ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
ΕΥΦΥΕΙΣ ΤΕΧΝΟΛΟΓΙΕΣ ΔΙΑΔΙΚΤΥΟΥ - WEB INTELLIGENCE

Τεχνικές Μείωσης Δεδομένων για Σύνολα Δεδομένων Πολλαπλών Ετικετών

(Data Reduction Techniques for Multilabel Datasets)

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΠΑΝΑΓΙΩΤΗ ΦΙΛΙΠΠΑΚΗ

Επιβλέπων: Στέφανος Ουγιάρογλου
Ε.ΔΙ.Π., ΔΙ.ΠΑ.Ε.

Θεσσαλονίκη, Μάρτιος, 2021

ΔΙΕΘΝΕΣ ΠΑΝΕΠΙΣΤΗΜΙΟ ΤΗΣ ΕΛΛΑΔΟΣ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ ΕΥΦΥΕΙΣ ΤΕΧΝΟΛΟΓΙΕΣ ΔΙΑΔΙΚΤΥΟΥ - WEB INTELLIGENCE

Τεχνικές Μείωσης Δεδομένων για Σύνολα Δεδομένων Πολλαπλών Ετικετών

(Data Reduction Techniques for Multilabel Datasets)

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΠΑΝΑΓΙΩΤΗ ΦΙΛΙΠΠΑΚΗ

Επιβλέπων: Στέφανος Ουγιάρογλου
Ε.ΔΙ.Π., ΔΙ.ΠΑ.Ε.

Εγκρίθηκε απο την τριμελή εξεταστική επιτροπή στις 6 Μαρτίου 2021.

*(*Υπογραφή)	*(*Υπογραφή)	*(*Υπογραφή)
.....	.....	.....
Στέφανος Ουγιάρογλου	Κωσταντίνος Διαμαντάρας	Δημήτριος Δέρβος
Ε.ΔΙ.Π. ΔΙ.ΠΑ.Ε.	Καθηγητής ΔΙ.ΠΑ.Ε.	Καθηγητής ΔΙ.ΠΑ.Ε.

Θεσσαλονίκη, Μάρτιος, 2021

(Υπογραφή)

.....
Παναγιώτης Φιλιππάκης
Σχολή Θετικών Επιστημών και Τεχνολογίας Ε.Α.Π.

Πρόλογος

Το πρόγραμμα μεταπτυχιακών σπουδών ”Ευφυείς Τεχνολογίες Διαδικτύου” περιλαμβάνει μαθήματα που δίνουν την ευκαιρία στους φοιτητές να κατανοήσσουν ότι οι ραγδαίες εξελίξεις στην τεχνολογία της συλλογής και αποθήκευσης δεδομένων σε συνδυασμό με την ευκολία με την οποία μπορούν να παραχθούν και να διαδοθούν τα δεδομένα έχουν προκαλέσει την εκρηκτική αύξηση του όγκου των δεδομένων, οδηγώντας στην τωρινή εποχή των μεγάλων δεδομένων **(Big Data)**. Η εξαγωγή γνώσης για χρήση απο αυτά τα μεγάλα σύνολα δεδομένων γίνεται ολοένα και σημαντικότερη για τη λήψη αποφάσεων σχεδόν σε όλους τους τομείς της κοινωνίας, συμπεριλαμβανομένων των επιχειρήσεων και της βιομηχανίας, των επιστημών και της μηχανολογίας, της ιατρικής και της βιοτεχνολογίας, των κυβερνήσεων αλλά και των ατόμων ξεχωριστά. Ωστόσο, η ποσότητα (όγκος) των δεδομέων, η πολυπλοκότητα (ποικιλία) τους και ο ρυθμός με τον οποίο συλλέγονται και υφίστανται επεξεργασία έχουν μεγαλώσει πάρα πολύ καθιστώντας δύσκολη την ανάλυση τους. Επομένως, υπάρχει τεράστια ανάγκη για αυτοματοποιημένα εργαλεία εξαγωγής χρήσιμης πληροφορίας απο τα μεγάλα σύνολα δεδομένων, παρά τις δυσκολίες που θέτουν το τεράστιο μέγεθος και η διαφορετικότητα τους.

Οι άνθρωποι έχουν έμφυτη την ικανότητα να τοποθετούν τα πράγματα σε κατηγορίες, δηλαδή να εκτελούν απλές εργασίες κατηγοριοποίησης όπως το φιλτράρισμα των ανεπιθύμητων (spam) μηνυμάτων ηλεκτρονικού ταχυδρομείου ή πιο εξειδικευμένες, όπως η αναγνώριση ουράνιων σωμάτων απο εικόνες τηλεσκοπίων. Ενώ η κατηγοριοποίηση με απλές τεχνικές Μηχανικής Μάθησης και Εξόρυξης Γνώσης συχνά αρκεί για απλά και μικρού όγκου σύνολα δεδομένων εκπαίδευσης με λίγα χαρακτηριστικά, για τα μεγαλύτερα και πιο πολύπλοκα σύνολα δεδομένων εκπαίδευσης απαιτούνται γρήγοροι αλγόριθμοι που είναι κατάλληλοι, για τους μεγάλους όγκους των δεδομένων ή απαιτείται η μείωση των δεδομένων με αλγόριθμους που αξιολογούν τα δεδομένα εκπαίδευσης και ”κρατούν” τα πιο αντιπροσωπευτικά στιγμιότυπα ή παράγουν νέα στιγμιότυπα που αντιπροσωπεύουν το αρχικό σύνολο δεδομένων εκπαίδευσης.

Εάν υπάρχουν μόνο δύο κατηγορίες, τότε αναφερόμαστε σε ένα δυαδικό πρόβλημα κατηγοριοποιήσης (binary). Ωστόσο, είναι πιθανόν, να υπάρχουν και περισσότερες από μια κατηγορίες. Τότε, το πρόβληβλημα ονομάζεται πρόβλημα κατηγοριοποίησης πολλων κατηγοριών (multiclass). Τέλος, είναι πιθανόν, ένα στιγμιότυπο να ανήκει σε περισσότερες από μια κατηγορίες. Τότε, το πρόβλημα ονομάζεται πρόβλημα πολλαπλών ετικετών (multilabel). Η κατηγοριοποίηση πολλαπλών ετικετών έχει χαρακτηριστεί ως πιο πρόσφατος όρος της τελευταίες δεκαετίες, σε σχέση με μια κατάσταση στην οποία περισσότερες από μία ετικέτες μπορούν να ανατεθούν σε ένα στιγμιότυπο [1] και αποτελεί το αντικείμενο της παρούσας εργασίας.

Σύμφωνα με τους [2], οι τεχνικές προεπεξεργασίας χωρίζονται συνήθως σε: προετοιμασία δεδομένων (per-processing) και σε μείωση δεδομένων (data reduction). Η προετοιμασία των δεδομένων περιλαμβάνει, μεταξύ άλλων, την ομαλοποίηση των δεδομένων, τον ”καθαρισμό” των δεδομένων και την ανίχνευση θορύβου, ενώ η μείωση των δεδομένων, όπως υποδηλώνει η ονομασία, μειώνει τον συνολικό όγκο των δεδομένων, διατηρώντας παράλληλα τις απαραίτητες πληροφορίες. Αυτές οι μέθοδοι μπορούν να κατηγοριοποιηθούν σε τρεις ομάδες: στη μείωση ή επιλογή διαστάσεων, και στην παραγωγή ή επιλογή στιγμιοτύπων [3] .

Υπάρχουν πολλές τεχνικές μείωσης δεδομένων διαθέσιμες στη βιβλιογραφία για προβλήματα κατηγοριοποίησης. Η συντριπτική πλειοψηφία αφορά δυαδικά προβλήματα κατηγοριοποίησης και προβληματα πολλών κατηγοριών. Ελάχιστες ερευνητικές προσπάθειες έχουν πραγματοποιήθει που να αφορούν τη μείωση δεδομένων εκπαίδευσης πολλαπλών ετικετών. Με αφορμή αυτή την παρατήρηση, η εργασία αυτή εξερευνά τις υπάρχουσες τεχνικές μείωσης δεδομένων εκπαίδευσης πολλαπλών ετικετών και προσφέρει νέες τεχνικές μείωσης των δεδομένων για σύνολα πολλαπλών ετικετών [4].

Περίληψη

Οι διαδικασίες κατηγοριοποίησης συναντιούνται σε ένα ευρύ φάσμα των ανθρώπινων δραστηριοτήτων. Με τον όρο κατηγοριοποίηση εννοούμε το να πραγματοποιηθεί μια πρόβλεψη για ένα νέο στιγμιότυπο με βάση τα διαθέσιμα δεδομένα εκπαίδευσης. Στόχος είναι το να δημιουργηθεί ένας κατηγοριοποιητής βάσει ενός συνόλου στιγμιοτύπων εκπαίδευσης ο οποίος θα είναι σε θέση να προβλέψει την κατηγορία ενός νέου στιγμιοτύπου με όσο το δυνατόν μεγαλύτερη ακρίβεια [5]. Σε μεγάλα σύνολα δεδομένων, είναι επιτακτική ανάγκη να μειώσουμε τα αρχικά δεδομένα, ώστε να μειωθεί και ο χρόνος επεξεργασίας ενώ παράλληλα να μην χαθεί πολύτιμη πληροφορία που θα καθιστά τον κατηγοριοποιητή λιγότερο αποτελεσματικό. Με άλλα λόγια, θα πρέπει η διαδικασία της κατηγοριοποίησης να παράγει τα ίδια ή και καλύτερα αποτελέσματα (προβλέψεις) χρησιμοποιώντας το νέο μειωμένο σύνολο δεδομένων σε σχέση με το αρχικό.

Υπάρχουν πολλές τεχνικές μείωσης δεδομένων εκπαίδευσης διαθέσιμες στη βιβλιογραφία για προβλήματα κατηγοριοποίησης. Οι τεχνικές αυτές είτε επιλέγουν πρότυπα (αντιπροσωπευτικά στιγμιότυπα) (Prototype Selection) είτε παράγουν πρότυπα συνοψίζοντας παρόμοια στιγμιότυπα (Prototype Generation). Η συντριπτική πλειοψηφία των τεχνικών αυτών αφορά προβλήματα κατηγοριοποίησης μονής κατηγορίας όπου κάθε στιγμιότυπο ανήκει σε μια και μόνο κατηγορία. Ελάχιστες ερευνητικές προσπάθειες έχουν πραγματοποιηθεί που να αφορούν τη μείωση δεδομένων εκπαίδευσης πολλαπλών ετικετών, δηλαδή στιγμιοτύπων που να ανήκουν σε περισσότερες από μια κατηγορίες. Ωστόσο, η απόδοση των τεχνικών μείωσης δεδομένων πολλαπλών ετικετών εξαρτάται σε μεγάλο βαθμό από παραμέτρους που προσδιορίζει ο χρήστης μέσω υπολογιστικά κοστοβόρων διαδικασιών. Επιπρόσθετα, οι τεχνικές μείωσης δεδομένων μονής κατηγορίας δεν μπορούν να εφαρμοστούν σε συνδυασμό με τις διαδεδομένες μεθόδους μετασχηματισμού προβλήματος πολλαπλών ετικετών σε πρόβλημα μονής κατηγορίας. Αυτές οι παρατηρήσεις αποτελούν το κίνητρο της παρούσας διπλωματικής εργασίας.

Η παρούσα διπλωματική εργασία συνεισφέρει στην ανάπτυξη νέων τεχνικών μείωσης δεδομένων εκπαίδευσης πολλαπλών ετικετών που δεν περιλαμβάνουν παραμέτρους. Για να επιτευχθεί ο στόχος χρησιμοποιήθηκε η βασική λειτουργία του αλγορίθμου συσταδοποίησης K-means ο οποίος όμως εκτελείται επαναληπτικά στις μη ομοιογενείς συστάδες που παράγονται. Στα σύνολα πολλαπλών ετικετών, μια συστάδα θεωρείται ομοιογενής όταν όλα τα στιγμιότυπα της συστάδας έχουν τουλάχιστον μια κοινή ετικέτα. Στο τέλος της επαναληπτικής διαδικασίας συσταδοποίησης όλες οι συστάδες γίνονται ομοιογενείς και τα κέντρα των συστάδων αποτελούν τα πρότυπα που συνθέτουν το μειωμένο σύνολο δεδομένων. Με βάση αυτή τη λειτουργία επαναληπτικής συσταδοποίησης δημιουργήσαμε δυο τεχνικές μείωσης δεδομένων που ανήκουν στην κατηγορία παραγωγής προτύπων. Οι τεχνικές που αναπτύχθηκαν ονομάστηκαν **MRHC1** και **MRHC2** και παράγουν αντιπροσωπευτικά στιγμιότυπα του αρχικού συνόλου, μειώνοντας έτσι σε μεγάλο βαθμό το αρχικό σύνολο δεδομένων σε προβλήματα πολλαπλών ετικετών. Επίσης, στα πλαίσια της παρούσας διπλωματικής εργασίας αναπτύχθηκαν παραλλαγές του γνωστού αλγορίθμου των Κ πλησιέστερων γειτόνων (**KNN**). Οι παραλλαγές ονομάστηκαν **MKNN1** και **MKNN2** και χρησιμοποιήθηκαν για να επιτύχουμε αποτελεσματική κατηγοριοποίηση σε σύνολα δεδομένων πολλαπλών ετικετών που έχουν παραχθεί από τεχνικές μείωσης δεδομένων.

Η απόδοση των προτεινόμενων αλγόριθμων ελέγχθηκε εκτελώντας πειράματα σε εννέα σύνολα δεδομένων εκπαίδευσης πολλαπλών ετικετών και για την αξιολόγηση τους μετρή-

θηκε η απώλεια Hamming Loss χρησιμοποιώντας 5-fold cross validation. Απο τα αποτελέσματα των πειραμάτων προκύπτει οτι οι προτεινόμενοι αλγόριθμοι επιτυγχάνουν δυο στόχους. Ο πρώτος είναι η σημαντική μείωση του αρχικού συνόλου δεδομένων πολλαπλών ετικετών που ισοδυναμεί με μείωση του χρόνου επεξεργασίας. Ο δεύτερος στόχος που επιτεύχθει είναι η διατήρηση της ακρίβειας (απώλεια Hamming Loss) στα ίδια επίπεδα με αυτή που επιτυγχάνει ο κατηγοριοποιητής που χρησιμοποιεί το αρχικό, μεγάλο σε μέγεθος, σύνολο δεδομένων εκπαίδευσης ενώ σε κάποιες περιπτώσεις παρατηρείται και βελτίωση της ακρίβειας.

Λέξεις Κλειδιά:«Κατηγοριοποίηση Πολλαπλών Ετικετών, Μείωση Δεδομένων, Παραγωγή Προτύπων, Κατηγοριοποίηση Κ εγγύτερων γειτόνων (KNN)»

Abstract

Classification tasks can be found in daily human activities. By the term of classification we mean that a prediction can be made for a new instance based on a training set. The main objective is to construct a classifier depended on a training set of instances that will be able to predict the label of a new instance with the highest accurancy [5]. In large datasets, it is necessary to reduce the initial training data, so that the time of processing can be reduced as well whereas simultaneously precious information not to be lost so as not to make the classifier less efficient. In other words, the procedure of classification should produce the same or better results (predictions) by using the new reduced dataset relatively to the initial.

There are a lot of data reduction techniques which are available in the literature concerning the problems of classification. These techniques either choose prototypes (representative instances) (prototype Selection) or produce prototypes by summarizing similar instances (Prototype Generation). The majority of these techniques concerns problems of the singe label classification in which each instance belongs to one and only label. Few researching work have been carried out regarding the reduction of multilabel training sets, that is to say instances which belong to more than one label. However, the performance of the techniques of the reduction of multilabel datasets depends on the existence of parameters which the user defines by means of computationally costly processes. In addition, the techniques of the reduction of single label datasets can't be applied in combination with the well known transformation methods that transform a multilabel classification problem into a single label problem. These observations constitute the motive of the present msc thesis.

The present thesis contributes to the development of new parameter-free data reduction techniques for multilabel datasets. In order to achieve this goal, the basic K-means clustering is performed on a repetitive basis in no homogeneous clusters which are produced. Within the multilabel datasets, a cluster is considered to be homogeneous when all the instances of cluster have at least one common label. At the end of the repeating procedure of clustering all the clusters become homogeneous and the clusters means centroids constitute prototypes which compose the reduced dataset. We have developed two data reduction techniques which belong to the category of prototype generation. The techniques were named **MRHC1** and **MRHC2** and produce representative instances of the initial dataset. Furthermore, present thesis contributes variations of the known K-nearest neighbors classification algorithm (**KNN**) which are appropriate for multilabel classification problems. These variations were named **MKNN1** and **MKNN2** and use multilabel datasets which have been produced by data reduction techniques in order to perform efficient classification.

The performance of the proposed algorithms was tested by conducting experiments in nine multilabel datasets. Their performance was evaluated by the metric of Hamming Loss and by using a five fold cross validation schema. Taking into consideration the experimental results we conclude that the two algorithms satisfy two objectives of the research. The first objective is the significant reduction of the initial multilabel dataset and simultaneously the reduction of the corresponding CPU processing time needed for the classificatyion task. The second objective which was achieved was the maintenance of accurancy (Hamming Loss) at the same levels with the one that the classifier achieves by using the initial, big size, training dataset. In some cases an improvement of the accuracy is observed.

Keywords:«Multilabel classification, Data Reduction, Prototype Generation, k-NN Clas-

sification»

Ευχαριστίες

Θερμά ευχαριστώ τον επιβλέποντα της διπλωματικής εργασίας, μέλος ΕΔΙΠ του Τμήματος Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου της Ελλάδος, κύριο Ουγιάρογλου Στέφανο και τον Καθηγητή του Τμήματος Εφαρμοσμένης Πληροφορικής του Πανεπιστημίου Μακεδονίας, κύριο Ευαγγελίδη Γεώργιο, για την συνδρομή τους, στην εκπόνηση της συγκεκριμένης διπλωματικής εργασίας, την καθοριστική καθοδήγηση σε κρίσιμες περιόδους και τη διάθεση του χρόνου τους.

Οφείλω να ομολογήσω οτι η συνδρομή τους στην σύλληψη της ιδέας και ολοκλήρωση της διπλωματικής εργασίας ήταν καθοριστική.

Εξίσου θερμά ευχαριστώ τα μέλη της εξεταστικής επιτροπής, τον Καθηγητή του Τμήματος Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου της Ελλάδος, κύριο Διαμαντάρα Κωσταντίνο και τον Καθηγητή του Τμήματος Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου της Ελλάδος, κύριο Δέρβο Δημήτριο για τις υποδείξεις τους, οι οποίες συνέβαλαν στη βελτίωση της Διπλωματικής εργασίας.

Ευχαριστίες επίσης οφείλονται στο φίλο μου Τάσο Γαζιώτη, πτυχιούχο του Τμήματος Εφαρμοσμένης Πληροφορικής του Πανεπιστημίου Μακεδονία για την πολύτιμη συνδρομή του στο σκέλος του προγραμματισμού των αλγόριθμων σε γλώσσα python.

Τέλος, θερμά ευχαριστώ την γυναίκα μου Παναγιώτα και τα παιδιά μου Πέτρο, Βασίλη και Σάββα για την αμέριστη συμπαράσταση και υπομονή που επέδειξαν κατά την διάρκεια της εκπόνησης της διπλωματικής εργασίας. Με την συμπαράστασή τους, έμεινα επικεντρωμένος στον στόχο μου και τελικά ολοκλήρωσα τη διπλωματική εργασία μου.

Πίνακας Περιεχομένων

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

1 Εισαγωγή

1.1 Κατηγοριοποίηση δεδομένων

Η αποδοτικότητα και η αποτελεσματικότητα των αλγορίθμων εξόρυξης δεδομένων είναι ένα σημαντικό ερευνητικό πρόβλημα που έχει προσελκύσει την προσοχή τόσο της ακαδημαϊκής κοινότητας όσο και της βιομηχανίας. Η κατηγοριοποίηση (ή εποπτευμένη μάθηση σύμφωνα με την ορολογία της μηχανικής μάθησης) είναι εργασία εξόρυξης. Οι αλγόριθμοι κατηγοριοποίησης (ή κατηγοριοποιητές) προσπαθούν να εκχωρήσουν νέα, μη ταξινομημένα δεδομένα σε ένα σύνολο προκαθορισμένων ετικετών ή αλλιώς κλάσεων, με βάση τα διαθέσιμα δεδομένα εκπαίδευσης, δηλαδή ένα σύνολο στιγμιοτύπων με γνωστές ετικέτες. Ένα τυπικό παράδειγμα κατηγοριοποίησης είναι η αντιστοίχιση ενός μηνύματος ηλεκτρονικού ταχυδρομείου σε ετικέτα ”spam” ή ετικέτα ”μη-spam”.

Με άλλα λόγια, ένας κατηγοριοποιητής είναι ένας αλγόριθμος που βασίζεται σε στιγμιότυπα εκπαίδευσης όπου η ετικέτα είναι γνωστή, ώστε να μάθει μια συνάρτηση που παράγει μια κατάλληλη έξοδο, όταν του δίνεται ένα νέο στιγμιότυπο για το οποίο δεν γνωρίζουμε την ετικέτα του. Για παράδειγμα, φανταστείτε οτι ένας υπολογιστής είναι ένα παιδί και εμείς είμαστε οι γονείς του, που το επιτηρούμε. Θέλουμε το παιδί να μάθει πως μοιάζει το γουρούνι. Θα δείξουμε στο παιδί αρκετές διαφορετικές εικόνες, μερικές απο τις οποίες θα απεικονίζουν γουρούνια και τα υπόλοιπα θα μπορούσαν να είναι εικόνες απο άλλα ζώα όπως γάτες, σκύλοι, κ.τ.λ. Όταν θα βλέπουμε ένα γουρούνι θα φωνάζουμε ”γουρούνι”. Όταν δεν είναι γουρούνι θα φωνάζουμε ”όχι γουρούνι”. Αυτή η διαδικασία, ονομάζεται εκπαίδευση του μοντέλου κατηγοριοποίησης. Αφού επαναλάβουμε την παραπάνω διαδικασία αρκετές φορές κάποια στιγμή που θα εμφανίσουμε μια νέα εικόνα και θα το ρωτήσουμε ”είναι γουρούνι;” και αυτό θα απαντήσει ”Ναι” ή ”Οχι”. Με αυτό τον τρόπο, εφαρμόζεται ο κατηγοριοποιητής για να προβλέψει μελοντικές καταστάσεις για τις οποίες δεν έχει συναντήσει κατα την εκπαίδευση. Αυτη η διαδικασία, ονομάζεται κατηγοριοποίηση.

Οι κατηγοριοποιητές μπορούν να χωριστούν σε δύο κύριες κατηγορίες αλγορίθμων: (i) πρόθυμοι κατηγοριοποιητές (eager classsifiers), και (ii) οκνηροί (lazy (ή με βάση των στιγμιοτύπων)) κατηγοριοποιητές. Και οι δύο κατηγορίες έχουν το ίδιο κίνητρο, δηλαδή να προβλέψουν την ακριβή ετικέτα. Ωστόσο, διαφέρουν ως προς τον τρόπο λειτουργίας τους. Ουσιαστικό ρόλο για την αποτελεσματικότητα των αλγόριθμων και των δύο κατηγοριών παίζει το διαθέσιμο σύνολο εκπαίδευσης. Ένας πρόθυμος κατηγοριοποιητής προ-επεξεργάζεται τα διαθέσιμα δεδομένα εκπαίδευσης και δημιουργεί ένα μοντέλο κατηγοριοποίησης, που στη συνέχεια χρησιμοποιείται για την κατηγοριοποίηση νέων στιγμιοτύπων. Από την άλλη, οι οκνηροί κατηγοριοποιητές δεν δημιουργούν μοντέλο. Στην πραγματικότητα, θεωρούν το σύνολο δεδομένων εκπαίδευσης ως το μοντέλο κατηγοριοποίησης. Ένας οκνηρός αλγόριθμος κατηγοριοποιεί ένα νέο στιγμιότυπο με τη σάρωση του συνόλου εκπαίδευσης την στιγμή της άφιξής του.

Δεδομένου ότι οι πρόθυμοι κατηγοριοποιητές δημιουργούν ένα μοντέλο κατηγοριοποίησης πριν από την άφιξη οποιουδήποτε νέου στιγμιότυπου, η διαδικασία κατηγοριοποίησης είναι συνήθως

πολύ γρήγορη. Παρ'ολο το γεγονός οτι οι οκνηροί κατηγοριοποιητές δεν σπαταλούν πολύ χρόνο για να δημιουργήσουν ένα μοντέλο κατηγοριοποίησης, η διαδικασία κατηγοριοποίησης είναι πιο χρονοβόρα από εκείνη των πρόθυμων κατηγοριοποιητών.

Ένα μειονέκτημα των πρόθυμων κατηγοριοποιητών είναι ότι πρέπει να δημιουργήσουν μια ενιαία υπόθεση που καλύπτει ολόκληρο το σύνολο εκπαίδευσης. Αυτό δεν είναι πάντα εφικτό, μπορεί να επηρεάσει την ακρίβεια και μπορεί να καταστήσει την κατασκευή του μοντέλου κατηγοριοποίησης εξαιρετικά χρονοβόρα και περίπλοκη εργασία προεπεξεργασίας. Από την άλλη, οι οκνηροί κατηγοριοποιητές χρησιμοποιούν ολόκληρο το σύνολο εκπαίδευσης και, ως εκ τούτου, μπορεί να υιοθετήσουν πιο πολύπλοκες υποθέσεις σχετικά με τα δεδομένα. Κατά συνέπεια, μπορούν να βελτιώσουν την ακρίβεια κατηγοριοποίησης. Ένα μειονέκτημα των οκνηρών κατηγοριοποιητών είναι ότι απαιτούν όλα τα δεδομένα εκπαίδευσης να είναι πάντα διαθέσιμα, γεγονός που οδηγεί σε υψηλές απαιτήσεις αποθήκευσης. Αντίθετα, στην πρόθυμη κατηγοριοποίηση μετά την κατασκευή του μοντέλου, τα δεδομένα εκπαίδευσης μπορούν να αφαιρεθούν για την εξοικονόμηση αποθηκευτικού χώρου.

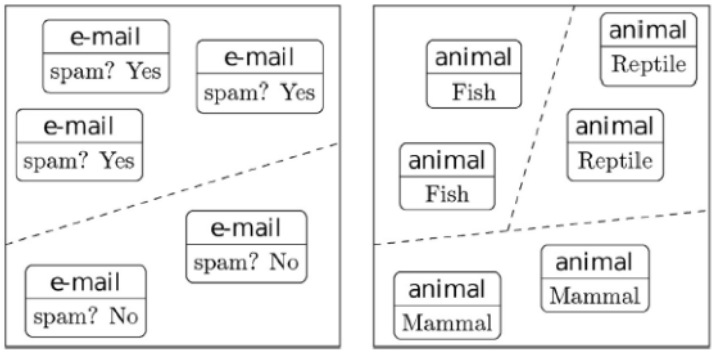
Κατά τις τελευταίες δεκαετίες, το πρόβλημα της κατηγοριοποίησης έχει προσελκύσει το ενδιαφέρον πολλών ερευνητών από διάφορους ερευνητικούς τομείς της επιστήμης των υπολογιστών. Ως εκ τούτου, διάφοροι πρόθυμοι και οκνηροί κατηγοριοποιητές έχουν προταθεί και είναι διαθέσιμοι στη διεθνή βιβλιογραφία.

Τα δέντρα απόφασης (decision trees) [6] αποτελούν μια γνωστή υποκατηγορία των πρόθυμων κατηγοριοποιητών. Με βάση τα διαθέσιμα δεδομένα εκπαίδευσης, αυτοί οι κατηγοριοποιητές δημιουργούν μια δομή δέντρου που κατηγοριοποιεί νέα στιγμιότυπα. Άλλοι πρόθυμοι κατηγοριοποιητές βασίζονται σε τεχνητά νευρωνικά δίκτυα [7, 8]. Ένα νευρωνικό δίκτυο εκπαιδεύεται πρώτα από τα στιγμιότυπα εκπαίδευσης και στη συνέχεια εκτελεί την διαιδκασία κατηγοριοποίησης. Οι κατηγοριοποιητές που βασίζονται σε πιθανότητες (probabilistic classifiers) ανήκουν επίσης στην κατηγορία των πρόθυμων κατηγοριοποιητών. Δημιουργούν ένα μοντέλο που βασίζεται σε πιθανότητες. Ένα χαρακτηριστικό παράδειγμα πιθανοτικού κατηγοριοποιητή είναι ο αφελής (naive) κατηγοριοποιητής Bayes [9, 10]. Μια άλλη υποκατηγορία των πρόθυμων αλγόριθμων περιλαμβάνει τους κατηγοριοποιητές που βασίζονται σε κανόνες συσχέτισης [11]. Ανακαλύπτουν τους κανόνες σύσχέτισης με βάση τα διαθέσιμα δεδομένα εκπαίδευσης. Οι κανόνες αυτοί χρησιμοποιούνται για σκοπούς κατηγοριοποίησης. Από την άλλη, η κατηγορία των οκνηρών κατηγοριοποιητών περιλαμβάνουν το γνωστό κατηγοριοποιητή των Κ Εγγύτερων Γειτόνων (KNN) [12] ο οποίος είναι το πιο χαρακτηριστικό παράδειγμα αυτής της κατηγορίας.

Σήμερα, οι μέθοδοι κατηγοριοποίησης πολλαπλών ετικετών (Multilabel Classification Methods) έχουν προταθεί και έχουν εφαρμογή σε πολλά προβλήματα όπως κατηγοριοποίηση πρωτεϊνικής λειτουργίας, κατηγοριοποίηση στη μουσική, κατηγοριοποίηση κειμένων, κατηγοριοποίηση φωτογραφιών, κατηγοριοποίηση αρχείων ήχου και βίντεο κ.τ.λ. Σε αυτού του είδους τα προβλήματα και σε αντίθεση με τα "συμβατικά" προβλήματα κατηγοριοποίησης, είναι πιθανόν ένα στιγμιότυπο να ανήκει ταυτόχρονα σε πολλές κατηγορίες. Σε αυτού του είδους τα προβλήματα, οι κατηγορίες ονομάζονται ετικέτες.

1.2 Κατηγορίες προβλημάτων κατηγοριοποίησης

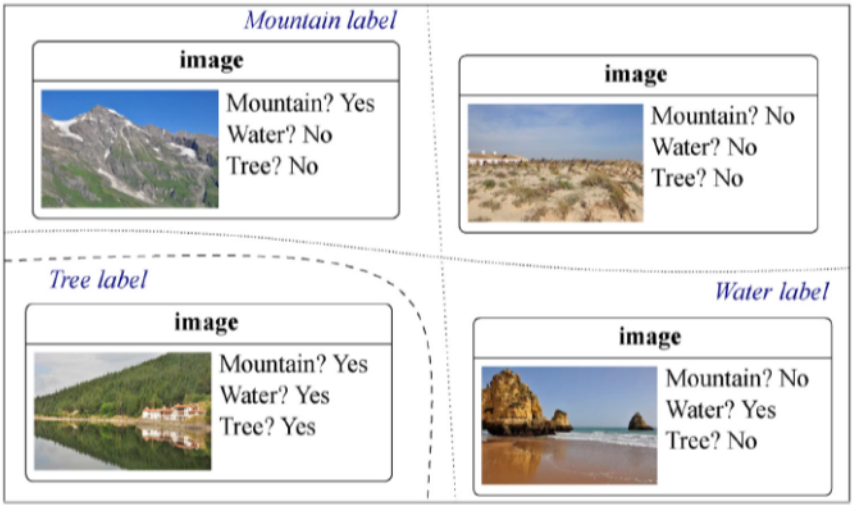
Η παραδοσιακή κατηγοριοποίηση μονής ετικέτας αφορά τη μάθηση από ένα σύνολο δεδομένων, που κάθε ένα από αυτά, ανήκει σε μια κατηγορία ετικετών L, από ένα σύνολο διαφορετικών ετικετών L, $L > 1$. Αν το σύνολο των ετικετών είναι $|L| = 2$, τότε το πρόβλημα χαρακτηρίζεται ως δυαδικό πρόβλημα κατηγοριοποίησης (ή φιλτράρισμα στην περίπτωση των δεδομένων κειμένου και web), ενώ αν $|L| > 2$, τότε ονομάζεται πρόβλημα κατηγοριοποίησης πολλών κατηγοριών. Παράδειγμα μονής κατηγοριοποίησης φαίνεται στα αριστερά του σχήματος 1.1 και κατηγοριοποίησης πολλών ετικετών στα δεξιά του ίδιου σχήματος.



Σχήμα 1.1: Παράδειγμα κατηγοριοποίησης μονής ετικέτας: Δυαδική κατηγοριοποίηση και κατηγοριοποίηση πολλών ετικετών. Οι διακεκομένες γραμμές παρουσιάζουν τα όρια μεταξύ των ετικετών.

Στην κατηγοριοποίηση πολλαπλών ετικετών, τα στιγμιότυπα συσχετίζονται με ένα σύνολο ετικετών $Y \subseteq L$. Στο παρελθόν, η κατηγοριοποίηση πολλαπλών ετικετών εφαρμόστηκε κυρίως για την κατηγοριοποίηση κειμένου και στα προβλήματα ιατρικής διάγνωσης. Τα έγγραφα κειμένου ανήκουν συνήθως σε περισσότερες από μία εννοιολογικές ετικέτες (κλάσεις). Για παράδειγμα, ένα άρθρο εφημερίδας σχετικά με τις αντιδράσεις της χριστιανικής εκκλησίας για την απελευθέρωση της ταινίας ”Da Vinci Code” μπορεί να κατηγοριοποιηθεί και στις δύο κατηγορίες (ετικέτες) Κοινωνία-Θρησκεία και Τέχνες-Ταινίες. Ομοίως στην ιατρική διάγνωση, ένας ασθενής μπορεί να κατηγοριοποιηθεί ταυτόχρονα και ως υπερτασικός και ως διαβητικός. Επομένως, υπάρχουν παραδείγματα προβλημάτων κατηγοριοποίησης, όπου υπάρχουν πολλές ετικέτες και ένα στιγμιότυπο μπορεί να διαθέτει περισσότερες από μια ετικέτες.

Σήμερα, παρατηρείται ότι οι μέθοδοι κατηγοριοποίησης πολλαπλών ετικετών είναι αναγκαίες όλο και περισσότερο στις σύγχρονες εφαρμογές, όπως η κατηγοριοποίηση πρωτεϊνικών λειτουργιών [13], μουσική κατηγοριοποίηση [14] και σημασιολογική κατηγοριοποίηση [15]. Στην σημασιολογική κατηγοριοποίηση, μια φωτογραφία μπορεί να ανήκει σε περισσότερες από μία εννοιολογικές ετικέτες, όπως ηλιοβασίλεμα και παραλία ταυτόχρονα. Ομοίως, στην κατηγοριοποίηση μουσικής ένα τραγούδι μπορεί να ανήκει σε περισσότερα από ένα είδη. Για παράδειγμα, πολλά τραγούδια του δημοφιλούς ροκ συγκροτήματος Scorpions μπορούν να χαρακτηριστούν τόσο ως ροκ όσο και ως μπαλάντες. Παράδειγμα στο σχήμα 1.2 όπου υπάρχουν τέσσερα στιγμιότυπα (φωτογραφίες) και τρείς ετικέτες: νερό, βουνό και δέντρα. Κάθε φωτογραφία ανήκει σε περισσότερες από μια ετικέτες.



Σχήμα 1.2: Παράδειγμα προβλήματος Πολλαπλών Ετικετών.Οι διακεκομένες γραμμές παρουσιάζουν τα όρια μεταξύ των ετικετών.

1.3 Κατηγοριοποιητής KNN

Ο αλγόριθμος των Κ εγγύτερων γειτόνων (KNN) είναι ένας απλός, εύκολος στην εφαρμογή αλγόριθμος κατηγοριοποίησης που μπορεί να χρησιμοποιηθεί για την επίλυση τόσο προβλημάτων κατηγοριοποίησης όσο και προβλημάτων παλινδρόμησης (regression). Αποτελεί αλγόριθμο λήψης αποφάσεων για τη Μηχανική Μάθηση και τη διαδικασία εξόρυξης γνώσης από δεδομένα. Θεωρείται ως ένας απο τους πιο αποτελεσματικούς αλγόριθμους στη μηχανική μάθηση, και μια από τις δέκα κορυφαίες μεθόδους εξόρυξης δεδομένων [16].

Σε ένα πρόβλημα κατηγοριοποίησης, για κάθε στιγμιότυπο x που πρέπει να κατηγοριοποιηθεί, ο αλγόριθμος KNN αναζητάει και ανακτάει τα k εγγύτερα στιγμιότυπα εκπαίδευσης στο x. Στη συνέχεια, το x κατηγοριοποιείται στην πλειοψηφούσα ετικέτα των k εγγύτερων στιγμιοτύπων.

Ο κατηγοριοποιητής KNN είναι εύκολα διαχειρίσιμος για k = 1 και για απεριόριστα στοιχεία το ποσοστό σφάλματος, είναι ασυμπτωτικά πολύ χειρότερο από το διπλάσιο του ελάχιστου δυνατού, που είναι το ποσοστό των Bayes. Δεδομένου ότι ο κατηγοριοποιητής KNN είναι ένας οκνηρός αλγόριθμος κατηγοριοποίησης, δεν χτίζει κάποιο μοντέλο κατηγοριοποίησης. Συνεπώς, το υπολογιστικό κόστος που απαιτείται για την εκπαίδευση είναι μηδενικό. Ο αλγόριθμος εξετάζει τα δεδομένα εκπαίδευσης κάθε φορά που ένα νέο στιγμιότυπο πρέπει να κατηγοριοποιηθεί.

Ένα πρόβλημα κατηγοριοποίησης έχει μια διακριτή τιμή σαν αποτέλεσμα. Για παράδειγμα ”του αρέσει ο ανανάς στην πίτσα” και ”δεν του αρέσει ο ανανάς στην πίτσα” είναι διακριτά. Δεν υπάρχει ενδιάμεση κατάσταση. Ο Πίνακας 1.1 εμφανίζει με ένα απλό παράδειγμα πως είναι τα δεδομένα εκπαίδευσης σε ένα πρόβλημα κατηγοριοποίησης. Έχουμε ένα γνώρισμα και μια ετικέτα. Στον πίνακα, μπορεί να προσπαθούμε να προβλέψουμε αν κάποιος θέλει ανανά (1) στην πίτσα του ή όχι (0) με βάση την ηλικία του.

Είναι συνήθης πρακτική να αντιπροσωπεύεται η έξοδος (ετικέτα) ενός αλγορίθμου κατηγοριο-

ποίησης ως ακέραιος αριθμός όπως 1, -1 ή 0. Σε αυτήν την περίπτωση, αυτοί οι αριθμοί είναι καθαρά αντιπροσωπευτικοί. Μαθηματικές πράξεις δεν θα πρέπει να εκτελούνται σε αυτά, διότι κάτι τέτοιο θα ήταν άνευ σημασίας.

Ένα πρόβλημα παλινδρόμησης έχει έναν πραγματικό αριθμό ως αποτέλεσμα. Για παράδειγμα, θα μπορούσαν να χρησιμοποιηθούν τα δεδομένα στον παρακάτω πίνακα 1.2 για να εκτιμηθεί το βάρος κάποιου ανθρώπου δεδομένου του ύψους του [16].

Τα δεδομένα που χρησιμοποιούνται σε μια περίπτωση παλινδρόμησης θα μοιάζουν με τα δεδομένα που εμφανίζονται στον Πίνακα 1.2, όπου παρουσιάζεται, μια ανεξάρτητη μεταβλητή (ή ένα σύνολο ανεξάρτητων μεταβλητών) και μια εξαρτημένη μεταβλητή, που τυγχάνει πρόβλεψης. Για παράδειγμα, ας θεωρηθεί, ότι το ύψος είναι η ανεξάρτητη μεταβλητή και το βάρος είναι η εξαρτημένη μεταβλητή. Επίσης, κάθε γραμμή ονομάζεται συνήθως παράδειγμα, στιγμιότυπο, παρατήρηση ή σημείο δεδομένων, ενώ κάθε στήλη (μη συμπεριλαμβανομένης της ετικέτας / εξαρτημένης μεταβλητής) συχνά ονομάζεται δείκτης πρόβλεψης, διάσταση, ανεξάρτητη μεταβλητή ή χαρακτηριστικό [16].

