναλλαγών και ενός ελάχιστου κατωφλίου στήριξης κ, καλούμαστε να βρούμε **όλα** τα συχνά πρότυπα και όχι μέρος αυτών».*

Παράδειγμα:

Έστω ο παρακάτω πίνακα που απεικονίζεται μια βάση συναλλαγών και το ελάχιστο κατώφλι υποστήριξης (min support) ορισμένο στην τιμή 3.

TID	Συναλλαγές	Ταξινόμηση συχνών αντικειμένων
100	f, a, c, d, g, l, m, p	f, c, a, m, p
200	a, b, c, f, l, m, o	f, c, a, b, m
300	b, f, h, j, o	f, b
400	b, c, k, s, p	c, b, p
500	a, f, c, e, l, p, m, n	f, c, a, m, p

Πίνακας 4: Πίνακας συναλλαγών

Η κατασκευή του Fp-tree μπορεί να σχεδιαστεί με βάση τις ακόλουθες παρατηρήσεις:

1. Γνωρίζοντας ότι μόνο τα συχνά αντικείμενα θα διαδραματίσουν κάποιο ρόλο στην εξόρυξη συχνών-πρότυπων, εκτελείται μία σάρωση της βάσης δεδομένων συναλλαγής ώστε να προσδιοριστούν και να απομονωθούν τα συχνά αντικείμενα.

2. Θα πρέπει να γίνει προσπάθεια το σύνολο των συχνών αντικειμένων σε μια συναλλαγή να αποθηκευτεί σε μια συμπαγή δομή ώστε να αποφευχθεί η επαναληπτική ανίχνευση της βάσης που είναι αρκετά κοστοβόρα.

3. Εάν υπάρχουν πολλαπλές συναλλαγές που μοιράζονται ένα σύνολο συχνών αντικειμένων, αυτές μπορούν να συγχωνευτούν και για τα κοινά σύνολα θα αυξάνει το πλήθος των εμφανίσεών τους. Για να ελέγξουμε αν δύο σύνολα αντικειμένων είναι πανομοιότυπα ταξινομούμε τα αντικείμενα στις συναλλαγές με μια καθορισμένη σειρά (είτε αύξουσα είτε φθίνουσα).

4. Αν δύο συναλλαγές έχουν κοινό πρόθεμα, σύμφωνα με την νέα ταξινόμηση, το πρόθεμα αυτό θα αποθηκευτεί **μόνο μία** φορά στην δομή, συνοδευόμενη από τον αριθμό των συναλλαγών που εμφανίζεται. Εάν τα αντικείμενα ταξινομούνται με φθίνουσα σειρά, βάσει του αριθμό εμφανίσεων τους, υπάρχουν μεγαλύτερες πιθανότητες τα προθέματα που είναι κοινά να είναι περισσότερα.

Λαμβάνοντας υπόψη τις παραπάνω παρατηρήσεις κατασκευάζουμε το frequent fp-tee ακολουθώντας τα εξής βήματα:

Αρχικά, διενεργείται μια σάρωση της Βάσης Δεδομένων ώστε να εντοπιστεί η λίστα των συχνών αντικείμενων. Έτσι έχουμε:

< (f:4),(c:4),(a:3),(b:3),(m:3),(p:3)>

Ο αριθμός μετά το ":" υποδηλώνει την υποστήριξη (support). Επίσης να τονίσουμε εδώ ότι τα αντικείμενα ταξινομούνται όπως φαίνεται σε φθίνουσα σειρά. Είναι μια χρήσιμη παρατήρηση που θα χρησιμοποιηθεί στην διαμόρφωση των μονοπατιών του δέντρου, και από δω και πέρα τα αντικείμενα μέσα στις συναλλαγές θα αναφέρονται ταξινομημένα.

Στην συνέχεια αρχίζει η κατασκευή του δέντρου. Αρχίζουμε από την ρίζα την οποία και επιγράφουμε ως **"null"**. Αμέσως μετά κάνουμε και δεύτερο πέρασμα στην βάση συναλλαγών ως εξής:

1. Η πρώτη συναλλαγή, **< (f:1),(c:1),(a:1),(m:1),(p:1) >**, οδηγεί στην κατασκευή του πρώτου κλαδιού του δέντρου. Κάθε αντικείμενο αποτελεί και έναν κόμβο, στον οποίο αποθηκεύουμε το όνομα και υποστήριξη του. Οι συναλλαγές παρατίθενται σύμφωνα με τη σειρά στη λίστα των συχνών αντικειμένων.

2. Η δεύτερη συναλλαγή **<f, c, a, b, m> ,** παρατηρούμε ότι έχει **κοινό πρόθεμα** με την πρώτη συναλλαγή το < f, c, a >. Αυτό σημαίνει ότι για το κοινό πρόθεμα

θα χρησιμοποιηθούν οι υπάρχοντες κόμβοι με την μόνη διαφορά ότι **η υποστήριξη τους θα αυξηθεί κατά 1**. Αμέσως μετά θα προστεθούν δύο κόμβοι (ο ένας κάτω από τον άλλο) απόγονοι του (α:2), οι (b:1) και (m:1).

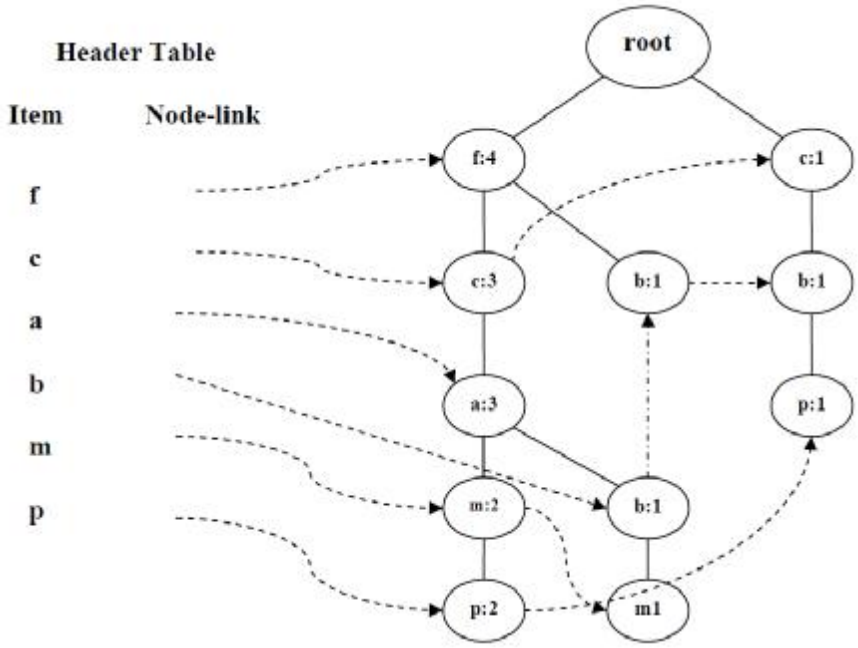
3. Η τρίτη συναλλαγή **<f, b> ,** έχει κοινό πρόθεμα μόνο το f το κόμβο του οποίου και αυξάνουμε την υποστήριξη κατά ένα (δηλαδή f:3). Από τον κόμβο f δημιουργούμε απόγονο τον (b:1).

4. Η τέταρτη συναλλαγή **< c, b, p> ,** δεν παρουσιάζει κανένα κοινό πρόθεμα οπότε ξεκινάμε την κατασκευή του δεύτερου κλαδιού του δέντρου με κόμβους (c:1), (b:1) και (p:1).

5. Η τελευταία συναλλαγή είναι πανομοιότυπη με την πρώτη με αποτέλεσμα να μην έχουμε την δημιουργία κανενός νέου κόμβου αλλά την αύξηση των υποστηρίξεων κατά ένα.

Για να διευκολυνθεί η πρόσβαση σε οποιοδήποτε κόμβο του δέντρου παράλληλα κατασκευάζουμε και έναν πίνακα ο οποίος αποθηκεύει τα λεγόμενα αντικείμενα οδηγούς. Πιο συγκεκριμένα για κάθε αντικείμενο (item) ο πίνακας (header table) περιέχει ένα δείκτη (node-link) που οδηγεί στον πρώτο κόμβο με το ίδιο όνομα όπως φαίνεται και στην εικόνα 8. Επίσης κάθε κόμβος του δέντρου είναι διασυνδεδεμένος με τον επόμενο κόμβο που έχει το ίδιο όνομα, επιτυγχάνοντας με αυτόν τον τρόπο πλήρης διασυνδεσιμότητα.

Παρακάτω φαίνεται το συνολικό Fp-tree του παραδείγματος.



Εικόνα 8: Global FP-tree

Με βάση λοιπόν την μεθοδολογία που παραθέσαμε μπορούμε να κωδικοποιήσουμε τα βήματα αλγορίθμου κατασκευής Fp-tree ως εξής:

1. Πρώτο πέρασμα της Βάση Δεδομένων ώστε να εντοπιστεί το σύνολο F των συχνών αντικειμένων, συνοδευόμενα από την υποστήριξη τους και ταξινόμησή του σε φθίνουσα διάταξη.

2. Δημιουργία της ρίζα του **Fp-tree** με όνομα **T** και με επιγραφή null

    a. Για κάθε συναλλαγή απομονώνουμε μόνο τα συχνά αντικείμενα τα οποία και τα αναδιατάσσουμε ανάλογα με την ταξινόμηση του βήματος 1. Δίνουμε σε κάθε συναλλαγή την μορφή **[p | P]** όπου p είναι το πρώτο αντικείμενο ενώ P είναι η εναπομείνασα λίστα.

    b. Καλούμε μια διαδικασία **insert_tree([p | P], T)** η οποία δουλεύει ως ακολούθως:

        i. Εάν το T έχει απόγονο N έτσι ώστε N.item-name = p.item-name τότε αύξησε την υποστήριξη του κατά 1.

        ii. Διαφορετικά δημιουργούμε κόμβο N με αρχική υποστήριξη 1.Ο πατέρας του ορίζεται η ρίζα του δέντρου ενώ φροντίσουμε να δημιουργήσουμε και τις σωστές συνδέσεις με τους συνονόματους κόμβους.

        iii. Εάν το P δεν είναι άδειο τότε καλούμε επαναληπτικά την διαδικασία **insert_tree([P, N]).**

Χρονική πολυπλοκότητα αλγορίθμου*:*

- Για την κατασκευή του FP-tree χρειάζονται δύο περάσματα στην Βάση Δεδομένων. Στο πρώτο πέρασμα, συλλέγονται τα σύνολα συχνών αντικειμένων και στο δεύτερο πέρασμα γίνεται η κατασκευή του FP-tree.

- Το κόστος εισαγωγής των συναλλαγών στο Fp-tree είναι **O(|freq(Transaction)|)**, όπου freq(Transaction) το σύνολο των συχνών αντικειμένων μέσα στη συναλλαγή (transaction).

3.2.2 Πληρότητας και συμπαγής δομή του Fp-tree

Δοθείσης μιας βάσης συναλλαγών και ενός κατωφλίου υποστήριξης **κ** ορίζουμε ως F το σύνολο των συχνών αντικειμένων στην Βάση. Για κάθε συναλλαγή T ορίζουμε ως **freq(T)** την προβολή συχνών αντικειμένων (frequent item projection) και ισούται με το σύνολο των συχνών αντικειμένων που υπάρχουν στην συναλλαγή και ισχύει ότι freq(T) = T ∩ F. Με βάση τις αρχές του Apriori, το σύνολο των προβολών των συναλλαγών της βάσης αρκεί για να μας δώσει το σύνολο των συχνών προτύπων χωρίς καμιά απώλεια και χωρίς διπλοεγγραφές (completeness).

Αποδεικνύεται ότι: *«Δοθείσης μίας Βάσης Δεδομένων και ενός κατωφλιού υποστήριξης κ το ολοκληρωμένο σύνολο των προβολών συχνών αντικειμένων των συναλλαγών μπορεί να παραχθεί από το Fp-tree».*

Από το παραπάνω συνάγεται ότι η δομή Fp-tree είναι η πιο σημαντική στην διαδικασία εξόρυξης προτύπων και για το λόγο αυτό θα ασχοληθούμε με το μέγεθος του [4][8].

Ισχύει ότι: *«Δοθείσης μιας Βάσης Δεδομένων (DB) και ενός κατωφλίου υποστήριξης κ και χωρίς να υπολογίζουμε την ρίζα null, το **μέγεθος του Fp-tree** φράσσεται από το* $\boldsymbol{\Sigma_{T \in DB} |freq(T)|}$ *ενώ το **ύψος του δέντρου** φράσσεται από το* $\boldsymbol{max_{T \in DB} \{|freq(T)|\}}$*»*

Απόδειξη

Με βάση την διαδικασία κατασκευής του δέντρου για κάθε συναλλαγή Τ στην Βάση υπάρχει ένα μονοπάτι στο Fp-tree που ξεκινά από τον κόμβο με επιγραφή του προθέματος της συναλλαγής. Αυτό σημαίνει ότι το σύνολο των κόμβων του μονοπατιού είναι ακριβώς το ίδιο με τα συχνά αντικείμενα που περιέχει η συναλλαγή (δηλαδή το μέγεθος freq(T)). **Άρα στην χειρότερη περίπτωση όπου δεν υπάρχει καθόλου προθεματική επικάλυψη συναλλαγών το μέγεθος του δέντρου (πλήθος κόμβων) είναι ίσος με** $\boldsymbol{\Sigma_{T \in DB} |freq(T)|}$**.**

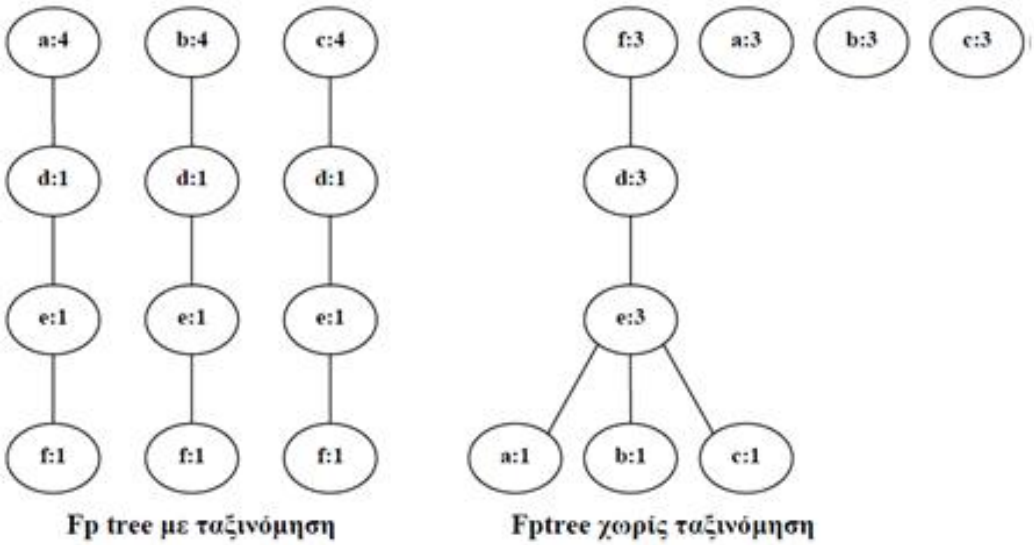
Με την ίδια λογική αφού το freq(T) είναι ο αριθμός των κόμβων σε μια συναλλαγή τότε το ύψος του δέντρου θα ισούται με το μήκος της συναλλαγής που περιέχει τα περισσότερα συχνά αντικείμενα χωρίς διπλοεγγραφές αντικειμένων.

Από την παραπάνω απόδειξη συμπεραίνουμε ότι **το μέγεθος του Fp-tree δεν μπορεί να ξεπερνά το μέγεθος της βάσης συναλλαγών**. Αυτό συμβαίνει γιατί κάθε συναλλαγή θα προσθέσει το πολύ ένα μονοπάτι για το Fp-tree, με μήκος ίσο με τον αριθμό των συχνών αντικειμένων στην εν λόγω συναλλαγή. Στην χειρότερη περίπτωση όπου δεν υπάρχει επικάλυψη συναλλαγών το μέγεθος της δομής θα ισούται με το μέγεθος της Βάσης Δεδομένων. Συνήθως είναι αρκετά μικρότερο ανάλογα με το ποσοστό της επικάλυψης. Σε αντίθεση τον αλγόριθμο Apriori που είναι δυνατόν να δημιουργήσει **εκθετικά αυξανόμενο** αριθμό υποψήφιων συχνών προτύπων, ένα FP-tree δεν μπορεί σε καμία περίπτωση να έχει εκθετικό αριθμό κόμβων.

Ένα ακόμη πλεονέκτημα του Fp-tree είναι η φθίνουσα ταξινόμηση των συχνών αντικειμένων. Αυτό γιατί ξεκινώντας από αντικείμενα με την μεγαλύτερη υποστήριξη σημαίνει ότι έχουν μεγαλύτερο αριθμό εμφανίσεων στην Βάση και άρα μεγαλύτερες πιθανότητες να μοιράζονται στις διάφορες συναλλαγές. Αυτή η ταξινόμηση ενισχύει την πυκνότητα της δομής του Fp-tree.

Ωστόσο, αυτό δεν σημαίνει ότι το δέντρο που είναι πάντα έτσι κατασκευασμένο επιτυγχάνει τη μέγιστη πυκνότητα. Για παράδειγμα, έστω οι συναλλαγές {adef, bdef, cdef, a, a, a, b, b, b, c, c, c} και έστω ότι το κατώφλι υποστήριξης έχει την τιμή 3. Τα

σύνολα που ικανοποιούν το κατώφλι υποστήριξης είναι τα εξής { a:4, b:4, c:4, d:3, e:3, f:3}. Ακολουθώντας φθίνουσα ταξινόμηση έχουμε a→b→c→d→e→f και η κατασκευή του δέντρου περιλαμβάνει δώδεκα κόμβους, όπως φαίνεται και στο πρώτο σχήμα της εικόνας 9. Ενώ με την ταξινόμηση f→d→e→a→b→c οι κόμβοι του δέντρου γίνονται 9, όπως φαίνεται και στο δεύτερο σχήμα της εικόνας 9.



Εικόνα 9: Fp-trees με και χωρίς ταξινόμηση

Το παράδειγμα δόθηκε προκειμένου για να γίνει αντιληπτό ότι παρά το γεγονός ότι η φθίνουσα ταξινόμηση στην γενική περίπτωση είναι αποδοτική, θα πρέπει να γνωρίζουμε την ιδιαιτερότητα που μπορεί να παρουσιάζει η Βάση Δεδομένων [4][8].

3.2.3 Εξόρυξη συχνών προτύπων με χρήση του Fp-tree

Σε αυτή την ενότητα θα αναπτύξουμε τον αλγόριθμο εξόρυξης συχνών προτύπων , FP-Growth, βάσει της δομής που αναφερθήκαμε προηγουμένως [4][8]. Προτού παραθέσουμε τον ψευδοκώδικα του αλγορίθμου θα αναλύσουμε κάποιες σημαντικές ιδιότητες του Fp-tree που θα μας φανούν χρήσιμες.

Βασικές Αρχές για τον Fp-Growth αλγόριθμο

Ιδιότητα 1: *Για κάθε συχνό αντικείμενο a_i όλα τα πιθανά πρότυπα που περιέχουν το a_i είναι προσβάσιμα ακολουθώντας τους συνδέσμους των κόμβων με την επιγραφή a_i, ξεκινώντας από την κεφαλή του a_i που είναι αποθηκευμένη σε πίνακα*.

Η ιδιότητα αυτή δεν χρειάζεται απόδειξη αφού προκύπτει απευθείας από τον τρόπο κατασκευής της δομής.

Προκειμένου να συνάγουμε και τις υπόλοιπες ιδιότητες παραθέτουμε το ακόλουθο παράδειγμα.

Παράδειγμα:

Έστω το Fp-tree που προκύπτει από τον πίνακα 4. Ξεκινάμε την εξόρυξη από το τέλος του πίνακα με τους δείκτες κεφαλές βάσει του δέντρου στην εικόνα 8.

- Για τον κόμβο p (p:3) ακολουθώντας τους συνδέσμους ανακαλύπτουμε ότι ακολουθεί δύο μονοπάτια. Το πρώτο είναι το <f:4, c:3, a:3, m:2, p:2> και το δεύτερο το <c:1, b:1, p:1>. Το πρώτο μονοπάτι υποδηλώνει ότι **το σετ {f, c, a, m, p} εμφανίζεται 2 φορές**, το σετ {f, c, a} 3 φορές, ενώ το {f} μόνο του εμφανίζεται 4 φορές. Με την ίδια λογική το δεύτερο μονοπάτι μας δείχνει **ότι το σετ {c, b, p} εμφανίζεται μόνο μία φορά**.

  Τα δύο μονοπάτια {fcam:2} και {cb:1} αποτελούν την **προθεματική βάση (subpattern-base) του αντικειμένου p** και ονομάζεται **p' s conditional pattern base**.

  Η κατασκευή ενός Fp-tree για την προθεματική βάση του p ονομάζεται **conditional Fp-tree του p** και οδηγεί στο κλαδί (c:3). Αυτό γιατί έχουμε ορίσει κατώφλι υποστήριξης ίσο με 3 αποκλείοντας έτσι το b. Το μόνο συχνό πρότυπο που μπορεί να εξαχθεί είναι το **(cp:3)** και η διαδικασία εύρεσης προτύπων για το p θεωρείται λήξασα.

- Για τον κόμβο m (m:3) ακολουθώντας τους συνδέσμους έχουμε και εδώ δύο μονοπάτια. Το πρώτο είναι <f:4 c:3, a:3, m:2> και το δεύτερο <f:4,c:3,a:3,b:1,m:1>. Καταλήγουμε στην προθεματική βάση του m που αποτελείται από: **<fca:2>** και **<fcab:1>.** Με βάση το κατώφλι υποστήριξης εξαιρούμε το b και έχουμε το conditional Fp-tree του m που αποτελείται από ένα κλαδί το **<f:3,c:3,a:3>.** Έτσι το μόνο συχνό πρότυπο που μπορεί να εξαχθεί είναι το **fcam:3.**

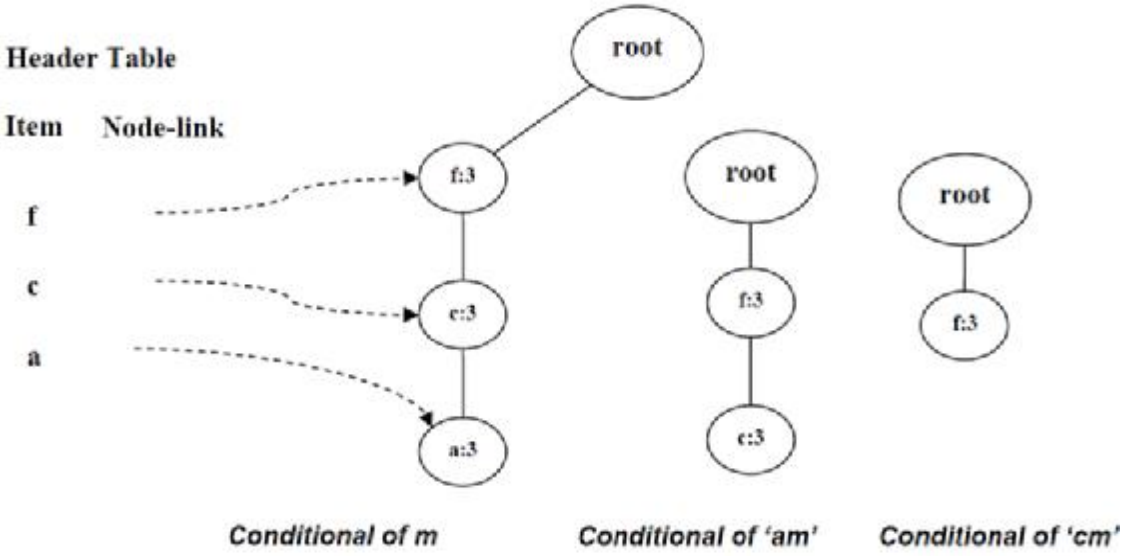
  Πριν προχωρήσουμε στο επόμενο αντικείμενο του πίνακα με τους δείκτες, θα πρέπει, εφόσον το πρότυπο που βρήκαμε περιέχει παραπάνω από 1 αντικείμενο, να ακολουθηθεί επαναληπτικά η ίδια διαδικασία που με παραπάνω. Θα πρέπει δηλαδή ως επόμενο βήμα να κατασκευάσουμε τα conditional Fp-tree για τα πρότυπα **(am:3), (cm:3), (fm:3)**. Για την πρώτη περίπτωση εξάγουμε τα συχνά πρότυπα **<cam:3>, <fam:3>.** Για την δεύτερη έχουμε εξαγωγή του **<fcm:3>** και τέλος για την τρίτη περίπτωση δεν έχουμε εξαγωγή προτύπου.

