ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Εξερεύνηση Ομοιότητας Ερωτημάτων μέσω Τεχνικών Μηχανικής Μάθησης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΓΕΩΡΓΙΟΥ Ξ. ΣΙΑΧΑΜΗ

Επιβλέπουσα : Βηρένα Καντερέ
Επίκουρη Καθηγήτρια Ε.Μ.Π

Αθήνα, Οκτώβριος 2019

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Εξερεύνηση Ομοιότητας Ερωτημάτων μέσω Τεχνικών Μηχανικής Μάθησης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΓΕΩΡΓΙΟΥ Ξ. ΣΙΑΧΑΜΗ

Επιβλέπουσα : Βηρένα Καντερέ
Επίκουρη Καθηγήτρια Ε.Μ.Π

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 21η Οκτωβρίου 2019.

.....	.....	.....
Βηρένα Καντερέ	Γεώργιος Στάμου	Νικόλαος Παπασπύρου
Επικ. Καθηγήτρια Ε.Μ.Π	Αναπλ. Καθηγητής Ε.Μ.Π	Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2019

.....

ΓΕΩΡΓΙΟΣ Ξ. ΣΙΑΧΑΜΗΣ

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Ευχαριστίες

Πρώτα από όλα θα ήθελα να ευχαριστήσω την επιβλέπουσα Αναπληρώτρια Καθηγήτρια κα. Βηρένα Καντερέ για το ενδιαφέρον που μου έδειξε, τις γνώσεις που μου μετέδωσε και την άριστη καθοδήγηση κατά την διάρκεια της εκπόνησης της παρούσας διπλωματικής εργασίας, βοηθώντας με έτσι να αποκομίσω σημαντικά εφόδια για την επαγγελματική και ερευνητική μου πορεία. Επίσης ευχαριστώ πολύ την οικογένεια μου, τους συγγενείς και τους κοντινούς μου ανθρώπους που με στήριξαν σε όλη τη διάρκεια των σπουδών μου, έδειξαν αμέριστη υπομονή και κατανόηση και συντέλεσαν στο να φτάσω στην ευχάριστη αυτή στιγμή.

Περίληψη

Η παρούσα διπλωματική εργασία ανήκει στον τομέα της Ολοκλήρωσης Δεδομένων (Data Integration). Συγκεκριμένα μελετά την ομοιότητα ερωτημάτων για την προσεγγιστική μεταγραφή ερωτημάτων στο πλαίσιο μιας βάσης δεδομένων ομότιμων κόμβων. Σε αυτού του είδους τις βάσεις δεδομένων, κάθε κόμβος διαθέτει μια ιδιωτική βάση δεδομένων, η οποία έχει το δικό της ανεξάρτητο σχήμα. Όταν γίνεται ένα ερώτημα σε έναν κόμβο, το ερώτημα αυτό διαδίδεται και στους υπόλοιπους κόμβους για να απαντηθεί κι από τις δικές τους βάσεις δεδομένων και να επιστραφούν τα αποτελέσματα στον ερωτηθέντα. Λόγω του ανεξάρτητου σχήματος για την βάση κάθε κόμβου για να μπορεί ένα ερώτημα να απαντηθεί σε έναν άλλο κόμβο πρέπει να μεταγραφεί στο αντίστοιχο σχήμα. Η μεταγραφή αυτή γίνεται με την βοήθεια αντιστοιχίσεων μεταξύ των διαφόρων σχημάτων, οι οποίες όμως στη γενική περίπτωση δεν είναι πλήρεις και οι κλασσικοί αλγόριθμοι μεταγραφής δεν λειτουργούν. Έτσι απαιτείται η χρήση ενός προσεγγιστικού αλγορίθμου για την επίτευξη της μεταγραφής. Ο αλγόριθμος αυτός κάνει χρήση συναρτήσεων ομοιότητας ερωτημάτων.

Σκοπός της εργασίας είναι η μελέτη της απόδοσης του αλγορίθμου για τις διάφορες διαθέσιμες συναρτήσεις ομοιότητας και για διάφορα είδη ερωτημάτων, ώστε να επιτευχθεί η αυτοματοποίηση της επιλογής συνάρτησης ομοιότητας ανάλογα με το ερώτημα και να βελτιωθεί η απόδοση του αλγορίθμου. Η ερευνητική διαδικασία που ακολουθείται κάνει μια πρώτη προσπάθεια χρήσης τεχνικών μηχανικής μάθησης στο συγκεκριμένο πλαίσιο και μελετά την δομική ομοιότητα των ερωτημάτων που έως σήμερα δεν είχε ερευνηθεί. Για το σκοπό αυτό πραγματοποιείται σε τέσσερα στάδια.

Αρχικά δίνονται οι ιδέες και οι στόχοι και αποδίδονται οι απαραίτητοι ορισμοί, ενώ πραγματοποιείται και ο σχεδιασμός όλης της υπόλοιπης ερευνητικής διαδικασίας. Στη συνέχεια παράγονται τα διάφορα είδη ερωτημάτων ανάλογα με τις ανάγκες των στόχων που τέθηκαν. Στο τρίτο στάδιο χρησιμοποιούνται τεχνικές ομαδοποίησης από την μηχανική μάθηση για την εκτενή κατηγοριοποίηση των ερωτημάτων και εκπαιδεύεται ένας ταξινομητής βασισμένος στη μηχανική μάθηση για την αντιστοίχιση ερωτημάτων στις κατηγορίες που δημιουργήθηκαν. Τέλος, εκτελέστηκαν πειράματα μεταγραφής ερωτημάτων από τις παραπάνω κατηγορίες για την εύρεση και την απόδοση σε κάθε κατηγορία της συνάρτησης ομοιότητας ερωτημάτων με την καλύτερη απόδοση για αυτήν την κατηγορία.

Τα αποτελέσματα της εργασίας είναι ενθαρρυντικά για την περαιτέρω ερευνητική μελέτη του θέματος και οι δυνατές επεκτάσεις της έχουν να προσφέρουν πολλά στο πεδίο.

Λέξεις Κλειδιά: Ομοιότητα ερωτημάτων, Μεταγραφή Ερωτημάτων, Ομαδοποίηση Ερωτημάτων, Προσεγγιστική Μεταγραφή Ερωτημάτων, Αναπαράσταση Ερωτημάτων, Μηχανική Μάθηση.

Abstract

The present thesis lies in the field of Data Integration. More specifically, it studies query similarity for approximate query rewriting for a Peer – to – Peer database environment. In this type of databases, each peer owns a private database, which has its own independent schema. When a query occurs to a peer, it is transferred across the P2P network to be answered from the other peers as well. Due to each peer's independent database schema for a query to be answered, it has to be translated to the equivalent schema. The translation occurs through rewriting of the query. The rewriting is based on mapping between the schemas, which are not complete in the general case and thus the classical rewriting algorithms are not working. Therefore, the use of an approximate query rewriting algorithm is needed for the rewriting to be successful. Such an algorithm makes use of query similarity functions.

The purpose of the present thesis is to study the effectiveness of the approximate query rewriting algorithm for the different available query similarity functions and for different query categories, so as the process of choosing a similarity function for the rewriting to be automated and the algorithm's effectiveness to be maximized. Our research process makes use of machine learning techniques for the first time in this field and it discusses the pure structural query similarity that has not been researched yet. For that purposes, the process is divided into four steps.

Firstly, the ideas and the goals of the thesis are given and the needed definitions are formed, while the plan of the whole process is discussed. In the next step, the different queries are produced in accordance with our research goals. During the third step, clustering techniques of machine learning are used for the proper categorization of the produced queried and a machine learning classifier is trained upon the created categories. Finally, we performed some rewriting experiments where a few queries of each category were rewritten, in order to find for each category the best suited query similarity function according to rewriting effectiveness.

The results of the research process are promising for a more in-depth study of the query similarity problem in the future and the possible extensions of this thesis have a lot to contribute to the field.

Keywords: Query Similarity, Query Rewriting, Query Clustering, Approximate Query Rewriting, Query Representation, Machine Learning.

Πίνακας Περιεχομένων

Κατάλογος Σχημάτων

1. Εισαγωγή

1.1 Κίνητρο

Η τεχνολογική εξέλιξη και η επιρροή των ηλεκτρονικών υπολογιστών σήμερα είναι κάτι αδιαμφισβήτητο. Ο αριθμός των χρησιμοποιούμενων συσκευών αυξάνει ραγδαία και συνεχώς. Πριν μερικά χρόνια σε κάθε άτομο αναλογούσε με δυσκολία μια συσκευή. Σήμερα, άνθρωποι όλων των ηλικιών κατέχουν και χρησιμοποιούν παραπάνω από μια. Άλλωστε συσκευές με υπολογιστικές δυνατότητες χρησιμοποιούνται πλέον σχεδόν σε κάθε πτυχή της καθημερινότητας και η πληροφορική και οι εφαρμογές αυτής προσφέρουν λύσεις σε κάθε επιστημονικό και εργασιακό κλάδο. Ταυτόχρονα ο Παγκόσμιος Ιστός έχει εδραιωθεί πλέον στις ζωές μας και η πρόσβαση σε αυτόν θεωρείται βασική καθημερινή ανάγκη. Όλα αυτά έχουν οδηγήσει τα τελευταία χρόνια σε μια ραγδαία καθημερινή παραγωγή τεράστιου όγκου δεδομένων και πληροφορίας, χαρακτηρίζοντας την εποχή μας ως "εποχή των Μεγάλων Δεδομένων" (Big Data Era).

Η νέα αυτή εποχή των μεγάλων δεδομένων έχει ως αποτέλεσμα και την εισαγωγή νέων προβλημάτων και αναγκών στον κλάδο της Διαχείρισης Δεδομένων. Κατ' αρχήν, με την εξάπλωση του Παγκόσμιου Ιστού και των υπολογιστών αυξήθηκε ο αριθμός των χρηστών που απαιτούν πρόσβαση στις πηγές πληροφορίας προκαλώντας φαινόμενα συμφόρησης στα δίκτυα και στις κλασσικές αρχιτεκτονικές για την Διαχείριση Δεδομένων. Ως εκ τούτου νέες αρχιτεκτονικές όπως τα δίκτυα ομότιμων κόμβων (peer – to – peer / P2P) και οι κατανεμημένες λύσεις, έρχονται πλέον να αντικαταστήσουν τις έως τώρα επικρατούσες κεντρικοποιημένες αρχιτεκτονικές διαμοιράζοντας τον φόρτο εργασίας και την πληροφορία σε πολλούς εξυπηρετητές. Ταυτόχρονα ο νέος αυτός μεγάλος όγκος δεδομένων αποτελεί τεράστια πηγή πληροφορίας για τις εταιρίες που προσπαθούν να αναπτύξουν υποδομές αλλά και λογισμικό για να καταφέρουν να διαχειριστούν και να αξιοποιήσουν στο έπακρο τις πληροφορίες που αυτά κρύβουν. Έτσι δημιουργούν μεγάλες Αποθήκες Δεδομένων (Data Warehouses) για να αποθηκεύσουν τα δεδομένα τους και να τα επεξεργαστούν αποτελεσματικά, ή αναζητούν εξωτερικές πηγές δεδομένων για να εφαρμόσουν σε αυτά τεχνικές εξόρυξης πληροφορίας και μηχανικής μάθησης.

Όμως οι μη κεντρικοποιημένες αρχιτεκτονικές αλλά και γενικότερα οι εξωτερικές πηγές δεδομένων, είτε αυτές χρησιμοποιούνται για εταιρικούς σκοπούς ή όχι, παρότι μπορεί εννοιολογικά να ταυτίζονται, δεν εξασφαλίζουν πάντοτε την ύπαρξη ίδιου σχήματος και κατ' επέκταση οργάνωσης. Ως αποτέλεσμα η πρόσβαση στην περιεχόμενη πληροφορία να αντιμετωπίζει δυσκολίες. Παρότι στις ομογενείς κατανεμημένες βάσεις δεδομένων κάθε βάση-κόμβος έχει το ίδιο σχήμα καθώς αποτελεί τμήμα μια αρχικής ενιαίας βάσης η οποία έχει κατατμηθεί, σε peer – to – peer βάσεις δεδομένων αυτό δεν αποτελεί τον κανόνα παρά την εξαίρεση. Συνεπώς εφόσον συνήθως όλες οι peer – to – peer βάσεις του δικτύου δε μιλούν την ίδια γλώσσα, είναι απαραίτητη η μετάφραση των ερωτημάτων στο σχήμα του εκάστοτε κόμβου ώστε να είναι εφικτή η ανάκτηση της πληροφορίας. Για την μετάφραση αυτή διάφοροι μηχανισμοί έχουν αναπτυχθεί. Ως επί των πλείστων οι μηχανισμοί αυτοί περιλαμβάνουν αλγόριθμους μεταγραφής ερωτημάτων μέσω αντιστοιχίσεων (mappings) εκφρασμένων συνήθως ως όψεις (views). Παρότι οι αλγόριθμοι αυτοί είναι αποτελεσματικοί

προϋποθέτουν την ύπαρξη πλήρων αντιστοιχίσεων μεταξύ των σχέσεων των σχημάτων. Πρακτικά κάτι τέτοιο είναι σπάνια εφικτό για διάφορους λόγους. Ως συνέπεια στις πρακτικές εφαρμογές οι αλγόριθμοι αυτοί αποτυγχάνουν.

Μια λύση στο πρόβλημα της πρακτικής εφαρμογής των αναπτυχθέντων αλγορίθμων έδωσε ένας νέος αλγόριθμος [1] που αξιοποιεί έναν από τους προϋπάρχοντες αλγόριθμους, τον MiniCon [2], και προσπαθεί να μεταγράψει το ερώτημα προσεγγιστικά. Με αυτό τον τρόπο είναι δυνατή η μετεγγραφή του ερωτήματος από το ένα σχήμα στο άλλο ακόμα και όταν οι αντιστοιχήσεις δεν είναι πλήρεις ανακτώντας έτσι τη ζητούμενη πληροφορία στο βαθμό που το επιτρέπουν οι υπάρχοντες αντιστοιχήσεις. Αν και τα αποτελέσματα του αλγορίθμου είναι ήδη ικανοποιητικά, ίσως υπάρχει ακόμα χώρος για βελτίωση μιας και διάφορες πτυχές του αλγορίθμου δεν έχουν ερευνηθεί διεξοδικά. Επί προσθέτως οι εξελίξεις στην μηχανική μάθηση μας δίνουν πλέον πολύ περισσότερες δυνατότητες εξερεύνησης των διάφορων παραμέτρων, οδηγώντας σε μια γενικότερη τάση χρήσης τεχνικών μηχανικής μάθησης για την επίλυση διάφορων προβλημάτων και την βελτιστοποίηση υπαρχουσών λύσεων στον ευρύ τομέα της Διαχείρισης Δεδομένων. Ακολουθώντας αυτή την τάση τεχνικές μηχανικής μάθησης θα μπορούσαν να χρησιμοποιηθούν για την βελτιστοποίηση των παραμέτρων του παραπάνω αναφερθέντος αλγορίθμου.

1.2 Αντικείμενο της διπλωματικής

Η παρούσα διπλωματική εργασία έχει στόχο την εξερεύνηση των συναρτήσεων ομοιότητας ερωτημάτων και γενικότερα της ομοιότητας ερωτημάτων μέσα στο πλαίσιο της προσεγγιστικής μεταγραφής ερωτημάτων κάνοντας χρήση τεχνικών μηχανικής μάθησης και εκτιμώντας τα αποτελέσματα χρησιμοποιώντας μεταξύ άλλων μεθόδους μέτρησης απώλειας πληροφορίας. Εκ φύσεως η εξερεύνηση αυτή αποτελεί κυρίως μια πειραματική διαδικασία και μπορεί να αναλυθεί στα εξής στάδια:

Στο πρώτο στάδιο της ερευνητικής διαδικασίας πραγματοποιήθηκε ο σχεδιασμός και η προετοιμασία της ίδιας της βάσης δεδομένων που χρησιμοποιήθηκε στο σύνολο των πειραμάτων, καθώς και των δεδομένων αυτής αλλά και των ερωτημάτων που μπορούν να πραγματοποιηθούν σε αυτή και που συγκεντρώνουν το μεγαλύτερο ενδιαφέρον στη διαδικασία. Αποτελεί το πλέον σημαντικό στάδιο της ερευνητικής προσέγγισης καθώς ο σωστός σχεδιασμός και η λήψη αποφάσεων σχετικά με τις υποθέσεις και τα ερωτήματα που θα κληθούν να απαντηθούν σε αυτή καθορίζουν και όλη την πορεία της έρευνας. Εδώ έγινε η λήψη αποφάσεων σχετικά με το σχήμα και το είδος της βάσης δεδομένων, το είδος των ερωτημάτων που θα ερευνηθούν καθώς και τις παραμέτρους οι οποίες θα εξεταστούν. Επίσης καθώς τα δεδομένα σπάνια είναι διαθέσιμα σε μορφές οι οποίες είναι εύκολο να αξιοποιηθούν, στο στάδιο αυτό τα δεδομένα μας οργανώθηκαν στην αναφερθείσα βάση δεδομένων και πέρασαν από μία διαδικασία καθαρισμού για τυχόν παρατυπίες που δημιουργούσαν προβλήματα στο υποκείμενο σύστημα διαχείρισης βάσεων δεδομένων.

Στο δεύτερο στάδιο, πραγματοποιήθηκε η παραγωγή των ερωτημάτων προς την βάση δεδομένων σύμφωνα με τις αποφάσεις που πάρθηκαν στο προηγούμενο στάδιο. Για την παραγωγή των ερωτημάτων υλοποιήθηκε κατάλληλο λογισμικό ώστε να εξασφαλίσει και την τυχαιότητα των

παραγόμενων ερωτημάτων αλλά και ένα μεγάλο και επαρκές πλήθος ερωτημάτων που είναι αναγκαίο για την εφαρμογή μηχανισμών μηχανικής μάθησης στην πειραματική διαδικασία. Από τα ερωτήματα που παράχθηκαν έγινε δειγματοληψία ώστε να συγκροτηθεί ένα τελικό σύνολο ερωτημάτων τόσο για την εκπαίδευση των συστημάτων μηχανικής μάθησης όσο και για την τελική αξιολόγηση των υποθέσεων και την λήψη των αποτελεσμάτων της έρευνας.

Στο τρίτο στάδιο της διαδικασίας, σχεδιάστηκε και υλοποιήθηκε λογισμικό το οποίο με χρήση μηχανισμών μηχανικής μάθησης έχει ως στόχο να ανακαλύψει συσχετίσεις μεταξύ των ερωτημάτων που παράχθηκαν στο προηγούμενο στάδιο και να τα ταξινομήσει αποδίδοντας τους ετικέτες. Σε δεύτερο χρόνο, το σύστημα αυτό χρησιμοποιώντας τα παραδείγματα που πλέον έχουν ετικέτες μαθαίνει και ταξινομεί κάθε εισερχόμενο ερώτημα στην αντίστοιχη κατηγορία. Η διαδικασία αυτή είναι απαραίτητη για την αυτοματοποίηση της επιλογής των παραμέτρων του αλγορίθμου προσεγγιστικής μεταγραφής ερωτημάτων.

Στο τέταρτο στάδιο, χρησιμοποιήθηκε το προηγουμένως ανεπτυγμένο σύστημα μηχανικής μάθησης σε συνδυασμό με τον μηχανισμό προσεγγιστικής μεταγραφής ερωτημάτων, όπως αυτός είχε υλοποιηθεί σε προηγούμενη εργασία [1], με σκοπό την δημιουργία ενός ενιαίου συστήματος. Το σύστημα αυτό αναλόγως με το είδος του ερωτήματος που λαμβάνει προς μεταγραφή επιλέγει αυτόματα την κατάλληλη συνάρτηση ομοιότητας για να χρησιμοποιηθεί από τον αλγόριθμο μεταγραφής. Η συνάρτηση που επιλέγεται είναι αυτή που βελτιστοποιεί την απόδοση του αλγορίθμου όσον αφορά την ποιότητα του ερωτήματος που λαμβάνεται μετά τη μεταγραφή. Η μέτρηση αυτής της ποιότητας λαμβάνεται με διάφορες μορφές.

Συνοψίζοντας, η παρούσα εργασία περιλαμβάνει τον σχεδιασμό μιας πειραματικής διαδικασίας, την προετοιμασία των πειραματικών δεδομένων, σχεδιασμό και την υλοποίηση ενός συστήματος μηχανικής μάθησης για την κατηγοριοποίηση των ερωτημάτων και την πειραματική αξιολόγηση της ποιότητας της αρχικής υπόθεσης της έρευνας στο συγκεκριμένο περιβάλλον.

1.3 Οργάνωση του τόμου

Η παρούσα διπλωματική εργασία χωρίζεται σε 6 κεφάλαια.

Το πρώτο κεφάλαιο αποτελεί μια γενική εισαγωγή στο αντικείμενο, στο κίνητρο και τους στόχους της εργασίας.

Στο δεύτερο κεφάλαιο αναπτύσσεται η σχετική θεωρία και οι σχετικές προϋπάρχουσες εργασίες που αποτελούν τη βάση για το σχεδιασμό, τις ιδέες και την υλοποίηση της ερευνητικής διαδικασίας. Αρχικά γίνεται μια γενική εισαγωγή περί των δικτύων P2P, των Αποθηκών Δεδομένων και των ειδικών κατηγοριών ερωτημάτων που σχετίζονται με ανάλογα ερευνητικά ενδιαφέροντα. Στη συνέχεια αναλύεται η σχέση μεταξύ της Διαχείρισης Δεδομένων και της Μηχανικής Μάθησης και παρουσιάζονται εργασίες και λύσεις που αποτέλεσαν εφαλτήριο των ιδεών για την παρούσα εργασία. Τέλος γίνεται μια εισαγωγή στον αλγόριθμο προσεγγιστικής μεταγραφής ερωτημάτων, στις συναρτήσεις ομοιότητας που αυτός χρησιμοποιεί καθώς και σε σχετικές εργασίες γύρω από την ομοιότητα ερωτημάτων.

Το τρίτο κεφάλαιο αφορά τον σχεδιασμό και την υλοποίηση της παρούσας ερευνητικής διαδικασίας. Παρουσιάζονται αναλυτικά οι υποθέσεις προς εξερεύνηση, οι βασικές ιδέες, ο σχεδιασμός και οι παραδοχές για κάθε στάδιο της εργασίας. Γίνεται εκτενή αναφορά στις επιλογές υλοποίησης και επεξηγούνται αναλυτικά διάφορα κομμάτια του λογισμικού για όλα τα στάδια της εργασίας.

Στο τέταρτο κεφάλαιο αναλύονται τα αποτελέσματα της πειραματικής διαδικασίας και αξιολογείται η επαλήθευση ή όχι των αρχικών υποθέσεων. Αρχικά παρουσιάζονται τα πειράματα που πραγματοποιήθηκαν και τα αποτελέσματα αυτών. Στη συνέχεια, αναλύονται τα παραγόμενα αποτελέσματα και εξετάζεται αν επαληθεύουν τις αρχικές υποθέσεις κι αν προσφέρουν κάποια διαφορετική οπτική και λύση από τις ήδη υπάρχουσες.

Το πέμπτο κεφάλαιο αποτελεί τον επίλογο της παρούσας εργασίας. Εδώ συνοψίζονται τα αποτελέσματα της εργασίας, εκφράζεται ένα γενικό συμπέρασμα και προτείνονται σημαντικές μελλοντικές επεκτάσεις και ιδέες προς έρευνα σχετικές με το αντικείμενο της εργασίας.

Τέλος στο έκτο κεφάλαιο παρατίθεται η σχετική βιβλιογραφία που χρησιμοποιήθηκε στη διάρκεια διεκπεραίωσης της εργασίας και περιέχεται και σε αυτόν εδώ τον τόμο.

2. Θεωρία και Σχετικές Εργασίες

Το αντικείμενο της παρούσας εργασίας, όπως έχει ήδη αναφερθεί, είναι η μελέτη και η εξερεύνηση τα ομοιότητας ερωτημάτων και των συναρτήσεων που την υπολογίζουν. Ως εργαλείο για το σκοπό αυτό έχει επιλεγεί η χρήση τεχνικών μηχανικής μάθησης ενώ το πλαίσιο στο οποίο εξετάζεται η ομοιότητα σχετίζεται με τη μεταγραφή ερωτημάτων. Πριν προχωρήσουμε στην περιγραφή της δουλειάς που πραγματοποιήθηκε στην παρούσα εργασία είναι σημαντικό να συζητήσουμε τις υπάρχουσες σχετικές εργασίες αλλά και να δώσουμε διάφορες θεωρητικές λεπτομέρειες των διαφόρων επιστημονικών πεδίων και των διαφόρων τεχνικών που προσπαθούμε να συνδυάσουμε στην υλοποίηση της. Ξεκινάμε δίνοντας κάποιες βασικές πληροφορίες σχετικές με τα περιβάλλοντα εφαρμογής της μεταγραφής ερωτημάτων που αποτελεί το πλαίσιο εφαρμογής της εργασίας. Συνεχίζουμε δίνοντας το απαραίτητο θεωρητικό υπόβαθρο για την κατανόηση τόσο της μεταγραφής ερωτημάτων όσο και της ομοιότητας αυτών και παρουσιάζουμε σχετικές εργασίες που αποτέλεσαν πηγή έμπνευσης και προβληματισμού. Τέλος αναφερόμαστε στη μηχανική μάθηση που αποτελεί το βασικό εργαλείο της εργασίας μας, δίνοντας τους απαραίτητους ορισμούς, αναδεικνύοντας την ολοένα και αυξανόμενη σχέση της με τη Διαχείριση Δεδομένων και συζητώντας τεχνικές της που χρησιμοποιήθηκαν στη δική μας αλλά και σε άλλες σχετικές εργασίες που επίσης παρουσιάζονται.

2.1 Δίκτυα Ομότιμων Κόμβων και Βάσεις Δεδομένων

Τα δίκτυα ομότιμων κόμβων αποτελούν μια σύγχρονη μορφή δικτύων που έχουν ως στόχο την ίση κατανομή του φόρτου εργασίας μεταξύ των συμμετεχόντων. Σύμφωνα με την εργασία [15] ως δίκτυο ισότιμων κόμβων μπορεί να θεωρηθεί κάθε δίκτυο στο οποίο οι συμμετέχοντας μοιράζονται ένα μέρος των υπολογιστικών και δικτυακών τους δυνατοτήτων, είτε πρόκειται για επεξεργαστική ισχύ, χώρο αποθήκευσης ή χωρητικότητα δικτύου, ώστε να μπορέσει το δίκτυο ως σύνολο να παράσχει το περιεχόμενο και τις υπηρεσίες του όποιες κι αν είναι αυτές. Ταυτόχρονα, θα πρέπει κάθε κόμβος του δικτύου να έχει άμεση πρόσβαση σε αυτές τις δυνατότητες χωρίς ενδιάμεσες οντότητες. Έτσι κάθε συμμετέχοντας κόμβος είναι συγχρόνως και πάροχος και καταναλωτής των συνολικών δυνατοτήτων του δικτύου.

Τα δίκτυα ομότιμων κόμβων γνώρισαν μεγάλη άνθιση και πολλές εφαρμογές αναπτύχθηκαν πάνω σε αυτά. Χαρακτηριστικά παραδείγματα υπηρεσιών και αντίστοιχων εφαρμογών είναι ο διαμοιρασμός αρχείων (file sharing) με εφαρμογές όπως το Gnutella και το Freenet, ο κατανεμημένος υπολογισμός (distributed computing) με πλήθος εφαρμογών όπου απαιτείται υψηλή υπολογιστική ισχύς, η άμεση επικοινωνία με μηνύματα με εφαρμογές όπως το MSN Messenger και το AOL Instant Messenger και ο διαμοιρασμός δεδομένων (data sharing) με κύρια εφαρμογή του τις βάσεις δεδομένων ομότιμων κόμβων (P2P).

Οι βάσεις δεδομένων ομότιμων κόμβων ουσιαστικά απαρτίζονται από κλασσικές σχεσιακές βάσεις δεδομένων σε κάθε κόμβο του δικτύου ομότιμων κόμβων. Ο κάθε κόμβος ενός δικτύου

ομότιμων κόμβων είναι ανεξάρτητος και συνεπώς η βάση του διαθέτει ένα δικό της τοπικό σχήμα το οποίο αν και έχει την ίδια θεματολογία με τις βάσεις των άλλων κόμβων διαφέρει σε οργάνωση. Επίσης, σε αυτά τα συστήματα όταν ένας κόμβος θέτει κάποιο ερώτημα στη βάση δεδομένων του κοινοποιεί το ερώτημα και προς τις βάσεις των υπόλοιπων κόμβων του δικτύου. Όπως είπαμε όμως κάθε βάση έχει το δικό της σχήμα και συνεπώς για να μπορέσει να εκτελεστεί το κοινοποιημένο μήνυμα σε μια άλλη βάση ενός άλλου κόμβου θα πρέπει να έχει μεταφραστεί κατάλληλα. Η μετάφραση αυτή γίνεται μέσω όψεων και υπάρχει πλήθος εργασιών, όπως οι [19], [20], πάνω στον τομέα. Όμως όλες οι παραπάνω κλασσικές εργασίες μεταγραφής ερωτημάτων υποθέτουν όπως έχει ήδη ειπωθεί πλήρεις αντιστοιχίσεις. Μια λύση στο πρόβλημα αυτό δίνει η εργασία [1] κι ο προσεγγιστικός αλγόριθμος που προτείνει.

2.2 Αποθήκες Δεδομένων και Σχήματα

Με την αύξηση των δεδομένων κατανεμημένες υποδομές χρησιμοποιούνται πλέον και στις δομές που υλοποιούν αποθήκες δεδομένων. Συνεπώς προβλήματα μεταγραφής όπως αυτά που παρουσιάζονται στις P2P βάσεις δεδομένων μπορούν να εμφανιστούν κι εδώ. Σύμφωνα με την Oracle [16], μια αποθήκη δεδομένων είναι μια σχεσιακή βάση που δεν έχει σκοπό την επεξεργασία συναλλαγών αλλά την ανάλυση των δεδομένων που περιέχει. Συνήθως τα δεδομένα αυτά έχουν ιστορικό χαρακτήρα και μπορεί να προέρχονται από διάφορες πηγές. Οι αποθήκες δεδομένων συνήθως ακολουθούν ένα εκ των star ή snowflake σχέδιο για το σχήμα τους.

Το star schema [17] έχει έναν κεντρικό πίνακα του οποίου το πρωτεύων κλειδί (primary key) είναι σύνθετο, δηλαδή αποτελείται από πολλαπλά γνωρίσματα. Κάθε ένα από αυτά τα γνωρίσματα είναι ένα ξένο κλειδί σε έναν από τους εναπομείναντες πίνακες. Μια τέτοια εξάρτηση ξένου κλειδιού υπάρχει για κάθε έναν από αυτούς τους πίνακες, ενώ δεν υπάρχουν πουθενά αλλού στο σχήμα άλλα ξένα κλειδιά.

Αντίστοιχα, το snowflake schema [18] έχει έναν κεντρικό πίνακα του οποίου το πρωτεύων κλειδί είναι σύνθετο, δηλαδή αποτελείται από πολλαπλά γνωρίσματα. Κάθε ένα από αυτά τα γνωρίσματα είναι ένα ξένο κλειδί προς κάποιον από τους εναπομείναντες πίνακες, οι οποίοι, με τη σειρά τους, έχουν κάποιο από τα γνωρίσματα που δεν συμμετέχουν στο πρωτεύον κλειδί τους ως ξένο κλειδί προς κάποιον άλλο διαφορετικό πίνακα. Αυτό συνεχίζεται αναδρομικά με τους εναπομείναντες πίνακες, μέχρι αυτοί να εξαντληθούν, σχηματίζοντας αλυσίδες ή δέντρα από εξαρτήσεις ξένων κλειδιών με ρίζα τον κεντρικό πίνακα, δηλαδή κάθε πίνακας στο σχήμα (εκτός από τον κεντρικό πίνακα) δείχνεται από ακριβώς ένα τέτοιο ξένο κλειδί.