Ο αλγόριθμος κατηγοριοποίησης KNN υποθέτει ότι παρόμοια στιγμιότυπα υπάρχουν σε κοντινή απόσταση. Με άλλα λόγια, παρόμοια στιγμιότυπα είναι κοντά το ένα στο άλλο. Παρατηρήστε στην Εικόνα 1.3 ότι τις περισσότερες φορές, παρόμοια σημεία δεδομένων είναι κοντά το ένα στο άλλο. Ο αλγόριθμος KNN εξαρτάται από το κατα πόσο ισχύει αυτή η υπόθεση ώστε ο αλγόριθμος να είναι χρήσιμος. Ο αλγόριθμος κατηγοριοποίησης KNN βασίζεται στην ιδέα της ομοιότητας (μερικές φορές ονομάζεται απόσταση, ή εγγύτητα) με βάση τον υπολογισμό της απόστασης μεταξύ των σημείων σε ένα γράφημα. Υπάρχουν και άλλοι τρόποι υπολογισμού της απόστασης, και ένας τρόπος μπορεί να είναι καλύτερος απο έναν άλλο, ανάλογα με το πρόβλημα προς επίλυση. Ωστόσο, η ευθεία απόσταση γραμμής (που ονομάζεται και ευκλείδεια απόσταση) είναι η πιο δημοφιλής επιλογή [16].

Για $k = 1$, ο κατηγοριοποιητής KNN είναι επίσης γνωστός ως κατηγοριοποιητής εγγύτερου γείτονα (ή κανόνας 1-NN). Ο κατηγοριοποιητής 1NN, εκχωρεί σε ένα άγνωστο στιγμιότυπο x, την ετικέτα του εγγύτερου στιγμιότυπου εκπαίδευσης σύμφωνα με τη μετρική απόστασης (π.χ ευκλείδια απόσταση). Ο κατηγοριοποιητής KNN, με το $k > 1$, είναι μια γενίκευση της προσέγγισης 1NN όπου η ετικέτα του στιγμιοτύπου x ορίζεται ως ίση με την ετικέτα που αντιπροσωπεύεται απο την πλειοψηφία των k εγγύτερων γείτονων της, στο σύνολο εκπαίδευσης.

Πίνακας 1.1: Δεδομένα προβλήματος κατηγοριοποίησης

Ηλικία	Αρέσει ο ανανάς στην πίτσα
42	1
65	0
50	1
76	1
96	0
50	1
46	0

Πίνακας 1.2: Δεδομένα προβλήματος παλινδρόμησης

Ύψος(Inches)	Βάρος(Pounds)
65.78	112.99
71.52	136.49
69.40	153.03
68.22	142.34
67.79	144.30
68.70	123.30
69.80	141.49

Το σχήμα 1.4 απεικονίζει ένα δισδιάστατο παράδειγμα της διαδικασίας κατηγοριοποίησης με τον κατηγοριοποιητή KNN. Πιο συγκεκριμένα, παρουσιάζει ένα σύνολο δεδομένων με δύο ετικέτες, τετράγωνα και κύκλους και ένα νεο στιγμιότυπο (Q) που πρέπει να κατηγοριοποιηθεί σε μία από αυτές τις ετικέτες. Αν οριστεί το k να είναι ίσο με τρία (συνεχόμενη γραμμή κύκλου στο σχήμα 1.4), το Q κατηγοριοποιείται στην ετικέτα κύκλος, επειδή δύο από τους τρεις εγγύτερους γείτονες του είναι κύκλοι. Από την άλλη, εάν οριστεί το k να είναι ίσο με πέντε (διακεκομμένη γραμμή κύκλου στο σχήμα 1.4), τότε το στιγμιότυπο Q κατηγοριοποιείται στην ετικέτα τετράγωνο, επειδή τρείς απο τους πέντε εγγύτερους γείτονες του είναι τετράγωνα [12].

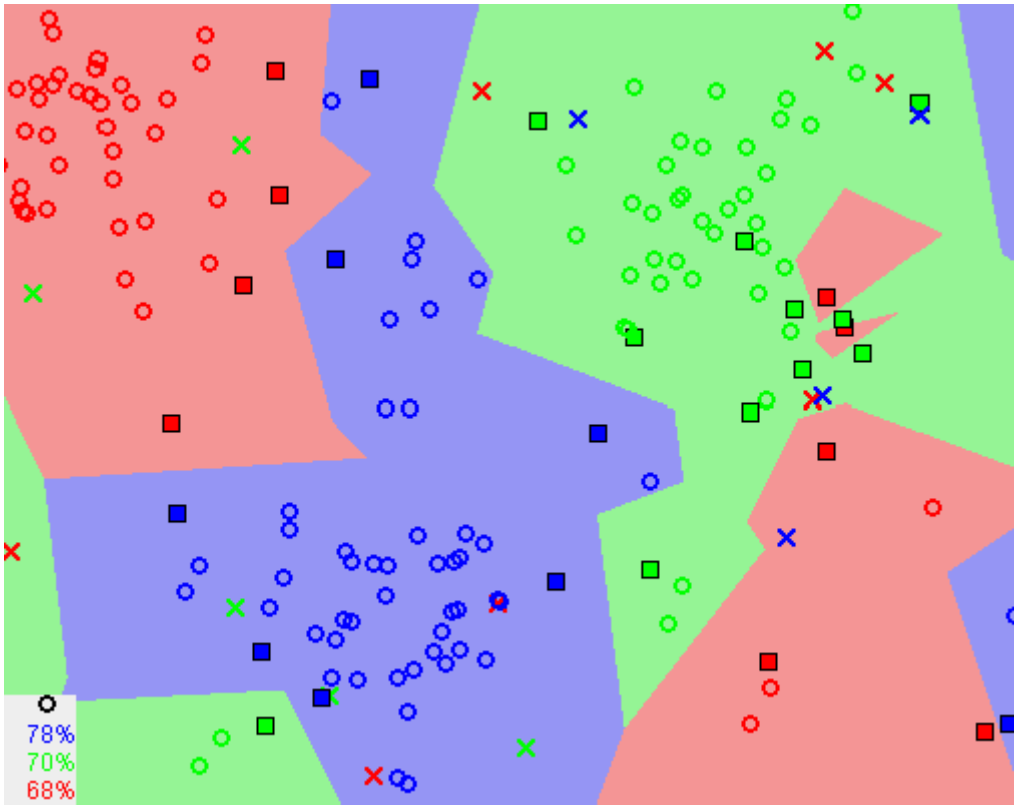
Η απόδοση της κατηγοριοποίησης του Αλγορίθμου 1 εξαρτάται σίγουρα από την επιλογή της τιμής της παραμέτρου K. Η τιμή του K που επιτυγχάνει την υψηλότερη ακρίβεια κατηγοριοποίησης εξαρτάται από το σύνολο δεδομένων που χρησιμοποιείται και ο προσδιορισμός του, συνεπάγεται συνήθως τη ρύθμιση μέσω χρονοβόρων εργασιών προεπεξεργασίας δοκιμών και σφαλμάτων (trial-and-error). Αν και ο προσδιορισμός του K δεν μπορεί να ακολουθήσει κανένα γενικό κανόνα και το ”καλύτερο” K μπορεί να είναι εντελώς διαφορετικό για διαφορετικά σύνολα δεδομένων, οι μεγαλύτερες τιμές του K είναι κατάλληλες για σύνολα δεδομένων με θόρυβο, δεδομένου ότι εξετάζουν μεγαλύτερες γειτονιές. Όμως έτσι, δεν καθορίζονται σαφώς τα όρια μεταξύ των διακριτών ετικετών. Αντίθετα, οι μικρές τιμές της παραμέτρου K κάνουν τον κατηγοριοποιητή περισσότερο ευαίσθητο στο θόρυβο. Ως εκ τούτου, σε περιπτώσεις δεδομένων εκπαίδευσης που περιέχουν θόρυβο, η κατηγοριοποίηση είναι λιγότερο ακριβής.

Algorithm 1 KNN

Input: X:training data, Y:class labels of X,x:unknown sample

1: **for** i = 1 to || **do**
2: Compute distance $d(X_i, x)$
3: **end for**
4: Compute set I containing indices for the k smallest distances $d(X_i, x)$
5: **return** majority label for $\{Y_i$ where $i \in I\}$

Αξίζει να σημειωθεί ότι ακόμη και η καλύτερη τιμή του K μπορεί να μην είναι η βέλτιστη. Αυτό συμβαίνει επειδή ο κατηγοριοποιητής KNN χρησιμοποιεί μια μονο τιμή K για όλα τα δεδομένα που πρόκειται να κατηγοριοποιηθούν. Διαφορετικές τιμές K μπορεί να είναι βέλτιστες για διαφορετικές περιοχές του χώρου δεδομένων. Κατά συνέπεια, ευρετικές μέθοδοι για δυναμική αποφαση της τιμής του K μπορούν να υιοθετηθούν και να επιτύχουν υψηλότερη ακρίβεια από



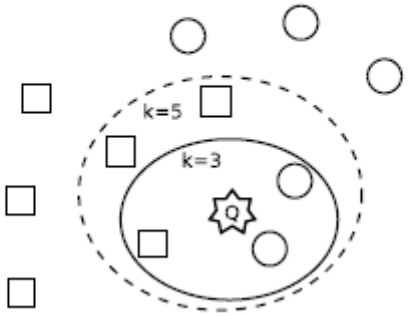
Σχήμα 1.3: Παράδειγμα εγγύτητας στιγμιότυπων ενός προβλήματος

τον κατηγοριοποιητή KNN με ”καλύτερο” προσδιορισμό της τιμής του K [17, 18].

Σε περιπτώσεις δυαδικών προβλήματων κατηγοριοποίησης (σύνολα δεδομένων με δύο ετικέτες), το K θα πρέπει να οριστεί σε περιττή τιμή για την αποφυγή ισοψηφιών (και οι δύο ετικέτες να είναι οι πιο κοινές) κατά τη διάρκεια της ψηφοφορίας των εγγύτερων γειτόνων. Σε περιπτώσεις μη δυαδικών προβλημάτων, το K μπορεί να έχει οποιαδήποτε τιμή. Σε αυτή την περίπτωση, πιθανές ισοψηφίες κατά τη διάρκεια της ψηφοφορίας, επιλύονται επιλέγοντας είτε μια τυχαία ”πιο κοινή” κλάση, είτε την κλάση του εγγύτερου γείτονα. Το δημοφιλές λογισμικό Weka [19] και πολλά άλλα εργαλεία λογισμικού δεδομένων εξόρυξης / μηχανικής μάθησης, επιλύουν τις περιπτώσεις ισοψηφίας των ετικετών, με τυχαίο τρόπο.

Ένα άλλο σημαντικό ζήτημα που πρέπει να αντιμετωπιστεί είναι η επιλογή της κατάλληλης μετρικής για τον υπολογισμό της απόστασης μεταξύ των στιγμιότυπων. Ασφαλώς, η απόφαση αυτή θα πρέπει να λάβει υπ’ όψιν τους τύπους δεδομένων των χαρακτηριστικών του συνόλου δεδομένων (μεταβλητές). Σε περιπτώσεις πραγματικών ή/και ακέραιων χαρακτηριστικών, η ευκλείδεια απόσταση (Euclidean distance) είναι η συνήθης χρησιμοποιούμενη μέτρηση απόστασης. Ωστόσο, άλλες μετρικές απόστασης μπορεί να υιοθετηθούν (π.χ.Mahalanobis, Manhatan, Minkowski, Chebyshev) [20].

Πολλές άλλες παρόμοιες μετρικές έχουν προταθεί για τη διαχείριση ονομαστικών χαρακτηριστικών (μη μετρικών χώρων). Ωστόσο, όλα τα πειράματα σε αυτή τη διπλωματική εργασία διεξάγονται σε σύνολα δεδομένων με πραγματικούς ή ακέραιους αριθμούς στα χαρακτηριστικά. Ως εκ τούτου, υιοθετείται η ευκλείδεια απόσταση ως μέτρηση απόστασης. Κατά συνέπεια τα



Σχήμα 1.4: Διαδικασία κατηγοριοποίησης με KNN για K=3 και K=5

στιγμιότυπα που περιγράφονται από (n) χαρακτηριστικά, θεωρούνται σημεία δεδομένων (ή διανύσματα) στον n-διάστασεων ευκλείδειο μετρικό χώρο, και η ευκλείδεια απόσταση μεταξύ των σημείων p και q δίνεται από τον τύπο 1.1.

$$d(p,q) = d(q,p) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \ldots + (q_n - p_n)^2} = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2} \qquad (1.1)$$

Μια πολλή σημαντική λεπτομέρεια είναι οτι διαφορετικές τιμές χαρακτηριστικών στα στιγμιότυπα μπορούν να επηρεάσουν την τιμή απόστασης. Τα χαρακτηριστικά με μεγαλύτερο εύρος τιμών έχουν μεγαλύτερη βαρύτητα στη μέτρηση της ευκλείδιας απόστασης, από τα χαρακτηριστικά με μικρότερο εύρος τιμών. Ας θεωρηθεί ότι το χαρακτηριστικό ”μισθός” κυμαίνεται από 800 έως 5000 και το χαρακτηριστικό ”αριθμός των παιδιών” λαμβάνει τιμές από το 0 έως το 6.

Παρόλο που και τα δύο χαρακτηριστικά έχουν την ίδια σημασία, ο ”μισθός” έχει μεγαλύτερο αντίκτυπο στον υπολογισμό απόστασης από τον ”αριθμό παιδιών”. Επομένως, το εύρος τιμών των χαρακτηριστικών θα πρέπει να κανονικοποιηθεί σε ένα συγκεκριμένο αριθμητικό διάστημα (π.χ. [0, 1]). Ας θεωρηθεί ότι ένα σύνολο δεδομένων περιέχει (n) στιγμιότυπα και ένα χαρακτηριστικό (e) πρέπει να κανονικοποιηθεί στο διάστημα [0,1]. Η τιμή του χαρακτηριστικού του i-στου στιγμιότυπου $i = 1, ...n$ κανονικοποιείται ως εξής:

$$normalized(e_i) = \frac{e_i - E_{min}}{E_{max} - E_{min}} \qquad (1.2)$$

όπου το E_{min} και E_{max} είναι οι ελάχιστες και οι μέγιστες τιμές για το χαρακτηριστικό (e) αντίστοιχα. Η κανονικοποίηση δεδομένων είναι μια κοινή διαδικασία προεπεξεργασίας σε πολλές εργασίες εξόρυξης δεδομένων [12].

Αξίζει να σημειωθεί ότι ο αλγόρθμος KNN είναι κατάλληλος τόσο για προβλήματα κατηγοριοποίησης μονής ετικέτας όσο και για προβλήματα κατηγοριοποίησης πολλαπλών ετικετών. Αυτο συμβαίνει επειδή ο KNN είναι οκνηρός κατηγοριοποιητής και δεν χτίζει κάποιο μοντέλο κατηγοριοποίησης. Η διαδικασία αναζήτησης εγγύτερων γειτόνων δεν είναι διαφορετική σε ένα πρόβλημα μονής ετικέτας από ότι σε ένα πρόβλημα πολλαπλών ετικετών. Κατά τη διαδικασία της ”ψηφοφορίας” για την ανάδειξη της πλειοψηφούσας ετικέτας, οι K εγγύτεροι γείτονες, οι

οποίοι είναι ίδιοι είτε σε ένα πρόβλημα πολλαπλών ετικετών είτε όχι, αποφασίζουν για το ποιες ετικέτες θα ανατεθούν στο προς κατηγοριοποίηση στιγμιότυπο.

Αν χρησιμοποιηθεί ο μετασχηματισμός προβλήματος Binary Relevance (βλέπε Κεφάλαιο 2), τα δεδομένα εκπαίδευσης πολλαπλών ετικετών μετασχηματίζονται με τέτοιον τρόπο ώστε να υπάρχουν τόσες στήλες όσες και οι ετικέτες και με ”1” να δηλώνεται η ύπαρξη αυτής της ετικέτας στο στιγμιότυπο, ενώ με ”0” να δηλώνεται η μη ύπαρξη. Στη συνέχεια, για κάθε ετικέτα δημιουργείται ένα μοντέλο κατηγοριοποίησης. Ο KNN είναι οκνηρός κατηγοριοποιητής. Δεν χτίζει κάποιο μοντέλο. Άρα, είναι κατάλληλος να συνδυαστεί με τον μετασχηματισμό Binary Relevance. Έτσι, αρκεί η μια και μοναδική αναζήτηση στα δεδομένα εκπαίδευσης για την ανάκτηση των εγγύτερων γειτόνων και στη συνέχεια, στο στάδιο της ψηφοφορίας, να θεωρηθεί οτι έχουμε τόσους κατηγοριοποιητές KNN, όσες και οι ετικέτες, όπου ο κάθε ένας αποφασίζει για την ανάθεση ή όχι της κάθε ετικέτας στο προς κατηγοριοποίηση στιγμιότυπο.

1.4 Μειονεκτήματα κατηγοριοποιητή KNN - Μέθοδοι αντιμετώπισης

Όπως ήδη αναφέρθηκε, τα πλεονεκτήματα του αλγορίθμου κατηγοριοποίησης KNN είναι, οτι είναι απλός και εύκολος στην εφαρμογή του. Δεν δημιουργεί μοντέλο. Ο αλγόριθμος είναι ευέλικτος και μπορεί να χρησιμοποιηθεί για προβλήματα κατηγοριοποίησης, παλινδρόμησης και αναζήτησης [21]. Ωστόσο, σε κάθε περίπτωση, ο κατηγοριοποιητής KNN παρουσιάζει διάφορα προβλήματα όπως μεγάλες απαιτήσεις αποθήκευσης, υψηλή υπολογιστική πολυπλοκότητα, για την αναζήτηση εγγύτερων γειτόνων και χαμηλή ανοχή στο θόρυβο και στα εσφαλμένα στιγμιότυπα [21].

Πιο αναλυτικά, ο κατηγοριοποιητής KNN πρέπει να υπολογίζει όλες τις αποστάσεις μεταξύ κάθε υπό κατηγοριοποίηση στιγμιότυπου και όλων των στιγμιοτύπων που είναι αποθηκευμένα στο σύνολο εκπαίδευσης. Σε περιπτώσεις μεγάλων συνόλων δεδομένων, αυτό το μειονέκτημα καθιστά τη χρήση του σε ορισμένες περιπτώσεις μια απαγορευτική διαδικασία.

Για παράδειγμα, αν θεωρηθεί ότι σύστημα κατηγοριοποίησης αποθηκεύει 100.000 στιγμιότυπα εκπαίδευσης. Επιπλέον, αν θεωρηθεί ότι το σύστημα θα πρέπει να κατηγοριοποιήσει περίπου 50.000 στιγμιότυπα με την εκτέλεση του κατηγοριοποιητή KNN. Αυτό σημαίνει ότι το σύστημα θα πρέπει να υπολογίσει πέντε δισεκατομμύρια αποστάσεις. Αν και σήμερα τα συστήματα υπολογιστών είναι εξοπλισμένα με ισχυρούς επεξεργαστές, αυτοί οι υπολογισμοί είναι χρονοβόροι και μη αποδεκτοί σε περιπτώσεις που υπάρχουν χρονικοί περιορισμοί. Θα πρέπει να αναφερθεί ότι, εκτός από το μέγεθος του συνόλου εκπαίδευσης, το υπολογιστικό κόστος της κατηγοριοποίησης εξαρτάται επίσης από τις διαστάσεις του συνόλου δεδομένων. Όσο μεγαλύτερη είναι η διάσταση των δεδομένων, τόσο περισσότερο υπολογιστικό κόστος απαιτείται για τον υπολογισμό απόστασης.

Μια άλλη αδυναμία του κατηγοριοποιητή KNN είναι οι μεγάλες απαιτήσεις σε χώρο αποθήκευσης, για τα δεδομένα εκπαίδευσης. Σε αντίθεση με τους πρόθυμους κατηγοριοποιητές που μπορούν να απορρίψουν τα δεδομένα εκπαίδευσης, μετά την κατασκευή του μοντέλου, ο κα-

τηγοριοποιητής KNN χρειάζεται τα δεδομένα εκπαίδευσης, να είναι πάντα διαθέσιμα. Κατά συνέπεια, ο κατηγοριοποιητής KNN πρέπει να εκτελείται σε συστήματα υπολογιστών που να διαθέτουν αρκετή κύρια μνήμη για την αποθήκευση των δεδομένων εκπαίδευσης.

Η τελευταία αδυναμία είναι ότι ο κατηγοριοποιητής KNN, όπως και πολλοί άλλοι αλγόριθμοι κατηγοριοποίησης, είναι αλγόριθμος ευαίσθητος στο θόρυβο. Ειδικότερα, η ακρίβεια εξαρτάται σε μεγάλο βαθμό από τα δεδομένα εκπαίδευσης. Θόρυβος και εσφαλμένα δεδομένα, καθώς και ακραίες τιμές και επικαλύψεις μεταξύ περιοχών δεδομένων διαφορετικών ετικετών, οδηγούν σε μικρότερη ακρίβεια. Η χρήση υψηλών τιμών K επεκτείνει την εξεταζόμενη γειτονιά και, ως εκ τούτου, μπορεί να διορθώσει εν μέρει αυτό το μειονέκτημα. Ωστόσο αυτή η τεχνική συνεπάγεται μεγάλο αριθμό εκτελέσης δοκιμών και σφαλμάτων (trial-and-error) για τον προσδιορισμό της κατάλληλης τιμής K [12].

Οι αδυναμίες αυτές μπορεί να καταστήσουν τη χρήση του, αναποτελεσματική. Στα προβλήματα κατηγοριοποίησης μονής ετικέτας, οι παραπάνω αδυναμίες, αντιμετωπίζονται υιοθετώντας κάποια τεχνική μείωσης δεδομένων εκπαίδευσης ή κάποια μέθοδος δεικτοδότησης (βλέπε Κεφάλαιο 2). Οι τεχνικές μείωσης δεδομένων παράγουν ή επιλέγουν πρότυπα, τα οποία αντιπροσωπεύουν τα πολλά σε αριθμό, στιγμιότυπα του συνόλου εκπαίδευσης. Αντίθετα, οι μέθοδοι δεικτοδότησης, κατασκευάζουν μια δομή δεδομένων που είναι ικανή να επιτύχει την ανάκτηση εγγύτερων γειτόνων χωρίς να υπολογίζονται όλες οι πιθανές αποστάσεις, μεταξύ του, υπό κατηγοριοποίηση στιγμιοτύπου και όλων των δεδομένων εκπαίδευσης.

1.5 Κίνητρο και Συνεισφορά

Οι αδυναμίες που αναφέρθηκαν παραπάνω υφίστανται με τον ίδιο ακριβώς τρόπο τόσο σε προβλήματα μονής ετικέτας όσο και σε σύνολα δεδομένων πολλαπλών ετικετών. Οι αδυναμίες αυτές που αναφέρθηκαν αποτελούν ενεργό ερευνητικό πρόβλημα και έχουν προσελκύσει την ερευνητική κοινότητα, εξόρυξης δεδομένων και της Μηχανικής Μάθησης. Αποτέλεσμα είναι το να έχουν προταθεί διάφοροι μέθοδοι για γρήγορη κατηγοριοποίηση κοντινότερων γειτόνων οι οποίες είναι εφαρμόσιμες σε προβλήματα κατηγοριοποίησης μονής ετικέτας.

Όπως αναφέρθηκε ήδη, τα προβλήματα αυτά μπορούν να επιλυθούν χρησιμοποιώντας μια τεχνική μείωσης δεδομένων που είναι διαθέσιμη στη βιβλιογραφία. Υπάρχουν πολλές τεχνικές μείωσης δεδομένων διαθέσιμες στη βιβλιογραφία για προβλήματα κατηγοριοποίησης. Οι τεχνικές αυτές είτε επιλέγουν πρότυπα (αντιπροσωπευτικά στιγμιότυπα) (Prototype Selection) είτε δημιουργούν πρότυπα (Prototype Generation). Η συντριπτική πλειοψηφία των τεχνικών αυτών αφορά προβλήματα κατηγοριοποίησης μονής ετικέτας είτε αυτά είναι προβληματα κατηγοριοποίησης πολλών κατηγοριών είτε δυαδικά. Ελάχιστες ερευνητικές προσπάθειες έχουν πραγματοποιηθεί που να αφορούν τη μείωση δεδομένων εκπαίδευσης πολλαπλών ετικετών, δηλαδή στιγμιοτύπων που να ανήκουν σε περισσότερες από μια ετικέτες. Ωστόσο, η απόδοση των τεχνικών μείωσης δεδομένων πολλαπλών ετικετών που έχουν προταθεί εξαρτάται σε μεγάλο βαθμό από παραμέτρους που προσδιορίζει ο χρήστης μέσω υπολογιστικά κοστοβόρων διαδικασιών.

Επιπρόσθετα, η χρήση μεθόδων μετασχηματισμού προβλήματος (π.χ. Binary Relevance) η οποία θεωρητικά θα καθιστούσε δυνατή τη χρήση τεχνικών μείωσης δεδομένων μονής ετικέτας, πρακτικά είναι ανευ ουσίας αφού είτε απαιτείται η δημιουργία τόσων μειωμένων (συμπυκνωμένων) συνόλων όσες οι ετικέτες στο πρόβλημα, είτε η μείωση που επιτυγχάνεται δεν είναι μεγάλη.

Αυτές οι παρατηρήσεις αποτελούν το κίνητρο για την εκπόνηση της παρούσας διπλωματικής εργασίας. Η παρούσα διπλωματική εργασία εκπονήθηκε με σκοπό να βοηθήσει σε δυο βασικούς τομείς που αφορούν τα δεδομένα πολλαπλών ετικετών μιας και η βιβλιογραφία που υπάρχει είναι περιορισμένη, λόγω της φύσης του προβήματος και της πολυπλοκότητάς του. Πρώτον, την υλοποίηση νέων αλγορίθμων κατηγοριοποίησης στο χώρο των δεδομένων πολλαπλών ετικετών και δευτερον, στην υλοποίηση νέων μεθόδων για την μείωση των αρχικών δεδομένων (κέρδος σε χρόνο και μνήμη). Όλα αυτά βέβαια θα πρέπει να επιτευχθούν χωρίς μείωση της ακρίβειας, αλλά με ταυτόχρονη μείωση του χρόνου εκτέλεσης.

Συγκεκριμένα η παρούσα διπλωματική εργασία συνεισφέρει στην ανάπτυξη νέων τεχνικών μείωσης δεδομένων εκπαίδευσης πολλαπλών ετικετών που δεν περιλαμβάνουν παραμέτρους. Για να επιτευχθεί ο στόχος χρησιμοποιήθηκε η βασική λειτουργία του αλγορίθμου συσταδοποίησης K-means, ο οποίος όμως εκτελείται επαναληπτικά στις μη ομοιογενείς συστάδες που παράγονται. Στα σύνολα πολλαπλών ετικετών, μια συστάδα θεωρείται ομοιογενής όταν όλα τα στιγμιότυπα της συστάδας έχουν τουλάχιστον μια κοινή ετικέτα. Στο τέλος της διαδικασίας όλες οι συστάδες γίνονται ομοιογενείς και τα κέντρα των συστάδων αποτελούν τα πρότυπα που συνθέτουν το μειωμένο σύνολο δεδομένων. Με βάση αυτή τη λειτουργία επαναληπτικής συσταδοποίησης δημιουργήθηκαν δυο τεχνικές μείωσης δεδομένων που ανήκουν στην κατηγορία παραγωγής προτύπων. Οι τεχνικές που αναπτύχθηκαν ονομάστηκαν **MRHC1** και **MRHC2** και παράγουν αντιπροσωπευτικά στιγμιότυπα του αρχικού συνόλου, μειώνοντας έτσι σε μεγάλο βαθμό το αρχικό σύνολο δεδομένων σε προβλήματα πολλαπλών ετικετών. Επίσης, στα πλαίσια της παρούσας διπλωματικής εργασίας αναπτύχθηκαν παραλλαγές του γνωστού αλγορίθμου των K εγγύτερων γειτόνων (**KNN**). Οι παραλλαγές ονομάστηκαν **MKNN1** και **MKNN2**, και χρησιμοποιήθηκαν για να επιτευχθεί αποτελεσματική κατηγοριοποίηση σε σύνολα δεδομένων πολλαπλών ετικετών που έχουν παραχθεί από τεχνικές μείωσης δεδομένων.

1.6 Οργάνωση Διπλωματικής Εργασίας

Τα υπόλοιπα τμήματα της παρούσας διπλωματικής εργασίας είναι οργανωμένα ως ακολούθως: Το κεφάλαιο 2 παρουσιάζει τις μεθόδους μετασχηματισμού προβλημάτων πολλαπλών ετικετών, τις τεχνικές μείωσης δεδομένων σε σύνολα πολλαπλών ετικετών (επιλογή προτύπων, παραγωγή προτύπων) και αναλυτική περιγραφή του αλγορίθμου **RHC** [22] (μονής ετικέτας) που αποτέλεσε και τη βάση για την ανάπτυξη των αλγορίθμων μείωσης δεδομένων πολλαπλών ετικέτων που προτείνονται από αυτή την εργασία.

Το κεφάλαιο 3 κάνει μια ανασκόπηση της πρόσφατης βιβλιογραφίας σχετικά με τεχνικές μείωσης δεδομένων και κατηγοριοποίησης σε σύνολα πολλαπλών ετικετών.

Το κεφάλαιο 4 αναλύει διεξοδικά τους δυο προτεινόμενους αλγορίθμους κατηγοριοποίησης πολλαπλών ετικετών, τους **MKNN1** και **MKNN2** οι οποίοι αναπτύχθηκαν για να μπορέσουν να κατηγοριοποιήσουν τέτοιου είδους δεδομένα με όσο γίνεται με πιο αποτελεσματικό τρόπο. Επίσης, παρουσιάζει δυο νέους αλγόριθμους μείωσης δεδομένων πολλαπλών ετικετών **MRHC1** και **MRHC2** με τους οποίους επιτυγχάνεται μεγάλη μείωση των αρχικών δεδομένων εκπαίδευσης (prototype generation).

Το κεφάλαιο 5 παρουσιάζει τα σύνολα δεδομένων που χρησιμοποιήθηκαν στην πειραματική μελέτη που εκπονήθηκε, τη διαδικασία που ακολουθήθηκε για την υλοποίηση της πειραματικής μελέτης και τις μετρικές που χρησιμοποιήθηκαν για την αξιολόγηση των αποτελεσμάτων.

Τέλος, το κεφάλαιο 6 παρουσιάζει χρήσιμα συμπεράσματα που βασίζονται στα αποτελέσματα των πειραμάτων και δίνει κάποιες κατευθύνσεις για μελλοντική έρευνα στο συγκεκριμένο επιστημονικό πεδίο.

2 Βασικές γνώσεις

Οι τεχνικές κατηγοριοποίησης πολλών ετικετών ομαδοποιούνται σε δύο βασικές κατηγορίες, τις μεθόδους μετασχηματισμού προβλήματος (problem transformation methods) και τις μεθόδους προσαρμογής αλγορίθμων (algorithm adaptation methods). Σε αυτό το κεφάλαιο παρουσιάζονται αυτές οι μέθοδοι, ο αλγόριθμος K μέσων και ο αλγόριθμος μείωσης δεδομένων μονής ετικέτας Reduction through Homogeneous Clustering (RHC) οι οποίοι αποτελούν τη βάση των αλγορίθμων μείωσης δεδομένων πολλαπλών ετικετών που προτείνονται από την παρούσα διπλωματική εργασία.

2.1 Μέθοδοι Μετασχηματισμού Προβλήματος (Problem Transformation Methods)

Οι μέθοδοι μετασχηματισμού προβλήματος μετατρέπουν την διαδικασία εκπαίδευσης κατά την κατηγοριοποίηση πολλών ετικετών σε μία ή περισσότερες διαδικασίες κατηγοριοποίησης μοναδικής ετικέτας [23]. Έπειτα, χρησιμοποιείται κάποιος γνωστός αλγόριθμος κατηγοριοποίησης και τα αποτελέσματα των προβλέψεων για μία ετικέτα, μετασχηματίζονται σε προβλέψεις πολλών ετικετών [24]. Το κύριο πλεονέκτημα αυτών των μεθόδων, είναι η ευελιξία τους, καθώς είναι ανεξάρτητες από τον αλγόριθμο που επιλέγεται. Ανάλογα με το πρόβλημα, κάποιοι κατηγοριοποιητές μπορεί να έχουν καλύτερη απόδοση από άλλους και υπάρχει περιθώριο επιλογής του κατάλληλου.

Η πρώτη και προφανής μέθοδος μετασχηματισμού προβλήματος είναι το να μετατρέψουμε το σύνολο δεδομένων πολλαπλών ετικετών σε σύνολο δεδομένων μιας ετικέτας επιλέγοντας για κάθε στιγμιότυπο του συνόλου εκπαίδευσης στην τύχη μια ετικέτα από αυτές που διαθέτει. Αν για παράδειγμα, μια ταινία είναι ταυτόχρονα Δράμα και Περιπέτεια, επιλέγουμε στη τύχη την μια απο τις δύο ετικέτες. Ένας δεύτερος και επίσης προφανής μετασχηματισμός είναι το να διαγράψουμε όλα τα στιγμιότυπα απο τα δεδομένα εκπαίδευσης που διαθέτουν περισσότερες απο μια ετικέτες. Αυτές οι δύο λύσεις οδηγούν στην απώλεια πληροφορίας. Συνεπώς, δεν θα ασχοληθούμε περαιτέρω στα πλαίσια αυτής της εργασίας.

Μια καλή λύση μετασχηματισμού είναι το να αντιγραφεί τόσες φορές ένα στιγμιότυπο πολλαπλών ετικετών όσες είναι και οι διαφορετικές ετικέτες που διαθέτει και κάθε αντίγραφο να διαθέτει μια διαφορετική ετικέτα. Η λύση αυτή αν και φαίνεται καλή, στην πράξη πολλαπλασιάζει σε μέγεθος το σύνολο δεδομένων ενώ παράλληλα οι τεχνικές μείωσης δεδομένων μονής ετικέτας, οδηγούνται σε μικρή μείωση δεδομένων αφού θεωρούν ότι τα αντίγραφα των στιγμιοτύπων βρίσκονται κοντά στα όρια απόφασης και έτσι παραμένουν πάντα στο συμπυκνωμένο σύνολο που κατασκευάζουν. Συνεπώς ο μετασχηματισμός αυτός δεν μπορεί να συνδυαστεί με τεχνικές μείωσης δεδομένων.

Η τεχνική Binary Relevance θεωρείται ο πιο δημοφιλής μετασχηματισμός. Βάσει αυτού του μετασχηματισμού, τα δεδομένα εκπαίδευσης πολλαπλών ετικετών μετασχηματίζονται με τέτοιον τρόπο ώστε να υπάρχουν τόσες στήλες όσες και οι ετικέτες και με ”1” να δηλώνεται η ύπαρξη

αυτής της ετικέτας στο στιγμιότυπο, ενώ με ”0” να δηλώνεται η μη ύπαρξη. Στη συνέχεια, για κάθε ετικέτα δημιουργείται ένα μοντέλο κατηγοριοποίησης. Όπως αναφέρθηκε στο Κεφάλαιο 1, αυτός ο μετασχηματισμός μπορεί να συνδυαστεί ιδανικά με τον κατηγοριοποιητή KNN επειδή ο KNN είναι οκνηρός και δεν χτίζει κάποιο μοντέλο. Ωστόσο, στην περίπτωση που απαιτείται μείωση δεδομένων, ο μετασχηματισμός απαιτεί, το να εκτελεστεί ο αλγόριθμος μείωσης δεδομένων μονής ετικέτας για κάθε ετικέτα και να δημιουργηθεί ένα συμπυκνωμένο σύνολο για κάθε ετικέτα. Η διαδικασία αυτή θα δημιουργήσει τόσα συμπυκνωμένα σύνολα όσες και οι ετικέτες, με αποτέλεσμα η μείωση δεδομένων τελικά, είτε να μην επιτυγχάνεται σε μεγάλο βαθμό, είτε τα δεδομένα τελικά να αυξάνονται.

Τέλος, ο μετασχηματισμός Label Powerset θεωρεί ότι ο κάθε πιθανός συνδυασμός των ετικετών αποτελεί μια ξεχωριστή ετικέτα. Ο μετασχηματισμός αυτός είναι εφαρμόσιμος μόνο όταν το πλήθος ετικετών και το πλήθος των συνδυασμών είναι μικρό και το σύνολο δεδομένων εκπαίδευσης μεγάλο και αυτό γιατί πρέπει να υπάρχει ικανός αριθμός στιγμιοτύπων για κάθε συνδυασμό ετικετών. Ο μετασχηματισμός αυτός μπορεί να χρησιμοποιηθεί σε συνδυασμό με κάποια τεχνική μείωσης δεδομένων μονής ετικέτας εφόσον ικανοποιούνται οι προαναφερθέντες προϋποθέσεις.

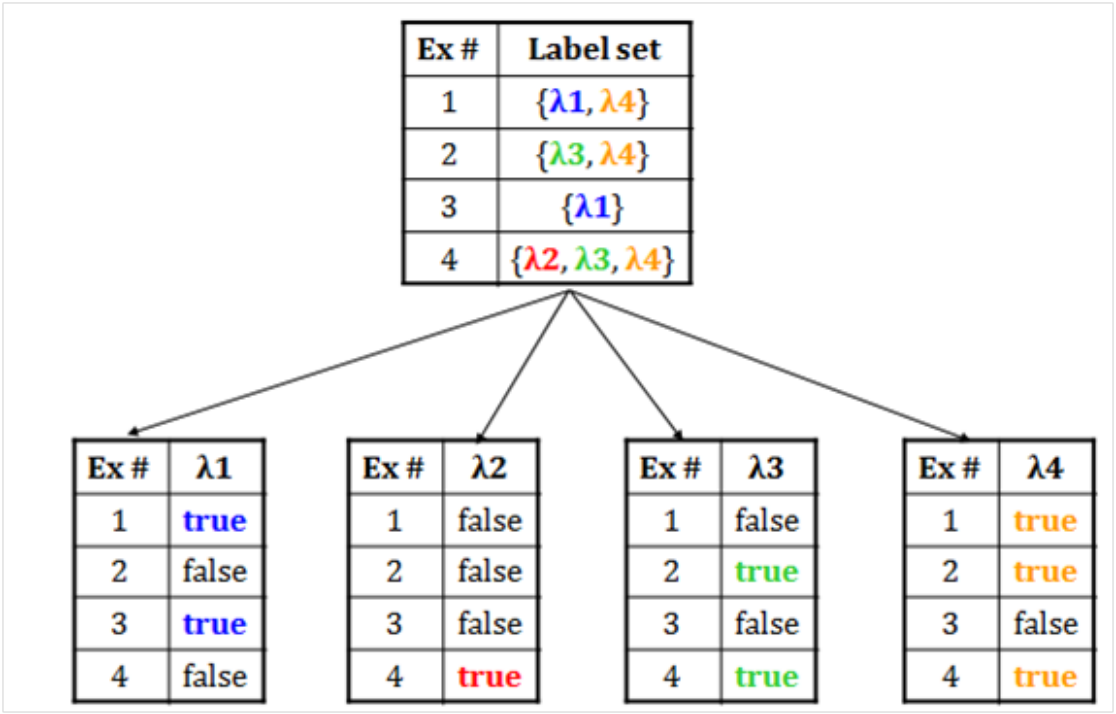
Ο μετασχηματισμός Random k-Labelsets είναι μια παραλλαγή του Label Powerset που προσπαθεί να αντιμετωπίσει τα μειονεκτήματα που αναφέρθηκαν. Ωστόσο, η αποτελεσματικότητα του μετασχηματισμού k-Labelsets εξαρτάται σε μεγάλο βαθμό από τις τυχαίες διασπάσεις που θα γίνουν.