  Από τα πρότυπα αυτά μπορούμε να δημιουργήσουμε conditional Fp-tree για το <cam:3> και να εξάγουμε το συχνό πρότυπο **<fcam:3>.**

  Συμπερασματικά για τον κόμβο m έχουμε τα εξής 8 συχνά πρότυπα:

  { (m:3), (am:3), (cm:3), (fm:3), <cam:3>, <fam:3>,<fcam:3> }

Η απεικόνιση των δομών που κατασκευάζονται για τον κόμβο m φαίνονται παρακάτω εικόνα:



Εικόνα 10: Conditional Fp-trees

Οι ιδιότητες που προκύπτουν από τα παραπάνω είναι οι εξής [4][8]:

Ιδιότητα 2: *Ο υπολογισμός των συχνών προτύπων με κατάληξη το αντικείμενο* a_i *περιλαμβάνει τους κόμβους του Fp-tree που αναφέρονται* ***μόνο στα μονοπάτια του*** a_i.

Αυτό έχει μια πολύ σημαντική 'παρενέργεια' στην απομόνωση των μονοπατιών όταν πρόκειται για την κατασκευή του conditional: **Οι κόμβοι των μονοπατιών αυτών θα πρέπει να έχουν υποστήριξη ίση με την υποστήριξη του κόμβου** a_i**.** Έτσι ενώ είδαμε για τον κόμβο m υπήρχε το μονοπάτι < f:4 c:3, a:3, m:2 >, στο conditional το μονοπάτι μετατρέπεται σε **< f:2, c:2,a:2 >.**

Από την ιδιότητα 2 προκύπτουν δύο ακόμη ιδιότητες που είναι σημαντικές στην εξόρυξη συχνών προτύπων:

Ιδιότητα 2.1: *Έστω α ένα σετ αντικειμένων στην βάση και Β το αντίστοιχο conditional Fp-tree. Έστω επίσης ότι β είναι ένα σύνολο αντικειμένων στο Β.* ***Τότε η υποστήριξη του α*** $\cup$ ***β μέσα στην Βάση είναι ίση με την υποστήριξη του β μέσα στο Β.***

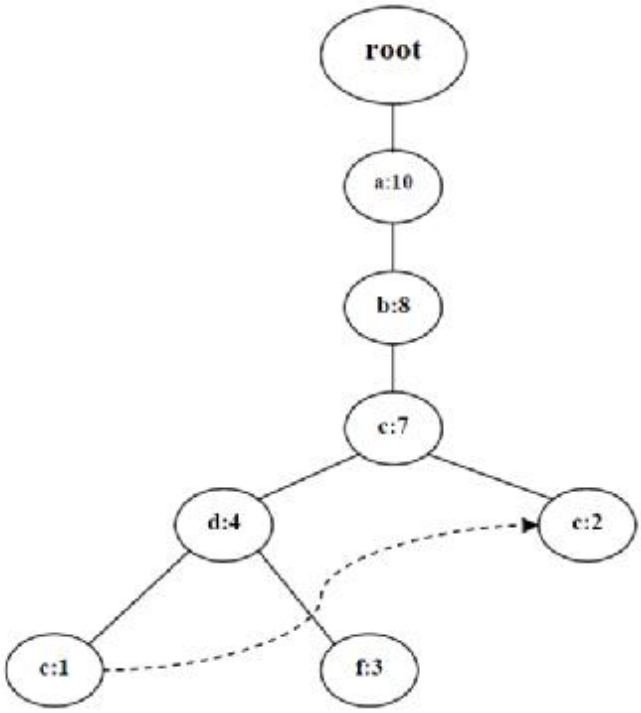
Ιδιότητα 2.2: *Έστω α ένα συχνό πρότυπο στην βάση και Β το αντίστοιχο conditional Fp-tree. Έστω επίσης β ένα σύνολο αντικειμένων στο Β.* ***Τότε το α*** $\cup$ ***β είναι συχνό μέσα στην Βάση αν και μόνο αν το β είναι συχνό μέσα στο Β.***

3.2.3 Βελτιωμένες τεχνικές Fp-Growth αλγορίθμου

Ο αλγόριθμος που περιγράφηκε στην προηγούμενη ενότητα λειτουργεί ικανοποιητικά για οποιοδήποτε Fp-tree. Παρόλα αυτά υπάρχουν διάφορες βελτιώσεις που μπορούν να γίνουν. Μια από αυτές ονομάζεται **single prefix path Fp-tree** και χρησιμοποιεί μια παραλλαγή του Fp-tree. Το δέντρο αυτό αποτελείται ένα μοναδικό μονοπάτι (πρόθεμα) που ξεκινά από την ρίζα και εκτείνεται μέχρι τον πρώτο κόμβο που ξεκινά διακλάδωση. Αυτός ο κόμβος έχει ως απογόνους του περισσότερους από έναν κόμβους [4][8]. Για να αναλύσουμε την λειτουργία αυτής της δομής ας δούμε το παρακάτω παράδειγμα.

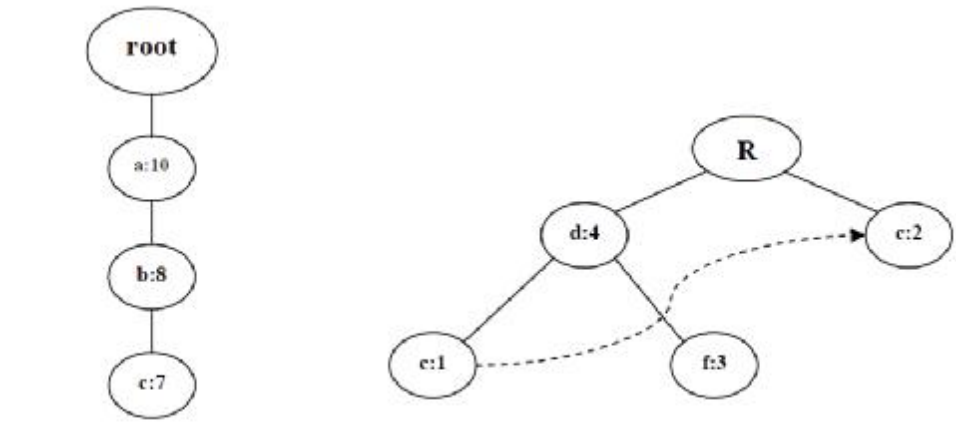
Παράδειγμα

Έστω το δέντρο που απεικονίζεται παρακάτω.



Εικόνα 11:Fp-tree

Όπως φαίνεται πληρεί τις προδιαγραφές που περιγράψαμε άρα πρόκειται για ένα single prefix Fp-tree. Μπορούμε να εφαρμόσουμε τον FP-Growth αλγόριθμο ώστε να εξαχθούν τα συχνά πρότυπα. Αυτό που θα κάνουμε όμως είναι να εκμεταλλευτούμε την ιδιαιτερότητα της δομής και να την σπάσουμε σε δύο κομμάτια: **α) ένα κομμάτι αποτελεί ο μονοκόμματος κλάδος <α:10, b:8, c:7> και β) το υπόλοιπο μέρος του δέντρου με την προσθήκη μιας εικονικής ρίζας R.**



Εικόνα 12:Single και Multipath κομμάτι του Fp-tree

Η εξόρυξη προτύπων θα γίνει ξεχωριστά για τα δύο κομμάτια και στη συνέχεια θα συνδυαστούν μεταξύ τους.

Για το πρώτο κομμάτι τα πράγματα είναι απλά. Τα συχνά πρότυπα ανιχνεύονται απαριθμώντας όλα τα δυνατά μονοπάτια που προκύπτουν. Για τα μονοπάτια αυτά η υποστήριξη (αριθμός εμφανίσεων) τους είναι η ελάχιστη από τις υποστηρίξεις των αντικειμένων που συμμετέχουν στο μονοπάτι. Έτσι λοιπόν έχουμε τα εξής συχνά πρότυπα:

{ (a:10), (b:8), (c:7), (ab:8), (ac:7), (bc:7), (abc:7) }

Για το δεύτερο κομμάτι εφαρμόζουμε τον κλασσικό αλγόριθμο FP-Growth με αποτέλεσμα τα παρακάτω πρότυπα:

{ (d:4), (c:3), (f:3), (df:3) }

Θα μπορούσε να πει κανείς ότι η δουλειά μας τελειώνει με την συγχώνευση των δύο συνόλων προτύπων. Παρατηρώντας όμως τα δύο επιμέρους σχήματα θα μπορούσε κάποιος να δει ότι **το πρώτο κομμάτι αποτελεί το conditional Fp-tree για κάθε πρότυπο από το δεύτερο κομμάτι**. Αυτή η παρατήρηση μας οδηγεί στο συμπέρασμα ότι υπάρχουν πρότυπα που θα προκύψουν τον συνδυασμό του κάθε προτύπου του δεύτερου κομματιού με τα πρότυπα του πρώτου.

Για παράδειγμα για το πρότυπο (d:4) έχουμε **(d:4) X freq(πρώτου κομματιού)** το οποίο μας δίνει:

{ (ad:4), (bd:4), (cd:4), (abd:4), (acd:4), (bcd:4), (abcd:4) }

Η διαδικασία αυτή γίνεται και για τα υπόλοιπα πρότυπα.

Συμπερασματικά μπορούμε να πούμε ότι τα συχνά πρότυπα προκύπτουν από τις εξής 3 πηγές: **α) συχνά πρότυπα του πρώτου κομματιού β) συχνά πρότυπα του δεύτερου κομματιού και γ) freq(πρώτο κομμάτι) X freq(δεύτερο κομμάτι)**

Μια τελευταία παρατήρηση είναι η μεθοδολογία του single prefix Fp-tree μας δίνει το complete σετ των συχνών προτύπων. Αυτό σημαίνει αφενός ότι **τα συχνά πρότυπα που παράγονται είναι διακριτά μεταξύ τους** και αφετέρου ότι **δεν υπάρχει συχνό πρότυπο που να παράγεται εκτός των (α), (β), (γ).**

Μια σύντομη απόδειξη αυτού ξεκινά με την παραδοχή ότι το αρχικό Fp-tree μας δίνει το πλήρες (complete) σύνολο των προτύπων. Κάθε πρότυπο που παράγεται από το συνολικό δέντρο όμως μπορεί να παραχθεί είτε από το πρώτο κομμάτι (single prefix) είτε από το δεύτερο είτε από το συνδυασμό τους. Άρα η μέθοδος μας δίνει το σύνολο των συχνών προτύπων.

Όσον αφορά την διακριτότητα των συχνών προτύπων, τα πρότυπα που παράγονται από τις δύο δομές είναι σίγουρα διακριτά μεταξύ τους. Η εξέταση της διακριτότητας στρέφεται τώρα στον συνδυασμό των δύο δομών. Και εδώ όμως τα πρότυπα είναι διακριτά **γιατί κάθε συχνό πρότυπο του single prefix κομματιού μπορεί να θεωρηθεί ως η conditional βάση ενός συχνού αντικειμένου του δεύτερου κομματιού**, με υποστήριξη την μικρότερη των δύο μερών.

3.2.4 Αλγόριθμος FP-Growth για single prefix Fp-tree

Στην ενότητα αυτή θα παρουσιάσουμε σε ψευδογλώσσα τον αλγόριθμο εύρεσης συχνών προτύπων με τον FP-Growth [4][8]. Πρώτα από όλα ορίζουμε:

Είσοδος: Βάση συναλλαγών, κατώφλι υποστήριξης κ

Έξοδος: Το πλήρες σύνολο συχνών προτύπων

Method: call FP-Growth(FP-tree, null).

```
Procedure FP-Growth(Tree, α){
1. if Tree contains a single prefix path
   //Mining single prefix-path Fp-tree
2. then {
3.   let P be the single prefix-path part of Tree;
4.   let Q be the multipath part with the top branching node
     replaced by a null root;
5.   for each combination (denoted as β) of the nodes in the
     path P do
6.        generate pattern β ∪ α with support = minimum
          support of nodes in β;
7.   let freq_pattern_set(P) be the set of patterns so
     generated;
}
8. else let Q be Tree;
```

```
9. for each item a_i in Q do {
   //Mining multipath Fp-tree
10.  generate pattern β = a_i ∪ α with support = a_i.support;
11.  construct β's conditional pattern-base and then β's
     conditional Fp-tree Tree_β;
12.  if Tree_β ≠ ∅
13.  then call FP-Growth(Tree_β, β);
14.  let freq_pattern_set(Q) be the set of patterns so
     generated;
}
15. return (freq_pattern_set(P) ∪ freq_pattern_set(Q) ∪
(freq_pattern_set(P)×freq_pattern_set(Q)))
}
```

Ανάλυση: Ο αλγόριθμος λειτουργεί δημιουργώντας σύνολα προτύπων για κάθε συχνό αντικείμενο β. Αυτά τα σύνολα προτύπων αποτελούνται από τα προθέματα του β στις συναλλαγές, με προσαρμοσμένες τις υποστηρίξεις. Τα σύνολα προτύπων αυτά αποτελούν τη βάση για την κατασκευή αντίστοιχων conditional Fp-trees. Τα conditional αυτά σύμφωνα με την θεωρητική μελέτη είναι σίγουρα μικρότερης κλίμακας από το σύνολο προτύπων που αντιπροσωπεύουν. Επίσης η διαδικασία του συνόλου προτύπων γίνεται επαναληπτικά με ολοένα και μικρότερο σύνολο που έχει ως επίπτωση και μικρότερα conditional Fp-trees.

Η επαναληπτική αυτή λογική που ομοιάζει με την λογική του διαίρει και βασίλευε είναι πολύ πιο αποδοτική από την παραγωγή και έλεγχο ενός τεράστιου αριθμού από υποψήφια συχνά πρότυπα.

3.3 Μέτρα Ενδιαφέροντος των Κανόνων Συσχέτισης

Έστω ένας κανόνας συσχέτισης, που έχει την μορφή LHS g RHS, όπου LHS είναι αριστερό μέλος του κανόνα (υπόθεση) και RHS το δεξί μέλος του κανόνα (συμπέρασμα). Στη συνέχεια αναλύονται ορισμένα από τα πιο γνωστά μέτρα ενδιαφέροντος των κανόνων όπως περιγράφονται από τους Μ.Χαλκίδης, Μ. Βαζιργιάννης (2005), οι τιμές των οποίων θα μας απασχολήσουν σε επόμενη ενότητα [2].

Υποστήριξη:

Η υποστήριξη (support) ενός κανόνα συσχέτισης είναι το ποσοστό όλων των περιπτώσεων στο σύνολο δεδομένων που ικανοποιούν έναν κανόνα. Πιο συγκεκριμένα, η υποστήριξη ορίζεται ως:

$$Support = \frac{n(LHS \cap RHS)}{N} \text{ ή}$$

$$Support = p(LHS \cap RHS)$$

όπου το N είναι ο συνολικός αριθμός υπό εξέταση περιπτώσεων και το n(LHS) δείχνει τον αριθμό περιπτώσεων που ικανοποιούν το αριστερό μέλος.

Η υποστήριξη μπορεί να θεωρηθεί ως ένδειξη του πόσο συχνά ένας κανόνας εμφανίζεται σε ένα σύνολο στοιχείων και κατά συνέπεια πόσο σημαντικός είναι ένας κανόνας.

Παράδειγμα: Υποθέτουμε ότι υπάρχουν 1000 περιπτώσεις και το LHS και RHS καλύπτουν 100 περιπτώσεις. Η κάλυψη είναι 100/1000 = 0.1.

Εμπιστοσύνη:

Η εμπιστοσύνη (confidence) ενός κανόνα συσχέτισης είναι το ποσοστό των περιπτώσεων που καλύπτονται από το LHS του κανόνα και οι οποίες καλύπτονται επίσης από το RHS:

$$Confidence = \frac{p(RHS \cap LHS)}{n(LHS)} \text{ ή}$$

$$Confidence = \frac{p(RHS \cap LHS)}{P(LHS)}$$

όπου το n(LHS) δείχνει τον αριθμό περιπτώσεων που καλύπτονται από το αριστερό μέλος. Η εμπιστοσύνη παίρνει τις τιμές στο διάστημα [0, 1].

Μια τιμή της εμπιστοσύνης κοντά στο 1 είναι ένδειξη ενός σημαντικού κανόνα συσχέτισης.

Παράδειγμα: Υποθέστε ότι το LHS καλύπτει 100 περιπτώσεις και το RHS καλύπτει 50 των περιπτώσεων που καλύπτονται από το LHS. Η εμπιστοσύνη είναι 50/100 = 0,5.

Τα μέτρα ενδιαφέροντος που συζητήσαμε παραπάνω, η υποστήριξη και η εμπιστοσύνη, χρησιμοποιούνται ευρέως στην διαδικασία εξαγωγής κανόνων συσχέτισης και είναι επίσης γνωστά ως μέτρα Agrawal και Srikant's Itemset μέτρα. Με βάση τους ορισμούς τους, θα μπορούσαμε να πούμε ότι η εμπιστοσύνη αντιστοιχεί στην ισχύ ενώ η υποστήριξη στην στατιστική σημασία ενός κανόνα.

Leverage:

Το μέτρο Leverage ενός κανόνα συσχέτισης είναι το ποσοστό των πρόσθετων περιπτώσεων που καλύπτονται και από το LHS και από το RHS πάνω από εκείνο που αναμένεται εάν τα LHS και τα RHS ήταν ανεξάρτητα. Αυτό είναι ένα μέτρο της σπουδαιότητας της συσχέτισης που περιλαμβάνει τόσο την *εμπιστοσύνη* όσο και την

κάλυψη (δηλαδή, το ποσοστό των περιπτώσεων που έχουν τις τιμές των γνωρισμάτων που ορίζονται στο LHS) του κανόνα. Ειδικότερα ορίζεται ως:

$$Leverage = p(RHS \mid LHS) - [p(LHS) \cdot p(RHS)]$$

Το Leverage παίρνει τις τιμές στο διάστημα [-1, 1]. Οι τιμές του leverage που είναι ίσες ή κάτω από 0, δείχνουν μια ισχυρή ανεξαρτησία μεταξύ LHS και RHS. Αφ' ετέρου, οι τιμές του leverage κοντά στο 1 είναι ένδειξη ενός σημαντικού κανόνα συσχέτισης.

Παράδειγμα: Υποθέστε ότι υπάρχουν 1000 περιπτώσεις, το LHS καλύπτει 200 περιπτώσεις, το RHS καλύπτει 100 περιπτώσεις, και το RHS καλύπτει 50 των περιπτώσεων που καλύπτονται από το LHS. Το ποσοστό των περιπτώσεων που θα αναμενόταν να καλυφθεί και από το LHS και από το RHS, εάν ήταν ανεξάρτητες είναι: (200/1000) · (100/1000) = 0,02. Κατόπιν η ισχύς είναι (0,05 – 0,02) = 0,03. Ο συνολικός αριθμός περιπτώσεων που αυτή η τιμή της ισχύς αντιπροσωπεύει είναι 30.

Lift:

Το μέτρο lift ενός κανόνα συσχέτισης ορίζεται ως η εμπιστοσύνη διαιρούμενη με το ποσοστό όλων των περιπτώσεων που καλύπτονται από το RHS. Αυτό είναι ένα μέτρο της σπουδαιότητας της συσχέτισης και είναι ανεξάρτητη από την *κάλυψη (coverage)*.

$$Lift = \frac{p(LHS \cap RHS)}{p(LHS) \cdot p(RHS)} \text{ ή}$$

$$Lift = \frac{Confidence}{p(RHS)}$$

Παίρνει τιμές στο R_+ (το διάστημα των πραγματικών θετικών αριθμών). Με βάση τις τιμές του Lift οδηγούμαστε στα ακόλουθα συμπεράσματα για το ενδιαφέρων των κανόνων:

1. Lift g 1 σημαίνει ότι το RHS και το LHS είναι ανεξάρτητα, το οποίο δείχνει ότι ο κανόνας δεν παρουσιάζει κάποιο ενδιαφέρον.
2. Τιμές lift κοντά στο $+\propto$. Εδώ, έχουμε τις ακόλουθες περιπτώσεις:
    a. RHS $\subseteq$ LHS ή LHS $\subseteq$ RHS. Εάν οποιαδήποτε από αυτές τις περιπτώσεις ικανοποιείται, μπορούμε να καταλήξουμε στο συμπέρασμα ότι ο κανόνας δεν παρουσιάζει κάποιο ενδιαφέρον.
    b. P(RHS) είναι κοντά στο 0 ή το P(RHS|LHS) είναι κοντά στο 1. Η πρώτη περίπτωση δείχνει ότι ο κανόνας δεν είναι σημαντικός. Αφ' ετέρου, η δεύτερη περίπτωση είναι μια καλή ένδειξη ότι ο κανόνας είναι ενδιαφέρον.

3. Lift = 0 σημαίνει ότι P(RHS|LHS) = 0 ⇔ P(RHS ∩ LHS) = 0, το οποίο δείχνει ότι ο κανόνας δεν είναι σημαντικός.

Παράδειγμα: Υποθέστε ότι υπάρχουν 1000 περιπτώσεις, το LHS καλύπτει 200 περιπτώσεις, το RHS καλύπτει 100 περιπτώσεις, και το RHS καλύπτει 50 απ' τις περιπτώσεις που καλύπτονται από το LHS. Η εμπιστοσύνη είναι 50/200 = 0,25. Το ποσοστό όλων των περιπτώσεων που καλύπτονται RHS είναι 100/1000 = 0,1. Το lift είναι 0,25/0,1 =2,5.

Περαιτέρω συζήτηση: Βασιζόμενοι στον καθορισμό των κανόνων συσχέτισης και των σχετικών μέτρων είναι προφανές ότι η *υποστήριξη* είναι μια ένδειξη σπουδαιότητας του κανόνα βασισμένη στο σύνολο των δεδομένων που τον υποστηρίζουν. Παραδείγματος χάριν, υποθέτοντας τον κανόνα A g B, μια υψηλή *υποστήριξη* του κανόνα είναι ένδειξη ότι ένας μεγάλος αριθμός εγγραφών περιέχει τόσο το αριστερό όσο και το δεξί μέλος αυτού του κανόνα και έτσι μπορεί να θεωρηθεί ως αντιπροσωπευτικός κανόνας του συνόλου δεδομένων μας. Επιπλέον, η *εμπιστοσύνη* εκφράζει την εμπιστοσύνη με βάση τα διαθέσιμα δεδομένα ότι όταν ισχύει το αριστερό μέλος του κανόνα, τότε το δεξί μέλος επίσης ισχύει.