2.3 Συζευκτικά ερωτήματα

Σύμφωνα με την πηγή [26], μπορούμε να ορίσουμε τα συζευκτικά ερωτήματα ως εξής:

Τα συζευκτικά ερωτήματα είναι ερωτήματα πρώτης τάξης μιας συγκεκριμένης μορφής

$$\{< e_1, \dots, e_n > \mid \exists x_1 \dots x_m \psi\}$$

όπου ψ είναι μια (χωρίς ισότητα) σύζευξη σχεσιακών ατόμων, δηλαδή ατόμων της μορφής $R(d_1, \dots, d_k)$ όπου $d_1, \dots, d_k$ είναι μεταβλητές σταθερών. Επιπροσθέτως, είναι απαραίτητο ότι οποιαδήποτε μεταβλητή μεταξύ των $e_1, \dots, e_n$ πρέπει επίσης να υπάρχει και σε ένα από τα σχεσιακά άτομα του ψ. Η τελευταία αυτή συνθήκη, καλείται περιορισμός εύρους και είναι απαραίτητη για την ανεξαρτησία τομέα.

Αν δούμε τα συζευκτικά ερωτήματα με τη οπτική του λογικού προγραμματισμού, τότε η πλειάδα εξόδου ενός συζευκτικού ερωτήματος καλείται μερικές φορές «κεφαλή» του ερωτήματος και η σύζευξη των ατόμων μέρος του «σώματος» του ερωτήματος.

Ο παραπάνω ορισμός αναφέρεται μονάχα σε ερωτήματα που δεν περιέχουν ισότητα στο σώμα του ερωτήματος. Εάν η κλάση CQ περιέχει όλα τα ερωτήματα αυτού του τύπου, τότε η κλάση $CQ^{=}$ η οποία ορίζει τα ερωτήματα που μπορούν αν περιέχουν ισότητες στο σώμα τους ορίζει τα ίδια ερωτήματα με κάποιους επιπλέον περιορισμούς. Ανάλογα με τον προηγούμενο περιορισμό εύρους, για την κλάση $CQ^{=}$ θα πρέπει κάθε μεταβλητή μεταξύ των $e_1, \dots, e_n$, ως συνέπεια των ατομικών ισοτήτων του ψ, ισούται με κάποια σταθερά ή κάποια μεταβλητή που υπάρχει σε ένα από τα σχεσιακά άτομα του ψ. Επίσης τα ερωτήματα της κλάσης $CQ^{=}$ έχουν μια πρόσθετη ευχάριστη ιδιότητα ότι η κεφαλή ενός ερωτήματος περιορίζεται στο να αποτελείται μόνο από διακριτές μεταβλητές.

Τα συζευκτικά ερωτήματα ανήκουν ένα πολύ συγκεκριμένο κομμάτι της σχεσιακής άλγεβρας. Συγκεκριμένα στο κομμάτι που χρησιμοποιεί μόνο επιλογή, προβολή και καρτεσιανό γινόμενο και ονομάζεται SPC άλγεβρα. Μέσω της SPC άλγεβρας είναι φανερή η αντιστοίχιση των συζευκτικών ερωτημάτων σε συγκεκριμένα SQL προγράμματα.

Για παράδειγμα, το συζευκτικό ερώτημα $ans(x, y) : - R(x, z), x = c, S(x, y, z)$ ανήκει στην $CQ^{=}$ κλάση και αντιστοιχεί στο SQL ερώτημα:

$$SELECT\ r.1, s.2$$
$$FROM\ R\ r, S\ s$$
$$WHERE\ r.1 = c\ AND\ s.1 = r.1\ AND\ r.3 = s.2$$

Συνοψίζοντας τα συζευκτικά ερωτήματα είναι ερωτήματα πρώτης τάξης τα οποία είναι ταυτόχρονα και πρακτικά εκφραστικά και σχετικά εύκολα διαχειρίσιμα από αλγοριθμικής άποψης. Ως ένα υποσύνολο του σχεσιακού λογισμού ορίζονται από μια φόρμουλα που χρησιμοποιεί μόνο άτομα, συζεύξεις και υπαρξιακή ποσοτικοποίηση. Λόγω και της στενής τους σχέσης με SQL προγράμματα που περιέχουν την where πρόταση παίζουν πολύ σημαντικό ρόλο σε διάφορες περιπτώσεις βελτιστοποίησης και ανασύνθεσης.

2.4 Συνδέσεις (Joins)

Η συνδέσεις και οι λειτουργία σύνδεσης (join operation) αποτελούν βασικές έννοιες των βάσεων δεδομένων και της σχεσιακής άλγεβρας. Όπως αναφέρει η εργασία [27] η λειτουργία σύνδεσης των βάσεων δεδομένων χρησιμοποιείται για να συνδυάσει πλειάδες από δύο διαφορετικές σχέσεις με βάση κάποια κοινή πληροφορία. ΟΙ πλειάδες συνδυάζονται όταν ικανοποιούν μια καθορισμένη συνθήκη σύνδεσης.

Στην SQL, οι συνδέσεις παρουσιάζονται με ακριβώς αυτόν τον τρόπο. Σε ένα SQL ερώτημα δύο πίνακες συνδυάζονται με βάση κάποια συνθήκη σύνδεσης η οποία εκφράζεται μέσω μιας συνθήκης ισότητας πάνω σε ένα ή και περισσότερα γνωρίσματα στη where πρόταση του ερωτήματος.

Για παράδειγμα στο ερώτημα:

$$SELECT\ r.1, s.2$$
$$FROM\ R\ r, S\ s$$
$$WHERE\ r.1 = c\ AND\ s.1 = r.1\ AND\ r.3 = s.2$$

οι ισότητες $s.1 = r.1$ και $r.3 = s.2$ εκφράζουν μια συνθήκη σύνδεσης των δύο πινάκων και για τον υπολογισμό του αποτελέσματος του ερωτήματος θα συνδυαστούν οι πλειάδες των πινάκων που ικανοποιούν αυτή την κοινή πληροφορία.

Όπως ήδη αναφέραμε, υπάρχει στενή αντιστοιχία μεταξύ των SQL ερωτημάτων και των συζευκτικών ερωτημάτων. Συνεπώς όπως τα πρώτα περιέχουν και εκφράζουν συνδέσεις μεταξύ των πινάκων, έτσι και τα συζευκτικά ερωτήματα μπορούν να εκφράσουν την ίδια λειτουργία. Στα συζευκτικά ερωτήματα όμως δεν έχουμε ισότητες. Εδώ οι συνδέσεις παρουσιάζονται με επανάληψη των ονομάτων των μεταβλητών στα άτομα του σώματος των ερωτημάτων.

Για παράδειγμα στο συζευκτικό ερώτημα:

$$ans(x, y) :- R(x, z), x = c, S(x, y, z)$$

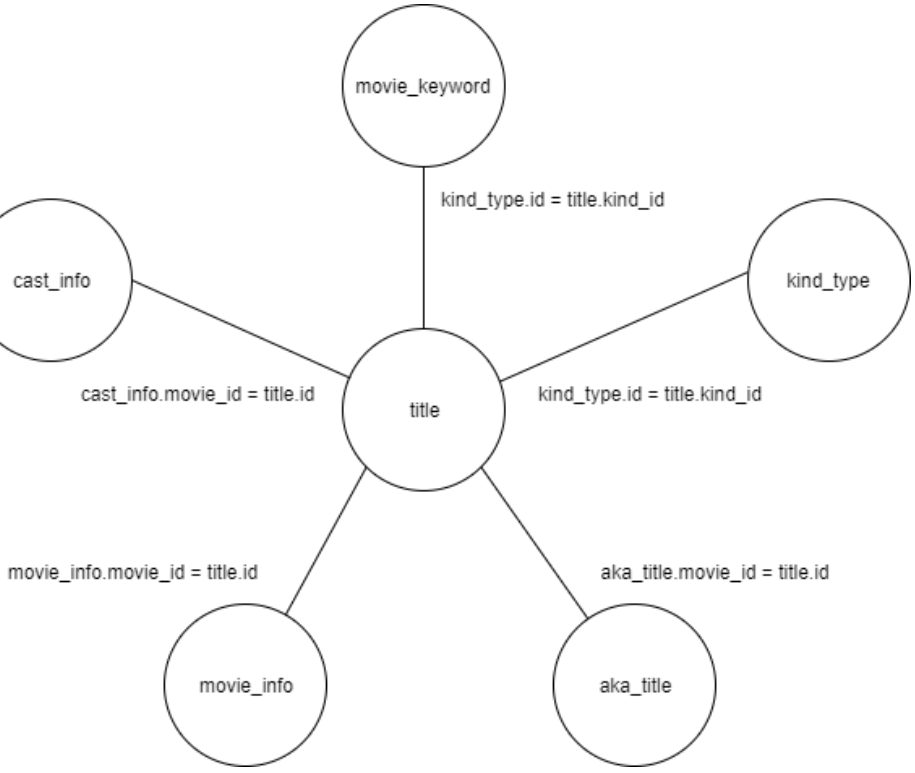
η σύνδεση των δύο ατόμων (πινάκων) γίνεται μέσω των κοινών μεταβλητών x, z και αποτυπώνει την ίδια συνθήκη ισότητας με το προηγούμενο SQL ερώτημα.

2.5 Γράφοι Ερωτημάτων

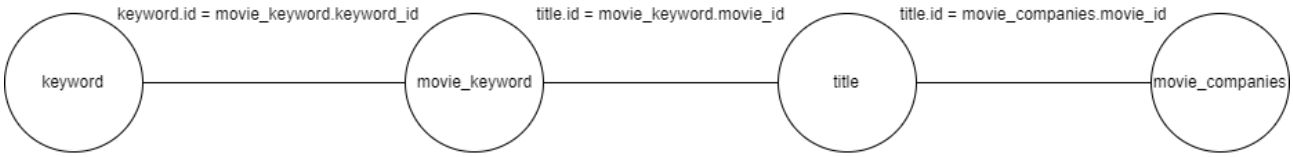
Κάθε ερώτημα προς μία βάση δεδομένων μπορεί να αντιστοιχιστεί σε έναν γράφο στον οποίο αντικατοπτρίζονται όλες οι συνδέσεις (joins) μεταξύ των πινάκων της βάσης που συμμετέχουν στο ερώτημα. Αυτή η αναπαράσταση του ερωτήματος αποτελεί ένα πολύ χρήσιμο εργαλείο σε μεγάλο εύρος προβλημάτων όπως η βελτιστοποίηση ερωτημάτων (query optimization) [21] και το πρόβλημα της σειράς ικανοποίησης των συνδέσεων (join ordering problem) [22]. Άλλωστε έχει παρατηρηθεί πως γράφοι ερωτημάτων με διαφορετικές δομές παρουσιάζουν διαφορετικά προβλήματα για τη βελτιστοποίηση τους ([23],[24]). Ταυτόχρονα, οι γράφοι των ερωτημάτων αποτελούν έναν καλό τρόπο κατηγοριοποίησης αυτών και έκφρασης των ακραίων περιπτώσεων που μπορεί να προκύψουν.

Ακολουθώντας αυτή την προσέγγιση για τα ερωτήματα μπορούμε να διακρίνουμε τις εξής κατηγορίες [25]:

- Chain (join) query graphs: Σε αυτή την κατηγορία γράφων ερωτημάτων κάθε κόμβος (πίνακας) έχει τουλάχιστον έναν γείτονα (συνδεδεμένο πίνακα) και το πολύ δύο.
- Star (join) query graphs: Οι star γράφοι ερωτημάτων, κατ' αντιστοιχία με το star schema που έχει ήδη αναφερθεί, αποτελούνται από έναν κεντρικό κόμβο (πίνακα), ο οποίος συνδέεται με όλους τους άλλους κόμβους (πίνακες) του γράφου (ερωτήματος). Όλες οι συνδέσεις των πινάκων γίνονται μέσω του κεντρικού κόμβου (πίνακα).
- Clique (join) query graphs: Σε αυτό το είδος γράφων ερωτημάτων κάθε κόμβος (πίνακας) συνδέεται με όλους τους άλλους κόμβους (πίνακες) του γράφου.
- Cycle (join) query graphs: Οι κόμβοι αυτού του είδους γράφου ερωτημάτων συνδέονται σε κλειστή μορφή. Κάθε κόμβος (πίνακας) του γράφου έχει ακριβώς δύο γείτονες και ο αριθμός των κόμβων και των συνδέσεων είναι ακριβώς ο ίδιος.



Σχήμα 2.1: Star query graph



Σχήμα 2.2: Chain query graph

2.6 Προσεγγιστικός Αλγόριθμος Μεταγραφής

Όπως έχει ήδη αναφερθεί πολλές φορές, οι κλασσικοί αλγόριθμοι μεταγραφής ερωτημάτων βασίζονται σε υποθέσεις που δύσκολα ικανοποιούνται στην πραγματικότητα. Ως αποτέλεσμα, οι αλγόριθμοι αυτοί δεν μπορούν να εφαρμοστούν σε πραγματικές εφαρμογές και η ανάγκη για αποτελεσματική μεταγραφή ερωτημάτων παραμένει. Μια λύση σε αυτή την κατεύθυνση δίνει η εργασία [1] με το σύστημα προσεγγιστικής μεταγραφής που προτείνει.

Όπως και οι κλασσικοί αλγόριθμοι μεταγραφής ερωτημάτων έτσι και ο αλγόριθμος μεταγραφής του συστήματος βασίζεται στις αντιστοιχίσεις (mappings) μεταξύ των διαφόρων πηγών δεδομένων που εμπλέκονται. Η βασική διαφορά τους όμως είναι ότι ο προσεγγιστικός αλγόριθμος δεν υποθέτει πλήρεις αντιστοιχίσεις (complete mappings) αλλά δουλεύει με οποιαδήποτε υπαρκτή αντιστοίχιση. Οι αντιστοιχίσεις αυτές εκφράζονται ως Local-as-Views (LAV) αντιστοιχίσεις, δηλαδή παρουσιάζοντας το τοπικό σχήμα ως όψεις του άλλου απομακρυσμένου σχήματος. Ο σύνηθες τρόπος παρουσίασης τέτοιων αντιστοιχίσεων είναι σε συζευκτική μορφή. Για την πρακτική εφαρμογή τους όμως στον αλγόριθμο μεταγραφής του συστήματος οι αντιστοιχίσεις απεικονίζονται πραγματικά ως όψεις.

Βασική ιδέα του αλγορίθμου είναι η αξιοποίηση των κλασσικών αλγορίθμων μεταγραφής με LAV αντιστοιχίσεις με την συμπλήρωση επιπλέον λειτουργιών για την επίτευξη μιας μεταγραφής του ερωτήματος ναι μεν προσεγγιστικά αλλά με τον καλύτερο δυνατό τρόπο. Για την επίτευξη αυτού του στόχου το σύστημα επιχειρεί να μετατρέψει, ή αλλιώς να "χαλαρώσει", το ερώτημα ώστε αυτό να μπορέσει να μεταγραφεί χρησιμοποιώντας τους κλασσικούς αλγόριθμους.

Το σύστημα δουλεύει σε δύο βήματα. Στο πρώτο βήμα εκτελεί μια προεπεξεργασία των ερωτημάτων προς μεταγραφή. Στο βήμα αυτό, δηλαδή, γίνεται η χαλάρωση του ερωτήματος με σκοπό να βρεθούν τα κατάλληλα υποερωτήματα που μπορούν να μεταφραστούν επιτυχώς μέσω των διαθέσιμων αντιστοιχίσεων. Για την εύρεση του κατάλληλου υποερωτήματος που μπορεί να μεταφραστεί η διαδικασία της προεπεξεργασίας λαμβάνει υπόψιν της την ομοιότητα των υποψηφίων υποερωτημάτων με το αρχικό ερώτημα. Περισσότερες λεπτομέρειες σχετικά με την ομοιότητα ερωτημάτων θα δοθούν στη συνέχεια. Για την κατασκευή των υποψήφιων υποερωτημάτων, κατά την προεπεξεργασία, διατρέχεται το δοθέν ερώτημα και αποκόπτεται από αυτό κάθε ιδιότητα που δεν περιέχεται στην κεφαλή της αντίστοιχης LAV αντιστοίχισης. Επίσης ένα υπάρχουν σύνδεσμοι (joins)

με δύο ιδιότητες εκ των οποίων έστω και η μία να μην μπορεί να μεταφραστεί , αριθμητικοί περιορισμοί ή σταθερές που δεν μπορούν να μεταφραστούν, τότε αυτοί/ες αποκόπτονται. Έτσι προκύπτουν διάφορα μειωμένα υποερωτήματα τα οποία εξετάζονται προς μετάφραση με προτεραιότητα με βάση το πόσο μοιάζουν με το αρχικό. Έτσι στο δεύτερο βήμα τα υποψήφια μειωμένα υποερωτήματα μεταφράζονται κατά προτεραιότητα με βάση τον κλασικό αλγόριθμο μεταγραφής Minicon [2].

Σύμφωνα με την εργασία [1], η συνάρτηση που υπολογίζει την ομοιότητα των ερωτημάτων που χρησιμοποιείται κατά την προεπεξεργασία των ερωτημάτων αποτελεί για τον αλγόριθμο μαύρο κουτί, δηλαδή θεωρείται άγνωστη και δίνεται από το χρήστη.

2.7 Ομοιότητα ερωτημάτων

Σε περιβάλλοντα μηχανικής μάθησης, όπως θα δούμε παρακάτω, η έννοια της ομοιότητας των δεδομένων είναι θεμελιώδης. Όμως το τι σημαίνει ομοιότητα δεδομένων και πως αυτή μετριέται δεν είναι κάτι απόλυτο και μονοσήμαντα ορισμένο. Όπως θα αναλυθεί στην συνέχεια υπάρχουν πολλοί τρόποι να μετρήσει κανείς την ομοιότητα και το τι σημαίνει ποιοτικά η ομοιότητα εξαρτάται από την εφαρμογή και το είδος των δεδομένων. Για παράδειγμα, θα μπορούσε σε μια εφαρμογή να μας ενδιαφέρει η ομοιότητα ως προς τα χρώματα μιας εικόνας και σε μια άλλη η ομοιότητα ως προς το περιεχόμενο.

Αν αναλογιστούμε πως τα ερωτήματα χρηστών προς οποιαδήποτε πηγή είναι και αυτά με τη σειρά τους δεδομένα, τότε μπορούμε αναλόγως να μιλήσουμε και για ομοιότητα ερωτημάτων. Σαν αντικείμενο έρευνας η ομοιότητα ερωτημάτων δεν έχει ακόμα ερευνηθεί σημαντικά. Παρόλα αυτά υπάρχουν εργασίες που εμπλέκουν την ομοιότητα ερωτημάτων σε διάφορα ερευνητικά περιβάλλοντα. Οι περισσότερες από αυτές εξετάζουν την ομοιότητα των ρωτημάτων κυρίως από τη σημασιολογική οπτική τους.

Σημαντική συνεισφορά στην έρευνα γύρω από την ομοιότητα ερωτημάτων κάνει η εργασία [32]. Οι συγγραφείς της εργασίας αυτής ερευνούν την ομοιότητα ερωτημάτων στο πλαίσιο της βασισμένης στην συλλογικότητα αναζήτησης στο διαδίκτυο και εξετάζουν και συγκρίνουν δέκα διαφορετικούς τρόπους μέτρησης της ομοιότητας των ερωτημάτων. Σαν ομοιότητα ερωτημάτων ορίζουν το πόσο σημασιολογικά κοντά είναι το περιεχόμενο των διαφόρων ερωτημάτων, δηλαδή πόσο δύο ερωτήματα έχουν ως πρόθεση να ανακαλύψουν την ίδια πληροφορία. Οι μετρικές που εξετάζονται ακολουθούν διάφορες προσεγγίσεις. Μια προσέγγιση βασίζεται στους όρους που συμμετέχουν στα ερωτήματα και παρουσιάζει ως τρόπους μέτρησης της ομοιότητας την απόσταση επεξεργασίας και έναν συνδυασμό απόστασης επεξεργασίας και επικάλυψης όρων. Μια άλλη προσέγγιση, που αποτελεί και την πλέον δημοφιλή, προτείνει μετρικές που βασίζονται στα αποτελέσματα των ερωτημάτων. Συγκεκριμένα προτείνει την μέτρηση μέσω του υπολογισμού του ποσοστού της επικάλυψης των αποτελεσμάτων δύο ερωτημάτων, την μέτρηση μέσω της διαφοράς θέσης κοινών αποτελεσμάτων και την μέτρηση μέσω του πόσο διαφέρουν αυτές οι θέσεις για αντίστοιχα αποτελέσματα. Η τελευταία προσέγγιση της εργασίας είναι βασισμένη στις επιλογές των χρηστών. Ο πρώτος τρόπος που προτείνεται είναι το ποσοστό επικάλυψης των αποτελεσμάτων που

επιλέχθηκαν από το χρήστη για τα δύο ερωτήματα. Ο δεύτερος τρόπος που προτείνει αυτή η προσέγγιση μετρά την ομοιότητα συσχετίζοντας το πόσες φορές επιλέχθηκε το κάθε αποτέλεσμα από τον χρήστη που έθεσε το κάθε ερώτημα. Ο τρίτος και ο τέταρτος τρόπος μέτρησης αφορά, όπως και στην προηγούμενη προσέγγιση, τις διαφορές θέσης των επιλεγμένων αποτελεσμάτων. Συγκρίνοντας τις παραπάνω προσεγγίσεις και τους τρόπους μέτρησης η εργασία κατέληξε πως η προσέγγιση που βασίζεται στους όρους των ερωτημάτων είναι η πλέον αποτελεσματική.

Μια δεύτερη εργασία που χρησιμοποιεί την έννοια της ομοιότητας είναι η [33]. Η συγκεκριμένη εργασία μάλιστα πραγματεύεται την ομοιότητα ερωτημάτων στο πλαίσιο της συνειρμικής απάντησης ερωτημάτων, δηλαδή σε ένα πλαίσιο σχετικό με την παρούσα εργασία. Οι συγγραφείς της εργασίας [33] ορίζουν κάθε ερώτημα τους ως μια οντότητα που αποτελείται από τρία συστατικά: έναν θεματικό γράφο, μια εξαγόμενη λίστα γνωρισμάτων και ένα σύνολο συνθηκών επιλογής. Και σε αυτή την εργασία ο όρος ομοιότητα ερωτημάτων έχει την έννοια της σημασιολογικής απόστασης των ερωτημάτων όπως και στην προαναφερθείσα εργασία [32]. Η μέτρηση της ομοιότητας αυτής βασίζεται στα παραπάνω τρία συστατικά των ερωτημάτων. Η συνολική ομοιότητα δύο ερωτημάτων, λοιπόν, αποτελείται από το άθροισμα των ομοιοτήτων των επιμέρους συστατικών τους. Μάλιστα οι συγγραφείς διακρίνουν πως μπορεί να αποδοθεί διαφορετική βαρύτητα στην συνεισφορά της ομοιότητας των επιμέρους συστατικών στην συνολική ομοιότητα ανάλογα με την εφαρμογή, και ως εκ τούτου στο άθροισμα κάθε επιμέρους ομοιότητας σταθμίζεται από το αντίστοιχο βάρος. Αναφερόμενοι στις επιμέρους ομοιότητες, μετρούν την ομοιότητα του θέματος μέσω του λόγου του μεγέθους των κοινών θεματικών συνιστωσών προς το μέγεθος των συνολικών θεματικών συνιστωσών και των δύο ερωτημάτων. Αντίστοιχα για την ομοιότητα των εξαγόμενων λιστών γνωρισμάτων μετρούν το λόγο του αριθμού των γνωρισμάτων που περιέχονται στην εξαγόμενη λίστα και των δύο ερωτημάτων προς τον αριθμό των γνωρισμάτων του ενδιαφερόμενου ερωτήματος. Με αυτό τον τρόπο αντικατοπτρίζουν την εγγύτητα των πτυχών που ρωτιούνται στα δύο ερωτήματα. Τέλος μετρούν την ομοιότητα των συνθηκών επιλογής μέσω του γινομένου της ομοιότητας των αντιστοιχίσεων των γνωρισμάτων που συμμετέχουν στις συνθήκες των δύο γνωρισμάτων και της ομοιότητας των σταθερών όρων αυτών των συνθηκών.

Συνεχίζοντας, όπως ήδη αναφέραμε στην προηγούμενη υποενότητα, ο αλγόριθμος προσεγγιστικής μεταγραφής ερωτημάτων χρησιμοποιεί σαν μαύρο κουτί συναρτήσεις ομοιότητας ερωτημάτων. Παρότι στην συγκεκριμένη εργασία [1] αναφέρεται μονάχα μία συνάρτηση ομοιότητας και χωρίς ιδιαίτερη ανάλυση, το θέμα της ομοιότητας ερωτημάτων στο πλαίσιο της προσεγγιστικής μεταγραφής ερωτημάτων προσεγγίζεται σε μια δεύτερη εργασία [34], επέκταση της προηγούμενης. Στην εργασία [34] ορίζονται δύο μορφές ομοιότητας ερωτημάτων, η δομική ομοιότητα και η σημασιολογική. Η πρώτη ορίζεται ως η ομοιότητα των απαντήσεων όσον αφορά τον τρόπο που τα δεδομένα είναι δομημένα και η δεύτερη ως η ομοιότητα του περιεχομένου των απαντήσεων. Όπως αναφέρεται το δεύτερο είδος ομοιότητας είναι πολύ δύσκολο να μετρηθεί και στο πλαίσιο εφαρμογής της συγκεκριμένης εργασίας η λήψη των αποτελεσμάτων για την εύρεση του πλέον όμοιου με το αρχικό ερωτήματος δεν έχει νόημα αφού υφίσταται η ανάγκη να βρεθεί η ομοιότητα πριν γίνει η μεταγραφή και ληφθούν τα αποτελέσματα. Στην εργασία αναφέρεται επίσης πως ο ορισμός της δομικής ομοιότητας θα πρέπει να περιλαμβάνει τόσο έναν ποιοτικό τρόπο μέτρησης όσο και έναν ποσοτικό. Για το σκοπό αυτό:

- ως ερώτημα θεωρείται ένα σύνολο στοιχείων,

- ως ποιοτική μετρική ομοιότητας θεωρείται μια συνάρτηση η οποία δοθέντων δύο ερωτημάτων δείχνει αν ένα στοιχείο των δύο ερωτημάτων ικανοποιείται, δηλαδή αν ένα στοιχείο από οποιοδήποτε ερώτημα αναπαρίσταται σημασιολογικά από κάποιο στοιχείο του άλλου ερωτήματος,
- ως ποσοτική μετρική ομοιότητας θεωρείται μια συνάρτηση που ποσοτικοποιεί: πόσο σημαντική είναι η αναπαράσταση ενός στοιχείου του αρχικού ερωτήματος από ένα ή περισσότερα στοιχεία του προσεγγιστικού ερωτήματος, την σημασιολογική συσχέτιση των στοιχείων του αρχικού ερωτήματος και την διαφοροποίηση του προσεγγιστικού ερωτήματος από το από το αρχικό ως προς επιπλέον περιορισμούς που μπορεί να υπάρχουν στο πρώτο.

Συνολικά η εργασία προτείνει ότι μια συνάρτηση ομοιότητας ερωτημάτων θα πρέπει να διαθέτει τα εξής χαρακτηριστικά:

- να είναι μονότονη ως αφορά την αναπαράσταση των στοιχείων του αρχικού από στοιχεία στο προσεγγιστικό ερώτημα,
- να μειώνεται με την ύπαρξη επιπλέον στοιχείων στο προσεγγιστικό ερώτημα
- και να είναι αμφίδρομη, δηλαδή η ομοιότητα του αρχικού από το προσεγγιστικό να διαφέρει από αυτή του προσεγγιστικού από το αρχικό.

Επίσης η εργασία [34] προτείνει τρεις συναρτήσεις ομοιότητας ως παραδείγματα. Η πρώτη συνάρτηση ομοιότητας υποθέτει ότι δεν υπάρχουν συσχετίσεις των ομοιοτήτων μεμονωμένων στοιχείων και το εκφράζει μέσω ενός ολικού αθροίσματος των μεμονωμένων ομοιοτήτων. Η δεύτερη υποθέτει μια ισχυρή συσχέτιση μεταξύ των ομοιοτήτων των μεμονωμένων στοιχείων των ερωτημάτων και το εκφράζει μέσω ενός γινομένου του αθροίσματος των γνωρισμάτων επιλογής (select attributes) και του αθροίσματος των περιορισμών. Η τρίτη συνάρτηση ομοιότητας εστιάζει περισσότερο στη συσχέτιση των περιορισμών πάρα των υπολοίπων στοιχείων και το εκφράζει μέσω του γινομένου των ομοιοτήτων των περιορισμών αντί για το άθροισμα τους. Οι μαθηματικοί ορισμοί των συναρτήσεων όπως δίνονται στην εργασία [34] παρουσιάζονται στον παρακάτω πίνακα.

$M_{sim_1}(Q_{orig}, Q_{apprx}) = \frac{\sum w_i \cdot sat(E_i, E_i') - \sum w_j \cdot sat(E_j', E_j')}{\sum w_i \cdot sat(E_i, E_{i,})}$
$M_{sim_2}(Q_{orig}, Q_{apprx}) = \frac{(\sum w_i \cdot sat(S_i, S_i')) \cdot (\sum w_i \cdot sat(C_i, C_i')) - \sum w_j \cdot sat(C_j', C_j')}{\sum w_i \cdot sat(S_i, S_{i,}) \cdot \sum w_i \cdot sat(C_i, C_{i,})}$
$M_{sim_3}(Q_{orig}, Q_{apprx}) = \frac{(\sum w_i \cdot sat(S_i, S_i')) \cdot (\Pi w_i \cdot sat(C_i, C_i')) - \sum w_j \cdot sat(C_j', C_j')}{\sum w_i \cdot sat(S_i, S_{i,}) \cdot \Pi w_i \cdot sat(C_i, C_{i,})}$

Τέλος, σε όλες τις παραπάνω εργασίες για την εύρεση της ομοιότητας χρησιμοποιήθηκαν κλασσικές μέθοδοι και δεν έγινε χρήση τεχνικών μηχανικής μάθησης. Όπως όμως θα δούμε και παρακάτω τα συστήματα μηχανικής μάθησης χρησιμοποιούν εκ φύσεως την ομοιότητα και την απόσταση των δεδομένων στα οποία εφαρμόζονται. Συνεπώς αν τα ερωτήματα μπορέσουν να αναπαρασταθούν κατά τρόπο που να ορίζει την ομοιότητα των ερωτημάτων όπως είναι επιθυμητή αλλά και να είναι εύκολα διαχειρίσιμα από τα συστήματα μηχανικής μάθησης τότε η έννοια της ομοιότητας δεδομένων που είναι έμφυτη στη μηχανική μάθηση ταυτίζεται με την επιθυμητή ομοιότητα ερωτημάτων. Μια τέτοια χρήση της μηχανικής μάθησης είναι εμφανής στην εργασία [35], όπου οι συγγραφείς ομαδοποιούν τα δεδομένα με μηχανική μάθηση και κάνοντας χρήση της ανωτέρω ομοιότητας και δημιουργούν ένα σύστημα που βρίσκει ερωτήματα με την ίδια πρόθεση. Αντίστοιχα

στην εργασία [36], οι συγγραφείς χρησιμοποιούν μηχανική μάθηση για να δημιουργήσουν ένα σύστημα συστάσεων για το πλάνο εκτέλεσης ενός ερωτήματος που βασίζεται στον συνδυασμό δύο διαφορετικών εκφράσεων της ομοιότητας ερωτημάτων. Συγκεκριμένα συγκρίνουν τα ερωτήματα ως προς το περιεχόμενο αλλά κι ως προς τα πλάνα εκτέλεσης τους.