Στη συνέχεια αυτού του κεφαλαίου περιγράφονται περαιτέρω οι ευρέως χρησιμοποιούμενοι μετασχηματισμοί προβλήματος, δηλαδή οι μετασχηματισμοί των δεδομένων πολλαπλών ετικετών σε δεδομένα μονής ετικέτας. Οι μετασχηματισμοί που θα περιγραφούν περαιτέρω είναι ο μετασχηματισμός Binary Relevance, ο μετασχηματισμός Label Powerset και η παραλλαγή Random k-Labelsets [25]. Στη συνέχεια, παρουσιάζονται οι γνωστοί μέθοδοι προσαρμογής αλγορίθμων σε προβλήματα πολλαπλών ετικετών.

2.1.1 Μέθοδος Binary Relevance (BR)

Η πιο γνωστή και συχνά χρησιμοποιούμενη μέθοδος μετασχηματισμού προβλήματος καλείται δυαδική συνάφεια (Binary Relevance - BR). Σε αυτήν κάθε πρόβλημα πολλών ετικετών με L αριθμό ετικετών μετατρέπεται σε L δυαδικά προβλήματα μοναδικής ετικέτας, χρησιμοποιώντας L δυαδικούς κατηγοριοποιητές, έναν για κάθε ετικέτα. Επίσης, το αρχικό σύνολο δεδομένων μετασχηματίζεται σε L σύνολα δεδομένων που περιέχουν όλα τα αρχικά στιγμιότυπα περιλαμβάνοντας, έναν χαρακτηρισμό που δηλώνει αν ανήκουν στην εκάστοτε ετικέτα ή όχι, για όλες τις ετικέτες [26]. Έπειτα, κάθε δυαδικός κατηγοριοποιητής είναι υπεύθυνος για τις προβλέψεις, για το κάθε πρόβλημα ξεχωριστά [25]. Στο παρακάτω σχήμα 2.1 φαίνεται ένα παράδειγμα του τρόπου λειτουργίας της μεθόδου.

Παραδείγματος χάρη, για την κατηγοριοποίηση ενός νέου στιγμιότυπου δίνεται ως αποτέλεσμα



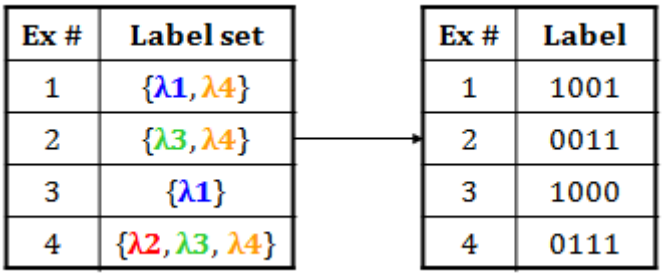
Σχήμα 2.1: Τρόπος Λειτουργίας Binary Relevance-BR

η ένωση όλων των προβλέψεων των δυαδικών κατηγοριοποιητών [27] [28]. Παρά το γεγονός ότι είναι μεθοδολογικά απλή και σχετικά γρήγορη, η μέθοδος BR δεν μοντελοποιεί σχέσεις μεταξύ ετικετών, καθώς κατασκευάζει ένα σύνορο απόφασης ξεχωριστά για κάθε ετικέτα. Επίσης η μέθοδος αυτή μπορεί να επηρεαστεί από το πρόβλημα των μη ισορροπημένων δεδομένων, καθώς κάθε δυαδικός κατηγοριοποιητής είναι πιθανόν να έχει πολύ περισσότερα αρνητικά στιγμιότυπα, απ' ότι θετικά [25].

2.1.2 Μέθοδος Label Powerset (LP) ή Label Combination (LC)

Μια άλλη βασική μέθοδος μετασχηματισμού προβλήματος είναι η χρήση δυναμοσυνόλων ετικετών (Label Powersets - LP) ή αλλιώς συνδυασμός ετικετών (Label Combination - LC). Η ιδιαιτερότητα της μεθόδου είναι ότι θεωρεί το κάθε μοναδικό συνδυασμό ετικετών ως μία ξεχωριστή ετικέτα σε μια νέα διαδικασία κατηγοριοποίησης μοναδικής ετικέτας [29]. Έτσι, η κάθε ετικέτα είναι ουσιαστικά ένα στοιχείο του δυναμοσυνόλου, του συνόλου όλων των ετικετών. Για κάθε νεο, προς κατηγοριοποίηση στιγμιότυπο, ο κατηγοριοποιητής μοναδικής ετικέτας της μεθόδου, δίνει ως αποτέλεσμα την πιο πιθανή ετικέτα (στην ουσία ένα σύνολο ετικετών). Η διαδικασία απεικονίζεται σχηματικά παρακάτω. Έστω ότι για ένα σύνολο εκπαίδευσης ο πίνακας των ετικετών που χαρακτηρίζει το κάθε στιγμιότυπο είναι όπως στο σχήμα 2.2.

Υποθέτοντας ότι η παρακάτω κατανομή πιθανοτήτων (Πίνακας 2.1) παράχθηκε από την μέθοδο LP με εκπαίδευση στα δεδομένα του σχήματος 2.2, σε ένα νέο κείμενο x ανατίθεται η πιο πιθανή κλάση (ως σύνολο ετικετών). Οι πιθανότητες για κάθε απλή ετικέτα υπολογίζονται



Σχήμα 2.2: Τρόπος λειτουργίας της μεθόδου Label Powerset (LP)

από το άθροισμα των πιθανοτήτων των κλάσεων (c) που την περιέχουν (το 1 σε κάθε ετικέτα συμβολίζει την επισήμανση αυτής της ετικέτας στο νέο κείμενο και το 0 το αντίθετο).

Πίνακας 2.1: Υπολογισμός της πιο πιθανής ετικέτας (ως σύνολο ετικετών) με την μέθοδο LP

c	P(c/x)	λ1	λ2	λ3	λ4
λ1,4(1001)	0,7	1	0	0	1
λ3,4(0011)	0,2	0	0	1	1
λ1(1000)	0,1	1	0	0	0
λ2,3,4(0111)	0,0	0	1	1	1
	Σ P(c/x)λj	0,8	0,0	0,2	0,9

Η μέθοδος αυτή λαμβάνει υπόψη σχέσεις και εξαρτήσεις μεταξύ ετικετών επιτυγχάνοντας καλύτερη ακρίβεια, σε αντίθεση με τη μέθοδο BR. Ωστόσο, μπορεί να κατηγοριοποιήσει νέα στιγμιότυπα, μόνο με τους συνδυασμούς ετικετών που έχει συναντήσει ήδη κατά την εκπαίδευση των δεδομένων, με αποτέλεσμα να υπάρχει υπερ-εκπαίδευση [25]. Ακόμη, η μέθοδος LP μπορεί να έχει υψηλή υπολογιστική πολυπλοκότητα και να είναι αρκετά αργή, καθώς απαιτεί τουλάχιστον, τόσες ετικέτες στις κατηγοριοποιήσεις μοναδικής ετικέτας, όσες είναι οι ξεχωριστές ετικέτες των πολλαπλών δεδομένων εκπαίδευσης. Το ανώτατο όριο είναι min(L,2^L), όπου L ο αριθμός των ετικετών. Τέλος, ο μεγάλος αριθμός ετικετών, πολλές από τις οποίες σχετίζονται με πολύ λίγα παραδείγματα στιγμιοτύπων, κάνει τη διαδικασία εκμάθησης ακόμη δυσκολότερη [29].

2.1.3 Μέθοδος Random k-Labelsets (RAkEL)

Η μέθοδος των k τυχαίων συνόλων ετικετών (Random k-Labelsets - RAkEL), κατασκευάζει ένα τυχαίο σύνολο από LP κατηγοριοποιητές, οι οποίοι αναλύθηκαν παραπάνω. Πιο συγκεκριμένα, το αρχικό σύνολο των ετικετών διαιρείται τυχαία σε ένα αριθμό υποσυνόλων (n) ετικετών μικρότερου μεγέθους (k) [29] [28]. Για κάθε ένα από αυτά τα n υποσύνολα ετικετών μεγέθους k, εκπαιδεύεται ένας κατηγοριοποιητής πολλών ετικετών με τη χρήση της LP μεθόδου [27].

Στο Πίνακα 2.2 απεικονίζεται ένα παράδειγμα του τρόπου χρήσης της μεθόδου για $k = 3$ (το μέγεθος των υποσυνόλων) και $n = 7$ τυχαία υποσύνολα. Δεδομένου ενός νέου στιγμιότυπου προς κατηγοριοποίηση, στο τμήμα των προβλέψεων αναπαριστώνται οι προβλέψεις του κατηγο-

Πίνακας 2.2: Υπολογισμός της πιο πιθανής ετικέτας (ως σύνολο ετικετών) με την μέθοδο LP

model	labelsets (k=3)	predictions					
		λ1	λ2	λ3	λ4	λ5	λ6
h1	λ1, λ2, λ6	1	0	-	-	-	-
h2	λ2, λ3, λ4	-	1	1	0	-	-
h3	λ3, λ5, λ6	-	-	0	-	0	1
h4	λ2, λ4, λ5	-	0	-	0	0	-
h5	λ1, λ4, λ5	1	-	-	0	1	-
h6	λ1, λ2, λ6	1	0	1	-	-	-
h7	λ1, λ2, λ6	0	-	-	1	-	0
	Average votes	3/4	1/4	2/3	1/4	1/3	2/3
	Final prediction	1	0	1	0	0	1

ριοποιητή (multi-label με χρήση μεθόδου LP) για κάθε ένα από τα μοντέλα που έχουν παραχθεί. Τέλος αφού υπολογιστεί η πιθανότητα για κάθε ετικέτα (average votes) χρησιμοποιείται κάποιο κατώφλι (στη συγκεκριμένη περίπτωση 0,5) για να καθοριστεί η τελική πρόβλεψη για το νέο στιγμιότυπο.

Όσον αφορά τον ορισμό των παραμέτρων k και n, το k πρέπει να είναι σχετικά μικρό για να μπορεί να εξομαλύνει τα προβλήματα της LP μεθόδου και το n πρέπει να είναι σχετικά μεγάλο για να υπάρχουν πολλά μοντέλα από τα οποία θα καθοριστεί η πιθανότητα.

Η μέθοδος αυτή καταφέρνει να λάβει υπόψη τις συσχετίσεις μεταξύ των ετικετών, ενώ παράλληλα αποφεύγει τα προβλήματα της LP μεθόδου [30]. Ακόμη ένα πλεονέκτημά της είναι ότι έχει μικρότερη υπολογιστική πολυπλοκότητα λόγω των πιο απλών υπο-προβλημάτων της και ότι έχει πιο ισορροπημένα σύνολα δεδομένων εκπαίδευσης. Τέλος σε αντίθεση με την LP μέθοδο μπορεί να προβλέψει υποσύνολα ετικετών που δεν υπάρχουν στα δεδομένα εκπαίδευσης [28].

2.1.4 Μέθοδοι Προσαρμογής Αλγορίθμων (Algorithm Adaptation Methods)

Οι μέθοδοι που ανήκουν στην κατηγορία προσαρμογής αλγορίθμων επεκτείνουν συγκεκριμένους αλγόριθμους έτσι ώστε να μπορούν να χειριστούν δεδομένα πολλών ετικετών κατευθείαν, σε αντίθεση με τις μεθόδους μετασχηματισμού προβλημάτων που χρησιμοποιούν κάποιον γνωστό αλγόριθμο κατηγοριοποίησης χωρίς παραλλαγή [30].

Οι μέθοδοι αυτές συνήθως σχεδιάζονται για ένα συγκεκριμένο πεδίο εφαρμογής και δεν είναι τόσο ευέλικτες, ούτε εφαρμόζονται γενικά όσο οι μέθοδοι μετασχηματισμού προβλημάτων. Ωστόσο, κατά κανόνα περιλαμβάνουν εσωτερικά μετασχηματισμούς προβλημάτων που μπορούν συχνά να εφαρμοστούν σε μία σειρά κατηγοριοποιητών [25].

Ανάμεσα στους αλγόριθμους που έχουν προσαρμοστεί ώστε να χειρίζονται δεδομένα πολλών ετικετών είναι οι: Naïve Bayes, SVM (Rank-SVM), Adaboost (Adaboost.MH), Δέντρα Αποφάσεων (Multi-label C4.5) και Νευρωνικά Δίκτυα (BP-MLL) [31]. Επίσης, πολλές μέθοδοι βασίζονται στον οκνηρό (lazy) αλγόριθμο μάθησης KNN (ML-KNN, BR-KNN, LP-KNN) [32].

Στο [33] οι συγγραφείς προσάρμοσαν τον αλγόριθμο C4.5 για δεδομένα πολλαπλών ετικετών.

Τροποποίησαν τον τύπο υπολογισμού εντροπίας ως εξής:

$$entropy(S) = -\sum_{i=1}^{N}(p(c_i)\cdot\log_2 p(c_i) + q(c_i)\cdot\log_2 q(c_i)) \tag{2.1}$$

όπου $p(c_i)$ είναι η σχετική συχνότητα της ετικέτας c_i και $q(c_i) = 1 - p(c_i)$. Αυτό επιτρέπει πολλαπλές ετικέτες στα φύλλα του δέντρου.

Οι αλγόριθμοι Adaboost MH και Adaboost MR [34] είναι δύο επεκτάσεις του AdaBoost [35] για την κατηγοριοποίηση πολλαπλών ετικετών. Και οι δύο εφαρμόζουν AdaBoost σε αδύναμους κατηγοριοποιητές του τύπου $H : X \times L \to R$. Στον AdaBoost MH εάν το πρόσιμο της εξόδου των αδύναμων κατηγοριοποιητών είναι θετικό για ένα νέο στιγμιότυπο x και μια ετικέτα l τότε θεωρούμε ότι αυτό το στιγμιότυπο μπορεί να επισημανθεί με l, ενώ αν το πρόσιμο είναι αρνητικό τότε αυτό το στιγμιότυπο δεν επισημαίνεται με l. Στον AdaBoost MR, η έξοδος των αδύναμων κατηγοριοποιητών εξετάζεται για την κατάταξη κάθε μίας από τις ετικέτες στο L.

Αν και αυτοί οι δύο αλγόριθμοι προσαρμόζονται σε μια συγκεκριμένη προσέγγιση μάθησης, παρατηρούμε ότι στον πυρήνα τους, χρησιμοποιούν πραγματικά ένα μετασχηματισμένο πρόβλημα: Κάθε στιγμιότυπο (x, Y) διασπάται σε $|L|$ στιγμιότυπα $(x, l, Y[l])$, για όλα τα $l \in L$, όπου $Y[l] = 1$ εάν $l \in Y$ και $[l] = -1$ διαφορετικά. Ο πίνακας 2.4 δείχνει το αποτέλεσμα του μετασχηματισμού του συνόλου δεδομένων του πίνακα 2.3 με τη χρήση αυτής της μεθόδου [23].

Ο ML-KNN [13] είναι μια προσαρμογή του αλγόριθμου KNN για δεδομένα πολλαπλών ετικετών. Στην ουσία, ο ML-KNN χρησιμοποιεί τον αλγόριθμο KNN ανεξάρτητα για κάθε ετικέτα l. Βρίσκει τα Κ εγγύτερα στιγμιότυπα στο σύνολο στιγμιότυπων δοκιμής και τα κατηγοριοποιεί τουλάχιστον με l, ως θετικά και τα υπόλοιπα ως αρνητικά. Αυτό που διαφοροποιεί κυρίως αυτή τη μέθοδο από την εφαρμογή του αρχικού αλγόριθμου KNN στο μετασχηματισμένο πρόβλημα είναι η χρήση προηγούμενων πιθανοτήτων. Ο ML-KNN έχει επίσης τη δυνατότητα να παράγει μια κατάταξη των ετικετών ως έξοδο. Συγκεκριμένα, ο αλγόριθμος ML-KNN βασίζεται σε πληροφορίες στατιστικής, όπως οι προγενέστερες και μεταγενέστερες πιθανότητες των συχνοτήτων εμφάνισης κάθε ετικέτας, που εξάγονται από τα Κ εγγύτερα στιγμιότυπα. Έπειτα, χρησιμοποιεί την αρχή MAP (maximum a posteriori) έτσι ώστε να καθορίσει το σύνολο των ετικετών του νέου στιγμιότυπου προς κατηγοριοποίηση.

Στο [36] οι συγγραφείς παρουσιάζουν δύο συστήματα κατηγοριοποίησης εγγράφων πολλαπλών ετικετών, τα οποία βασίζονται στον κατηγοριοποιητή KNN. Η κύρια συμβολή της εργασίας τους είναι στο στάδιο της προεπεξεργασίας για την αποτελεσματική εκπροσώπηση των εγγράφων. Για την κατηγοριοποίηση ενός νέου στιγμιότυπου, τα συστήματα βρίσκουν αρχικά τα Κ εγγύτερα

Πίνακας 2.3: Παράδειγμα συνόλου πολλαπλών ετικετών

Στιγμιότυπο	sports	Religion	Science	Politics
1	X			X
2			X	X
3	X			
4		X	X	

Πίνακας 2.4: Μετασχηματισμός δεδομένων

Στιγμιότυπο	Ετικέτα (l)	Y[l]
1	Sports	1
1	Religion	-1
1	Science	-1
1	Politics	1
2	Sports	-1
2	Religion	-1
2	Science	1
2	Politics	1
3	Sports	1
3	Religion	-1
3	Science	-1
3	Politics	-1
4	Sports	-1
4	Religion	1
4	Science	1
4	Politics	-1

στιγμιότυπα. Στη συνέχεια, για κάθε εμφάνιση ετικέτας σε κάθε ένα από αυτά τα στιγμιότυπα, αυξάνεται ένας αντίστοιχος μετρητής για την εν λόγω ετικέτα. Τελικά, στην έξοδο έχουμε τις N ετικέτες με τις μεγαλύτερες μετρήσεις. Το N επιλέγεται με βάση τον αριθμό των ετικετών του στιγμιότυπου. Αυτή είναι μια ακατάλληλη στρατηγική για ρεαλιστική χρήση, καθώς ο αριθμός των ετικετών ενός στιγμιότυπου είναι άγνωστος.

Στο [37] ο συγγραφέας ορίζει ένα πιθανολογικό μοντέλο σύμφωνα με το οποίο, κάθε ετικέτα παράγει διαφορετικές λέξεις. Με βάση αυτό το μοντέλο, ένα έγγραφο πολλαπλών ετικετών παράγεται από ένα μείγμα κατανεμημένων λέξεων για κάθε ετικέτα. Οι παράμετροι του μοντέλου υπολογίζονται με βάση την μέγιστη εκ των υστέρων (maximum a posteriori) εκτίμηση από το σύνολο των εγγράφων εκπαίδευσης, χρησιμοποιώντας τη μεγιστοποίηση προσδοκίας (Expectation Maximization) για να υπολογίσουν ποιες ετικέτες, είναι και μίξη βαρών και κατανεμημένων λέξεων για κάθε ετικέτα. Με δεδομένο ένα νέο έγγραφο, το σύνολο ετικετών που είναι πιο πιθανό, επιλέγεται με τον κανόνα Bayes. Αυτή η προσέγγιση για την κατηγοριοποίηση ενός νέου εγγράφου ακολουθεί την λογική, ότι κάθε διαφορετικό σύνολο ετικετών, θεωρείται ανεξάρτητο, ως μια νέα ετικέτα.

Στο [38] οι συγγραφείς παρουσιάζουν έναν αλγόριθμο κατάταξης για την κατηγοριοποίηση πολλαπλών ετικετών. Ο αλγόριθμος τους ακολουθεί τη φιλοσοφία των SVMs (Support Vector Machines): είναι ένα γραμμικό μοντέλο που προσπαθεί να ελαχιστοποιήσει μια συνάρτηση κόστους, διατηρώντας παράλληλα ένα μεγάλο περιθώριο. Η συνάρτηση κόστους που χρησιμοποιούν είναι η απώλεια κατάταξης, η οποία ορίζεται, ως το μέσο κλάσμα των ζευγών ετικετών που κατηγοριοποιούνται εσφαλμένα. Ωστόσο, όπως αναφέρθηκε προηγουμένως, το μειονέκτημα ενός αλγορίθμου κατάταξης είναι ότι δεν παράγει στην έξοδο ένα σύνολο ετικετών.

Στο [39] οι συγγραφείς παρουσιάζουν δύο βελτιώσεις για τον κατηγοριοποιητή (SVM), σε συν-

δυασμό, με τη μέθοδο, δυαδική συνάφεια, για την κατηγοριοποίηση πολλαπλών ετικετών. Η κύρια ιδέα είναι να επεκταθεί το αρχικό σύνολο δεδομένων με $|L|$ επιπλέον χαρακτηριστικά, που περιέχουν τις προβλέψεις κάθε δυαδικού κατηγοριοποιητή. Στη συνέχεια, ένας δεύτερος γύρος εκπαίδευσης των $|L|$ νέων δυαδικών κατηγοριοποιητών λαμβάνει χώρα, αυτή τη φορά χρησιμοποιώντας τα εκτεταμένα σύνολα δεδομένων. Για την κατηγοριοποίηση ενός νέου στιγμιότυπου, χρησιμοποιούνται αρχικά οι δυαδικοί κατηγοριοποιητές του πρώτου γύρου και η έξοδός τους προσαρτάται στα χαρακτηριστικά του στιγμιότυπου, για να σχηματίσουν ένα μετα-στιγμιότυπο. Αυτό το μετα-στιγμιότυπο κατηγοριοποιείται στη συνέχεια από τους δυαδικούς κατηγοριοποιητές του δεύτερου γύρου. Μέσω αυτής της επέκτασης, η προσέγγιση λαμβάνει υπόψη τις πιθανές εξαρτήσεις μεταξύ των διαφόρων ετικετών. Επιπρόσθετα επισημαίνουν, ότι αυτή η βελτίωση είναι στην πραγματικότητα μια εξειδικευμένη περίπτωση εφαρμογής στοιβάδας [40] (μια μέθοδος για το συνδυασμό πολλαπλών κατηγοριοποιητών) με βάση την μέθοδο δυαδικής συνάφειας .

Η δεύτερη βελτίωση [39] είναι ο (SVM-specific) και αφορά το περιθώριο του SVM σε προβλήματα κατηγοριοποίησης πολλαπλών ετικετών. Βελτιώνουν το περιθώριο α) αφαιρώντας πολύ παρόμοια αρνητικά στιγμιότυπα εκπαίδευσης, που βρίσκονται εντός μιας απόστασης κατωφλίου, από το αντληθέν υπερσύνολο (hyperplane), και β) αφαιρώντας τα αρνητικά στιγμιότυπα εκπαίδευσης, μιας πλήρους ετικέτας εάν είναι πολύ παρόμοια με τη θετική ετικέτα, με βάση μια μήτρα σύγχυσης, που εκτιμάται χρησιμοποιώντας οποιοδήποτε γρήγορο και μέτριας ακριβειας κατηγοριοποιητή, σε ένα σύνολο επικύρωσης (validation set). Επισημαίνουν ότι η δεύτερη προσέγγιση, για τη βελτίωση του περιθωρίου, είναι στην πραγματικότητα ανεξάρτητη από τον SVM. Ως εκ τούτου, θα μπορούσε επίσης να χρησιμοποιηθεί ως επέκταση της μεθόδου δυαδικής συνάφειας [23].

Ο αλγόριθμος MMAC [41] είναι ένας αλγόριθμος που ακολουθεί το παράδειγμα της ταξινόμησης συσχέτισης, η οποία ασχολείται με την κατασκευή κανόνων κατηγοριοποίησης, χρησιμοποιώντας κανόνες συσχέτισης (association rules). Ο MMAC μαθαίνει ένα αρχικό σύνολο κανόνων κατηγοριοποίησης, μέσω κανόνων συσχέτισης, καταργεί τα στιγμιότυπα που σχετίζονται με αυτό το σύνολο κανόνων και μαθαίνει αναδρομικά έναν νέο κανόνα που έχει οριστεί από τα υπόλοιπα στιγμιότυπα μέχρι να μην μείνουν άλλα τέτοια στιγμιότυπα. Αυτά τα σύνολα πολλαπλών κανόνων μπορεί να περιέχουν κανόνες με παρόμοιες προϋποθέσεις, αλλά διαφορετικές ετικέτες στη δεξιά πλευρά. Οι κανόνες αυτοί συγχωνεύονται σε έναν μονό κανόνα πολλαπλών ετικετών. Οι ετικέτες κατατάσσονται σύμφωνα με την υποστήριξη των αντίστοιχων επιμέρους κανόνων.

2.2 Χαρακτηριστικά Συνόλων Δεδομένων Πολλαπλών Ετικετών και Μετρικές Αξιολόγησης Αποτελεσμάτων

Στην ενότητα αυτή αναφέρονται κάποιες έννοιες που αφορούν τα δεδομένα πολλαπλών ετικετών και παρουσιάζονται κάποιες μετρικές που έχουν προταθεί για την αξιολόγηση των κατηγοριοποιητών πολλαπλών ετικετών.

2.2.1 Χαρακτηριστικά Μεγέθη Συνόλων Δεδομένων Πολλαπλών Ετικετών

Σε ορισμένα σύνολα δεδομένων, ο αριθμός των ετικετών κάθε στιγμιότυπου είναι μικρός σε σύγκριση με τις ετικέτες $|L|$, ενώ σε άλλες είναι μεγάλος. Αυτό μπορεί να είναι μια παράμετρος που επηρεάζει τις επιδόσεις των διαφόρων κατηγοριοποιητών πολλαπλών ετικετών. Ετσι παρουσιάζονται οι έννοιες της πληθικότητας ετικετών (Label Cardinality-LCARD) και της πυκνότητας (Label Density) για συνόλα πολλαπλών ετικετών [26].

Ας θεωρηθεί λοιπόν, οτι υπάρχει ένα σύνολο πολλαπλών ετικετών με $|D|$ στιγμιότυπα (x_i, Y_i), i = 1...$|D|$.

- Ο μέσος αριθμός ετικετών που σχετίζεται με κάθε στιγμιότυπο i (Label Cardinality-LCARD) δίνεται απο τον τύπο:

$$LC(D) = \frac{1}{|D|}\sum_{i=1}^{|D|}|Y_i| \tag{2.2}$$

- Label Density (πυκνότητα ετικέτας) είναι ο μέσος αριθμός των ετικετών των στιγμιοτύπων $|D|$ διαιρούμενος με τον αριθμό των ετικετών $|L|$ δίνεται απο τον τύπο:

$$LD(D) = \frac{1}{|D|}\sum_{i=1}^{|D|}\frac{|Y_i|}{|L|} \tag{2.3}$$

Η πληθικότητα των ετικετών (LCD) είναι ανεξάρτητη από τον αριθμό των ετικετών $|L|$ στο πρόβλημα κατηγοριοποίησης. Χρησιμοποιείται για την ποσοτικοποίηση του αριθμού των εναλλακτικών ετικετών που χαρακτηρίζουν τα στιγμιότυπα ενός συνόλου δεδομένων εκπαίδευσης πολλαπλών ετικετών. Η πυκνότητα των ετικετών (LD) λαμβάνει υπόψη τον αριθμό των ετικετών στο πρόβλημα κατηγοριοποίησης. Δύο σύνολα δεδομένων με την ίδια πληθικότητα ετικετών, αλλά με μεγάλη διαφορά στον αριθμό των ετικετών (διαφορετική πυκνότητα ετικετών) ενδέχεται να μην εμφανίζουν τις ίδιες ιδιότητες και να προκαλούν διαφορετική συμπεριφορά, στους κατηγοριοποιητές πολλαπλών ετικετών [26]. Οι δύο μετρήσεις σχετίζονται μεταξύ τους με την σχέση:

$$LC(D) = |L|\cdot LD(D) \tag{2.4}$$

.

2.2.2 Μετρικές Αξιολόγησης Αποτελεσμάτων Κατηγοριοποίησης για Προβλήματα Πολλαπλών ετικετών

Η αξιολόγηση των μεθόδων που κατηγοριοποιούν δεδομένα πολλαπλών ετικετών απαιτεί διαφορετικές μετρικές, από εκείνες που χρησιμοποιούνται στην περίπτωση δεδομένων μίας ετικέτας. Η ενότητα αυτή παρουσιάζει τις διάφορες μετρικές που έχουν προταθεί στο παρελθόν για την αξιολόγηση της κατηγοριοποίησης και χωρίζονται στις i) διχοτομήσεις (bipartitions) και τις ii)

κατατάξεις (rankings) σε σχέση με την αλήθεια έξοδου (ground truth) των δεδομένων πολλαπλών ετικετών. Αυτή η διπλωματική εργασία θα ασχοληθεί με την πρώτη κατηγορία μετρικών αξιολόγησης [26].

Επίσης να τονιστεί οτι υπάρχουν δύο σημαντικές μέθοδοι στην κατηγοριοποίηση δεδομένων πολλαπλών ετικετών: η multi-label κατηγοριοποίηση (MLC) και η κατάταξη ετικετών (Label Ranking-LR). Η MLC ασχολείται με το να μάθει ένα μοντέλο, να παράγει ένα διχοτομημένο συνόλο ετικετών με σχετικές και άσχετες ετικέτες σε σχέση με ένα νέο στιγμιότυπο. [26].

Τόσο η MLC όσο και η LR είναι σημαντικές για την εξόρυξη δεδομένων πολλαπλών ετικετών. Για παράδειγμα σε μια εφαρμογή φιλτραρίσματος ειδήσεων, στον χρήστη πρέπει να εμφανίζονται μόνο τα ενδιαφέροντα άρθρα, αλλά είναι επίσης σημαντικό να εμφανίζονται τα πιο ενδιαφέροντα στην κορυφή μιας λίστας. Ιδανικό θα ήταν να εφαρμοστούν μεθόδοι που θα είναι σε θέση να εξορύξουν τόσο μια καταταξη όσο και να δημιουργήσουν μια διχοτόμηση (bipartition) ετικετών από δεδομένα πολλαπλών ετικετών [26].

Για τους ορισμούς αυτών των μετρικών, έστω οτι εξετάζεται το σύνολο δεδομένων πολλαπλών ετικετών (x_i, Y_i),με $i = 1...m$, όπου $Yi \subseteq L$ είναι το σύνολο των ετικετών αληθείας και $L = l_j : j = 1...q$ είναι το σύνολο όλων των ετικετών. Αν x_i είναι ένα νέο στιγμιότυπο, το σύνολο των ετικετών που προβλέπει ένας κατηγοριοποιητής πολλαπλών ετικετών MLC, έστω οτι είναι Z_i [26].

Μετρικές Διχοτόμησης (Bipartitions) Ορισμένες από τις μετρικές που αξιολογούν τις διχοτόμησεις, υπόλογίζονται με βάση το μέσο όρο, των διαφορών, των πραγματικών (actual) και των προβλεπόμενων (predicted) ετικετών σε όλα τα στιγμιότυπα του σύνολο δεδομένων αξιολόγησης (evaluation data set ή testing set). Άλλες μετρικές αποσυνθέτουν τη διαδικασία αξιολόγησης σε ξεχωριστές εκτιμήσεις για κάθε ετικέτα, για τις οποίες (αξιολογήσεις) στη συνέχεια υπολογίζουν τον μέσο όρο για όλες τις ετικέτες. Με βάση λοιπόν το παραπάνω παράδειγμα ορίζονται:

- Example-based

  Η μετρική απώλειας **Hamming Loss** ορίζεται ως εξής:

  $$Hamming - Loss = \frac{1}{m}\sum_{i=1}^{m}\frac{|Y_i \Delta Z_i|}{M} \tag{2.5}$$

  ,όπου Δ είναι η συμμετρική διαφορά των δυο συνόλων, η οποία είναι ισοδύναμη με την δυαδική (XOR) λογική σχέση.

  Στην κατηγοριοποίηση πολλαπλών ετικετών η εσφαλμένη κατηγοριοποίηση δεν θεωρείται σωστή ή λάθος κατηγοριοποίηση. Μια πρόβλεψη που περιέχει ένα υποσύνολο των πραγματικών κλάσεων (ετικετών) θα πρέπει να θεωρείται καλύτερη από μια πρόβλεψη που δεν περιέχει κανένα από αυτά. Έτσι η ακρίβεια δεν μετράει τα σωστά κατηγοριοποιημένα στιγμιότυπα. Η μετρική Hamming Loss υπολογίζει την απώλεια που παράγεται στη συμβολοσειρά bits των ετικετών κατά τη διάρκεια, της πρόβλεψης και το κάνει αυτό

εκτελώντας μια πράξη XOR μεταξύ των πραγματικών και των προβλεπόμενων ετικετών και στη συνέχεια βρίσκει το μέσο όρο σε όλο το σύνολο δεδομένων.

Παρακάτω δίνονται ορισμένα παραδείγματα για αυτή την μετρική, μιας και είναι το βασικό κριτήριο αξιολόγησης των συνόλων δεδομένων που χρησιμοποιήθηκε σε αυτή τη διπλωματική εργασία. Έστω οτι ορίζεται ένα σύνολο δεδομένων το οποίο έχει δυο (2) στιγμιότυπα και δύο (2) ετικέτες σε κάθε στιγμιότυπο τότε αν:

- **Περίπτωση 1η**: Πραγματικά στιγμιότυπα ίδια με αυτά που προβλέφτηκαν.
  Έστω οτι τα πραγματικά είναι: $Actual = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$ και αυτά που προβλέφθηκαν είναι: $predicted = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$ τοτε προκύπτει: $Actual \oplus Predicted = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$
  Άρα το Hamming Loss είναι: **HL=0.0**
- **Περίπτωση 2η**: Πραγματικά στιγμιότυπα τελείως διαφορετικά με αυτά που προβλέφτηκαν.
  Έστω οτι τα πραγματικά είναι: $Actual = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$ και αυτά που προβλέφθηκαν είναι: $predicted = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$ τοτε προκύπτει: $Actual \oplus Predicted = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$
  Άρα το Hamming Loss είναι: **HL=4/(2*2)=1**
- **Περίπτωση 3η**: Πραγματικά στιγμιότυπα μερικώς διαφορετικά με αυτά που προβλέφτηκαν.
  Έστω οτι τα πραγματικά είναι: $Actual = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$ και αυτά που προβλέφθηκαν είναι: $predicted = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$ τοτε προκύπτει: $Actual \oplus Predicted = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
  Άρα το Hamming Loss είναι: **HL=(1 + 1)/(2*2)=0.5**

Συμπερασματικά παρατηρείται ότι η τιμή της μετρικής κυμαίνεται μεταξύ των τιμών 0 και 1. Όσο μικρότερη είναι η τιμή της μετρικής, τόσο **καλύτερος** είναι ο κατηγοριοποιητής.

Η ακρίβεια κατηγοριοποίησης (Classification accuracy) ή ακρίβεια υποσυνόλου (subset accurancy) ορίζεται ως εξής:

$$ClassificationAccurancy = \frac{1}{m}\sum_{i=1}^{m} I(Z_i = Y_i) \tag{2.6}$$

όπου $I(true) = 1$ and $I(false) = 0$.

Πρόκειται για ένα πολύ αυστηρό μέτρο αξιολόγησης καθώς απαιτεί το σύνολο πρόβλεψης ετικετών (predicted set of labels) να είναι ακριβώς ίδιο με το σύνολο αληθείας ετικετών (true set of labels).

Άλλα μέτρα είναι τα εξής:

$$Precision = \frac{1}{m}\sum_{i=1}^{m}\frac{(Y_i \cap Z_i)}{Z_i} \tag{2.7}$$

$$Recall = \frac{1}{m}\sum_{i=1}^{m}\frac{(Y_i \cap Z_i)}{Y_i} \tag{2.8}$$

$$F1 = \frac{1}{m}\sum_{i=1}^{m}\frac{2|(Y_i \cap Z_i)|}{|Z_i| + |Y_i|} \tag{2.9}$$

$$Accuracy = \frac{1}{m}\sum_{i=1}^{m}\frac{|Y_i \cap Z_i|}{|Y_i \cup Z_i|} \tag{2.10}$$

- Label-based

  Οποιοδήποτε γνωστό μέτρο για δυαδική αξιολόγηση μπορεί να χρησιμοποιηθεί, όπως η ακρίβεια (accurancy), η καμπύλη ROC, η ορθότητα (precision) και η ευαισθησία (recall). Ο υπολογισμός αυτών των μέτρων για όλες τις ετικέτες μπορεί να επιτευχθεί χρησιμοποιώντας δύο λειτουργίες με χρήση του μέσου όρου, που ονομάζονται macro-averaging και micro-averaging [42]. Οι λειτουργίες αυτές λαμβάνονται συνήθως υπόψη για ακρίβεια και ευαισθησία κατά μέσο όρο, όπως και ο αρμονικός μέσος όρος τους (F-measure) στην επιστήμη ανάκτησης πληροφορίας (Information Retrieval) [43].

  Αν θεωρηθεί οτι μια αξιολόγηση σε ένα δυαδικό πρόβλημα οτι είναι το $B(tp, tn, fp, fn)$ που βασίζεται στον υπολογισμό του αριθμού των θετικά αληθή (true positives-tp), αρνητικά αληθή (true negatives-tn), θετικά ψευδή (false positives-fp) και αρνητικά ψευδή (false negatives-fn). Αν tp_λ,tn_λ,fp_λ,fn_λ είναι αντίστοιχα ο αριθμός των θετικά αληθή (true positives-tp), αρνητικά αληθή (true negatives-tn), θετικά ψευδή (false positives-fp) και αρνητικά ψευδή (false negatives-fn) για μια ετικέτα (λ). Τότε οι μετρικές macro-averaged και micro-averaged για το (B), υπολογίζονται ως εξής:

$$B_{macro} = \frac{1}{q}\sum_{\lambda=1}^{q} B(tp_\lambda, fp_\lambda, tn_\lambda, fn_\lambda) \tag{2.11}$$

$$B_{micro} = B(\sum_{\lambda=1}^{q} tp_\lambda, \sum_{\lambda=1}^{q} fp_\lambda, \sum_{\lambda=1}^{q} tn_\lambda, \sum_{\lambda=1}^{q} fn_\lambda) \tag{2.12}$$

  Πρέπει να τονιστεί ότι ο μικρο-μέσος όρος (micro-averaging) έχει το ίδιο αποτέλεσμα με τον μάκρο-μέσο όρο (macro-averaging) για ορισμένα μέτρα, όπως η ακρίβεια (accuracy), ενώ διαφέρει για άλλα μέτρα, όπως η ορθότητα (precision), η ευαισθησία (recall) και η καμπύλη ROC. Σημειώστε επίσης ότι η μέση (μακρο(macro), μικρο(micro)) ακρίβεια (accuracy) και η απώλεια Hamming Loss παίρνουν τιμές στο διάστημα [0,1], καθώς η απώλεια Hamming Loss είναι στην πραγματικότητα το μέσο δυαδικό σφάλμα κατηγοριοποίησης [26].