Αν και η υποστήριξη και η εμπιστοσύνη είναι χρήσιμα στην εξόρυξη κανόνων συσχέτισης σε πολλές εφαρμογές, μπορούν να μας παραπλανήσουν σε μερικές περιπτώσεις. Με βάση το πλαίσιο υποστήριξης-εμπιστοσύνης ένας κανόνας μπορεί να προσδιοριστεί ως ενδιαφέρον ακόμα κι αν η εμφάνιση του A δεν υπονοεί την εμφάνιση του B. Σε αυτή την περίπτωση το Lift και το Leverage θα μπορούσαν να θεωρηθούν ως εναλλακτικά μέτρα ενδιαφέροντος τα οποία δίνουν επίσης μια ένδειξη για το LHS και RHS.

Επίσης το Lift είναι ένα άλλο μέτρο, το οποίο μπορεί να δώσει μια ένδειξη μεγάλης σημασίας του κανόνα, ή του πόσο ενδιαφέρον είναι κανόνας. Το Lift αντιπροσωπεύει το πλεονέκτημα πρόβλεψης που ένας κανόνας προσφέρει πέρα από την απλή υπόθεση που βασίζεται στην συχνότητα του συμπεράσματος του κανόνα (RHS). Κατά συνέπεια, το Lift μπορεί να είναι μια ένδειξη εάν ένας κανόνας θα μπορούσε να θεωρηθεί ως αντιπροσωπευτικός των δεδομένων ώστε να χρησιμοποιηθεί στο στάδιο της λήψης αποφάσεων. Για παράδειγμα, έστω ένας κανόνας A + B g G με εμπιστοσύνη 85% και υποστήριξη (G) = 90%. Λόγω της υψηλής εμπιστοσύνης του κανόνα μπορούμε να καταλήξουμε στο συμπέρασμα ότι είναι ένας σημαντικός κανόνας. Από την άλλη πλευρά όμως το δεξί μέλος του κανόνα αντιπροσωπεύει το 90% των υπό μελέτη στοιχείων, δηλαδή ένα μεγάλο μέρος των δεδομένων περιέχει το G. Συνεπώς ο κανόνας μπορεί να μην είναι πολύ ενδιαφέρον δεδομένου ότι υπάρχει μια υψηλή πιθανότητα το δεξί μέλος του κανόνα (G) να ικανοποιηθεί από ένα υψηλό ποσοστό των υπό εξέταση δεδομένων αλλά συγχρόνως η σπουδαιότητα του κανόνα (RHS) είναι υψηλά υποστηριζόμενη. Κατά συνέπεια αυτός ο κανόνας μπορεί να έχει νόημα για τη διαδικασία λήψης αποφάσεων ή την εξαγωγή του γενικού κανόνα όσον αφορά στην συμπεριφορά των δεδομένων. Τέλος, το Leverage εκφράζει την υπερ-αντιπροσώπευση του κανόνα σε σχέση με την αντιπροσώπευση

του στο σύνολο δεδομένων εάν δεν υπάρχει καμία αλληλεπίδραση μεταξύ LHS και RHS.

Ο ΑΛΓΟΡΙΘΜΟΣ ΤΗΣ ΔΕΙΓΜΑΤΟΛΗΨΙΑΣ

Για να διευκολύνουμε το αποτελεσματικό μέτρημα των στοιχειοσυνόλων σε μεγάλες Βάσεις Δεδομένων, μπορούμε να χρησιμοποιήσουμε δειγματοληψία της Βάσης. Ο πρωτότυπος "*Αλγόριθμος της Δειγματοληψίας*" μειώνει τον αριθμό των περασμάτων της Βάσης σε ένα, στην καλύτερη περίπτωση, και δύο, στη χειρότερη περίπτωση. Το δείγμα της Βάσης Δεδομένων επιλέγεται με τέτοιον τρόπο έτσι ώστε να μπορεί να χωρέσει στη μνήμη. Στη συνέχεια, οποιοσδήποτε αλγόριθμος, όπως ο Apriori, χρησιμοποιείται για να βρει τα συχνά στοιχειοσύνολα στο δείγμα. Αυτά θεωρούνται σαν ενδεχομένως συχνά (**potentially large - PL**) και χρησιμοποιούνται ως υποψήφια για μέτρημα, χρησιμοποιώντας ολόκληρη τη Βάση Γνώσης. Επιπλέον υποψήφιοι μπορούν να καθοριστούν με εφαρμογή της συνάρτησης αρνητικού ορίου (negative border) ως προς τα συχνά στοιχειοσύνολα από το δείγμα. Ολόκληρο το σύνολο των υποψηφίων γίνεται το $C = BD^{-}(PL) \cup PL$. Η συνάρτηση του αρνητικού ορίου είναι μία γενίκευση του Apriori-Gen αλγορίθμου. **Ορίζεται ως το ελάχιστο σύνολο στοιχειοσυνόλων, τα οποία δεν ανήκουν στο PL, αλλά των οποίων όλα τα υποσύνολα ανήκουν στο PL**.

Παράδειγμα:

Χρησιμοποιούμε τον αλγόριθμο της δειγματοληψίας για να βρούμε όλα τα συχνά στοιχειοσύνολα στα δεδομένα ενός παντοπωλείου όπου **sup = 20%**. Οι συναλλαγές του παντοπωλείου που εξετάζουμε φαίνονται στον παρακάτω πίνακα:

Πέρασμα	**Υποψήφιο**	**Συχνά Στοιχειοσύνολα**
1	{Μπύρα}, {Ψωμί}, {Ζάχαρη}, {Γάλα}, {Βούτυρο}	{Μπύρα}, {Ψωμί}, {Γάλα}, {Βούτυρο}
2	{Μπύρα, Ψωμί}, {Μπύρα, Γάλα}, {Μπύρα, Βούτυρο}, {Ψωμί, Γάλα}, {Ψωμί, Βούτυρο}, {Γάλα, Βούτυρο}	{Ψωμί, Βούτυρο}

Πίνακας 5: Πίνακας συχνών στοιχειοσυνόλων

Υποθέτουμε ότι το δείγμα της Βάσης Δεδομένων καθορίζεται να είναι οι πρώτες δύο συναλλαγές:

$Ds = \{t_1 = \{\Psi\omega\mu\acute{\iota}, Ζάχαρη, Βούτυρο\}, t_2 = \{\Psi\omega\mu\acute{\iota}, Βούτυρο\}\}$

Αν μειώσουμε το sup στο 10%, τότε, για να είναι ένα στοιχειοσύνολο συχνό θα πρέπει να εμφανίζεται στο δείγμα τουλάχιστον σε 0,1 X 2 συναλλαγές. Συνεπώς, θα

πρέπει να εμφανίζεται σε μία από τις δύο συναλλαγές. Όταν εφαρμοστεί ο Apriori στο Ds παίρνουμε:

PL = {{Ψωμί}, {Ζάχαρη}, {Βούτυρο}, {Ψωμί, Ζάχαρη}, {Ψωμί, Βούτυρο}, {Ζάχαρη, Βούτυρο}, {Ψωμί, Ζάχαρη, Βούτυρο}}

Όταν εφαρμόζουμε το αρνητικό όριο, παίρνουμε:
$BD^{-}(PL)$ = {{Μπύρα}, {Γάλα}}

Έτσι, χρησιμοποιούμε το ακόλουθο σύνολο των υποψηφίων για μέτρημα κατά τη διάρκεια του πρώτου περάσματος της Βάσης Δεδομένων: C = {{Ψωμί}, {Ζάχαρη}, {Βούτυρο}, {Ψωμί, Ζάχαρη}, {Ψωμί, Βούτυρο}, {Ζάχαρη, Βούτυρο}, {Ψωμί, Ζάχαρη, Βούτυρο}, {Μπύρα}, {Γάλα}}

Να σημειώσουμε πως κατά τη διάρκεια αυτού του περάσματος χρησιμοποιούμε το s=20% και το εφαρμόζουμε και στις πέντε συναλλαγές της Βάσης Δεδομένων. Για να είναι συχνό ένα στοιχειοσύνολο, θα πρέπει να το έχουμε σε 20% X 5 ή τουλάχιστον σε μία συναλλαγή. Στη συνέχεια βρίσκουμε ότι και το {Μπύρα} και το {Γάλα} είναι συχνά. Επομένως έχουμε ML = {{Μπύρα}, {Γάλα}}. Ακολουθώντας τον αλγόριθμο, θέτουμε C = L. Εφαρμόζοντας το αρνητικό όριο παίρνουμε: $BD^{-}(C)$ = {{Μπύρα, Ψωμί}, {Μπύρα, Ζάχαρη}, {Μπύρα, Γάλα}, {Μπύρα, Βούτυρο}, {Ψωμί, Γάλα}, {Ζάχαρη, Γάλα}, {Γάλα, Βούτυρο}}

Εφόσον αποκαλύφθηκαν νέα (μάλλον συχνά) στοιχειοσύνολα, εφαρμόζουμε εκ νέου τον αλγόριθμο και αυτή τη φορά βρίσκουμε όλα τα στοιχειοσύνολα μεγέθους τρία. Η τελική του εφαρμογή βρίσκει όλα τα στοιχειοσύνολα μεγέθους τέσσερα και κάνουμε ένα πέρασμα της Βάσης Δεδομένων χρησιμοποιώντας όλα τα υπόλοιπα στοιχειοσύνολα που δεν είναι ήδη γνωστό ότι είναι συχνά.

Ο κώδικας του αλγόριθμου έχει ως εξής:

```
Input:
     I    //Itemsets
     D    //Database of transactions
     s    //Support
Output:
     L    //Large Itemsets
Sampling Algorithm:
     Ds = Sample drawn from D;
     PL = Apriori (I, Ds, smalls);
     C = PL U BD-(PL);
     L = ∅;
     for each Ii ανήκει στο C do
          Ci=0; //Initial counts for each itemset are 0;
     for each tj ανήκει στο D do
          if Ii ανήκει στο tj then
```

```
        Ci= Ci+1;
    for each Ii ανήκει στο C do
        if Ci≥(sX|D|) do
            L = L U Ii;
    ML={x|x ανήκει στο BD-(PL) ^ x ανήκει στο L};
    //Missing Large Itemsets
    if ML≠∅, then
        C=L; //Set candidates to be the large itemsets
        repeat
            C = C U BD-(C);
            /*Expand candidate sets using negative border*/
                until no new itemsets are added to C;
                for each Ii ανήκει στο C do
                    Ci=0;
            /*Initial counts for each itemset are 0 */
                for each tj ανηκει στο D do
            /*Second scan count */
                        for each Ii ανήκει στο C do
                            if Ii ανήκει στο tj, then
                                Ci = Ci +1;
                if Ci≥(sX|D|) do
                    L = L U Ii;
End
```

Ο αλγόριθμος της Δειγματοληψίας δείχνει ότι η εφαρμογή του Apriori αλγορίθμου στο δείγμα, εκτελείται χρησιμοποιώντας μία Υποστήριξη που καλείται **smalls**. Εδώ smalls μπορεί να είναι οποιαδήποτε τιμή για την υποστήριξη, η οποία φέρει τιμή μικρότερη από το s. Η ιδέα είναι ότι, με το **να μειώσουμε την Υποστήριξη**, θα ανακαλυφθούν τα περισσότερα από τα πραγματικά συχνά στοιχειοσύνολα ολόκληρης της Βάσης Δεδομένων [3].

3.4 Εμπειρική Ανάλυση

Η εμπειρική ανάλυση αποτελεί την κύρια εναλλακτική έναντι της μαθηματικής ανάλυσης για την μέτρηση της αποτελεσματικότητα ενός αλγορίθμου [7]. Μερικοί στόχοι που επιδιώκονται κατά την ανάλυση αυτή συμπεριλαμβάνουν α) τον έλεγχο της ακρίβειας ενός θεωρητικού ισχυρισμού σχετικά με την απόδοση του αλγορίθμου, β) τη σύγκριση της αποδοτικότητας μερικών αλγορίθμων για την επίλυση του ίδιου προβλήματος ή των διαφορετικών υλοποιήσεων του ίδιου αλγορίθμου, γ) την κατασκευή μιας υπόθεσης σχετικά με την κλάση αποδοτικότητας ενός αλγορίθμου και δ) την επιβεβαίωση της αποδοτικότητας ενός συγκεκριμένου προγράμματος που υλοποιεί τον αλγόριθμο σε μια συγκεκριμένη μηχανή.

Για την επίλυση υπάρχουν δύο εναλλακτικές. Η πρώτη εναλλακτική επιλογή είναι η **εισαγωγή ενός μετρητή** (ή μετρητών) στο πρόγραμμα που υλοποιεί τον αλγόριθμο, ώστε να μετρά τον αριθμό των επαναλήψεων εκτέλεσης της βασικής λειτουργίας του αλγορίθμου. Αυτό είναι κάτι που μπορεί σχεδόν πάντοτε να γίνει χωρίς δυσκολία. Θα πρέπει όμως να προσεχθούν οι περιπτώσεις όπου η βασική λειτουργία βρίσκεται σε περισσότερες της μίας θέσης στο πρόγραμμα οπότε πρέπει όλες αυτές να προσμετρηθούν.

Η δεύτερη εναλλακτική λύση, η οποία θα μας απασχολήσει στην παρούσα εργασία, είναι η **χρονομέτρηση του προγράμματος** που υλοποιεί το συγκεκριμένο αλγόριθμο. Ο απλούστερος τρόπος για να γίνει αυτό είναι με τη χρήση μιας εντολής συστήματος όπως για παράδειγμα, της **time** στο UNIX. Εναλλακτικά μπορούμε να μετρήσουμε το χρόνο εκτέλεσης ενός τμήματος κώδικα του αυτού ($\mathbf{t_{start}}$) και αμέσως μετά την ολοκλήρωση του ($\mathbf{t_{finish}}$) και κατόπιν υπολογίζοντας τη διαφορά μεταξύ των $t_{start} - t_{finish}$. Στην JAVA διατίθεται η μέθοδος **currentTimeMills()** της κλάσης System.

3.5 Το λογισμικό WEKA

Το WEKA (Wekato Enviroment for knowledge Analysis) είναι ένα λογισμικό (software) για εξόρυξη δεδομένων σε Java το οποίο περιέχει υλοποιημένες μεθόδους για:

- Προεπεξεργασία Δεδομένων
- Ταξινόμηση
- Συσταδοποίηση
- Εύρεση Κανόνων Συσχέτισης

Είναι ένα από τα πιο δημοφιλή εργαλεία εξόρυξης πληροφορίας κυρίως στον Ακαδημαϊκό χώρο. Έχει αναπτυχθεί σε Java από ερευνητική ομάδα του Πανεπιστημίου Waikato της Νέας Ζηλανδίας. Η εφαρμογή περιλαμβάνει υλοποιήσεις διαφόρων αλγόριθμων εξόρυξης γνώσης καθώς και τεχνικές προεπεξεργασίας, μοντελοποίησης, αλλά και τεχνικές οπτικοποίησης των δεδομένων. Η διεπαφή της εφαρμογής χαρακτηρίζεται από τη φιλικότητα προς τον χρήστη, ενώ η ανάπτυξη σε Java διασφαλίζει την μεταφερσιμότητα (portability) σε διαφορετικές πλατφόρμες.

Η εφαρμογή προϋποθέτει την ύπαρξη των δεδομένο σε ένα απλό αρχείο (csv ή arff) όπου τα πεδία εγγραφής είναι χωρισμένα με κόμμα (,). Το WEKA δεν μπορεί να χειριστεί απευθείας δεδομένα Σχεσιακών Βάσεων (απαιτείται ο παραπάνω μετασχηματισμός τους). Είναι διαθέσιμο για δωρεάν εγκατάσταση από την ιστοσελίδα: http://www.cs.waikato.ac.nz/ml/weka [5].

3.5.1 Εφαρμογή του αλγορίθμου Apriori στο WEKA

Η ΣΤΡΑΤΗΓΙΚΗ ΥΛΟΠΟΙΗΣΗΣ ΤΟΥ ΑΛΓΟΡΙΘΜΟΥ APRIORI

Σε αυτό το σημείο, θα δείξουμε το τρόπο με τον οποίο εξαγάγαμε τους Κανόνες Συσχέτισης, χρησιμοποιώντας το WEKA και τον αλγόριθμο Apriori:

Αρχικά έπρεπε να μετατρέψουμε τη **Βάση Γνώσης** του **πίνακα 3** που έχουμε παραθέσει στο Κεφάλαιο 3 (ενότητα 3.1.2), σε μορφή τέτοια που να είναι αναγνώσιμη από το WEKA. Δημιουργώντας ένα νέο αρχείο απλού κειμένου (από το σημειωματάριο των Ms Windows) εισάγαμε τα δεδομένα του πίνακά μας στην παρακάτω μορφή:

```
@relation transactions
@attribute bread {Y,N}
@attribute coffee {Y,N}
@attribute milk {Y,N}
@attribute sugar {Y,N}
@data
Y, N, Y, N
N, Y, N, N
Y, N, Y, Y
N, Y, N, Y
Y, N, Y, Y
Y, Y, Y, N
Y, N, N, Y
Y, Y, Y, Y
N, N, Y, Y
Y, Y, N, Y
```

Στην νέα της μορφή, η Βάση Γνώσης μας περιλαμβάνει ορισμένες κωδικοποιημένες εκφράσεις. Η δήλωση "**@relation transactions**" είναι ένα αλφαριθμητικό που δηλώνει το όνομα της αναφοράς. Αμέσως μετά από κάθε δήλωση "**@attribute**" ακολουθεί το όνομα του χαρακτηριστικού και ο τύπος δεδομένων του. Κάθε ένας από αυτά στην πραγματικότητα αποτελεί και ένα κριτήριο, δηλαδή ένα πεδίο - τίτλο στήλης του πίνακας και έναν υποψήφιο παράγοντα που μπορεί να επηρεάσει την γενική ικανοποίηση του πελάτη. Τα κριτήρια αυτά τα αποκαλούμε "Attributes", δηλαδή ιδιότητες της Βάσης Γνώσης. Αμέσως μετά τον τίτλο της κάθε ιδιότητας εμφανίζονται μέσα σε άγκιστρα οι πιθανές τιμές της κάθε μεταβλητής, οι τιμές αυτές εκφράζουν τον τύπο δεδομένων που επιδέχεται η κάθε ιδιότητα. Εδώ οι τελεστές "Y" και "N" εκφράζουν τις αντίστοιχες τιμές "1" και "0" του πίνακα 3 .

Αμέσως μετά την έκφραση "**@data**" η πρώτη εγγραφή που αναγράφεται είναι η εξής: "Y, N, Y, N". Κάθε κόμμα που παρεμβάλλεται ανάμεσα στο εκάστοτε "Y" ή "N"

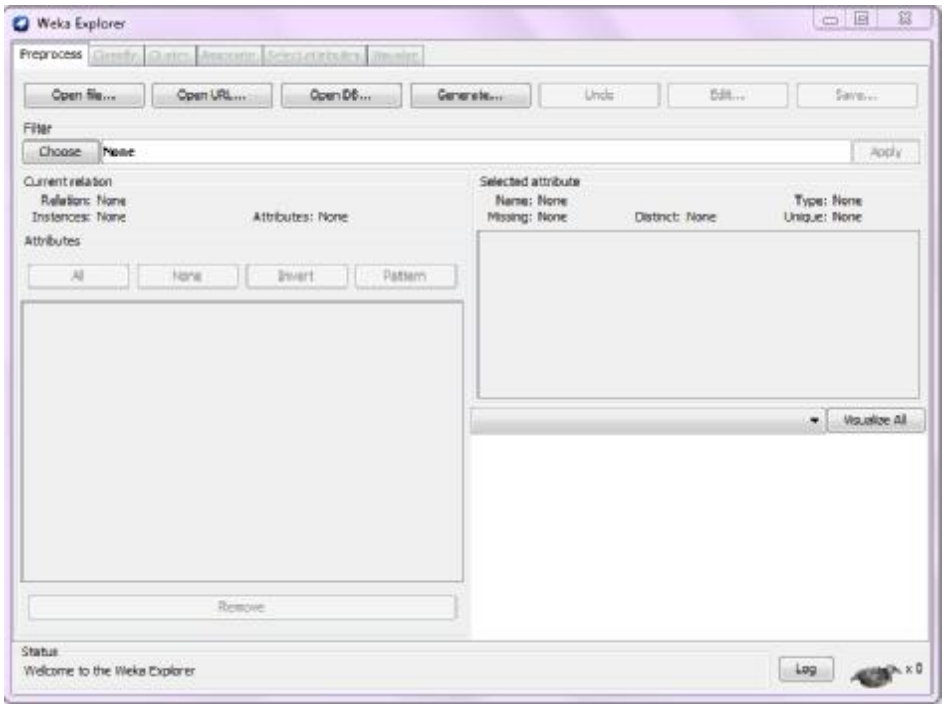
αντίστοιχα, αναπαριστά ένα διαχωριστικό μετάβασης σε επόμενη ιδιότητα - κριτήριο. Η σειρά αντιστοιχίας - αντιστοίχισης των ιδιοτήτων είναι η ίδια με αυτήν που έχουν δηλωθεί αρχικά. Συνεπώς, στο παράδειγμά μας, η πρώτη γραμμή απεικονίζει το 1^{ο} καλάθι αγορών και οι τελεστές "Y" και "N" τα προϊόντα που υπάρχουν μέσα σε αυτό.

Με βάση τα παραπάνω, αντιλαμβανόμαστε ότι για να οδηγηθούμε σε ένα τελικό σχηματισμό κανόνων, είναι μία διαδικασία που για να την κάνουμε μόνοι μας απαιτεί πάρα πολύ χρόνο (χρόνος ο οποίος αυξάνεται **εκθετικά** με το πλήθος των εγγραφών), τεράστια προσοχή και συνεχή επιτήρηση, δεδομένου ότι η Βάση Γνώσης υπό πραγματικές συνθήκες αποτελείται από μεγάλο όγκο εγγραφών οι οποίες διαρκώς αυξάνονται. Ευτυχώς, για αυτόν ακριβώς τον σκοπό δημιουργίας και **εξόρυξης** των κανόνων από μία Βάση Γνώσης, έχουν σχεδιαστεί λογισμικά που χρησιμοποιούν αλγόριθμους μέσω των οποίων εξάγονται άμεσα οι κανόνες και μας παραδίδονται έτοιμοι, γλιτώνοντάς μας από μία ιδιαίτερα χρονοβόρα και κοπιαστική εργασία. Ένα τέτοιο λογισμικό επιλέξαμε και εμείς και δεν είναι άλλο από το **WEKA** που αναφέραμε παραπάνω.