2.8 Μηχανική Μάθηση

Τα τελευταία χρόνια η ραγδαία αύξηση των δεδομένων που παράγονται καθημερινά έχει δώσει νέα ώθηση στον τομέα της Μηχανικής Μάθησης. Αν και σαν επιστημονικός τομέας, η Μηχανική Μάθηση εξερευνάται ήδη από τις αρχές της δεκαετίας του 1960, στις μέρες μας βρίσκει την μεγαλύτερη της ανάπτυξη.

Συγκεκριμένα, ο όρος Μηχανική Μάθηση εισήχθη στον επιστημονικό χώρο για πρώτη φορά το 1959 από τον Arthur Samuel στο άρθρο του «Some Studies in Machine Learning Using the Game of Checkers». Με βάση τον ορισμό «χωρίς να είναι ρητά προγραμματισμένο» που συχνά αποδίδεται στον Samuel, η Μηχανική Μάθηση μπορεί να οριστεί ως το επιστημονικό πεδίο που μελετά τους αλγορίθμους και τα στατιστικά μοντέλα που χρησιμοποιούνται από τα υπολογιστικά συστήματα για να εκτελέσουν συγκεκριμένες εργασίες χωρίς την χρήση ρητών εντολών, αλλά βασισμένα σε πρότυπα και συμπεράσματα.

Αργότερα ένας πιο φορμαλιστικός και μαθηματικός ορισμός των προβλημάτων Μηχανικής Μάθησης δόθηκε από τον Tom Mitchell. Συγκεκριμένα σύμφωνα με τον Mitchell [3]:

«Ένα πρόγραμμα υπολογιστή λέγεται ότι μαθαίνει από μια εμπειρία E σε σχέση με κάποια κλάση εργασιών T και μέτρηση απόδοσης P εάν η απόδοση του στις εργασίες της κλάσης T , όπως μετριέται από την P, βελτιώνεται με την εμπειρία E.»

Σήμερα, η Μηχανική Μάθηση αποτελεί πλέον έναν από τους πιο δημοφιλείς ερευνητικούς τομείς, οι εφαρμογές του οποίου εκτείνονται και σε διάφορους άλλους τομείς προσφέροντας λύσεις σε σημαντικά προβλήματα και αυτοματοποιώντας δύσκολες και χρονοβόρες διαδικασίες. Οι βιοεπιστήμες, η ρομποτική και η ανάλυση δεδομένων για επιχειρησιακούς σκοπούς αποτελούν παραδείγματα τομέων στους οποίους η Μηχανική Μάθηση βρίσκει μεγάλο πεδίο εφαρμογής.

2.9 Μηχανική Μάθηση στη Διαχείριση Δεδομένων

Όπως ήδη αναφέρθηκε, η αύξηση των ημερησίως παραγόμενων δεδομένων έδωσε νέα ώθηση στο πεδίο της Μηχανικής Μάθησης, καθώς σήμανε μεγαλύτερη διαθεσιμότητα δεδομένων για την εκπαίδευση των συστημάτων Μηχανικής Μάθησης. Δεν υπήρξε όμως ο μοναδικός λόγος. Καθοριστικός παράγοντας στην εκ νέου άνθηση αυτού του επιστημονικού πεδίου ήταν και τα επιτεύγματα και οι εξελίξεις στον τομέα της Διαχείρισης Δεδομένων.

Όπως αναφέρεται σε προηγούμενες εργασίες [4, 5], η Διαχείριση Δεδομένων έχει προσφέρει πολλές λύσεις σε δυσεπίλυτα προβλήματα της Μηχανικής Μάθησης, είτε αυτή εφαρμόζεται σε κάποια γραμμή παραγωγής είτε για ερευνητικούς σκοπούς. Η εργασία [4] συζητά λύσεις αλλά και υπάρχουσες προκλήσεις που συναντιόνται σε σημαντικές και χρονοβόρες διαδικασίες συστημάτων παραγωγής που χρησιμοποιούν Μηχανική Μάθηση. Συγκεκριμένα τονίζεται πως προβλήματα που αφορούν την κατανόηση, την εγκυρότητα, τον καθαρισμό αλλά και τον εμπλουτισμό των δεδομένων που χρησιμοποιούνται για την εκπαίδευση αυτών των συστημάτων βρίσκουν ή χρίζουν λύσεων στον τομέα της Διαχείρισης Δεδομένων.

Στην εργασία [5] παρουσιάζονται πολλές εργασίες και συστήματα που προσπαθούν να ενσωματώσουν σε υπάρχοντα συστήματα διαχείρισης δεδομένων αλγόριθμους μηχανικής μάθησης έχοντας ως στόχο την μείωση του χρόνου που δαπανάται για την ανάλυση δεδομένων αποφεύγοντας την μεταφορά δεδομένων μεταξύ συστημάτων. Συζητιούνται επίσης συστήματα κι εργασίες Μηχανικής Μάθησης που εμπνέονται από τεχνικές και συστήματα βάσεων δεδομένων καθώς και εργασίες που αφορούν προπαρασκευαστικές διαδικασίες που δεν αποτελούν μέρος των συστημάτων Μηχανικής Μάθησης αλλά παίζουν καθοριστικό ρόλο στην απόδοση αυτών και θα μπορούσαν να επωφεληθούν από λύσεις και εφαρμογές της Διαχείρισης Δεδομένων. Ιδιαίτερο ενδιαφέρον παρουσιάζουν οι εργασίες που αναφέρονται στην ανωτέρω εργασία και είναι εμπνευσμένες από το παραδοσιακό πρόβλημα βελτιστοποίησης ερωτημάτων στις Βάσεις Δεδομένων, οι οποίες χρησιμοποιούν μετεγγραφές απλοποίησης και συνεπώς σε γενικότερο πλαίσιο αντιμετωπίζουν ανάλογα προβλήματα με το πρόβλημα της παρούσας εργασίας.

Από τις αναφερθείσες εργασίες γίνεται ολοφάνερη η συμβολή της Διαχείρισης Δεδομένων στην Μηχανική Μάθηση. Όμως αυτή η σχέση μεταξύ των δύο πεδίων δεν είναι μονομερής. Η εργασία [6] αποτελεί σημείο αναφοράς για την διμερή σχέση των δύο επιστημονικών πεδίων καθώς παρουσιάζει την φυσική συνεργασία μεταξύ Μηχανικής Μάθησης και Ολοκλήρωσης Δεδομένων, ένα από τα βασικά θέματα έρευνας στη Διαχείριση Δεδομένων. Στην εργασία [6], λοιπόν, από τη μία μεριά εξετάζεται η συμβολή της Μηχανικής Μάθησης σε τελευταίας τεχνολογίας τεχνικές ολοκλήρωσης δεδομένων σε προβλήματα όπως η αντιστοίχιση οντοτήτων, η σύνθεση δεδομένων και η αντιστοίχιση σχημάτων, ενώ από την άλλη συνοψίζεται η εξάρτηση των τεχνικών Μηχανικής Μάθησης και των πολυεπίπεδων εφαρμογών αυτών από την εφαρμογή των τεχνικών ολοκλήρωσης δεδομένων στα δεδομένα που χρησιμοποιούνται από τα συστήματα Μηχανικής Μάθησης.

Στο ίδιο πλαίσιο, η εργασία [7] προσπαθεί να αναδείξει την αντίστροφη οπτική της σχέσης Μηχανικής Μάθησης – Διαχείρισης Δεδομένων και παρουσιάζει λύσεις και οφέλη της Μηχανικής Μάθησης στις διαδικασίες και στις τεχνικές που εφαρμόζονται στην Διαχείριση Δεδομένων. Στο πρώτο μέρος της εργασίας παρουσιάζονται τεχνικές Μηχανικής Μάθησης που είτε δίνουν λύσεις σε δυσεπίλυτα προβλήματα είτε παρουσιάζουν χρονικά αποδοτικότερες εναλλακτικές για τις υπάρχουσες διαδικασίες επεξεργασίας δεδομένων, όπως η συσχέτιση οντοτήτων, η επιδιόρθωση δεδομένων και ο συνδυασμός δεδομένων και γνώσης. Στο δεύτερο μέρος της εργασίας παρουσιάζονται τεχνικές Μηχανικής Μάθησης που είτε βρίσκουν εφαρμογή σε διαδικασίες ολοκλήρωσης δεδομένων, όπως η αντιστοίχιση σχημάτων και η ανακάλυψη περιορισμών, είτε επωφελούνται από αυτές, καθώς και σε διαδικασίες συμπερασμού ερωτημάτων, προσεγγιστικής επεξεργασίας ερωτημάτων και παραμετροποίησης συστημάτων Διαχείρισης Δεδομένων.

2.10 Μέθοδοι Μηχανικής Μάθησης

Στη μηχανική μάθηση υπάρχουν δύο κλασσικές μέθοδοι μάθησης, η επιβλεπόμενη και η μη επιβλεπόμενη μάθηση. Κατά την επιβλεπόμενη μάθηση, ο άνθρωπος παρέχει στο σύστημα μηχανικής μάθησης παραδείγματα των διαφόρων κλάσεων που εμπεριέχονται στα δεδομένα με σκοπό το σύστημα να καταφέρει να συμπεράνει την συνάρτηση μέσω της οποίας τα εισερχόμενα δεδομένα αντιστοιχίζονται στις κλάσεις στις οποίες ανήκουν. Κατά την μη επιβλεπόμενη μάθηση, δεν παρέχεται στο σύστημα κανένα παράδειγμα των κλάσεων και γενικότερα καμία πληροφορία σχετικά με τα δεδομένα και το σύστημα καλείται να αυτο-οργανωθεί ώστε να βρει στα δεδομένα που του προσφέρονται σχέσεις και μοτίβα τα οποία ήταν προηγουμένως άγνωστα.

Στην εποχή των Μεγάλων Δεδομένων, η ραγδαία αυτή αύξηση του όγκου των δεδομένων κάνει αναγκαία την προσαρμογή των συστημάτων μηχανικής μάθησης στις ανάγκες της νέας εποχής. Πλέον τα συστήματα καλούνται να εκπαιδευτούν και να μάθουν να επιτελούν τις εργασίες τους σε περισσότερα και μεγαλύτερου εύρους δεδομένα. Αυτό έχει σαν αποτέλεσμα στα συστήματα επιβλεπόμενης μάθησης να απαιτείται πλέον πολύ μεγαλύτερος όγκος παραδειγμάτων για την αποτελεσματική εκπαίδευση τους. Επιπλέον, τα δεδομένα αυτά λόγω και του μεγάλου όγκου τους είτε μπορεί να κρύβουν πτυχές άγνωστες προς τους ερευνητές είτε να εμπεριέχουν κάποιο θόρυβο. Είναι πολύ συχνή η περίπτωση στην οποία στο σύνολο των διαθέσιμων δεδομένων υπάρχουν πολύ λίγα ταξινομημένα δεδομένα, τα οποία μπορούν να χρησιμοποιηθούν ως παραδείγματα.

Για την επίλυση των ανωτέρω προβλημάτων μια άλλη μέθοδος μάθησης έχει αρχίσει να εφαρμόζεται. Αυτή η μέθοδος μάθησης αποτελεί συνδυασμό των δύο προαναφερθέντων μεθόδων και ονομάζεται ημι-επιβλεπόμενη μάθηση. Η μέθοδος της ημι-επιβλεπόμενης μάθησης έχει διάφορους τρόπους εφαρμογής και δίνει σημαντικές λύσεις σε διάφορα προβλήματα σχετιζόμενα με τα προαναφερθέντα. Μια μορφή ημι-επιβλεπόμενης μάθησης είναι η Απομακρυσμένη Επίβλεψη και η οποία όπως περιγράφτηκε για πρώτη φορά στην εργασία [8] χρησιμοποιεί μια Βάση Γνώσης και για κάθε ζεύγος μιας σχέσης από τη Βάση Γνώσης βρίσκει αναφορές σε μια μεγάλη μη ταξινομημένη συλλογή κειμένων και εξάγει χαρακτηριστικά κειμένου για να εκπαιδεύσει έναν ταξινομητή. Μια γενικότερη διατύπωση της συγκεκριμένης μεθόδου θα ήταν η χρήση μιας Βάσης γνώσης για την εξαγωγή σχέσεων και κατηγοριών σε μη ταξινομημένα εκ των προτέρων δεδομένα. Η μέθοδος της Απομακρυσμένης Επίβλεψης έχει ευρεία χρήση σε εργασίες όπως οι [9], [10], [11].

Η ημι-επιβλεπόμενη μάθηση μπορεί να εκφραστεί και με άλλους τρόπους εκτός της Απομακρυσμένης Επίβλεψης. Στην εργασία [12] χρησιμοποιούνται τεχνικές μη επιβλεπόμενης μηχανικής μάθησης για την ταξινόμηση των αρχικών δεδομένων και την απόδοση ετικετών στις παραγόμενες κατηγορίες. Συγκεκριμένα, χρησιμοποιείται αλγόριθμος ιεραρχικής ομαδοποίησης για την ταξινόμηση σε κατηγορίες των ανεπεξέργαστων δεδομένων και για την απόδοση ετικετών στις κατηγορίες με τη χρήση πληροφοριών κλειδιά. Μετά την απόδοση των ετικετών, τα κατηγοριοποιημένα δεδομένα μπορούν να χρησιμοποιηθούν σε οποιοδήποτε σύστημα επιβλεπόμενης μάθησης ως παραδείγματα για την εκπαίδευση και τον έλεγχο της απόδοσης του συστήματος.

Εκτενή μελέτη της μεθόδου ημι-επιβλεπόμενης μάθησης γίνεται στην εργασία [13]. Οι συγγραφείς παρουσιάζουν και συγκρίνουν οχτώ αλγορίθμους ημι-επιβλεπόμενης μάθησης και προτείνουν μια νέα μέθοδο που συνδυάζει τα οφέλη μη επιβλεπόμενης και επιβλεπόμενης μάθησης

για την χρήση και των δεδομένων χωρίς ετικέτες και των δεδομένων με διαθέσιμες ετικέτες. Πιο αναλυτικά, αρχικά εφαρμόζουν στο σύνολο των δεδομένων τους (με ετικέτες ή χωρίς) διαφορετικούς αλγόριθμους ομαδοποίησης μη επιβλεπόμενης μάθησης δημιουργώντας έτσι πολλές διαφορετικές διαμερίσεις των δεδομένων. Στη συνέχεια επαυξάνουν τα χαρακτηριστικά των δεδομένων τους προσθέτοντας στον πίνακα των αρχικών χαρακτηριστικών για κάθε παράδειγμα την κατηγορία στην οποία ανήκουν σε κάθε διαμέριση. Τέλος εκπαιδεύουν έναν ταξινομητή πάνω στα επαυξημένα παραδείγματα που είχαν εξ' αρχής και εν συνεχεία κάνουν χρήση αυτού για να ταξινομήσουν τα επαυξημένα παραδείγματα που δεν είχαν αρχικά ετικέτες.

Μια άλλη προσέγγιση της ημι-επιβλεπόμενης μάθησης εφαρμόζεται στην εργασία [14]. Στην εργασία αυτή, οι συγγραφείς εφαρμόζουν μη επιβλεπόμενη μάθηση για να χωρίσουν τα δεδομένα σε ομάδες καταγράφοντας έτσι την κατανομή των κλάσεων των δεδομένων, εν συνεχεία χρησιμοποιούν έναν ελάχιστο αριθμό από κατηγοριοποιημένα παραδείγματα για να αποδώσουν κατηγορίες στις προηγουμένως κατασκευασμένες ομάδες και τέλος να χρησιμοποιήσουν αυτό το επαυξημένο σύνολο ταξινομημένων παραδειγμάτων για την εκπαίδευση ενός ταξινομητή σε μια διαδικασία επιβλεπόμενης μάθησης.

2.11 Αλγόριθμοι Ομαδοποίησης (Clustering)

Σύμφωνα με τα μαθήματα της Google πάνω στη Μηχανική Μάθηση που μπορεί να βρει κανείς διαδικτυακά [28], η ομαδοποίηση (clustering) είναι μια απόλυτα φυσική διαδικασία που και εμείς οι άνθρωποι χρησιμοποιούμε σαν στρατηγική για διάφορες εργασίες. Όπως για παράδειγμα αναφέρεται, όταν κάποιος θέλει να μάθει κάτι καινούργιο όπως τραγούδια και μουσική μια στρατηγική που μπορεί να εφαρμόσει είναι να χωρίσει τα τραγούδια που έχει στη διάθεση του σε ομάδες με κάποια σημασία. Αυτό μπορεί να σημαίνει δημιουργία ομάδων κατά είδος, κατά χρονολογία ή οποιαδήποτε άλλη πληροφορία θεωρείται σημαντική για το άτομο. Διαφορετικές ομαδοποιήσεις θα μας οδηγήσουν στο να μάθουμε κάποια διαφορετική πληροφορία για τα τραγούδια. Όποια κι αν είναι όμως η ομαδοποίηση που θα πραγματοποιηθεί θα έχουμε μάθει κάτι καινούργιο. Έτσι και για την μηχανική μάθηση η τεχνική της ομαδοποίησης αποσκοπεί στην άντληση κάποιας υποφαινόμενης πληροφορίας και στην κατανόηση ενός θέματος ή/και των δεδομένων. Μάλιστα βρίσκει πολύ μεγάλη εφαρμογή σε πολλούς και διάφορους τομείς, όπως στη διαμέριση της αγοράς, στην ανάλυση των κοινωνικών δικτύων κ.α.

Εάν θέλαμε να δώσουμε έναν κλασσικό ορισμό για την ομαδοποίηση, θα μπορούσαμε να την περιγράψουμε όπως στην εργασία [29]:

- Παραδείγματα, στην ίδια ομάδα, πρέπει να μοιάζουν όσο το δυνατό περισσότερο.
- Παραδείγματα, σε διαφορετικές ομάδες, πρέπει να διαφέρουν όσο το δυνατόν περισσότερο.
- Η μέτρηση της ομοιότητας και της διαφορετικότητας πρέπει να είναι ξεκάθαρη και να έχει πρακτικό νόημα.

Στην ομαδοποίηση η χρήση της ομοιότητας ή της διαφορετικότητας, ή αλλιώς απόστασης, εξαρτάται από το είδος των δεδομένων. Όταν έχουμε ποσοτικά δεδομένα προτιμάται η απόσταση ενώ όταν τα δεδομένα μας είναι ποιοτικά προτιμάται η ομοιότητα. Υπάρχουν πολλές και διάφορες συναρτήσεις απόστασης ή ομοιότητας και η χρήση τους εξαρτάται τις περισσότερες φορές από το σκοπό και την εφαρμογή.

Για την πραγματοποίηση της ομαδοποίησης υπάρχουν διάφοροι αλγόριθμοι. Ο πλέον διαδεδομένος και γνωστός είναι ο αλγόριθμος K-μέσων (K-means) [30]. Ο αλγόριθμος αυτός αποθηκεύει σημεία που ονομάζονται μέσοι (centroids) τους οποίους χρησιμοποιεί για να ορίσει τις διάφορες ομάδες. Για να είναι κάποιο παράδειγμα από τα δεδομένα σε μία ομάδα θα πρέπει να βρίσκεται πιο κοντά στον μέσο της συγκεκριμένης ομάδας από οποιονδήποτε μέσο άλλης ομάδας. Ο αλγόριθμος K-μέσων απαιτεί να γνωρίζει τον αριθμό των ομάδων που καλείται να διαχωρίσει. Δοθέντος αυτού του αριθμού, ο αλγόριθμος αρχικοποιεί τυχαία τόσους μέσους όσος ο αριθμός των ομάδων και υπολογίζει τις αποστάσεις όλων των παραδειγμάτων που έχουμε στα δεδομένα μας από όλους τους μέσους. Κάθε παράδειγμα αντιστοιχίζεται στην ομάδα του πιο κοντινού μέσου. Αφού χωριστούν τα παραδείγματα στις νέες ομάδες με βάσεις τις αποστάσεις τους από τους μέσους, ο μέσος κάθε ομάδας επαναπροσδιορίζεται και μεταφέρεται στο κέντρο της ομάδας. Η διαδικασία αυτή επαναλαμβάνεται έως ότου υπάρξει σύγκλιση, δηλαδή δεν υπάρχει (σημαντική) διαφοροποίηση στις ομάδες και στις θέσεις των μέσων. ΟΙ τελικοί αυτοί μέσοι είναι αυτοί που αποθηκεύονται.

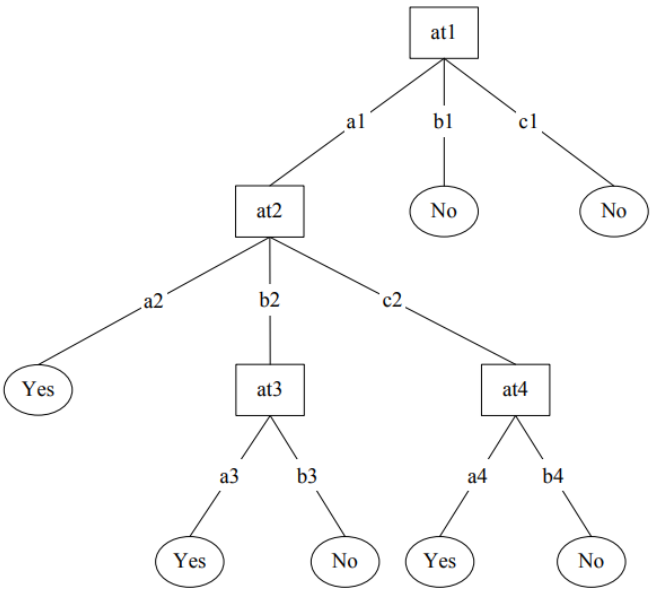
Μια άλλη εξίσου δημοφιλής ομάδα αλγορίθμων ομαδοποίησης είναι οι ιεραρχικοί (hierarchical). Αντίθετα με τον αλγόριθμο K-μέσων, για τους ιεραρχικούς αλγόριθμους [31] η γνώση του αριθμού των ομάδων είναι προαιρετική και η τελική ομαδοποίηση μπορεί να πραγματοποιηθεί μέσω ενός κατωφλίου της απόστασης των ομάδων. Υπάρχουν πολλές υλοποιήσεις ιεραρχικών αλγορίθμων αλλά όλες έχουν μια εκ των δύο προσεγγίσεων, την από πάνω προς τα κάτω (top-down) ή την από κάτω προς τα πάνω (bottom-up). ΟΙ ιεραρχικοί αλγόριθμοι με την top-down προσέγγιση ονομάζονται διαχωριστικοί (divisive) και ξεκινούν έχοντας όλα τα δεδομένα σε μία ομάδα και με βάσεις τις μεταξύ τους αποστάσεις σε κάθε βήμα διαχωρίζουν τις ομάδες σε δύο υπο-ομάδες. Τερματίζουν όταν όλα τα παραδείγματα βρεθούν σε δική τους ομάδα. Οι bottom-up ιεραρχικοί αλγόριθμοι ονομάζονται συσσωρευτικοί (agglomerative) και ξεκινούν έχοντας κάθε παράδειγμα σε διαφορετική ομάδα και ενώνοντας σε κάθε βήμα ομάδες ανά δύο με βάσει την απόσταση τους τερματίζουν όταν απομείνει μόνο μία ομάδα. Δίνοντας κάποιο κατώφλι απόστασης μπορεί κανείς μέσω δενδρογράμματος να λάβει τον τελικό αριθμό ομάδων και ποια παραδείγματα συμμετέχουν σε αυτές (κυρίως ποιοτικά και όχι με ακρίβεια). Ξανατρέχοντας τον αλγόριθμο και δίνοντας των αριθμό των ομάδων που έχουν βρεθεί μπορεί κανείς να πάρει τις ετικέτες συμμετοχής στην ομάδα για κάθε παράδειγμα.

Αν και ο αλγόριθμος K-μέσων είναι πιο αποδοτικός και ανθεκτικός στην ποιότητα των δεδομένων η απαίτηση του να είναι γνωστός ο αριθμός των ομάδων είναι πολλές φορές ένα σοβαρό μειονέκτημα. Αντίθετα οι ιεραρχικοί αλγόριθμοι άρουν αυτή την απαίτηση και δίνουν μια καλύτερη εικόνα για την επιλογή του τελικού αριθμού των ομάδων.

2.12 Αλγόριθμοι Ταξινόμησης (Classification)

Η ταξινόμηση παραδειγμάτων εμπίπτει στον τομέα της επιβλεπόμενης μηχανικής μάθησης. Ως λειτουργία έχει ως στόχο την απόδοση σε νέα εισερχόμενα δεδομένα της κατάλληλης ετικέτας αφού πρώτα το σύστημα έχει εκπαιδευτεί με παραδείγματα με ετικέτες. Υπάρχουν πολλοί και διάφοροι αλγόριθμοι που επιτελούν την λειτουργία της ταξινόμησης και η επιλογή ποιος θα χρησιμοποιηθεί εξαρτάται τις περισσότερες φορές από την εφαρμογή και την πειραματική αξιολόγηση της απόδοσης τους στα δεδομένα ενδιαφέροντος. Η εργασία [37] κάνει μια ενδιαφέρουσα σύνοψη για τους υπάρχοντες αλγόριθμους ταξινόμηση.

Ένας από τους πλέον γνωστούς αλγόριθμους ταξινόμησης που ανήκει στην κατηγορία των αλγορίθμων που βασίζονται στην λογική είναι ο αλγόριθμος δέντρων αποφάσεων (Decision trees). Τα δέντρα αποφάσεων είναι δέντρα τα οποία ταξινομούν τα παραδείγματα με βάση τις τιμές κάθε χαρακτηριστικού τους. Κάθε κόμβος του δέντρου αναπαριστά ένα χαρακτηριστικό και κάθε κλαδί μια τιμή που ο κόμβος μπορεί να πάρει. Τα παραδείγματα ταξινομούνται ξεκινώντας από τη ρίζα και με βάση τις τιμές των επιμέρους χαρακτηριστικών.



Σχήμα 2.3: Δέντρο αποφάσεων

Ένας άλλος αλγόριθμος που ανήκει στην κατηγορία των στατιστικών αλγορίθμων μάθησης/ταξινόμησης είναι ο αλγόριθμος γραμμικής ανάλυσης διακρίσεων. Γενικά, οι στατιστικοί αλγόριθμοι χαρακτηρίζονται από την παρουσία ενός υποκείμενου πιθανοτικού μοντέλου που δίνει την πιθανότητα ένα παράδειγμα να ανήκει σε μια κατηγορία και δεν εκτελούν απλά μια ταξινόμηση. Ο αλγόριθμος γραμμικής ανάλυσης διακρίσεων, εν προκειμένου, προσπαθεί να βρει το γραμμικό συνδυασμό των χαρακτηριστικών που διαχωρίζει με τον καλύτερο τρόπο δύο ή περισσότερες κλάσεις των δεδομένων.

Στην κατηγορία των στατιστικών αλγορίθμων ανήκει και ο αλγόριθμος Naive Bayes. Ο αλγόριθμος αυτός υποθέτει ότι τα χαρακτηριστικά των παραδειγμάτων είναι ανεξάρτητα και βασίζεται στην επιλογή της κατάλληλης κλάσης μέσω του υπολογισμού της πιθανότητας κάθε κλάσης με τον τύπο του Bayes:

$$\hat{y} = \underset{k\in\{1,\dots,K\}}{arg\max}\, p(C_k) \prod_{i=1}^{n} p(x_i|C_k).$$

Πολύ δημοφιλής αλγόριθμος ταξινόμησης είναι επίσης ο αλγόριθμος των κ-Κοντινότερων γειτόνων (k-Nearest Neighbours), ο οποίος ανήκει στην κατηγορία των αλγορίθμων ταξινόμησης βασισμένων σε παραδείγματα. Η βασική ιδέα του αλγόριθμου αυτού είναι ότι σε ένα σύνολο δεδομένων τα παραδείγματα με παρόμοιες ιδιότητες θα βρίσκονται κοντά μεταξύ τους. Έτσι για κάθε νέο μη ταξινομημένο παράδειγμα που δίνεται στον αλγόριθμο, αυτός προσπαθεί να το ταξινομήσει με βάση τις ετικέτες των παραδειγμάτων εκπαίδευσης και πόσο το υπό ταξινόμηση παράδειγμα απέχει από αυτά. Συγκεκριμένα ο αλγόριθμος βρίσκει τα κ κοντινότερα παραδείγματα εκπαίδευσης και συλλέγει τις ετικέτες τους. Η ετικέτα με τις περισσότερες εμφανίσεις είναι κι αυτή του τελικά αποδίδεται στο ζητούμενο παράδειγμα. Για την εύρεση των κοντινότερων γειτόνων χρησιμοποιείται η έννοια της απόστασης που αναφέρθηκε και στους αλγόριθμους ομαδοποίησης και μπορεί να έχει διάφορες εκφράσεις, π.χ. η ευκλίδεια απόσταση ή η απόσταση minkowski να είναι από τις πιο δημοφιλείς. Όπως γίνεται εύκολα κατανοητό ο αριθμός κ των ζητούμενων γειτόνων δίνεται από το χρήστη και επηρεάζει σημαντικά την απόδοση του αλγόριθμου.

Μια ακόμα τεχνική ταξινόμησης είναι οι μηχανές διανυσμάτων υποστήριξης. Είναι ιδανικές για γραμμικά διαχωρίσιμα προβλήματα καθώς μειώνουν το ανώτατο όριο εκτιμώμενου λάθους γενίκευσης. Αν τα δεδομένα εκπαίδευσης είναι γραμμικά διαχωρίσιμα τότε υπάρχει ένα ζεύγος παραμέτρων ($\boldsymbol{w}$, b) τέτοιο ώστε:

$$\boldsymbol{w}^T\boldsymbol{x_i} + b \geq 1\,, \forall \boldsymbol{x_i} \in P$$
$$\boldsymbol{w}^T\boldsymbol{x_i} + b \leq 1\,, \forall \boldsymbol{x_i} \in N$$

με την απόφαση να δίνεται από τη συνάρτηση:

$$f_{\boldsymbol{w},b}(\boldsymbol{x}) = sgn(\boldsymbol{w}^T\boldsymbol{x_i} + b)$$

όπου το $\boldsymbol{w}$ είναι το διάνυσμα βαρών και το b είναι το κατώφλι της απόστασης.

3. Σχεδιασμός και Υλοποίηση

Όπως έχουμε ήδη αναφέρει, η παρούσα εργασία διερευνά την ομοιότητα ερωτημάτων μέσω μηχανικής μάθησης στο πλαίσιο της προσεγγιστικής μεταγραφής ερωτημάτων. Συγκεκριμένα στόχος είναι να ερευνήσουμε την ομοιότητα ερωτημάτων στο πλαίσιο της εργασίας [1]. Εμπνεόμενοι από την ανάγκη για αποδοτική μεταγραφή ερωτημάτων, όπως παρουσιάστηκε την παράγραφο 1.1, αλλά και από την υπάρχουσα βιβλιογραφία, προσπαθούμε να προσεγγίσουμε την έννοια της ομοιότητας ερωτημάτων από μια διαφορετική πλευρά με σκοπό να αυτοματοποιήσουμε και να βελτιστοποιήσουμε το σύστημα της εργασίας [1]. Η πορεία της ερευνητικής διαδικασίας που ακολουθήθηκε αναλύεται στα επόμενα τέσσερα στάδια.

3.1 Πρώτο Στάδιο: Σχεδιασμός, Υποθέσεις, Ορισμοί

Στο στάδιο αυτό έγινε ο πλήρης σχεδιασμός της ακόλουθης ερευνητικής πορείας, τέθηκαν οι στόχοι και οι υποθέσεις προς εξέταση, δόθηκαν οι ορισμοί των εννοιών που μας απασχόλησαν καθ' όλη τη διάρκεια της υλοποίησης και επιλέχθηκαν και μορφοποιήθηκαν τα πειραματικά δεδομένα. Αποτέλεσε ίσως το καθοριστικότερο στάδιο αφού οι αποφάσεις που λάβαμε σε αυτό επηρέασαν όλη τη διαδικασία. Σαν πρώτο βήμα τέθηκαν οι στόχοι και αναλύθηκαν οι ιδέες της παρούσας εργασίας. Στη συνέχεια διαμορφώσαμε τις ιδέες αυτές σε υποθέσεις και ορίσαμε τις απαραίτητες έννοιες, ενώ στο τέλος επιλέξαμε τα πειραματικά δεδομένα μας και τα επεξεργαστήκαμε ώστε να μπορούμε να τα χειριστούμε αποδοτικά στα πειράματά μας.