Ιεραρχικές (Hierarchical) Μετρικές Η ιεραρχική απώλεια είναι μια τροποποιημένη έκδοση της απώλειας Hamming Loss που λαμβάνει υπ' όψιν μια υπάρχουσα ιεραρχική δομή των ετικε-

τών. Εξετάζει τις προβλεπόμενες ετικέτες (predicted labels) από πάνω προς τα κάτω σύμφωνα με την ιεραρχία και κάθε φορά που η πρόβλεψη για μια ετικέτα είναι λάθος, το δευτερεύον δέντρο που έχει τις ρίζες του σε αυτόν τον κόμβο, δεν υπολογιζεται στην απώλεια. Έστω ότι $anc(\lambda)$ να είναι το σύνολο όλων των κόμβων προγόνων του [26]. Η ιεραρχική απώλεια καθορίζεται απο τον παρακάτω τύπο:

$$H - Loss = \frac{1}{m}\sum_{i=1}^{m}|\{\lambda : \lambda \in Y_i \Delta Z_i, anc(\lambda) \cap (Y_i \Delta Z_i) = \varnothing\}| \tag{2.13}$$

2.3 Τεχνικές μείωσης Δεδομένων (Data Reduction Techniques -DRT)

Πολλές από τις πρόσφατες ερευνητικές προσπάθειες επικεντρώνονται στη μείωση του υπολογιστικού κόστους του κατηγοριοποιητή KNN. Έχουν προταθεί πολυάριθμοι αλγόριθμοι και τεχνικές που μπορούν να εξομαλύνουν τα αδύνατα σημεία του κατηγοριοποιητή. Μια πιθανή κατηγοριοποίηση των μεθόδων για τη βελτίωση του κατηγοριοποιητή KNN είναι: (i) μέθοδοι Δείκτοδότησης πολλαπλών χαρακτηριστικών (Multi-attribute Indexes) και (ii) Τεχνικές μείωσης δεδομένων (Data Reduction Techniques)

Οι μέθοδοι Δεικτοδότησης πολλαπλών χαρακτηριστικών (Multiattribute indexing) μπορούν να επιταχύνουν τις εγγύτερες αναζητήσεις γειτόνων και, κατά συνέπεια, μπορούν να επιταχύνουν τον κατηγοριοποιητή KNN. Ωστόσο, οι απαιτήσεις αποθήκευσης αυξάνονται, δεδομένου ότι, εκτός από τα δεδομένα εκπαίδευσης, πρέπει να αποθηκευτεί και ο δείκτης. Επιπλέον, οι δείκτες μπορούν να εφαρμοστούν μόνο σε σύνολα δεδομένων με μέτρια διάσταση (π.χ. 2-10). Σε υψηλότερες διαστάσεις, λόγω του φαινόμενου της **κατάρας των διαστάσεων** (*dimentionality curse*), η τεχνική της σειριακής αναζήτησης είναι πιο αποτελεσματική, από ότι, η τεχνική της χρήσης δεικτών [12].

Συνεπώς, είναι προτιμότερο να εφαρμοστεί πρώτα μια τεχνική μείωσης των διαστάσεων (Dimentionality Reduction Techique), όπως είναι η ανάλυση κύριων συνιστωσών (Principal Component Analysis-PCA) [44]. Δυστυχώς, η PCA μπορεί να οδηγήσει σε σημαντική απώλεια πληροφοριών. Επιπλέον, κάθε στιγμιότυπο προς κατηγοριοποίηση θα πρέπει να μετασχηματίζεται, πριν την εφαρμογή, της κατηγοριοποίησης και της εύρεσης των εγγύτερων γειτόνων του. Επομένως, η κατηγοριοποίηση μπορεί να γίνει λιγότερο αποτελεσματική.

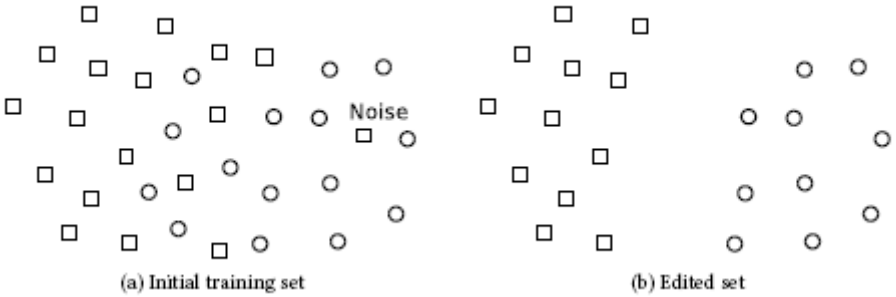
Οι τεχνικές μείωσης δεδομένων (DRT) χωρίζονται σε δυο κατηγορίες: (i) τη μείωση των στιγμιοτύπων του αρχικού συνόλου δεδομένων (item reduction) και, (ii) την μείωση των διαστάσεων των στιγμιοτύπων (dimensionality reduction). Αυτή η διπλωματική εργασία επικεντρώνεται στην πρώτη κατηγορία.

Οι τεχνικές μείωσης δεδομένων μπορούν να αντιμετωπίσουν αποτελεσματικά όλες τις αδυναμίες και μπορούν να ομαδοποιηθούν σε δύο κύριες κατηγορίες αλγορίθμων: i) τους αλγόριθμους επιλογής προτύπων (Prototype Selection-PS) και (ii) τους αλγόριθμους σύνοψης προτύπων (Prototype Abstraction) τους οποίους συναντάμε στη βιβλιογραφία και ως αλγορίθμους παραγωγής

προτύπων (Prototype Generation-PG).

Οι αλγόριθμοι επιλογής προτύπων (PS) επιλέγουν αντιπροσωπευτικά στιγμιότυπα (ή πρότυπα) από το αρχικό σύνολο εκπαίδευσης ενώ οι αλγόριθμοι παραγωγής προτύπων παράγουν νέα στιγμιότυπα συνοψίζοντας τα παρόμοια στιγμιότυπα εκπαίδευσης και τα χρησιμοποιούν ως πρότυπα. Στην πραγματικότητα, κάθε πρότυπο-στιγμιότυπο αντιπροσωπεύει μια συγκεκριμένη περιοχή στιγμιότυπων του πολυδιάστατου χώρου του σύνολου δεδομένων [12].

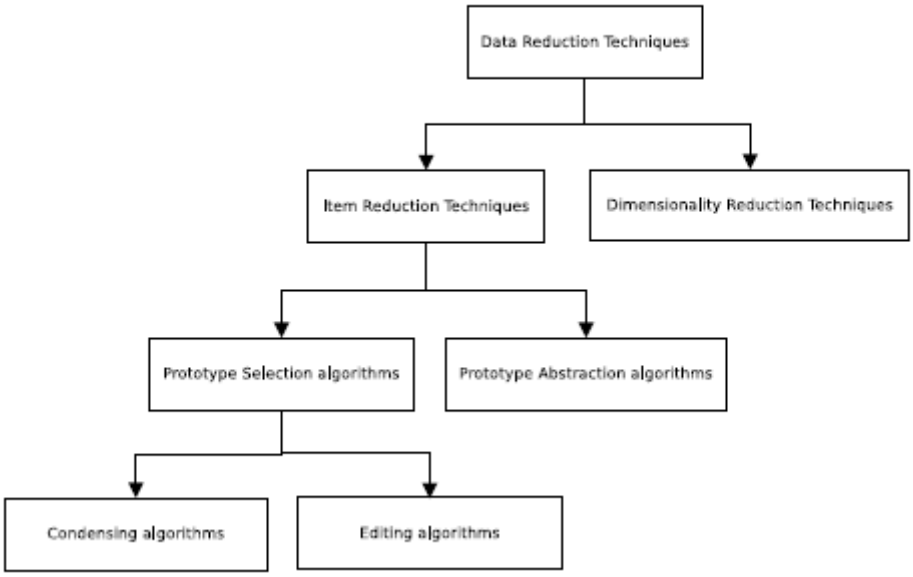
Οι αλγόριθμοι επιλογής προτύπων(PS) χωρίζονται σε δύο υποκατηγορίες. Μπορούν να είναι είτε αλγόριθμοι συμπύκνωσης (condensing) ή επεξεργασίας(editing). Οι αλγόριθμοι παραγωγής και συμπύκνωσης (condensing) προτύπων έχουν το ίδιο κίνητρο. Στόχος τους είναι να δημιουργήσουν ένα μικρό αντιπροσωπευτικό σύνολο των αρχικών δεδομένων εκπαίδευσης. Αυτό συνήθως ονομάζεται σύνολο συμπύκνωσης (condensing set). Η χρήση ενός συνόλου συμπύκνωσης έχει τα οφέλη του χαμηλού κόστους, υπολογιστικών απαιτήσεων και αποθήκευσης, ενώ η ακρίβεια κατηγοριοποίησης δεν επηρεάζεται. Από την άλλη, οι αλγόριθμοι επεξεργασίας αποσκοπούν στη βελτίωση της ακρίβειας και όχι στην επίτευξη υψηλού ποσοστού μείωσης. Για να επιτευχθεί αυτό, προσπαθούν να βελτιώσουν την ποιότητα του συνόλου εκπαίδευσης, αφαιρώντας το θόρυβο, τις ακραίες τιμές και τα εσφαλμένα στιγμιότυπα, εξομαλύνοντας τα όρια απόφασης μεταξύ των ετικετών (βλέπε σχήμα 2.3).



Σχήμα 2.3: Εξομάλυνση ορίων αποφάσεων και αφαίρεση θορύβου

Στην ιδανική περίπτωση, ένας αλγόριθμος επεξεργασίας δημιουργεί ένα επεξεργασμένο σύνολο εκπαίδευσης, χωρίς επικαλύψεις μεταξύ των ετικετών. Στο σχήμα 2.4 συνοψίζουμε τις προαναφερθείσες κατηγορίες σε μια ιεραρχία. Αξίζει να σημειωθεί ότι ορισμένοι αλγόριθμοι συμπύκνωσης ενσωματώνουν την ιδέα της επεξεργασίας και ονομάζονται υβριδικοί αλγόριθμοι.

Οι τεχνικές μείωσης δεδομένων μπορούν να αξιολογηθούν με βάση τρία κριτήρια. Το πρώτο είναι το ποσοστό μείωσης, που δείχνει πόσο μικρότερο είναι το μέγεθος του συνόλου συμπύκνωσης, σε σχέση με το μέγεθος του αρχικού συνόλου εκπαίδευσης. Πρακτικά, είναι ο λόγος του αριθμού των στιγμιοτύπων που αφαιρέθηκαν, σε σχέση με τον αριθμό των αρχικών στιγμιοτύπων του συνόλου εκπαίδευσης. Προφανώς, όσο υψηλότερο είναι το ποσοστό μείωσης, τόσο πιο γρήγορη γίνεται η κατηγοριοποίηση με τον αλγόριθμο KNN. Ένα άλλο σημαντικό κριτήριο είναι η ακρίβεια κατηγοριοποίησης που επιτυγχάνεται από τον κατηγοριοποιητή KNN όταν εκτελείται πάνω στο σετ συμπύκνωσης. Το τρίτο κριτήριο είναι το υπολογιστικό κόστος στο στάδιο της προεπεξεργασίας, που είναι στην ουσία, το κόστος που απαιτείται για την κατασκευή

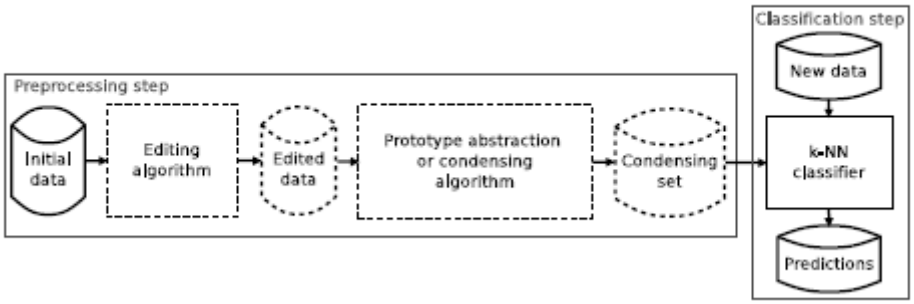


Σχήμα 2.4: Κατηγοριοποίηση τεχνικών μείωσης δεδομένων. Μια ιεραρχική ταξινόμηση.

του συνόλου συμπύκνωσης.

Παρόλο που η επεξεργασία (editing) έχει εντελώς διαφορετικό στόχο από τις άλλες τεχνικές μείωσης δεδομένων, μπορεί να χρησιμοποιηθεί για να βελτιώσει την απόδοσή, αυξάνοντας τα ποσοστά μείωσης ή και τα επίπεδα ακρίβειας. Πιο συγκεκριμένα, τα ποσοστά μείωσης πολλών αλγορίθμων παραγωγής προτύπων και συμπύκνωσης εξαρτώνται απο το επίπεδο θορύβου στα στιγμιότυπα εκπαίδευσης. Τα υψηλά επίπεδα θορύβου στο σύνολο εκπαίδευσης αποτρέπουν πολλούς αλγόριθμους συμπύκνωσης ή αφαίρεσης προτύπων από την επίτευξη υψηλών ποσοστών μείωσης. Αυτό έχει σαν αποτέλεσμα, όσο υψηλότερο είναι το επίπεδο θορύβου, τόσο χαμηλότερα είναι τα ποσοστά μείωσης που επιτύγχάνονται. Ως εκ τούτου, η αποτελεσματική εφαρμογή τέτοιων αλγορίθμων, συνεπάγεται την αφαίρεση του θορύβου από τα δεδομένα, πριν την εφαρμογή, της διαδικασίας κατηγοριοποίησης. Ως εκ τούτου, ένας αλγόριθμος επεξεργασίας θα πρέπει να χρησιμοποιείται σε δεδομένα εκπαίδευσης με θόρυβο, ώστε είτε να βελτιωθεί η ακρίβεια, είτε να γίνει η διαδικασία συμπύκνωσης και παραγωγής προτύπων πιο αποτελεσματική [12].

Στο σχήμα 2.5 συνοψίζεται η διαδικασία κατηγοριοποίησης KNN μέσω της μείωσης των δεδομένων. Η όλη διαδικασία περιλαμβάνει δύο φάσεις, προεπεξεργασία και κατηγοριοποίηση. Βεβαια, η φάση προεπεξεργασίας είναι προαιρετική. Γενικά, υπάρχουν τέσσερις πιθανοί τύποι προεπεξεργασίας: (i) μη προεπεξεργασία, (ii) μόνο επεξεργασία, (iii) μόνο συμπύκνωση, και (iv) επεξεργασία και συμπύκνωση. Εάν το σύνολο εκπαίδευσης δεν περιέχει στιγμιότυπα θορύβου και παραπλανητικά στιγμιότυπα και το μέγεθός του είναι μικρό, τότε δεν απαιτείται προεπεξεργασία. Όταν το μέγεθος του συνόλου εκπαίδευσης είναι μικρό, αλλά περιέχει θόρυβο, μόνο ένας αλγόριθμος επεξεργασίας θα πρέπει να εκτελείται κατά την προεπεξεργασία. Από την άλλη, σε περιπτώσεις μεγάλων και χωρίς θόρυβο εκπαιδευτικών συνόλων, θα πρέπει να εκτελείται μείωση δεδομένων χωρίς επεξεργασία (editing) [12]. Τέλος, σε περιπτώσεις μεγάλων



Σχήμα 2.5: Κατηγοριοποίηση KNN μέ χρήση τεχνικών Μείωσης Δεδομένων

δεδομένων εκπαίδευσης με θόρυβο, και τα δύο είδη αλγορίθμων προεπεξεργασίας πρέπει να εφαρμοστούν. Ο στόχος μιας πλήρους διαδικασίας προεπεξεργασίας μείωσης δεδομένων, είναι η δημιουργία ενός συνόλου συμπύκνωσης, απαλλαγμένο από θόρυβο, διατηρώντας ή δημιουργώντας για κάθε ετικέτα επαρκή αριθμό προτύπων που είναι απαραίτητα για την κατηγοριοποίηση των K εγγύτερων γειτόνων.

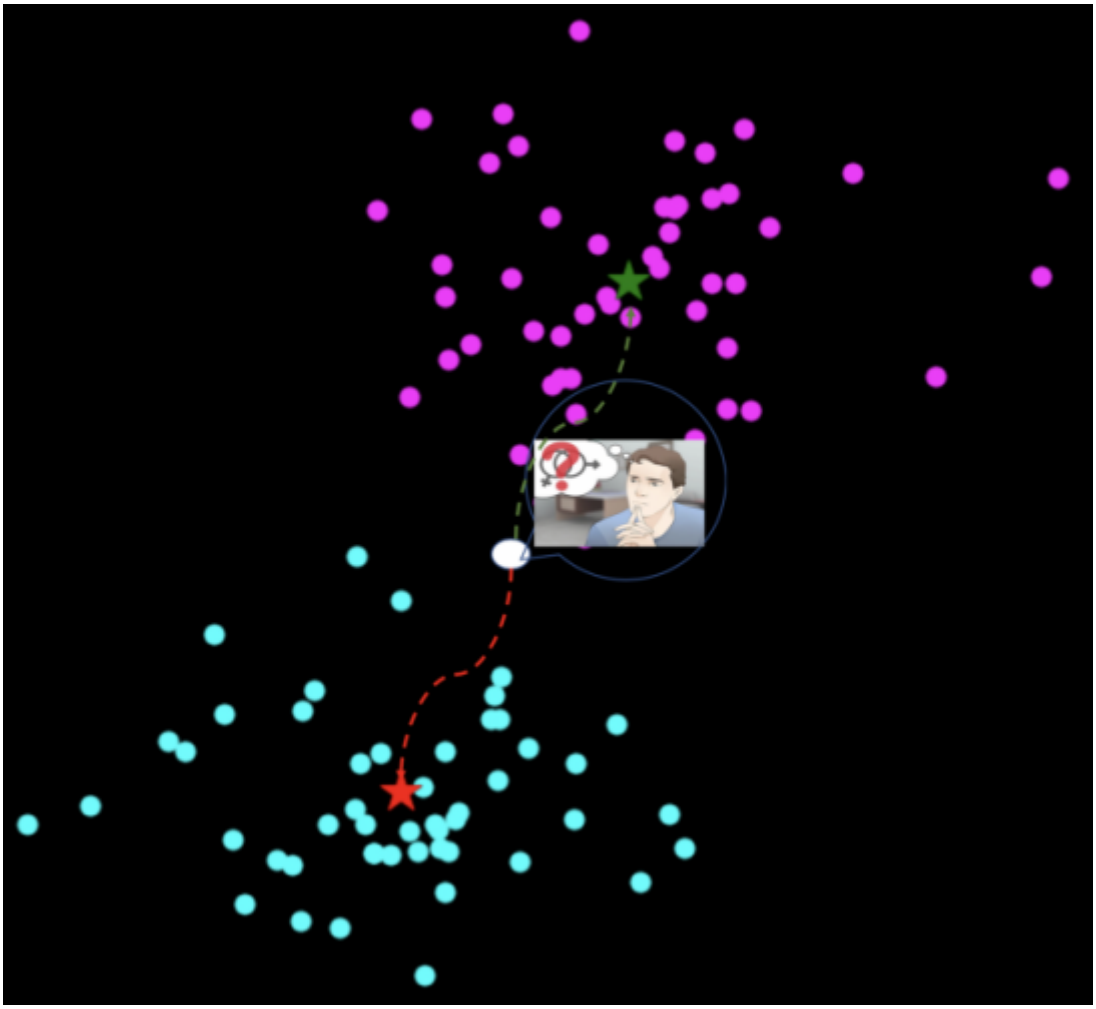
2.4 Συσταδοποίηση - Αλγόριθμος K μέσων (K-means)

2.4.1 Συσταδοποίηση

Η συσταδοποίηση είναι μια από τις πιο κοινές τεχνικές ανάλυσης-διερεύνησης των δεδομένων ενός συνόλου, που χρησιμοποιείται για να αποκτήσει μια διαίσθηση σχετικά με τη δομή των στιγμιοτύπων σε ένα σύνολο δεδομένων. Μπορεί να οριστεί ως η εργασία προσδιορισμού συστάδων στα δεδομένα, έτσι ώστε τα στιγμιότυπα στην ίδια υποομάδα (συστάδα) να είναι πολύ παρόμοια, ενώ τα στιγμιότυπα ανάμεσα στις συστάδες είναι πολύ διαφορετικά μεταξύ τους. Με άλλα λόγια, προσπαθεί να ανιχνεύσει ομοιογενείς συστάδες εντός των δεδομένων, έτσι ώστε τα στιγμιότυπα σε κάθε συστάδα να είναι όσο το δυνατόν πιο παρόμοια σύμφωνα με ένα μέτρο ομοιότητας, όπως η Ευκλείδεια απόσταση [45]. Στο σχήμα 2.6 παρουσιάζονται τα στιγμιότυπα ενός συνόλου δεδομένων πριν την εκτέλεση της μεθόδου συσταδοποίησης K μέσων. Η ανάλυση συσταδοποίησης μπορεί να γίνει με βάση τα χαρακτηριστικά των στιγμιοτύπων, όπου γίνεται προσπάθεια να ανιχνευθούν συστάδες στιγμιοτύπων με βάση τα χαρακτηριστικά τους ή με βάση τα στιγμιότυπα όπου γίνεται προσπάθεια να ανιχνευθούν συστάδες χαρακτηριστικών που βασίζονται σε στιγμιότυπα. Αυτή η διπλωματική εργασία ασχολείται με την συσταδοποίηση με βάση τα χαρακτηριστικά των στιγμιοτύπων του συνόλου δεδομένων.

Παράδειγμα συσταδοποίησης είναι η κατάτμηση της αγοράς, όπου γίνεται προσπάθεια να εντοπιστούν πελάτες που είναι παρόμοιοι μεταξύ τους, είτε από την άποψη των συμπεριφορών ή των χαρακτηριστικών [45].

Σε αντίθεση με την κατηγοριοποίηση, η συσταδοποίηση θεωρείται μια μη εποπτεύομενη μέθοδος μάθησης, καθώς δεν δίνεται η αλήθεια (ground truth), για να συγκριθεί με την έξοδο του



Σχήμα 2.6: Στιγμιοτύπα πριν εφαρμογή αλγορίθμου Κ μέσων

αλγορίθμου συσταδοποίησης με τις πραγματικές ετικέτες και ως εκ τούτου να αξιολογηθεί η απόδοση. Με τη συσταδοποίηση, απλά γίνεται προσπάθεια να διερευνηθεί η δομή των στιγμιοτύπων, ομαδοποιώντας αυτά, σε ξεχωριστές συστάδες (υποομάδες). Όσο λοιπόν αφορά την συσταδοποίηση, θα γίνει περιγραφεί του αλγόριθμου Κ μέσων που θεωρείται ως ένας από τους πιο χρησιμοποιούμενους αλγόριθμους συσταδοποίησης λόγω της απλότητάς του [45]. Ο αλγόριθμος αυτός αποτελεί τη βάση της μεθόδου μείωσης δεδομένων που αναπτύχθηκε στα πλαίσια της παρούσας διπλωματικής εργασίας και για αυτό τον λόγο παρουσιάζεται αναλυτικά σε αυτό το κεφάλαιο.

2.4.2 Αλγόριθμος Κ μέσων (K-means)

Ο αλγόριθμος Κ μέσων είναι ένας επαναληπτικός αλγόριθμος που προσπαθεί να διαιρέσει το σύνολο δεδομένων σε διακριτές μη επικαλυπτόμενες συστάδες (υποομάδες) που ορίζονται από ένα προκαθορισμένο αριθμό Κ (όπου Κ ο αριθμός των συστάδων), έτσι ώστε κάθε στιγμιότυπο να ανήκει σε μία μόνο συστάδα. Προσπαθεί να ομαδοποιήσει τα στιγμιότυπα μίας συστάδας έτσι ώστε τα παρόμοια στιγμιότυπα να τοποθετηθούν στην ίδια συστάδα ενώ τα διαφορετικά

σε διαφορετική συστάδα. Εκχωρεί στιγμιότυπα σε μια συστάδα έτσι ώστε το άθροισμα της τετραγωνικής απόστασης μεταξύ των στιγμιοτύπων και του κέντρου της συστάδας (αριθμητικός μέσος όρος όλων των στιγμιοτύπων που ανήκουν σε αυτή την συστάδα) να είναι το ελάχιστο. Όσο λιγότερες διακυμάνσεις υφίστανται μέσα στις συστάδες, τόσο πιο ομοιογενή (παρόμοια) είναι τα στιγμιότυπα εντός της ίδιας συστάδας [45].

Ο τρόπος λειτουργίας του αλγορίθμου K μέσων έχει ως εξής:

- Καθορίζεται ο αριθμός των συστάδων (αριθμός K).
- Προετοιμάζονται τα κέντρα των συστάδων με το ανακάτεμα πρώτα των στιγμιότυπων του συνόλου δεδομένων και στη συνέχεια την τυχαία επιλογή των στιγμιοτύπων K για κέντρα χωρίς αντικατάσταση.
- Επανάλαμβάνεται η διαδικασία εύρεσης των κέντρων των συστάδων μέχρι να μην υπάρχει καμία αλλαγή στα κέντρα αυτών, δηλαδή η εκχώρηση στιγμιότυπων σε συστάδες δεν αλλάζει.
- Υπολογίζεται το άθροισμα της τετραγωνικής απόστασης μεταξύ των στιγμιοτύπων και όλων των κέντρων.
- Αντιστοιχίζεται κάθε στιγμιότυπο στην πλησιέστερη συστάδα (κεντρο).
- Υπολογίζονται τα κέντρα για τις συστάδες, λαμβάνοντας το μέσο όρο, όλων των στιγμιότυπων που ανήκουν σε κάθε συστάδα.

Συνοπτικά παρακάτω φαίνονται τα παραπάνω βήματα στον αλγόριθμό 2. Η προσέγγιση που ο K

Algorithm 2 k-means

Input: K:the number of clusters, D:a data set containing n objects
Output:A set of k clusters

1: arbitrarily choose k objects from D as the initial cluster centers;
2: **repeat**
3: (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;
4: update the cluster means, that is, calculate the mean value of the objects for each cluster;
5: **until** no change;

μέσων ακολουθεί για να λύσει το πρόβλημα καλείται προσδοκία-μεγιστοποίηση (Expectation-Maximization). Το βήμα Ε αντιστοιχίζει τα στιγμιότυπα του συνόλου δεδομένων στην πλησιέστερη συστάδα. Το βήμα Μ υπολογίζει το κεντρο κάθε συστάδας [45]. Η αντικειμενική συνάρτηση είναι:

$$J = \sum_{i=1}^{m} \sum_{k=1}^{K} w_{ik} ||x^i - \mu_k||^2 \tag{2.14}$$

,όπου $w_i k$ = 1 για το στιγμιότυπο x_i αν ανήκει στην συστάδα k, αλλιώς το $w_i k$ = 0. Επίσης το μ_k είναι το κεντρο της συστάδας του x_i .

Είναι ένα πρόβλημα ελαχιστοποίησης δύο μερών. Πρώτα ελαχιστοποιείται το J, w_{ik} και θεωρείται το μ_k σταθερό. Στη συνέχεια, ελαχιστοποιείται J, μ_k και θεωρείται το w_{ik} σταθερό. Αν εξεταστεί πιο τεχνικά, διαφοροποιούνται τα J, w_{ik} πρώτα και ενημερώνει τις αναθέσεις της συστάδας (Ε-βήμα). Στη συνέχεια διαφοροποιούνται τα J, μ_k και επανυπολογίζονται τα κεντρα, μετά τις αναθέσεις στην συστάδα από το προηγούμενο βήμα (Μ-βήμα).

Ως εκ τούτου, το **Ε-βήμα** φαίνεται στην σχέση 2.15 και είναι:

$$\frac{\vartheta J}{\vartheta w_i k} = \sum_{i=1}^{m}\sum_{k=1}^{k} w_{ik}||x^i - \mu_k||^2 \Rightarrow w_{ik} = \begin{cases} 1 & \text{if } k = argmin_j||x^i - \mu_j||^2 \\ 0 & otherwise \end{cases} \tag{2.15}$$

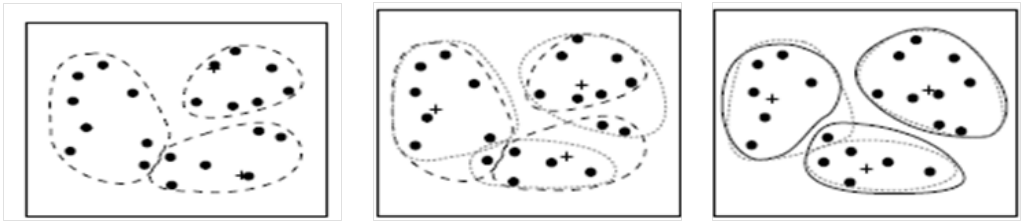
Με άλλα λόγια, αντιστοιχίζεται το στιγμιότυπο x^i στην πλησιέστερη συστάδα που αποφασίζεται από το άθροισμα της τετραγωνικής απόστασης από το κέντρο της συστάδας.
Και το **Μ-βήμα** φαίνεται στην σχέση 2.16 και είναι:

$$\frac{\vartheta J}{\vartheta_k} = 2\sum_{i=1}^{m} w_{ik}(x^i - \mu_k) = 0$$

$$\Rightarrow \mu_k = \frac{\sum_{i=1}^{m} w_{ik}x^i}{\sum_{i=1}^{m} w_{ik}} \tag{2.16}$$

δηλαδή επανυπολογισμός των κέντρων κάθε συστάδας, ώστε να αντικατοπτρίζονται οι νέες αναθέσεις των κέντρων.

Δεδομένου ότι οι αλγόριθμοι συστάδοποίησης, συμπεριλαμβανομένου του Κ μέσων, χρησιμοποιούν μετρικές απόστασης για να προσδιορίσουν την ομοιότητα μεταξύ των στιγμιοτύπων, προτείνεται η κανονικοποιήση των δεδομένων, καθώς σχεδόν πάντα τα χαρακτηριστικά σε οποιοδήποτε σύνολο δεδομένων θα έχουν διαφορετικές μονάδες μέτρησης, όπως η ηλικία, έναντι του εισοδήματος.



(a) Αρχική συσταδοποίηση (b) Επανάληψη (c) Τελική συσταδοποίηση

Σχήμα 2.7: Συσταδοποίηση με χρήση του Αλγορίθμου Κ μέσων

Δεδομένης της επαναληπτικής φύσης του αλγορίθμου Κ μέσων και του τυχαίου καθορισμού των κέντρων στην αρχή του αλγορίθμου, οι διαφορετικές αρχικοποιήσεις μπορεί να οδηγήσουν σε διαφορετικές συσταδοποιήσεις, καθώς ο αλγόριθμος Κ μέσων μπορεί να σταματήσει σε ένα τοπικό βέλτιστο και μπορεί να μην συγκλίνει στο ολικό βέλτιστο. Επομένως, συνιστάται να

υλοποιείται ο αλγόριθμος χρησιμοποιώντας διαφορετικές αρχικοποιήσεις των κέντρων και να επιλέγονται τα αποτελέσματα της υλοποίησης, που απέδωσαν το χαμηλότερο άθροισμα τετραγωνικής απόστασης [45].

Μια οπτική αναπαράσταση της λειτουργίας του Αλγορίθμου φαίνεται στο σχήμα 2.7, όπου στο σχήμα 2.7(b) γίνεται ανανέωση των κέντρων των συστάδων και στην συνέχεια γίνεται επαναληπτική εύρεση των κέντρων των συστάδων που φαίνονται στο σχήμα 2.7(c)(το κέντρο κάθε συστάδας είναι το σύμβολο +).

2.5 Ο αλγόριθμος "Μείωση μέσω ομοιογενών συστάδων" (RHC)

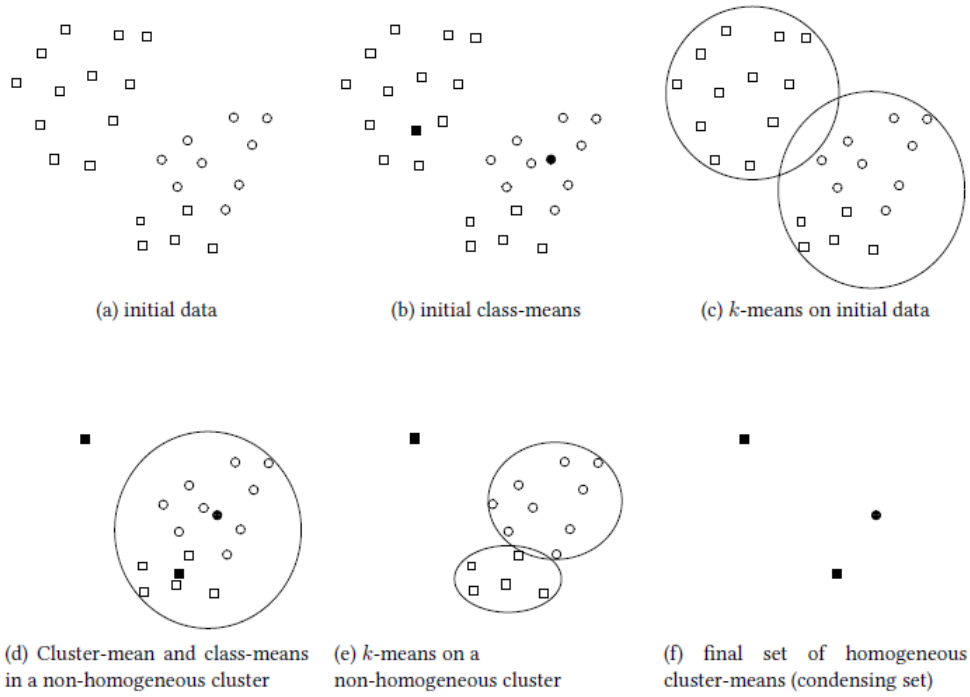
Ο αλγόριθμος "Μείωσης μέσω ομοιογενών συστάδων" (RHC) είναι ένας μη παραμετρικός αλγόριθμος παραγωγής προτύπων με σκοπό την αποτελεσματική κατηγοριοποίηση Κ εγγύτερων γειτόνων. Βασίζεται σε μια απλή ιδέα που εφαρμόζει αναδρομικά τον γνωστό αλγόριθμο Κ μέσων. Ειδικότερα, ο RHC συνεχίζει να κατασκευάζει συστάδες έως ότου αυτές γίνουν ομοιογενείς, δηλαδή (περιέχουν στιγμιότυπα μόνο μιας συγκεκριμένης ετικέτας).

Αρχικά, ο RHC θεωρεί το σύνολο στιγμιοτύπων εκπαίδευσης ως μη ομοιογενής συστάδα.
Ο αλγόριθμος αρχίζει με τον υπολογισμό του μέσου (κέντρου) για κάθε ετικέτα υπολογίζοντας το μέσο όρο των χαρακτηριστικών των αντίστοιχων στιγμιοτύπων του συνόλου εκπαίδευσης. Επομένως, για ένα σύνολο δεδομένων με n ετικέτες, ο αλγόριθμος υπολογίζει n κέντρα ετικετών. Στη συνέχεια, ο RHC εκτελεί τον αλγόριθμο Κ μέσων και δημιουργεί n συστάδες χρησιμοποιώντας τα n κέντρα ετικετών ως αρχικά μέσα για τον αλγόριθμο Κ μέσων. Αν μια συστάδα είναι ομοιογενή, δηλαδή περιλαμβάνει στιγμιότυπα μόνο μιας ετικέτας, το κέντρο της συστάδας αποτελεί πρότυπο και τοποθετείται στο σύνολο συμπύκνωσης. Συνεπώς, τα κέντρα των ομοιογενών συστάδων συνθέτουν το σύνολο συμπύκνωσης (condensing set). Για κάθε μη ομοιογενής συστάδα, τα ανωτέρω εφαρμόζονται αναδρομικά. Ο αλγόριθμος ολοκληρώνει την εκτέλεση του, όταν όλες οι συστάδες είναι ομοιογενής. Στο τέλος, το σύνολο συμπύκνωσης περιέχει όλα τα μέσα των στιγμιοτύπων των ομοιογενών συστάδων. Σημειώνεται ότι χρησιμοποιώντας τα μέσα ετικετών, ως αρχικά μέσα για την συσταδοποίηση, με τον αλγόριθμο Κ μέσων, ο αριθμός των συστάδων καθορίζεται αυτόματα [22].

Το μέσο στιγμιότυπο m κάθε συστάδας ή ετικέτας C, υπολογίζεται με βάση την μέση τιμή των n χαρακτηριστικών τιμών των στιγμιότυπων x_i, $i = 1, 2...|C|$ που ανήκουν στο C. Πιο τυπικά, τα n χαρακτηριστικά $m.d_j$ υπολογίζονται ως εξής:

$$m.d_j = \frac{1}{|C|} \sum_{x_i \epsilon C} x_i.d_j, j = 1, 2....., n \tag{2.17}$$

Το σχήμα 2.8 παρουσιάζει ένα δισδιάστατο παράδειγμα εκτέλεσης του RHC. Αν θεωρηθεί ότι ένα σύνολο δεδομένων περιέχει είκοσι έξι (26) στοιχεία δύο (2) ετικετών: τετράγωνα και κύκλους (σχήμα 2.8(a)). Ο RHC υπολογίζει το μέσο μιας ετικέτας για τα τετράγωνα και το μέσο μιας ετικέτας για τους κύκλους (σχήμα 2.8(b)). Στη συνέχεια, ο αλγόριθμος Κ μέσων συσταδοποίησης,



Σχήμα 2.8: Αφαίρεση δεδομένων μέσω του RHC.

χρησιμοποιεί τα δύο μέσα ετικετών, ως αρχικά μέσα και κατασκευάζει δύο συστάδες. Η μια συστάδα περιέχει μόνο τετράγωνα, ενώ η άλλη συστάδα περιέχει στιγμιότυπα και των δύο ετικετών (σχήμα 2.8(c)). Για την ομοιογενής συστάδα, ο RHC αποθηκεύει το μέσο της συστάδας στο σύνολο συμπύκνωσης ως πρότυπο με ετικέτα τετράγωνο (2.8(d)). Για τα στιγμιότυπα της μη ομοιογενούς συστάδας, ο RHC δημιουργεί δύο ομοιογενείς συστάδες (σχήματα 2.8(d,e)) ακολουθώντας την ίδια διαδικασία. Κατά συνέπεια, δύο ακόμη πρότυπα αποθηκεύονται στο σετ συμπύκνωσης. Έτσι, το τελικό σύνολο συμπύκνωσης περιέχει τρία πρότυπα από τα είκοσι έξι στιγμιότυπα του αρχικού συνόλου εκπαίδευσης (σχήμα 2.8(f)). Προφανώς, ο αλγόριθμος RHC παράγει πολλά πρότυπα για περιοχές δεδομένων κοντά στα σύνορα απόφασης και λίγα πρότυπα για τις ”κεντρικές” περιοχές δεδομένων ετικετών. Ως εκ τούτου, οσο περισσότερες ετικέτες και θόρυβος υπάρχει στα δεδομένα, τόσο περισσότερα σύνορα υπάρχουν και, ως εκ τούτου, επιτυγχάνεται χαμηλότερο ποσοστό μείωσης. Στην πραγματικότητα, όταν ο αλγόριθμος εκτελείται σε ένα σύνολο δεδομένων χωρίς θόρυβο, σχηματίζει λίγες μεγάλες συστάδες. Από την άλλη, εάν χρησιμοποιείται ένα σύνολο δεδομένων με υψηλό επίπεδο θορύβου, κατασκευάζονται πολλές μικρές συστάδες. Επιπλέον, με τη χρηση των κέντρων των ετικετών ως αρχικά μέσα για την συσταδοποίηση Κ μέσων, ο RHC αυξάνει την πιθανότητα γρήγορης εύρεσης μεγάλων ομοιογενών συστάδων και επιτυγχάνει υψηλό ποσοστό μείωσης (όσο μεγαλύτερες είναι οι ομοιογενείς συστάδες που κατασκευάζονται, τόσο υψηλότερος ρυθμός μείωσης επιτυγχάνεται). [12].