Τα αρχεία που περιέχουν το σύνολο των δεδομένων που επεξεργάζεται το WEKA για εξόρυξη πρέπει να έχουν συγκεκριμένο format και να αποθηκεύονται με την επέκταση "**.arff**". Αυτό σημαίνει πως το αρχείο απλού κειμένου txt που δημιουργήσαμε και αποτελεί την Βάση Γνώσης, επιβάλλεται να το αποθηκεύσουμε εκ νέου ώστε να αποκτήσει file type "arff". Έχοντας ολοκληρώσει αυτήν την διαδικασία τρέχουμε το WEKA και το αρχικό interface του εμφανίζεται στην εικόνα 13:

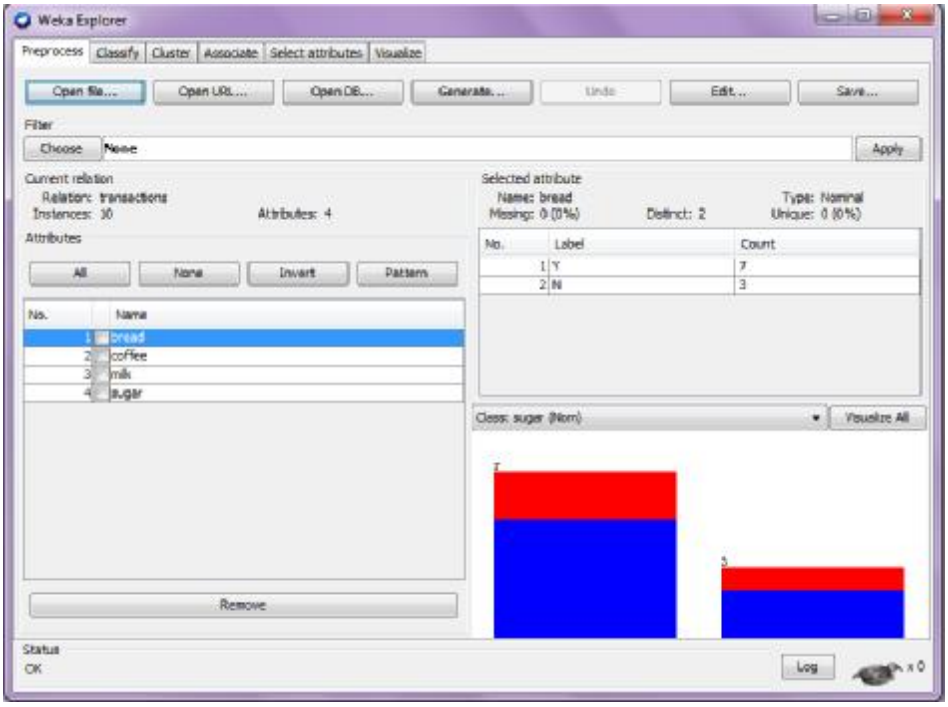
Εικόνα 13: Αρχικό interface του Weka

Επιλέγουμε το κουμπί "**Explorer**" και εμφανίζεται το παρακάτω παράθυρο (εικόνα 14):



Εικόνα 14: **Καρτέλα** Progress

Από το παράθυρο αυτό επιλέγουμε το κουμπί "**Open file...**" και αναζητούμε το arff αρχείο που περιέχει τη Βάση Γνώσης από την οποία θέλουμε να εξάγουμε τους Κανόνες Συσχέτισης. Ολοκληρώνοντας αυτήν την διαδικασία το παράθυρο διαμορφώνεται ως εξής (εικόνα 15):

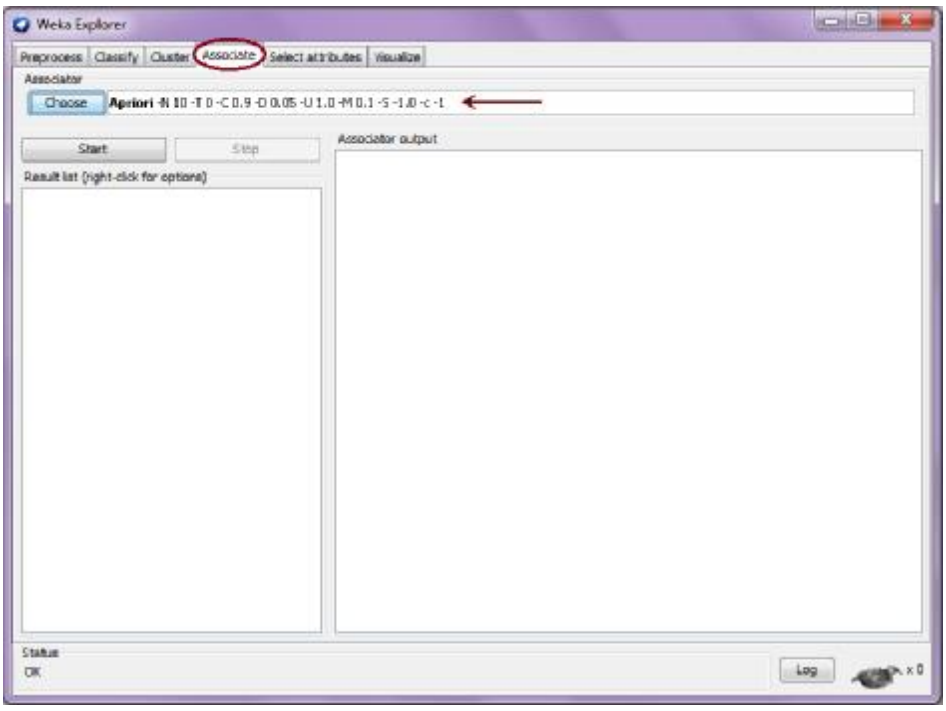


Εικόνα 15: **Εισαγωγή αρχείου του αλγορίθμου** Apriori

Παρατηρούμε πως, από τη στιγμή που φορτώνουμε το αρχείο arff στην εφαρμογή, το WEKA απευθείας απαριθμεί το πλήθος των εγγραφών της Βάσης Γνώσης

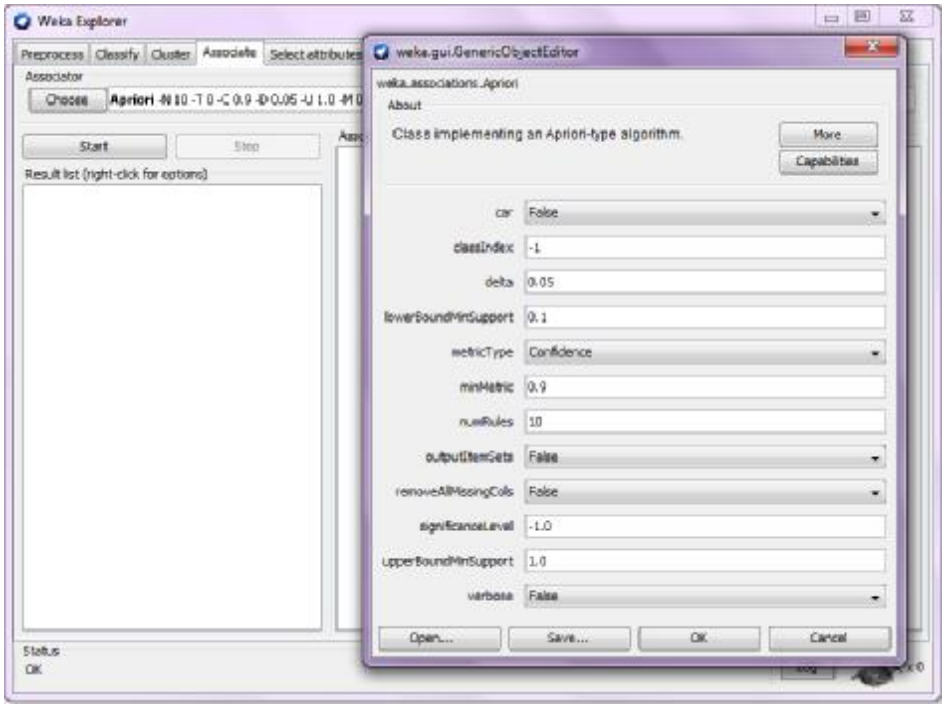
(**Instances: 10**), καθώς επίσης και το πλήθος των ιδιοτήτων που έχουμε θέσει (**Attributes: 4**). Αυτό βέβαια προϋποθέτει πως κάποιες εντολές - εκφράσεις τις έχουμε δώσει εμείς με τον κατάλληλο τρόπο ώστε να είναι αναγνώσιμες από το WEKA. Εκεί άλλωστε έγκειται και η δομή με την οποία έχουμε πληκτρολογήσει τα δεδομένα στο σημειωματάριο των Ms Windows. Ακριβώς κάτω από το πλήθος των ιδιοτήτων, παρατηρούμε πως εμφανίζεται μία λίστα με τους τίτλους που έχουμε αποδώσει σε κάθε ιδιότητα, συνοδευόμενοι από checkboxes. Στο πλαίσιο “Selected attribute” εμφανίζονται κάποια χαρακτηριστικά της επιλεγμένης ιδιότητας όπως το όνομα (name), ο τύπος (type), το ποσοστό των ελλιπών δεδομένων (missing), ο αριθμός των διαφορετικών τιμών (distinct) και τέλος, το πλήθος και ποσοστό των περιπτώσεων που έχουν τιμή χαρακτηριστικού διαφορετική από την τιμή των άλλων περιπτώσεων για την επιλεγμένη ιδιότητα (unique). Ακριβώς από κάτω, δίνεται ο πίνακας συχνότητας εμφάνισης των τιμών της ιδιότητας. Πατώντας το κουμπί “visualize all” εμφανίζονται σε ξεχωριστό παράθυρο τα ιστογράμματα όλων των ιδιοτήτων (attributes). Ακόμη, το WEKA μας δίνει την δυνατότητα να μην συμπεριλάβουμε όλες τις ιδιότητες στην ανάλυση, κλικάρωντας τις επιθυμητές προς διαγραφή ιδιότητες και πατώντας το κουμπί “remove” ή να επεξεργαστούμε τις τιμές αυτών με το κουμπί “edit”.

Σε αυτό το σημείο, όπου έχουν απαριθμηθεί οι εγγραφές και οι ιδιότητες, επιθυμούμε από το WEKA να μας εξάγει τους Κανόνες Συσχέτισης. Για τις ανάγκες εξόρυξης της Γνώσης, το WEKA περιλαμβάνει αρκετούς, ήδη εγκατεστημένους, αλγόριθμους, ανάλογα με τον τρόπο που θέλουμε να προβάλλουμε τα δεδομένα της Βάσης Γνώσης. Για να εξάγουμε, λοιπόν τους κανόνες θα χρησιμοποιήσουμε τον αλγόριθμο Apriori στον οποίο αναφερθήκαμε σε προηγούμενη ενότητα. Παραθέτουμε παρακάτω καρέ - καρέ τον τρόπο επιλογής του αλγόριθμου μέσω του WEKA:



Εικόνα 16: Καρτέλα Associate

Επιλέγουμε από τον παράθυρο του Explorer την καρτέλα "**Associate**", εξ' ορισμού επιλέγετε ο αλγόριθμος Apriori (εικόνα 17).

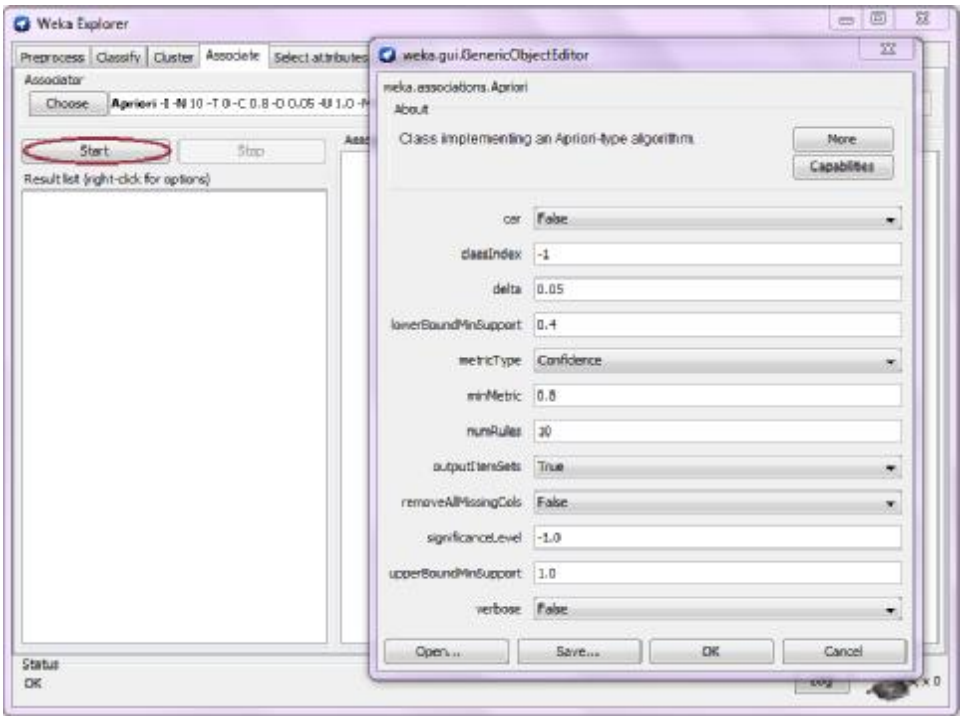


Εικόνα 17: **Πλαίσιο διαλόγου** "GenericObjectEditor" **αλγορίθμου** Apriori

Στη συνέχεια κάνοντας αριστερό κλίκ στην περιοχή όπου αναγράφετε ο επιλεγμένος αλγόριθμος, εμφανίζεται το πλαίσιο διαλόγου "**GenericObjectEditor** " (εικόνα 15).

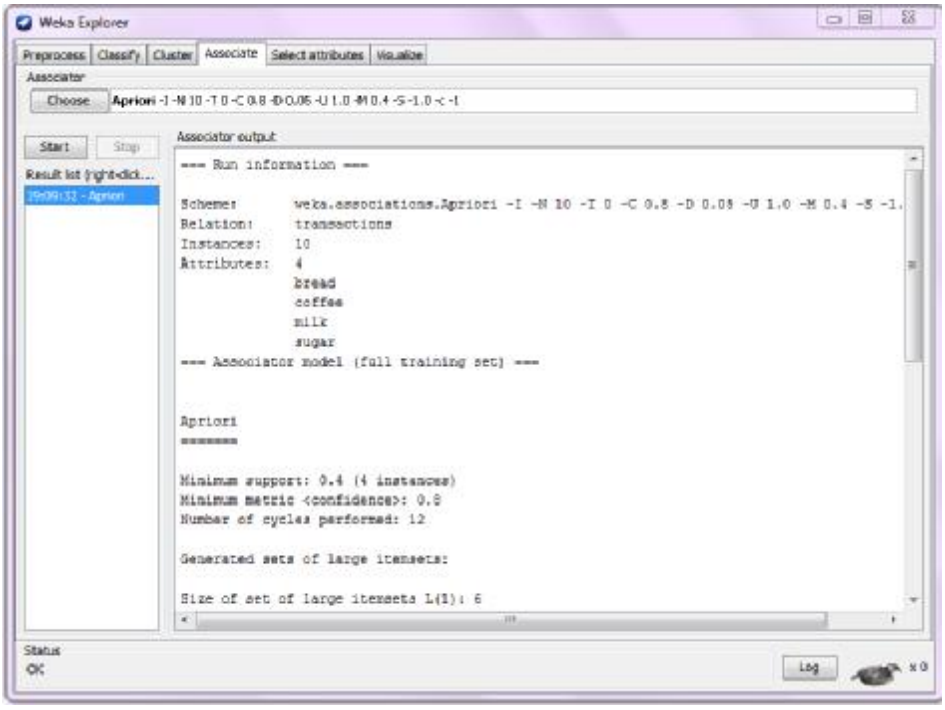
Εμφανίζονται κάποιες επιλογές, ορισμένες από τις οποίες είναι η ελάχιστη **υποστήριξη** (min **support**) "**upperBoundMinSupport**" με προεπιλεγμένη τιμή "**1.0**", δηλαδή το 100% των στοιχείων των δεδομένων. Η υποστήριξη μειώνεται με βήμα 5% (**delta=0.05**) έως ότου **είτε** βρεθούν 10 κανόνες (**numRules=10**) με απαιτούμενη ελάχιστη **εμπιστοσύνη** (**minimum confidence**) 90% (**minMetric=0.9**) είτε η ελάχιστη υποστήριξη φτάσει σε ένα κάτω όριο 10% (**lowerBoundMinSupport = 0.1**). Υπάρχουν τέσσερις διαφορετικοί τύπου μετρικής (**Confidence**, Lift, Leverage, Conviction) – εξ'ορισμού χρησιμοποιείται η **metricType=Confidence**. Τέλος, εάν επιλέξουμε **outputItemSets = True** τότε εμφανίζονται όλα τα στοιχειοσύνολα μαζί με το πλήθος εμφάνισής τους στο σύνολο των δεδομένων.

Στο συγκεκριμένο παράδειγμα οι επιλογές θα γίνουν ως εξής, το ελάχιστο όριο υποστήριξης (lowerBoundMinSupport) θα είναι ίσο με 0.4, η ελάχιστη εμπιστοσύνη (minMetric) θα πάρει την τιμή 0.8 και στην επιλογή outputItemSets επιλέγουμε True. Στην συνέχεια πατάμε το κουμπί "**Start**" (εικόνα 18):



Εικόνα 18: Έναρξη αλγορίθμου Apriori

Μετά την εκτέλεση του αλγορίθμου τα αποτελέσματα εμφανίζονται στο παράθυρο "**Associator output**" (εικόνα 19)



Εικόνα 19: Έξοδος αλγορίθμου Apriori

Η έξοδος του αλγορίθμου είναι η εξής:

```
=== Run information ===

Scheme:       weka.associations.Apriori -I -N 10 -T 0 -C 0.8 -D 0.05 -U 1.0 -M 0.4 -S -1.0 -c -1
Relation:     transactions
Instances:    10
Attributes:   4
              bread
              coffee
              milk
              sugar
=== Associator model (full training set) ===


Apriori
=======

Minimum support: 0.4 (4 instances)
Minimum metric <confidence>: 0.8
Number of cycles performed: 12

Generated sets of large itemsets:

Size of set of large itemsets L(1): 6

Large Itemsets L(1):
bread=Y 7
coffee=Y 5
coffee=N 5
milk=Y 6
milk=N 4
sugar=Y 7

Size of set of large itemsets L(2): 6

Large Itemsets L(2):
bread=Y coffee=N 4
bread=Y milk=Y 5
bread=Y sugar=Y 5
coffee=N milk=Y 4
coffee=N sugar=Y 4
milk=Y sugar=Y 4

Best rules found:

 1. milk=Y 6 ==> bread=Y 5    conf:(0.83)
 2. coffee=N 5 ==> bread=Y 4    conf:(0.8)
 3. coffee=N 5 ==> milk=Y 4    conf:(0.8)
 4. coffee=N 5 ==> sugar=Y 4    conf:(0.8)
```

Πιο συγκεκριμένα, ο αλγόριθμος Apriori έτρεξε 12 φορές για να παραχθούν οι κανόνες με διαφορετικές τιμές για το minimum support. (Number of cycles performed = 12). Παράγονται δύο στοιχειοσύνολα L1 και L2 με 6 στοιχεία το καθένα. Αυτά περιλαμβάνουν όλα τα στοιχεία τα οποία παρουσιάζουν ελάχιστη υποστήριξη (min support) μεγαλύτερη του 40%. Τέλος εμφανίζονται οι καλύτεροι κανόνες συσχέτισης (Best rules found) δηλαδή αυτοί που έχουν εμπιστοσύνη τουλάχιστον 80%. Μας

ενδιαφέρει να κρατήσουμε αυτούς όπου και τα δύο προϊόντα περιέχονται στο καλάθι αγορών, τέτοιος κανόνας είναι ο πρώτος με εμπιστοσύνη 83% (**milk=Y 6 ==> bread=Y 5 conf:(0.83)**).

3.5.2 Εφαρμογή του αλγορίθμου FP-Growth στο WEKA

Για να μπορέσουμε να χρησιμοποιήσουμε τον αλγόριθμο FP-Growth μέσω της εφαρμογής "WEKA", θα πρέπει το arff αρχείο που χρησιμοποιήθηκε για τον apriori στην προηγούμενη ενότητα, να **τροποποιηθεί** κατάλληλα. Συγκεκριμένα, θα πρέπει οι τιμές των μεταβλητών, που εδώ αντιστοιχούν στα προϊόντα των συναλλαγών (transactions' items) να είναι boolean type, δηλαδή, να φέρουν την τιμή **true** (αν το προϊόν υπάρχει στο καλάθι) ή **false** (αν δεν υπάρχει), σε αντίθεση με τον apriori, όπου αντιστοίχως βάζαμε Y (Yes αν υπάρχει) ή N (No αν δεν υπάρχει).

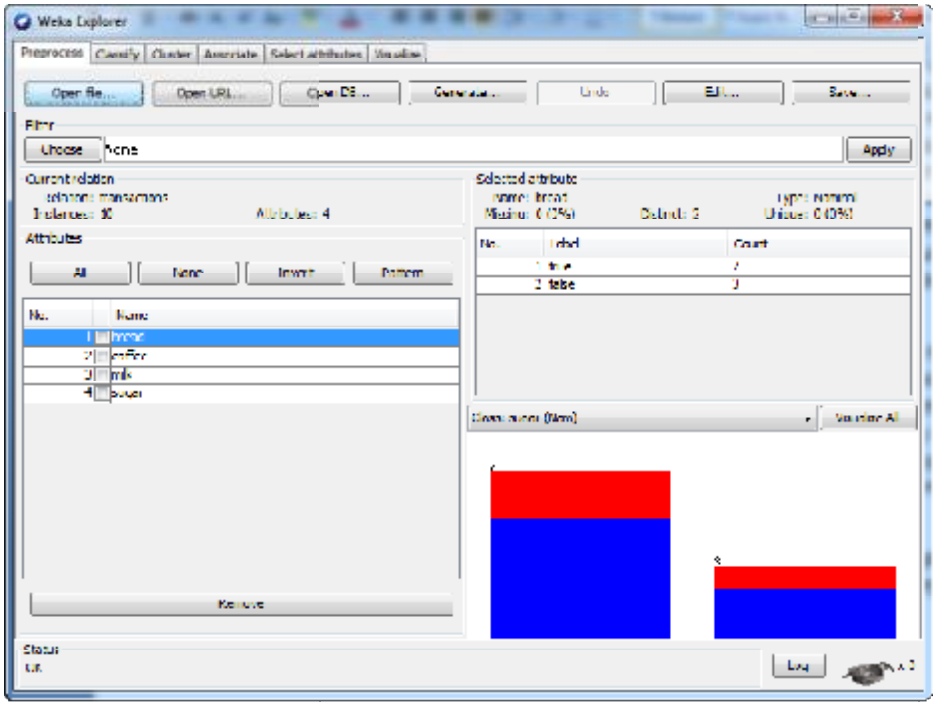
Για να είναι ουσιαστικότερη η σύγκριση μεταξύ των δύο αλγορίθμων (fp-growth και apriori) θα κάνουμε χρήση του ίδιου παραδείγματος, δηλαδή του ίδιου arff αρχείου, κάνοντας τις τροποποιήσεις που αναφέραμε στην προηγούμενη παράγραφο. Εκ νέου, λοιπόν, μετατρέπουμε τη **Βάση Γνώσης** του **πίνακα 3** που έχουμε παραθέσει στο Κεφάλαιο 3 (ενότητα 3.1.2), σε μορφή τέτοια που να είναι αναγνώσιμη από το WEKA, αυτή τη φορά για τον αλγόριθμο FP-Growth. Δημιουργώντας ένα νέο αρχείο απλού κειμένου (από το σημειωματάριο των Ms Windows) εισάγαμε τα δεδομένα του πίνακά μας στην παρακάτω μορφή:

```
@relation transactions
@attribute bread {true,false}
@attribute coffee {true,false}
@attribute milk {true,false}
@attribute sugar {true,false}

@data
true, false, true, false
false, true, false, false
true, false, true, true
false, true, false, true
true, false, true, true
true, true, true, false
true, false, false, true
true, true, true, true
false, false, true, true
true, true, false, true
```

Στην νέα της μορφή, η Βάση Γνώσης μας περιλαμβάνει σχεδόν τις ίδιες κωδικοποιημένες εκφράσεις με την αντίστοιχη για τον apriori. Η δήλωση "@relation transactions" είναι ένα αλφαριθμητικό που δηλώνει το όνομα της αναφοράς. Αμέσως μετά από κάθε δήλωση @attribute ακολουθεί το όνομα του χαρακτηριστικού και ο τύπος δεδομένων του. Τα κριτήρια αυτά τα αποκαλούμε "Attributes", δηλαδή ιδιότητες της Βάσης Γνώσης. Αμέσως μετά τον τίτλο της κάθε ιδιότητας εμφανίζονται μέσα σε άγκιστρα οι πιθανές τιμές της κάθε μεταβλητής, οι τιμές αυτές εκφράζουν τον τύπο δεδομένων που επιδέχεται η κάθε ιδιότητα. Εδώ οι τελεστές "**true**" και "**false**" εκφράζουν τις αντίστοιχες τιμές "1" και "0" του πίνακα 3 .