3.1.1 Στόχοι και Ιδέες

Όπως είδαμε στο προηγούμενο κεφάλαιο, η εργασία [1] κάνει χρήση συναρτήσεων ομοιότητας ερωτημάτων για να επιλέξει το καλύτερο υποερώτημα του δοθέντος ερωτήματος που μπορεί να μεταγραφεί. Όπως τονίζεται στην εργασία η συνάρτηση ομοιότητας που χρησιμοποιείται αποτελεί μαύρο κουτί για το σύστημα, δηλαδή το σύστημα αγνοεί ποια είναι αυτή, πως υλοποιείται και τη ακριβώς μετρά. Συνεπώς η επιλογή της κατάλληλης συνάρτησης αφήνεται στην ευχέρεια του χρήστη του συστήματος. Εάν ο χρήστης όμως δεν έχει το κατάλληλο υπόβαθρο, δεν είναι σε θέση να επιλέξει την πιο αποδοτική συνάρτηση για το ερώτημά του.

Επιπροσθέτως, στην εργασία [34], που αποτελεί μια συνέχεια της εργασίας [1] και προσπαθεί να ορίσει την ομοιότητα ερωτημάτων που χρησιμοποιεί η εργασία [1], παρουσιάζονται τρείς υποψήφιες για επιλογή από το χρήστη συναρτήσεις ομοιότητας για τον αλγόριθμο μεταγραφής. Κάθε μια από αυτές τις συναρτήσεις μετρά διαφορετικά την ομοιότητα δίνοντας έμφαση σε διαφορετικές πτυχές των ερωτημάτων. Όπως φαίνεται μάλιστα από τα αποτελέσματα της εργασίας [34] κάθε συνάρτηση δίνει διαφορετικά σε ποιότητα αποτελέσματα ανάλογα με τις άλλες παραμέτρους του αλγόριθμου και το πως είναι διαμορφωμένο το ερώτημα.

Ταυτόχρονα, όπως ήδη αναφέρθηκε στην παράγραφο 2.5, ερωτήματα που ανήκουν σε διαφορετικές κατηγορίες με βάση το γράφο τους παρουσιάζουν διαφορές στον τρόπο με τον οποίο βελτιστοποιούνται. Μάλιστα στις περισσότερες από τις εργασίες που εξετάζουν διαφόρων ειδών βελτιστοποιήσεις ερωτημάτων ή ασχολούνται με την μεταγραφή ή την απάντηση ερωτημάτων, συνήθως εξετάζονται οι δύο μεγαλύτερες ομάδες ερωτημάτων που εμφανίζονται συχνότερα σε συστήματα διαχείρισης δεδομένων, τα ερωτήματα τύπου star join και chain join, καθώς αυτά παρουσιάζουν το μεγαλύτερο ενδιαφέρον αλλά και διαφορές στον τρόπο με τον οποίο αντιμετωπίζονται.

Λαμβάνοντας τα παραπάνω υπόψιν, είναι εύλογο να αναρωτηθεί κανείς εάν διαφορετικού τύπου ερωτήματα θα μεταγράφονταν καλύτερα με κάποια συγκεκριμένη συνάρτηση από τις τρεις διαθέσιμες της εργασίας [34]. Προσπαθώντας να δώσει μια απάντηση στο ερώτημα αυτό, η κύρια ιδέα της παρούσας εργασίας είναι η αυτοματοποίηση της διαδικασίας επιλογής της συνάρτησης ομοιότητας που χρησιμοποιείται από τον αλγόριθμο μεταγραφής της εργασίας [1]. Η αυτόματη επιλογή θα έχει ως κριτήριο την κατηγορία στην οποία ανήκει το ερώτημα και τα δομικά του χαρακτηριστικά.

Μάλιστα η ιδέα πίσω από την κατηγοριοποίηση των ερωτημάτων είναι να ερευνηθούν και άλλα χαρακτηριστικά εκτός του τύπου σύνδεσης των σχέσεων που συμμετέχουν στο ερώτημα. Θεωρούμε ότι χαρακτηριστικά όπως το πλήθος των γνωρισμάτων επιλογής (select attributes) ή ο αριθμός των περιορισμών (where clauses) μπορούν να επηρεάσουν σημαντικά την ποιότητα της μεταγραφής και συνεπώς είναι σημαντικό να ερευνηθεί και η επιρροή τους στην ομοιότητα των ερωτημάτων αφού μέσω αυτής επιλέγεται και το ερώτημα που εν τέλει θα μεταφραστεί.

Συνοψίζοντας, οι βασικοί στόχοι και οι ιδέες της εργασίας θα μπορούσαν να αναλυθούν ως εξής:

- Αρχικός στόχος της εργασίας είναι η προσπάθεια κατηγοριοποίησης των ερωτημάτων σε κατηγορίες με βάση τα δομικά τους χαρακτηριστικά.
- Στη συνέχεια, ένας δεύτερος στόχος της εργασίας είναι η διαμόρφωση ενός συστήματος που θα μπορεί να αναγνωρίζει επαρκώς την κατηγορία ενός εισερχόμενου ερωτήματος
- Τελικός στόχος είναι η εφαρμογή αυτών των κατηγοριών για την εύρεση της καλύτερης συνάρτησης ομοιότητας για κάθε κατηγορία και στη συνέχεια την αυτόματη επιλογή της συνάρτησης ομοιότητας για κάθε εισερχόμενο ερώτημα.

Για να επιτευχθούν οι παραπάνω στόχοι, λήφθηκε η απόφαση να χρησιμοποιηθεί σαν βασικό εργαλείο η μηχανική μάθηση και οι διάφορες τεχνικές της ακολουθώντας έτσι και την επικρατούσα επιστημονική τάση, όπως αναλύθηκε στην παράγραφο 2.9.

3.1.2 Υποθέσεις και Ορισμοί

Για να προχωρήσουμε στην υλοποίηση των παραπάνω στόχων και ιδεών είναι σημαντικό να αναλύσουμε περισσότερο τις ιδέες ώστε να διαμορφώσουμε τις υποθέσεις πάνω στις οποίες

στηρίχτηκε ο σχεδιασμός και η υλοποίηση της πειραματικής διαδικασίας για την υλοποίηση των στόχων.

Η βασικότερη ιδέα της εργασίας είναι ότι διαφορετικά είδη ερωτημάτων για διαφορετικές συναρτήσεις ομοιότητας έχουν διαφορετικά αποτελέσματα κατά την μεταγραφή τους με το σύστημα της εργασίας [1]. Η διαφορά των αποτελεσμάτων έγκειται στην ποιότητα τους. Η ιδέα αυτή πηγάζει από τη διαφορετική συμπεριφορά που επιδεικνύουν διαφορετικά είδη ερωτημάτων σε άλλα προβλήματα όπως η βελτιστοποίηση ερωτημάτων. Αποτελεί την βασικότερη ιδέα πάνω στην οποία υλοποιείται η παρούσα εργασία και θα μπορούσε να εκφραστεί με τις εξής υποθέσεις:

Υπόθεση 1

Ερωτήματα με διαφορετικά δομικά χαρακτηριστικά μεταγράφονται προσεγγιστικά με τον καλύτερο δυνατό τρόπο με τη χρήση διαφορετικής συνάρτησης ομοιότητας ερωτημάτων.

Υπόθεση 2

Ερωτήματα με κοινά ή παραπλήσια δομικά χαρακτηριστικά μεταγράφονται προσεγγιστικά με τον καλύτερο δυνατό τρόπο με τη χρήση της ίδιας συνάρτησης ομοιότητας ερωτημάτων.

Όπως γίνεται αντιληπτό από τη διατύπωση των παραπάνω υποθέσεων όταν αναφερόμαστε σε είδη ή κατηγορίες ερωτημάτων, αναφερόμαστε σε ομάδες ερωτημάτων με κοινά η παραπλήσια χαρακτηριστικά. Τι εννοούμε όμως όταν λέμε κοινά ή παραπλήσια χαρακτηριστικά;

Στην παράγραφο 2.7 του προηγούμενου κεφαλαίου συζητήσαμε περί της ομοιότητας ερωτημάτων και πως αυτή παρουσιάζεται και μετριέται στη σχετική βιβλιογραφία. Όσες εργασίες σχετικές με την ομοιότητα ερωτημάτων εξετάστηκαν προσεγγίζουν την ομοιότητα από τη σημασιολογική της πλευρά. Ακόμα και οι εργασίες που μιλούν για δομική ομοιότητα, όπως η εργασία [34], στην ουσία δεν μιλούν για τα δομικά χαρακτηριστικά των ερωτημάτων αλλά για την ύπαρξη στο ένα ερώτημα στοιχείων που αντιπροσωπεύουν σημασιολογικά στοιχεία του άλλου ερωτήματος. Στην περίπτωση της μεταγραφής ερωτημάτων του αρχικού ερωτήματος. Στην παρούσα εργασία όμως αυτού του είδους η ομοιότητα δεν είναι αρκετή ούτε εξυπηρετεί την ιδέα για την κατηγοριοποίηση των ερωτημάτων που τέθηκε ως στόχος.

Για το λόγο αυτό, επιχειρούμε να δώσουμε έναν νέο ορισμό της δομικής ομοιότητας ερωτημάτων εστιάζοντας πραγματικά στη δομή των ερωτημάτων και όχι στην σημασιολογία και το περιεχόμενο τους. Το ενδιαφέρον σε αυτή τη μορφή ομοιότητας επικεντρώνεται σε στοιχεία όπως ο αριθμός συνδέσεων, ο τύπος των συνδέσεων αυτών, ο αριθμός των σταθερών περιορισμών, ο αριθμός των γνωρισμάτων επιλογής κ.α. Προφανώς και τα δομικά στοιχεία που εξετάζονται για την δομική ομοιότητα ερωτημάτων σχετίζονται άμεσα με το πλαίσιο εφαρμογής της και τις ανάγκες που αυτή εξυπηρετεί. Παρόλα αυτά ακριβώς επειδή τα ερωτήματα εξετάζονται με τρόπο ανεξάρτητο του περιεχομένου θεωρούμε ότι μπορεί να αποδώσει μια πιο γενική ομοιότητα ερωτημάτων και να συμβάλει στη δημιουργία ενός συστήματος κατηγοριοποίησης των ερωτημάτων που δεν θα εξαρτάται από τη σημασιολογία των βάσεων δεδομένων πάνω στις οποίες χρησιμοποιείται. Στην περίπτωση μας μια τέτοια ομοιότητα εξασφαλίζει ότι εκπαιδευμένο σύστημα μεταγραφής μπορεί να χρησιμοποιηθεί αποδοτικά σε πολλά διαφορετικά πλαίσια θεματολογίας.

Πριν προχωρήσουμε στην απόδοση του ορισμού της ομοιότητας που μας απασχολεί στην παρούσα εργασία είναι αναγκαίο να ορίσουμε μια επαρκή για το πρόβλημα μας αναπαράσταση ενός ερωτήματος. Τα ερωτήματα των χρηστών συνήθως εκφράζονται σε SQL ερωτήματα. Με αυτή τη μορφή άλλωστε τα ερωτήματα επεξεργάζονται από τα σχεσιακά συστήματα διαχείρισης δεδομένων. Από την άλλη οι LAV αντιστοιχήσεις που χρησιμοποιούνται από τον αλγόριθμο μεταγραφής αλλά και τα ερωτήματα που μεταγράφονται από το αλγόριθμο μέσω του αλγορίθμου Minicon απαιτείται να είναι σε συζευκτική μορφή (CQF). Στην παρούσα εργασία καμία από τις παραπάνω μορφές δεν μπορούν να αποτελέσουν τον τρόπο αναπαράστασης των ερωτημάτων μας αφού η απαιτούμενη πληροφορία ναι μεν εμπεριέχεται και μπορεί να εξαχθεί από αυτά αλλά δεν παρέχεται σε μορφή διαχειρίσιμη.

Παρόλα αυτά, καθώς στο πλαίσιο αναφοράς της παρούσας εργασίας χρησιμοποιούνται και οι δύο παραπάνω μορφές των ερωτημάτων, θα διατηρήσουμε τους ορισμούς για αυτές όπως έχουν δοθεί στην εργασία [1]. Έτσι:

Ορισμός 1

Ένα SQL ερώτημα ορίζεται ως ένα σύνολο στοιχείων, ένα για κάθε γνώρισμα επιλογής (select attribute) και ένα για κάθε περιορισμό/συνθήκη (where condition) και συμβολίζεται ως Q^{SQL}.

Ορισμός 2

Ένα συζευκτικό ερώτημα είναι της μορφής $Q(X) :- R_1(X_1)\, R_2(X_2) \dots R_n(X_n)$ *όπου* Q *και* $R_i, i = 1, \dots, n$ *είναι ονόματα κατηγορημάτων και* $X_i, i = 1, \dots, n$ *είναι πλειάδες, δηλαδή κάθε ένα είναι ένα διατεταγμένο σύνολο από μεταβλητές γνωρισμάτων και σταθερές. Συμβολίζεται ως* Q^{CQF}

Για τους σκοπούς αυτής της εργασίας ένα ερώτημα ορίζεται ως εξής:

Ορισμός 3

Ορίζουμε ως ερώτημα χρήστη Q^V *ένα διάνυσμα* v *με έξι διαστάσεις. Κάθε διάσταση του διανύσματος αντιπροσωπεύει ένα δομικό χαρακτηριστικό του αντίστοιχου* Q^{SQL}. *Με την σειρά σε κάθε διάσταση αποδίδονται τα παρακάτω χαρακτηριστικά: ποσοστό star joins (prc_star), ποσοστό chain joins (prc_chain), αριθμός joins (joins), ποσοστό γνωρισμάτων επιλογής από το σύνολο των γνωρισμάτων επιλογής όλων των πινάκων (prc_select), ποσοστό πινάκων που χρησιμοποιούνται από το σύνολο των διαθέσιμων πινάκων (prc_from), αριθμός σταθερών ή αριθμητικών περιορισμών (constraints). Συμβολίζουμε:*

$$Q^V = v = [prc\_star\,, prc\_chain, joins, prc\_select, prc\_from, constraints]$$

Με βάση των παραπάνω ορισμό ενός ερωτήματος χρήστη είναι εύκολο τώρα να δώσουμε έναν ορισμό για την δομική ομοιότητα.

Ορισμός 4

Έστω δύο SQL ερωτήματα Q_1^{SQL} και Q_2^{SQL} που αναπαρίστανται ως Q_1^V και Q_2^V αντίστοιχα. Ορίζουμε ως δομική ομοιότητα των ερωτημάτων Q_1^{SQL} και Q_2^{SQL} το πόσο μοιάζουν στα δομικά χαρακτηριστικά τους όπως αυτά αναπαρίστανται στα διανύσματα Q_1^V και Q_2^V και υπολογίζεται ως το αντίστροφο της απόστασης μεταξύ των διανυσματικών τους αναπαραστάσεων Q_1^V και Q_2^V,

$$Structural\ Similarity = \frac{1}{1 + \ distance(Q_1^V / Q_2^V)}$$

Είναι εύκολα αντιληπτό πως όσο η απόσταση των διανυσμάτων μικραίνει τόσο περισσότερο όμοια θεωρούμε τα ερωτήματα. Μικρότερη απόσταση σημαίνει και μικρότερες διαφορές στα δομικά τους χαρακτηριστικά και άρα τα ερωτήματα μοιάζουν δομικά περισσότερο. Ακόμα και ενστικτωδώς να το σκεφτούμε ερωτήματα που έχουν τα ίδια ή παραπλήσια ποσοστά ή αριθμό συνδέσεων, γνωρισμάτων επιλογής, πινάκων και περιορισμών μοιάζουν στο πως είναι κατασκευασμένα. Επίσης θα πρέπει να σημειώσουμε πως ο τρόπος με τον οποίο θα μετρηθεί η απόσταση των διανυσμάτων είναι στην ευχέρεια του χρήστη και εξαρτάται από το πλαίσιο εφαρμογής του ορισμού.

Να σημειωθεί πως καθώς η δομική ομοιότητα είναι αντιστρόφως ανάλογη της απόστασης, στη συνέχεια της εργασίας ασχολούμαστε κυρίως με την απόσταση των διανυσμάτων υπονοώντας την ομοιότητα των ερωτημάτων.

Με βάση τους παραπάνω ορισμούς μπορούμε επίσης να κάνουμε την παρακάτω υπόθεση:

Υπόθεση 3

Ερωτήματα που είναι δομικά όμοια πρέπει να έχουν τα διανύσματά τους στην ίδια ομάδα εάν χωρίσουμε σε ομάδες τα διανύσματα με βάση την απόσταση τους.

Λαμβάνοντας υπόψιν τις υποθέσεις και τους ορισμούς που δόθηκαν σε αυτή την παράγραφο περιγράφεται ο σχεδιασμός και η επιλογή των δεδομένων στην επόμενη παράγραφο.

3.1.3 Σχεδιασμός και Πειραματικά Δεδομένα

Με βάση τις ιδέες και τους ορισμούς των προηγούμενων παραγράφων η πειραματική διαδικασία μας θέλουμε να εκτελεί τις παρακάτω λειτουργίες. Ξεκινώντας, σε κάθε περίπτωση το σύστημα θα πρέπει να μπορεί να μετατρέπει ένα ερώτημα από την SQL μορφή του στη διανυσματική του. Κατά την διάρκεια εκπαίδευσης του θα πρέπει να μπορεί να ομαδοποιήσει τα διανυσματικά ερωτήματα που έχουν δημιουργηθεί σε προηγούμενο βήμα και να αποδώσει σε αυτά ετικέτες που θα δείχνουν την ομάδα στην οποία ανήκουν. Συνεχίζοντας την εκπαίδευση, θα πρέπει να μπορεί να εκπαιδεύσει έναν ταξινομητή χρησιμοποιώντας τα διανυσματικά ερωτήματα τα οποία πλέον έχουν ετικέτες σαν παραδείγματα ώστε ο ταξινομητής να μπορεί να ταξινομήσει κάθε καινούργιο ερώτημα

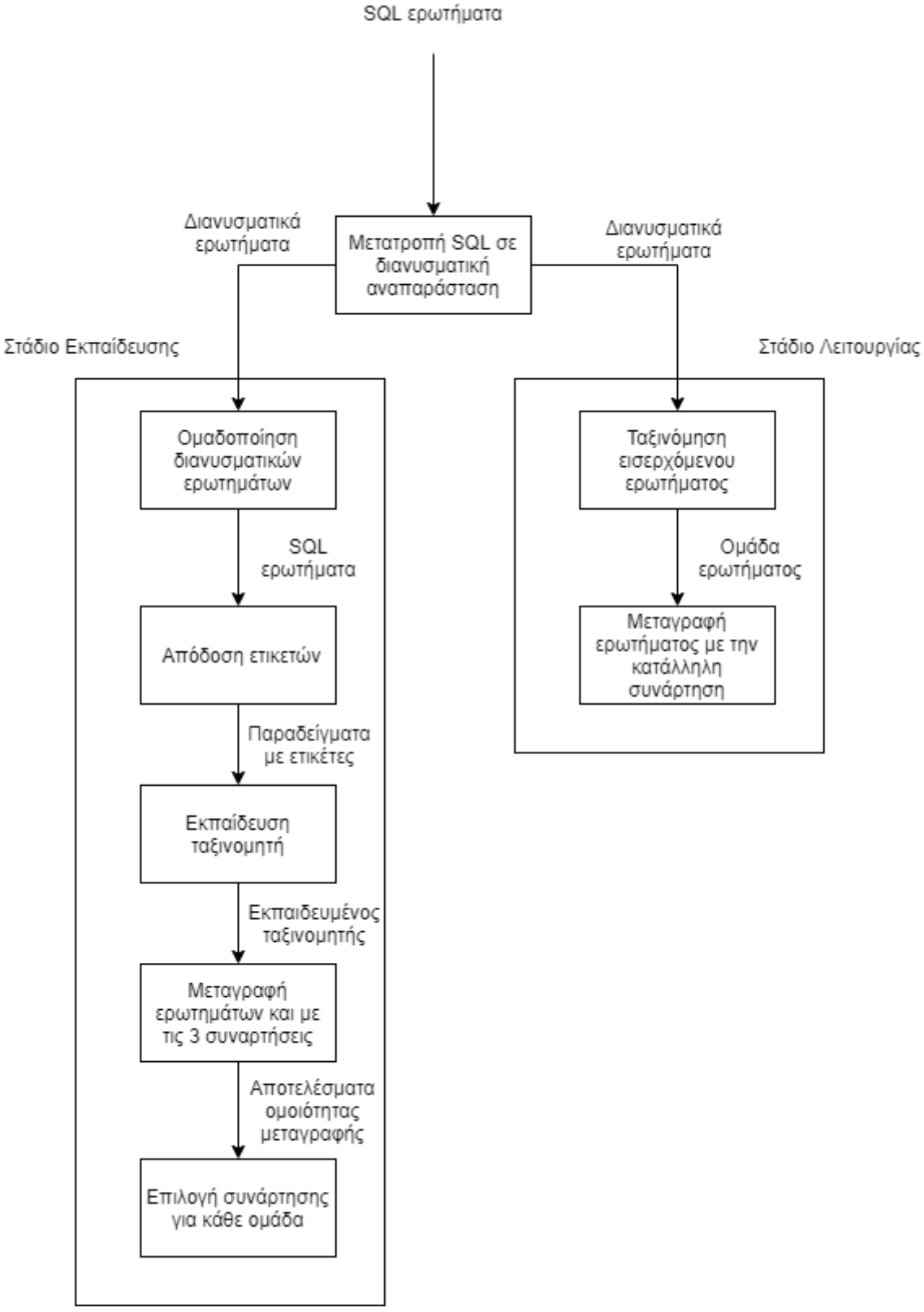
στη σωστή ομάδα. Τέλος στο στάδιο της εκπαίδευσης θα πρέπει να μπορεί να μεταγράψει μια επιλογή ερωτημάτων από κάθε ομάδα και να επιλέξει την κατάλληλη συνάρτηση για την ομάδα αυτή. Στο στάδιο λειτουργίας το σύστημα θα πρέπει να μπορεί να βρει μέσω του ταξινομητή την ομάδα στην οποία ανήκει το εισερχόμενο ερώτημα και να μεταγράψει το ερώτημα χρησιμοποιώντας την αντίστοιχη συνάρτηση ομοιότητας για την συγκεκριμένη ομάδα. Όλα τα βήματα της διαδικασίας παρουσιάζονται και σχηματικά στο διάγραμμα a.a.

Για να επιτελέσουμε τις παραπάνω λειτουργίες σχεδιάσαμε το λογισμικό της πειραματικής διαδικασίας έτσι ώστε οι λειτουργίες να επιτελούνται από διαφορετικά μέρη του λογισμικού, τα οποία μπορούν να λειτουργούν τόσο αυτόνομα όσο και σαν ένα ενιαίο σύστημα. Η σχεδιαστική αυτή επιλογή έγινε τόσο για να μπορούν να ελεγχθούν οι επιμέρους υποθέσεις αυτόνομα όσο και να μπορεί η πειραματική διαδικασία να διαφοροποιηθεί στο μέλλον αναλόγως με τις υποθέσεις υπό εξέταση. Τα διάφορα μέρη του λογισμικού παρουσιάζονται αναλυτικά σε επόμενη παράγραφο.

Έχοντας ξεκαθαρίσει τις λειτουργίες που πρέπει αν επιτελούνται στην πειραματική διαδικασία, το επόμενο βήμα του σχεδιασμού είναι επιλογή των ερωτημάτων τα οποία θα ελεγχθούν. Έχοντας ως πλαίσιο εφαρμογής την μεταγραφή ερωτημάτων και ειδικότερα την εργασία [1], τα ερωτήματα που θα εξετάσουμε θα είναι ανάλογα με αυτά που εξετάζονται στις εργασίες γύρω από τη μεταγραφή ερωτημάτων. Αρχικά από την βιβλιογραφία μας προκύπτει ότι τα ερωτήματα που έχουν ιδιαίτερο ενδιαφέρον στο συγκεκριμένο ερευνητικό πεδίο είναι αυτά που οι γράφοι τους ανήκουν στις κατηγορίες star join και chain join που αναφέρθηκαν στην παράγραφο 2.5. Αυτά τα είδη ερωτημάτων απασχολούν άλλωστε και την εργασία [1]. Επίσης από τημ εργασία [1] εξετάζονται μόνο τα ερωτήματα τύπου SPJ (select – project - join) επαυξημένα από συγκριτικούς περιορισμούς και περιορισμούς τιμών.. Τα SPJ ερωτήματα αποτελούν ένα υποσύνολο ερωτημάτων που παρότι είναι τα πιο απλά σε μορφή ερωτήματα και δεν περιέχουν σύνθετες πληροφορίες, είναι και από τα πλέον δημοφιλή σε πραγματικές συνθήκες λειτουργίας ενός συστήματος διαχείρισης βάσεων δεδομένων. Ακολουθώντας, λοιπόν, τα παραδείγματα της βιβλιογραφίας και τις επιλογές της εργασίας [1] που αποτελεί το πλαίσιο εφαρμογής μας, αποφασίσαμε να μελετήσουμε κι εμείς SPJ ερωτήματα με τους ανάλογους περιορισμούς που οι γράφοι τους ανήκουν είτε στην star join κατηγορία είτε στην chain join είτε αποτελούν έναν συνδυασμό αυτών. Μάλιστα θεωρούμε πως ιδιαίτερο ενδιαφέρον έχει η μελέτη ερωτημάτων που συνδυάζουν τις δύο κατηγορίες γράφων σε διαφορετικά ποσοστά.

Επίσης μελετώντας την εργασία [34], που αποτελεί επίσης μια βάση για την έρευνα μας, αλλά και γενικότερα τις εργασίες πάνω στην ομοιότητα ερωτημάτων είναι εύλογο να αναρωτηθεί κανείς αν άλλες ιδιότητες ενός ερωτήματος παίζουν κάποιον ρόλο. Θεωρούμε πως μεγάλο ενδιαφέρον έχει η μελέτη της συμπεριφοράς ερωτημάτων που περιέχουν διαφορετικό αριθμό γνωρισμάτων επιλογής και περιορισμών καθώς επίσης και συνδέσεων (joins). Η απουσία ενός γνωρίσματος επιλογής όταν ο αριθμός των γνωρισμάτων επιλογής είναι μεγάλος αποκρύπτει πολύ μικρότερη πληροφορία από ότι όταν τα γνωρίσματα επιλογής του αρχικού ερωτήματος είναι λίγα. Αντίστοιχα και για τους περιορισμούς, ένας περιορισμός συνεισφέρει περισσότερο στην πληροφορία του αποτελέσματος όταν το πλήθος των περιορισμών είναι μικρό από ότι όταν υπάρχουν πολλοί περιορισμοί.

Συνοψίζοντας όσον αφορά τα ερωτήματα προς εξέταση, στην παρούσα εργασία μας ενδιαφέρουν τα ερωτήματα τύπου SPJ με περιορισμούς σύγκρισης και τιμών, τα οποία είναι είτε star join είτε chain join είτε ένας συνδυασμός αυτών και διαθέτουν διαφορετικά πλήθη γνωρισμάτων επιλογής, περιορισμών και συνδέσεων.



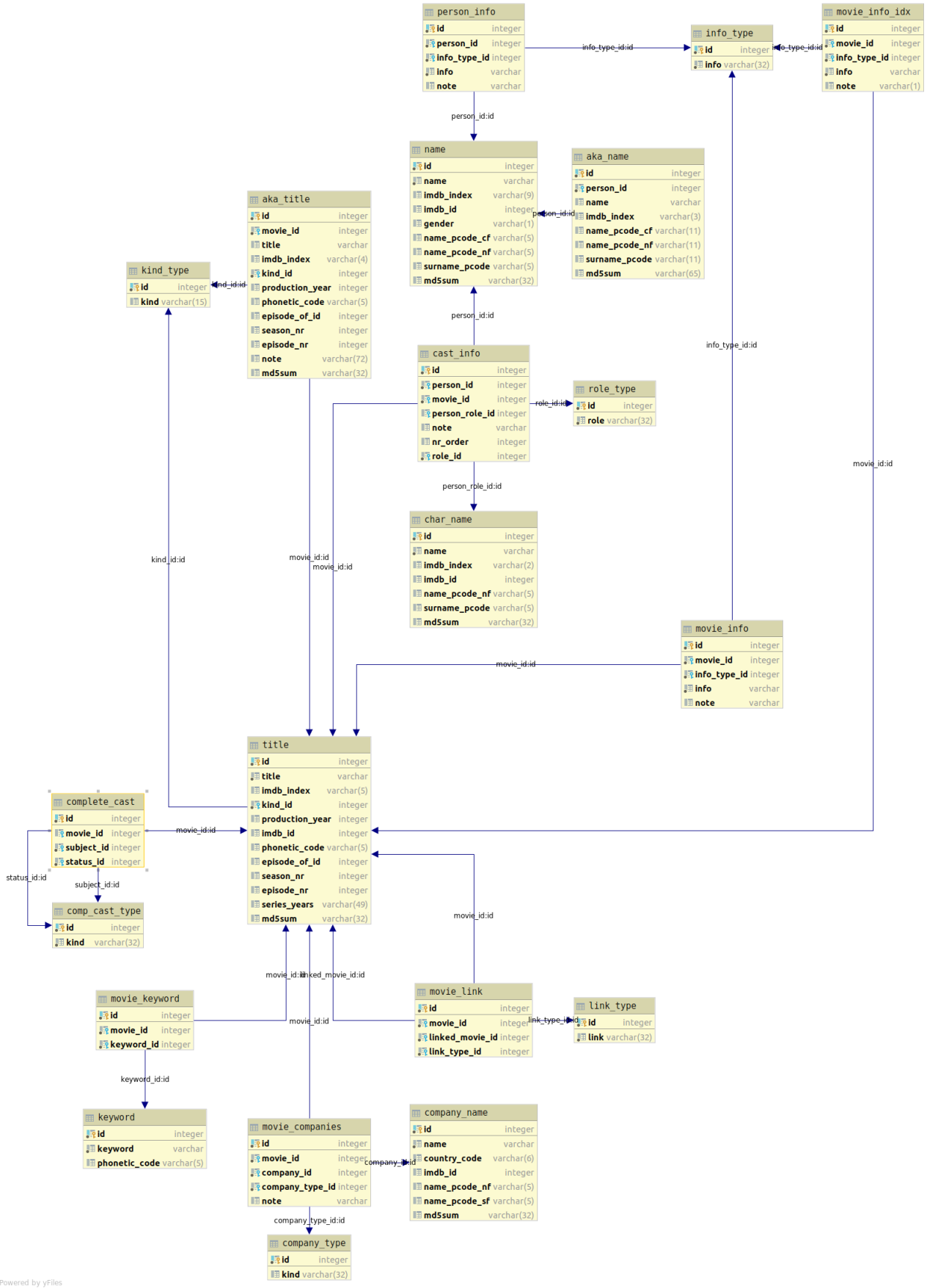
Σχήμα 3.1: Λογική λειτουργίας συστήματος

Για να μελετήσουμε τα παραπάνω ερωτήματα χρειάζεται να επιλέξουμε και τα κατάλληλα πειραματικά δεδομένα πάνω στα οποία θα παράγουμε τα ερωτήματα. Η συνήθης τακτική για τέτοιου είδους προβλήματα και πειραματικές διαδικασίες είναι η δημιουργία συνθετικών δεδομένων. Αν και η σχετική βιβλιογραφία μας ακολουθεί αυτή την τακτική, εμείς αποφασίσαμε να ακολουθήσουμε την προσέγγιση της εργασίας [22]. Στην εργασία [22] επιλέγεται ως σύνολο δεδομένων το σύνολο δεδομένων της ιστοσελίδας ταινιών IMDb, ένα πραγματικό σύνολο δεδομένων. Η εργασία [22] αφορά το πρόβλημα της σειράς ικανοποίησης συνδέσεων, ένα πρόβλημα που εστιάζει επίσης στα ερωτήματα και διαθέτει παραπλήσια φιλοσοφία. Συνεπώς θεωρούμε ότι το συγκεκριμένο σύνολο δεδομένων μπορεί να καλύψει τις απαιτήσεις μας για την παρούσα πειραματική διαδικασία και επιπλέον ως ένα πραγματικό σύνολο δεδομένων μπορεί να προσφέρει πρακτικά χαρακτηριστικά και παραπάνω δυνατότητες αξιολόγησης για μελλοντικές επεκτάσεις της εργασίας. Το σχήμα αυτού του συνόλου δεδομένων που επιλέχθηκε για την εργασία παρουσιάζεται στο σχήμα b.b.