Ο αλγόριθμος 3 εμφανίζει μια μη αναδρομική υλοποίηση του RHC. Χρησιμοποιεί μια δομή δεδομένων ουράς (Queue), για να χωρέσει τις μη επεξεργασμένες συστάδες. Αρχικά, ολόκληρο το σύνολο εκπαίδευσης (TS) αποτελεί μια μη επεξεργασμένη συστάδα και τοποθετείται στην ουρά (γραμμή 3). Σε κάθε επανάληψη (repeat-until), ο RHC αφαιρεί απο την ουρά, την συστάδα

Algorithm 3 RHC

Input: Training Set (TS), **Output:** Condensing Set (CS)

```
1:  {Stage 1: Queue Initialization}
2:  Queue ← ∅
3:  Enqueue(Queue, TS)
4:  {Stage 2: Construction of condensing set}
5:  CS ← ∅
6:  repeat
7:     C ← Dequeue(Queue)
8:     if C is homogeneous then
9:        r ← mean of C
10:       CS ← CS ∪ {r}
11:    else
12:       M ← ∅ {M is the set of class-means}
13:       for each class L in C do
14:          m_L ← mean of L
15:          M ← M ∪ {m_L}
16:       end for
17:       NewClusters ← K-MEANS(C, M)
18:       for each cluster C ∈ NewClusters do
19:          Enqueue(Queue, C)
20:       end for
21:    end if
22: until IsEmpty(Queue)
23: return CS
```

C από την κεφαλή της ουράς (γραμμή 7) και ελέγχει αν η συστάδα C είναι ομοιογενής ή όχι. Εάν είναι (γραμμή 8), ο μέσος της συστάδας τοποθετείται στο σύνολο συμπύκνωσης (CS) ως πρότυπο (γραμμή 10) και τα στιγμιότυπα καταργούνται. Διαφορετικά, ο RHC υπολογίζει μια λίστα απο κέντρα ετικετών (M), ένα για κάθε ξεχωριστή ετικέτα που υπάρχει στο σύνολο C (γραμμές 13–16). Στη συνέχεια, ο RHC καλεί τον αλγόριθμο Κ μέσων, με παραμέτρους την τρέχουσα μη ομοιογενής συστάδα C και τη λίστα των αρχικών μέσων ετικέτας M που χρησιμοποιείται ως αρχικά μέσα. Το αποτέλεσμα είναι ένα νέο σύνολο μη επεξεργασμένων συστάδων (NewClusters) (γραμμή 17), όλα αυτά τοποθετούνται στην ουρά (γραμμές 18–20). Ο βρόχος επανάληψης συνεχιζεται μέχρι η ουρά να παραμείνει κενή (γραμμή 22), δηλαδή, μέχρι να μην υπάρχουν πλέον μη ομοιογενείς συστάδες [12].

3 Τεχνικές Μείωσης Δεδομένων Πολλαπλών Ετικετών: Ανασκόπηση βιβλιογραφίας

Κατα την έρευνα της σχετικής βιβλιογραφίας διαπιστώθηκε ότι ένας σχετικά μικρός αριθμός τεχνικών μείωσης δεδομένων πολλαπλών ετικετών έχουν προταθεί. Παρ'όλα αυτά όπως θα παρουσιαστεί παρακάτω, η απόδοση όλων των τεχνικών εξαρτάται σε μεγάλο βαθμό από διάφορες παραμέτρους που προτείνουν οι συγγραφείς τους. Η παρατήρηση αυτή δεν επιτρέπει να συγκριθούν με τις τεχνικές μείωσης που προτείνονται από αυτή τη διπλωματική εργασία αφού, όπως θα γίνει στη συνέχεια κατανοητό, η απόδοση τους δεν εξαρτάται απο παραμέτρους. Παρόλα αυτά, για την πληρότητα της διπλωματικής εργασίας παρουσιάζονται σε αυτή τη ενότητα.

3.1 Βελτίωση της κατηγοριοποίησης πολλαπλών ετικετών με χρήση του αλγόριθμου KNN σε σενάρια επιλογής πρωτοτύπων με χρήση προτάσεων κλάσης

Σε αυτή την μελέτη προτείνεται απο τους συγγραφείς μια εφαρμογή του Prototype Selection με πολύ διαφορετικό τρόπο. Εδώ, το Prototype Selection (PS) χρησιμοποιείται ως στάδιο προεπεξεργασίας για την επιλογή των πιο υποσχόμενων ετικετών, οι οποίες θα χρησιμοποιηθούν για την κατηγοριοποίηση στο αρχικό σύνολο δεδομένων. Πρέπει να σημειωθεί ότι αυτή η προσέγγιση δεν περιορίζει την ανάπτυξη αλγορίθμων PS καθώς η απόδοσή της, ως διαδικασία δεύτερου σταδίου, επηρεάζεται σε μεγάλο βαθμό από το αρχικό βήμα PS. Στην πραγματικότητα, όσο καλύτερα χρησιμοποιείται ο βασικός αλγόριθμος PS, τόσο καλύτερη είναι η απόδοση που αναμένεται να επιτευχθεί με την διαδικασία αυτή [46].

Τα αποτελέσματα έδειξαν ότι αυτή η τεχνική παρέχει μια αντιστάθμιση μεταξύ της ακρίβειας και της απόδοσης. Στις καλύτερες περιπτώσεις, η στρατηγική ισοσταθμίζει την ακρίβεια της κατηγοριοποίησης KNN με το 30% των αποστάσεων που υπολογίζονται. Επιπλέον, κατά την παρουσία θορύβου, η αναζήτησή επιτυγχάνει αξιοσημείωτη απόδοση, καθώς, σε συνδυασμό με τον κατάλληλο αλγόριθμο PS, βελτιώνει την κατηγοριοποίηση του KNN με υψηλότερη απόδοση. Επιπλέον, σε όλες τις εξεταζόμενες περιπτώσεις, οι στατιστικές δοκιμές αποκάλυψαν ότι η ακρίβεια του KNN είναι σημαντικά καλύτερη από αυτή που μπορεί επιτευχτεί με μόνο PS [46].

3.2 Μελέτη των τεχνικών μετασχηματισμού δεδομένων για την προσαρμογή αλγόριθμων επιλογής προτύπων μονής ετικέτας σε κατηγοριοποίηση πολλαπλών ετικετών

Σε αυτή τη μελέτη οι συγγραφείς εφάρμοσαν τις πιο συνηθισμένες τεχνικές μετασχηματισμού δεδομένων σε σύνολα δεδομένων πολλαπλών ετικετών ως προκαταρκτικό στάδιο, όπως γίνεται σε σύνολα μονής ετικέτας. Πιο συγκεκριμένα οι μέθοδοι μετασχηματισμού που χρησιμοποιήθηκαν είναι η δυαδική συνάφεια (Binary Relevance), η εξαρτημένη δυαδική συνάφεια, η μέθοδος

των δυναμοσύνολων ετικετών (label powerset) και η k-labelsets (random k-labelsets) [47].

Τα αποτελέσματα που προέκυψαν από τη σύγκριση των νέων μεθόδων (με βάση τον μετασχηματισμό δεδομένων) με τις προϋπάρχουσες μεθόδους (με βάση την προσαρμογή της μεθόδου), τους έδωσαν τα ακόλουθα αποτελέσματα:

- Η μέθοδος μετασχηματισμού με χρήση δυναμοσυνόλων ετικετών είναι εντελώς μη αποδεκτή στην περίπτωση των αλγορίθμων επιλογής προτύπων, εκτός εάν βρεθεί κάποια τεχνική, για να αποφευχθεί, η λανθασμένη αναγνώριση κάποιων στιγμιότυπων, ως θόρυβος, κατα κύριο λόγο στις υποεκπροσωπούμενες ετικέτες.

- Η μέθοδος μετασχηματισμού Κ τυχαίων συνόλων ετικετών είναι λιγότερο ”επιθετική” απο την προηγούμενη, αλλά, γενικά, ”πάσχει” από παρόμοια προβλήματα.

- Η εξαρτημένη δυαδική συνάφεια προσφέρει ελαφρώς καλύτερα αποτελέσματα από την παραδοσιακή δυαδική συνάφεια, οπότε συνιστάται.

- Η χρήση του αλγόριθμου επιλογής προτύπων, για παράδειγμα του local set-based smoother (LSSm) [48], μπορεί να προσφέρει παρόμοια καλά αποτελέσματα όταν προσαρμόζεται μέσω δυαδικής συνάφειας ή μέσω Κ τυχαίων συνόλων ετικετών.

- Καμία μέθοδος επιλογής πρωτοτύπου δεν είναι ικανή να βελτιώσει τα αποτελέσματα, για όλα τα μέτρα μέτρησης ενός κατηγοριοποιητής, που έχει εκπαιδευτεί στο αρχικό σύνολο δεδομένων. Τέλος οι συγγραφείς καταληγουν, ότι αυτή τη στιγμή, οι υπάρχουσες μέθοδοι επιλογής προτύπου για πολλαπλές ετικέτες, δεν παρέχουν ακόμη τα πλεονεκτήματα που έχουν δώσει οι αντίστοιχοι μέθοδοι, στην κατηγοριοποίηση μονής ετικέτας.

3.3 Τοπικά σύνολα για επιλογή προτύπων (Local sets for prototype selection)

Σε αυτήν την μελέτη οι συγγραφείς προτείνουν μια προσαρμογή της έννοιας του τοπικού συνόλου [49] σε δεδομένα πολλαπλών ετικετών και η αποτελεσματικότητά της επιβεβαιώνεται στο σχεδιασμό δύο νέων αλγορίθμων (HDLSSm, HDLSBo) [50] που απέδωσαν ανταγωνιστικά αποτελέσματα. Μία από τις προσαρμογές ”καθαρίζει” τα σύνολα δεδομένων, για να βελτιώσει τις προγνωστικές δυνατότητές τους, ενώ η άλλη στοχεύει στη μείωση του μεγέθους του συνόλου δεδομένων. Και οι δύο δοκιμάζονται και συγκρίνονται με μεθόδους επιλογής προτύπων που είναι διαθέσιμες στην βιβλιογραφία για εκμάθηση συνόλων πολλαπλών ετικετών [50].

Σε αυτή την μελέτη έκαναν χρήση δυο κατηγοριοποιητών, του MLkNN [51] και του IBLR-ML [52] για να αξιολογήσουν την ποιότητα των επιλεγμένων υποσυνόλων. Επέλεξαν αυτούς τους δυο, γιατί θεωρούνται καλοί κατηγοριοποιητές όταν συνδυάζονται με μεθόδους επιλογής προτύπων. Τα πειράματα που πραγματοποίησαν, τους έδειξαν ότι και οι δύο αλγόριθμοι μπορούν να βελτιώσουν τις προγνωστικές ικανότητες του αρχικού συνόλου δεδομένων για ορισμένες τιμές [50].

- Ο HDLSSm, ως αλγόριθμος επεξεργασίας (editing), επιτυγχάνει χαμηλότερη συμπίεση, αλλά υψηλή ακρίβεια.

- Ο HDLSBo επιτυγχάνει υψηλότερα ποσοστά μείωσης σε βάρος της μείωσης της ακρίβειας.

3.4 Παραγωγή προτύπων για σύνολα πολλαπλών ετικετών βασισμένα στο Granular Computing

Σε αυτή την μελέτη οι συγγραφείς προτείνουν τρείς αλγόριθμους παραγωγής προτύπων (GP1mlTS, GP2mlTS, GP3mlTS) για σύνολα δεδομένων πολλαπλών ετικετών βασισμένα σε granular computing [53].

Για την διεξαγωγή των πειραμάτων χρησιμοποιήθηκε ο αλγόριθμος κατηγοριοποίησης ML-KNN. Επίσης σαν μετρική χρησιμοποιήθηκε η απώλεια Hamming Loss. Στα πειράματα χρησιμοποιήθηκαν δώδεκα (12) σύνολα δεδομένων. Για κάθε σύνολο δεδομένων υπολογίστηκε η απώλεια Hamming Loss (HL) με την χρήση της 10-fold cross validation. Το σύνολο εκπαίδευσης (training set) χρησιμοποιήθηκε για την παραγωγή προτύπων. Από τα αποτελέσματα συμπέραναν ότι, οι μέθοδοι τους επιτυγχάνουν ποσοστό μείωσης δεδομένων υψηλότερο από 20% στα περισσότερα σύνολα δεδομένων. Οι μέθοδοι GP1mlTS και GP2mlTS έχουν παρόμοια συμπεριφορά. Ωστόσο, η μέθοδος GP3mlTS πέτυχε ποσοστό μείωσης δεδομένων υψηλότερο από 90% [53].

Επίσης απο τα αποτελέσματα της μελέτης φαίνεται ότι τα πρότυπα που δημιουργούνται για κάθε σύνολο δεδομένων οδηγούν σε τιμές HL παρόμοιες με αυτές που λαμβάνονται στο αρχικό σύνολο δεδομένων. Μόνο στην περίπτωση ενός συνόλου δεδομένων, υπάρχει σημαντική διαφορά στην τιμή HL, ειδικά όταν χρησιμοποιούν το σύνολο προτύπων που δημιουργείται από τη μέθοδο GP3mlTS. Αυτό οφείλεται στο γεγονός ότι αυτό το σύνολο δεδομένων έχει λίγες ετικέτες, έτσι δημιουργούνται λίγα πρότυπα. Εν ολίγοις, τα αποτελέσματα έδειξαν ότι η παραπάνω μέθοδος, παρέχει μια καλή εξισορρόπηση, μεταξύ της απόδοσης του αλγορίθμου και του αριθμού των στιγμιότυπων εκπαίδευσης στο σύνολο δεδομένων [53].

3.5 Επεξεργασία των δεδομένων εκπαίδευσης και κατηγοριοποίηση πολλαπλών ετικετών με βάση τους K-εγγύτερους γείτονες

Οι συγγραφείς της μελέτης αυτής, προτείνουν μια πρωτότυπη μέθοδο για την επεξεργασία δεδομένων με πολλαπλές ετικέτες αναγνωρίζοντας και εξαλείφοντας λανθασμένα ή ”ανώμαλα” στιγμιότυπα. Ο σκοπός αυτής της μεθόδου είναι τριπλός:

1. Την αύξηση της ποιότητας των στιγμιότυπων του συνόλου εκπαίδευσης κάνοντας τα πιο αξιόπιστα.

2. Τη βελτίωση των επιδόσεων του κατηγοριοποιητή που εφαρμόζεται στα προκύπτοντα δεδομένα εκπαίδευσης.

3. Την αύξηση του χρόνο απόκρισης του κατηγοριοποιητή.

Αυτή η μέθοδος βασίζεται στον κανόνα του K-εγγύτερου γείτονα (K-NN) για την κατηγοριοποίηση πολλαπλών ετικετών και σε ένα κριτήριο αξιολόγησης που χρησιμοποιείται τοπικά στο σύνολο Nx των K-εγγύτερων γειτόνων του x, για την αξιολόγηση της ποιότητας ενός στιγμιότυπου x. Με βάση αυτό το κριτήριο αξιολόγησης, καταφέρνουν να διαγράψουν τα πιο ”άσχετα” ή τα χειρότερα στιγμιότυπα, από το αρχικό σύνολο δεδομένων εκπαίδευσης. [54].

Στην παραπάνω μέθοδο εφαρμόσανε τον κατηγοριοποιητή EMLkNN [55] και τον Rank-SVM [56]. Από τα αποτελέσματα των πειραμάτων παρατηρήσαν ότι ο EMLkNN που εφαρμόζεται σε επεξεργασμένα δεδομένα, βελτιώνει την απόδοση για όλες τις μετρικές που βασίζονται σε προβλέψεις, εκτός από την μετρική HL, καθώς αυτό παραμένει ίδιο, πριν και μετά, την επεξεργασία.

- Όσον αφορά την μέθοδο Rank-SVM, τα αποτελέσματα στα επεξεργασμένα σύνολα δεδομένων, είναι καλύτερα από εκείνα, των αρχικών συνόλων δεδομένων για όλα τα μέτρα (μετρικές βάση πρόβλεψης και μετρικές βάση κατάταξης).

- Ο Rank-SVM όταν εφαρμόζεται σε επεξεργασμένα σύνολα δεδομένων, δίνει καλύτερη απόδοση στην πλειοψηφία των μέτρων αξιολόγησης, στα περισσότερα σύνολα δεδομένων, που χρησιμοποίησαν στην πειραματική τους μελέτη. Για ένα σύνολο δεδομένων επιτεύχθει, καλύτερη απόδοση στις μετρικές που βασίζονται στην κατάταξη με τη μέθοδο Rank-SVM, ενώ τα καλύτερα αποτελέσματα σύμφωνα με τις μετρικές βάση πρόβλεψης, ελήφθησαν από τον αλγόριθμο EMLkNN. Επίσης παρατήρησαν ότι ο χρόνος λειτουργίας των δύο κατηγοριοποιητών (EMLkNN και Rank-SVM) μειώνεται σημαντικά στα πειράματά τους, εκτός από ένα σύνολο δεδομένων. Σε γενικές γραμμές κατέληξαν ότι, ο EMLkNN είναι ταχύτερος από τον Rank-SVM [54].

4 Προτεινόμενες Τεχνικές και Αλγόριθμοι

Οι ”συμβατικοί” αλγόριθμοι μείωσης δεδομένων δεν είναι κατάλληλοι για να εφαρμοστούν σε συνδυασμό με κάποια μέθοδο μετασχηματισμού προβλήματος, σε δεδομένα πολλαπλών ετικετών, αφού αυτό θα είχε σαν αποτέλεσμα την δημιουργία πολλών συμπυκνωμένων συνόλων, συνήθως ένα για κάθε ετικέτα. Συνεπώς, το επιθυμητό αποτέλεσμα που είναι η μείωση των δεδομένων δεν επιτυγχάνεται. Στόχος της εργασίας είναι το να προτείνει νέους αλγορίθμους μείωσης δεδομένων πολλαπλών ετικετών, καθώς και κατηγοριοποιητές εγγύτερων γειτόνων που χρησιμοποιούν τα συμπυκνωμένα σύνολα που παράγουν οι προτεινόμενοι αλγόριθμοι για να επιτύχουν υψηλή ακρίβεια.

Οι προτεινόμενοι αλγόριθμοι μείωσης δεδομένων ανήκουν στην κατηγορία παραγωγής προτύπων και ονομάστηκαν Multilabel Reduction through Homogeneous Clustering 1 (MRHC1) και Multilabel Reduction through Homogeneous Clustering 2 (MRHC2). Όπως εύκολα γίνεται κατανοητό, από το όνομα που δόθηκε στους αλγορίθμους, και οι δύο προτεινόμενοι αλγόριθμοι αποτελούν παραλλαγές του αλγορίθμου RHC που παρουσιάστηκε αναλυτικά στο Κεφάλαιο 2.

Ο MRHC1 παράγει πρότυπα που ανήκουν σε μια ετικέτα. Τελικά, τα πρότυπα αυτά χρησιμοποιούνται από τους προτεινόμενους κατηγοριοποιητές Multilabel KNN1 (MKNN1) και Multilabel KNN2 (MKNN2) οι οποίοι, αναζητώντας τα εγγύτερα πρότυπα, τελικά αποφασίζουν για το ποιες ετικέτες θα αποδοθούν στο υπο κατηγοριοποίηση στιγμιότυπο. Συνεπώς, ο αλγόριθμος MRHC1 μπορεί να χρησιμοποιηθεί σε συνδυασμό είτε με τον MKNN1 είτε με τον MKNN2. Αντίθετα, ο MRHC2 παράγει πρότυπα πολλαπλών ετικετών. Τα πρότυπα αυτά μπορούν να χρησιμοποιηθούν είτε από τον ”συμβατικό” KNN είτε από τον προτεινόμενο κατηγοριοποιητή με όνομα Multilabel KNN 2 (MKNN2). Το παρόν κεφάλαιο παρουσιάζει αναλυτικά τους προαναφερθέντες αλγορίθμους, οι οποίοι αποτελούν την συνεισφορά της παρούσας διπλωματικής εργασίας.

4.1 Ο Αλγόριθμος MRHC1

Αρχικά, το σύνολο δεδομένων εκπαίδευσης θεωρείται μια μη-ομοιογενής συστάδα. Για κάθε ετικέτα, ο αλγόριθμος βρίσκει το κέντρο της συγκεκριμένης ετικέτας, βρίσκοντας τον μέσο όρο κάθε διάστασης των στιγμιοτύπων που ανήκουν στην συγκεκριμένα ετικέτα. Αν το σύνολο δεδομένων εκπαίδευσης έχει δέκα ετικέτες, ο αλγόριθμος θα υπολογίσει δέκα κέντρα ετικετών. Στη συνέχεια, ο MRHC1 εκτελεί συσταδοποίηση K-μέσων χρησιμοποιώντας τα κέντρα ετικετών ως αρχικά μέσα για τον αλγόριθμο K-μέσων και βρίσκει τόσες συστάδες όσα και τα κέντρα ετικετών. Αν προκύψει συστάδα που είναι ομοιογενής, τότε το κέντρο της συστάδας αποτελεί πρότυπο και αποθηκεύεται στο συμπυκνωμένο σύνολο (condensing set).

Μια συστάδα θεωρείται ομοιογενής, όταν όλα τα στιγμιότυπα της συστάδας διαθέτουν, κοινή μια ετικέτα. Αυτό φαίνεται στον Αλγόριθμο 4 στις γραμμές (6-12) όπου εκτελείται ο έλεγχος του κατα πόσο η συστάδα είναι ομοιογενής ή όχι. Συνεπώς, το πρότυπο που αποθηκεύεται στο συμπυκνωμένο σύνολο έχει ως ετικέτες την κοινή ετικέτα.

Για παράδειγμα, αν μια ομοιογενής συστάδα αποτελείται από τα ακόλουθα στιγμιότυπα τα οποία διαθέτουν 3 διαστάσεις (γνωρίσματα) (a,b,c) και 3 ετικέτες (x,y,z), το 1 δηλώνει ότι το στιγμιότυπο διαθέτει την αντίστοιχη ετικέτα, ενώ το 0 ότι δεν τη διαθέτει). Έστω λοιπόν τα στιγμιότυπα:

- 1ο στιγμιότυπο: $\{a, b, c, x, y, z\}$ = $\{2, 1, 1, 0, 1, 1\}$

- 2ο στιγμιότυπο: $\{a, b, c, x, y, z\}$ = $\{1, 2, 3, 0, 1, 0\}$

- 3ο στιγμιότυπο: $\{a, b, c, x, y, z\}$ = $\{3, 2, 1, 1, 1, 1\}$

Τότε το πρότυπο που θα αποθηκευτεί στο συμπυκνωμένο σύνολο μετα την εφαρμογή του αλγορίθμου MRHC1 θα είναι το: $\{a, b, c, x, y, z\}$ = $\{2, 1.67, 1.33, 0, 1, 0\}$. Στο πρότυπο θα αποδοθεί μόνο η ετικέτα y αφού αυτή είναι η κοινή ετικέτα η οποία προκάλεσε την ομοιογενοποίηση της συστάδας.

Για κάθε μη ομογενής συστάδα, η παραπάνω διαδικασία επαναλαμβάνεται αναδρομικά. Δηλαδή, ο αλγόριθμος βρίσκει το κέντρο της κάθε ετικέτας, στην συστάδα, βρίσκοντας τον μέσο όρο κάθε διάστασης των στιγμιοτύπων της συστάδας που ανήκουν στην συγκεκριμένη ετικέτα, ο αλγόριθμος συσταδοποίησης Κ-μεσων εκτελείται και παράγει τόσες συστάδες, όσες και οι ετικέτες μέσα στη συστάδα.

Η διαδικασία ολοκληρώνεται όταν όλες οι συστάδες που προκύψουν είναι ομοιογενείς. Τα κέντρα των ομοιογενών συστάδων αποτελούν πρότυπα και συνθέτουν το τελικό συμπυκνωμένο σύνολο. Σε κάθε πρότυπο τοποθετείται η κοινή ετικέτα. Συνεπώς, το κάθε πρότυπο αντιπροσωπεύει μια ετικέτα.

Algorithm 4 MRHC1

Input: TS
Output: CS

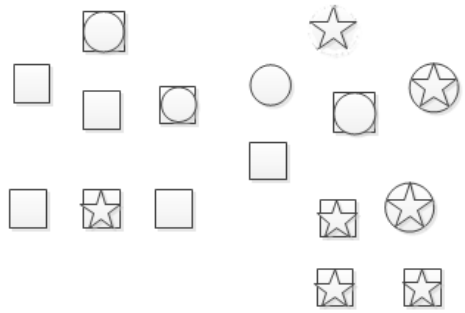
{**Stage 1: Queue Initialization**}
$Queue \leftarrow \varnothing$
Enqueue($Queue$, TS)
{**Stage 2: Construction of condensing set**}
$CS \leftarrow \varnothing$
Function homogeneous(C)
if (instances of Cluster C has all common one label) **then**
  return $TRUE$
else
  return $FALSE$
end if
EndFunction
repeat
  $C \leftarrow$ Dequeue($Queue$)
  if homogeneous(C) $\leftarrow TRUE$ **then**
    $r \leftarrow$ mean of C
    $CS \leftarrow CS \cup \{r\}$
  else
    $M \leftarrow \varnothing$ {M is the set of class-means}
    for each class L in C **do**
      $m_L \leftarrow$ mean of L
      $M \leftarrow M \cup \{m_L\}$
    end for
    $NewClusters \leftarrow K$-MEANS(C, M)
    for each cluster $C \in NewClusters$ **do**
      Enqueue($Queue$, C)
    end for
  end if
until IsEmpty($Queue$)
return CS

Στο σχήμα 4.1 παρουσιάζεται ένα παράδειγμα δυο διαστάσεων, στο οποίο εφαρμόζεται ο αλγόριθμος MRHC1. Αν θεωρηθεί οτι το σύνολο δεδομένων πολλαπλών ετικετών αποτελείται απο δεκαέξι στιγμιότυπα και τρείς ετικέτες (κύκλος, αστέρι, τετράγωνο) (σχήμα 4.1(a)). Ο MRHC1 υπολογίζει ένα μέσο για την ετικέτα κύκλου, ένα μέσο για την ετικέτα αστέρι και ένα μέσο για την ετικέτα τετράγωνο (σχήμα 4.1 (b)). Στην συνέχεια ο αλγόριθμος συσταδοποίησης Κ μέσων, δημιουργεί τρεις συστάδες (σχήμα 4.1(c)) με βάση τα κέντρα του προηγούμενου βήματος. Όλες οι συστάδες περιέχουν στιγμιότυπα που ανήκουν και στις τρεις ετικέτες. Για την πρώτη συστάδα αριστερά, επειδή όλα τα στιγμιότυπα έχουν κοινή την κλάση τετράγωνο (ομοιογενής συστάδα) προκύπτει ένα νέο στιγμιότυπο (ετικέτας τετράγωνο) το οποίο αποθηκεύεται στο σύνολο συμπύκνωσης (condensing set) (σχήμα 4.1(d)). Ενώ για την συστάδα που βρίσκεται κάτω δεξιά, επειδή όλα τα στιγμιότυπα έχουν κοινή την κλάση αστέρι (ομοιογενής συστάδα) προκύπτει ένα νέο στιγμιότυπο (ετικέτας αστέρι) το οποίο αποθηκευεται στο σύνολο συμπύκνωσης (condensing set) (σχήμα 4.1(d)). Στην συνέχεια ο αλγόριθμος λειτουργεί αναδρο-

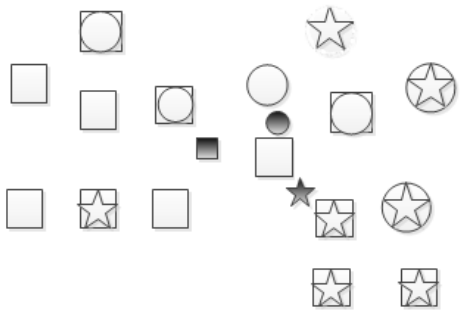
μικά και υπολογίζει τρία νέα μέσα για την ανομοιογενή συστάδα (πάνω δεξιά) που φαίνεται στο (σχήμα 4.1(e)), δηλαδή ένα στιγμιότυπο για κάθε ετικέτα που υπάρχει σε αυτή την συστάδα. Στο αμέσως επόμενο βήμα παρατηρείται οτι δημιουργούνται απο αυτή την ανομοιογενή συστάδα, τρεις διαφορετικές συστάδες (σχήμα 4.1(f)), με στιγμιότυπα που έχουν ετικέτες τον κύκλο, το τετράγωνο και το αστέρι αντίστοιχα. Τελικά πέντε πρότυπα αποθηκεύονται στο σύνολο συμπύκνωσης (condesing set) (σχήμα 4.1(g)). Έτσι το τελικό σύνολο συμπύκνωσης αποτελείται απο πέντε πρότυπα (δυο ετικέτας αστέρι, δυο ετικέτας τετράγωνο και ένα ετικέτας κύκλος) αντί για δεκαέξι στιγμιότυπα που ήταν το αρχικό σύνολο εκπαίδευσης.

Άρα, ενώ το αρχικό σύνολο δεδομένων είναι πολλαπλών ετικετών (multilabel), στο συμπυκνωμένο σύνολο, κάθε πρότυπο ανήκει σε μια κλάση (ετικέτα). Αν σε μια ομοιογενής συστάδα υπάρχουν περισσότερες από μια κοινές ετικέτες, ο MRHC1 επιλέγει ως κοινή ετικέτα την πολυπληθέστερη από αυτές σε ολόκληρο το σύνολο δεδομένων και την τοποθετεί στο πρότυπο.

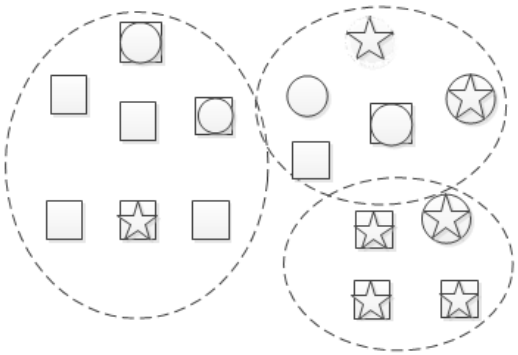
Βασικό μειονέκτημα του MRHC1 είναι ότι υπάρχει περίπτωση κάποιες ετικέτες να μην αντιπροσωπεύονται στο συμπυκνωμένο σύνολο αφού είναι πιθανόν να μην αποτελέσουν καμία φορά κοινή ετικέτα σε κάποια συστάδα. Όσο περισσότερες ετικέτες διαθέτει το σύνολο δεδομένων και όσο υπάρχουν έντονες άνισες κατανομές στις ετικέτες (μια ετικέτα είναι αρκετά δημοφιλής ενώ κάποιες άλλες σπάνιες), το πρόβλημα αυτό γίνεται εντονότερο αφού είναι πιθανόν τελικά το συμπυκνωμένο σύνολο να διαθέτει πρότυπα συγκεκριμένων ετικετών.



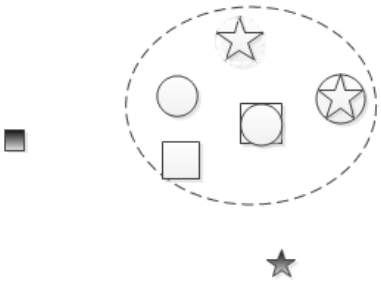
(a) initial data



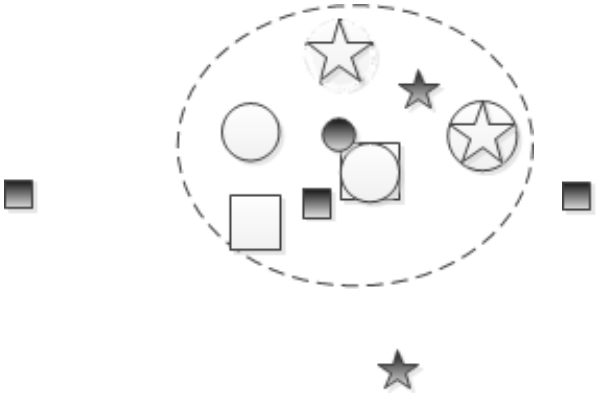
(b) initial class-means



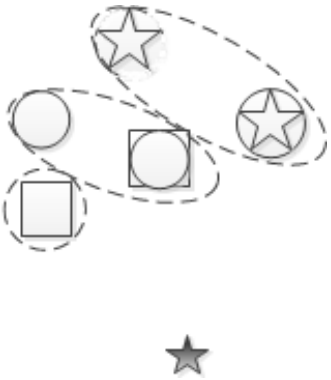
(c) k-means on initial data



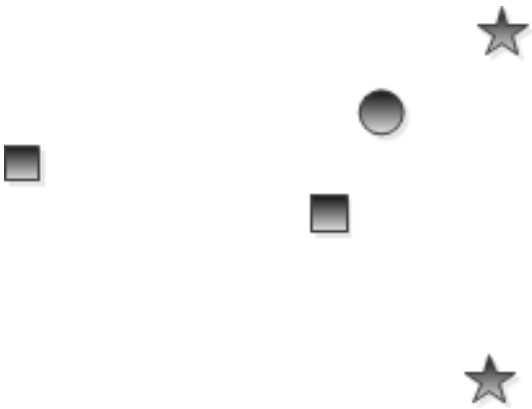
(d) Cluster-mean and class-means in a non-homogeneous cluster



(e) k-Means on the non-homogeneous cluster



(f) The updated condensing set



(g) final set of homogeneous cluster-means (condensing set)

Σχήμα 4.1: Data abstraction through MRHC1

4.2 Ο αλγόριθμος κατηγοριοποίησης MKNN1

Ο αλγόριθμος MKNN1 που προτείνεται, φαίνεται παρακάτω και στην ουσία ακολουθεί την φιλοσοφία των (K) εγγύτερων γειτόνων. Στην εκκίνηση του, ο αλγόριθμος ορίζει οτι το πλήθος (K) των εγγύτερων γειτόνων θα ισούται με το πλήθος των ετικετών των στιγμιότυπων του συνόλου δεδομένων, αυξημένος κατά 1. Άρα αν για παράδειγμα το σύνολο δεδομένων έχει είκοσι (20) ετικέτες, τότε ορίζεται το K=21. Στην συνέχεια ο αλγόριθμος ψάχνει να βρει τους K+1 εγγύτερους γείτονες τους οποίους ταξινομεί κατά αυξουσα σειρά με βάση την ευκλείδια απόσταση, δηλαδή απο την μικρότερη ευκλείδια απόσταση προς την μεγαλύτερη ευκλείδια απόσταση. Κάθε φορά υπολογίζει στους εγγύτερους γείτονες, πόσες φορές εμφανίζεται μία ετικέτα. Δηλαδή αν στον 1ο εγγύτερο εμφανιστούν οι ετικέτες {A,B,C,D}, στον 2ο εγγύτερο γείτονα εμφανιστούν οι ετικέτες {A,E,F,G} τότε ο αλγόριθμος σταματά και αποφασίζει οτι το νέο στιγμιότυπο του συνόλου ελέγχου ανήκει στις ετικέτες {A,B,C,D,E,F,G}. Δηλαδή αφού σε δύο εγγύτερα στιγμιότυπα ανακάλυψε δυο ίδιες κλάσεις, το στιγμιότυπο ελέγχου, θα κατηγοριοποιηθεί στις κλάσεις, που έχουν τα δυο αυτά εγγύτερα στιγμιότυπα.

Προφανώς ότι ο MKNN1 μπορεί να εφαρμοστεί σε δεδομένα πολλαπλών ετικετών, όμως μπορεί να εφαρμοστεί και σε συνδυασμό με τον MRHC1, και συγκεκριμένα στα πρότυπα που ο MRHC1 παράγει, με τον εξής απλό τρόπο: Ο αλγόριθμος ελέγχει την ετικέτα των κοντινότερων προτύπων μέχρις ότου να βρεθεί πρότυπο με ίδια ετικέτα με την ετικέτα ενός προηγούμενου πρότυπου. Αν για παράδειγμα, το κοντινότερο πρότυπο έχει την ετικέτα A και το δεύτερο κοντινότερο πρότυπο έχει την ετικέτα B, ο αλγόριθμος εξετάζει και την ετικέτα του τρίτου κοντινότερου πρότυπου. Αν αυτή είναι A ή B, ο αλγόριθμος σταματάει και στο προς κατηγοριοποίηση στιγμιότυπο αποδίδονται και οι δύο ετικέτες. Διαφορετικά, ο αλγόριθμος συνεχίζει ελέγχοντας την ετικέτα του τέταρτου κοντινότερου προτύπου κοκ. Μόλις ο αλγόριθμος συναντήσει μια ετικέτα προτύπου που έχει συναντήσει και πιο πριν, ο αλγόριθμος σταματάει και στο στιγμιότυπο που πρέπει να κατηγοριοποιηθεί αποδίδονται οι ετικέτες όλων των προηγούμενων προτύπων. Στην ακραία περίπτωση που κάθε κοντινότερο πρότυπο είναι διαφορετικής ετικέτας, ο αλγόριθμος σταματάει ελέγχοντας το $K+1$ κοντινότερο πρότυπο το οποίο, από τη στιγμή που το K ορίστηκε να έχει ως τιμή το πλήθος των ετικετών αυξημένο κατά ένα, είναι βέβαιο ότι η ετικέτα θα έχει εμφανιστεί σε έναν από τα προηγούμενα πρότυπα.

4.3 Ο Αλγόριθμος κατηγοριοποίησης MKNN2

Ο αλγόριθμος MKNN2 βασίζεται και αυτός στην αναζήτηση εγγύτερων γειτόνων και έχει κάποια όμοια στοιχεία με αυτά του MKNN1. Στην εκκίνηση του, ο αλγόριθμος MKNN2 ορίζει οτι το πλήθος K των εγγύτερων γειτόνων θα ισούται με το πλήθος των ετικετών των στιγμιότυπων του συνόλου δεδομένων. Συνεπώς αν για παράδειγμα το σύνολο δεδομένων έχει είκοσι (20) ετικέτες, ο αλγόριθμος θέτει $K = 20$. Στην συνέχεια, ο αλγόριθμος ψάχνει να ανακαλύψει τους K εγγύτερους στο στιγμιότυπο, προς κατηγοριοποίηση, γείτονες τους οποίους ταξινομεί κατα αύξουσα σειρά, με βάση την ευκλείδια απόσταση, δηλαδή απο την μικρότερη ευκλείδια απόσταση, προς την μεγαλύτερη ευκλείδια απόσταση. Κάθε φορά αποθηκεύει τις ετικέτες όλων

Algorithm 5 MKNN1

Input: X:training data,x:unknown sample
Output: $Pred$:class labels prediction for x

1: $k \leftarrow 1$
2: $m_{max} \leftarrow$ number of labels + 1
3: $flag \leftarrow FALSE$
4: $Pred \leftarrow \varnothing$
5: **while** $(k \leq m_{max})$ and $(flag == FALSE)$ **do**
6: $L \leftarrow$ labels of k-th nearest to x neighbor
7: **if** There is $l \in L$ exists in $Pred$ **then**
8: $flag \leftarrow TRUE$
9: **else**
10: $Pred \leftarrow Pred \cup \{l\}$
11: $k++$
12: **end if**
13: **end while**
14: **return** $Pred$

των εγγύτερων γείτονων. Δηλαδή αν στον 1ο εγγύτερο εμφανιστούν οι ετικέτες {A,B,C,D}, στον 2ο εγγύτερο γείτονα εμφανιστούν οι ετικέτες {A,E,F,G} και στον K εγγύτερο γείτονα εμφανιστούν οι ετικέτες {A,B,C,D,E,K,L} τότε ο αλγόριθμος αποφασίζει οτι το νέο στιγμιότυπο του συνόλου ελέγχου, ανήκει στις ετικέτες {A,B,C,D,E,F,G,K,L}. Δηλαδή, σε αυτή την περίπτωση το στιγμιότυπο ελέγχου θα κατηγοριοποιηθεί σε όλες τις ετικέτες που έχουν όλα τα εγγύτερα στιγμιότυπα.