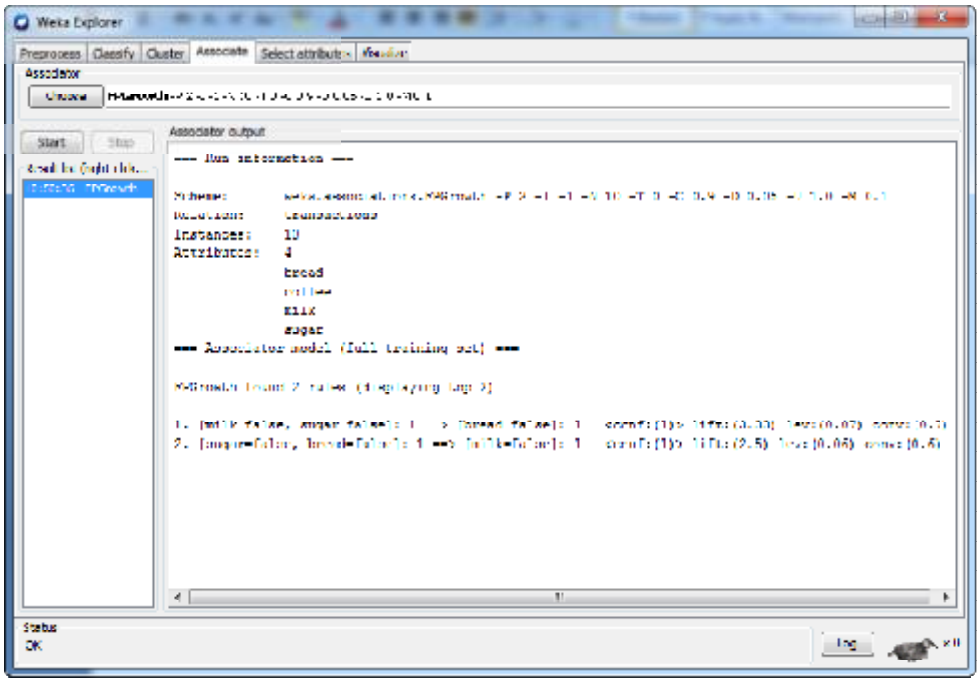
Αμέσως μετά την έκφραση "@data" η πρώτη εγγραφή που αναγράφεται είναι η εξής: "true, false, true, false". Κάθε κόμμα που παρεμβάλλεται ανάμεσα στο εκάστοτε "true" ή "false" αντίστοιχα, αναπαριστά ένα διαχωριστικό μετάβασης σε επόμενη ιδιότητα - κριτήριο. Η σειρά αντιστοιχίας - αντιστοίχισης των ιδιοτήτων είναι η ίδια με αυτήν που έχουν δηλωθεί αρχικά. Συνεπώς, στο παράδειγμά μας, η πρώτη γραμμή απεικονίζει το $1^{ο}$ καλάθι αγορών και οι τελεστές "true" και "false" τα προϊόντα που υπάρχουν (ή δεν υπάρχουν) μέσα σε αυτό.

Φορτώνουμε το, ειδικά διαμορφωμένο για τον αλγόριθμο fp-growth.arff αρχείο στο WEKA με βάση τη διαδικασία που περιγράψαμε στην προηγούμενη ενότητα και έχουμε το εξής αποτέλεσμα (εικόνα 20):



Εικόνα 20: **Εισαγωγή αρχείου για τον αλγόριθμο** FP-Growth

Πηγαίνουμε εκ νέου στην καρτέλα "Associate" και αυτή τη φορά επιλέγουμε τον αλγόριθμο FP-Growth. Τρέχουμε τον αλγόριθμο και το αποτέλεσμα φαίνεται στην εικόνα 21:



Εικόνα 21: Έξοδος αλγορίθμου FP-Growth

Παρατηρούμε πως παράγονται δύο κανόνες:

1. Αν ο πελάτης **ΔΕΝ** αγοράσει γάλα **και** ζάχαρη, τότε **ΔΕΝ** θα αγοράσει ψωμί
2. Αν ο πελάτης **ΔΕΝ** αγοράσει ζάχαρη **και** ψωμί, τότε **ΔΕΝ** θα αγοράσει γάλα

Οι δύο αυτοί κανόνες εξάγονται *αν δεν πειράξουμε τις παραμέτρους του αλγορίθμου*. Έστω, τώρα, ότι θέτουμε τις ίδιες τιμές παραμέτρων που είχαμε θέσει στο παράδειγμα με τον apriori, δηλαδή το ελάχιστο όριο υποστήριξης (**lowerBoundMinSupport**) να είναι ίσο με 0.4, η ελάχιστη εμπιστοσύνη (**minMetric**) να έχει την τιμή 0.8. Τότε το αποτέλεσμα φαίνεται παρακάτω:

```
=== Run information ===

Scheme:       weka.associations.FPGrowth -P 2 -I -1 -N 10 -T 0 -C 0.8 -D 0.05 -U 1.0 -M 0.4 -S
Relation:     transactions
Instances:    10
Attributes:   4
              bread
              coffee
              milk
              sugar
=== Associator model (full training set) ===

No rules found!
```

Παρατηρούμε πως, για τις ίδιες παραμέτρους, ο FP-Growth δεν εξάγει κανόνες.

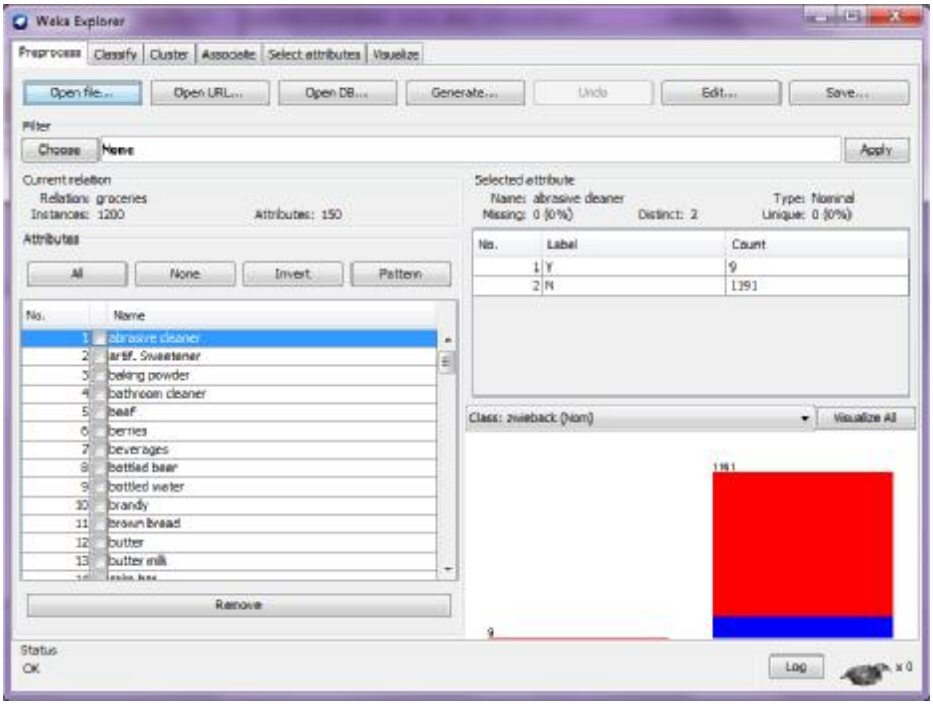
3.5.3 Εφαρμογή Apriori και FP-Growth για μεγαλύτερα Training Sets

Στις δύο προηγούμενες ενότητες χρησιμοποιήσαμε πειραματικά στο WEKA ένα Training Set **δέκα** εγγραφών (και άρα δέκα καλαθιών αγοράς) για **τέσσερα** διαφορετικά προϊόντα. Με βάση αυτό το Training Set επιλέξαμε τους αλγόριθμους Apriori και FP-Growth με σκοπό την εξαγωγή Κανόνων Συσχέτισης και καταγράψαμε τα συμπεράσματά μας.

Είναι, όμως, γεγονός πως ένα Training Set δέκα μόνο εγγραφών *δεν αντιπροσωπεύει ρεαλιστικά μία πραγματική Βάση Δεδομένων μιας επιχείρησης*. Για το λόγο αυτό, μέσω διαδικτύου [12], αναζητήσαμε και βρήκαμε δύο πιο ρεαλιστικές Βάσεις Δεδομένων με πολύ περισσότερα προϊόντα και, φυσικά, πολύ περισσότερες εγγραφές.

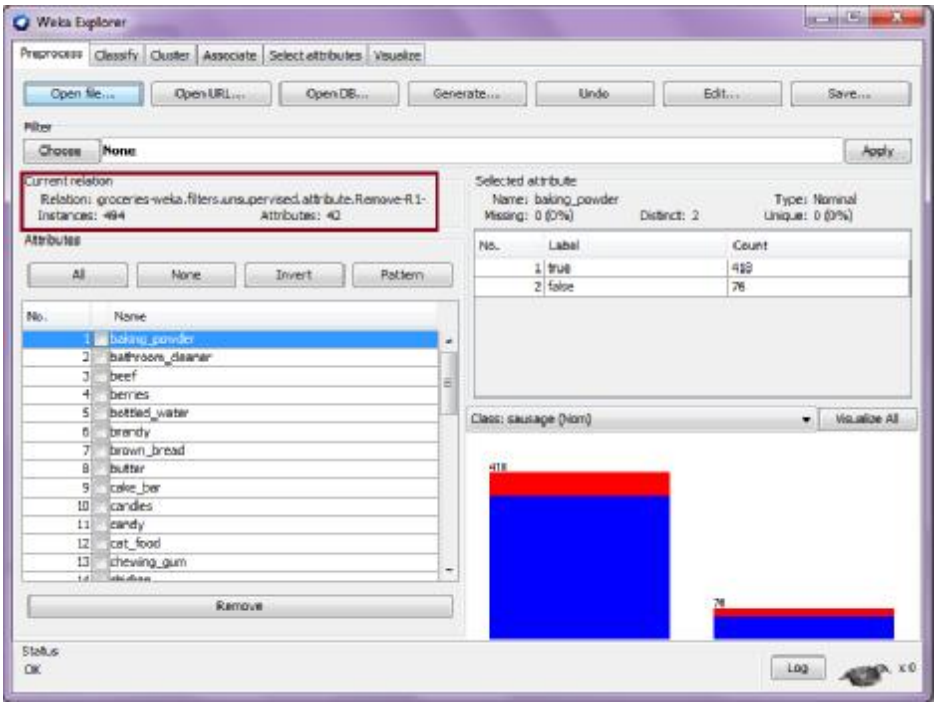
Η πρώτη Βάση με την οποία θα ασχοληθούμε καλείται “groceries” και είναι διαθέσιμη στην ιστοσελίδα *https://github.com/stedy/Machine-Learning-with-R-datasets* με την κατάληξη “.csv”. Το σύνολο αυτό περιλαμβάνει **9835 συναλλαγές** από τις οποίες κρατήσαμε τις πρώτες **1200** και τα **150 προϊόντα** που εμφανίζονται σε αυτές.

Ακολουθώντας την διαδικασία που προ αναφέραμε, φορτώνουμε την Βάση στο WEKA (εικόνα 22).



Εικόνα 22: training set “groceries.arff” (Y,N)

Αρχικά τροποποιούμε τα δεδομένα και με το κουμπί **“save”** αποθηκεύουμε την βάση ως καινούργιο αρχείο.



Εικόνα 23: **νέο** training set “groceries_TF.arff”

Όπως φαίνεται και στην παραπάνω εικόνα, **το νέο μας Training Set** αποτελείται από **494** καλάθια αγοράς (βλ. Instances) και η επιχείρηση εμπορεύεται **42** διαφορετικά προϊόντα. Η δομή του νέου μας αρχείου με όνομα "groceries_TF.arff" που εισήχθη στο WEKA είναι διαμορφωμένη έτσι ώστε η παρουσία ενός προϊόντος στο καλάθι αγοράς να φέρει την τιμή "true" και, αντιστοίχως, η απουσία ενός προϊόντος από το καλάθι αγοράς να φέρει την τιμή "false"

Επιλέγοντας, λοιπόν, τον **Apriori** για την εξαγωγή Κανόνων Συσχέτισης (ορίζοντας εκ νέου ελάχιστη εμπιστοσύνη στο 80%) με βάση το νέο Training Set έχουμε:

```
Minimum support: 0.95 (469 instances)
Minimum metric <confidence>: 0.8
Number of cycles performed: 1

Generated sets of large itemsets:

Size of set of large itemsets L(1): 11

Size of set of large itemsets L(2): 34

Size of set of large itemsets L(3): 46

Size of set of large itemsets L(4): 29

Size of set of large itemsets L(5): 7
```

```
Best rules found:

 1. mustard=true 494 ==> cake_bar=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 2. cake_bar=true 494 ==> mustard=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 3. popcorn=true 494 ==> cake_bar=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 4. cake_bar=true 494 ==> popcorn=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 5. salt=true 494 ==> cake_bar=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 6. cake_bar=true 494 ==> salt=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 7. popcorn=true 494 ==> mustard=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 8. mustard=true 494 ==> popcorn=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 9. salt=true 494 ==> mustard=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
10. mustard=true 494 ==> salt=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
11. salt=true 494 ==> popcorn=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
12. popcorn=true 494 ==> salt=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
13. mustard=true popcorn=true 494 ==> cake_bar=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
14. cake_bar=true popcorn=true 494 ==> mustard=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
15. cake_bar=true mustard=true 494 ==> popcorn=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
16. popcorn=true 494 ==> cake_bar=true mustard=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
17. mustard=true 494 ==> cake_bar=true popcorn=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
18. cake_bar=true 494 ==> mustard=true popcorn=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
19. mustard=true salt=true 494 ==> cake_bar=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
20. cake_bar=true salt=true 494 ==> mustard=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
21. cake_bar=true mustard=true 494 ==> salt=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
22. salt=true 494 ==> cake_bar=true mustard=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
23. mustard=true 494 ==> cake_bar=true salt=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
24. cake_bar=true 494 ==> mustard=true salt=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
25. popcorn=true salt=true 494 ==> cake_bar=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
26. cake_bar=true salt=true 494 ==> popcorn=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
27. cake_bar=true popcorn=true 494 ==> salt=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
28. salt=true 494 ==> cake_bar=true popcorn=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
29. popcorn=true 494 ==> cake_bar=true salt=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
30. cake_bar=true 494 ==> popcorn=true salt=true 494    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
```

Παρατηρούμε με βάση τα παραπάνω αποτελέσματα ότι εξάγονται **30 κανόνες** (*έχουμε ήδη θέσει μέγιστο όριο την εξαγωγή maximum τριάντα κανόνων*). Μεταφράζοντας τους κανόνες που έχουν εξαχθεί από το WEKA χρησιμοποιώντας τον Apriori έχουμε τα εξής χρήσιμα συμπεράσματα:

- **Κανόνας 13**: Αν ο πελάτης αγοράσει μουστάρδα και ποπ κορν, τότε θα αγοράσει και κέικ. Από τους κανόνες 14 και 15 προκύπτει πως οποιοσδήποτε συνδυασμός δύο εκ των τριών παραπάνω προϊόντων σε ένα καλάθι, θα έχει ως αποτέλεσμα και την παρουσία του τρίτου προϊόντος στο ίδιο καλάθι
- **Κανόνας 20**: Αν ο πελάτης αγοράσει μουστάρδα και αλάτι, τότε θα αγοράσει και κέικ. Από τους κανόνες 21 και 22 προκύπτει πως οποιοσδήποτε συνδυασμός δύο εκ των τριών παραπάνω προϊόντων σε ένα καλάθι, θα έχει ως αποτέλεσμα και την παρουσία του τρίτου προϊόντος στο ίδιο καλάθι
- Όλοι οι υπόλοιποι κανόνες εμπεριέχουν (σε μικρότερα στοιχειοσύνολα) το συμπέρασμα των προαναφερθέντων κανόνων

Συνεπώς, καταλαβαίνουμε πως *κάθε κανόνας δεν οδηγεί από μόνος του και σε ένα συμπέρασμα*, αλλά αντιθέτως τα χρήσιμα συμπεράσματα προκύπτουν από μία **συνολικότερη επισκόπηση των κανόνων**.

Χρησιμοποιώντας **το ίδιο Training Set**, αλλά επιλέγοντας ως αλγόριθμο εξαγωγής κανόνων τον **FP-Growth**, παίρνουμε το αποτέλεσμα που φαίνεται παρακάτω:

```
FPGrowth found 1 rules (displaying top 1)

1. [chicken=false]: 57 ==> [butter=false]: 57   <conf:(1)> lift:(5.2) lev:(0.09) conv:(46.04)
```

Επομένως, σε αντίθεση με τον Apriori όπου είχαμε 30 κανόνες (θα είχαμε περισσότερους αν είχαμε αυξήσει το όριο της σχετικής παραμέτρου του WEKA), χρησιμοποιώντας τον **FP-Growth** έχουμε την εξαγωγή **ενός μόνο κανόνα** (διατηρώντας τις ίδιες ακριβώς παραμέτρους που είχαμε θέσει και για τον apriori). Ο κανόνας αυτός μεταφράζεται ως εξής:

- Αν σε ένα καλάθι αγοράς **δεν** υπάρχει σαν προϊόν το κοτόπουλο, τότε **δεν** θα υπάρχει και το βούτυρο

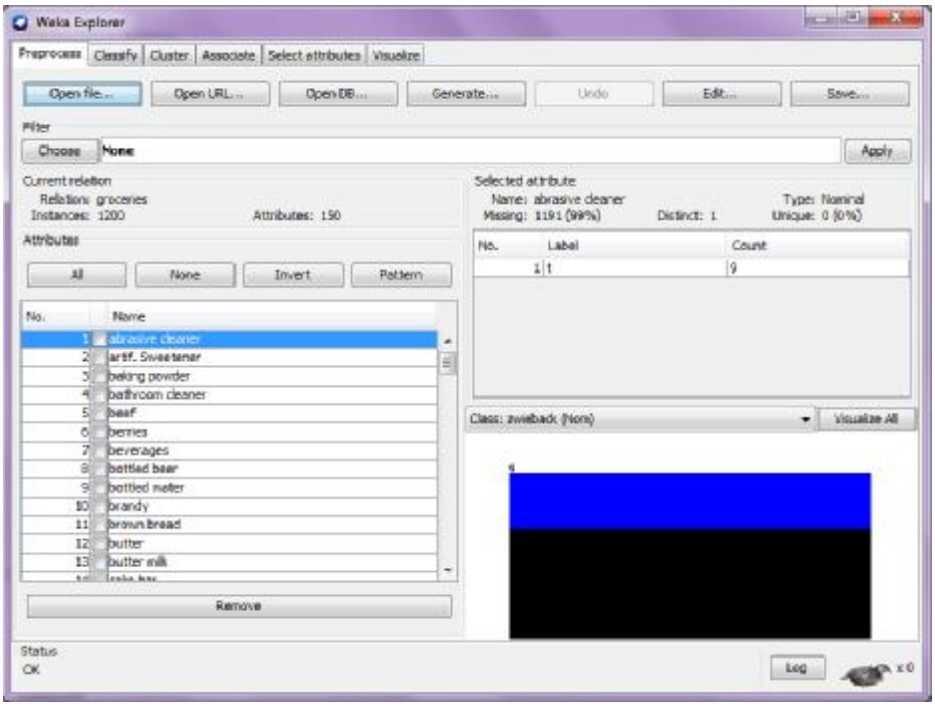
Συνεπώς, επιβεβαιώνεται εκ νέου πως ο FP-Growth λειτουργεί **συμπληρωματικά** σε σχέση με τον Apriori. Εκεί που ο **Apriori** δημιουργεί συσχετίσεις **παρουσίας** προϊόντων στο ίδιο καλάθι, ο **FP-Growth** δημιουργεί συσχετίσεις **απουσίας** προϊόντων από ένα καλάθι αγοράς.

Εάν επιλέγαμε να φορτώσουμε ολόκληρη την Βάση στο WEKA, πριν την επεξεργαστούμε, θα δαπανούσαμε αρκετό χρόνο προκειμένου να βρεθούν οι συσχετίσεις και λόγο των ελάχιστων θετικών τιμών (true) που υπάρχουν, η έξοδος του αλγορίθμου θα εμφάνιζε μόνο συσχετίσεις προϊόντων που δεν υπάρχουν στο καλάθι αγοράς. Για το λόγο αυτό αντικαθιστούμε τις τιμές “false” με το σύμβολο “?”. Με αυτό τον τρόπο οι τιμές δηλώνονται ως χαμένες (missing values) και ο αλγόριθμος δεν τις συμπεριλαμβάνει στην εύρεση συσχετίσεων.

Το αρχείο θα έχει την μορφή:

```
@relation groceries
@attribute "abrasive cleaner" { t}
@attribute "artif. Sweetener" { t}
@attribute "baking powder" { t}
@attribute "bathroom cleaner" { t}
@attribute beef { t}
@attribute berries { t}
[…]

@data
?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,t,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?
,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,
t,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,t,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?
?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,? […]
```

Έπειτα φορτώνουμε την τροποποιημένη Βάση στο WEKA (εικόνα 24):



Εικόνα 24: groceries.arff missing values

Στην καρτέλα "**Associate**" επιλέγουμε τον αλγόριθμο Apriori και ορίζουμε τις παραμέτρους ως εξής: υποστήριξη 1% (**lowerBoundMinSupport=0.01**), εμπιστοσύνη 50% (**minMetric=0.5**) και τον αριθμό των κανόνων ίσο με 30 (**numRules=30**).