Παρότι τα δεδομένα που χρησιμοποιήθηκαν είχαν χρησιμοποιηθεί και στην εργασία [22] περιείχαν χαρακτήρες που δημιουργούσαν πρόβλημα κατά την εισαγωγή στο σύστημα διαχείρισης βάσεων δεδομένων της παρούσας εργασίας. Επειδή η διατήρηση της ακεραιότητας των πραγματικών δεδομένων θεωρήθηκε πολύ σημαντική για μελλοντικές επεκτάσεις, οι χαρακτήρες αυτοί είτε διορθώθηκαν είτε αφαιρέθηκαν με όσο το δυνατόν μη παρεμβατικό τρόπο ήταν αυτό δυνατό, ώστε να μην επηρεαστούν σημαντικά τα πραγματικά δεδομένα.

Τέλος, το σύστημα της εργασίας [1] επηρεάζεται από διάφορες παραμέτρους. Επειδή η ταυτόχρονη μελέτη όλων των παραμέτρων δεν είναι δυνατή, η παρούσα εργασία επικεντρώνεται στα ερωτήματα και τα διάφορα είδη αυτών και θεωρεί σταθερές και δεδομένες όλες τις άλλες παραμέτρους. Χαρακτηριστικά αναφέρεται ότι για την μεταγραφή χρησιμοποιείται μονάχα ένα σταθερό σύνολο αντιστοιχίσεων (mappings) και δεν εξετάζεται στην παρούσα φάση η συμπεριφορά του συστήματος για διαφορετικά σύνολα αντιστοιχίσεων.

Να σημειωθεί πως για την υλοποίηση της πειραματικής διαδικασίας και του απαραίτητου λογισμικού χρησιμοποιήθηκε το σύστημα διαχείρισης βάσεων δεδομένων PostgreSQL και η γλώσσα προγραμματισμού Python 3. Πρόκειται για εργαλεία ελεύθερου λογισμικού που χαίρουν άκρας εκτίμησης και προσφέρουν πολλές δυνατότητες και βιβλιοθήκες και πλούσια τεχνική τεκμηρίωση αυτών.



Σχήμα 3.2: Σχήμα βάσης

3.2 Δεύτερο Στάδιο: Παραγωγή ερωτημάτων

Έχοντας λάβει την απόφαση να κάνουμε χρήση τεχνικών μηχανικής μάθησης, είναι ξεκάθαρο πως θα χρειαστεί μεγάλος αριθμός ερωτημάτων τόσο για την σωστή κατηγοριοποίηση των ερωτημάτων και την επαρκή εκπαίδευση του ταξινομητή. Η πρώτη σκέψη ήταν η παραγωγή των ερωτημάτων με το χέρι ώστε τα ερωτήματα να έχουν κάποιο σημασιολογικό περιεχόμενο. Όμως το μεγάλο πλήθος των απαιτούμενων ερωτημάτων καθιστούσε αυτή την επιλογή πολύ χρονοβόρα και δύσκολη. Άλλωστε για την παρούσα εργασία το σημασιολογικό περιεχόμενο των ερωτημάτων δεν μας ενδιαφέρει. Τα μόνα χαρακτηριστικά των ερωτημάτων που μας ενδιαφέρουν είναι τα δομικά. Συνεπώς το αν τα ερωτήματα είχαν κάποιο νόημα για τον άνθρωπο ήταν δευτερεύον.

Έτσι για την γρήγορη και αποτελεσματική παραγωγή ερωτημάτων διαφόρων χαρακτηριστικών απευθυνθήκαμε στην λύση της αυτόματης παραγωγής μέσω κατάλληλου λογισμικού. Το λογισμικό παράχθηκε από εμάς στη γλώσσα προγραμματισμού Python 3 και ο τρόπος λειτουργίας του αναλύεται παρακάτω. Το λογισμικό έχει διαμορφωθεί έτσι ώστε να παράγει ερωτήματα που να εξυπηρετούν τους στόχους τις έρευνάς μας και να εκτελείται αυτόνομα σε σχέση με τα υπόλοιπα μέρη της πειραματικής διαδικασίας. Ξεκινά διαβάζοντας από την υποκείμενη βάση το σχήμα. Σε πρώτη φάση χρησιμοποιώντας τις πληροφορίες του σχήματος δημιουργεί ερωτήματα για κάθε σχέση με διαφορετικό πλήθος γνωρισμάτων επιλογής, ξεκινώντας από την επιλογή ενός μόνο γνωρίσματος και τελειώνοντας με την επιλογή όλων των διαθέσιμων γνωρισμάτων της σχέσης. Στη συνέχεια συλλέγει σε μια λίστα από αυτά τα ερωτήματα το ερώτημα για κάθε σχέση που περιέχει όλα τα γνωρίσματα της σχέσης και με βάση μια λίστα από τις δυνατές συνδέσεις δημιουργεί ερωτήματα τύπου star join. Χρησιμοποιώντας την ίδια λίστα με τα "μεγάλα" ερωτήματα και τη λίστα με τις δυνατές συνδέσεις παράγει και τα ερωτήματα τύπου chain join. Κάνοντας χρήση των star join ερωτημάτων που έχουν ήδη δημιουργηθεί δημιουργεί star join ερωτήματα με διαφορετικό πλήθος γνωρισμάτων επιλογής. Αντίστοιχη διαδικασία ακολουθείται και για τα chain join ερωτήματα. Εν συνεχεία τα δύο ήδη (star και chain) συνδυάζονται και παράγουν μεικτά ερωτήματα με διάφορα πλήθη γνωρισμάτων επιλογής. Τέλος το λογισμικό εισάγει στα ερωτήματα που έχουν δημιουργηθεί σε όλες τις παραπάνω φάσεις περιορισμούς, είτε τιμών είτε σύγκρισης, με τη χρήση μιας λίστας από πιθανούς περιορισμους. Τα ερωτήματα από όλες τις φάσεις αποθηκεύονται σε αρχεία με το αντίστοιχο όνομα και από κάθε αρχείο επιλέγεται ένας αριθμός από τυχαία ερωτήματα για να συγκροτηθεί το αρχείο για τα πειράματα μας.

Σε όλη τη διαδικασία παραγωγής των ερωτημάτων όπου ήταν αναγκαία η επιλογή μιας παραμέτρου προτιμήθηκε αυτή να επιλεγεί τυχαία ώστε να υπάρχει όσο το δυνατόν μεγαλύτερη ποικιλία στα παραγόμενα ερωτήματα. Να σημειωθεί ότι οι λίστες με τις διαθέσιμες συνδέσεις και τους διαθέσιμους περιορισμούς αποτελούν μέρος του προγράμματος και προς το παρόν δεν μπορούν να δοθούν εξωτερικά από το χρήστη.

Στη συνέχεια παρουσιάζουμε συνοπτικά τις βασικότερες συναρτήσεις του λογισμικού για την παραγωγή των ερωτημάτων:

dif_num_select(schema): Παίρνει σαν όρισμα το σχήμα της υποκείμενης βάσης και δημιουργεί ερωτήματα για κάθε σχέση του σχήματος προσθέτοντας σε κάθε επανάληψη και ένα παραπάνω γνώρισμα επιλογής. Επιστρέφει ένα λεξικό με κλειδιά τις σχέσης του σχήματος και τιμές λίστες από ερωτήματα για την κάθε σχέση.

star_joins(queries, fk_joins, parser): Παίρνει σαν ορίσματα μια λίστα από ερωτήματα, μια λίστα από διαθέσιμες συνδέσεις, και έναν αναλυτή ερωτημάτων. Με βάση τα ερωτήματα και την λίστα συνδέσεων παράγει νέα ερωτήματα τύπου star join. Επιστρέφει μια λίστα ερωτημάτων τύπου star join.

chain_joins(queries, fk_joins, parser): Παίρνει σαν ορίσματα μια λίστα από ερωτήματα, μια λίστα από διαθέσιμες συνδέσεις, και έναν αναλυτή ερωτημάτων. Με βάση τα ερωτήματα και την λίστα συνδέσεων παράγει νέα ερωτήματα τύπου chain join. Επιστρέφει μια λίστα ερωτημάτων τύπου chain join.

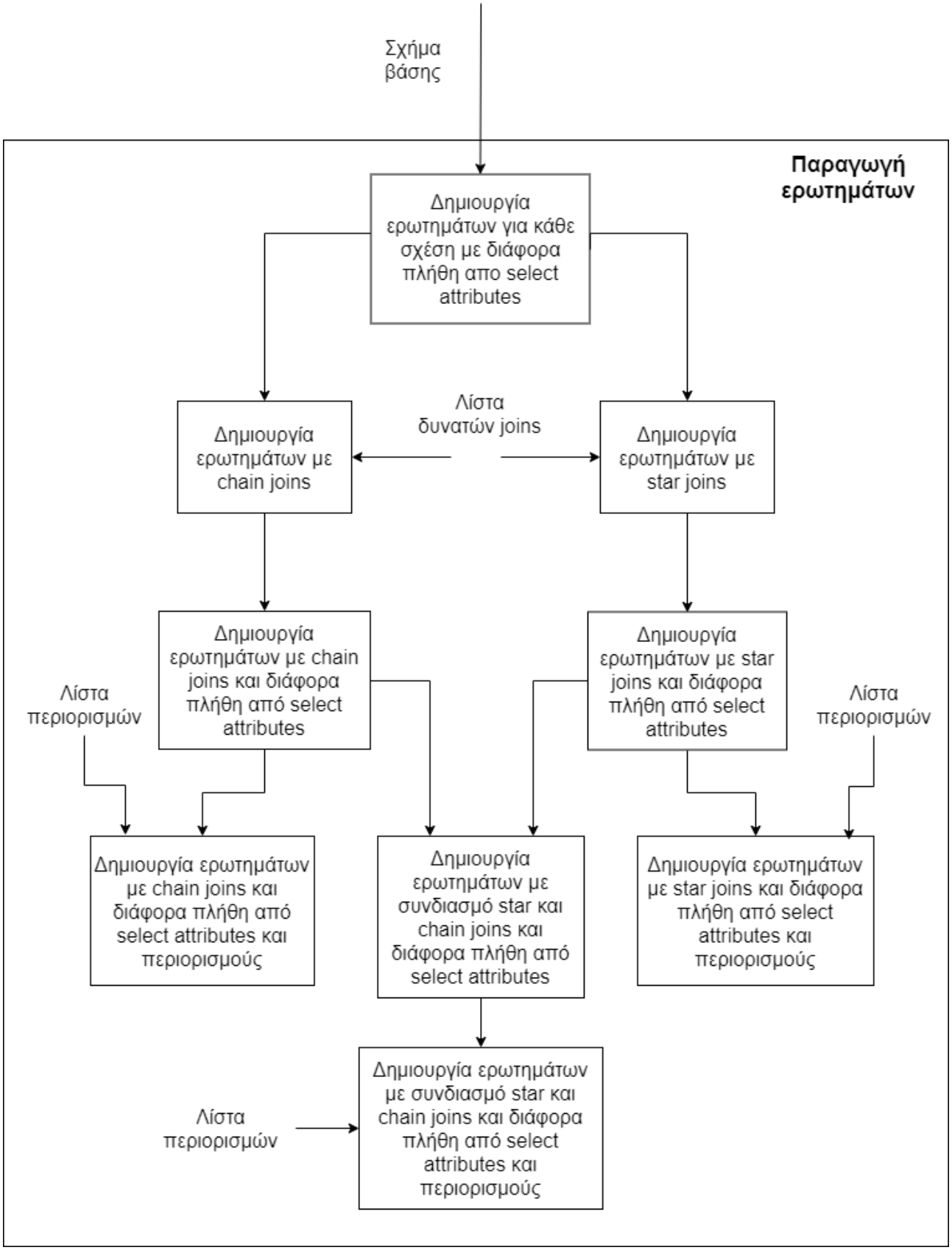
dif_num_select_with_joins(given_queries, parser): Παίρνει σαν όρισμα μια λίστα από ερωτήματα και έναν αναλυτή ερωτημάτων. Από τα δοθέντα ερωτήματα δημιουργεί νέα ερωτήματα με διαφορετικό αριθμό γνωρισμάτων επιλογής. Επιστρέφει τη λίστα με τα νέα ερωτήματα.

combine_joins(stars, chains, parser): Παίρνει σαν ορίσματα μια λίστα από star join ερωτήματα, μια λίστα από chain join ερωτήματα και έναν αναλυτή ερωτημάτων. Συνδυάζει τα ερωτήματα που μπορούν να συνδυαστούν από τις δύο λίστες και δημιουργεί μεικτά ερωτήματα. Επιστρέφει μια λίστα με μικτά ερωτήματα.

add_constraints(queries, constraints, parser): Παίρνει σαν ορίσματα μια λίστα από ερωτήματα, μια λίστα από περιορισμούς και έναν αναλυτή ερωτημάτων. Εισάγει στα δοθέντα ερωτήματα όπου υπάρχει η δυνατότητα έναν τυχαίο αριθμό περιορισμών. Επιστρέφει μια λίστα ερωτημάτων με περιορισμούς.

Για την ανάλυση των ερωτημάτων από το λογισμικό σε όλη την εργασία χρησιμοποιήθηκε ο αναλυτής simpleSQLParser από τα παραδείγματα της βιβλιοθήκης pyparsing.

Ο κώδικας για την δημιουργία των ερωτημάτων παρατίθεται στο παράρτημα Α.



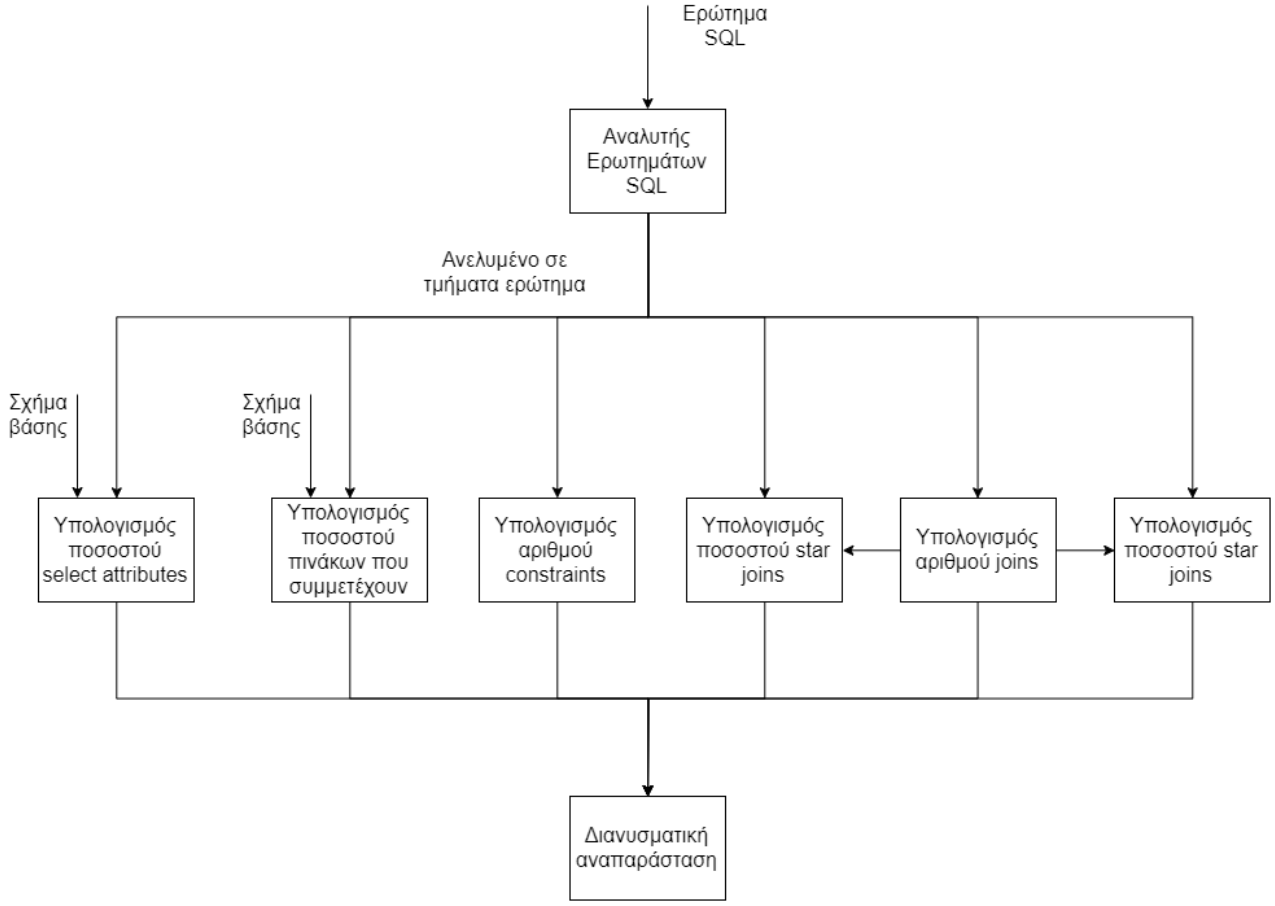
Σχήμα 3.3: Διαδικασία παραγωγής ερωτημάτων

3.3 Τρίτο Στάδιο: Ομαδοποίηση Ερωτημάτων και Ταξινόμηση

Βασικός στόχος της παρούσας εργασίας είναι η ομαδοποίηση των ερωτημάτων σε ομάδες όμοιων ερωτημάτων, σύμφωνα με τον ορισμό της ομοιότητας που δώσαμε στην παράγραφο 3.1.2. Στην παράγραφο αυτή επισημάναμε πως η ομοιότητα δύο ερωτημάτων μπορεί να δοθεί από την απόσταση των διανυσματικών τους αναπαραστάσεων. Όσο μικρότερη είναι η απόσταση των διανυσματικών αναπαραστάσεων τόσο μεγαλύτερη είναι η ομοιότητα των SQL ερωτημάτων. Όμως η έννοια της απόστασης μεταξύ των παραδειγμάτων είναι η βάση των αλγορίθμων ομαδοποίησης. Συνεπώς είναι πολύ εύκολο να μετρήσουμε την ομοιότητα μέσω της απόστασης στις τεχνικές ομαδοποίησης της μηχανικής μάθησης. Για τους αλγόριθμους και την εφαρμογή των τεχνικών μηχανικής μάθησης χρησιμοποιήθηκαν οι βιβλιοθήκες numpy, scikit-learn και pandas της python και οι υλοποιήσεις που αυτές προσφέρουν.

Για να πραγματοποιηθεί, όμως, η ομαδοποίηση των ερωτημάτων έπρεπε πρώτα τα ερωτήματα να μετατραπούν στην διανυσματική τους αναπαράσταση. Για το σκοπό αυτό δημιουργήσαμε έναν δικό μας αναλυτή/μετατροπέα ερωτημάτων που είχε ως βάση για την ανάλυση των ερωτημάτων τον simpleSQLParser. Η λειτουργία του μετατροπέα στηρίζεται στην αναπαράσταση των ερωτήματων που έχουμε επιλέξει για την παρούσα εργασία. Ο μετατροπέας διαβάζει από ένα αρχείο τα ερωτήματα προς μετατροπή σε διανυσματική μορφή και εξάγει τα απαραίτητα χαρακτηριστικά για κάθε ένα από αυτά. Συγκεκριμένα, αρχικά για κάθε ερώτημα ο μετατροπέας συλλέγει των αριθμό των σχέσεων και τον αριθμό των συνολικών γνωρισμάτων της βάσης και υπολογίζει μέσω αυτών το ποσοστό των γνωρισμάτων που έχει ένα ερώτημα σε σχέση με το σύνολο των γνωρισμάτων και το ποσοστό των σχέσεων που συμμετέχουν. Στη συνέχεια υπολογίζει τον αριθμό των star joins, chain joins και των συνολικών συνδέσεων καθώς και των περιορισμών τιμών ή σύγκρισης και υπολογίζει τα ποσοστά για τα star και chain χαρακτηριστικά. Δεν γίνεται διάκριση μεταξύ περιορισμών τιμών και περιορισμών σύγκρισης. Ως chain join θεωρούμε όλες τις αλυσίδες συνδέσεων στις οποίες κάθε σχέση εμφανίζεται το πολύ δύο φορές ενώ για τα star joins απαιτούμε μια σχέση να εμφανίζεται σε τουλάχιστον τρεις συνδέσεις. Η βασική συνάρτηση του λογισμικού μας που επιτελεί αυτή μετατροπή είναι η extract_features(self,tokens) που παίρνει ως όρισμα τα διάφορα στοιχεία του ερωτήματος αναλυμένα από τον simpleSQLParser και επιστρέφει την διανυσματική αναπαράσταση του ερωτήματος. Τέλος, ο μετατροπέας αποθηκεύει τις αναπαραστάσεις σε ένα αρχείο κειμένου ώστε αυτές να χρησιμοποιηθούν μετέπειτα από τους αλγόριθμους μηχανικής μάθησης.

Αφού γίνει η μετατροπή των ερωτημάτων, ξεκινά η διαδικασία της ομαδοποίησης. Για την δημιουργία, λοιπόν, των κατηγοριών των ερωτημάτων αποφασίσαμε να χρησιμοποιήσουμε έναν ιεραρχικό αλγόριθμο ομαδοποίησης. Πιο αναλυτικά ο αλγόριθμος που επιλέχθηκε ανήκει στην κατηγορία των συγχωνευτικών ιεραρχικών αλγορίθμων και η υλοποίηση που χρησιμοποιήθηκε είναι αυτή που παρέχεται από τη βιβλιοθήκη scikit-learn. Η επιλογή αυτή βασίστηκε στο γεγονός πως δεν γνωρίζαμε εκ των προτέρων τον κατάλληλο αριθμό ομάδων αφού θέλαμε να ανακαλύψουμε μέσω της ομαδοποίησης ομοιότητες που δεν γνωρίζαμε εξ' αρχής. Επίσης μέσω του δενδροδιαγράμματος που προσφέρει ένας ιεραρχικός αλγόριθμος ομαδοποίησης ήταν εύκολο να οπτικοποιήσουμε τις αποστάσεις και να επιλέξουμε τον αριθμό των ομάδων που μας εξυπηρετούσε. Οι ομάδες που δημιουργήθηκαν από τον αλγόριθμο ομαδοποίησης ήταν και οι κατηγορίες που αποδόθηκαν στα ερωτήματα. Μετά τη δημιουργία των ομάδων, αυτές ελέγχθηκαν ποιοτικά και αναγνωρίστηκαν τα χαρακτηριστικά των ερωτημάτων που τις απαρτίζουν.



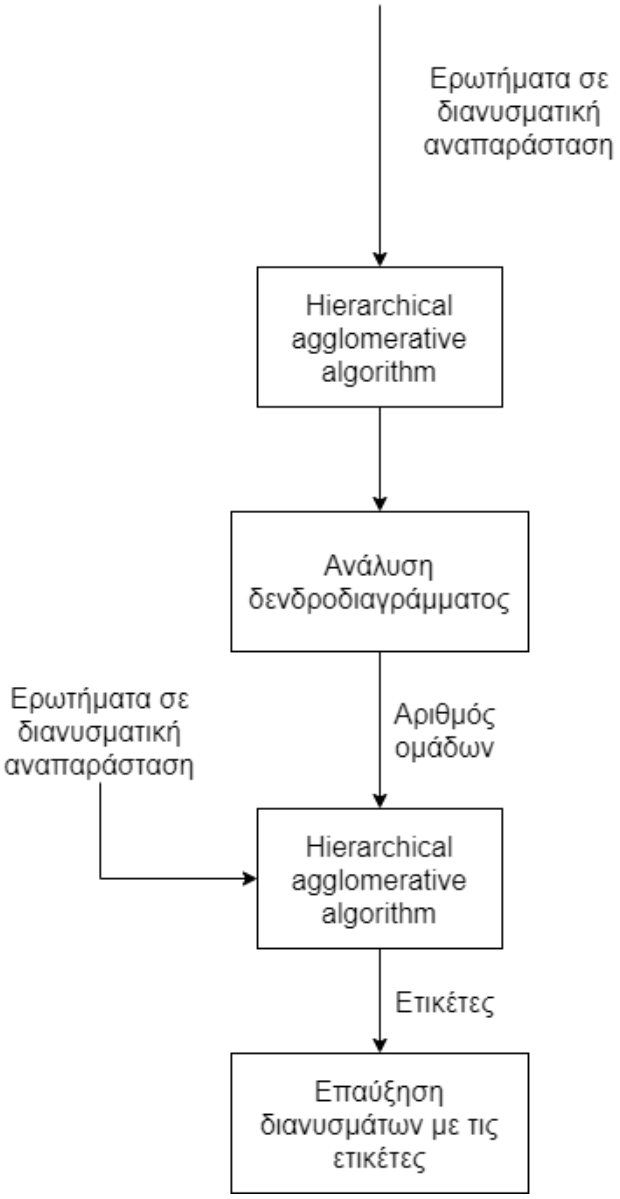
Σχήμα 3.4: Διαδικασία μετατροπής αναπαράστασης

Πιο αναλυτικά, η λειτουργία του λογισμικού για την ομαδοποίηση των ερωτημάτων περιγράφεται παρακάτω. Αρχικά το λογισμικό φορτώνει τις αναπαραστάσεις των ερωτημάτων από το αρχείο στο οποίο έχουν αποθηκευτεί στην κατάλληλη μορφή για την χρήση τους από τον αλγόριθμο ομαδοποίησης. Στη συνέχεια τα διανύσματα ομαδοποιούνται με τον ιεραρχικό αλγόριθμο χωρίς να δοθεί κάποιος αριθμός ομάδων και λαμβάνεται το δενδροδιάγραμμα. Από το δενδροδιάγραμμα αποφασίζεται ο αριθμός των ομάδων και ο αλγόριθμος χρησιμοποιείται ξανά με δεδομένο αριθμό ομάδων για την απόδοση του αριθμού ομάδας στα ερωτήματα. Αυτοί οι αριθμοί χρησιμοποιούνται για να επαυξήσουν τα διανύσματα ως ετικέτες.

Ως παράμετροι του αλγορίθμου για την μέτρηση της απόστασης και την συγχώνευση των ομάδων ήταν οι προεπιλεγμένες. Χρησιμοποιήθηκε η ευκλείδεια απόσταση και η συγχώνευση ward που έχει ως στόχο την ελαχιστοποίηση της διασποράς στις ομάδες που συγχωνεύονται. Οι παράμετροι αυτές όπως και οι υπόλοιπες που χρησιμοποιήθηκαν κατά προεπιλογή ήταν σταθερές και ίδιες για όλες τις προσεγγίσεις και τα πειράματα που επιχειρήθηκαν.

Η πρώτη προσέγγιση για την ομαδοποίηση των ερωτημάτων ήταν η χρήση των διανυσμάτων ακριβώς όπως αυτά παράχθηκαν. Χρησιμοποιήθηκαν και τα έξι χαρακτηριστικά της αναπαράστασης και προέκυψαν οχτώ ομάδες/κατηγορίες. Ύστερα από μελέτη και ανάλυση των ερωτημάτων που απαρτίζουν την κάθε μία από τις οχτώ ομάδες διαπιστώθηκε πως κατά την ομαδοποίηση τα χαρακτηριστικά που έδιναν τον αριθμό των συνδέσεων και των περιορισμών κυριάρχισαν στο

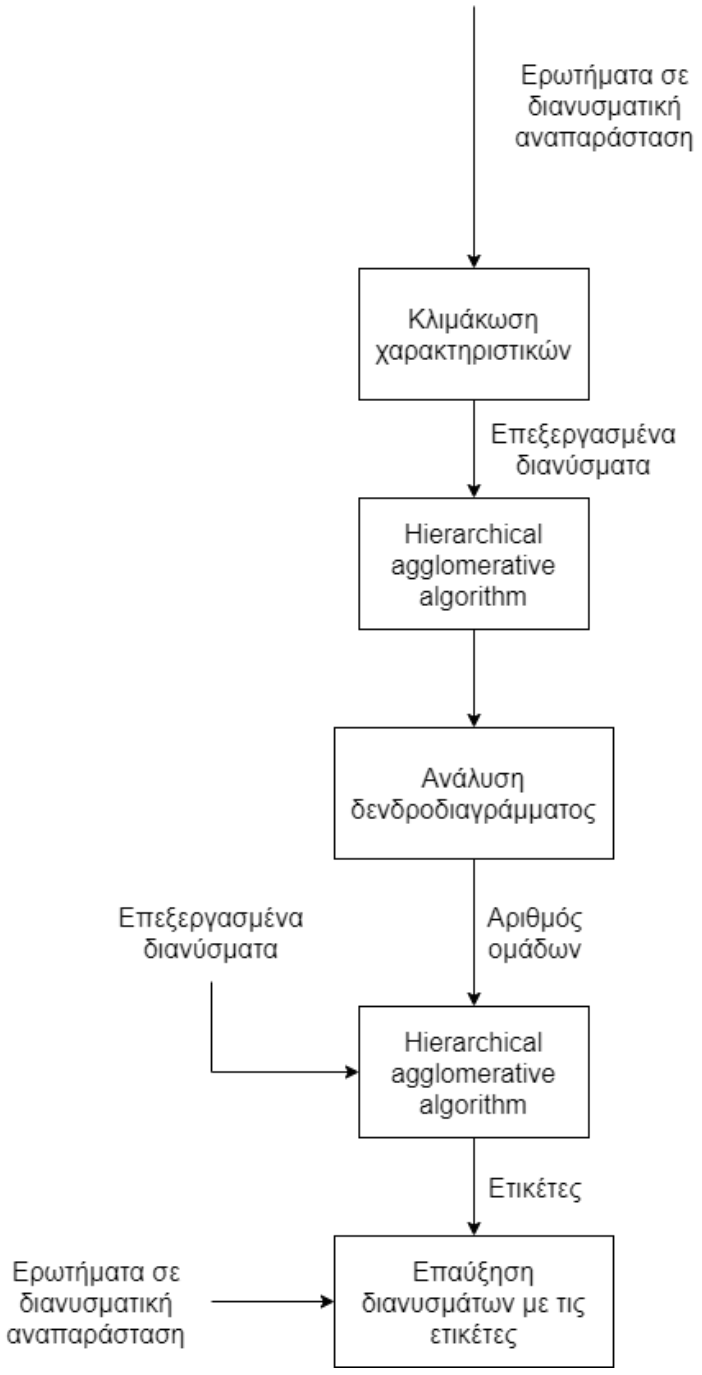
διαχωρισμό των ομάδων και καθόρισαν το πως θα χωριστούν τα ερωτήματα. Τα αποτελέσματα αυτής της προσέγγισης θα συζητηθούν στο επόμενο κεφάλαιο αναλυτικότερα.