Algorithm 6 MKNN2

Input: X:training data, x:unknown sample, k:number of nearest examined neighbors
Output: $Pred$:class labels prediction for x

1: $Pred \leftarrow$ labels of k nearest to x neighbors in X
2: **return** $Pred$

4.4 Ο Αλγόριθμος MRHC2

Ο αλγόριθμος MRHC2 είναι παρόμοιος με τον MRHC1. Ωστόσο, το συμπυκνωμένο σύνολο που παράγει είναι πολλαπλών ετικετών. Όπως και στον MRHC1, αρχικά, το σύνολο δεδομένων εκπαίδευσης θεωρείται μια μη-ομοιογενής συστάδα και για κάθε ετικέτα, ο αλγόριθμος MRHC2 βρίσκει το κέντρο της συγκεκριμένης ετικέτας βρίσκοντας τον μέσο όρο κάθε διάστασης των στιγμιοτύπων που ανήκουν στην συγκεκριμένα ετικέτα. Στη συνέχεια, ο MRHC2, όπως ο MRHC1, εκτελεί συσταδοποίηση K-μέσων χρησιμοποιώντας τα κέντρα ετικετών, ως αρχικά μέσα και βρίσκει τόσες συστάδες όσα και τα κέντρα ετικετών. Για κάθε ομοιογενή συστάδα, το κέντρο της συστάδας αποτελεί πρότυπο και αποθηκεύεται στο συμπυκνωμένο σύνολο.

Όπως ο MRHC1, έτσι και ο MRHC2 θεωρεί ότι μια συστάδα είναι ομογενής όταν όλα τα στιγμιότυπα της συστάδας διαθέτουν τουλάχιστον μια κοινή ετικέτα. Ο MRHC2 θα αποθηκεύσει στο συμπυκνωμένο σύνολο το κέντρο της συστάδας ως πρότυπο. Ωστόσο, σε αντίθεση με τον MRHC1, θα αποδώσει στο πρότυπο την κοινή ετικέτα καθώς επίσης και τις ετικέτες που διαθέτουν τα περισσότερα απο τα μισά στιγμιότυπα στην συστάδα. Αυτό φαίνεται στον Αλγόριθμο 7 στις γραμμές (6-12) όπου εκτελείται ο έλεγχος του κατα πόσο η συστάδα είναι ομοιογενής ή όχι. Για παράδειγμα, αν μια συστάδα αποτελείται από τα ακόλουθα στιγμιότυπα τα οποία διαθέτουν τρείς (3) διαστάσεις (γνωρίσματα) (a,b,c) και τρείς (3) ετικέτες (x,y,z), το 1 δηλώνει ότι το στιγμιότυπο διαθέτει την αντίστοιχη ετικέτα, ενώ το 0 ότι δεν τη διαθέτει).Έστω λοιπόν τα στιγμιότυπα:

- 1ο στιγμιότυπο: $\{a, b, c, x, y, z\}$ = $\{2, 1, 1, 0, 1, 1\}$
- 2ο στιγμιότυπο: $\{a, b, c, x, y, z\}$ = $\{1, 2, 3, 0, 1, 0\}$
- 3ο στιγμιότυπο: $\{a, b, c, x, y, z\}$ = $\{3, 2, 1, 1, 1, 1\}$

Τότε το πρότυπο που θα αποθηκευτεί στο συμπυκνωμένο σύνολο μετα την εφαρμογή του αλγορίθμου MRHC1 θα είναι το: $\{a, b, c, x, y, z\}$ = $\{2, 1.67, 1.33, 0, 1, 1\}$. Στο πρότυπο θα αποδοθεί η ετικέτα y αφού αυτή είναι η κοινή ετικέτα η οποία προκάλεσε την ομοιογενοποίηση της συστάδας. Επίσης, στο πρότυπο, θα αποδοθεί η ετικέτα z γιατί περισσότερα από τα μισά στιγμιότυπα της συστάδας διαθέτουν την ετικέτα z. Αντίθετα, δεν θα αποδοθεί στο πρότυπο η ετικέτα x αφού λιγότερα από τα μισά στιγμιότυπα διαθέτουν την ετικέτα x.

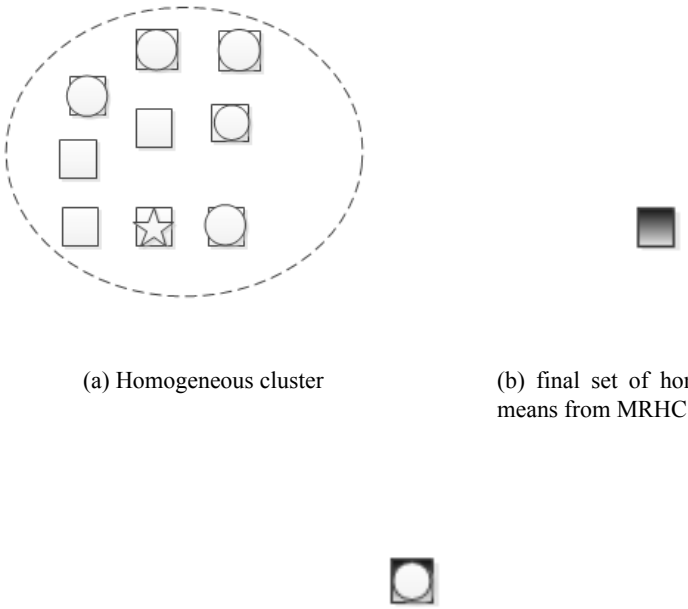
Όπως ο MRHC1, για κάθε μη ομογενής συστάδα, ο MRHC2 επαναλαμβάνει την παραπάνω διαδικασία αναδρομικά για τα στιγμιότυπα της μη-ομοιογενούς συστάδας. Η διαδικασία ολοκληρώνεται όταν όλες οι συστάδες που προκύψουν είναι ομοιογενείς.

Algorithm 7 MRHC2

Input: TS
Output: CS

{**Stage 1: Queue Initialization**}
$Queue \leftarrow \varnothing$
Enqueue($Queue$, TS)
{**Stage 2: Construction of condensing set**}
$CS \leftarrow \varnothing$
Function homogeneous(C)
if (instances with common label of Cluster C > (quantity of instances/2)) **then**
  return $TRUE$
else
  return $FALSE$
end if
EndFunction
repeat
  $C \leftarrow$ Dequeue($Queue$)
  if homogeneous(C) $\leftarrow TRUE$ **then**
    $r \leftarrow$ mean of C
    $CS \leftarrow CS \cup \{r\}$
  else
    $M \leftarrow \varnothing$ {M is the set of class-means}
    for each class L in C **do**
      $m_L \leftarrow$ mean of L
      $M \leftarrow M \cup \{m_L\}$
    end for
    $NewClusters \leftarrow K$-MEANS(C, M)
    for each cluster $C \in NewClusters$ **do**
      Enqueue($Queue$, C)
    end for
  end if
until IsEmpty($Queue$)
return CS

Στο σχήμα (4.2(a)) παρουσιάζεται ένα παράδειγμα δυο διαστάσεων, όπου έχει καταλήξει σε μια ομοιογενή συστάδα που έχει όλες τις ετικέτες. Σε αυτή την φάση, αν εφαρμοστεί η μέθοδος MRHC1 θα προκύψει ένα στιγμιότυπο το οποίο θα είναι ετικέτας τετράγωνο σχήμα (4.2(b), το οποίο θα αποτέλεσει και πρότυπο του τελικού συνόλου συμπύκνωσης (condensing set). Αν εφαρμοστεί ο MRHC2, θα παραχθεί ένα στιγμιότυπο το οποίο θα ανήκει ταυτόχρονα στην ετικέτα τετράγωνο και στην ετικέτα κύκλος, σχήμα (4.2(c), το οποίο, θα αποτελέσει και το πρότυπο του τελικού συμπυκνωμένου συνόλου. Αυτό μας δείχνει ξεκάθαρα οτι στην περίπτωση του MRHC1, η μέθοδος λαμβάνει υπ' όψιν της, όλα τα στιγμιότυπα που έχουν κοινή ετικέτα τον κύκλο, οπότε αυτή είναι η ετικέτα που θα έχει το πρότυπο. Αντίθετα, στην περίπτωση του MRHC2, η μέθοδος λαμβάνει υπ' όψιν της, οτι πάνω απο τα μισά στιγμιότυπα, πέντε, σε σύνολο εννέα στιγμιοτύπων, έχουν κοινή ετικέτα τον κύκλο και το τεράγωνο. Όποτε παράγει ένα πρότυπο με αυτές τις ετικέτες, το οποίο θα αποτέλεσει και το πρότυπο του τελικού συμπυκνωμένου συνόλου.



(a) Homogeneous cluster

(b) final set of homogeneous cluster-means from MRHC1 (condensing set)

(c) final set of homogeneous cluster-means from MRHC2 (condensing set)

Σχήμα 4.2: Data abstraction through MRHC1 and MRHC2 in a homogeneous cluster

4.5 Συνδυασμός MRHC2 με τον συμβατικό κατηγοριοποιητή KNN

Όπως ήδη αναφέρθηκε, τα πρότυπα που παράγονται από τον MRHC2 μπορούν να χρησιμοποιηθούν από τον συμβατικό αλγόριθμο KNN. Συγκεκριμένα, αν υιοθετηθεί αυτή η προσέγγιση, το προς κατηγοριοποίηση στιγμιότυπο τελικά αποκτά τις ετικέτες που κατα πλειοψηφία έχουν τα K εγγύτερα πρότυπα σε αυτό. Εξάλλου, ακριβώς με τον ίδιο τρόπο, ο αλγόριθμος κατηγοριοποίησης KNN θα ανέθετε, στο προς κατηγοριοποίηση, στιγμιότυπο τις ετικέτες της πλειοψηφίας των εγγύτερων στιγμιοτύπων του αρχικού συνόλου δεδομένων εκπαίδευσης αν δεν εφαρμοζόταν μείωση δεδομένων.

Στην πραγματικότητα, από την στιγμή που ο KNN είναι ένας οκνηρός κατηγοριοποιητής, δηλαδή δεν δημιουργεί κάποιο μοντέλο κατηγοριοποίησης, η διαδικασία αυτή που περιγράφηκε είναι όμοια με αυτή της κατηγοριοποίησης KNN σε συνδυασμό με τη μέθοδο μετασχηματισμού προβλήματος Binary Relevance. Σημειώνεται, ότι η μέθοδος μετασχηματισμού Binary Relevance και ο κατηγοριοποιητής KNN δεν μπορεί να συνδυαστεί με κάποια τεχνική μείωσης δεδομένων μονής ετικέτας. Αυτο συμβαίνει επειδή θα έπρεπε να δημιουργηθεί ένα ξεχωριστό σύνολο συμπύκνωσης για κάθε ετικέτα. Με άλλα λόγια, αν υιοθετηθεί μια τέτοια προσέγγιση, αντί να μειωθούν τα δεδομένα πολλαπλών ετικετών, θα προέκυπταν τόσο συμπυκνωμένα σύνολα, όσες και οι ετικέτες, με συνέπεια, σε ορισμένες περιπτώσεις θα προκαλούνταν ακόμη και αύξηση τελικά των δεδομένων εκπαίδευσης αντί για μείωση.

5 Πειραματική Μελέτη

Σε αυτό το κεφάλαιο παρουσιάζεται η πειραματική μελέτη η οποία εκπονήθηκε στα πλαίσια της παρούσας διπλωματικής εργασίας. Ο σκοπός των πειραμάτων είναι να συγκριθεί η απόδοση των δυο νέων αλγορίθμων μείωσης δεδομένων καθώς και των κατηγοριοποιητών που παρουσιάστηκαν στο Κεφάλαιο 4. Συγκεκριμένα, στα πλαίσια της πειραματικής μελέτης, οι αλγόριθμοι MRHC1 και MRHC2 εφαρμόστηκαν σε συνδυασμό με τους MKNN1 και MKNN2 καθώς και με τον συμβατικό KNN σε συνδυασμό με την προσέγγιση μετασχηματισμού πρόβλήματος, Binary Relevance, στο συμπυκωμένο σύνολο που παράγει ο MRHC2. Στο τέλος όλες αυτές οι νέοι μέθοδοι συγκρίθηκαν μεταξύ τους και με τη μέθοδο μετασχηματισμού προβλήματος δυαδικής συνάφειας σε συνδυασμό με τον συμβατικό KNN ο οποίος εκτελείται στο αρχικό σύνολο δεδομένων εκπαίδευσης πολλαπλών ετικετών.

Στην Ενότητα 5.1 δίνεται μια αναλυτική περιγραφή της διαδικασίας που ακολουθήθηκε στην πειραματική μελέτη. Παρουσιάζεται η προετοιμασία των δεδομένων που αφορά την κανονικοποίηση (normalization) τους, ο τρόπος διαχωρισμού των αρχικών συνόλων σε σύνολο εκπαίδευσης και σύνολο δοκιμής, οι μετρικές που χρησιμοποιήθηκαν για την αξιολόγηση των αποτελεσμάτων και τέλος το τελευταίο κομμάτι, αφορά την γλώσσα προγραμματισμού και τις τεχνολογίες που χρησιμοποιήθηκαν για την υλοποίηση των αλγορίθμων.

Στην Ενότητα 5.2 δίνεται μία αναλυτική περιγραφή των συνόλων δεδομένων που χρησιμοποιήθηκαν στα πειράματά μας. Εκτός αυτού, αναφέρονται επίσης τα κριτήρια που έπρεπε να εκπληρώνουν αυτά τα σύνολα για να μπορέσουν να χρησιμοποιηθούν.

Στην Ενότητα 5.3 παρουσιάζονται τα αποτελέσματα των πειραμάτων μας σε μία σειρά αναλυτικών πινάκων. Αναφέρονται οι κατηγορίες στις οποίες χωρίστηκαν τα πειράματα, καθώς και τα κριτήρια τα οποία υπολογίστηκαν. Επιπλέον, επισημαίνονται παρατηρήσεις για τα πιο αξιοσημείωτες πειραματικές μετρήσεις των αποτελεσμάτων.

Στην Ενότητα 5.4 συγκρίνονται τα αποτελέσματα των δύο μεθόδων, δηλαδή των μεθόδων MRHC1 και MRHC2 με κατηγοριοποιητές MKNN1 και MKNN2 και με την απλή κατηγοριοποίηση με χρήση κατηγοριοποιητή KNN σε συνδυασμό με την τεχνική Binary Relevance, σημειώνοντας παράλληλα τα πλεονεκτήματα και τα μειονεκτήματά τους.

Στην ενότητα 5.5 κλείνει το παρόν κεφάλαιο με έναν γενικό σχολιασμό της πειραματικής διαδικασίας που ακολουθήθηκε και των αποτελεσμάτων της.

5.1 Πειραματική Διαμόρφωση

Τα σύνολα δεδομένων που χρησιμοποιήθηκαν αποτελούνταν μόνο απο αριθμητικά δεδομένα (χαρακτηριστικά) και οι ετικέτες, ήταν δυαδικές (Binary), δηλαδη το ”1” σημαίνει ύπαρξη ετικέτας, ενω το ”0” μη ύπαρξη ετικέτας. Επειδή όμως τα χαρακτηριστικά είχαν τιμές διαφορετικού εύρους αυτό θα είχε σαν αποτέλεσμα κατα την διαδικασία της κατηγοριοποίησης, το κάθε ένα απο αυτά να έχει διαφορετική βαρύτητα. Γι’ αυτό το λόγο πριν την χρήση τους, επιλέχθηκε να κανονικοποιηθούν. Αυτό σημαίνει οτι όλα τα χαρακτηριστικά απόκτησαν εύρος τιμών ανάμεσα στο διάστημα [0,1].

Η συγκεκριμένη διαδικασία έγινε με την συγγραφή και εκτέλεση συγκεκριμένου προγράμματος σε γλώσσα C για όλα τα σύνολα δεδομένων. Έτσι δόθηκε σε όλα τα χαρακτηριστικά καθε συνόλου δεδομένων το ίδιο βάρος χωρίς να αλλοιωθεί η σημασία των τιμών τους.

Στην συνέχεια όλα τα σύνολα δεδομένων χωρίστηκαν σε σύνολα εκπαίδευσης και σύνολα ελέγχου με χρήση της μεθόδου 5-fold-cross-validation.

Η μέθοδος της διαστάυρωσης ή Cross-Validation αποτελεί μια στατιστική τεχνική η οποία προσπαθεί να επιτύχει όσο τον δυνατόν πιο ασφαλή εκτίμηση της δυνατότητας γενίκευσης ενός μοντέλου, δηλαδή να προβλεφθεί η επίδοσή του με βάση κάποιο κριτήριο πάνω σε άγνωστα δεδομένα τα οποία δεν έχει χρησιμοποιήσει κατά την εκπαίδευσή του. Ως κριτήριο μπορεί να χρησιμοποιηθεί οποιοδήποτε είναι κατάλληλο για το συγκεκριμένο πρόβλημα όπως, για παράδειγμα, η ακρίβεια (accuracy) για προβλήματα κατηγοριοποίησης ή το μέσο τετραγωνικό σφάλμα (MSE) για προβλήματα παλινδρόμησης. Στην δικιά μας περίπτωση χρησιμοποιήθηκε η μετρική απώλειας Hamming Loss που αναλύθηκε στο Κεφάλαιο 2.

Η βασική φιλοσοφία της μεθόδου είναι ο τυχαίος διαχωρισμός των δεδομένων σε δύο τμήματα, το train set (σύνολο εκπαίδευσης) και το test ή validation set (το σύνολο ελέγχου). Το μοντέλο εκπαιδεύεται πάνω στο σύνολο εκπαίδευσης και η επίδοσή του συγκρίνεται με ένα σύνολο ελέγχου. Ένα τέτοιο πείραμα καλείται fold. Επειδή ένα μόνο fold μπορεί να εξάγει σχετικά μη ασφαλή αποτελέσματα, συνήθως εκτελούνται K folds και υπολογίζεται η μέση τιμή της επίδοσης για να εκτιμηθεί με ασφαλέστερο τρόπο η αναμενόμενη επίδοση του μοντέλου. Το K μπορεί να λάβει οποιαδήποτε θετική ακέραια τιμή, πχ. 5, 10, 100 κλπ. Όσο μεγαλύτερη είναι η τιμή του K τόσο πιο ασφαλής μπορεί να θεωρηθεί η εκτίμηση που πραγματοποιείται. Υπάρχουν διάφορες στατιστικές αναλύσεις στις οποίες μπορεί να χρησιμοποιηθεί η μέθοδος Cross-Validation ανάλογα με το ζητούμενο αποτέλεσμα. Μια σχηματική παράσταση των όσο αναφέρθηκαν παρουσιάζεται στο σχήμα 5.1. Όπως αναφέρθηκε και παραπάνω αφου τα σύνολα χωρίστηκαν με την μεθοδο της διασταύρωσης στην συνέχεια για κάθε ένα ζευγάρι (train-test) υπολογίστηκε η μετρική απώλειας Hamming Loss και το τελικό αποτέλεσμα των πειραμάτων ήταν ο μέσος όρος αυτών για κάθε σύνολο δεδομένων.

Για την διεξαγωγή των πειραμάτων χρησιμοποιήθηκε ένας **υπολογιστής με επεξεργαστή AMD A10-7700 3.4 GHZ και μνήμη RAM 8GB** και το περιβάλλον **Anacoda Navigator 1.9.12-Spider 4.1.4** στο οποίο σε γλώσσα **pyhton 3.8.3** προγραμματίστηκαν οι αλγόριθμοι (MKNN1,

Σχήμα 5.1: Επικύρωση πέντε (5) folds

MKNN2, Binary Relevance). Ο προγραμματισμός των αλγορίθμων (MKNN1 και MKNN2) υλοποιήθηκε με χρήση νημάτων (threads)-παράλληλος προγραμματισμός. Το Multi- threading επιτρέπει να εκτελεστούν παράλληλα πολλές διαφορετικές διεργασίες με την προυπόθεση ότι υπάρχουν διαθέσιμοι πολλοί επεξεργαστές. Αυτό έχει σαν αποτέλεσμα να επιταγχύνεται η εκτέλεση ενός προγράμματος. Τα νήματα κάνουν κοινή χρήση της μνήμης και των πόρων του υπολογιστικού συστήματος. Στην περίπτωση μας επειδή όπως προαναφέρθηκε για καθε σύνολο δεδομένων ο αλγόριθμος εκτελούνταν 5 φορές (5 folds) επιλέχθηκε η συγκεκριμένη μέθοδος για λόγους ευκολίας και ταχύτητας.

Για την υλοποίηση των προγραμμάτων χρησιμοποιήθηκαν βιβλιοθήκες και απο το **scikit-learn** [57]. Μερικές απο αυτές είναι:

- η ”sklearn.metrics import hamming_loss”, για τον υπολογισμό της μετρικής απώλειας Hamming Loss.
- ”from skmultilearn.problem_transform import binaryrelevance”, για την υλοποίηση του αλγορίθμου Δυαδικής Συνάφειας (BR-Binary Relevance).
- ”from sklearn.neighbors import KNeighborsClassifier”, για την υλοποίηση του αλγορίθμου των Κ εγγύτερων γειτόνων.

  Καθώς και κάποιες βιβλιοθήκες της Python όπως:

- η ”matplotlib.pyplot” που χρησιμοποιήθηκε για την οπτικοποίηση των αποτελεσμάτων Hamming Loss σε μορφή ραβδογράμματος.
- η ”time” χρησιμοποιήθηκε γαι την μέτρηση του χρόνου εκτέλεσης των αλγορίθμων κατηγοριοποίησης.
- η ”pandas” για το χειρισμό και ανάλυση δεδομένων. Ειδικότερα, προσφέρει δομές δεδομένων και λειτουργίες για το χειρισμό αριθμητικών πινάκων και χρονοσειρών.
- η ”math” μας παρέχει πρόσβαση σε ορισμένες κοινές μαθηματικές λειτουργίες και σταθερές στην Python, οι οποίες χρησιμοποιήθηκαν σε όλο τον κώδικά μας για πιο πολύπλοκους μαθηματικούς υπολογισμούς.

- η ”threading” για την εκτέλεση των προγραμμάτων με χρήση Νημάτων.
- η ”unicodecsv” για την ανάγνωση των συνόλων δεδομένων απο αρχείο csv.
- η ”numpy” για την χρήση πινάκων στο πρόγραμμα μας.

Επίσης πρέπει να αναφερθεί οτι επειδή εφαρμόστηκε η επικύρωση 5-folds, τόσο ο αλγόριθμος 5, όσο και ο αλγόριθμος 6 εκτελούνται αντίστοιχα, πέντε φορές ο καθένας, σε πέντε παράλληλα νήματα προγραμματισμού που αποτελούνται απο τα ζεύγη του (συνόλου εκπαίδευσης-συνόλου ελέγχου). Σε κάθε εκτέλεση υπολογίζεται η απώλεια Hamming Loss και στο τέλος η συνολική απώλεια Hamming Loss του αλγορίθμου ισούται με τον μέσο όρο των πέντε Hamming Loss που προέκυψαν απο τις πέντε εκτελέσεις του αλγορίθμου. Επίσης οι μέθοδοι παραγωγής προτύπων MRHC1 και MRHC2 αναπτύχθηκαν σε γλώσσα προγραμματισμού C++ κάνοντας χρήση του περιβάλλον CodeBlocks 20.03, το οποίο είναι ένα πληρες ολοκληρωμένο περιβάλλον ανάπτυξης (Integrated Development Environment), για συγγραφή και μετάφραση κώδικα. Ο λόγος που επιλέχτηκε η C++ για τους MRHC1 και MRHC2 είναι επειδή η ανάπτυξή τους, στηρίχθηκε σε μια ήδη υπάρχουσα υλοποίηση του αλγορίθμου RHC για την υλοποίηση των δυο νέων παραλλαγών.

5.2 Περιγραφή Συνόλων Δεδομένων

Τα σύνολα δεδομένων πολλαπλών ετικετών που χρειάζονταν για τα πειράματα έπρεπε να πληρούν ορισμένα κριτήρια. Το πρώτο και πιο σημαντικό από αυτά, είναι οτι θα πρέπει να είναι δυαδικά (*Binary*) και να έχουν πολλαπλές κλάσεις (*multi-label*). Εκτός αυτού, θα πρέπει να είναι εξειδικευμένα σύνολα δεδομένων τα οποία προορίζονται για θέματα κατηγοριοποίησης πολλαπλών ετικετών.

Για να εκτελέσουμε τις Τεχνικές Μείωσης Δεδομένων καθώς και τους κατηγοριοποιητές εγγύτερων γειτόνων χρησιμοποιώντας την Ευκλείδεια απόσταση, θα πρέπει όλα τα χαρακτηριστικά να είναι αριθμητικά. Αυτό, βοηθάει και στο να αξιολογηθούν τα αποτελέσματα μεταξύ των διαφορετικών συνόλων δεδομένων.

Τέλος, θεωρήθηκε ότι για να έχουν νόημα τα πειράματα μας θα ήταν προτιμότερο να εφαρμοστούν πάνω σε σύνολα δεδομένων που περιέχουν τουλάχιστον πεντακόσια (500) στιγμιότυπα. Τα σύνολα δεδομένων που τελικά χρησιμοποιήθηκαν φαίνονται παρακάτω. Ακολουθεί μία σύντομη περιγραφή των συνόλων δεδομένων που χρησιμοποιήθηκαν στα πειράματα. Συνολικά χρησιμοποιήθηκαν εννέα σύνολα δεδομένων πολλαπλών ετικετών. Όλα τα σύνολα δεδομένων είναι διαθέσιμα στο Mulan [31] καθώς επίσης και στην ιστοσελίδα [58].

5.2.1 Σύνολο Δεδομένων CAL500

Το σύνολο δεδομένων CAL500 (Computer Audition Lab) προέρχεται από μια μελέτη περίπτωσης στην οποία ζητήθηκε από τους συμμετέχοντες να σχολιάσουν 502 δημοφιλή τραγούδια του

Δυτικού κόσμου. Με βάση τα αποτελέσματα της έρευνας προέκυψε ένα λεξιλόγιο 174 ετικετών (Labels) σχετικά με σημασιολογικές έννοιες, όπως συναισθηματικό περιεχόμενο, είδος, όργανα, φωνητικά χαρακτηριστικά, ακουστική ποιότητα του τραγουδιού και όροι χρήσης. Κάθε ένα από τα τραγούδια σχολιάστηκε από τουλάχιστον τρία άτομα. Για να δημιουργηθεί το σύνολο δεδομένων πληρώσανε 66 προπτυχιακούς φοιτητές να σχολιάσουν τα τραγούδια με βάση τις παραπάνω σημασιολογικές έννοιες. Κάθε ένας από αυτούς ανταμείφθηκε με 10$ ανά ώρα για να ακούσουν και να σχολιάσουν την μουσική σε έναν υπολογιστή στο εργαστήριο του πανεπιστημίου. Η διαδικασία σχολιασμού των τραγουδιών διήρκησε πέντε λεπτά και οι περισσότεροι συμμετέχοντες ανέφεραν ότι η ακρόαση και η εμπειρία ήταν ευχάριστη.

Για κάθε τραγούδι συλλέχθηκε μια συλλογή από ανθρώπινα σχόλια, όπου κάθε σχόλιο είναι ένα διάνυσμα αριθμών που εκφράζουν την απάντηση ενός θέματος. Κάθε σχόλιο αναπαρίσταται με τους αριθμούς 1 ή -1, αν συμφωνεί ή όχι αντίστοιχα και 0 αν δεν είναι σίγουρος. Στην συνέχεια πήρανε όλους αυτούς τους ενδείκτες για κάθε τραγούδι και τους συμπιέσανε σε ένα διάνυσμα, παρατηρώντας το επίπεδο συμφωνίας για όλους τους ενδείκτες. Η σημασιολογική βαρύτητα είναι:

$$[y]_i = max\left(0, [\frac{\#(PositiveVotes) - \#(NegativeVotes)}{\#(Annotations)}]_i\right) \tag{5.1}$$

Για παράδειγμα αν για ένα τραγούδι έχουμε τέσσερις ενδείκτες, με τιμές +1,+1,0,-1,τότε το $[y]_i = \frac{1}{4}$. Τα σημασιολογικά βάρη χρησιμοποιούνται για εκτίμηση παραμέτρων [59]. Πιο αναλυτικά τα χαρακτηριστικά του συνόλου δεδομένων είναι σύμφωνα με τον πίνακα 5.1

Πίνακας 5.1: Στοιχεία Συνόλου Δεδομένων CAL500

Τομέας	Στιγμιότυπα	Γνωρίσματα	Ετικέτες	Πληθικότητα (Cardinality)	Πυκνότητα (Density)	Διακριτότητα (Distict)
Μουσική	502	68	174	26.044	0.150	502

5.2.2 Σύνολο Δεδομένων Emotions

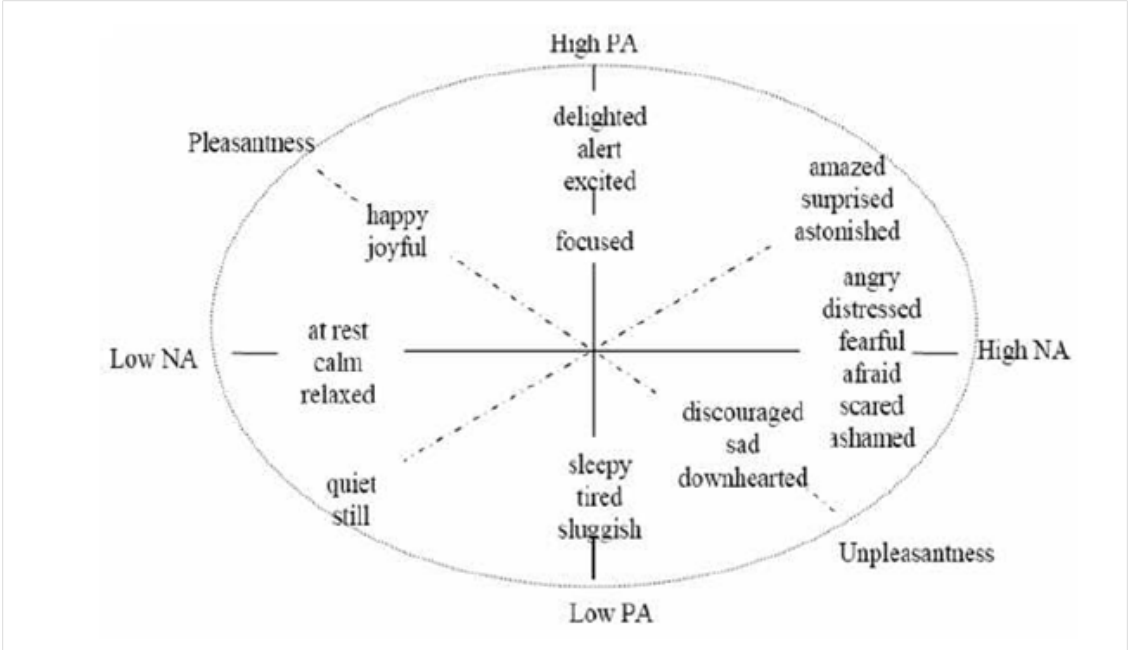
Το σύνολο δεδομένων αποτελείται από 100 τραγούδια από επτά διαφορετικά είδη μουσικής: κλασική, ρέγκε, χιπ-χοπ, techno και τζαζ. Η συλλογή δημιουργήθηκε από 233 μουσικά άλμπουμ, επιλέγοντας 3 τραγούδια από κάθε άλμπουμ. Από κάθε τραγούδι έγινε εξαγωγή μιας περιόδου 30 δευτερολέπτων αφού πρώτα εξαιρέθηκαν τα πρώτα 30 δευτερόλεπτα.
Τα αποσπάσματα ήχου που πρόεκυψαν αποθηκεύτηκαν και μετατράπηκαν σε αρχεία με ρυθμό δειγματοληψίας 22.050HZ,16 bit ανά δείγμα και μονοφωνικά. Στην συνέχεια παρουσιάζονται τα χαρακτηριστικά του αρχείο ήχου και η διαδικασία επισήμανσης συναισθημάτων. Για τη διαδικασία εξαγωγής χαρακτηριστικών χρησιμοποιήθηκε το εργαλείο Marsyas [60]. Τα εξαγόμενα χαρακτηριστικά εμπίπτουν σε δύο κατηγορίες: τα ρυθμικά και τα ηχοχρώματος.

Ρυθμικά χαρακτηριστικά Τα ρυθμικά χαρακτηριστικά προέκυψαν με την εξαγωγή περιοδικών αλλαγών από ένα ιστόγραμμα ρυθμού. Υλοποιήθηκε ένας αλγόριθμος που προσδιορίζει τα ανώτατα όρια της μουσικής (peaks) χρησιμοποιώντας την αυτόματη σχέση. Επιλέξανε τα δύο υψηλότερα ανώτατα όρια και υπολογίσανε τα πλάτη τους, κτύποι ανά λεπτό- BMPS (beats per

minutes) και τον (υψηλό -χαμηλό) λόγο των κτύπων ανά λεπτό (BPMS). Επιπλέον, υπολογίστηκαν 3 χαρακτηριστικά αθροίζοντας τα ιστογράμματα μεταξύ 40-90, 90-140 και 140-250 BPMs αντίστοιχα. Η όλη διαδικασία οδήγησε σε ένα σύνολο 8 ρυθμικών χαρακτηριστικών.

Χαρακτηριστικά ηχοχρώματος Οι συντελεστές cepstral [61] συχνότητας Mel (MFCCs) χρησιμοποιούνται για αναγνώριση ομιλίας και μοντελοποίηση μουσικής. Για την εξαγωγή των MFCCs χαρακτηριστικών, το σήμα χωρίστηκε σε πλαίσια, όπου, το φάσμα εύρους υπολογίστηκε για κάθε πλαίσιο. Στη συνέχεια, υπολογίστηκε ο αλγόριθμος του και μετατράπηκε σε κλίμακα Mel. Επιλέξανε τα πρώτα 13 MFCCs. Ένα άλλο σύνολο τριών χαρακτηριστικών γνωρισμάτων, που σχετίζονται με τις υφές τόνου, εξήχθη με χρήση του μετασχηματισμού Fourier (FFT): κέντρο φάσματος, φασματικής ανάπτυξης και φασματικής ροής. Για καθένα από τα 16 προαναφερθέντα χαρακτηριστικά (13 MFCCs, 3 FFT) υπολογίσανε τη μέση (mean), τυπική απόκλιση-standard deviation (std), τη μέση τυπική απόκλιση-mean standard deviation (mean std) και την τυπική απόκλιση της τυπικής απόκλισης- standard deviation of standard deviation (std std) σε όλα τα πλαίσια. Αυτό οδήγησε σε συνολικά 64 χαρακτηριστικά ηχοχρώματος.

Κατάταξη Συναισθημάτων Το μοντέλο Tellegen-Watson-Clark [62] χρησιμοποιήθηκε για την κατάταξη των δεδομένων με βάση τα συναισθήματα. Αποφάσισαν να χρησιμοποιήσουν αυτό το συγκεκριμένο μοντέλο, επειδή ο συναισθηματικός χώρος της μουσικής είναι αφηρημένος, με πολλά συναισθήματα και μια εφαρμογή μουσικής, που βασίζεται στην διάθεση του ακροατή, θα πρέπει να συνδυάζεται με μια σειρά από διαθέσεις και συναισθήματα. Για την επίτευξη αυτού του στόχου χωρίς τη χρήση υπερβολικού αριθμού ετικετών, κατέληξαν να κρατήσουν μόνο έξι κύριες συναισθηματικές καταστάσεις από αυτό το μοντέλο που φαίνεται στο σχήμα 5.2 [63].



Σχήμα 5.2: Συναισθηματικές καταστάσεις συνόλου ”Emotions”

Πίνακας 5.2: Περιγραφή ετικετών συνόλου Emotions

Class (ετικέτα)	Περιγραφή	Αριθμός στιγμιότυπων
L1 Amazed	Amazed-surprised	173
L2 Happy	Happy-pleased	166
L3 Relaxing	Relaxing-calm	264
L4 Quiet	Quiet-still	148
L5 Sad	Sad-lonely	168
L6 Angry	Angry-fearful	189

Πίνακας 5.3: Στοιχεία Συνόλου Δεδομένων Emotions

Τομέας	Στιγμιότυπα	Γνωρίσματα	Ετικέτες	Πληθικότητα (Cardinality)	Πυκνότητα (Density)	Διακριτότητα (Distict)
Μουσική	593	72	6	1.869	0.311	27

Ενώ οι αντίστοιχες ετικέτες του συνόλου παρουσιάζονται στον πίνακα 5.2.
Πιο αναλυτικά τα χαρακτηριστικά του συνόλου δεδομένων είναι σύμφωνα με τον πίνακα 5.3.

5.2.3 Σύνολο δεδομένων Water quality

Τα δεδομένα προέρχονται από το υδρο-μετεωρολογικό ινστιτούτο της Σλοβενίας που παρακολουθεί την ποιότητα του νερού των ποταμών της Σλοβενίας και διατηρεί μια βάση δεδομένων για την ποιότητα των υδάτων. Τα στοιχεία καλύπτουν περίοδο έξι ετών (1990-1995). Τα βιολογικά δείγματα λαμβάνονται δύο φορές το χρόνο, μία φορά το καλοκαίρι και μία το χειμώνα, βιολογικά δείγματα λαμβάνονται και ενδιάμεσα (περίοδοι μεταξύ μετρήσεων που ποικίλλουν από ένα έως αρκετούς μήνες) για κάθε τόπο δειγματοληψίας.

Τα φυσικά και χημικά δείγματα περιλαμβάνουν μετρήσεις 16 διαφορετικών παραμέτρων: βιολογική ζήτηση οξυγόνου (BOD), ηλεκτρική αγωγιμότητα, χημική ζήτηση οξυγόνου (K2Cr2O7 και KMnO4), συγκεντρώσεις από Cl, CO2, NH4, PO4, SiO2, NO2, NO3 και διαλυμένο οξυγόνο (O2), αλκαλικότητα (pH), κορεσμός οξυγόνου, θερμοκρασία του νερού και της ολικής σκληρότητας.

Τα βιολογικά δείγματα περιλαμβάνουν μια λίστα από ταξινομικές βαθμίδες που υπάρχουν στον τόπο δειγματοληψίας καθώς και την πυκνότητά τους. Η συχνότητα εμφάνισης (πυκνότητα) κάθε ταξινομικής βαθμίδας καταγράφεται από εμπειρογνώμονα βιολόγο σε τρία διαφορετικά ποιοτικά επίπεδα: 1=δευτερεύοντος, 3=συχνά και 5=απόλυτα. Πιο συγκεκριμένα τα χημικά δείγματα περιλαμβάνουν τα χαρακτηριστικά όπως (Τοποθεσία, Έτος, Μήνας, Ημέρα, λίστα από 16 τιμές) .Περιέχουν φυσικοχημικές μετρήσεις. Αποτελούνται από 2580 γεγονότα.

Ενώ τα βιολογικά δείγματα περιέχουν (Τοποθεσία, Ημέρα, Μήνας, Έτος, λίστα από ταξινομικές βαθμίδες): Αποτελούνται από 1106 γεγονότα.

Συνολικά το σύνολο δεδομένων είναι αρκετά καθαρό, αλλά όχι τέλειο. Δεκατέσσερις φυσικο-

χημικές μετρήσεις έχουν ελλείπουσες τιμές. Επιπλέον, αν και οι βιολογικές μετρήσεις συνήθως λαμβάνονται ακριβώς την ίδια ημέρα με κάποια φυσικοχημική μέτρηση, για 43 βιολογικές μετρήσεις, δεν υπάρχουν φυσικοχημικά δεδομένα για την ίδια ημέρα. Λόγω των λίγων ελλιπών τιμών που περιέχει το σύνολο δεδομένων, αγνοήθηκαν τα στιγμιότυπα με τις ελλείπουσες τιμές στα πειράματά. Έτσι έμειναν 1060 δείγματα νερού τα οποία έχουν πλήρεις βιολογικές και φυσικοχημικές πληροφορίες διαθέσιμες [64]. Πιο αναλυτικά τα χαρακτηριστικά του συνόλου Δεδομένων είναι σύμφωνα με τον πίνακα 5.4.