Τα αποτελέσματα δίνονται παρακάτω.

```
Apriori
=======

Minimum support: 0.01 (12 instances)
Minimum metric <confidence>: 0.5
Number of cycles performed: 20

Generated sets of large itemsets:

Size of set of large itemsets L(1): 84

Size of set of large itemsets L(2): 230

Size of set of large itemsets L(3): 44

Best rules found:

1. turkey=t whisky=t 15 ==> yogurt=t 12    conf:(0.8)
2. citrus fruit=t whisky=t 18 ==> yogurt=t 14    conf:(0.78)
3. newspapers=t zwieback=t 18 ==> yogurt=t 13    conf:(0.72)
4. domestic eggs=t zwieback=t 17 ==> yogurt=t 12    conf:(0.71)
5. packaged fruit/vegetables=t sweet spreads=t 17 ==> yogurt=t 12
conf:(0.71)
```

```
6. curd=t whisky=t 19 ==> yogurt=t 13    conf:(0.68)
7. whisky=t zwieback=t 25 ==> yogurt=t 17    conf:(0.68)
8. whisky=t zwieback=t 25 ==> packaged fruit/vegetables=t 16
conf:(0.64)
9. fruit/vegetable juice=t zwieback=t 27 ==> yogurt=t 17 conf:(0.63)
10. frozen vegetables=t packaged fruit/vegetables=t 24 ==> yogurt=t 15
conf:(0.63)
11. rubbing alcohol=t whisky=t 23 ==> yogurt=t 14    conf:(0.61)
12. mayonnaise=t root vegetables=t 22 ==> yogurt=t 13    conf:(0.59)
13. pet care=t zwieback=t 21 ==> yogurt=t 12    conf:(0.57)
14. curd=t zwieback=t 30 ==> yogurt=t 17    conf:(0.57)
15. newspapers=t yogurt=t 23 ==> zwieback=t 13    conf:(0.57)
16. meat spreads=t 25 ==> packaged fruit/vegetables=t 14    conf:(0.56)
17. fruit/vegetable juice=t zwieback=t 27 ==> packaged
fruit/vegetables=t 15    conf:(0.56)
18. turkey=t zwieback=t 29 ==> yogurt=t 16    conf:(0.55)
19. fruit/vegetable juice=t packaged fruit/vegetables=t 31 ==> yogurt=t
17 conf:(0.55)
20. packaged fruit/vegetables=t popcorn=t 22 ==> yogurt=t 12
conf:(0.55)
21. sweet spreads=t yogurt=t 22 ==> packaged fruit/vegetables=t 12
conf:(0.55)
22. ham=t 28 ==> yogurt=t 15    conf:(0.54)
23. frozen vegetables=t yogurt=t 28 ==> packaged fruit/vegetables=t 15
conf:(0.54)
24. packaged fruit/vegetables=t whisky=t 40 ==> yogurt=t 21
conf:(0.53)
25. popcorn=t yogurt=t 23 ==> packaged fruit/vegetables=t 12
conf:(0.52)
26. rubbing alcohol=t whisky=t 23 ==> packaged fruit/vegetables=t 12
conf:(0.52)
27. frozen vegetables=t 54 ==> yogurt=t 28    conf:(0.52)
28. rubbing alcohol=t zwieback=t 33 ==> yogurt=t 17    conf:(0.52)
29. packaged fruit/vegetables=t zwieback=t 49 ==> yogurt=t 25
conf:(0.51)
30. butter=t 60 ==> yogurt=t 30    conf:(0.5)
```

Ο αλγόριθμος χρειάστηκε 20 επαναλήψεις και παρήγαγε τρία στοιχειοσύνολα L(1), L(2) και L(3) με 84, 230 και 44 στοιχεία αντίστοιχα. Η εμπιστοσύνη των κανόνων ξεκινάει από 80% (conf:(0.8)) έως 50% (conf:(0.5)) που είναι και το κατώτατο όριο που θέσαμε. Από τους 30 κανόνες παρατηρούμε ότι οι 22 από αυτούς έχουν δεξιό μέλος του κανόνα (RHS) το “yogurt”, οι 8 έχουν RHS “packaged fruit/vegetables” και ένας κανόνας έχει RHS το “zwieback”.

Κάποια χρήσιμα συμπεράσματα που μπορούμε να εξάγουμε είναι τα εξής:

- **Κανόνας 1:** Εάν κάποιος αγοράσει γαλοπούλα (turkey) **και** ουίσκι (whiskey) είναι πιθανό να αγοράσει και γιαούρτι (yogurt) με πιθανότητα 80%.
- **Κανόνας 2:** Κάποιος που αγοράζει εσπεριδοειδή (citrus fruit) και ουίσκι έχει πιθανότητα 78% να αγοράσει και γιαούρτι.
- **Κανόνας 3:** Ο κανόνας αυτός σχετίζετε με τον **κανόνα 15**. Στον πρώτο όταν κάποιος αγοράζει εφημερίδα (newspaper) και φρυγανιές (zwieback) θα

αγοράσει και γιαούρτι με πιθανότητα 72%. Ενώ με το συνδυασμό {εφημερίδα, γιαούρτι} θα αγοράσει και φρυγανιές με πιθανότητα 57%.

- Για τους **κανόνες 7** και **8** ο συνδυασμός {ουίσκι, φρυγανιές} θα έχει ως συνέπεια την ύπαρξη στο καλάθι αγοράς και του γιαουρτιού με πιθανότητα 68% ή συσκευασμένων φρούτων/λαχανικών (packaged fruit/vegetables) με πιθανότητα 64%

Από το σύνολο των κανόνων μπορεί να απαντηθεί η ερώτηση *«Τι αγοράζουν οι πελάτες πριν επιλέξουν να αγοράσουν γιαούρτι;»*. Βλέπουμε ότι τα ποιο συχνά προϊόντα στο καλάθι είναι το ουίσκι, όπου εμφανίζετε 8 φορές και το γιαούρτι όπου εμφανίζεται 10 φορές. Επομένως 10 στα 30 καλάθια έχουν ένα από τα δύο προϊόντα στο αριστερό μέλος του κανόνα (LHS) τις φρυγανιές και 8 στα 30 το ουίσκι. Από αυτούς οι 6 στους 8 κανόνες που έχουν ουίσκι στο LHS έχουν και γιαούρτι στο RHS και 9 στους 10 κανόνες που έχουν φρυγανιές στο LHS έχουν και γιαούρτι στο RHS.

Εκτελούμε τον FP-Growth με τις ίδιες παραμέτρους. Η έξοδος του αλγορίθμου φαίνετε παρακάτω.

```
FPGrowth found 29 rules (displaying top 29)

1. [turkey=t, whisky=t]: 15 ==> [yogurt=t]: 12   <conf:(0.8)>
lift:(3.21) lev:(0.01) conv:(2.82)
2. [citrus fruit=t, whisky=t]: 18 ==> [yogurt=t]: 14   <conf:(0.78)>
lift:(3.12) lev:(0.01) conv:(2.7)
3. [zwieback=t, newspapers=t]: 18 ==> [yogurt=t]: 13   <conf:(0.72)>
lift:(2.9) lev:(0.01) conv:(2.25)
4. [packaged fruit/vegetables=t, sweet spreads=t]: 17 ==> [yogurt=t]:
12   <conf:(0.71)> lift:(2.83) lev:(0.01) conv:(2.13)
5. [zwieback=t, domestic eggs=t]: 17 ==> [yogurt=t]: 12   <conf:(0.71)>
lift:(2.83) lev:(0.01) conv:(2.13)
6. [whisky=t, curd=t]: 19 ==> [yogurt=t]: 13   <conf:(0.68)>
lift:(2.75) lev:(0.01) conv:(2.04)
7. [zwieback=t, whisky=t]: 25 ==> [yogurt=t]: 17   <conf:(0.68)>
lift:(2.73) lev:(0.01) conv:(2.09)
8. [zwieback=t, whisky=t]: 25 ==> [packaged fruit/vegetables=t]: 16
<conf:(0.64)> lift:(3.35) lev:(0.01) conv:(2.02)
9. [zwieback=t, fruit/vegetable juice=t]: 27 ==> [yogurt=t]: 17
<conf:(0.63)> lift:(2.53) lev:(0.01) conv:(1.84)
10. [packaged fruit/vegetables=t, frozen vegetables=t]: 24 ==>
[yogurt=t]: 15   <conf:(0.63)> lift:(2.51) lev:(0.01) conv:(1.8)
11. [rubbing alcohol=t, whisky=t]: 23 ==> [yogurt=t]: 14
<conf:(0.61)> lift:(2.44) lev:(0.01) conv:(1.73)
12. [root vegetables=t, mayonnaise=t]: 22 ==> [yogurt=t]: 13
<conf:(0.59)> lift:(2.37) lev:(0.01) conv:(1.65)
13. [zwieback=t, pet care=t]: 21 ==> [yogurt=t]: 12   <conf:(0.57)>
lift:(2.29) lev:(0.01) conv:(1.58)
14. [zwieback=t, curd=t]: 30 ==> [yogurt=t]: 17   <conf:(0.57)>
lift:(2.27) lev:(0.01) conv:(1.61)
15. [yogurt=t, newspapers=t]: 23 ==> [zwieback=t]: 13   <conf:(0.57)>
lift:(4.27) lev:(0.01) conv:(1.81)
16. [meat spreads=t]: 25 ==> [packaged fruit/vegetables=t]: 14
<conf:(0.56)> lift:(2.93) lev:(0.01) conv:(1.69)
```

```
17. [zwieback=t, fruit/vegetable juice=t]: 27 ==> [packaged
fruit/vegetables=t]: 15   <conf:(0.56)> lift:(2.91) lev:(0.01)
conv:(1.68)
18. [zwieback=t, turkey=t]: 29 ==> [yogurt=t]: 16   <conf:(0.55)>
lift:(2.21) lev:(0.01) conv:(1.56)
19. [packaged fruit/vegetables=t, fruit/vegetable juice=t]: 31 ==>
[yogurt=t]: 17   <conf:(0.55)> lift:(2.2) lev:(0.01) conv:(1.55)
20. [packaged fruit/vegetables=t, popcorn=t]: 22 ==> [yogurt=t]: 12
<conf:(0.55)> lift:(2.19) lev:(0.01) conv:(1.5)
21. [yogurt=t, sweet spreads=t]: 22 ==> [packaged fruit/vegetables=t]:
12   <conf:(0.55)> lift:(2.86) lev:(0.01) conv:(1.62)
22. [ham=t]: 28 ==> [yogurt=t]: 15   <conf:(0.54)> lift:(2.15)
lev:(0.01) conv:(1.5)
23. [yogurt=t, frozen vegetables=t]: 28 ==> [packaged
fruit/vegetables=t]: 15   <conf:(0.54)> lift:(2.81) lev:(0.01)
conv:(1.62)
24. [packaged fruit/vegetables=t, whisky=t]: 40 ==> [yogurt=t]: 21
<conf:(0.53)> lift:(2.11) lev:(0.01) conv:(1.5)
25. [yogurt=t, popcorn=t]: 23 ==> [packaged fruit/vegetables=t]: 12
<conf:(0.52)> lift:(2.73) lev:(0.01) conv:(1.55)
26. [rubbing alcohol=t, whisky=t]: 23 ==> [packaged
fruit/vegetables=t]: 12   <conf:(0.52)> lift:(2.73) lev:(0.01)
conv:(1.55)
27. [frozen vegetables=t]: 54 ==> [yogurt=t]: 28   <conf:(0.52)>
lift:(2.08) lev:(0.01) conv:(1.5)
28. [zwieback=t, rubbing alcohol=t]: 33 ==> [yogurt=t]: 17
<conf:(0.52)> lift:(2.07) lev:(0.01) conv:(1.46)
29. [packaged fruit/vegetables=t, zwieback=t]: 49 ==> [yogurt=t]: 25
<conf:(0.51)> lift:(2.05) lev:(0.01) conv:(1.47)
```

Παρατηρούμε ότι ο Fp-Growth εξάγει 29 κανόνες, δηλαδή έναν λιγότερο από τον Apriori. Να σημειώσουμε εδώ ότι ακόμη και με μεγαλύτερο αριθμό κανόνων numRules ο Apriori εξάγει συνολικά 30 κανόνες με τις δοθείσες παραμέτρους. Οι 29 αυτοί κανόνες που εξήγαγε ο Fp-Growth είναι κοινοί με τους 29 πρώτους του Apriori. Μπορούμε να εξάγουμε κάποια επιπλέον συμπεράσματα βασιζόμενοι στις τιμές των lift και leverage.

Για τον πρώτο κανόνα `[turkey=t, whisky=t]: 15 ==> [yogurt=t]: 12 <conf:(0.8)> lift:(3.21) lev:(0.01) conv:(2.82)` ισχύει ότι όποιος αγοράζει γαλοπούλα και ουίσκι αγοράζει και γιαούρτι με πιθανότητα 80%. Η τιμή του lift είναι 3.21 και είναι μεγαλύτερη του 1 δηλαδή ο κανόνας έχει θετική συσχέτιση μεταξύ του LHS και του RHS. Το leverage του κανόνα έχει τιμή 0.01 πολύ κοντά στο 0 που φανερώνει ισχυρή ανεξαρτησία μεταξύ του LHS και του RHS. Κοιτώντας προσεκτικά το σύνολο των κανόνων παρατηρούμε ότι όλοι έχουν τιμή στο leverage ίση με 0.01. Άρα δεν υπάρχει κάποιος κανόνας που να ικανοποιεί πλήρως όλα τα μέτρα ανάλυσης.

Δοκιμάζουμε να ξανα τρέξουμε τους αλγορίθμους Apriori και Fp-Growth αυτή την φορά με υποστήριξη 1% (**lowerBoundMinSupport=0.01**) και εμπιστοσύνη 60% (**minMetric=0.6**). Και οι δύο αλγόριθμοι εξάγουν 11 κανόνες, τους 11 πρώτους που

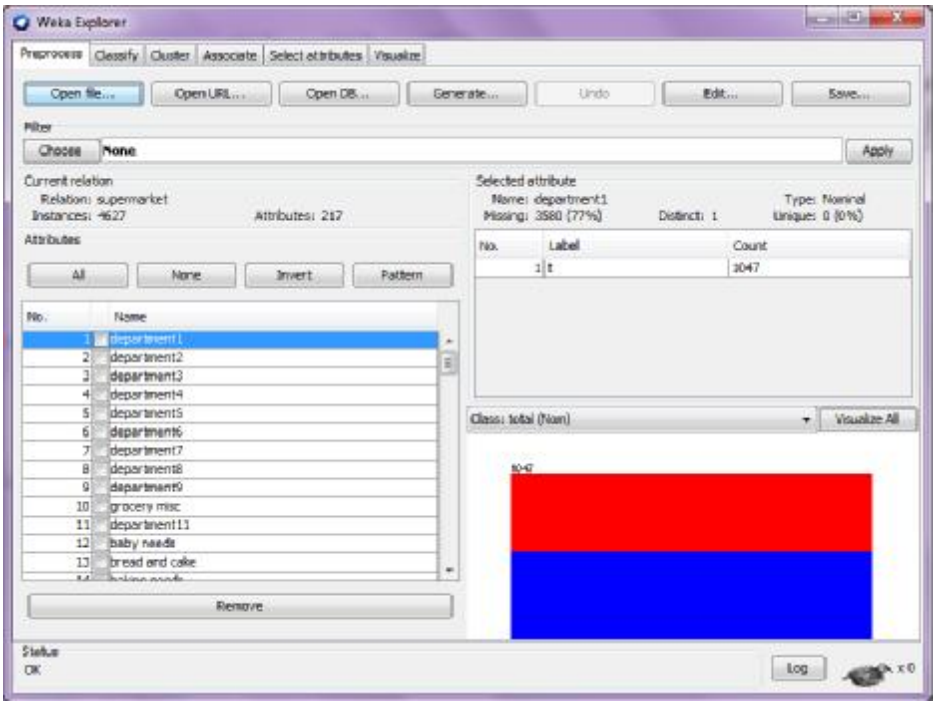
ήδη αναλύσαμε. Διατηρώντας την υποστήριξη στο 1%, αλλάζουμε την εμπιστοσύνη στο 70%. Αυτή την φορά έχουμε 5 κανόνες και για τους δύο αλγόριθμου και τέλος, ορίζοντας την εμπιστοσύνη στο 80% ο Apriori εξάγει έναν κανόνα ενώ ο Fp-Growth δεν εξάγει κανέναν κανόνα από τη Βάση.

Επόμενο βήμα είναι να αλλάξουμε την τιμή της υποστήριξης. Ορίζουμε λοιπόν την υποστήριξη στο 2% και την εμπιστοσύνη 50%. Ο Apriori εξάγει συνολικά 3 κανόνες και 2 κανόνες εξάγει ο Fp-Growth. Οι έξοδοι των αλγορίθμων δίνονται παρακάτω.

```
Apriori
=======

Minimum support: 0.02 (24 instances)
Minimum metric <confidence>: 0.5
Number of cycles performed: 20

Generated sets of large itemsets:

Size of set of large itemsets L(1): 62

Size of set of large itemsets L(2): 63

Size of set of large itemsets L(3): 1

Best rules found:

 1. frozen vegetables=t 54 ==> yogurt=t 28    conf:(0.52)
 2. packaged fruit/vegetables=t zwieback=t 49 ==> yogurt=t 25
conf:(0.51)
 3. butter=t 60 ==> yogurt=t 30    conf:(0.5)


FPGrowth found 2 rules (displaying top 2)

1. [frozen vegetables=t]: 54 ==> [yogurt=t]: 28   <conf:(0.52)>
lift:(2.08) lev:(0.01) conv:(1.5)
2. [packaged fruit/vegetables=t, zwieback=t]: 49 ==> [yogurt=t]: 25
<conf:(0.51)> lift:(2.05) lev:(0.01) conv:(1.47)
```

Σε αυτό το σημείο οποιαδήποτε αύξηση στην τιμή της εμπιστοσύνης έχει ως αποτέλεσμα να μην εξάγονται κανόνες από κανέναν από τους δύο αλγορίθμους.

Η δεύτερη Βάση την οποία θα αναλύσουμε είναι η “**supermarket.arff**” και είναι διαθέσιμη στην ιστοσελίδα του WEKA. Σε αυτό το σύνολο υπάρχουν **4627 συναλλαγές** (instances) πελατών για **217 προϊόντα** (attributes) σε **216** διαφορετικά **καταστήματα**. Εάν το συνολικό ποσό που δαπανήθηκε είναι μικρότερο των **100$** το attribute “**total**” παίρνει την τιμή **low** διαφορετικά την τιμή **high**.



Εικόνα 25: training set “supermarket.arff”

Ακολουθώντας την ίδια διαδικασία που αναφέραμε προηγούμενος, ορίζουμε τη υποστήριξη στο 30%, την εμπιστοσύνη στο 70% και επιλέγουμε να εξάγουμε τους 10 πρώτους κανόνες συσχέτισης.

Οι έξοδοι των αλγορίθμων Apriori και Fp-Growth φαίνονται παρακάτω.

```
Apriori
=======
Minimum support: 0.3 (1388 instances)
Minimum metric <confidence>: 0.7
Number of cycles performed: 14

Generated sets of large itemsets:

Size of set of large itemsets L(1): 25

Size of set of large itemsets L(2): 69

Size of set of large itemsets L(3): 20

Best rules found:
1. biscuits=t vegetables=t 1764 ==> bread and cake=t 1487
conf:(0.84)
2. total=high 1679 ==> bread and cake=t 1413    conf:(0.84)
3. biscuits=t milk-cream=t 1767 ==> bread and cake=t 1485
conf:(0.84)
4. biscuits=t fruit=t 1837 ==> bread and cake=t 1541    conf:(0.84)
5. biscuits=t frozen foods=t 1810 ==> bread and cake=t 1510
conf:(0.83)
6. frozen foods=t fruit=t 1861 ==> bread and cake=t 1548
conf:(0.83)
```

```
7. frozen foods=t milk-cream=t 1826 ==> bread and cake=t 1516
conf:(0.83)
8. baking needs=t milk-cream=t 1907 ==> bread and cake=t 1580
conf:(0.83)
9. milk-cream=t fruit=t 2038 ==> bread and cake=t 1684
conf:(0.83)
10. baking needs=t biscuits=t 1764 ==> bread and cake=t 1456
conf:(0.83)


FPGrowth found 67 rules (displaying top 10)

 1. [vegetables=t, biscuits=t]: 1764 ==> [bread and cake=t]: 1487
<conf:(0.84)> lift:(1.17) lev:(0.05) conv:(1.78)
 2. [total=high]: 1679 ==> [bread and cake=t]: 1413   <conf:(0.84)>
lift:(1.17) lev:(0.04) conv:(1.76)
 3. [milk-cream=t, biscuits=t]: 1767 ==> [bread and cake=t]: 1485
<conf:(0.84)> lift:(1.17) lev:(0.05) conv:(1.75)
 4. [fruit=t, biscuits=t]: 1837 ==> [bread and cake=t]: 1541
<conf:(0.84)> lift:(1.17) lev:(0.05) conv:(1.73)
 5. [frozen foods=t, biscuits=t]: 1810 ==> [bread and cake=t]: 1510
<conf:(0.83)> lift:(1.16) lev:(0.04) conv:(1.69)
 6. [fruit=t, frozen foods=t]: 1861 ==> [bread and cake=t]: 1548
<conf:(0.83)> lift:(1.16) lev:(0.05) conv:(1.66)
 7. [milk-cream=t, frozen foods=t]: 1826 ==> [bread and cake=t]:
1516   <conf:(0.83)> lift:(1.15) lev:(0.04) conv:(1.65)
 8. [milk-cream=t, baking needs=t]: 1907 ==> [bread and cake=t]:
1580   <conf:(0.83)> lift:(1.15) lev:(0.04) conv:(1.63)
 9. [fruit=t, milk-cream=t]: 2038 ==> [bread and cake=t]: 1684
<conf:(0.83)> lift:(1.15) lev:(0.05) conv:(1.61)
10. [baking needs=t, biscuits=t]: 1764 ==> [bread and cake=t]: 1456
<conf:(0.83)> lift:(1.15) lev:(0.04) conv:(1.6)
```

Οι συνολικοί κανόνες που μπορούν να εξάγουν οι αλγόριθμοι είναι 67. Μερικοί από τους κανόνες που εξάγονται είναι οι εξής:

- **Κανόνας 1**: Όποιος αγοράζει μπισκότα (biscuits) και λαχανικά (vegetables) αγοράζει και ψωμί και κέικ (bread and cake) με πιθανότητα 84%. Η τιμή του lift είναι 1,17 που φανερώνει ισχυρή συσχέτιση των LHS και RHS του κανόνα. Η τιμή του leverage είναι 0.05 δηλαδή τα LHS και RHS παρουσιάζουν ισχυρή ανεξαρτησία.
- **Κανόνας 2**: Αυτός ο κανόνας μεταφράζεται ως εξής, οι πελάτες των οποίων το συνολικό κόστος των αγορών τους στο καλάθι ξεπερνά τα 100$ συμπεριλαμβάνουν στις αγορές τους και ψωμί και κέικ με πιθανότητα 84%.

Επανεκτελώντας τους αλγορίθμους αυτή την φορά με υποστήριξη 40% και εμπιστοσύνη 70% οι διαθέσιμοι κανόνες μειώνονται στους 10. Μια αύξηση της υποστήριξης στο 50% μειώνει τον αριθμό των εξαγόμενων κανόνων σε 3. Από αυτό το σημείο και έπειτα οποιαδήποτε αύξηση της υποστήριξης δεν εξάγει κανόνες.

Οι κανόνες με εμπιστοσύνη 70% και υποστήριξη 50% είναι οι εξής:

```
Apriori
=======

Minimum support: 0.5 (2314 instances)
Minimum metric <confidence>: 0.7
Number of cycles performed: 10

Generated sets of large itemsets:

Size of set of large itemsets L(1): 10

Size of set of large itemsets L(2): 2

Best rules found:

 1. milk-cream=t 2939 ==> bread and cake=t 2337    conf:(0.8)
 2. fruit=t 2962 ==> bread and cake=t 2325    conf:(0.78)
 3. bread and cake=t 3330 ==> milk-cream=t 2337    conf:(0.7)

FPGrowth found 3 rules (displaying top 3)

1. [milk-cream=t]: 2939 ==> [bread and cake=t]: 2337   <conf:(0.8)>
lift:(1.1) lev:(0.05) conv:(1.37)
2. [fruit=t]: 2962 ==> [bread and cake=t]: 2325   <conf:(0.78)>
lift:(1.09) lev:(0.04) conv:(1.3)
3. [bread and cake=t]: 3330 ==> [milk-cream=t]: 2337   <conf:(0.7)>
lift:(1.1) lev:(0.05) conv:(1.22)
```

Ο **κανόνας 1** δείχνει ότι όταν η κρέμα γάλακτος (milk – cream) βρίσκεται στο LHS του κανόνα και το ψωμί και κέικ (bread and cake) στο RHS του κανόνα η εμπιστοσύνη είναι στο 80%. Ενώ, βάσει του **κανόνα 3**, όταν το ψωμί και κέικ βρίσκονται στο LHS και η κρέμα γάλακτος στο RHS του κανόνα, η εμπιστοσύνη μειώνεται σε 70%. Στον **κανόνα 2** όταν τα φρούτα (fruits) είναι στο LHS το ψωμί και κέικ βρίσκεται στο RHS με πιθανότητα 78%.