Σχήμα 3.5: Διαδικασία ομαδοποίησης – 1η προσέγγιση

Καθώς η αρχική επιθυμία μας ήταν ο καθορισμός των ομάδων με την ουσιαστική συμμετοχή όσο το δυνατόν περισσότερων χαρακτηριστικών, επιχειρήθηκε και μια δεύτερη προσέγγιση. Στη δεύτερη αυτή προσέγγιση περάσαμε τα διανύσματα των ερωτημάτων από μια συνάρτηση προεπεξεργασίας. Συγκεκριμένα χρησιμοποιήσαμε μια διαδικασία κλιμάκωσης που παρέχει η scikit-learn για να αποδώσουμε τα χαρακτηριστικά μας ανάλογα μεγέθη. Στην προσέγγιση αυτή προέκυψαν 10 κατηγορίες και ύστερα από μελέτη αυτών των κατηγοριών διαπιστώθηκε πως υπήρξε συνεισφορά στη διαμόρφωση των ομάδων από όλα τα χαρακτηριστικά σε κάποιο βαθμό. Τα αποτελέσματα κι

αυτής της προσέγγισης θα αναλυθούν περισσότερο στο επόμενο κεφάλαιο. Η προσέγγιση αυτή όμως ήταν και η επικρατούσα και αυτή που χρησιμοποιήθηκε και στη συνέχεια της εργασίας. Επειδή όμως η διαδικασία κλιμάκωσης δεν είναι δυνατή για τα εισερχόμενα νέα ερωτήματα οι ετικέτες ενσωματώθηκαν στα αρχικά διανύσματα, όπως ήταν πριν την προεπεξεργασία. Αυτό το σύνολο διανυσμάτων θα χρησιμοποιηθεί στη συνέχεια.

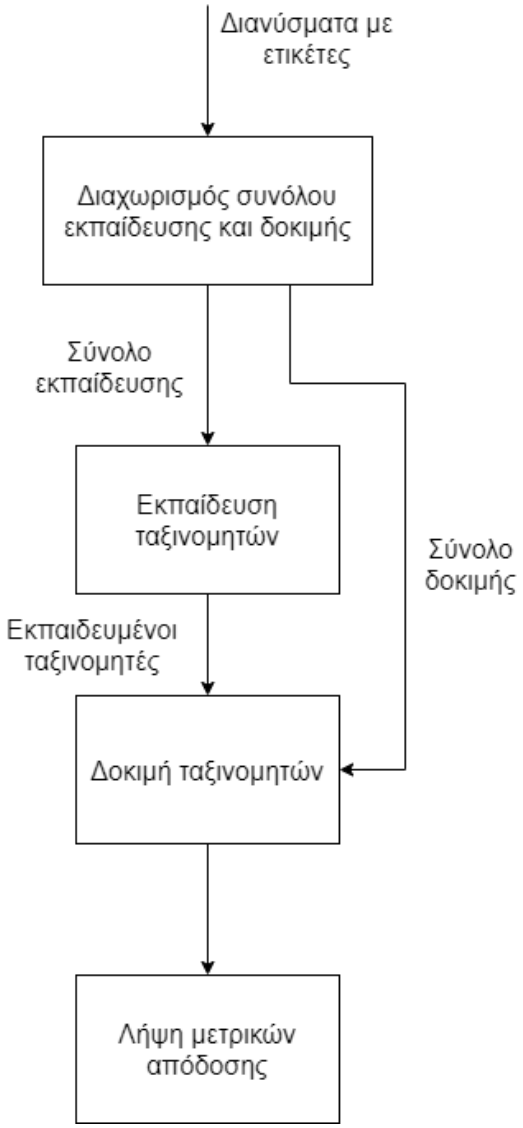


Σχήμα 3.6: Διαδικασία ομαδοποίησης – 2η προσέγγιση

Μετά την απόδοση των ετικετών στα παραδείγματα, σειρά έχει η εκπαίδευση ενός ταξινομητή ώστε κάθε νέο εισερχόμενο ερώτημα να ταξινομείται αυτόματα. Όπως αναφέρθηκε στην παράγραφο 2.12 υπάρχουν διάφοροι αλγόριθμοι ταξινόμησης και η επιλογή κάποιου εξαρτάται από την εφαρμογή και την απόδοση του σε αυτή. Με αυτό υπ' όψιν για να επιλέξουμε τον ταξινομητή που θα χρησιμοποιηθεί στο συγκεκριμένο πλαίσιο και για τα συγκεκριμένα δεδομένα πειραματιστήκαμε με διάφορους διαθέσιμους αλγόριθμους ταξινόμησης στην βιβλιοθήκη scikit-learn. Συγκεκριμένα δοκιμάσαμε την απόδοση των αλγορίθμων κ-Κοντινότερων Γειτόνων (k-Nearest Neighbours), δέντρων αποφάσεων (decision trees), γραμμικής ανάλυσης διακρίσεων (LDA), Naive Bayes, μηχανών διανυσμάτων υποστήριξης (SVM). Για τους παραπάνω αλγόριθμους συγκρίναμε της μετρικές precision, recall και F1 score και επιλέξαμε αυτόν με την καλύτερη απόδοση βάσει των μετρικών.

Πιο αναλυτικά, η εκπαίδευση του ταξινομητή έγινε με τη συνήθη διαδικασία εκπαίδευσης ενός ταξινομητή. Αφού τα διανύσματα/παραδείγματα και οι ετικέτες τους φορτώθηκαν στην κατάλληλη δομή για την επεξεργασία τους από τους αλγόριθμους μηχανικής μάθησης διαχωρίσαμε τις ετικέτες από τα διανύσματα διατηρώντας την αντιστοίχιση. Έπειτα χωρίσαμε τα παραδείγματα σε δύο σύνολα, ένα για εκπαίδευση και ένα για δοκιμή του μοντέλου ταξινόμησης. Χρησιμοποιήθηκαν οι προκαθορισμένες παράμετροι και συνεπώς το σύνολο εκπαίδευσης περιείχε το 75% των αρχικών παραδειγμάτων και το σύνολο δοκιμής το 25%. Τα συνολικά διαθέσιμα ερωτήματα που χρησιμοποιήθηκαν σε όλη τη διαδικασία ήταν 462. Η τοποθέτηση των ερωτημάτων στα δύο σύνολα ήταν τυχαία. Στη συνέχεια χρησιμοποιούμε το σύνολο εκπαίδευσης για να εκπαιδεύσουμε κάθε ταξινομητή και το σύνολο δοκιμής για να ελέγξουμε την απόδοση τους. Ο καλύτερος ταξινομητής αποθηκεύεται για χρήση στο στάδιο λειτουργίας. Τα αποτελέσματα των μετρικών για την απόδοση των ταξινομητών αναλύονται στο επόμενο κεφάλαιο.

Ο κώδικας του λογισμικού για όλο αυτό το στάδιο παρατίθεται στο παράρτημα Α.



Σχήμα 3.7: Διαδικασία εκπαίδευσης ταξινομητή

3.4 Τέταρτο Στάδιο

Έχοντας πλέον τον ταξινομητή μας εκπαιδευμένο, το επόμενο βήμα για την επίτευξη των στόχων μας είναι ο συνδυασμός του μετατροπέα του δεύτερου σταδίου και του ταξινομητή από το τρίτο στάδιο μαζί με το προϋπάρχον σύστημα προσεγγιστικής μεταγραφής ώστε αρχικά να μελετηθεί και να βρεθεί για κάθε κατηγορία ερωτημάτων η καλύτερη συνάρτηση ομοιότητας από τις τρεις διαθέσιμες.

Το νέο σύστημα μεταγραφής που προκύπτει λειτουργεί όπως περιγράφεται ακολούθως. Αρχικά διαβάζει από ένα αρχείο το ερώτημα προς μεταγραφή και δημιουργεί τη διανυσματική του αναπαράσταση. Στη συνέχεια φορτώνει τον αποθηκευμένο ταξινομητή, από τον οποίο ζητά να ταξινομήσει την αναπαράσταση του δοθέντος ερωτήματος. Στη συνέχεια καλείται το προϋπάρχον σύστημα για να κάνει τη μεταγραφή και στο τέλος αναλύονται τα αποτελέσματα του. Το σύστημα

προσεγγιστικής μεταγραφής κατά την φάση εξερεύνησης της βέλτιστης συνάρτησης για κάθε ομάδα καλείται τρεις φορές για κάθε διαθέσιμη συνάρτηση ομοιότητας. Στη φάση λειτουργίας ανάλογα με την απόφαση του ταξινομητή επιλέγεται και η συνάρτηση ομοιότητας.

Για την εξερεύνηση των συναρτήσεων επιλέχθηκαν 10 ερωτήματα από κάθε ομάδα κατευθείαν από τα αποτελέσματα της ομαδοποίησης. Ουσιαστικά κατά τη φάση της εξερεύνησης ο ταξινομητής δεν χρησιμοποιήθηκε για την εύρεση της κατηγορίας αφού ήταν ήδη γνωστή. Για κάθε ερώτημα τρέξαμε το σύστημα μεταγραφής και για τις τρεις συναρτήσεις και συλλέξαμε τα αποτελέσματα. Στα αποτελέσματα περιλαμβάνεται και το ποσοστό ομοιότητας σύμφωνα με τη συνάρτηση ομοιότητας που χρησιμοποιήθηκε. Για την εξερεύνηση μας χρησιμοποιήθηκε το συγκεκριμένο ποσοστό από κάθε συνάρτηση ως μετρική. Η επιλογή αυτή ακολουθεί την επιλογή ως μετρική της αντίστοιχης ποσότητας για ανάλογη αξιολόγηση των συναρτήσεων από την εργασία [34].

Για την εξερεύνηση χρησιμοποιήσαμε ένα μόνο σύνολο αντιστοιχήσεων. Προσπαθήσαμε να αποδώσουμε σε αυτό όσο περισσότερες ιδιομορφίες μπορούσαμε ώστε να καλύψουμε αρκετές περιπτώσεις. Το σύνολο των αντιστοιχήσεων παρουσιάζεται παρακάτω σε μορφή συζευκτικών ερωτημάτων.

Αντιστοιχίσεις

local1_aka_title(id, movie_id, title, imdb_index, production_year, phonetic_code, season_nr, note, md5sum) :- **aka_title**(id, movie_id, title, imdb_index, kind_id, production_year, phonetic_code, episode_of_id, season_nr, episode_nr, note, md5sum) production_year < 1970

local2_aka_title(id, movie_id, title, imdb_index, production_year, phonetic_code, season_nr, note, md5sum) :- **aka_title**(id, movie_id, title, imdb_index, kind_id, production_year, phonetic_code, episode_of_id, season_nr, episode_nr, note, md5sum) production_year > 1970

local1_char_name(id, name, surname_pcode) :- **char_name**(id, name, imdb_index, imdb_id, name_pcode_nf, surname_pcode, md5sum)

local2_char_name(id, imdb_index, imdb_id) :- **char_name**(id, name, imdb_index, imdb_id, name_pcode_nf, surname_pcode, md5sum)

local1_company_name(id, name) :- **company_name**(id, name, country_code, imdb_id, name_pcode_nf, name_pcode_sf, md5sum)

local2_company_name(id, name) :- **company_name**(id, name, '[us]', imdb_id, name_pcode_nf, name_pcode_sf, md5sum)

local1_movie_companies(id , movie_id, note) :- **movie_companies**(id, movie_id, company_id, company_type_id, note)

local2_movie_companies(id , company_id, company_type_id) :- **movie_companies**(id, movie_id, company_id, company_type_id, note)

local_cast_info(movie_id, person_role_id, nr_order, role) :- **cast_info**(id, person_id, movie_id, person_role_id, note, nr_order, role_id) **role_type**(role_id, role)

local_company_type(id, kind) :- **company_type**(id, kind)

local_comp_cast_type(id, kind) :- **comp_cast_type**(id, kind)

local_complete_cast(id, movie_id, subject_id) :- **complete_cast**(id, movie_id, subject_id, status_id)

local_keyword(id, keyword_info) :- **keyword**(id, keyword_info, phonetic_code)

local_kind_type(id, kind) :- **kind_type**(id, kind)

local_link_type(id, link) :- **link_type**(id, link)

local_movie_info(id, movie_id, info, info_type) :- **movie_info**(id, movie_id, info_type_id, info, note) **info_type**(info_type_id, info_type)

local_movie_info_idx(id, movie_id, info, info_type) :- **movie_info_idx**(id, movie_id, info_type_id, info, note) **info_type**(info_type_id, info_type)

local_movie_keyword(id, movie_id, keyword_id) :- **movie_keyword**(id, movie_id, keyword_id)

local_movie_link(id, movie_id, linked_movie_id, link_type_id) :- **movie_link**(id, movie_id, inked_movie_id, link_type_id)

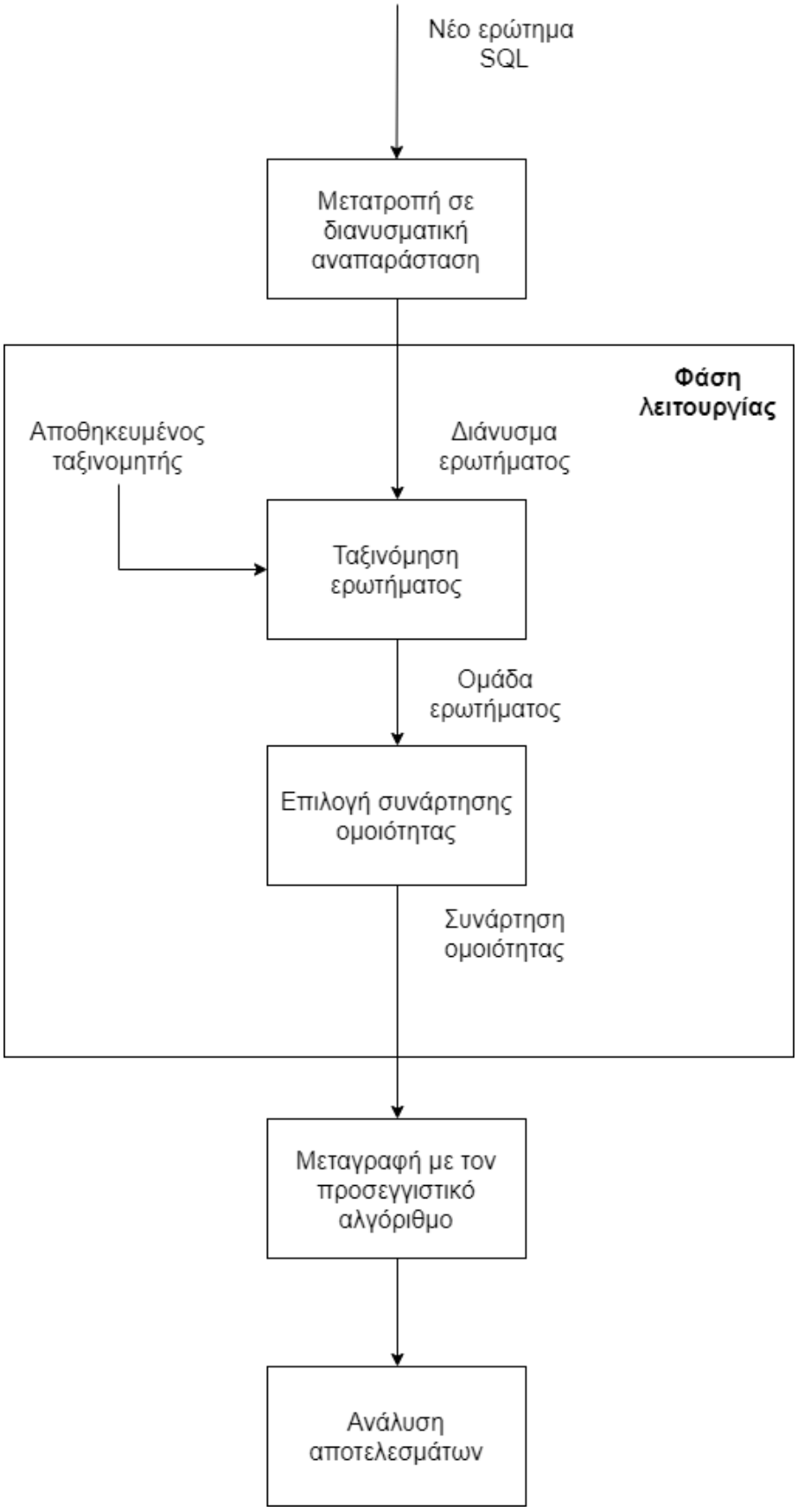
local_name(person_id, name, name_pcode_cf, surname_pcode, md5sum, other_name) :- **name**(person_id, name, imdb_index, imdb_id, gender, name_pcode_cf, name_pcode_nf,

surname_pcode, md5sum) **aka_name**(id, person_id, other_name, aka_imdb_index, aka_name_pcode_cf, aka_name_pcode_nf, aka_surname_pcode, aka_md5sum)

local_person_info(id, person_id, note, info_type) :- **person_info**(id, person_id, info_type_id, info, note) **info_type**(info_type_id, info_type)

local_title(id, title, imdb_index, kind_id, production_year, episode_of_id, season_nr, episode_nr) :- **title**(id, title, imdb_index, kind_id, production_year, imdb_id, phonetic_code, episode_of_id, season_nr, episode_nr, series_years, ,md5sum)

Τα αποτελέσματα της εξερεύνησης αυτής παρουσιάζονται αναλυτικά στο επόμενο κεφάλαιο.



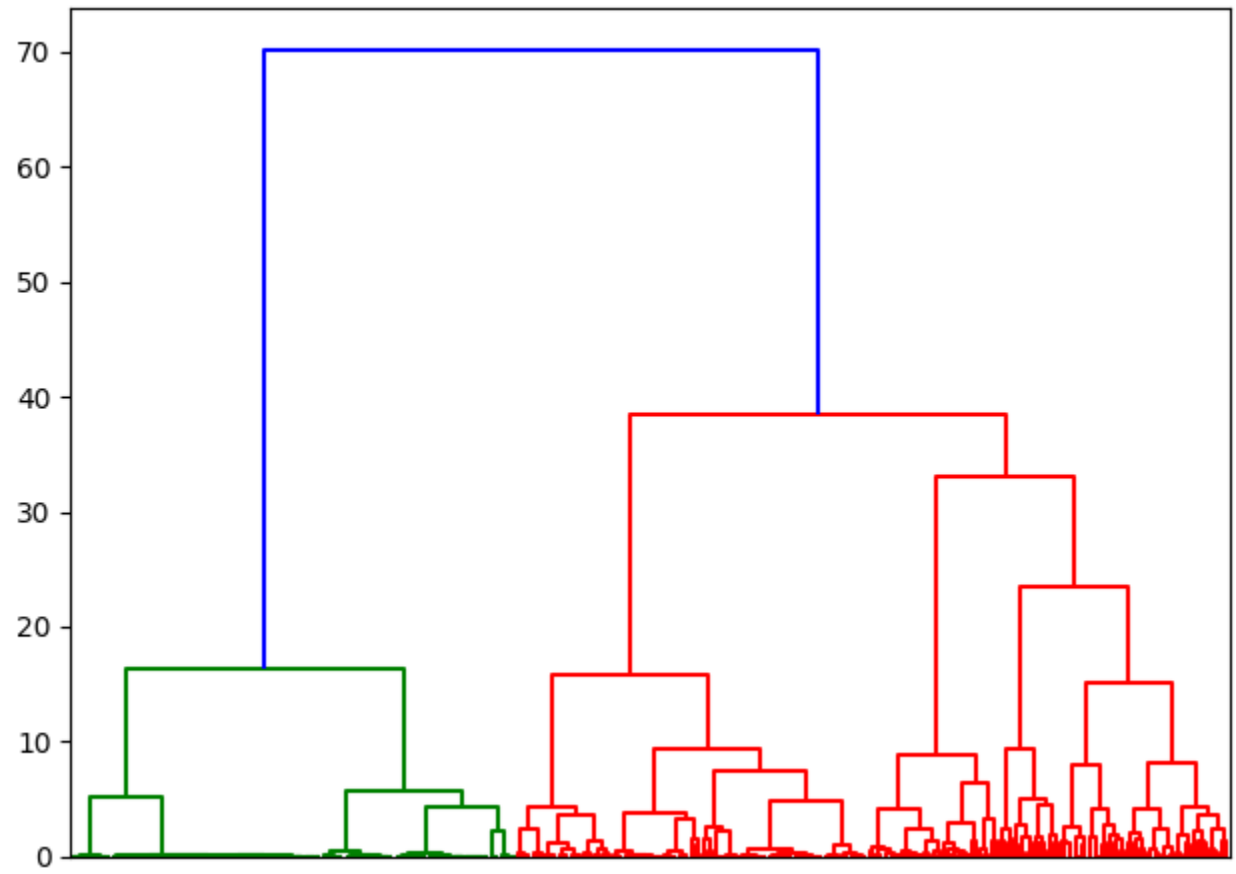
Σχήμα 3.8: Διαδικασία μεταγραφής

4. Αποτελέσματα και Σχολιασμός

Έχοντας ολοκληρώσει την περιγραφή τόσο του σχεδιασμού όσο και της υλοποίησης του λογισμικού αλλά και της πειραματικής διαδικασίας ως σύνολο, στο κεφάλαιο αυτό θα παρουσιάσουμε και θα σχολιάσουμε τα αποτελέσματα όλων των επιμέρους φάσεων. Θα ξεκινήσουμε με τα αποτελέσματα της ομαδοποίησης των ερωτημάτων, στη συνέχεια θα παρουσιάσουμε τις αποδόσεις των αλγορίθμων ταξινόμησης και στο τέλος θα παρουσιάσουμε τα αποτελέσματα των πειραμάτων της μεταγραφής.

4.1 Αποτελέσματα Ομαδοποίησης

Η ομαδοποίηση των ερωτημάτων κατά την δομική τους ομοιότητα μέσω μηχανικής μάθησης αποτέλεσε έναν από τους βασικούς στόχους της εργασίας. Όπως ήδη αναφέρθηκε στο κεφάλαιο 3, για την ομαδοποίηση των ερωτημάτων ακολουθήθηκαν δύο προσεγγίσεις. Η πρώτη προσέγγιση διαχειρίστηκε τα διανύσματα των ερωτημάτων όπως αυτά παράχθηκαν από τον μετατροπέα χωρίς κάποια προεπεξεργασία. Το πρώτο πέρασμα του ιεραρχικού αλγορίθμου μας έδωσε το δενδροδιάγραμμα του σχήματος 4.1.



Σχήμα 4.1: Δενδροδιάγραμμα 1ης προσέγγισης

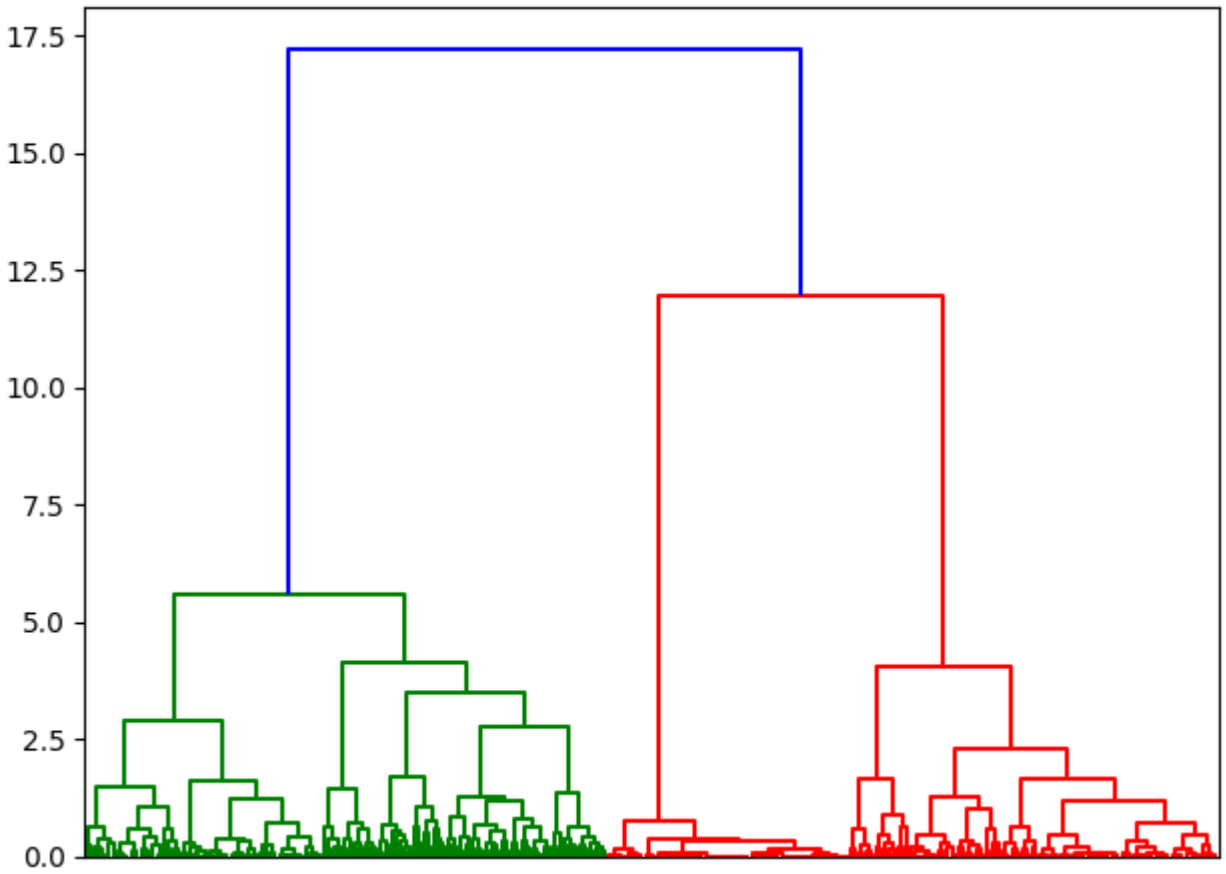
Λόγω του μεγάλου αριθμού χαρακτηριστικών θέλαμε όσο το δυνατόν περισσότερες ομάδες στο βαθμό όμως που η απόσταση μεταξύ των ομάδων στο δενδροδιάγραμμα είναι ευδιάκριτη. Η επιλογή του αριθμού έγινε με το κριτήριο της οριζόντιας γραμμής, όπου οι ομάδες είναι τόσες όσες κάθετες γραμμές που τέμνει η οριζόντια γραμμή στο όριο που έχουμε επιλέξει. Στην συγκεκριμένη περίπτωση επιλέχθηκε ο αριθμός 8. Μετά την εφαρμογή του ιεραρχικού αλγορίθμου για 8 ομάδες, μελετήσαμε τα χαρακτηριστικά των ερωτημάτων που συμμετείχαν σε κάθε ομάδα. Στον παρακάτω πίνακα παρουσιάζονται τα κυριότερα χαρακτηριστικά των ερωτημάτων που περιέχονταν σε κάθε ομάδα.

Αριθμός Ομάδας	Περιεχόμενα
0	Περιέχει ερωτήματα τύπου star, chain και συνδυασμού αυτών με 3 ή 4 συνδέσεις (joins) και 0 – 2 περιορισμούς τιμών ή σύγκρισης.
1	Περιέχει ερωτήματα συνδυασμού των τύπων star και chain με 5 – 11 συνδέσεις (joins) και 3 – 6 περιορισμούς τιμών ή σύγκρισης.
2	Περιέχει ερωτήματα τύπου chain με 1 ή 2 συνδέσεις (joins) και 0 ή 1 περιορισμούς τιμών ή σύγκρισης.
3	Περιέχει ερωτήματα τύπου star, chain και συνδυασμού αυτών με 7 - 11 συνδέσεις (joins) και 0 – 2 περιορισμούς τιμών ή σύγκρισης.
4	Περιέχει ερωτήματα τύπου star, chain και συνδυασμού αυτών με 4 - 7 συνδέσεις (joins) και 1 - 5 περιορισμούς τιμών ή σύγκρισης.
5	Περιέχει ερωτήματα χωρίς συνδέσεις (joins) και με 0 ή 1 περιορισμούς τιμών ή σύγκρισης.
6	Περιέχει ερωτήματα τύπου star, chain και συνδυασμού αυτών με 4 - 6 συνδέσεις (joins) και 0 περιορισμούς τιμών ή σύγκρισης.
7	Περιέχει ερωτήματα τύπου star, chain και συνδυασμού αυτών με 2 - 5 συνδέσεις (joins) και 3 - 5 περιορισμούς τιμών ή σύγκρισης.

Από τα αποτελέσματα αυτά είναι εύκολο να παρατηρήσει κανείς ότι δεν συμμετείχαν όλα τα χαρακτηριστικά στη διαμόρφωση των ομάδων. Στις περισσότερες ομάδες το είδος των συνδέσεων

(chain ή join) δεν επηρέασε τη συμμετοχή ενός ερωτήματος στην ομάδα και στις λίγες ομάδες που βλέπουμε μόνο έναν τύπο συνδέσεων είναι κυρίως λόγω του ότι δεν υπήρχαν ερωτήματα με τον αριθμό συνδέσεων της ομάδας. Ένα τέτοιο παράδειγμα είναι η ομάδα 2 όπου δεν θα μπορούσαν να υπάρχουν ερωτήματα τύπου star αφού απαιτούνται τουλάχιστον 3 συνδέσεις για να έχω πιθανώς star σύνδεση. Επίσης ούτε το ποσοστό των γνωρισμάτων επιλογής συμμετείχε στη διαμόρφωση των ομάδων. Είναι ξεκάθαρο πως ο αριθμός των συνδέσεων και ο αριθμός των περιορισμών ήταν τα χαρακτηριστικά που κυριάρχησαν. Αν κανείς λάβει υπόψιν του τον τρόπο που αναπαρίστανται τα ερωτήματα και το ότι η ομαδοποίηση βασίζεται στην μέτρηση της απόστασης των διανυσμάτων, ο τρόπος που διαμορφώθηκαν οι ομάδες είναι λογικός. Με εξαίρεση τους αριθμούς των συνδέσεων και των περιορισμών, τα υπόλοιπα χαρακτηριστικά είναι ποσοστά που κυμαίνονται μεταξύ 0 και 1. Αντίθετα οι συνδέσεις και οι περιορισμοί είναι ακέραιοι αριθμοί. Συνεπώς αποστάσεις των διανυσμάτων επηρεάζονται πρακτικά μονάχα από την απόσταση των ακεραίων.

Λόγω της αδυναμίας να χρησιμοποιηθούν όλα τα χαρακτηριστικά για την ομαδοποίηση με την πρώτη προσέγγιση, επιχειρήθηκε μια δεύτερη προσέγγιση που βασίστηκε στην προεπεξεργασία των διανυσμάτων. Όπως αναφέραμε οι διαφορές στην κλίμακα μέτρησης των χαρακτηριστικών οδήγησε στα προβλήματα της πρώτης προσέγγισης. Έτσι στη δεύτερη προσέγγιση τα διανύσματα πριν ομαδοποιηθούν προεπεξεργάζονται περνώντας από μια διαδικασία κλιμάκωσης των χαρακτηριστικών τους. Το δενδροδιάγραμμα που προέκυψε με αυτή την προσέγγιση παρουσιάζεται στον σχήμα δ.δ.