Πίνακας 5.4: Στοιχεία Συνόλου Δεδομένων Water quality

Τομέας	Στιγμιότυπα	Γνωρίσματα	Ετικέτες	Πληθικότητα (Cardinality)	Πυκνότητα (Density)	Ποικιλία (Diversity)
Χημεία	1060	16	14	5.073	0.362	0.778

5.2.4 Σύνολο δεδομένων Scene

Πρόκειται για ένα σύνολο δεδομένων εικόνων, το οποίο περιέχει 2407 εικόνες, όπου κάθε εικόνα ανήκει σε 6 κατηγορίες: παραλία, ηλιοβασίλεμα, φύλλωμα πτώσης, πεδιάδα, βουνό και αστικό τοπίο. Κάθε εικόνα περιγράφεται με 294 οπτικά αριθμητικά χαρακτηριστικά που αναπαριστώνται με χωρικές χρωματικές ροπές στο χώρο LUV [65]. Κάθε εικόνα χωρίζεται σε 49 μπλοκ χρησιμοποιώντας ένα πλέγμα επτά γραμμών και επτά στηλών, έτσι τα χαρακτηριστικά κάθε εικόνας είναι $2 \times 3 \times 7 \times 7 = 294$ [66]. Πιο αναλυτικά τα χαρακτηριστικά του συνόλου δεδομένων είναι σύμφωνα με τον πίνακα 5.5.

Πίνακας 5.5: Στοιχεία Συνόλου Δεδομένων Scene

Τομέας	Στιγμιότυπα	Γνωρίσματα	Ετικέτες	Πληθικότητα (Cardinality)	Πυκνότητα (Density)	Διακριτότητα (Distict)
Εικόνα	2407	294	6	1.074	0.179	15

5.2.5 Σύνολο δεδομένων Yeast

Το σύνολο δεδομένων μαγιάς περιέχει εκφράσεις μικροσυστοιχίων και φύλο-γενετικά προφίλ για 2417 γονίδια μαγιάς. Καθένα από τα οποία σχετίζεται με ένα σύνολο 14 λειτουργικών κατηγοριών (π.χ. Μεταβολισμός, ενέργεια κ.λπ.) γονιδίων από την ολοκληρωμένη βάση δεδομένων γονίδιων μαγιάς του Κέντρου Πληροφοριών του Μονάχου για πρωτεϊνικές ακολουθίες. Κάθε γονίδιο εκφράζεται με 103 αριθμητικά χαρακτηριστικά. Τα δεδομένα παράγονται με βάση τα δεδομένα πρωτεΐνης, και μπορούν να χρησιμοποιηθούν για την αξιολόγηση της απόδοσης ανίχνευση δομικής πρωτεϊνικής λειτουργίας [66]. Πιο αναλυτικά τα χαρακτηριστικά του συνόλου δεδομένων είναι σύμφωνα με τον πίνακα 5.6.

Πίνακας 5.6: Στοιχεία Συνόλου Δεδομένων Yeast

Τομέας	Στιγμιότυπα	Γνωρίσματα	Ετικέτες	Πληθικότητα (Cardinality)	Πυκνότητα (Density)	Διακριτότητα (Distict)
Βιολογία	2417	103	14	4.237	0.303	198

Πίνακας 5.7: Είδη πουλιών Συνόλου Δεδομένων Birds

Κωδικός	Όνομα
BRCR	Brown Creeper
PAWR	Pacific Wren
PSFL	Pacific-slope Flycatcher
RBNU	Red-breasted Nuthatch
DEJU	Dark-eyed Junco
OSFL	Olive-sided Flycatcher
HETH	Hermit Thrush
CBCH	Chestnut-backed Chickadee
VATH	Varied Thrush
HEWA	Hermit Warbler
SWTH	Swainson's Thrush
HAFL	Hammond's Flycatcher
WETA	Western Tanager
BHGB	Black-headed Grosbeak
GCKI	Golden Crowned Kinglet
WAVI	Warbling Vireo
MGWA	MacGillivray's Warbler
STJA	Stellar's Jay
CONI	Common Nighthawk

5.2.6 Σύνολο δεδομένων Birds

Τα πτηνά έχουν χρησιμοποιηθεί ευρέως ως βιολογικοί δείκτες για έρευνα. Ανταποκρίνονται γρήγορα στις περιβαλλοντικές αλλαγές και μπορούν να χρησιμοποιηθούν για να εξάγουν συμπεράσματα σχετικά με άλλους οργανισμούς (π.χ. έντομα από τα οποία τρέφονται). Παραδοσιακές μέθοδοι συλλογής δεδομένων σχετικά με τα πτηνά συνεπάγεται δαπανηρή ανθρώπινη προσπάθεια. Μια πολλά υποσχόμενη εναλλακτική λύση είναι η ακουστική παρακολούθηση.

Υπάρχουν πολλά πλεονεκτήματα στην καταγραφή του ήχου των πτηνών σε σύγκριση με τις ανθρώπινες έρευνες, αυξημένη χρονική και χωρική ανάλυση και έκταση, δυνατότητα εφαρμογής σε απομακρυσμένες τοποθεσίες, μειωμένη προκατάληψη παρατηρητή και ενδεχομένως, χαμηλότερο κόστος.

Το συγκεκριμένο σύνολο δεδομένων ήχου συλλέχθηκε στο H. J. Andrews (HJA) Μακροπρόθεσμο Πειραματικό Ερευνητικό Δάσος, στην οροσειρά Cascade του Όρεγκον. Από το 2009, τα μέλη της ομάδας του Όρεγκον State University Bioacoustics έχουν συλλέξει πάνω από 10 TB δεδομένα ήχου στο HJA χρησιμοποιώντας συσκευές μέτρησης και καταγραφής ήχου (Songmeter). Το πλήρες σύνολο δεδομένων αποτελείται από 645 εγγραφές ήχου, δέκα δευτερολέπτων σε μη συμπιεσμένη μορφή WAV (συχνότητα δειγματοληψίας 16kHz, 16 bits ανά δείγμα, μονοφωνικό) [67]. Υπάρχουν 19 είδη πουλιών σε αυτό το σύνολο δεδομένων (πίνακας 5.7).

Πίνακας 5.8: Στοιχεία Συνόλου Δεδομένων Birds

Τομέας	Στιγμιότυπα	Γνωρίσματα	Ετικέτες	Πληθικότητα (Cardinality)	Πυκνότητα (Density)	Διακριτότητα (Distict)
Ήχος	645	260	19	1.014	0.053	133

Το υποσύνολο των 645 εγγραφών έχει ως στόχο να παρέχει μια επαρκή κάλυψη για όλες τις τοποθεσίες καταγραφής, αρκετές ημέρες, σε πολλά έτη, και αρκετές ώρες της ημέρας γύρω από την αυγή, όταν τα πουλιά είναι πιο ενεργά. Κάθε εγγραφή ήχου δέκα δευτερολέπτων συνδυάζεται με ένα σύνολο ειδών που υπάρχουν. Αυτά τα σύνολα ετικετών ελήφθησαν, ακούγοντας τον ήχο και κοιτάζοντας φασματογραφήματα. Αρκετοί εμπειρογνώμονες επιθεώρησαν κάθε καταγραφή, και ο καθένας παρείχε τη δική του σειρά ετικετών, μαζί με ένα συντελεστή εμπιστοσύνη της εκτίμησης. Το τελικό σύνολο ετικετών διαμορφώθηκε από τον συντελεστή εμπιστοσύνης και την σταθμισμένη πλειοψηφία. Τα ονόματα αρχείων WAV κωδικοποιούν τη θέση (μια από τα 13 τοποθεσίες) και την ημερομηνία/ώρα συλλογής της εγγραφής [67]. Πιο αναλυτικά τα χαρακτηριστικά του συνόλου δεδομένων είναι σύμφωνα με τον πίνακα 5.8.

5.2.7 Σύνολο δεδομένων CHD49

Το σύνολο δεδομένων της στεφανιαίας νόσου-Coronary Heart Disease (CHD) στην παραδοσιακή κινεζική ιατρική-Traditional Chinese Medicine (TCM) μετά από προ-επεξεργασία καταλήγει να έχει συνολικά 555 στιγμιότυπα, μεταξύ των οποίων 265 ασθενείς ήταν άνδρες, και 290 ασθενείς είναι γυναίκες. Αποτελείται από 125 συμπτώματα (χαρακτηριστικά) και 15 σύνδρομα για την διακριτική ικανότητα διάγνωσης, εκ των οποίων τα 6 πιο συχνά χρησιμοποιούμενα πρότυπα έχουν επιλεγεί και συμπεριλαμβάνονται στο σύνολο δεδομένων.

Μερικά από αυτά είναι:το z1 Σύνδρομο qi ανεπάρκειας καρδιάς, z2 Σύνδρομο yang ανεπάρκειας καρδιάς, z3 Σύνδρομο yin ανεπάρκειας καρδιάς, z4 Σύνδρομο στασιμότητας Qi, z5 Σύνδρομο φλέγματος Turbid και z6 Σύνδρομο στάσης αίματος. Το αρχικό σύνολο αποτελούνταν από 52 συμπτώματα (χαρακτηριστικά), από αυτά τρία (3) ήταν περιττά χαρακτηριστικά (όπως το οίδημα) και αφαιρέθηκαν από το τελικό σύνολο δεδομένων. Έτσι το τελικό σύνολο αποτελείται από ένα σύνολο δεδομένων με 49 συμπτώματα (χαρακτηριστικά). Ο ελάχιστος αριθμός ετικετών για κάθε στιγμιότυπο είναι ”0” και ο μέγιστος αριθμός ετικετών για κάθε παρουσία είναι πέντε. Ο μέσος όρος αριθμός ετικετών του δείγματος είναι 2,58. Τα χαρακτηριστικά του δείγματος είναι όλα διακριτά [68]. Πιο αναλυτικά τα χαρακτηριστικά του συνόλου δεδομένων είναι σύμφωνα με τον πίνακα 5.9.

Πίνακας 5.9: Στοιχεία Συνόλου Δεδομένων CHD49

Τομέας	Στιγμιότυπα	Γνωρίσματα	Ετικέτες	Πληθικότητα (Cardinality)	Πυκνότητα (Density)	Ποικιλία (Diversity)
Ιατρική	555	49	6	2.580	0.430	0.531

5.2.8 Σύνολο δεδομένων Image

Αυτό το σύνολο δεδομένων αποτελείται από 2.000 εικόνες που ανήκουν σε ετικέτες όπως έρημο, βουνά, θάλασσα, ηλιοβασίλεμα και δέντρα. Πάνω από 22% εικόνες ανήκουν σε πολλαπλές ετικέτες ταυτόχρονα και κάθε εικόνα συσχετίζεται με 1,24 ετικέτες, ετικέτας κατά μέσο όρο. Κάθε εικόνα αντιπροσωπεύεται από ένα διάνυσμα χαρακτηριστικών. Συγκεκριμένα, κάθε έ-γχρωμη εικόνα μετατρέπεται αρχικά στον χώρο CIE Luv, ο οποίος είναι ένας ομοιόμορφος, χρωματικός, αντιληπτός, χώρος έτσι ώστε οι αντιληπτές χρωματικές διαφορές να αντιστοιχούν πολύ σε ευκλείδειες αποστάσεις σε αυτόν τον χρωματικό χώρο. Στη συνέχεια, η εικόνα χωρίζεται σε 49 μπλοκ χρησιμοποιώντας ένα πλέγμα 7×7, όπου σε κάθε μπλοκ υπολογίζονται η πρώτη και η δεύτερη ροπή (μέση και διακύμανση) κάθε ζώνης, που αντιστοιχούν σε μια εικόνα χαμηλής ανάλυσης και σε υπολογιστικά φθηνά χαρακτηριστικά υφής αντίστοιχα. Τέλος, κάθε εικόνα μετατρέπεται σε διανύσματα διαστάσεων $49 \times 3 \times 2 = 294$ διαστάσεων (χαρακτηριστικών-γνωρισμάτων) [69]. Πιο αναλυτικά τα χαρακτηριστικά του συνόλου Δεδομένων είναι σύμφωνα με τον πίνακα 5.10.

Πίνακας 5.10: Στοιχεία Συνόλου Δεδομένων Image

Τομέας	Στιγμιότυπα	Γνωρίσματα	Ετικέτες	Πληθικότητα (Cardinality)	Πυκνότητα (Density)	Ποικιλία (Diversity)
Εικόνα	2000	294	5	1.236	0.247	0.625

5.2.9 Σύνολο δεδομένων Mediamill

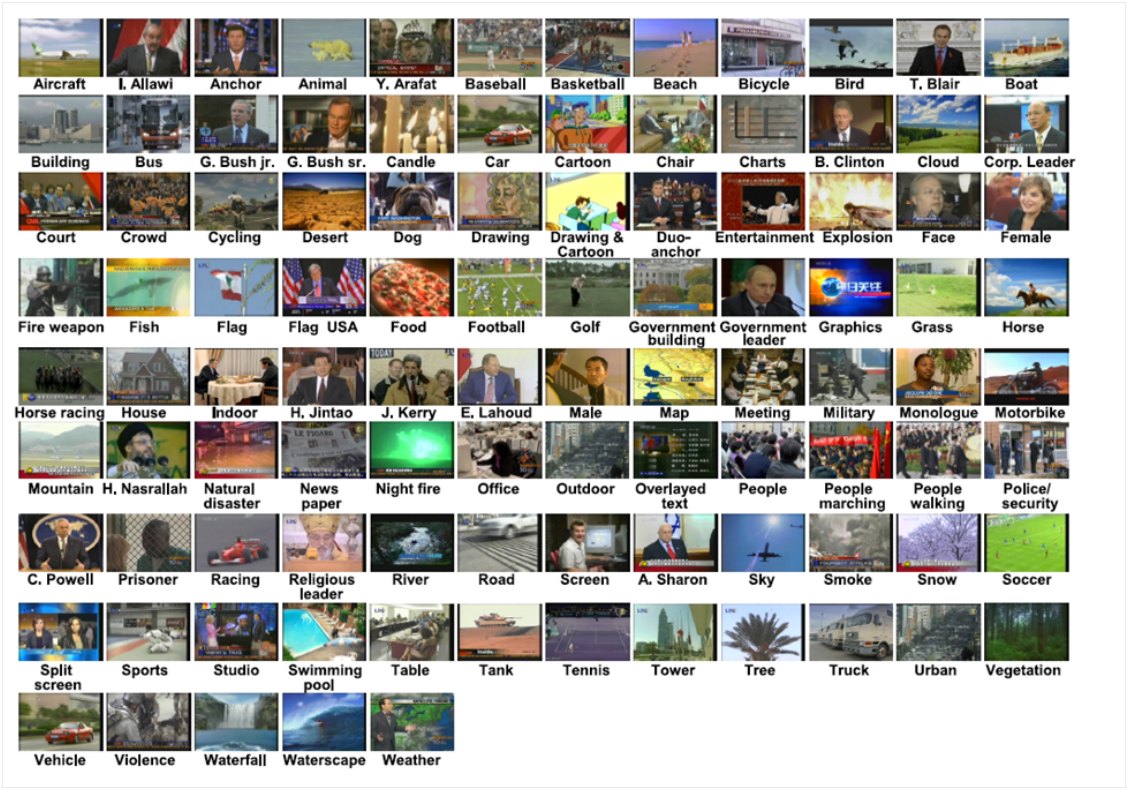
Αυτό το σύνολο δεδομένων περιέχει προ-υπολογισμένα χαρακτηριστικά πολυμέσων, χαμηλού επιπέδου από 85 ώρες διεθνούς βίντεο ειδήσεων μετάδοσης του TRECVID 2005/2006 που καταγράφηκαν σε μορφή MPEG-1 από την κοινοπραξία Γλωσσικών Δεδομένων (Linguistic Data Consortium). Αυτό το σύνολο δεδομένων περιέχει Αραβικές, Κινεζικές και Αμερικανικές εκπομπές ειδήσεων που καταγράφηκαν κατά τη διάρκεια του μήνα, Νοέμβριο του 2004. Το περιεχόμενο των ειδήσεων είναι ταξινομημένο με πολλαπλές ετικέτες. Κάθε βίντεο αντιπροσωπεύεται απο ένα διάνυσμα 120 αριθμητικών δεδομένων που αποτελούν τα χαρακτηριστικά αυτού. Παρακάτω παραθέτουμε το σχήμα 5.3 με τις 101 ετικέτες στις οποίες ταξινομούνται τα στιγμιότυπα ανάλογα με το περιεχόμενο, του βίντεο, που αφορά το καθένα [70].

Πιο αναλυτικά τα χαρακτηριστικά του συνόλου δεδομένων είναι σύμφωνα με τον πίνακα 5.11.

Πίνακας 5.11: Στοιχεία Συνόλου Δεδομένων mediamill

Τομέας	Στιγμιότυπα	Γνωρίσματα	Ετικέτες	Πληθικότητα (Cardinality)	Πυκνότητα (Density)	Διακριτότητα (Distinct)
Βίντεο	43907	120	101	4.376	0.043	6555

Ανακεφαλαιώνοντας τα βασικά χαρακτηριστικά των παραπάνω συνόλων δεδομένων πολλα-πλών ετικετών παρουσιάζονται, συνοπτικά στον πίνακα 5.12.



Σχήμα 5.3: Οπτική απεικόνιση των 101 σημασιολογικών ετικετών του συνόλου Mediamill

Πίνακας 5.12: Συνοπτικά τα βασικά στοιχεία όλων των συνόλων δεδομένων

Όνομα	Τομέας	Στιγμιότυπα	Γνωρίσματα	Ετικέτες
CAL500	Μουσική	502	68	174
Emotions	Μουσική	593	72	6
Water quality	Χημεία	1060	16	14
Scene	Εικόνα	2407	294	6
Yeast	Βιολογία	2417	103	14
Birds	Ήχος	645	260	19
CHD49	Ιατρική	555	49	6
Image	Εικόνα	2000	294	5
Mediamill	Βίντεο	43907	120	101

5.3 Αποτελέσματα Μετρήσεων μετά την εκτέλεση των πειραμάτων

Όπως αναφέρθηκε και παραπάνω για κάθε σύνολο δεδομένων πολλαπλών ετικετών εκτελέστηκαν δεκαπέντε (15) πειράματα, δηλαδη 15 διαφορετικές περιπτώσεις. Πιο συγκεκριμένα για τον κατηγοριοποιητή MKNN1 ξεχωρίζουν οι παρακάτω περιπτώσεις:

1. Χωρίς την χρήση του αλγόριθμου παραγωγής προτύπων (MRHC1) για K=N+1, όπου N το πλήθος των ετικετών των συνόλων δεδομένων.

2. Με χρήση του αλγόριθμου παραγωγής προτύπων (MRHC1) για K=N+1, όπου N το πλήθος των ετικετών των συνόλων δεδομένων.

Ενώ για τον κατηγοριοποιητή MKNN2 ξεχωρίζουν οι παρακάτω περιπτώσεις:

1. Χωρίς την χρήση του αλγόριθμου παραγωγής προτύπων (MRHC1) για K=N, όπου N το πλήθος των ετικετών των συνόλων δεδομένων.
2. Χωρίς την χρήση του αλγόριθμου παραγωγής προτύπων (MRHC1) για K=1.
3. Χωρίς την χρήση του αλγόριθμου παραγωγής προτύπων (MRHC1) για K=3.
4. Χωρίς την χρήση του αλγόριθμου παραγωγής προτύπων (MRHC1) για K=5.
5. Με χρήση του αλγόριθμου παραγωγής προτύπων (MRHC1).

Επίσης εκτελέστηκαν πειράματα με χρήση του BR (Binary Relevance), με τον οποίο και έγινε σύγκριση ξεχωρίζοντας τις παρακάτω περιπτώσεις:

1. Χωρίς την χρήση του αλγόριθμου παραγωγής προτύπων (MRHC1 ή MRHC2) για K=1.
2. Χωρίς την χρήση του αλγόριθμου παραγωγής προτύπων (MRHC1 ή MRHC2) για K=3.
3. Χωρίς την χρήση του αλγόριθμου παραγωγής προτύπων (MRHC1 ή MRHC2) για K=5.
4. Χωρίς την χρήση του αλγόριθμου παραγωγής προτύπων (MRHC1 ή MRHC2) για K=9.
5. Με χρήση του αλγόριθμου παραγωγής προτύπων (MRHC2) για K=1.
6. Με χρήση του αλγόριθμου παραγωγής προτύπων (MRHC2) για K=3.
7. Με χρήση του αλγόριθμου παραγωγής προτύπων (MRHC2) για K=5.
8. Με χρήση του αλγόριθμου παραγωγής προτύπων (MRHC2) για K=9.

Συνοψίζοντας τα αποτελέσματα των δεκαπέντε (15) πειραμάτων για κάθε σύνολο δεδομένων, προέκυψαν τα παρακάτω αποτελέσματα, σε μορφή πινάκων που παραθέτονται για κάθε περίπτωση συνολικά.

Αναλυτικότερα για το σύνολο δεδομένων **CAL500** προκύπτει ο πίνακας 5.13. Σε αυτό τον πίνακα φαίνεται οτι οι αλγόριθμοι παραγωγής προτύπων (MRHC1 και MRHC2) καταφέρνουν να επιτύχουν ποσοστό μείωσης δεδομένων που φτάνει το 40.85%, το μέτρο απώλειας Hamming Loss για MKNN1 και MKNN2, σε σχέση με τον κλασικό BR, παραμένει το ίδιο, όταν συνδυάζεται με συμπίεση δεδομένων.

Ο MRHC2 όταν εφαρμόζεται στον κλασικό BR έχει το ίδιο και καλύτερο Hamming Loss και σε κάποιες τιμές του (K) γίνεται ακόμη καλύτερο, ενω ο χρόνος εκτέλεσης του αλγορίθμου μειώνεται περίπου στο μισό με βάση, τον αρχικό χρόνο, χωρίς συμπίεση δεδομένων.

Για το σύνολο δεδομένων **Water-Quality** προκύπτει ο πίνακας 5.14. Σε αυτό τον πίνακα φαίνεται οτι οι αλγόριθμοι παραγωγής προτύπων (MRHC1 και MRHC2) καταφέρνουν να επιτύχουν

Πίνακας 5.13: Αποτελέσματα πειραμάτων στο σύνολο CAL500

Αλγόριθμος-Μέθοδος	Hamming Loss	Reduction Rate (%)	Distance Computations	Cpu Time (minutes)
MKNN1	0.2			0.29
MKNN2	0.8			0.31
MKNN1-MRHC1	0.14	40.85	712827.8	0.17
MKNN2-MRHC1	0.14	40.85	712827.8	0.19
MKNN2 (K=1)	0.19			0.29
MKNN2(K=3)	0.28			0.28
MKNN2(K=5)	0.34			0.28
BR(K=1)	0.19			0.24
BR(K=3)	0.16			0.25
BR(K=5)	0.15			0.26
BR(K=9)	0.14			0.25
BR(K=1)-MRHC2	0.17	40.549	694295.8	0.18
BR(K=3)-MRHC2	0.15	40.549	694295.8	0.18
BR(K=5)-MRHC2	0.14	40.549	694295.8	0.18
BR(K=9)-MRHC2	0.14	40.549	694295.8	0.18

ποσοστό μείωσης δεδομένων που φτάνει το 40.65%, το μέτρο απώλειας Hamming Loss για MKNN1 και MKNN2, σε σχέση με τον κλασικό BR, παραμένει το ίδιο, όταν συνδυάζεται με συμπίεση δεδομένων.

Ο MRHC2 όταν εφαρμόζεται στον κλασικό BR έχει το ίδιο και καλύτερο Hamming Loss και σε κάποιες τιμές του (K) γίνεται ακόμη καλύτερο, ενω ο χρόνος εκτέλεσης του αλγορίθμου παραμενει σχεδόν ίδιος με βάση, τον αρχικό χρόνο, χωρίς συμπίεση δεδομένων.

Πίνακας 5.14: Αποτελέσματα πειραμάτων στο σύνολο Water-Quality

Αλγόριθμος-Μέθοδος	Hamming Loss	Reduction Rate (%)	Distance Computations	Cpu Time (minutes)
MKNN1	0.4			0.67
MKNN2	0.59			0.70
MKNN1-MRHC1	0.35	40.65	369904.8	0.30
MKNN2-MRHC1	0.35	40.65	369904.8	0.30
MKNN2 (K=1)	0.38			0.64
MKNN2(K=3)	0.45			0.64
MKNN2(K=5)	0.5			0.65
BR(K=1)	0.38			0.026
BR(K=3)	0.36			0.027
BR(K=5)	0.35			0.027
BR(K=9)	0.33			0.028
BR(K=1)-MRHC2	0.37	40.637	369794	0.02
BR(K=3)-MRHC2	0.35	40.637	369794	0.02
BR(K=5)-MRHC2	0.34	40.637	369794	0.02
BR(K=9)-MRHC2	0.32	40.637	369794	0.02

Για το σύνολο δεδομένων **Scene** προκύπτει ο πίνακας 5.15. Σε αυτό τον πίνακα φαίνεται οτι οι αλγόριθμοι παραγωγής προτύπων καταφέρνουν να επιτύχουν ποσοστό μείωσης δεδομένων που φτάνει το 85.13%. Και εδώ το μέτρο απώλειας Hamming Loss για MKNN1 και MKNN2, σε σχέση με τον κλασικό αλγόριθμο BR, παραμένει σχεδόν το ίδιο.

Εύκολα γίνεται αντιληπτό οτι ο κλασικός BR όταν συνδυαστεί με τον αλγόριθμο μείωσης MRHC2, τότε ο χρόνος εκτέλεσής του, μειώνεται κατα 8 φορές με βάση, τον αρχικό χρόνο, χωρίς συμπίεση δεδομένων. Επίσης είναι πολύ εμφανές οτι το ποσοστό συμπίεσης των δεδομένων είναι ανάλογο του χρόνου εκτέλεσης του αλγορίθμου. Το Hamming Loss παραμένει το

ίδιο ή και καλύτερο σε κάποιες τιμές του (K).

Πίνακας 5.15: Αποτελέσματα πειραμάτων στο σύνολο Scene

Αλγόριθμος-Μέθοδος	Hamming Loss	Reduction Rate (%)	Distance Computations	Cpu Time (minutes)
MKNN1	0.16			26.92
MKNN2	0.24			26.25
MKNN1-MRHC1	0.17	85.13	480151.6	3.33
MKNN2-MRHC1	0.29	85.13	480151.6	3.43
MKNN2 (K=1)	0.13			26.46
MKNN2(K=3)	0.18			26.23
MKNN2(K=5)	0.22			26.40
BR(K=1)	0.13			0.57
BR(K=3)	0.12			0.57
BR(K=5)	0.11			0.56
BR(K=9)	0.12			0.57
BR(K=1)-MRHC2	0.12	85.13	480151.6	0.08
BR(K=3)-MRHC2	0.12	85.13	480151.6	0.08
BR(K=5)-MRHC2	0.12	85.13	480151.6	0.08
BR(K=9)-MRHC2	0.12	85.13	480151.6	0.08

Για το σύνολο δεδομένων **Yeast** προκύπτει ο πίνακας 5.16. Σε αυτό τον πίνακα φαίνεται οτι οι αλγόριθμοι παραγωγής MRHC1 και MRHC2 προτύπων καταφέρνουν να επιτύχουν ποσοστό μείωσης δεδομένων που φτάνει το 51.85% σε σχέση με το αρχικό σύνολο. Το μέτρο απώλειας Hamming Loss, για MKNN1 και MKNN2, παραμένει το ίδιο, όταν συνδυάζεται με αλγορίθμους μείωσης δεδομένων, σε σχέση με τον κλασικό BR και σε κάποιες τιμές του (K) είναι καλύτερο. Και εδώ ο χρόνος εκτέλεσης του αλγορίθμου μειώνεται πάνω απο το μισό, με βάση, τον αρχικό χρόνο, χωρίς συμπίεση δεδομένων.

Πίνακας 5.16: Αποτελέσματα πειραμάτων στο σύνολο Yeast

Αλγόριθμος-Μέθοδος	Hamming Loss	Reduction Rate (%)	Distance Computations	Cpu Time (minutes)
MKNN1	0.26			12.89
MKNN2	0.51			12.78
MKNN1-MRHC1	0.26	51.85	1341732.8	4.9
MKNN2-MRHC1	0.27	51.85	1341732.8	4.9
MKNN2 (K=1)	0.24			12.98
MKNN2(K=3)	0.31			12.66
MKNN2(K=5)	0.37			12.70
BR(K=1)	0.24			0.44
BR(K=3)	0.21			0.46
BR(K=5)	0.20			0.44
BR(K=9)	0.19			0.44
BR(K=1)-MRHC2	0.23	51.85	1341732.8	0.17
BR(K=3)-MRHC2	0.21	51.85	1341732.8	0.19
BR(K=5)-MRHC2	0.20	51.85	1341732.8	0.17
BR(K=9)-MRHC2	0.20	51.85	1341732.8	0.18

Για το σύνολο δεδομένων **Emotions** προκύπτει ο πίνακας 5.17. Σε αυτό τον πίνακα φαίνεται οτι οι αλγόριθμοι παραγωγής MRHC1 και MRHC2 προτύπων καταφέρνουν να επιτύχουν ποσοστό μείωσης δεδομένων που φτάνει το 65.73% σε σχέση με το αρχικό σύνολο.

Το μέτρο απώλειας Hamming Loss για MKNN1 και MKNN2, παραμένει το ίδιο, όταν συνδυάζεται με αλγορίθμους μείωσης δεδομένων, σε σχέση με τον κλασικό BR και σε κάποιες τιμές

του (K) είναι καλύτερο. Και εδώ ο χρόνος εκτέλεσης του αλγορίθμου μειώνεται με βάση, τον αρχικό χρόνο, χωρίς συμπίεση δεδομένων.

Πίνακας 5.17: Αποτελέσματα πειραμάτων στο σύνολο Emotions

Αλγόριθμος-Μέθοδος	Hamming Loss	Reduction Rate (%)	Distance Computations	Cpu Time (minutes)
MKNN1	0.26			0.41
MKNN2	0.37			0.40
MKNN1-MRHC1	0.26	65.73	65750.4	0.12
MKNN2-MRHC1	0.31	65.73	65750.4	0.12
MKNN2 (K=1)	0.24			0.42
MKNN2(K=3)	0.29			0.39
MKNN2(K=5)	0.34			0.40
BR(K=1)	0.24			0.015
BR(K=3)	0.22			0.016
BR(K=5)	0.20			0.016
BR(K=9)	0.19			0.017
BR(K=1)-MRHC2	0.22	65.73	65750.4	0.011
BR(K=3)-MRHC2	0.20	65.73	65750.4	0.011
BR(K=5)-MRHC2	0.20	65.73	65750.4	0.011
BR(K=9)-MRHC2	0.20	65.73	65750.4	0.011

Για το σύνολο δεδομένων **Birds** προκύπτει ο πίνακας 5.18. Σε αυτό τον πίνακα φαίνεται οτι οι αλγόριθμοι παραγωγής MRHC1 και MRHC2 προτύπων, καταφέρνουν να επιτύχουν ποσοστό μείωσης δεδομένων που φτάνει το 42.70% σε σχέση με το αρχικό σύνολο.

Το μέτρο απώλειας Hamming Loss για τον κλασικό αλγόριθμο BR παραμένει περίπου το ίδιο, όταν συνδυάζεται, με αλγορίθμους μείωσης δεδομένων (MRHC2), σε σχέση με τον κλασικό BR. Ενώ ο χρόνος εκτέλεσης του αλγορίθμου BR μειώνεται περίπου στο μισό με βάση, τον αρχικό χρόνο, χωρίς συμπίεση δεδομένων.

Πίνακας 5.18: Αποτελέσματα πειραμάτων στο σύνολο Birds

Αλγόριθμος-Μέθοδος	Hamming Loss	Reduction Rate (%)	Distance Computations	Cpu Time (minutes)
MKNN1	0.14			0.39
MKNN2	0.54			0.39
MKNN1-MRHC1	0.15	42.70	79763.8	0.22
MKNN2-MRHC1	0.39	42.70	79763.8	0.22
MKNN2 (K=1)	0.09			0.39
MKNN2(K=3)	0.17			0.39
MKNN2(K=5)	0.24			0.39
BR(K=1)	0.09			0.057
BR(K=3)	0.08			0.056
BR(K=5)	0.08			0.061
BR(K=9)	0.08			0.057
BR(K=1)-MRHC2	0.09	42.70	79763.8	0.035
BR(K=3)-MRHC2	0.08	42.70	79763.8	0.036
BR(K=5)-MRHC2	0.08	42.70	79763.8	0.035
BR(K=9)-MRHC2	0.09	42.70	79763.8	0.035

Για το σύνολο δεδομένων **CHD_49** προκύπτει ο πίνακας 5.19. Σε αυτό τον πίνακα φαίνεται οτι οι αλγόριθμοι παραγωγής MRHC1 και MRHC2 προτύπων καταφέρνουν να επιτύχουν ποσοστό μείωσης δεδομένων που φτάνει το 65.47% σε σχέση με το αρχικό σύνολο.

Το μέτρο απώλειας Hamming Loss για MKNN1 και MKNN2 παραμένει το ίδιο, όταν συνδυάζεται με αλγορίθμους μείωσης δεδομένων, σε σχέση με τον κλασικό BR και σε κάποιες τιμές

του (K) είναι καλύτερο. Και εδώ ο χρόνος εκτέλεσης του αλγορίθμου μειώνεται με βάση, τον αρχικό χρόνο, χωρίς συμπίεση δεδομένων.

Πίνακας 5.19: Αποτελέσματα πειραμάτων στο σύνολο CHD_49

Αλγόριθμος-Μέθοδος	Hamming Loss	Reduction Rate (%)	Distance Computations	Cpu Time (minutes)
MKNN1	0.36			0.23
MKNN2	0.41			0.23
MKNN1-MRHC1	0.35	65.47	65892.2	0.07
MKNN2-MRHC1	0.36	65.47	65892.2	0.07
MKNN2 (K=1)	0.36			0.23
MKNN2(K=3)	0.38			0.23
MKNN2(K=5)	0.40			0.23
BR(K=1)	0.36			0.012
BR(K=3)	0.34			0.012
BR(K=5)	0.33			0.013
BR(K=9)	0.31			0.014
BR(K=1)-MRHC2	0.35	65.47	65892.2	0.01
BR(K=3)-MRHC2	0.33	65.47	65892.2	0.009
BR(K=5)-MRHC2	0.32	65.47	65892.2	0.001
BR(K=9)-MRHC2	0.30	65.47	65892.2	0.009

Για το σύνολο δεδομένων **Image** προκύπτει ο πίνακας 5.20. Σε αυτό τον πίνακα φαίνεται οτι οι αλγόριθμοι παραγωγής MRHC1 και MRHC2 προτύπων, καταφέρνουν να επιτύχουν ποσοστό μείωσης δεδομένων, που φτάνει το 71.71% σε σχέση με το αρχικό σύνολο.

Το μέτρο απώλειας Hamming Loss για τον κλασικό αλγόριθμο BR, παραμένει περίπου το ίδιο, όταν συνδυάζεται με τον αλγόριθμο μείωσης δεδομένων (MRHC2), σε σχέση με τον κλασικό BR. Ενώ ο χρόνος εκτέλεσης του αλγορίθμου BR, μειώνεται περίπου τέσσερις φορές, με βάση τον αρχικό χρόνο, χωρίς συμπίεση δεδομένων.

Πίνακας 5.20: Αποτελέσματα πειραμάτων στο σύνολο Image

Αλγόριθμος-Μέθοδος	Hamming Loss	Reduction Rate (%)	Distance Computations	Cpu Time (minutes)
MKNN1	0.37			18.37
MKNN2	0.48			18.24
MKNN1-MRHC1	0.39	71.71	419476.4	4.6
MKNN2-MRHC1	0.49	71.71	419476.4	4.7
MKNN2 (K=1)	0.29			18.42
MKNN2(K=3)	0.41			18.07
MKNN2(K=5)	0.48			18.24
BR(K=1)	0.29			0.28
BR(K=3)	0.28			0.29
BR(K=5)	0.27			0.29
BR(K=9)	0.27			0.29
BR(K=1)-MRHC2	0.28	71.71	419476.4	0.07
BR(K=3)-MRHC2	0.27	71.71	419476.4	0.08
BR(K=5)-MRHC2	0.25	71.71	419476.4	0.08
BR(K=9)-MRHC2	0.25	71.71	419476.4	0.08

Για το σύνολο δεδομένων **Mediamill** προκύπτει ο πίνακας 5.21.

Σε αυτό τον πίνακα φαίνεται οτι οι αλγόριθμοι παραγωγής MRHC1 και MRHC2 προτύπων, καταφέρνουν να επιτύχουν ποσοστό μείωσης δεδομένων που φτάνει το 55,86% σε σχέση με το αρχικό σύνολο. Το μέτρο απώλειας Hamming Loss για τον κλασικό αλγόριθμο BR παραμένει περίπου το ίδιο, όταν συνδυάζεται, με τον αλγόριθμο μείωσης δεδομένων (MRHC2), σε σχέση

Πίνακας 5.21: Αποτελέσματα πειραμάτων στο σύνολο Mediamill

Αλγόριθμος-Μέθοδος	Hamming Loss	Reduction Rate (%)	Distance Computations	Cpu Time (minutes)
MKNN1	0.055			590.34
MKNN2	0.34			570.27
MKNN1-MRHC1	0.038	55.86	806819021.2	170.74
MKNN2-MRHC1	0.038	55.86	806819021.2	170.50
MKNN2 (K=1)	0.049			540.48
MKNN2(K=3)	0.073			540.98
MKNN2(K=5)	0.092			560.71
BR(K=1)	0.049			340.9
BR(K=3)	0.041			380.09
BR(K=5)	0.039			400.37
BR(K=9)	0.037			450.32
BR(K=1)-MRHC2	0.04	55.86	806819021.2	180.46
BR(K=3)-MRHC2	0.035	55.86	806819021.2	180.86
BR(K=5)-MRHC2	0.034	55.86	806819021.2	200.73
BR(K=9)-MRHC2	0.034	55.86	806819021.2	190.9

με τον κλασικό BR. Ενώ ο χρόνος εκτέλεσης του αλγορίθμου BR, μειώνεται περίπου τέσσερις φορές με βάση, τον αρχικό χρόνο, χωρίς συμπίεση δεδομένων.

Ανακεφαλαιώνοντας τα παραπάνω συμπεράσματα παρουσιάζεται ο πίνακας 5.22 στον οποίο φαίνονται οι μέσοι όροι όλων των αποτελεσμάτων των πειραμάτων.