3.6 Πειραματική Αξιολόγηση και Αποδοτικότητα

Τα πειράματα αξιολόγησης έγιναν σε ένα μηχάνημα AMD 1.40GHz, 3.60GB μνήμη και οι χρόνοι που μετρήθηκαν αναφέρονται σε run time και όχι CPU time.

Η διαδικασία εκτέλεσης και υπολογισμού του χρόνου εκτέλεσης των αλγορίθμων έγινε μέσω του λογισμικού Eclipse σε γλώσσα Java και με χρήση του πακέτου Weka.api (Παραρτήματα). Ο κώδικας έχει συνταχθεί εξ' ολοκλήρου με χρήση των μεθόδων που παρέχει το συγκεκριμένο πακέτο [11][13].

Για το σύνολο δεδομένων "supermarket.arff" επιλέξαμε να διατηρήσουμε την εμπιστοσύνη σταθερή στο 70% και η τιμή της υποστήριξης να παίρνει τιμές μεταξύ 10% και 60% και η αύξηση να γίνεται με βήμα 10% (πίνακας 6).

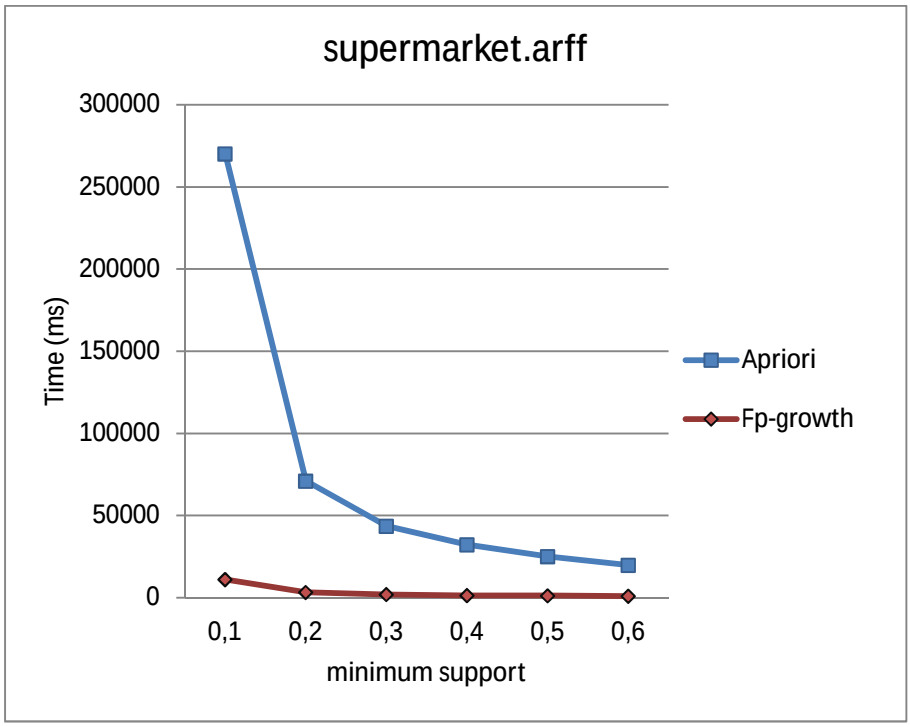
Algorithm	Support	Confidence	Number of rules	Run Time (ms)
Apriori	0.1	0.7	24570	270235
	0.2	0.7	769	71026
	0.3	0.7	67	43601
	0.4	0.7	10	32321
	0.5	0.7	3	25156
	0.6	0.7	No rules	19873
Fp-growth	0.1	0.7	24467	11177
	0.2	0.7	746	3381
	0.3	0.7	67	2040
	0.4	0.7	10	1448
	0.5	0.7	3	1289
	0.6	0.7	No rules	1049

Πίνακας 6: Χρονική ανάλυση "supermarket.arff"

Προκειμένου τα αποτελέσματα να είναι πιο ακριβή, χρησιμοποιήσαμε μια επαναληπτική μέθοδο στην οποία ο υπολογισμός του χρόνου, πχ. για υποστήριξη 10%, θα προκύπτει ως αποτέλεσμα του μέσου όρου του χρόνου που υπολογίστηκε για 10 επαναλήψεις. Όπως βλέπουμε και στον πίνακα 6, για μια υποστήριξη 10% ο Apriori χρειάζεται περίπου 4,5 λεπτά για την εξαγωγή 24.570 κανόνων. Ο FP-Growth, με την ίδια υποστήριξη, εξάγει 303 κανόνες λιγότερους από τον Apriori και η απαίτηση του σε χρόνο είναι περίπου 11 δευτερόλεπτα, δηλαδή περίπου 4,3 λεπτά λιγότερα από τον Apriori. Όσο η τιμή της υποστήριξης αυξάνει η χρονική διαφορά μεταξύ των αλγορίθμων μειώνει. Έτσι όταν η υποστήριξη φτάνει στο 50% η διαφορά στον χρόνο που απαιτείται για τον ίδιο αριθμό κανόνων είναι περίπου 24 δευτερόλεπτα.

Τα αποτελέσματα της χρονικής ανάλυσης του συνόλου supermarket.arff που απεικονίζονται στον πίνακα 6, δίνονται παρακάτω με την μορφή γραφήματος με σκοπό να γίνει πιο αισθητή η διαφορά που μόλις περιγράψαμε. Βλέπουμε λοιπόν, ότι ο Fp-Growth είναι γενικά ταχύτερος από τον Apriori. Το πόσο ταχύτερος είναι

εξαρτάται από το μέγεθος της υποστήριξης. Όταν η υποστήριξη παίρνει μεγάλες τιμές, δηλαδή ο αριθμός των συχνών προτύπων είναι μικρός, η διαφορά στην ταχύτητα είναι μικρή. Όσο η τιμή της υποστήριξης κατεβαίνει η διαφορά μεταξύ των δύο αλγορίθμων γίνεται πιο αισθητή και ειδικά στην τιμή 0,1 όπου η διαδικασία εξαγωγή κανόνων από τον Apriori είναι σημαντικά κοστοβόρα έναντι του FP-Growth.



Γράφημα 1: **Χρονική ανάλυση** "supermarket.arff"

Για το σύνολο δεδομένων "groceries.arff" επιλέξαμε να διατηρήσουμε την υποστήριξη σταθερή στο 1% και η εμπιστοσύνη να παίρνει τιμές μεταξύ 5% και 8% με βήμα αύξησης 1%. Οι χρόνοι που απαιτούνται για την εξαγωγή κανόνων μεταξύ των διαφόρων τιμών της εμπιστοσύνης δεν παρουσιάζουν σημαντική διαφορά για τον κάθε αλγόριθμο (πίνακας 7).

Algorithm	Support	Confidence	Number of rules	Run Time (ms)
Apriori	0.01	0.5	30	13024
	0.01	0.6	11	12923
	0.01	0.7	5	12413
	0.01	0.8	1	12408
	0.02	0.5	3	9521
Fp-growth	0.01	0.5	29	998
	0.01	0.6	11	615
	0.01	0.7	5	610
	0.01	0.8	No rules	570
	0.02	0.5	2	600

Πίνακας 7: **Χρονική ανάλυση** "groceries.arff

4. Απόκρυψη Κανόνων Συσχέτισης

Μέχρι τώρα είδαμε τους τρόπους με τους οποίους μπορούμε να ανακαλύψουμε συσχετίσεις ανάμεσα στα αντικείμενα μιας Βάσεις Δεδομένων με την χρήση αλγορίθμων Εξόρυξης Κανόνων Συσχέτισης. Η ερμηνεία των κανόνων αυτών μπορεί να μας δώσει πληροφορίες για τις αγοραστικές συνήθειες των πελάτες της επιχείρησης και να μας παρέχει ένα πλεονέκτημα που μπορεί να χρησιμοποιηθεί για την βελτιστοποίηση των πωλήσεων των προϊόντων της. Με την ίδια λογική η πληροφορία αυτή μπορεί να είναι ορατή και στους ανταγωνιστές μας. Δημιουργείται λοιπόν η ανάγκη για **απόκρυψη των κανόνων** αυτών ώστε να μην είναι ορατοί σε τρίτους.

Το παράδειγμα διαπραγμάτευσης και συναλλαγών που ακολουθεί διενεργήθηκε ανάμεσα στην αλυσίδα σούπερ μάρκετ **BigMart** (http://www.bigmart.com.np) και στην εταιρία πώλησης χαρτιού **Dedtrees** [10]. Ας υποθέσουμε ότι εμείς είμαστε οι διευθυντές της εταιρίας BigMart με την αλυσίδα των σούπερ μάρκετ. Η εταιρία Dedtrees μας προτείνει το εξής deal: Να αγοράζουμε από αυτήν χαρτί σε **χαμηλές**, για τα δεδομένα της αγοράς, **τιμές**, με την προϋπόθεση ότι θα τους δώσουμε **πρόσβαση** σε εκείνη τη Βάση Δεδομένων της εταιρία μας, η οποία περιλαμβάνει τις πληροφορίες σχετικά με τα καλάθια αγοράς των πελατών μας.

Αποδεχτήκαμε τη συμφωνία και έτσι η Dedtress ξεκίνησε την εξόρυξη γνώσης στη Βάση Δεδομένων των πελατών μας προκειμένου να οδηγηθεί σε συμπεράσματα σχετικά με την αγοραστική τους συμπεριφορά. Στην πραγματικότητα εφήρμοσε, σε πολύ μεγαλύτερη κλίμακα βέβαια, το παράδειγμα εξόρυξη αυτού του τύπου γνώσης μέσω κανόνων συσχέτισης που παραθέσαμε στην προηγούμενη ενότητα. Με την χρήση, λοιπόν, αλγορίθμων εξόρυξης κανόνων συσχέτισης, η εν λόγω εταιρία διαπίστωσε ότι οι πελάτες μας που αγοράζουν **αποβουτυρωμένο γάλα**, αγοράζουν επίσης και πράσινο χαρτί (**GreenPaper**).

Με βάση αυτό το συμπέρασμα που προέκυψε από τους κανόνες συσχέτισης, η Dedtrees αποφάσισε να ενεργοποιήσει μία **προσφορά** κατά την οποία με κάθε αγορά GreenPaper της εταιρίας Dedtrees θα παρέχει στους πελάτες μας ένα **κουπόνι έκπτωσης** της τάξεως των 50 σεντς στην τιμή του αποβουτυρωμένου γάλακτος. Αυτή η εκστρατεία οδήγησε την εταιρία Dedtrees στην πώληση τόσο μεγάλης ποσότητας δικό της χαρτιού μέσα από τα καταστήματά μας, που οδήγησε τις υπόλοιπες εταιρίες που μας προμήθευαν αντίστοιχο προϊόν εκτός συναγωνισμού και έτσι **μονοπώλησε** σχεδόν την αγορά στο χαρτί.

Σε εκείνο το κομβικό σημείο, εμείς ζητήσαμε από την Dedtrees με δεδομένο ότι συμβάλλαμε σημαντικά σε αυτήν την επιτυχία της, αφού εμείς ήμασταν οι διαμεσολαβητές πώλησης του χαρτιού τους στους πελάτες, να μας κάνει **καλύτερη χονδρική τιμή** στα προϊόντα που εμείς αγοράζαμε από αυτούς για να τα

μεταπουλήσουμε. Κάτι τέτοιο, όμως, η Dedtrees, εκμεταλλευόμενη το γεγονός που είχε υπερισχύσει συντριπτικά των υπολοίπων ανταγωνιστών της, μας το **αρνήθηκε**.

Το παραπάνω παράδειγμα, οδηγεί στο συμπέρασμα πως αν η BigMart με κατάλληλες **τεχνικές απόκρυψης της ευαίσθητης πληροφορίας** είχε παραδώσει μία φιλτραρισμένη Βάση Δεδομένων στην Dedtrees, τότε η δεύτερη **δεν θα μπορούσε να εξορύξει** αυτήν την τόσο σημαντική πληροφορία για το συνδυασμό του αποβουτυρωμένου γάλακτος και του χαρτιού και έτσι δεν θα μονοπωλούσε την αγορά επωφελούμενη μονομερώς, την ίδια ώρα που η BigMart έχανε οικονομικά γιατί για ένα διάστημα συνέχιζε να παραγγέλνει χαρτί και από άλλους προμηθευτές, χαρτί το οποίο ποτέ δεν κατάφερε να πουλήσει στην ποσότητα την οποία το αγόρασε.

Οι στρατηγικές απόκρυψης που προτείνονται [10] βασίζονται στην μείωση της υποστήριξης και τις εμπιστοσύνης, δηλαδή των μέτρων που καθορίζουν την σπουδαιότητα ενός κανόνα. Για να επιτευχθεί αυτό τροποποιούμε τις συναλλαγές αφαιρώντας ορισμένα στοιχεία ή εισάγοντας νέα στοιχεία ανάλογα με την στρατηγική απόκρυψης που ακολουθείται. Να σημειώσουμε εδώ ότι οι αλλαγές στην Βάση Δεδομένων που πραγματοποιούνται από αυτήν την διαδικασία θα πρέπει να περιοριστούν κατά τέτοιο τρόπο ώστε η απώλεια των πληροφοριών που θα προκύψει να είναι ελάχιστη. Κρίσιμος παράγοντας για την επίτευξη της ελάχιστης απώλειας πληροφοριών είναι η σωστή επιλογή αντικειμένων και συναλλαγών που θα τροποποιηθούν σε κάθε κανόνα συσχέτισης προς απόκρυψη.

4.1 Παραδοχές και συμβολισμοί

Γνωρίζουμε από τα προηγούμενα ότι ένας κανόνας συσχέτισης εντοπίζει εκείνα τα στοιχειοσύνολα που παρουσιάζονται αρκετά συχνά ώστε να θεωρηθούν *σχετικά* μεταξύ τους. Στην συνέχεια αντλεί εκείνους τους κανόνες των οποίων η συσχέτιση είναι αρκετά ισχυρή ώστε να θεωρηθεί πως παρουσιάζουν κάποιο ενδιαφέρον. Οι κανόνες αυτοί θεωρούνται «**ευαίσθητοι**» και στόχος μας είναι η απομόνωση τους από την Βάση Δεδομένων. Το πρόβλημα αυτό μπορεί να διατυπωθεί ως εξής:

«Δεδομένης μιας Βάσης Δεδομένων D, ενός συνόλου R των κανόνων συσχέτισης που έχουν αποκαλυφθεί από την D και ενός υποσυνόλου του R που θα καλείται R_H, πως μπορούμε να μετατρέψουμε την D σε μια Βάση Δεδομένων D' με τέτοιο τρόπο ώστε οι κανόνες του R να μπορούν να εξορυχτούν χωρίς να εξορυχτούν και οι κανόνες της R_H;»

Με άλλα λόγια, επιδιώκουμε μία μετατροπή της *D* (της αρχικής Βάσης Δεδομένων) σε μία *D'* (της Βάσης Δεδομένων που θα εξορυχτεί) η οποία θα **μεγιστοποιεί** τον αριθμό των κανόνων που θα μπορούν να εξορυχτούν από την $R - R_H$. Υπάρχουν δύο βασικές προσεγγίσεις που μπορούν να υιοθετηθούν: α) μπορούμε είτε να αποτρέψουμε τη δημιουργία των κανόνων στο R_H, αποκρύπτοντας τα συχνά σύνολα

από τα οποία προέρχονται και β) μειώνοντας την εμπιστοσύνη των ευαίσθητων κανόνων κάτω από το προκαθορισμένο όριο (min_conf).

Υποθέτουμε ότι οι συναλλαγές στην Βάση έχουν την εξής μορφή:

$$t = <TID, value\_of\_items, size>$$

όπου TID είναι ένα μοναδικό αναγνωριστικό τις συναλλαγής t και value_of_items είναι μια λίστα με τιμές όπου κάθε τιμή αντιπροσωπεύει ένα στοιχείο στο I.

TID	Items
T1	ABC
T2	ABC
T3	ABC
T4	AB
T5	A
T6	AC

TID	Items	Size
T1	111	3
T2	111	3
T3	111	3
T4	110	2
T5	100	1
T6	101	2

Εικόνα 26: α) Βάση Δεδομένων D, β) Β.Δ. D με προτεινόμενη σημειογραφία

Όπως φαίνετε και στον πίνακα 26.α, κάθε στοιχείο αποτελείται από κεφαλαία γράμματα του αγγλικού αλφαβήτου ταξινομημένα με αλφαβητική σειρά. Η ύπαρξη ή μη του στοιχείου στην συναλλαγή (πίνακας 26.β) αντιπροσωπεύετε από τις τιμές 1 και 0 αντίστοιχα. Για παράδειγμα, η βάση δεδομένων του πίνακα αποτελείται από τα στοιχεία I = {A, B, C} και η συναλλαγή που εξετάζουμε είναι η T6 = {A, C}. Βάσει της σημειογραφίας που αναφέραμε η συναλλαγή αυτή θα γραφεί ως εξής: t=<T6, [101], 2>.

Με βάση αυτά θα λέμε ότι το *t* **υποστηρίζει πλήρως** το *S* (στοιχειοσύνολο), εάν όλες οι τιμές του *t.value_of_items* είναι 1, διαφορετικά θα λέμε ότι το *S* **υποστηρίζεται μερικώς** από το *t*. Για παράδειγμα, εάν S = {A, B, C} = [1110], p = <T1, [1010], 2> και q = <T2, [1110], 3> τότε θα λέμε ότι το q υποστηρίζει πλήρως το S ενώ το p υποστηρίζει μερικώς το S.

Ένας κανόνας r αντιστοιχεί σε ένα στοιχεισύνολο. Το στοιχειοσύνολο αυτό αποτελείται από την ένωση του αριστερού μέλους του κανόνα με το δεξιό μέλος του ίδιου κανόνα και θα το συμβολίζουμε με I_r. Για να δηλώσουμε ότι ένα σύνολο συναλλαγών υποστηρίζει πλήρως το παραγόμενο στοιχειοσύνολο ενός κανόνα r (I_r) θα χρησιμοποιούμε τον συμβολισμό T_r. Με T_{lr} θα συμβολίζουμε ένα σύνολο συναλλαγών που *υποστηρίζει πλήρως* το LHS του κανόνα r και με T_{rr} το σύνολο συναλλαγών που *υποστηρίζει πλήρως* το RHS του κανόνα r. Τα σύνολα συναλλαγών που *υποστηρίζουν μερικώς* το LHS και RHS του κανόνα r θα συμβολίζονται με T'_{lr} και T'_{rr} αντίστοιχα.

Υποθέτουμε ότι ο κάθε κανόνας χαρακτηρίζεται από ένα επίπεδο ευαισθησίας. Το επίπεδο της ευαισθησίας καθορίζεται από την επιρροή που ασκεί ο κανόνας στο περιβάλλον στο οποίο ανήκει. Για παράδειγμα, σε ένα περιβάλλον λιανικής πώλησης, ένας κανόνας που μπορεί να χρησιμοποιηθεί για να ενισχύσει την πώληση

ενός συνόλου αντικειμένων θα μπορούσε να είναι ένας ευαίσθητος κανόνας. Ο αντίκτυπος του κανόνα στο λιανικό περιβάλλον είναι ο βαθμός που ο κανόνας αυξάνει τις πωλήσεις και κατά συνέπεια το κέρδος. Δεδομένου ότι μόνο οι συχνοί και ισχυροί κανόνες μπορούν να εξαχθούν από τους αλγορίθμους εξόρυξης δεδομένων, λέμε ότι μας ενδιαφέρει το επίπεδο ευαισθησίας μόνο των συχνών και ισχυρών κανόνων. Εάν ένας συχνός και ισχυρός κανόνας είναι πάνω από ένα ορισμένο επίπεδο ευαισθησίας η διαδικασία απόκρυψης θα πρέπει να εφαρμόζεται με τέτοιο τρόπο ώστε είτε η συχνότητα είτε η ισχύς του κανόνα να μειωθεί ώστε να φέρει την υποστήριξη και την εμπιστοσύνη του κανόνα κάτω από το *min_supp* και το *min_conf* αντίστοιχα. Ας υποθέσουμε ότι L είναι ένα συχνό στοιχεισύνολο με υποστήριξη μεγαλύτερη από το min_supp και $L_H \subseteq L$ είναι το σύνολο των συχνών στοιχειοσυνόλων που θέλουμε να αποκρύψουμε από την Βάση Δεδομένων D. Να σημειώσουμε εδώ ότι το L_H είναι το σύνολο των παραγόμενων στοιχειοσυνόλων του κανόνα στο R_H. Τέλος, συμβολίζουμε με T_Z το σύνολο των συναλλαγών που υποστηρίζουν ένα στοιχειοσύνολο Z και με την έκφραση A_D = |D|*AT L συμβολίζουμε τον μέσο αριθμό των στοιχείων στη βάση δεδομένων D όπου AT L είναι το μέσο μήκος της συναλλαγής.

Βασιζόμενοι στις παραπάνω παρατηρήσεις, θα επιχειρήσουμε σε αυτό το σημείο να κάνουμε ορισμένες παραδοχές, αναφορικά με την **ευελιξία** που υπάρχει στην Απόκρυψη Κανόνων Συσχέτισης. Οι Παραδοχές αυτές είναι οι παρακάτω:

1. Μπαίνουμε στην διαδικασία Απόκρυψης μόνο των κανόνων που υποστηρίζονται από ασυνεχή (disjoint) **μεγάλα στοιχειοσύνολα**
2. Αποκρύπτουμε Κανόνες Συσχέτισης τροποποιώντας τα ποσοστά των Δεικτών, είτε της **Εμπιστοσύνης**, είτε της **Υποστήριξης**
3. Επιλέγουμε την τροποποίηση των ποσοστών των Δεικτών (π.χ. τη μείωση) με γνώμονα **να μην προκληθούν ανεπιθύμητες ενέργειες** σε πληροφορίες που ΔΕΝ είναι ευαίσθητες
4. Αποκρύπτουμε **έναν κανόνα κάθε φορά** (όχι πολλούς ταυτόχρονα)
5. Μειώνουμε τους Δείκτες της Υποστήριξης ή της Εμπιστοσύνης **κατά μία μόνο μονάδα κάθε φορά**

Στην ενότητα που ακολουθεί, θα παραθέσουμε **ψευδοκώδικες** Αλγόριθμων Απόκρυψης Κανόνων Συσχέτισης με βάση τις πέντε παραπάνω παραδοχές. Σε κάθε περίπτωση, κοινός παρονομαστής και των πέντε παραδοχών που κάνουμε κατά τη διαδικασία απόκρυψης κανόνων, είναι η "Προσεκτική Απόκρυψη", δηλαδή οι κανόνες να αποκρύπτονται αφενός πολύ σταδιακά (ένας - ένας) και αφετέρου η μέθοδος που ακολουθείται, να εφαρμόζεται κατά τέτοιον τρόπο ώστε να μην επηρεαστούν ΜΗ ευαίσθητοι κανόνες που έχουν εξαχθεί και οι οποίοι μπορεί να μας οδηγήσουν σε ιδιαιτέρως επωφελή συμπεράσματα σχετικά με αποφάσεις που μπορεί να κληθεί να λάβει μια επιχείρηση.