Σχήμα 4.2: Δενδροδιάγραμμα 2ης προσέγγισης

Ο αριθμός των ομάδων που επιλέχθηκε βάση του δενδροδιαγράμματος ήταν 10. Στον παρακάτω πίνακα παρουσιάζονται τα κυριότερα χαρακτηριστικά των ερωτημάτων που περιέχονταν σε κάθε μία από τις 10 ομάδες

Αριθμός Ομάδας	Περιεχόμενα
0	Περιέχει ερωτήματα συνδυασμού των τύπων star και chain σε ποσοστά μεταξύ 35% - 65% και 80% - 20%, μεγάλο αριθμό συνδέσεων (> 50%) και περιορισμών (> 40%) αλλά σχετικά μικρού ποσοστού γνωρισμάτων επιλογής (< 50%).
1	Περιέχει ερωτήματα τύπου star με μικρό αριθμό συνδέσεων (< 50%) και χωρίς περιορισμούς και με μικρό ποσοστό γνωρισμάτων επιλογής (< 50%).
2	Περιέχει ερωτήματα τύπου chain με μέτριο αριθμό συνδέσεων (30% < joins < 70%) και με μεγάλο ποσοστό γνωρισμάτων επιλογής (> 60%). Οι περιορισμοί υπήρχαν σε μικρά και μεγάλα ποσοστά.
3	Περιέχει ερωτήματα τύπου star με μέτριο προς μικρό αριθμό συνδέσεων (< 60%) και μεγάλη διακύμανση στους περιορισμούς και με μικρό ποσοστό γνωρισμάτων επιλογής (< 40%). Σε ερωτήματα με πολύ μικρο ποσοστό γνωρισμάτων επιλογής παρατηρήθηκε μεγάλος αριθμός περιορισμών.
4	Περιέχει ερωτήματα τύπου star με μεγάλο αριθμό συνδέσεων (> 60%) και μεγάλη διακύμανση στους περιορισμούς, με μεγάλο ποσοστό γνωρισμάτων επιλογής (> 50%). Περιέχονται και ερωτήματα συνδυασμού star και join τύπων στα οποία οι star συνδέσεις έχουν ποσοστό 75%, ο αριθμός συνδέσεων είναι πολύ μεγάλος (> 80%) και το ποσοστό των γνωρισμάτων επιλογής πολύ μεγάλο (> 75%).
5	Περιέχει ερωτήματα χωρίς συνδέσεις και με μικρό αριθμό ή καθόλου περιορισμούς.
6	Περιέχει ερωτήματα τύπου star με μικρό αριθμό συνδέσεων (< 20%) και μεγάλη διακύμανση στους

	περιορισμούς, καθώς και με μικρό ποσοστό γνωρισμάτων επιλογής (< 40%).
7	Περιέχει ερωτήματα συνδυασμού των τύπων star και chain σε ποσοστά μεταξύ 65% - 35% και 60% - 40%, με μεγάλο αριθμό συνδέσεων (> 50%) και μικρό αριθμό περιορισμών (< 20%). Το ποσοστό γνωρισμάτων επιλογής έχει μεγάλη διακύμανση.
8	Περιέχει ερωτήματα τύπου star με μεγάλο αριθμό συνδέσεων (> 60%) και μεγάλη διακύμανση στους περιορισμούς, καθώς και με μικρο ποσοστό γνωρισμάτων επιλογής (< 50%). Περιέχονται και ερωτήματα συνδυασμού star και join τύπων στα οποία οι star συνδέσεις έχουν ποσοστό 70%, ο αριθμός συνδέσεων είναι μεγάλος (> 60%) και το ποσοστό των γνωρισμάτων επιλογής μικρό (< 40%).
9	Περιέχει ερωτήματα τύπου chain με διακύμανση στον αριθμό συνδέσεων (> 20%) και με μικρό ποσοστό γνωρισμάτων επιλογής (< 50%). Οι περιορισμοί είχαν πολύ μικρό ποσοστό (< 20%).

Όπως είναι φανερό η δεύτερη προσέγγιση δίνει καλύτερα αποτελέσματα καθώς όλα τα χαρακτηριστικά συμμετέχουν στη διαμόρφωση των ομάδων. Κάθε ομάδα μπορει να χαρακτηρίζεται από διαφορετικά χαρακτηριστικά όμως όλα τα χαρακτηριστικά συμμετέχουν στην διαμόρφωση τουλάχιστον μίας ομάδας. Το ποσοστό συμμετοχής των πινάκων είναι το ίδιο με τον αριθμό των πινάκων μετά την κλιμάκωση οπότε και δεν εστιάσαμε σε αυτό. Πιθανότατα θα μπορούσε και να παραληφθεί από την διανυσματική αναπαράσταση.

Με βάση τα αποτελέσματα μας είναι ασφαλές να θεωρήσουμε πως η υπόθεση 3, όπως αυτή διατυπώθηκε στην παράγραφο 3.1.2, επαληθεύεται και μπορεί να χρησιμοποιηθεί εκτενώς τόσο στην παρούσα εργασία, όσο και σε μελλοντικές.

4.2 Αποτελέσματα Ταξινόμησης

Όπως αναφέρθηκε στο προηγούμενο κεφάλαιο η επιλογή του κατάλληλου ταξινομητή είναι συνήθως μια πειραματική διαδικασία. Έτσι και στην παρούσα εργασία δοκιμάστηκαν πέντε τεχνικές ταξινόμησης μηχανικής μάθησης με την εκπαίδευση και τη δοκιμή τους να περιγράφεται στην παράγραφο 3.3. Τα αποτελέσματα αυτής της πειραματικής φάσης παρουσιάζονται στον παρακάτω πίνακα.

Αλγόριθμος/Τεχνική	Precision	Recall	F1-score
Gaussian Naive Bayes	0.93	0.91	0.91
k-Nearest Neighbours	0.82	0.78	0.77
Decision Trees	0.96	0.96	0.96
LDA	0.96	0.96	0.96
SVM	0.80	0.78	0.78

Όπως φαίνεται από τα αποτελέσματα των πειραμάτων την καλύτερη απόδοση την είχαν οι ταξινομητές Δέντρων Αποφάσεων και Γραμμικής Ανάλυσης Διακρίσεων, ενώ την χειρότεροι οι ταξινομητές Μηχανών Διανυσμάτων Υποστήριξης και κ-Κοντινότερων Γειτόνων. Βάσει των αποτελεσμάτων ο ταξινομητής που επιλέχθηκε ήταν αυτός των Δέντρων Αποφάσεων.

4.3 Αποτελέσματα Μεταγραφής

Όπως αναφέραμε για να ελέγξουμε τις βασικές μας υποθέσεις, όπως αυτές διατυπώθηκαν στην παράγραφο 3.1.2, επιλέξαμε 10 ερωτήματα από κάθε μια από τις κατηγορίες που προέκυψαν από το στάδιο της ομαδοποίησης τα οποία και μεταγράψαμε μέσω του συστήματος προσεγγιστικής μεταγραφής στο πειραματικό πλαίσιο που περιγράψαμε στην παράγραφο 3.4. Τα αποτελέσματα για κάθε κατηγορία παρουσιάζονται στους παρακάτω πίνακες. Οι συναρτήσεις 1, 2 , 3 αντιστοιχούν κατά σειρά στις συναρτήσεις ομοιότητας ερωτημάτων όπως αυτές συζητιούνται στην παράγραφο 2.7 και στην εργασία [34].

Κατηγορία 0	Συνάρτηση 1	Συνάρτηση 2	Συνάρτηση 3
Ερώτημα 1	72.540985	61.793373	-91.67505
Ερώτημα 2	62.96296	48.988094	13.541667
Ερώτημα 3	70.410987	62.9912	3.145741
Ερώτημα 4	57.9646	39.873947	2.028245
Ερώτημα 5	47.945206	30.220673	0.6696407
Ερώτημα 6	81.04575	76.111115	76.190475
Ερώτημα 7	69.547325	53.809525	-342.77344
Ερώτημα 8	78.169014	67.07727	-15.684705
Ερώτημα 9	62.745102	49.20463	124.65278
Ερώτημα 10	71.42857	54.213833	1.9326733

Κατηγορία 1	Συνάρτηση 1	Συνάρτηση 2	Συνάρτηση 3
Ερώτημα 1	53.465347	29.117647	3.6580882
Ερώτημα 2	70.21276	67.9	67.99375
Ερώτημα 3	54.268295	35.654762	3.3478656
Ερώτημα 4	77.41935	65.833336	66.614586
Ερώτημα 5	52.47525	39.411766	13.216911
Ερώτημα 6	71.14094	67.86667	67.99922
Ερώτημα 7	56.37584	37.266666	3.4996092
Ερώτημα 8	55.05618	35.76923	3.3646333
Ερώτημα 9	77.27273	64.22619	18.749899
Ερώτημα 10	61.03896	42.074074	3.7498374

Κατηγορία 2	Συνάρτηση 1	Συνάρτηση 2	Συνάρτηση 3
Ερώτημα 1	61.23348	39.96795	1.0015557
Ερώτημα 2	60.948902	40.059288	0.24626297
Ερώτημα 3	57.34767	37.833656	0.23271194
Ερώτημα 4	61.904762	38.764706	0.25275204
Ερώτημα 5	57.67442	46.714287	13.194437
Ερώτημα 6	56.73759	43.809525	3.385416
Ερώτημα 7	57.714283	54.70968	54.83619
Ερώτημα 8	55.021835	44.634144	13.414158
Ερώτημα 9	63.761467	37.556393	1.0278963
Ερώτημα 10	59.523808	43.36098	7.346081

Κατηγορία 3	Συνάρτηση 1	Συνάρτηση 2	Συνάρτηση 3
Ερώτημα 1	53.91705	28.310812	0.2216731
Ερώτημα 2	72.935776	70.97744	71.052505
Ερώτημα 3	49.152542	25.775862	0.20200531
Ερώτημα 4	68.786125	66.55073	66.6658
Ερώτημα 5	56.33188	36.068798	0.22170588
Ερώτημα 6	70.83333	67.915794	67.99792
Ερώτημα 7	61.25	38.94872	1.2820305
Ερώτημα 8	73.30317	70.98472	71.05259
Ερώτημα 9	56.33188	36.068798	0.22170547
Ερώτημα 10	68.2353	66.53333	66.66406

Κατηγορία 4	Συνάρτηση 1	Συνάρτηση 2	Συνάρτηση 3
Ερώτημα 1	73.40425	71.9375	71.99997
Ερώτημα 2	69.78193	68.35282	68.42104
Ερώτημα 3	71.38643	69.4376	69.49152
Ερώτημα 4	69.03915	67.26757	67.346924
Ερώτημα 5	71.05263	69.480675	69.56521
Ερώτημα 6	73.95833	71.947365	72
Ερώτημα 7	71.13096	68.372894	39.50369
Ερώτημα 8	72.126434	69.446754	48.681057
Ερώτημα 9	73.68421	71.94286	71.99999
Ερώτημα 10	71.75324	68.57565	-191.49799

Κατηγορία 5	Συνάρτηση 1	Συνάρτηση 2	Συνάρτηση 3
Ερώτημα 1	60.000004	2.15E+09	60.000004
Ερώτημα 2	40	-Infinity	40
Ερώτημα 3	73.333336	-Infinity	73.333336
Ερώτημα 4	75	2.15E+09	75
Ερώτημα 5	65	-Infinity	65
Ερώτημα 6	78.57143	78.66667	78.66667
Ερώτημα 7	53.571426	40	40
Ερώτημα 8	NAN	NAN	NAN
Ερώτημα 9	46.153847	43.333332	43.333332
Ερώτημα 10	94.44444	97.77778	97.77778

Κατηγορία 6	Συνάρτηση 1	Συνάρτηση 2	Συνάρτηση 3
Ερώτημα 1	64.210526	50.228573	19.047413
Ερώτημα 2	48.148148	42.5974	42.79762
Ερώτημα 3	48.85496	25.869566	3.2472827
Ερώτημα 4	87.80488	83.33333	83.33333
Ερώτημα 5	42.46575	38.076923	38.26923
Ερώτημα 6	69.44444	49.6875	49.98047
Ερώτημα 7	59.090908	33.333336	12.5
Ερώτημα 8	41.17647	21.071428	2.65625
Ερώτημα 9	65.11628	57.14286	57.14286
Ερώτημα 10	65.59139	64.411766	64.55882

Κατηγορία 7	Συνάρτηση 1	Συνάρτηση 2	Συνάρτηση 3
Ερώτημα 1	59.45946	39.916668	0.23437475
Ερώτημα 2	69.230774	53.27778	4.1666646
Ερώτημα 3	57.196968	41.26033	0.8877834
Ερώτημα 4	59.67742	44.037037	3.5416615
Ερώτημα 5	58.285713	47.367348	13.392849
Ερώτημα 6	62.92517	46.462315	175.78125
Ερώτημα 7	60.317463	45.008545	3.541665
Ερώτημα 8	75.333336	65.391304	18.478247
Ερώτημα 9	60.975613	47.4718	-89.84375
Ερώτημα 10	70.68965	49.930553	1.1718655

Κατηγορία 8	Συνάρτηση 1	Συνάρτηση 2	Συνάρτηση 3
Ερώτημα 1	74.149666	64.565216	18.478207
Ερώτημα 2	78.36257	63.292183	5.0925813
Ερώτημα 3	75.675674	62.07407	4.9999795
Ερώτημα 4	78.21782	65.64103	5.2884383
Ερώτημα 5	79.439255	68.351654	5.2884603
Ερώτημα 6	81.456955	70.70393	5.3571424
Ερώτημα 7	74.178406	63.596695	90.40179
Ερώτημα 8	77.439026	73.03168	73.07692
Ερώτημα 9	70.55215	56.736843	4.4999967
Ερώτημα 10	76.55502	68.80165	18.939392

Κατηγορία 9	Συνάρτηση 1	Συνάρτηση 2	Συνάρτηση 3
Ερώτημα 1	48.97959	16.5	4.9990234
Ερώτημα 2	74.19355	59.22222	18.517796
Ερώτημα 3	58.024693	42.839508	3.4721713
Ερώτημα 4	45.20548	23.492064	0.21647136
Ερώτημα 5	76.190475	61.122448	17.85697
Ερώτημα 6	100	100	100
Ερώτημα 7	66.42857	62.416668	62.49837
Ερώτημα 8	69.230774	58.291668	16.666647
Ερώτημα 9	72.86822	66.66667	66.66667
Ερώτημα 10	70.90909	69.333336	1.5624746

Το γενικό συμπέρασμα από τα παραπάνω αποτελέσματα είναι εύκολα αντιληπτό. Οι υποθέσεις 1 και 2 πάνω στις οποίες στηρίχθηκε και ο βασικός στόχος για την αυτοματοποίηση της διαδικασίας επιλογής της συνάρτησης ομοιότητας για το σύστημα προσεγγιστικής μεταγραφής δεν επιβεβαιώνονται. Οι υποθέσεις 1 και 2 υπέθεταν πως για διαφορετικές κατηγορίες ερωτημάτων το σύστημα μεταγραφής θα μας έδινε το καλύτερο αποτέλεσμα κάνοντας χρήση διαφορετικής συνάρτησης ομοιότητας. Όμως για όλες τις κατηγορίες, τα αποτελέσματα δείχνουν ότι η συνάρτηση 1 είναι η επικρατέστερη. Υπάρχουν μεν κάποια ελάχιστα παραδείγματα για τα οποία η συνάρτηση 3 επικρατεί, όμως αυτά αποτελούν την εξαίρεση. Πρέπει μάλιστα να σχολιάσουμε πως τα αποτελέσματα της συνάρτησης 3 σε ορισμένες περιπτώσεις παρουσιάζουν τιμές οι οποίες δεν είναι επιτρεπτές. Οι υποθέσεις 1 και 2 υπέθεταν πως για διαφορετικές κατηγορίες ερωτημάτων το σύστημα μεταγραφής θα μας έδινε το καλύτερο αποτέλεσμα κάνοντας χρήση διαφορετικής συνάρτησης ομοιότητας.

Οι λίγες εξαιρέσεις που υπάρχουν στα αποτελέσματα μας φανερώνουν πως οι υποθέσεις 1 και 2 μπορεί μεν να μην επαληθεύονται με τις συνθήκες των πειραμάτων που έγιναν, όμως με διαφορετικές πειραματικές συνθήκες τα πράγματα είναι πιθανό να είναι διαφορετικά. Άλλωστε υπήρξε ως σχεδιαστική επιλογή να αγνοήσουμε τις υπόλοιπες παραμέτρους του προβλήματος και να εστιάσουμε μονάχα στα ερωτήματα. Με αυτό τον τρόπο αφήσαμε εκτός της πειραματικής διαδικασίας έναν παράγοντα που παίζει καθοριστικό ρόλο στη μεταγραφή, τις αντιστοιχήσεις. Οι αντιστοιχίσεις είναι ο νούμερο ένα παράγοντας που επηρεάζει την ποιότητα της μεταγραφής και αν και δεν έχουν άμεση σχέση με τις συναρτήσεις ομοιότητας που χρησιμοποιούνται, τα ερωτήματα που μεταγράφονται και αξιολογούνται ως προς την ομοιότητα τους από την πειραματική μας διαδικασία είναι άμεσα συνδεδεμένα με αυτές. Συνεπώς η έλλειψη πολλών διαφορετικών αντιστοιχίσεων επηρεάζει όπως αποδείχθηκε τα αποτελέσματα μας και πιθανώς αποτελεί και τον κύριο λόγο μη επιβεβαίωσης των υποθέσεων μας.

5. Επίλογος

5.1 Συμπεράσματα

Ολοκληρώνοντας την παρούσα διπλωματική εργασία, επιχειρούμε μια ανασκόπηση των επιτευγμάτων και των συμπερασμάτων της. Υπενθυμίζουμε πως η παρούσα εργασία είναι ένα πρώτο βήμα έρευνας της ομοιότητας συναρτήσεων με τη βοήθεια της μηχανικής μάθησης. Οι εργασίες που ασχολούνται με την ομοιότητα ερωτημάτων είναι λίγες και το συγκεκριμένο αντικείμενο δεν έχει απασχολήσει ιδιαίτερα έως τώρα την επιστημονική κοινότητα. Ειδικότερα, η παρούσα εργασία προσέγγισε το αντικείμενο με μια διαφορετική οπτική που δεν είχε επιχειρηθεί έως τώρα.

Συνοψίζοντας τα συμπεράσματα που προέκυψαν από τα αποτελέσματα, συμπεραίνουμε ότι:

- Οι υποθέσεις 1 και 2 της παραγράφου 3.1.2 δεν ικανοποιούνται με της παρούσες πειραματικές συνθήκες και επιλογές. Παρόλα αυτά, ορισμένα αποτελέσματα είναι ενθαρρυντικά και πιθανώς εισάγοντας κι άλλες παραμέτρους στην πειραματική διαδικασία οι υποθέσεις μας να ικανοποιούνται.
- Αντίθετα η υπόθεση 3 θεωρούμε πως ικανοποιείται σε έναν πολύ μεγάλο βαθμό. Οι ομάδες που προέκυψαν κατά την πειραματική διαδικασία περιείχαν ερωτήματα με κοινά δομικά χαρακτηριστικά και στην δημιουργία των ομάδων συμμετείχαν όλα τα χαρακτηριστικά της αναπαράστασης που ορίσαμε.

Κρίνοντας τις συνεισφορές της εργασίας, μπορούμε να υποστηρίξουμε τα εξής:

- Αρχικά η εργασία προσφέρει το σχεδιασμό και την υλοποίηση ενός λογισμικού αυτόματης παραγωγής ερωτημάτων που δύναται μελλοντικά να εμπλουτιστεί ώστε να εξυπηρετήσει ανάγκες και άλλων προβλημάτων.
- Ερευνά το πρόβλημα της ομοιότητας ερωτημάτων από την καθαρά δομική τους οπτική και παρέχει για αυτό το σκοπό μια αναπαράσταση των ερωτημάτων και έναν ορισμό της δομικής ομοιότητας που δεν υπήρχε προηγουμένως στην βιβλιογραφία.
- Εισάγει στην μελέτη της ομοιότητας ερωτημάτων τεχνικές μηχανικής μάθησης και αποδεικνύει πειραματικά την συνεισφορά τους στο πεδίο.
- Παρέχει μέσω μηχανικής μάθησης έναν τρόπο ομαδοποίησης και αντιμετώπισης των ερωτημάτων που δεν έχει ερευνηθεί και που μπορεί να χρησιμοποιηθεί σε πλήθος προβλημάτων.
- Ερευνά πειραματικά την απόδοση διαφορετικών αλγορίθμων ταξινόμησης για την δομική ταξινόμηση των ερωτημάτων.

- Μελέτα την αυτοματοποίηση της διαδικασίας επιλογής των συναρτήσεων ομοιότητας ερωτημάτων για ερωτήματα συγκεκριμένων δομικών χαρακτηριστικών στο πλαίσιο της προσεγγιστικής μεταγραφής και συμπεραίνει ότι το είδος των ερωτημάτων ως μοναδική παράμετρος δεν μπορεί να καθορίσει την επιλογή συνάρτησης.
- Τέλος, τα αποτελέσματα της παρούσας εργασίας αποδεικνύουν πως ο ορισμός της δομικής ομοιότητας που παρέχεται έχει λογική υπόσταση και καταφέρνει να εξυπηρετήσει τον σκοπό του.

Έχοντας όλα τα παραπάνω κατά νου, κρίνουμε ότι στο σύνολο τους τα αποτελέσματα της πειραματικής διαδικασίας ήταν ενθαρρυντικά και οι συνεισφορές της εργασίας σημαντικές ώστε μελλοντικά να μελετηθεί σε μεγαλύτερο βάθος αυτή η προσέγγιση. Μελλοντικές επεκτάσεις προτείνονται στην επόμενη παράγραφο.

5.2 Σημαντικές επεκτάσεις

Όπως έχει ήδη αναφερθεί, η παρούσα εργασία αποτέλεσε ένα πρώτο βήμα στην απόδοση μιας μορφής ομοιότητας ερωτημάτων που δεν είχε ερευνηθεί έως τώρα και στην ένταξη τεχνικών της μηχανικής μάθησης στη μεταγραφή ερωτημάτων. Ως εκ τούτου υπάρχει μεγάλο εύρος επεκτάσεων και πολλά περιθώρια βελτίωσης. Στην παράγραφο αυτή λοιπόν θα παρουσιάσουμε τις σπουδαιότερες και πιο ενδιαφέρουσες επεκτάσεις καθώς και μελλοντικά πλάνα βελτίωσης της παρούσας εργασίας.

Λαμβάνοντας υπ' όψιν τόσο τα αποτελέσματα της παρούσας εργασίας όσο και τις εργασίες [1] και [34], το επόμενο βήμα ερευνητικά είναι η εισαγωγή και άλλων παραμέτρων στην πειραματική διαδικασία. Όπως αναφέρθηκε στην θεωρία μας, η μεταγραφή ερωτημάτων, είτε ακριβής είτε προσεγγιστική, απαιτεί αντιστοιχίσεις μεταξύ των σχημάτων που εμπλέκονται. Στην παρούσα εργασία κατασκευάστηκε και χρησιμοποιήθηκε ένα μόνο σύνολο αντιστοιχίσεων. Έχοντας πλέον μια καλύτερη εικόνα για τον ρόλο που παίζει η δομή των ερωτημάτων στην μεταγραφή ερωτημάτων θα ήταν πολύ ενδιαφέρον να ελέγξουμε τη συμπεριφορά των ερωτημάτων και για αντιστοιχίσεις διαφόρων χαρακτηριστικών. Μια τέτοια επέκταση θα μπορούσε να δώσει μια γενικότερη εικόνα του προβλήματος και πιθανώς να οδηγήσει στην επίτευξη και του τελικού στόχου της παρούσας εργασίας.

Μια άλλη ιδιαίτερα ενδιαφέρουσα επέκταση είναι η μελέτη μεγαλύτερου εύρους ερωτημάτων. Στην παρούσα εργασία ελέγχθηκε η συμπεριφορά μόνο των SPJ ερωτημάτων. Όμως υπάρχουν ακόμα πολλές κατηγορίες ερωτημάτων που χρησιμοποιούνται κατά κόρον και συνεπώς είναι απαραίτητη η μελέτη και αυτών. Μια τέτοια κατηγορία ερωτημάτων είναι τα ερωτήματα που περιέχουν συναρτήσεις άθροισης (aggregation queries). Οι συναρτήσεις άθροισης χρησιμοποιούνται συχνά στην ανάλυση των δεδομένων και αποτελούν ένα ιδιαίτερο κομμάτι κάθε συστήματος διαχείρισης δεδομένων. Παραδείγματα συναρτήσεων άθροισης είναι η count, η min, η max κ.α. Ένα πρώτο βήμα για την εξέλιξη της παρούσας εργασίας θα ήταν η μελέτη αυτών των ερωτημάτων.

Σημαντική επέκταση της εργασίας είναι και η μελέτη και η εισαγωγή περισσότερων χαρακτηριστικών των ερωτημάτων. Με τον τρόπο που έχει ορισθεί στην παρούσα εργασία η ομοιότητα ερωτημάτων, ο αριθμός των χαρακτηριστικών που μπορούν να μελετηθούν δεν περιορίζεται και οι μόνοι περιορισμοί μπορεί να οφείλονται στην πολυπλοκότητα του προβλήματος

από την πλευρά της εφαρμογής της μηχανικής μάθησης. Για παράδειγμα, ένα χαρακτηριστικό που στοχεύουμε να εντάξουμε στην έρευνά μας στο μέλλον είναι η εκλεκτικότητα τόσο των γνωρισμάτων επιλογής όσο και των περιορισμών.

Πέρα από τις επεκτάσεις στις παραμέτρους του αλγορίθμου μεταγραφής και τα ερωτήματα που μελετώνται, ενδιαφέρον θα είχε να επεκταθεί η εργασία και από την οπτική της αξιολόγησης της ποιότητας της μεταγραφής. Το τελευταίο διάστημα διάφοροι τρόποι μέτρησης απώλειας πληροφορίας έχουν αρχίσει να χρησιμοποιούνται για την αξιολόγηση λύσεων στην διαχείριση δεδομένων, και ιδίως στην ιδιωτικότητα. Πιστεύουμε πως αυτές οι μετρικές μπορούν να εφαρμοστούν και στην παρούσα εργασία για την αξιολόγηση των αποτελεσμάτων των διαφόρων μεταγραφών. Μάλιστα θεωρούμε ότι θα μπορούσαν να χρησιμοποιηθούν ως συναρτήσεις αξιολόγησης από τις τεχνικές μηχανικής μάθησης για την αυτοματοποίηση ακόμα περισσότερο του συστήματος μεταγραφής και την πιο αποτελεσματική μελέτη των παραμέτρων του.

Συνεχίζοντας, μια άλλη πτυχή των συναρτήσεων ομοιότητας που χρησιμοποιούνται από τον προσεγγιστικό αλγόριθμο μεταγραφής ερωτημάτων της εργασίας [1] και παρουσιάζονται στην εργασία [34] είναι η βαρύτητα που δίνεται σε κάθε στοιχείο των ερωτημάτων που ελέγχονται. Θα ήταν ενδιαφέρον λοιπόν να επεκταθεί η παρούσα εργασία ώστε με χρήση μηχανικής μάθησης να βρεθούν τα κατάλληλα βάρη για την επικρατούσα συνάρτηση της κάθε κατηγορίας ερωτημάτων ώστε να βελτιστοποιείται η ποιότητα της μεταγραφής. Κατά ανάλογο τρόπο θα ήταν ενδιαφέρουσα η δημιουργία των κατηγοριών των ερωτημάτων μέσω τεχνικών ομαδοποίησης με βάρη, ώστε ανάλογα με την εφαρμογή να μπορεί να αποδοθεί και διαφορετικός βαθμός συμμετοχής για κάθε χαρακτηριστικό.

Τέλος, πολλές μικρές και πιο πρακτικές επεκτάσεις και βελτιώσεις είναι επίσης δυνατές. Η εργασία αυτή αποτελεί ένα πρώτο βήμα στο συγκεκριμένο θέμα έρευνας και αφήνει διάφορες παραμέτρους των εργαλείων που χρησιμοποιεί ανεξερεύνητες. Άλλες τεχνικές μηχανικής μάθησης μπορεί να έχουν καλύτερα αποτελέσματα ή/και διαφορετικές παράμετροι στις επιλεγέντες τεχνικές μπορεί να βελτιώσουν τα αποτελέσματα μας. Επίσης υπάρχει η δυνατότητα τεχνικών επεκτάσεων για την απαλοιφή περισσότερου ανθρώπινου φόρτου εργασία κατά την πειραματική διαδικασία καθώς το βάρος της παρούσας εργασίας εστιάστηκε περισσότερο στην εξερεύνηση των ιδεών και όχι στην αποτελεσματική υλοποίηση του χρησιμοποιούμενου υλικού.

6. Βιβλιογραφία

[1] V. Kantere, G.Orfanoudakis, A. Kementsietsidis and T. Sellis. Query Relaxation across Heterogeneous Data Sources. In CIKM, 2015.

[2] Pottinger, Rachel and Alon Y. Halevy. "MiniCon: A scalable algorithm for answering queries using views." The VLDB Journal 10 (2000): 182-198.

[3] Mitchell, T. (1997). Machine Learning. McGraw Hill. p. 2. ISBN 978-0-07-042807-2.

[4] N. Polyzotis, S. Roy, S. E. Whang, M. Zinkevich. Data Management Challenges in Production Machine Learning. In SIGMOD, 2017.

[5] A. Kumar, M. Boehm, J. Yang. Data Management in Machine Learning: Challenges, Techniques, and Systems. In SIGMOD, 2017

[6] X. L. Dong, T. Rekatsinas. Data Integration and Machine Learning: A Natural Synergy. In SIGMOD, 2018.

[7] L. Berti-Equille, A. Bonifati, T. Milo. Machine Learning to Data Management: A Round Trip. In ICDE, 2018.

[8] Distant supervision for relation extraction without labeled data. M. Mintz, S. Bills, R. Snow, D. Jurafsky. In Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP, pages 1003–1011, Suntec, Singapore, 2-7 August 2009.

[9] Using distant supervision to augment manually annotated data for relation extraction. P. Su, G. Li, C. Wu, K. Vijay-Shanker. In PLoS ONE, 2019, 14(7): e0216913. https://doi.org/10.1371/journal.Pone.0216913

[10] http://deepdive.stanford.edu/distant_supervision

[11] Distant Supervision for Emotion Classification with Discrete Binary Values . J. Suttles, N. Ide. In: Gelbukh A. (eds) Computational Linguistics and Intelligent Text Processing. CICLing 2013. Lecture Notes in Computer Science, vol 7817. Springer, Berlin, Heidelberg.

[12] Auto-Dialabel: Labeling Dialogue Data with Unsupervised Learning. C. Shi1, Q. Chen, L. Sha, S. Li, X. Sun, H. Wang, L. Zhang. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 684–689, Brussels, Belgium, October 31 - November 4, 2018.

[13] Germain Forestier, Cédric Wemmert. Semi-supervised learning using multiple clusterings with limited labeled data. Information Sciences, Elsevier, 2016, 361–362, pp.48-65. ff10.1016/j.ins.2016.04.040ff. ffhal-01355189v2ff

[14] A semi-supervised cluster-and-label approach for utterance classification. A. Albalate, A. Suchindranath, D. Suendermann, W. Minker. In INTERSPEECH, 2010.

[15] A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications. R.Schollmeier. In Proceedings of the First International Conference on Peer-to-Peer Computing, 2001

[16] https://docs.oracle.com/cd/B10500_01/server.920/a96520/concept.htm#50413

[17] Star Schema. K. Morfonios, Y. Ioannidis. In: LIU L., ÖZSU M.T. (eds) Encyclopedia of Database Systems. Springer, Boston, MA, 2009. https://doi.org/10.1007/978-0-387-39940-9_888

[18] Snowflake Schema. K. Morfonios, Y. Ioannidis. In: LIU L., ÖZSU M.T. (eds) Encyclopedia of Database Systems. Springer, Boston, MA, 2009. https://doi.org/10.1007/978-0-387-39940-9_888

[19] Answering Queries Using Views: Second Edition. F. Afrati, R. Chirkova, H. V. Jagadish, 2019. doi: 10.2200/S00884ED2V01Y201811DTM054

[20] Answering queries using views: A survey. A. Y. Halevy. In VLDB Journal, 2001.

[21] Optimizing Joins in a Map-Reduce Environment. F. Afrati, J. Ullman. In EDBT, 2010

[22] How Good Are Query Optimizers, Really? V. Leis, A. Gubichev, A. Mirchev, P. Boncz, A. Kemper, T. Neumann. In Proceedings of the VLDB Endowment, Vol. 9, No. 3, 2015.

[23] Join-order optimization with cartesian products. B. Vance, 1998, dissertation https://scholararchive.ohsu.edu/concern/etds/3484zg90n

[24] Heuristic and Randomised Optimisation Techniques in Object-Oriented Database Systems. M. Steinbrunn. PhD thesis, dissertation, UniversitaÈt Passau, 1996.