4.2 Αλγόριθμοι Απόκρυψης

Συνεπώς, για την Απόκρυψη Κανόνων Συσχέτισης έχουμε τις εξής Στρατηγικές [10]:

1. Αν θέλουμε να κρύψουμε έναν κανόνα, πρέπει **να αλλάξουμε την υποστήριξη κάποιου μέρους της Βάσεις Δεδομένων** (δηλαδή, να μειώσουμε την υποστήριξη του στοιχειοσυνόλου παραγωγής). Βέβαια, από την άλλη πλευρά, οι αλλαγές στη βάση δεδομένων που εισήγαγε η διαδικασία απόκρυψης θα πρέπει να είναι περιορισμένες, κατά τέτοιο τρόπο ώστε η απώλεια των πληροφοριών που προκύπτουν από τη διεργασία είναι η ελάχιστη. Σύμφωνα με αυτό, προσπαθούμε να εφαρμόσουμε ελάχιστες αλλαγές στη βάση δεδομένων σε κάθε βήμα των αλγορίθμων απόκρυψης που προτείνουμε (apriori - δειγματοληψίας κτλ)
2. Η **μείωση στην υποστήριξη ενός στοιχειοσυνόλου S** μπορεί να γίνει με επιλογή μιας συναλλαγής t, που υποστηρίζει S και θέτοντας σε 0 τουλάχιστον μία από τις μη-μηδενικές τιμές της συγκεκριμένης t και των αντίστοιχων στοιχείων που αντιπροσωπεύουν αντικείμενα στο S
3. Η **αύξηση της υποστήριξης ενός στοιχειοσυνόλου S** μπορεί να επιτευχθεί με την επιλογή μιας συναλλαγής t που υποστηρίζει εν μέρει και ρυθμίζοντας σε 1 τις τιμές όλων των στοιχείων του S που αντιστοιχούσαν σε μηδενικές τιμές της t.
4. Για να είμαστε σε θέση να εντοπίσουμε κάποιους βιώσιμους τρόπους για τη μείωση είτε της υποστήριξης είτε της εμπιστοσύνης ενός κανόνα, πρέπει να αναλύσουμε τους τύπους που έχουμε ήδη παρουσιάσει για την εμπιστοσύνη και την υποστήριξη. Σε αυτό το σημείο θυμίζουμε ότι η Υποστήριξη (support): εκφράζει την πιθανότητα να βρεθεί το καλάθι $\{X_1...X_n, Y\}$ στη Βάση Δεδομένων και ισούται με τον λόγο των εγγραφών που περιλαμβάνουν το $\{X_1...X_n, Y\}$ προς το σύνολο των εγγραφών. Η Εμπιστοσύνη (confidence): εκφράζει την πιθανότητα να βρεθεί το Y σε ένα καλάθι που περιέχει τα $\{X_1,...X_n\}$ και ισούται με το λόγο των εγγραφών που περιλαμβάνουν το $\{X_1...X_n, Y\}$ προς το σύνολο των εγγραφών που περιλαμβάνουν τα X_i.
5. Με βάση τα παραπάνω, αν θέλουμε να μειωθεί η τιμή του εκάστοτε δείκτη, μπορούμε να υιοθετήσουμε οποιαδήποτε από τις ακόλουθες επιλογές: (α) **μπορούμε να μειώσετε τον αριθμητή**, διατηρώντας παράλληλα τον παρονομαστή σταθερό, ή (β) **θα αυξήσουμε τον παρονομαστή**, διατηρώντας παράλληλα τον αριθμητή σταθερό

Επομένως, παραθέτουμε παρακάτω πέντε αλγόριθμους, για κάθε μία από τις πέντε παραδοχές αντίστοιχα:

Αλγόριθμος 1

Ο αλγόριθμος που ακολουθεί, αυξάνει τον δείκτη της Υποστήριξης του κανόνα, μέχρι το σημείο εκείνο που η εμπιστοσύνη του θα μειωθεί τόσο όσο το ελάχιστο όριο που έχουμε θέσει για αυτήν (min_conf threshold):

INPUT: a set R_H of rules to hide, the source database D, the number $|D|$ of transactions in D, the *min_supp* threshold

OUTPUT: the database D transformed so that the rules in R_H cannot be mined

Begin
Foreach rule r in R_H **do** {
1. $T'l_r$ = {t in D / t partially supports l_r}
2. for each transaction of $T'l_r$ count the number of items of lr in it
3. sort the transaction in $T'l_r$ in descending order of the number of items of l_r supported
4. **repeat until** *Conf(r)<min_conf*
{
5. choose the transaction t ∈ $T'l_r$ with the highest number of items of l_r supported (t is the first transaction in $T'l_r$)
6. modify t to support l_r
7. increase the support of l_r by 1
8. recomputed the confidence of r
9. remove t from $T'l_r$
}
10. remove r from R_H
}
End

Αλγόριθμος 2

Ο αλγόριθμος που ακολουθεί μειώνει την υποστήριξη για κάθε επιλεγμένο κανόνα μειώνοντας τη συχνότητα της επακόλουθης μέσω των συναλλαγών που υποστηρίζουν τον κανόνα. Αυτή η διαδικασία συνεχίζεται έως ότου είτε η εμπιστοσύνη ή η υποστήριξη του κανόνα φτάσει να είναι κάτω από το ελάχιστο όριο (threshold):

INPUT: a set R_H of rules to hide, the source database D, the size of the database $|D|$, the *min_conf* threshold, the *min_supp* threshold

OUTPUT: the database D transformed so that the rules in R_H cannot be mined

Begin
Foreach rule r in R_H **do** {
1. T_r = {t in D / t fully supports r}

2. for each transaction of T_r count the number of items in it
3. sort the transaction in T_r in ascending order of the number of items supported
4. **repeat until (***conf(r)<min_conf* or*supp(r)<min_supp)*
   {
   5. choose the transaction t in T_r with the lowest number of items (the first transaction in T_r)
   6. choose the item *j* in r_r with the minimum impact on the ($|r_r|$-1)-itemsets
   7. delete *j* from *t*
   8. decrease the support of *r* by 1
   9. recompute the confidence of *r*
   10. remove *t* from T_r
   }
11. remove *r* from R_H
}

End

Αλγόριθμος 3

Ο ψευδοκώδικας του τρίτου αλγόριθμου αναπαριστά μία ακόμα στρατηγική, μέσω της οποίας ο αλγόριθμος μειώνει τον δείκτη υποστήριξης των ευαίσθητων κανόνων, μέχρι το σημείο εκείνο όπου ο δείκτης της εμπιστοσύνης βρεθεί κάτω από το εκάστοτε ελάχιστο όριο που έχουμε θέσει (min_conf_threshold), είτε μέχρι το σημείο εκείνο όπου ο ίδιος ο δείκτης της υποστήριξης βρεί κάτω από το εκάστοτε ελάχιστο όριο που έχει θέσει για την υποστήριξη (min_supp_threshold):

INPUT: a set R_H of rules to hide, the source database *D*, the size of the database |*D*|, the *min_conf* threshold, the *min_supp* threshold

OUTPUT: the database *D* transformed so that the rules in R_H cannot be mined

Begin

Foreach rule *r* in R_H **do** {

1. T_r = {*t* in *D* / *t* fully supports *r*}
2. for each transaction of T_r count the number of items in it
3. sort the transaction in T_r in ascending order of the number of items supported
4. **repeat until (***conf(r)<min_conf)*
   {

5. choose the transaction t in T_r with the lowest number of items (the first transaction in T_r)
6. choose the item j in r with the minimum impact on the (|r|-1)-itemsets
7. delete j from t
8. decrease the support of r by 1
9. recompute the confidence of r
10. remove t from T_r

}

11. remove r from R_H

}

End

Αλγόριθμος 4

Ο παρακάτω αλγόριθμος πραγματοποιεί απόκρυψη των ευαίσθητων κανόνων μειώνοντας τον δείκτη υποστήριξης των στοιχειοσυνόλων παραγωγής τους, μέχρι εκείνο το σημείο όπου η τιμή του δείκτη της υποστήριξης θα βρεθεί κάτω από το ελάχιστο όριο που έχουμε θέσει για τον εν λόγω δείκτη. Το στοιχείο που φέρει τη μέγιστη υποστήριξη, είναι κρυμμένο από την ελάχιστου εύρους συναλλαγή (minimum length transaction). Οι κανόνες R_H, όπως αντίστοιχα ισχύει το ίδιο και για τους τρεις προηγούμενους αλγόριθμους που παραθέσαμε, αφορούν κανόνες που δρομολογούνται για απόκρυψη. Τα στοιχειοσύνολα παραγωγής των κανόνων RH αποθηκεύονται στην μεταβλητή L_H και αποκρύπτονται ένας - ένας (βηματικά) οι κανόνες με μείωση του δείκτη της υποστήριξης:

INPUT: a set L of large itemsets, the set L_H of large itemsets to hide, the database D and the *min_supp* threshold

OUTPUT: the database D modified by the deletion of the large itemsets in L_H

Begin

1. Sort L_H in descending order of size and support of the large itemsets

foreach Z in L_H

{

2. Sort the transaction in T_Z in ascending order of transaction size
3. $N\_iterations = |T_Z| - min\_supp * |D|$

for k=1 **to** *N_iteration* **do**

{

4. Remove the maximal support item of Z from the next transaction in T_Z
5. Update the database, D

```
        }
      }
End
```

Αλγόριθμος 5

Ο τελευταίος αλγόριθμος που προτείνουμε, δρομολογεί την απόκρυψη των ευαίσθητων κανόνων, μειώνοντας την υποστήριξη των στοιχειοσυνόλων παραγωγής τους μέχρι εκείνο το σημείο όπου ο δείκτης της υποστήριξης βρεθεί κάτω από την τιμή που έχουμε θέσει ως ελάχιστο όριο. Αν υπάρχουν περισσότερα του ενός μεγάλα στοιχειοσύνολα προς απόκρυψη, ο αλγόριθμος τότε ταξινομεί τα μεγάλα αυτά στοιχειοσύνολα ως προς το μέγεθός τους. Με "Z" σε αυτόν τον αλγόριθμο αναπαριστούμε το "επόμενο" μεγάλο στοιχειοσύνολο που αποκρύπτεται:

INPUT: a set L of large itemsets, the set L_H of large itemsets to hide, the database D and the min_supp threshold

OUTPUT: the database D modified by the deletion of the large itemsets in L_H

Begin

1. Sort L_H in descending order of size and support of the large itemsets

foreach Z in L_H **do**

{

2. i=0
3. j=0
4. <T_Z>=sequence of transactions that support Z
5. <Z>=sequence of items in Z
6. $N\_iterations = |T_Z| - min\_supp * |D|$

for k=1 **to** $N\_iteration$ **do**

{

7. $\alpha = i^{th}$ item in <Z>
8. $t = j^{th}$ transaction in <T_Z>
9. Delete α from t
10. Update the database D
11. $i = (i+1)$ modulo size(Z)
12. $j = j+1$

}

}

End

Ποιο εκτενείς ανάλυση των αλγορίθμων σχετικά με την χρονική πολυπλοκότητα καθώς και πειραματική αξιολόγηση και σύγκρισή τους μπορεί να βρεθεί στην εργασία των V.S. Verykios, A.K. Elmagarmid et al.(2003).

5. Συμπεράσματα

Η εργασία αυτή αποτελεί ένα παράδειγμα αξιοποίησης των **αλγορίθμων εξόρυξης Κανόνων Συσχέτισης** από Βάσεις Δεδομένων που περιλαμβάνουν πληροφορίες δοσοληψιών που διαμορφώνουν τα "**καλάθια αγοράς**" των πελατών και που είναι πολύ πιθανόν να βοηθήσουν επιχειρήσεις και γενικότερα ανθρώπους που εμπλέκονται στη **Διαχείριση** μίας πελατειακής Βάσης Δεδομένων ή στη **Διοίκηση** μιας Επιχείρησης οπότε και καλούνται να πάρουν σημαντικές αποφάσεις σχετικά με τον τρόπο λειτουργίας της.

Μέσα από τη μελέτη μας διαπιστώσαμε πως η εξόρυξη αυτών των δεδομένων περιλαμβάνει αρκετούς **παράγοντες** που είναι σημαντικό να ληφθούν σοβαρά υπόψη. Αυτοί οι παράγοντες αφορούν αφενός την **αξιοπιστία** των Κανόνων Συσχέτισης οι οποίοι εξάγονται από τη Βάση και αφετέρου την **απόρριψη** εκείνων των κανόνων οι οποίοι δεν οδηγούν σε κάποιο **ουσιαστικό συμπέρασμα**.

Με γνώμονα τα παραπάνω, αποδείξαμε πως οι αλγόριθμοι **Apriori** και **FP-Growth** δύναται να αξιοποιηθούν κατάλληλα, δεχόμενοι τις κατάλληλες παραμετροποιήσεις, ώστε να μην περιορίζονται στην "ελεύθερη" παραγωγή κανόνων, αλλά με την παρέμβαση του χρήστη από την επιχείρηση να εξάγουν κανόνες που είναι αξιόπιστοι, θέτοντας κάθε φορά όρια για τα μεγέθη της *υποστήριξης* και της *εμπιστοσύνης*.

Ο Apriori και ο Fp-Growth εξάγουν σε μεγάλο βαθμό τον ίδιο αριθμό κανόνων. Ωστόσο, ο **χρόνος** που απαιτεί ο Apriori για την εξαγωγή κανόνων είναι αισθητά μεγαλύτερος του Fp-Growth για ορισμένες τιμές της υποστήριξης. Αυτό οφείλεται στην συμπαγή δομή του Fp-Growth όπου εξαλείφει την επαναλαμβανόμενη ανίχνευση των συναλλαγών. Ακόμη, με την κατάλληλη τροποποίηση ο FP-Growth μπορεί να λειτουργήσει συμπληρωματικά του Apriori, δηλαδή εκεί που ο Apriori εξάγει συσχετίσεις προϊόντων που **υπάρχουν** σε ένα καλάθι αγοράς, ο FP-Growth εξάγει συσχετίσεις προϊόντων που **απουσιάζουν** από ένα καλάθι αγοράς.

Μέσα από τους κανόνες συσχέτισης εξάγεται χρήσιμη γνώση που με κατάλληλη αξιοποίηση της μπορεί να βελτιώσει την ανταγωνιστική θέση της επιχείρησης. Πολλές φορές όμως η γνώση αυτή μπορεί να είναι προσβάσιμη σε τρίτους, ανταγωνιστές ίσος της επιχείρησης, αφαιρώντας της έτσι το όφελος που θα είχε σε διαφορετική περίπτωση. Για να αντιμετωπίσουμε το πρόβλημα αυτό χρησιμοποιούμε **τεχνικές απόκρυψης** των κανόνων εκείνων που παρουσιάζουν κάποια σπουδαιότητα.

Επιπρόσθετα, αξίζει να αναφέρουμε πως οι κανόνες συσχέτισης μπορούν να βρουν εφαρμογή **και σε άλλους τομείς**, πέραν αυτού της επιχειρηματικότητας, όπως στα **ιατρικά** δεδομένα και στην **εκπαίδευση**.

Επαγωγικά, λοιπόν, σκεπτόμενοι, καταλήγουμε στο συμπέρασμα πως κάθε μέθοδος εξόρυξης γνώσης από μία Βάση Δεδομένων μπορεί να βοηθήσει, αλλά σαν

διαδικασία επιβάλλεται να γίνεται με **στρατηγική** και προσεκτική εκ προοιμίου **μελέτη**. Αυτό διότι κάθε Βάση Δεδομένων περιέχει εξ' ορισμού μεγάλο όγκο πληροφορίας και, πολύ εύκολα, μπορεί να εξαχθεί γνώση που μπορεί να οδηγήσει σε **μη ασφαλή συμπεράσματα**.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Ι. Βλαχάβας, Π. Κεφαλάς, Ν. Βασιλειάδης, Φ. Κόκκορας, Η. Σακελλαρίου (2006) *"Τεχνητή νοημοσύνη" Γ' έκδοση*, Εκδόσεις Γκιούρδας

[2] Μ. Χαλκίδης, Μ. Βαζιργιάννης (2005) *"Εξόρυξη από Βάσεις Δεδομένων και τον Παγκόσμιο Ιστό" Β' έκδοση,* Εκδόσεις Τυπωθήτω

[3] Margaret H. Dunham *"Data mining / εισαγωγικά και προηγμένα θέματα εξόρυξης γνώσης από δεδομένα",* Επιμέλεια ελληνικής έκδοσης: Β. Βερύκιος, Γ. Θεοδωρίδης

[4] Ι. Π. Γουρδουλής (2008) *"Αλγόριθμοι Εξόρυξης δεδομένων για χειρισμό πολλαπλών υποστηρίξεων και αρνητικών συσχετίσεων"* Πανεπιστήμιο Πατρών

[5] Κ. Κίτρου, *"Αναβάθμιση δυναμικού ιστότοπου διαγνωστικού κέντρου με χρήση κατάλληλων εφαρμογών λογισμικού και κανόνων τεχνητής νοημοσύνης"*, Πανεπιστήμιο Πατρών

[6] Γ. Αντζουλάτος (2014) "Υπολογιστική Νοημοσύνη" - Τμήμα Διοίκησης Επιχειρήσεων - ΤΕΙ Δυτικής Ελλάδας

[7] Anany Levitin (2007) "*The design & analysis of algorithms"*, Επιμέλεια Ελληνικής Έκδοσης: Μ. Ρουμελιώτης", Εκδόσεις: Τζιόλα (2008)

[8] Jiawei Han, Jian Pei et. al (2004) *"Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach"*, Simon Fraser University

[9] P.N. Tan, M. Steinbach and V. Kumar (2006) *"Introduction to Data Mining, Addison Wesley"*, Επιμέλεια Ελληνικής Έκδοσης: Β. Βερύκιος και Σ. Σουραβλάς, Εκδόσεις: Τζιόλα (2010)

[10] V.S Verykios, A.K. Elmagarmid et. al (2003) "Association Rule Hiding"

[11] Noureddin Sadawi, https://github.com/nsadawi Πρόσβαση την 3η Απριλίου 2016

[12] Salem Marafi (2014) *Market Basket Analysis with R,* http://www.salemmarafi.com /code/market-basket-analysis-with-r/ Πρόσβαση την 21η Ιανουαρίου 2016

[13] http://weka.sourceforge.net/doc.stable/ Πρόσβαση την 3η Απριλίου 2016

ΠΑΡΑΡΤΗΜΑΤΑ

Αλγόριθμος Apriori (supermarket)

```
package weka.api;
import weka.associations.Apriori;
import weka.core.Instances;
import weka.core.converters.ConverterUtils.DataSource;

public class AssocRulesApriori{
      public static void main(String args[]) throws Exception{
            String dataset = "/users/user/Desktop/supermarket.arff";
            DataSource source = new DataSource(dataset);
            Instances data = source.getDataSet();

            for(double i=0.1d; i<=0.6d; i=i+0.1d){
                  long count = 0;
                  for(int j=1; j<=10; j++){
                        long Tstart=(int)System.currentTimeMillis();

                        Apriori model = new Apriori();
                        double Delta = 0.05;
                        double LMinSupport = i;
                        double MinMetric = 0.7;
                        int NumRules = 30000;
                        double UMinSupport = 1.0;

                        model.setDelta(Delta);
                        model.setLowerBoundMinSupport(LMinSupport);
                        model.setMinMetric(MinMetric);
                        model.setNumRules(NumRules);
                        model.setUpperBoundMinSupport(UMinSupport);
                        model.buildAssociations(data);

                        long Tfinish=(int)System.currentTimeMillis();
                        long Time=(Tfinish-Tstart);
                        count = count +Time;
                  }
                  long average = count/10;
                  System.out.println("Time: "+average);
            }     }     }
```

Αλγόριθμος Apriori (groceries)

```
package weka.api;
import weka.associations.Apriori;
import weka.core.Instances;
import weka.core.converters.ConverterUtils.DataSource;

public class AssocRulesApriori{
	public static void main(String args[]) throws Exception{
		String dataset = "/users/user/Desktop/groceries.arff";
		DataSource source = new DataSource(dataset);
		Instances data = source.getDataSet();

		for(double i=0.5d; i<=0.8d; i=i+0.1d){
			long count = 0;
			for(int j=1; j<=10; j++){
				long Tstart=(int)System.currentTimeMillis();

				Apriori model = new Apriori();
				double Delta = 0.05;
				double LMinSupport = 0.01;
				double MinMetric = i;
				int NumRules = 30000;
				double UMinSupport = 1.0;

				model.setDelta(Delta);
				model.setLowerBoundMinSupport(LMinSupport);
				model.setMinMetric(MinMetric);
				model.setNumRules(NumRules);
				model.setUpperBoundMinSupport(UMinSupport);
				model.buildAssociations(data);

				long Tfinish=(int)System.currentTimeMillis();
				long Time=(Tfinish-Tstart);
				count = count +Time;
			}
			long average = count/10;
			System.out.println("Time: "+average);
		}

	}

}
```

Αλγόριθμος FP-Growth (supermarket)

```
package weka.api;
import weka.associations.FPGrowth;
import weka.core.Instances;
import weka.core.converters.ConverterUtils.DataSource;

public class AssocRulesFpGrowth{
	public static void main(String args[]) throws Exception{
		String dataset = "/users/user/Desktop/supermarket.arff";
		DataSource source = new DataSource(dataset);
		Instances data = source.getDataSet();

		for(double i=0.1d; i<=0.6d; i=i+0.1d){
			long count = 0;
			for(int a=1; a<=10; a++){
				long Tstart=(int)System.currentTimeMillis();

				FPGrowth model = new FPGrowth();
				double Delta = 0.05;
				double LMinSupport = i;
				double MinMetric = 0.7;
				int NumRules = 30000;
				double UMinSupport = 1.0;

				model.setDelta(Delta);
				model.setLowerBoundMinSupport(LMinSupport);
				model.setNumRulesToFind(NumRules);
				model.setUpperBoundMinSupport(UMinSupport);
				model.setMinMetric(MinMetric);
				model.buildAssociations(data);

				long Tfinish=(int)System.currentTimeMillis();
				long Time=(Tfinish-Tstart);
				count = count +Time;
			}
			long average = count/10;
			System.out.println("Time: "+average);
		}
	}
}
```

Αλγόριθμος FP-Growth (groceries)

```
package weka.api;
import weka.associations.FPGrowth;
import weka.core.Instances;
import weka.core.converters.ConverterUtils.DataSource;

public class AssocRulesFpGrowth{
	public static void main(String args[]) throws Exception{
		String dataset = "/users/user/Desktop/groceries.arff";
		DataSource source = new DataSource(dataset);
		Instances data = source.getDataSet();

		for(double i=0.5d; i<=0.8d; i=i+0.1d){
			long count = 0;
			for(int a=1; a<=10; a++){
				long Tstart=(int)System.currentTimeMillis();

				FPGrowth model = new FPGrowth();
				double Delta = 0.05;
				double LMinSupport = 0.01;
				double MinMetric = i;
				int NumRules = 30000;
				double UMinSupport = 1.0;

				model.setDelta(Delta);
				model.setLowerBoundMinSupport(LMinSupport);
				model.setNumRulesToFind(NumRules);
				model.setUpperBoundMinSupport(UMinSupport);
				model.setMinMetric(MinMetric);
				model.buildAssociations(data);

				long Tfinish=(int)System.currentTimeMillis();
				long Time=(Tfinish-Tstart);
				count = count +Time;
			}
			long average = count/10;
			System.out.println("Time: "+average);
		}
	}
}
```