[25] https://www.cockroachlabs.com/blog/join-ordering-pt1/

[26] Conjunctive Query. V. Tannen. In: LIU L., ÖZSU M.T. (eds) Encyclopedia of Database Systems. Springer, Boston, MA, 2009. https://doi.org/10.1007/978-0-387-39940-9

[27] Join Processing in Relational Databases. P. Mishra, M. H. Eich. In ACM Computmg Surveys, Vol. 24, No 1, March 1992.

[28] https://developers.google.com/machine-learning/clustering/overview

[29] Algorithms for clustering data. A. Jain, R. Dubes. Prentice-Hall, Inc, Upper Saddle River, 1988.

[30] https://stanford.edu/~cpiech/cs221/handouts/kmeans.html , C. Piech, A. Ng.

[31] A study of hierarchical clustering algorithms. S. Patel, S. Sihmar, A. Jatain. In 2nd International Conference on Computing for Sustainable Global Development (INDIACom), 2015.

[32] A Comparative Analysis of Query Similarity Metrics for Community-Based Web Search. E. Balfe, B. Smyth. In ICCBR, 2005.

[33] Associative Query Answering via Query Feature Similarity. W.Chu, G. Zhang. In 1998.

[34] Query Similarity for Approximate Query Answering. V. Kantere. In DEXA, 2016.

[35] Query Intent Detection using Convolutional Neural Networks. H. B. Hashemi, A. Asiaee, R. Kraft. In WSDM QRUMS Workshop, 2016.

[36] A Recommendation System for Execution Plans Using Machine Learning. J. Zahir, A. El Qadi. In Mathematical and Computational Applications Journal, 2016.

[37] Supervised Machine Learning: A Review of Classification Techniques. S. B. Kotsiantis. In Informatica 31, 2007.

Παράρτημα Α

Στο παράρτημα αυτό παρατίθεται ο κώδικας του λογισμικού που υλοποιήθηκε και χρησιμοποιήθηκε στην παρούσα εργασία.

Παραγωγή ερωτημάτων

- **query_producer.py**

```
from simpleSQL import simpleSQLParser
from db import DBManager
import random



def dif_num_select(schema):
    queries = {}
    for t in schema.keys():
        queries[t] = []
        for i in range(1, len(schema[t])+1):
            attributes = ','.join(schema[t][:i])
            sql = "SELECT %s FROM %s" % (attributes, t)
            queries[t].append(sql)

    return queries



def star_joins(queries, fk_joins, parser):
    output_queries = []
    t_queries = {}
    for key in queries.keys():
        t_queries[key] = queries[key]
```

```
    for t in t_queries.keys():
        to_join = []
        avail = [tup for tup in fk_joins if (tup[0].split('.')[0] == t or tup[1].split('.')[0] == t)]
        print(t)
        #print(avail)

        old = parser.simpleSQL.parseString(t_queries[t]).asList()
        old_tokens = {'select': old[1], 'from': old[3], 'where': old[4]}
        #print(to_join)
        old_tokens['select'] = ', '.join(old_tokens['select'])
        old_tokens['from'] = ', '.join(old_tokens['from'])

        for j in range(len(avail)):
            if avail[j][0].split('.')[0] == t:
                other = str(avail[j][1].split('.')[0])
            else:
                other = str(avail[j][0].split('.')[0])
            to_join.append(other)

            new = parser.simpleSQL.parseString(t_queries[other]).asList()
            new_tokens = {'select': new[1], 'from': new[3], 'where': new[4]}
            combined_select = ', '.join(new_tokens['select']) + ', ' + old_tokens['select']
            combined_from = ', '.join(new_tokens['from']) + ', ' + old_tokens['from']
            #print(old_tokens['where'] == '')
            combined_where = old_tokens['where'] + avail[j][0] + ' = ' + avail[j][1] if old_tokens['where'] == '' else old_tokens['where'] + ' AND ' + avail[j][0] + ' = ' + avail[j][1]
            #print(combined_where)
            combined_query = 'SELECT %s FROM %s WHERE %s' % (combined_select, combined_from, combined_where)
            #print(old_tokens['where'].split(','))
            if(len(combined_where.split(' AND'))) >= 3:
                output_queries.append(combined_query)
            old_tokens['select'] = combined_select
            #print(combined_select)
```

```
            old_tokens['from'] = combined_from
            old_tokens['where'] = combined_where
            #print(combined_query)
            #input()

    return output_queries


def chain_joins(queries, fk_joins, parser):
    output_queries = []

    t_queries = {}
    for key in queries.keys():
        t_queries[key] = queries[key]

    for t in t_queries.keys():
        avail = [tup for tup in fk_joins if (tup[0].split('.')[0] == t or tup[1].split('.')[0] == t)]

        old = parser.simpleSQL.parseString(t_queries[t]).asList()
        old_tokens = {'select': old[1], 'from': old[3], 'where': old[4]}
        # print(to_join)
        old_tokens['select'] = ', '.join(old_tokens['select'])
        old_tokens['from'] = ', '.join(old_tokens['from'])
        starting_table = t

        for i in range(0, 8):
            empty = False
            while 1:
                print(avail)
                tup = random.choice(avail)
                if tup[0].split('.')[0] == starting_table:
                    other = str(tup[1].split('.')[0])
                else:
```

```
                other = str(tup[0].split('.')[0])
            print(other)
            print(old_tokens['from'].split(', '))
            print(other in old_tokens['from'].split(', '))
            if other in old_tokens['from'].split(', '):
                avail.remove(tup)
                print(avail)
                if len(avail) == 0:
                    empty = True
                    break
            else:
                break


        if not empty:
            new = parser.simpleSQL.parseString(t_queries[other]).asList()
            new_tokens = {'select': new[1], 'from': new[3], 'where': new[4]}
            combined_select = ', '.join(new_tokens['select']) + ', ' + old_tokens['select']
            combined_from = ', '.join(new_tokens['from']) + ', ' + old_tokens['from']
            #print(old_tokens['where'] == '')
            combined_where = old_tokens['where'] + tup[0] + ' = ' + tup[1] if old_tokens['where'] == '' else
old_tokens['where'] + ' AND ' + tup[0] + ' = ' + tup[1]
            #print(combined_where)
            combined_query = 'SELECT %s FROM %s WHERE %s' % (combined_select, combined_from,
combined_where)
            #print(old_tokens['where'].split(','))


            output_queries.append(combined_query)
            old_tokens['select'] = combined_select
            #print(combined_select)
            old_tokens['from'] = combined_from
            old_tokens['where'] = combined_where
            avail = [tup for tup in fk_joins if (tup[0].split('.')[0] == other or tup[1].split('.')[0] == other)]
            starting_table = other
        else:
```

```
            break

    return output_queries


def dif_num_select_with_joins(given_queries, parser):
    output_queries = []
    for query in given_queries:
        old = parser.simpleSQL.parseString(query).asList()
        old_tokens = {'select': old[1], 'from': old[3], 'where': old[4][1]}
        combine_where = ''
        if type(old_tokens['where'][0]) is list:
            for k in range(len(old_tokens['where'])):
                if old_tokens['where'][k] == 'and':
                    combine_where += ' and '
                else:
                    combine_where += ' '.join(old_tokens['where'][k])
            old_tokens['where'] = combine_where
        else:
            old_tokens['where'] = ' '.join(old_tokens['where'])

        num_of_selects = len(old_tokens['select'])
        for i in range(0, 5):
            times = random.randrange(1, num_of_selects)
            new_select = []
            for j in range(0, times):
                choice = random.choice(old_tokens['select'])
                while choice in new_select:
                    choice = random.choice(old_tokens['select'])
                new_select.append(choice)
            new_query = 'SELECT %s FROM %s WHERE %s' % (', '.join(new_select), ',
'.join(old_tokens['from']), old_tokens['where'])
            output_queries.append(new_query)
```

```
    return output_queries


def combine_joins(stars, chains, parser):
    output_queries = []
    for i in range(300):
        pick_star = random.choice(stars)
        star_tokenized = parser.simpleSQL.parseString(pick_star).asList()
        star_tokens = {'select': star_tokenized[1], 'from': star_tokenized[3], 'where': star_tokenized[4][1]}
        comb_where = ''
        if type(star_tokens['where'][0]) is list:
            for k in range(len(star_tokens['where'])):
                if star_tokens['where'][k] == 'and':
                    comb_where += ' and '
                else:
                    comb_where += ' '.join(star_tokens['where'][k])
            star_tokens['where'] = comb_where
        else:
            star_tokens['where'] = ' '.join(star_tokens['where'])


        while 1:
            pick_chain = random.choice(chains)
            chain_tokenized = parser.simpleSQL.parseString(pick_chain).asList()
            chain_tokens = {'select': chain_tokenized[1], 'from': chain_tokenized[3], 'where':
chain_tokenized[4][1]}


            common = 0
            common_table = ''
            for st in star_tokens['from']:
                if st in chain_tokens['from']:
                    common += 1
                    common_table = st
            if common != 1:
                continue
```

```
            else:
                comb_where = ''
                if type(chain_tokens['where'][0]) is list:
                    for k in range(len(chain_tokens['where'])):
                        if chain_tokens['where'][k] == 'and':
                            comb_where += ' and '
                        else:
                            comb_where += ' '.join(chain_tokens['where'][k])
                    chain_tokens['where'] = comb_where
                else:
                    chain_tokens['where'] = ' '.join(chain_tokens['where'])

                chain_tokens['select'] = [select for select in chain_tokens['select'] if select.split('.')[0] != common_table]
                chain_tokens['from'].remove(common_table)
                if not chain_tokens['select'] == []:
                    combined_select = ', '.join(star_tokens['select']) + ', ' + ', '.join(chain_tokens['select'])
                else:
                    combined_select = ', '.join(star_tokens['select'])
                combined_from = ', '.join(star_tokens['from']) + ', ' + ', '.join(chain_tokens['from'])
                combined_where = star_tokens['where'] + ' and ' + chain_tokens['where']
                combined_query = 'SELECT %s FROM %s WHERE %s' % (combined_select, combined_from, combined_where)
                output_queries.append(combined_query)
                break
    return output_queries


def add_constraints(queries, constraints, parser):
    output_queries = []
    for i in range(0, 100):
        if type(queries) is not list:
            pick_table = random.choice(list(queries))
            query = random.choice(queries[pick_table])
```

```
else:
    query = random.choice(queries)
tokenized = parser.simpleSQL.parseString(query).asList()
tokens = {'select': tokenized[1], 'from': tokenized[3], 'where': tokenized[4]}

avail = {}
for table in tokens['from']:
    for cons in constraints.keys():
        if cons.split('.')[0] == table:
            avail[cons] = constraints[cons]
to_add = []
for j in avail.keys():
    times = random.randint(0, 1)
    to_add += random.sample(avail[j], times)
to_add_as_where = ' and '.join(to_add)

if tokens['where'] != '':
    where_constraints = tokens['where'][1]
    comb_where = ''
    if type(where_constraints[0]) is list:
        for k in range(len(where_constraints)):
            if where_constraints[k] == 'and':
                comb_where += ' and '
            else:
                comb_where += ' '.join(where_constraints[k])
        where_constraints = comb_where
    else:
        where_constraints = ' '.join(where_constraints)
else:
    where_constraints = ''

new_select = ', '.join(tokens['select'])
new_from = ', '.join(tokens['from'])
```

```
        new_where = where_constraints + ' and ' + to_add_as_where if (where_constraints != '' and
to_add_as_where != '') else where_constraints + to_add_as_where
        if new_where == '':
            new_query = 'SELECT %s FROM %s' % (new_select, new_from)
        else:
            new_query = 'SELECT %s FROM %s WHERE %s' % (new_select, new_from, new_where)
        output_queries.append(new_query)
    return output_queries


def write_to_final(desc, list):
    if(len(list) < 50):
        for q in list:
            desc.write(q)
            desc.write('\n')
    else:
        to_write = random.sample(list, 50)
        for q in to_write:
            desc.write(q)
            desc.write('\n')


if __name__ == '__main__':


    fk_joins = [
        ('title.id', 'movie_keyword.movie_id'),
        ('title.id', 'movie_companies.movie_id'),
        ('title.id', 'movie_info_idx.movie_id'),
        ('title.id', 'movie_info.movie_id'),
        ('title.id', 'movie_link.movie_id'),
        ('title.id', 'complete_cast.movie_id'),
        ('title.id', 'aka_title.movie_id'),
        ('title.id', 'cast_info.movie_id'),
        ('title.kind_id', 'kind_type.id'),
        ('cast_info.role_id', 'role_type.id'),
        ('cast_info.person_role_id', 'char_name.id'),
```

```
    ('cast_info.person_id', 'name.id'),
    ('aka_name.person_id', 'name.id'),
    ('person_info.person_id', 'name.id'),
    ('person_info.info_type_id', 'info_type.id'),
    ('movie_info.info_type_id', 'info_type.id'),
    ('movie_info_idx.info_type_id', 'info_type.id'),
    ('movie_link.link_type_id', 'link_type.id'),
    ('aka_title.kind_id', 'kind_type.id'),
    ('keyword.id', 'movie_keyword.keyword_id'),
    ('complete_cast.status_id', 'comp_cast_type.id'),
    ('movie_companies.company_type_id', 'company_type.id'),
    ('movie_companies.company_id', 'company_name.id')
]

where_constraints = {
    'title.production':["title.production_year > 1950",
                        "title.production_year > 1991",
                        "title.production_year > 2007",
                        "title.production_year > 1988",
                        "title.production_year < 1950",
                        "title.production_year < 2009",
                        "title.production_year < 1996",
                        "title.production_year < 1980",
                        "title.production_year < 1957",
                        "title.production_year < 1980 and title.production_year > 1970",
                        "title.production_year < 1994 and title.production_year > 1950",
                        "title.production_year < 2004 and title.production_year > 2000",
                        "title.production_year < 1999 and title.production_year > 1945"],
    'title.title':["title.title = 'Iron Man'",
                   "title.title = 'Star Wars'",
                   "title.title = 'Cinderella'",
                   "title.title = 'Cars'",
                   "title.title = 'Iron Man'",
```

```
            "title.title = 'Rocky'",
            "title.title = 'Batman'",
            "title.title = 'Superman'",
            "title.title = 'Spiderman'",
            "title.title = 'Godfather'"],
'kind_type.kind':["kind_type.kind = 'movie'",
              "kind_type.kind = 'episode'",
              "kind_type.kind = 'tv series'",
              "kind_type.kind = 'video game'"],
'company_type.kind':["company_type.kind = 'distributors'",
                "company_type.kind = 'special effects companies'"],
'company_name.name':["company_name.name = 'Canal+'",
              "company_name.name = 'Paramount'",
              "company_name.name = 'Columbia'",
              "company_name.name = 'HBO'",
              "company_name.name = 'Fox'",
              "company_name.name = 'Universal Pictures'",
              "company_name.name = 'Warner Bros.'",
              "company_name.name = 'Lionsgate'",
              "company_name.name = 'Walt Disney Studios'",
              "company_name.name = 'MTV'"],
'company_name.country_code':["company_name.country_code = '[es]'",
                    "company_name.country_code = '[us]'",
                    "company_name.country_code = '[gr]'",
                    "company_name.country_code = '[fr]'",
                    "company_name.country_code = '[de]'",
                    "company_name.country_code = '[it]'",
                    "company_name.country_code = '[ca]'"],
'char_name.name':["char_name.name = 'Tony Stark'",
            "char_name.name = 'Captain America'",
            "char_name.name = 'Cinderella'",
            "char_name.name = 'Jack Sparrow'",
            "char_name.name = 'Batman'"],
```

```
'role_type.role':["role_type.role = 'actor'",
                  "role_type.role = 'director'",
                  "role_type.role = 'producer'",
                  "role_type.role = 'composer'"],
'name.name':["name.name = 'Depp, Johnny'",
             "name.name = 'Kidman, Nicole'",
             "name.name = 'Bloom, Orlando'",
             "name.name = 'Johansson, Scarlett'"],
'info_type.info':["info_type.info = 'birth name'",
                  "info_type.info = 'birth date'",
                  "info_type.info = 'spouse'",
                  "info_type.info = 'release dates'"],
'keyword.keyword':["keyword.keyword = 'love'",
                   "keyword.keyword = 'basketball'",
                   "keyword.keyword = 'tango'",
                   "keyword.keyword = 'evil'"],
'movie_info.info':["movie_info.info = 'English'",
                   "movie_info.info = 'Spanish'",
                   "movie_info.info = 'Greek'",
                   "movie_info.info = 'Comedy'",
                   "movie_info.info = 'Drama'",
                   "movie_info.info = 'Action'",
                   "movie_info.info = 'Mystery'"],
'aka_name.name':["aka_name.name = 'Short, Too'",
                 "aka_name.name = 'Monkey'",
                 "aka_name.name = 'Martin'",
                 "aka_name.name = 'Special, 38'"],
'aka_title.title':["aka_title.title = 'Dead Wrong'",
                   "aka_title.title = 'Deception'",
                   "aka_title.title = 'Femmes'"],
'cast_info.nr_order':["cast_info.nr_order > 3 and cast_info.nr_order < 10",
                      "cast_info.nr_order > 4 and cast_info.nr_order < 6",
                      "cast_info.nr_order > 10",
```

```
                "cast_info.nr_order < 10",
                "cast_info.nr_order > 15 and cast_info.nr_order < 20",
                "cast_info.nr_order > 11 and cast_info.nr_order < 14"]
}

dbman = DBManager()
schema = dbman.get_schema()
f = open("sample_queries.txt", 'w')
s = open("star_join_queries.txt", 'w')
c = open("chain_join_queries.txt", 'w')
dss = open('dif_select_star_joins.txt', 'w')
dsc = open('dif_select_chain_joins.txt', 'w')
cj = open('combined_joins_queries.txt', 'w')
cons = open("with_constraints.txt", 'w')
starcons = open("with_constraints_stars.txt", 'w')
chaincons = open("with_constraints_chains.txt", 'w')
combcons = open("with_constraints_combined.txt", 'w')
final_queries = open("final_queries.txt",'w')

for t in schema.keys():
    print(t, schema[t])

queries = dif_num_select(schema)
for t in schema.keys():
    for q in queries[t]:
        f.write(q)
        f.write('\n')
        print(q)

parser = simpleSQLParser()
big_queries = {}
for t in schema.keys():
    big_queries[t] = queries[t][-1]
```

```
stars = star_joins(big_queries, fk_joins, parser)
#print(big_queries)
for q in stars:
    s.write(q)
    s.write('\n')
    print(q)

chains = chain_joins(big_queries,fk_joins, parser)
for q in chains:
    c.write(q)
    c.write('\n')
    print(q)

dif_select_stars = dif_num_select_with_joins(stars, parser)
for q in dif_select_stars:
    dss.write(q)
    dss.write('\n')
    print(q)

dif_select_chains = dif_num_select_with_joins(chains, parser)
for q in dif_select_chains:
    dsc.write(q)
    dsc.write('\n')
    print(q)

combined = combine_joins(dif_select_stars, dif_select_chains, parser)
for q in combined:
    cj.write(q)
    cj.write('\n')
    print(q)

with_constraints = add_constraints(queries, where_constraints, parser)
```

```
for q in with_constraints:
    cons.write(q)
    cons.write('\n')
    print(q)

with_constraints_stars = add_constraints(stars, where_constraints, parser)
for q in with_constraints_stars:
    starcons.write(q)
    starcons.write('\n')
    print(q)

with_constraints_chains = add_constraints(chains, where_constraints, parser)
for q in with_constraints_chains:
    chaincons.write(q)
    chaincons.write('\n')
    print(q)

with_constraints_combined = add_constraints(combined, where_constraints, parser)
for q in with_constraints_combined:
    combcons.write(q)
    combcons.write('\n')
    print(q)

simple_queries =[]
for key in queries.keys():
    simple_queries += queries[key]

write_to_final(final_queries, simple_queries)
write_to_final(final_queries, stars)
write_to_final(final_queries, chains)
write_to_final(final_queries, dif_select_stars)
write_to_final(final_queries, dif_select_chains)
write_to_final(final_queries, combined)
```

```
    write_to_final(final_queries, with_constraints)
    write_to_final(final_queries, with_constraints_stars)
    write_to_final(final_queries, with_constraints_chains)
    write_to_final(final_queries, with_constraints_combined)

    starcons.close()
    chaincons.close()
    combcons.close()
    cons.close()
    cj.close()
    dss.close()
    dsc.close()
    c.close()
    s.close()
    f.close()
    final_queries.close()
```

Μετατροπή Αναπαράστασης

- **parser.py**

```
# source: https://sqlparse.readthedocs.io/en/latest/
# import sqlparse
from db import DBManager
from simpleSQL import simpleSQLParser
import numpy as np

class Parser:

    def __init__(self, filename, conn, sqlparser):
        self.sqlfile = filename
        self.conn = conn
        self.sqlparser = sqlparser
```

```
def extract_features(self, tokens):
    ov_columns = self.conn.get_number_of_columns()
    ov_tables = self.conn.get_tables()

    select_attributes = tokens['select']
    from_tables = tokens['from']
    prc_select = float(len(select_attributes)) / float(ov_columns)
    prc_from = float(len(from_tables)) / float(len(ov_tables))
    print(prc_select, prc_from)

    const_num = 0
    joins = 0
    star = 0
    chain = 0
    prc_star = 0
    prc_chain = 0
    named_joins = {}

    for table in from_tables:
        named_joins[table] = 0
    print(tokens['where'])
    if tokens['where'] != '':
        where_constraints = tokens['where'][1]
        print(where_constraints)
        if type(where_constraints[0]) is list:
            for c in range(0, len(where_constraints),2):
                if type(where_constraints[c][2]) is int:
                    const_num += 1
                elif type(where_constraints[c][2]) is str:
                    if where_constraints[c][2].startswith("'") and where_constraints[c][2].endswith("'"):
                        const_num += 1
                    else:
                        joins += 1
```

```
                left_side = where_constraints[c][0].split('.')[0]
                right_side = where_constraints[c][2].split('.')[0]
                named_joins[left_side] += 1
                named_joins[right_side] += 1
    else:
        if type(where_constraints[2]) is int:
            const_num += 1
        elif type(where_constraints[2]) is str:
            if where_constraints[2].startswith("'") and where_constraints[2].endswith("'"):
                const_num += 1
            else:
                joins += 1
                left_side = where_constraints[0].split('.')[0]
                right_side = where_constraints[2].split('.')[0]
                named_joins[left_side] += 1
                named_joins[right_side] += 1

count = 0
change = ''
for i in named_joins.keys():
    if named_joins[i] > 2:
        if named_joins[i] == 3:
            change = i
        count += 1
        star += named_joins[i]
    elif named_joins[i] == 2:
        chain += 1

if (not(star >= 3) and chain >= 1) or joins == 1:
    chain += 1

if count == 2:
    star -= named_joins[change]
```

```
            chain += 2

        print(joins)
        print(named_joins)
        print(star, chain)
        print(const_num)

        if joins !=0:
            prc_star = float(star) / float(joins)
            prc_chain = float(chain) / float(joins)

        return [prc_star, prc_chain, joins, prc_select, const_num, prc_from]

    def read_file(self):
        f = open(self.sqlfile)
        raw = f.read()
        f.close()
        return raw

    def parse_file(self):
        features = []
        raw = self.read_file()
        statements = raw.splitlines()
        print(statements)
        for i in statements:
            if i != '':
                print(i)
                f = self.parse_statement(i)
                features.append(f)
        return features

    def parse_statement(self, statement):
        parsed = self.sqlparser.simpleSQL.parseString(statement).asList()
```

```
        tokens = {'select': parsed[1], 'from': parsed[3], 'where': parsed[4]}
        return self.extract_features(tokens)


if __name__ == '__main__':
    sqlfile = '/home/george/diplwmatikh_new/clustering_test/src/final_queries.txt'
    conn = DBManager()
    sqlparser = simpleSQLParser()
    p = Parser(sqlfile, conn, sqlparser)
    features = p.parse_file()
    features_np = np.array(features)
    print(features_np)
    np.savetxt("_scaled_parsed_queries.csv", features_np, delimiter=',', fmt='%10.8f')
```

Ομαδοποίηση Ερωτημάτων

- **cluster.py**

```
# source: https://towardsdatascience.com/machine-learning-algorithms-part-12-hierarchical-agglomerative-clustering-example-in-python-1e18e0075019


import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
from sklearn.cluster import AgglomerativeClustering
import scipy.cluster.hierarchy as sch
from sklearn.preprocessing import MinMaxScaler


if __name__ == '__main__':
    dataset = pd.read_csv('./_scaled_parsed_queries.csv', header=None)
    print(dataset)
    #X = dataset.iloc[:, [3, 4]].values
    scaler = MinMaxScaler()
```

```
dataset = pd.DataFrame.from_records(scaler.fit_transform(dataset))
link = sch.linkage(dataset, method='ward')
dendrogram = sch.dendrogram(link)
plt.show()
# max_dists = sch.maxdists(link)
# print(max_dists)

model = AgglomerativeClustering(n_clusters=10, affinity='euclidean', linkage='ward')
model.fit(dataset)
labels = model.labels_
print(labels)
dataset[6] = labels
print(dataset)
dataset.to_csv('./scaled_labeled_queries.csv', header=None)
```

Ταξινόμιση ερωτημάτων

- **classifier.py**

```
# source : https://towardsdatascience.com/solving-a-simple-classification-problem-with-python-fruits-lovers-edition-d20ab6b071d2

from sklearn.naive_bayes import GaussianNB
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import classification_report
import joblib
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
```

```
if __name__ == '__main__':
    dataset = pd.read_csv('./scaled_labeled_queries.csv', header=None)
    dataset2 = pd.read_csv('./_scaled_parsed_queries.csv', header=None)
    print(dataset)
    data = dataset2.iloc[:, 1:7]
    labels = dataset[7]
    # print(data)
    # print(labels)

    X_train, X_test, y_train, y_test = train_test_split(data, labels, random_state=0)
    scaler = MinMaxScaler()
    # X_train = scaler.fit_transform(X_train)
    # X_test = scaler.transform(X_test)

    np.savetxt('./train_set.csv', X_train, delimiter=',')
    np.savetxt('./test_set.csv', X_test, delimiter=',')

    gnb = GaussianNB()
    gnb.fit(X_train, y_train)
    print('Accuracy of GNB classifier on training set: {:.2f}'
         .format(gnb.score(X_train, y_train)))
    print('Accuracy of GNB classifier on test set: {:.2f}'
         .format(gnb.score(X_test, y_test)))

    print('\n')
    knn = KNeighborsClassifier()
    knn.fit(X_train, y_train)
    print('Accuracy of K-NN classifier on training set: {:.2f}'
         .format(knn.score(X_train, y_train)))
    print('Accuracy of K-NN classifier on test set: {:.2f}'
         .format(knn.score(X_test, y_test)))
```

```
print('\n')
clf = DecisionTreeClassifier().fit(X_train, y_train)
print('Accuracy of Decision Tree classifier on training set: {:.2f}'
     .format(clf.score(X_train, y_train)))
print('Accuracy of Decision Tree classifier on test set: {:.2f}'
     .format(clf.score(X_test, y_test)))

print('\n')
lda = LinearDiscriminantAnalysis()
lda.fit(X_train, y_train)
print('Accuracy of LDA classifier on training set: {:.2f}'
     .format(lda.score(X_train, y_train)))
print('Accuracy of LDA classifier on test set: {:.2f}'
     .format(lda.score(X_test, y_test)))

print('\n')
svm = SVC()
svm.fit(X_train, y_train)
print('Accuracy of SVM classifier on training set: {:.2f}'
     .format(svm.score(X_train, y_train)))
print('Accuracy of SVM classifier on test set: {:.2f}'
     .format(svm.score(X_test, y_test)))

pred_gnb = gnb.predict(X_test)
pred_knn = knn.predict(X_test)
pred_clf = clf.predict(X_test)
pred_lda = lda.predict(X_test)
pred_svm = svm.predict(X_test)

print("Gaussian Classifier Report")
print(classification_report(y_test, pred_gnb))

print("KNN Classifier Report")
```

```
print(classification_report(y_test, pred_knn))

print("Decision Tree Classifier Report")
print(classification_report(y_test, pred_clf))

print("LDA Classifier Report")
print(classification_report(y_test, pred_lda))

print("SVM Classifier Report")
print(classification_report(y_test, pred_svm))

filename = './finalized_model.sav'
joblib.dump(clf, filename)
```

Μεταγραφή ερωτημάτων

- **RewritingSystem.py**

```
import sys
import os
import subprocess

import joblib
from sklearn.preprocessing import MinMaxScaler

from db import DBManager
from simpleSQL import simpleSQLParser
from parser import Parser
import numpy as np
from entropy import Entropy_Calc

def main(argv):
        filename = input("Give the name of the file with the query to be rewritten: ")
```

```
        conn = DBManager()
        sqlparser = simpleSQLParser()
        p = Parser(filename, conn, sqlparser)

        sim = input("\nGive the similarity function to be used: ")
        f = open(filename,"r")
        if os.stat(filename).st_size == 0:
                print("\nThis is an empty file!")
        else:
                sql = f.readline()
                print("The given query is: "+sql+"\n")
                print("Extracting features for the given query\n")
                features = p.parse_file()
                features_np = np.array(features)
                print("The features of the query are:")
                print(features_np)
                classifier = joblib.load('finalized_model.sav')
                classified = classifier.predict(features_np)
                print("The query belongs to class %s" % (classified[0]))
                print("Running the jar\n")
                subprocess.check_call(["java", "-jar","Rewriting.jar",filename,"Global", "Local", sim,
os.getcwd()+"/"])
                print("Rewriting completed without error")

                results = open("results.txt","r")
                lines = results.read().split('**********')
                rewritten = lines[-1].split('---------')[1] + ';'

                print("The rewritten query is:\n")
                print(rewritten)

if __name__ == "__main__":
        main(sys.argv[1:])
```

Βοηθητικές κλάσεις

- **db.py**

```
import subprocess
import psycopg2


class DBManager:

    # Constructor of the class. The needed properties for the connection with the DB are hard-coded for now.
    def __init__(self):
        self.verbose = False
        self.connect = None
        self.cursor = None
        self.currentDB = 'imdb_custom'
        self.user = 'george'
        self.password = '281794666'
        self.host = 'localhost'
        self.port = 5432
        self.connect_str = 'host=' + self.host + ' port=' + str(
            self.port) + ' dbname=' + self.currentDB + ' user=' + \
                       self.user + " password=" + self.password
        self.connect_db()

    # <-------------------------------------------------------------------------------------->

    # Use the psycopg2 library to connect to the database and set the cursor and connect variables.
    def connect_db(self):
        print("Connecting to the database [%s] ..." % self.currentDB)
        try:
            if self.verbose:
                subprocess.check_call(['echo', 'Trying to connect to %s' % (self.currentDB)])
            self.connect = psycopg2.connect(self.connect_str)
```

```
            self.cursor = self.connect.cursor()
        except psycopg2.OperationalError as e:
            # click.echo(e, file=sys.stderr)
            raise ConnectionError(
                "Cannot connect to the database [%s] @ [%s]:[%s]. Check connection, username, password and database name." % (
                    self.currentDB, self.host, self.port))
        return self


    # <-------------------------------------------------------------------------------------->


    def get_tables(self, ):
        sql = "SELECT TABLE_NAME FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_SCHEMA='public'"
        self.cursor.execute(sql)
        tables = self.cursor.fetchall()
        return [t for t in tables]


    def get_number_of_columns(self):
        sql = "SELECT COUNT(*) FROM INFORMATION_SCHEMA.columns WHERE TABLE_SCHEMA='public'"
        self.cursor.execute(sql)
        return self.cursor.fetchall()[0][0]


    def get_schema(self):
        tables = self.get_tables()
        schema = {}
        for t in tables:
            sql = "SELECT column_name FROM information_schema.columns WHERE table_schema = 'public' and table_name = '%s';"%(t)
            self.cursor.execute(sql)
            att = [t[0]+"."+a[0] for a in self.cursor.fetchall()]
            schema[t[0]] = att
        return schema
```

```
def execute_query(self, query):
    if query != '':
        self.cursor.execute(query)
        self.connect.commit()
        return self.cursor.fetchall()
    else:
        print('empty query!')
```