Πίνακας 5.22: Μέσοι όροι αποτελεσμάτων πειραμάτων

Αλγόριθμος-Μέθοδος	Avg Hamming Loss	Avg Reduction Rate (%)	Avg Distance Computations	Avg Cpu Time (minutes)
MKNN1	0.24			72.27
MKNN2	0.47			69.95
MKNN1-MRHC1	0.23	57.67	90039391.22	20.49
MKNN2-MRHC1	0.23	57.67	90039391.22	20.49
MKNN2 (K=1)	0.21			66.70
MKNN2(K=3)	0.28			66.65
MKNN2(K=5)	0.33			68.88
BR(K=1)	0.21			38.06
BR(K=3)	0.20			42.41
BR(K=5)	0.19			44.67
BR(K=9)	0.18			50.22
BR(K=1)-MRHC2	0.20	57.63	90037319.8	20.11
BR(K=3)-MRHC2	0.19	57.63	90037319.8	20.16
BR(K=5)-MRHC2	0.18	57.63	90037319.8	22.36
BR(K=9)-MRHC2	0.18	57.63	90037319.8	21.27

5.4 Σύγκριση των Αποτελεσμάτων

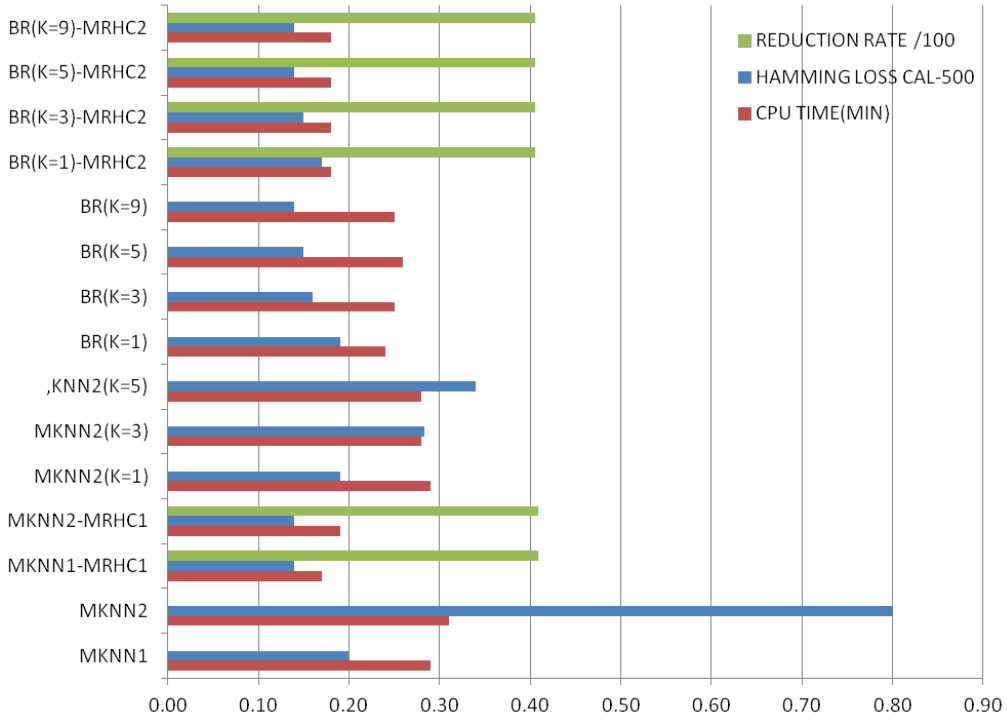
Σε αυτή την ενότητα φαίνεται σε μορφή διαγραμμάτων τα αποτελέσματα των πειραμάτων χωριστά για κάθε σύνολο δεδομένων πολλαπλών ετικετών. Συγκρίνοντας και μελετώντας τα διαγράμματα αυτά παρατηρούνται και εξάγονται τα παρακάτω συμπεράσματα.

Οι νέοι αλγόριθμοι κατηγοριοποίησης MKNN1 και MKNN2 οι οποίοι βασίζονται στη φιλοσοφία της αναζήτηση εγγύτερων γειτόνων δείχνουν οτι καταφέρνουν να κατηγοριοποιήσουν αρκετά κάλα ένα νέο στιγμιότυπο που αποτελείται απο πολλαπλές ετικέτες. Ο MRHC1 μετατρέπει ένα πρόβλημα πολλαπλών ετικετών σε πρόβλημα μονής ετικέτας. Απο μόνη της, αυτή η διαδικασία θα περίμενε κανείς να έχει μεγάλη απώλεια πληροφορίας. Αν' αυτού παρατηρείται

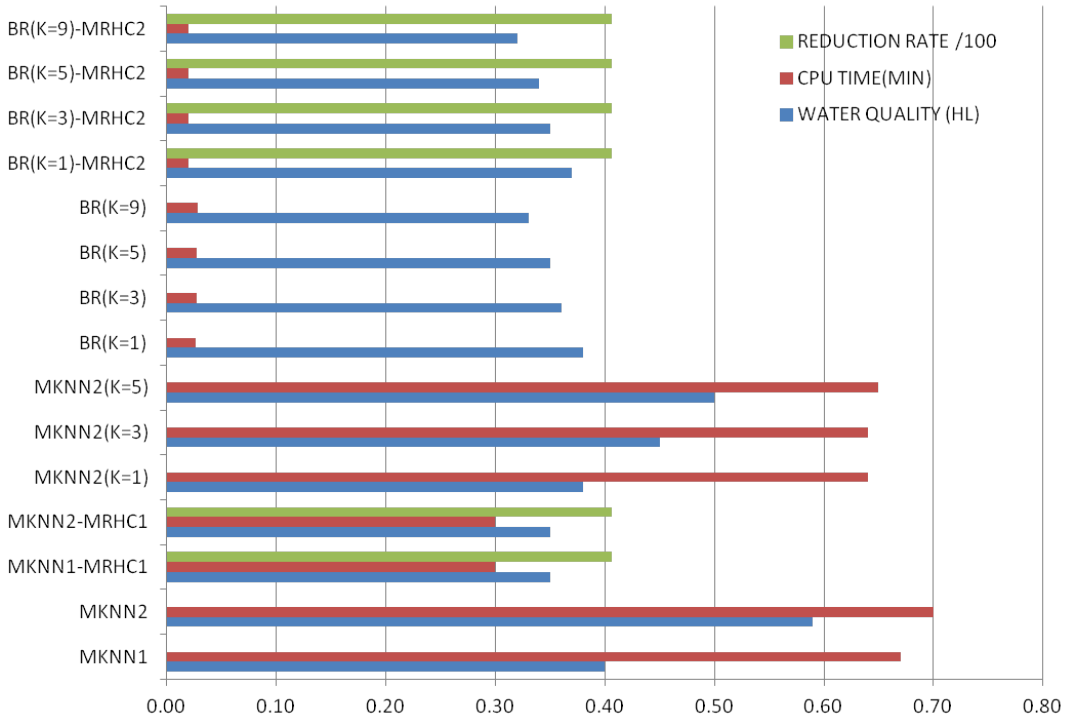
οτι ο MKNN1 επιτυγχάνει ακρίβεια που παραμένει σε ικανοποιητικά επίπεδα, ενώ ο χρόνος εκτέλεσης μειώνεται σε σχέση με τους αλγορίθμους που δεν χρησιμοποιούν μείωση δεδομένων.

Βέβαια ανάλογα με το πλήθος των δεδομένων που έχει το κάθε σύνολο παρατηρείται οτι αυξάνεται και ο χρόνος εκτέλεσης του. Αρα για μεγάλα σύνολα δεδομένων θα προκύψει μεγάλο υπολογιστικό κόστος. Επίσης κάποιες μικρές διαφορές που παρατηρούνται στο μέτρο απώλειας Hamming Loss πιθανόν να οφείλεται στο γεγονός οτι σε κάποια σύνολα δεδομένων μπορεί να υπάρχει θόρυβος (περιττά στιγμιότυπα) ή στην ύπαρξη στιγμιοτύπων που βρίσκονται μακριά απο τα υπόλοιπα στιγμιότυπα του συνόλου (outliers). Αυτά λοιπόν τα μειονεκτήματα που έχει εκ κατασκευής ο Αλγόριθμος του εγγύτερου γείτονα (KNN), επιλύονται πλήρως με την εφαρμογή δυο διαφορετικών αλγορίθμων παραγωγής προτύπων, τους MRHC1 και MRHC2.

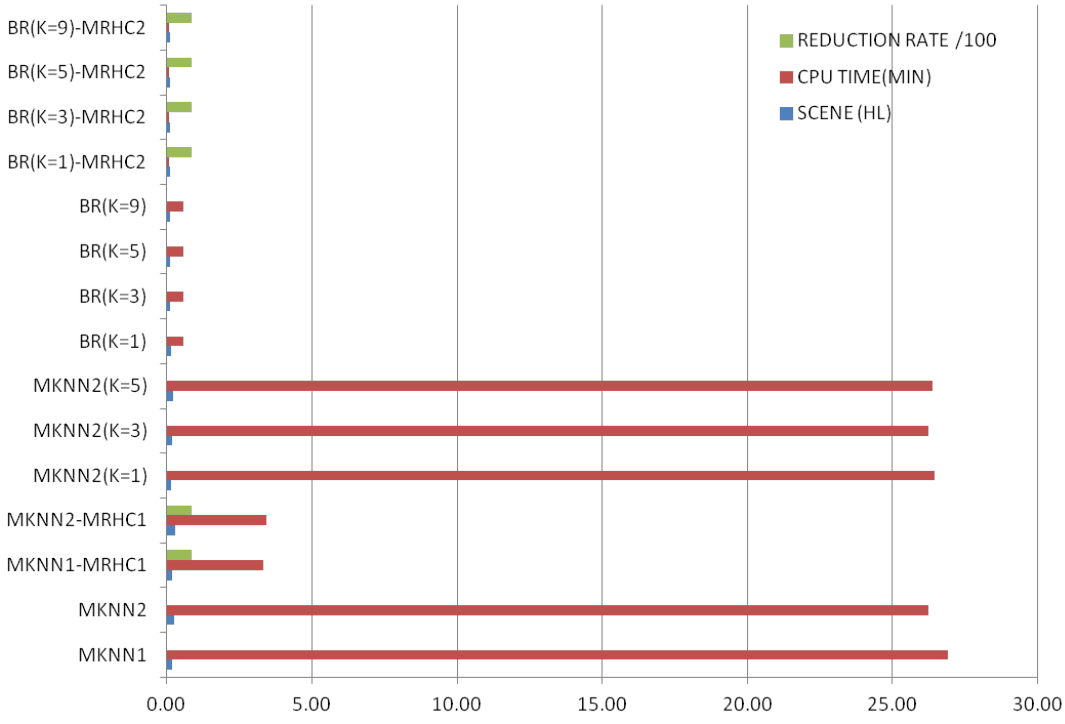
Σε όλες τις περιπτώσεις που απεικονίζονται παρακάτω φαίνεται οτι όταν εφαρμόζεται τεχνική μείωσης δεδομένων επιτυγχάνονται τρια διαφορετικά πράγματα. Αρχικά επιτυγχάνεται μείωση των αρχικών δεδομένων απο 40.65% έως 85.13%, αυτό έχει σαν αποτέλεσμα να μειώνεται σε πολύ μεγάλο βαθμό το υπολογιστικό κόστος κατά την κατηγοριοποίηση. Επίσης με βάση αυτές τις μεθόδους που βασίζονται κατα κύριο λόγο στις ομοιογενείς συστάδες αφαιρούνται απο τα αρχικά σύνολα, τα στιγμιότυπα που βρίσκονται μακριά απο τα υπόλοιπα στιγμιότυπα (outliers). Αυτό έχει σαν αποτέλεσμα να παραμένει σε χαμηλές τιμές η απώλεια Hamming Loss, ενώ σε ορισμένες περιπτώσεις, αυτή η μετρική μειώνεται και άλλο. Όλα αυτά βέβαια σε μικρότερο χρόνο εκτέλεσης.



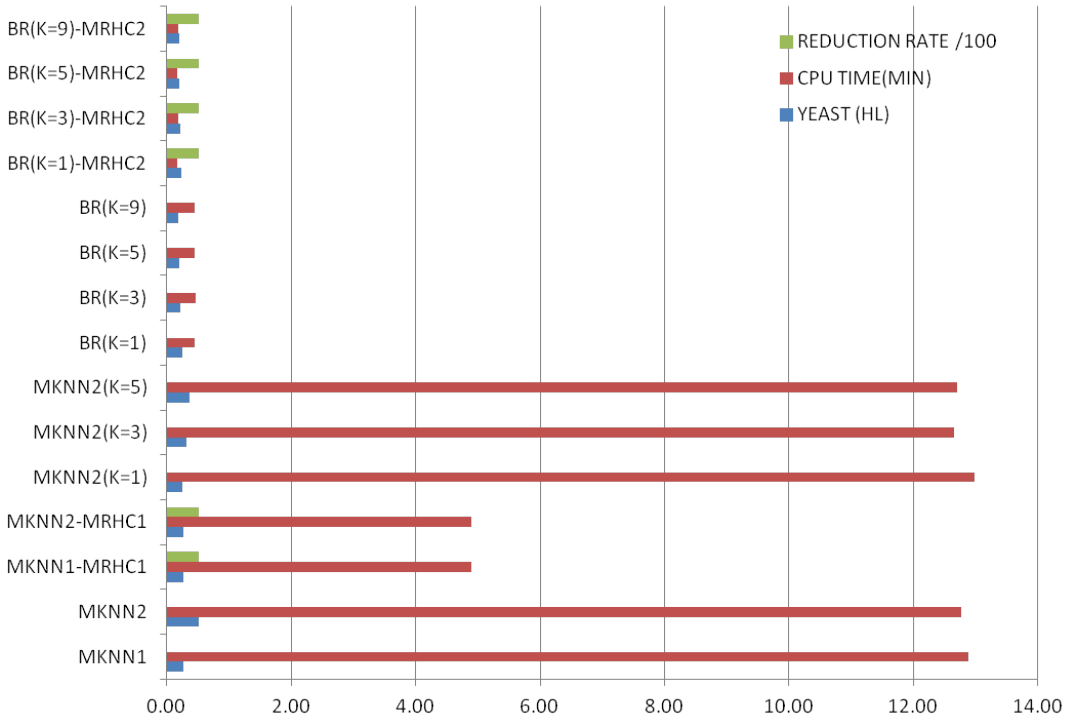
Σχήμα 5.4: Αναπαράσταση πειραμάτων CAL500



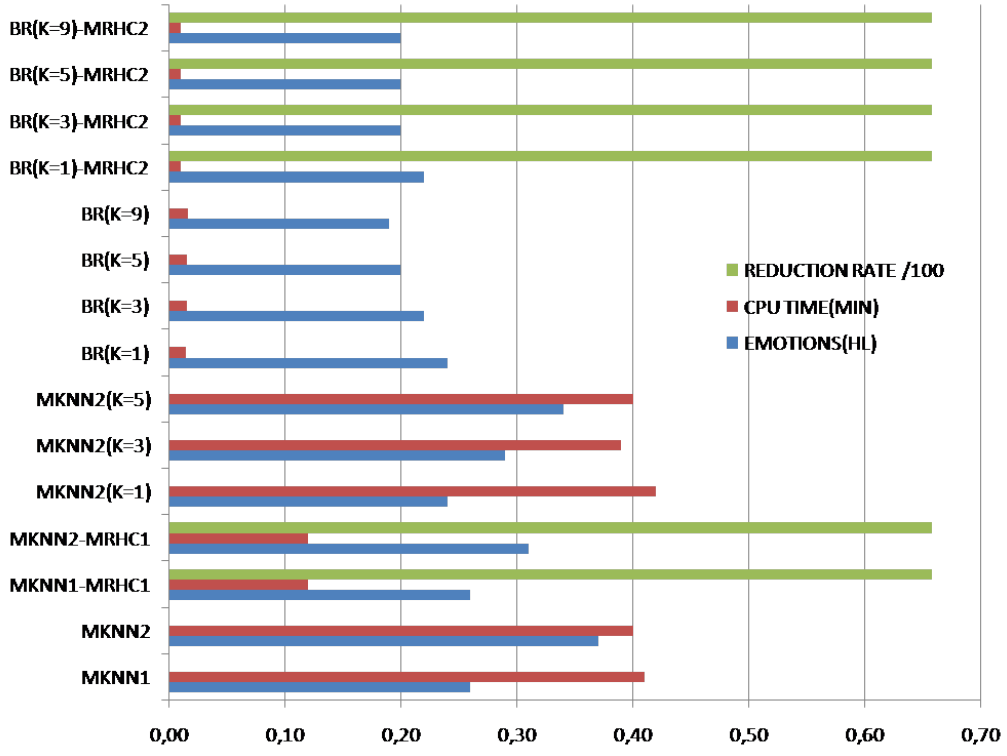
Σχήμα 5.5: Αναπαράσταση πειραμάτων WATER QUALITY



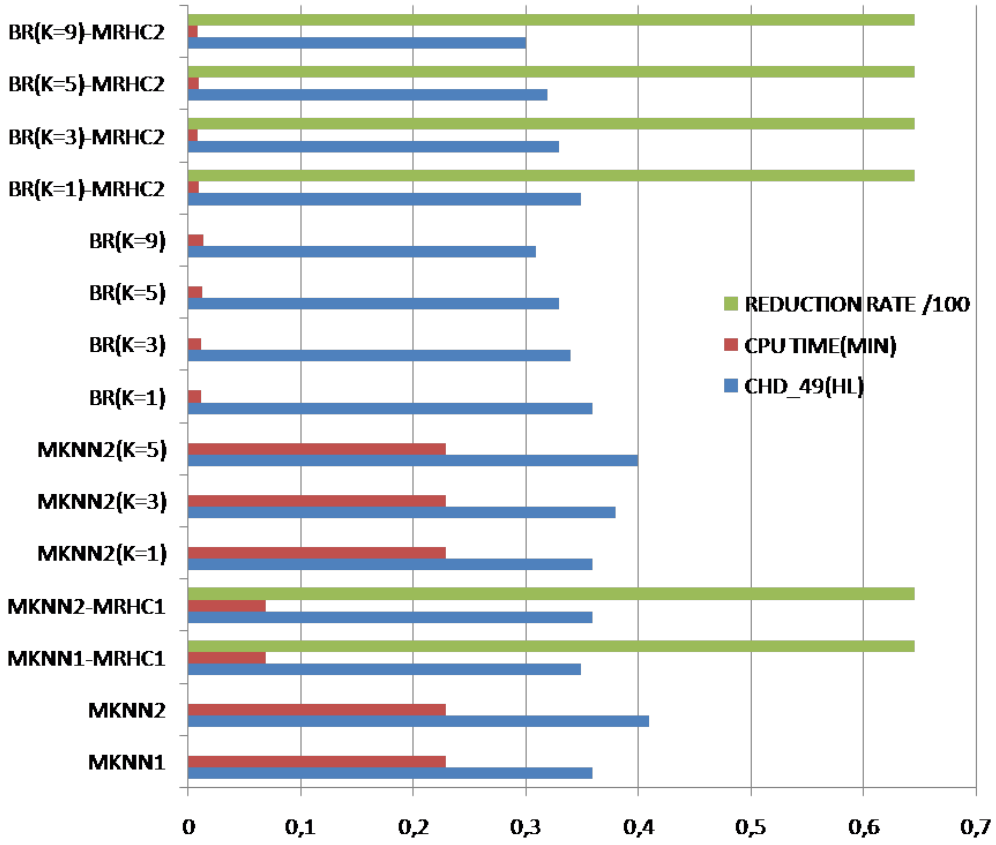
Σχήμα 5.6: Αναπαράσταση πειραμάτων SCENE



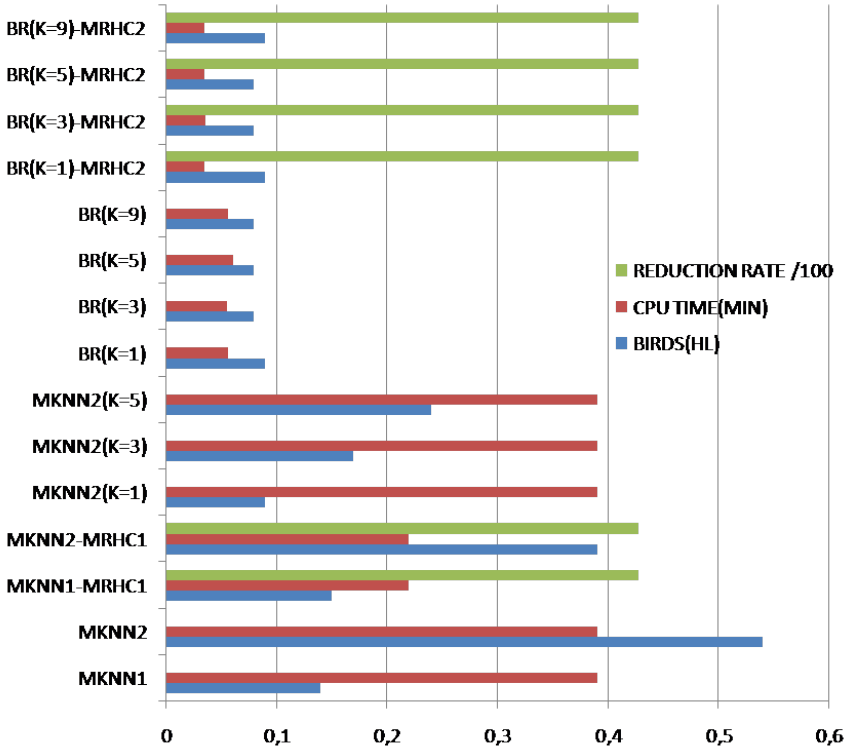
Σχήμα 5.7: Αναπαράσταση πειραμάτων YEAST



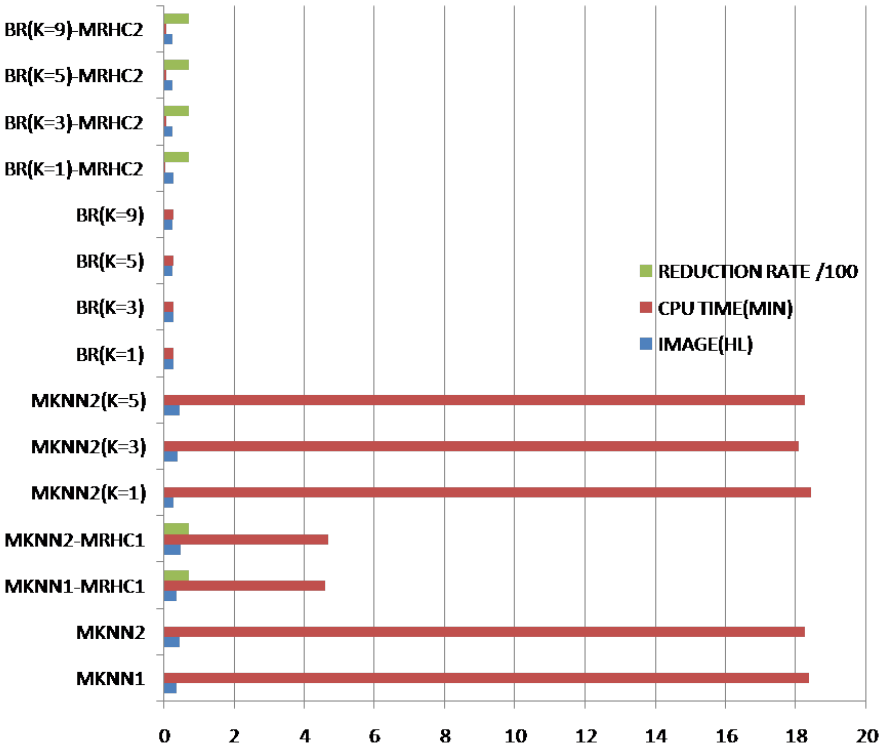
Σχήμα 5.8: Αναπαράσταση πειραμάτων EMOTIONS



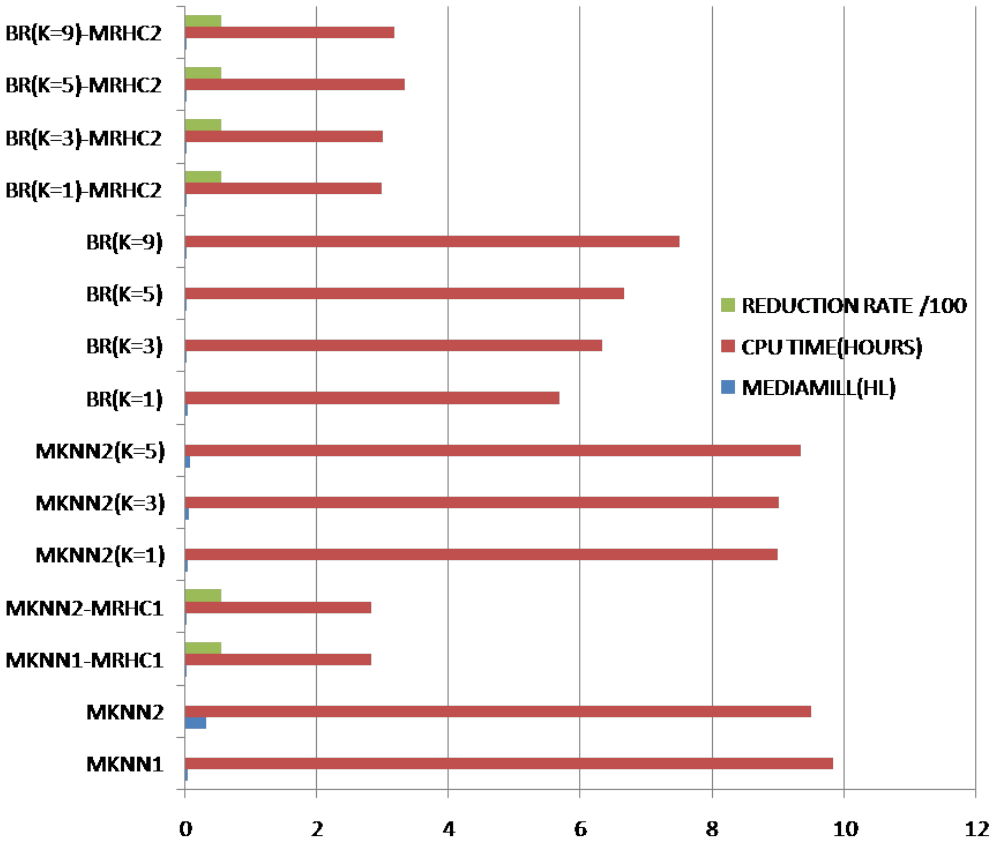
Σχήμα 5.9: Αναπαράσταση πειραμάτων CHD49



Σχήμα 5.10: Αναπαράσταση πειραμάτων BIRDS

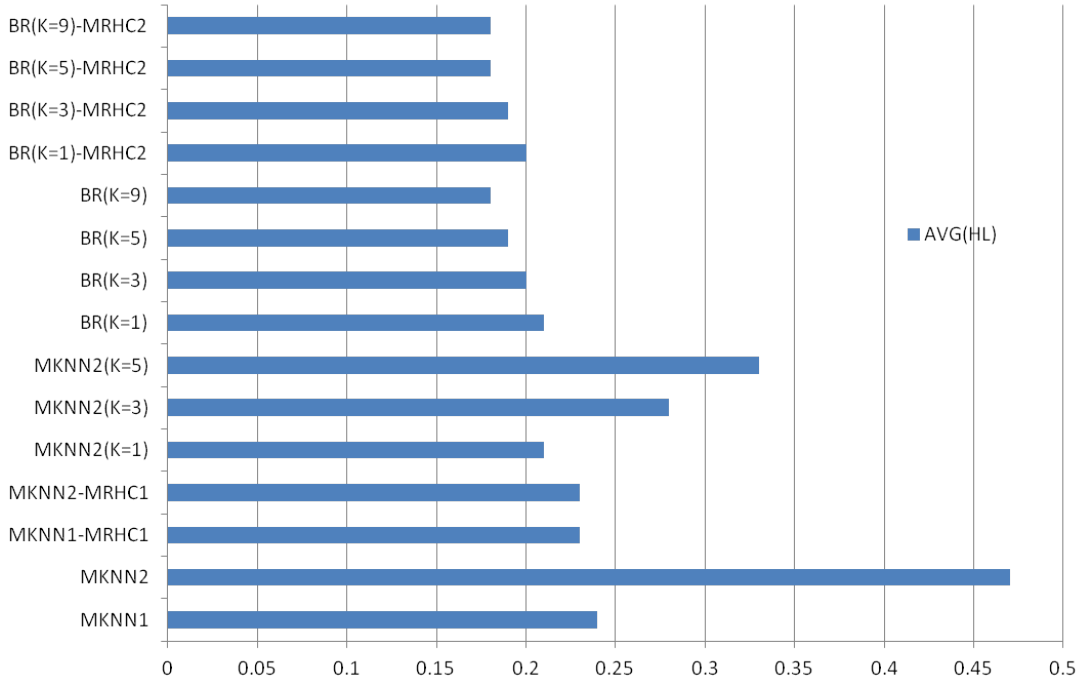


Σχήμα 5.11: Αναπαράσταση πειραμάτων IMAGE

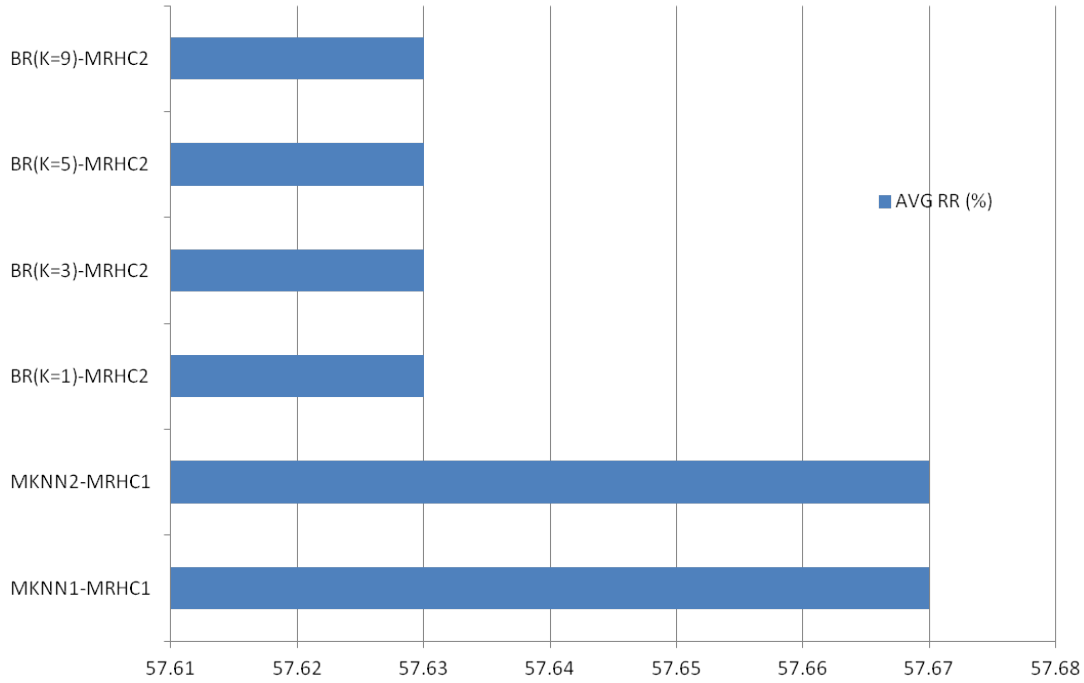


Σχήμα 5.12: Αναπαράσταση πειραμάτων MEDIAMILL

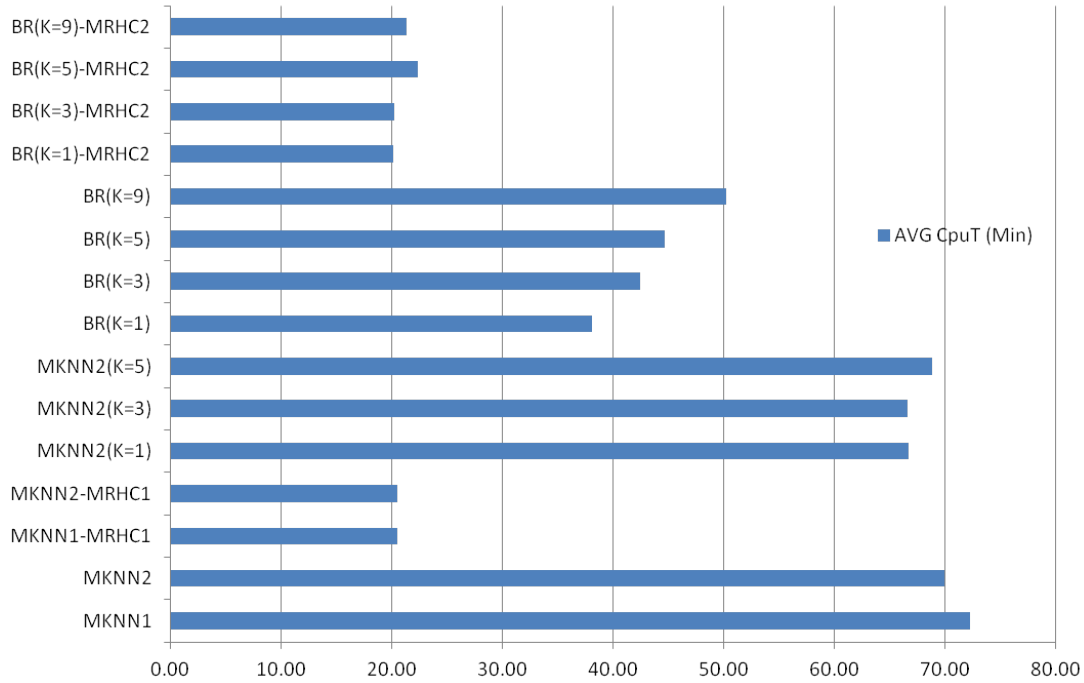
Επιπλέον παρουσιάζεται το σχήμα 5.13 στο οποίο φαίνονται συγκεντρωτικά οι μέσοι όροι, της μετρικής απώλειας (HL), στο σχήμα 5.14 στο οποίο φαίνονται συγκεντρωτικά οι μέσοι όροι, του ρυθμού συμπίεσης (RR), στο σχήμα 5.15 στο οποίο φαίνονται συγκεντρωτικά οι μέσοι όροι του χρόνου εκτέλεσης (CpuT) όλων των συνόλων δεδομένων, ενώ στο σχήμα 5.16 φαίνονται συγκεντρωτικά οι μέσοι όροι του υπολογισμού αποστάσεων (DC) όλων των συνόλων δεδομένων.



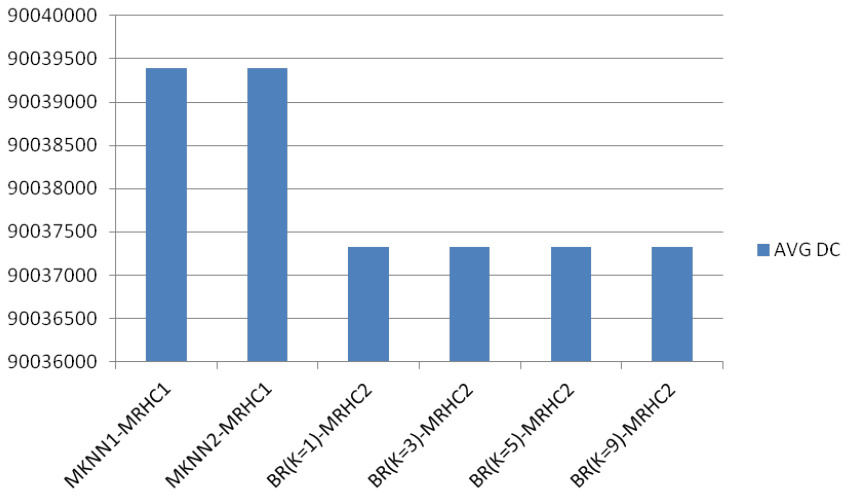
Σχήμα 5.13: Αποτελέσματα μέσου όρου HL όλων των συνόλων δεδομένων



Σχήμα 5.14: Αποτελέσματα μέσου όρου RR όλων των συνόλων δεδομένων



Σχήμα 5.15: Αποτελέσματα μέσου όρου CpuT όλων των συνόλων δεδομένων



Σχήμα 5.16: Αποτελέσματα μέσου όρου υπολογισμού αποστάσεων όλων των συνόλων δεδομένων

5.5 Συμπεράσματα πειραματικής μελέτης

Σε αυτό το κεφάλαιο αρχικά παρουσιάστηκαν τα σύνολα δεδομένων που χρησιμοποιήθηκαν στην πειραματική μελέτη η οποία εκπονήθηκε στα πλαίσια της παρούσας διπλωματικής εργασίας. Στην συνέχεια παρουσιάστηκαν τα αποτελέσματα των πειραμάτων. Μέσω των μεθόδων μας, υλοποιήθηκε με επιτυχία η μείωση του πλήθους των στιγμιοτύπων, σε σύνολα πολλαπλών ετικετών, σε πολύ μεγάλο βαθμό και αυτό επετεύχθει με την παραγωγή αντιπροσωπευτικών στιγμιοτύπων. Όπως φάνηκε, αυτή η διαδικασία είχε απο την μια, μεγάλη επίπτωση στην μείωση του υπολογιστικού χρόνου εκτέλεσης των αλγορίθμων (μέχρι και οκτώ φορές λιγότερος χρόνος), χωρίς όμως απο την άλλη να χαθεί χρήσιμη πληροφορία απο τα σύνολα, καθώς η απώλεια Hamming Loss (ακρίβεια κατηγοριοποίησης) παρέμεινε σε υψηλά επίπεδα, όπως ήταν, πριν την μείωση των δεδομένων.

Μάλιστα, σε αρκετές των περιπτώσεων η απώλεια Hamming Loss (ακρίβεια κατηγοριοποίησης), φαίνεται οτι έγινε ακόμη καλύτερη, γεγονός που δείχνει οτι απο τα αρχικά σύνολα δεν αφαιρέθηκε χρήσιμη πληροφορία, ενώ παράλληλα, εξομαλύνθηκε η επίδραση του θορύβου που πιθανότατα υπήρχε σε αρχικά δεδομένα. Στην συνέχεια σχολιάστηκαν και συγκρίθηκαν τα αποτελέσματα μεταξύ τους. Παρατηρήθηκε πως και για τους δυο νέους αλγορίθμους (MKNN1 και MKNN2) κατηγοριοποίησης συνόλων πολλαπλών ετικετών, η απώλεια Hamming Loss παραμένει πολύ κοντά στην κλασική μέθοδο δυαδικής συνάφειας (Binary Relevance) που εφαρμόζεται σε τέτοιου είδους σύνολα δεδομένων. Με βάση τα συγκεντρωτικά σχήματα που παρατέθηκαν στο τέλος του κεφαλαίου 5 φαίνεται οτι συνολικά για όλα τα σύνολα δεδομένων ο άλγόριθμος MRHC1 επιτυγχάνει λίγο καλύτερο (χαμηλότερο) μέσο όρο απώλειας HL, χαμηλό μέσο όρο χρόνου εκτέλεσης και υψηλό μέσο όρο συμπίεσης δεδομένων. Απο την άλλη ο μέσος όρος του υπολογισμού των αποστάσεων είναι σχεδόν ίδιος και για τους δυο αλγόριθμους (MRHC1,MRHC2).

6 Συμπεράσματα και Μελλοντική Έρευνα

Υπάρχουν πολλές τεχνικές μείωσης δεδομένων εκπαίδευσης διαθέσιμες στη βιβλιογραφία για προβλήματα κατηγοριοποίησης. Ωστόσο, η συντριπτική πλειοψηφία των τεχνικών αυτών αφορά προβλήματα κατηγοριοποίησης μονής ετικέτας. Ελάχιστες ερευνητικές εργασίες αφορούν τη μείωση δεδομένων εκπαίδευσης πολλαπλών ετικετών. Η απόδοση αυτών των ελάχιστων προσεγγίσεων εξαρτώνται, σε μεγάλο βαθμό, από παραμέτρους που προσδιορίζει ο χρήστης, μέσω υπολογιστικά κοστοβόρων διαδικασιών. Επιπρόσθετα, οι τεχνικές μείωσης δεδομένων μονής ετικέτας δεν μπορούν να εφαρμοστούν σε συνδυασμό με τις διαδεδομένες μεθόδους μετασχηματισμού προβλήματος πολλαπλών ετικετών σε πρόβλημα μονής ετικέτας. Αυτές οι παρατηρήσεις αποτέλεσαν το κίνητρο της διπλωματικής εργασίας.

Στόχος της εργασίας ήταν η ανάπτυξη αλγορίθμων μείωσης δεδομένων εκπαίδευσης πολλαπλών ετικετών, οι οποίοι να μην περιλαμβάνουν παραμέτρους, που ο χρήστης πρέπει να προσδιορίσει. Επιπρόσθετος στόχος ήταν η ανάπτυξη νέων αποτελεσματικών αλγορίθμων κατηγοριοποίησης εγγύτερων γειτόνων, που χρησιμοποιούν, τα πρότυπα που παράγουν οι προτεινόμενοι αλγόριθμοι μείωσης δεδομένων για να εκτελέσουν κατηγοριοποίηση πολλαπλών ετικετών. Οι αλγόριθμοι μείωσης δεδομένων πολλαπλών ετικετών που αναπτύχθηκαν σκοπεύουν στο να μειώσουν όσο γίνεται περισσότερο τον όγκο των δεδομένων ενός συνόλου πολλαπλών ετικετών, έτσι ώστε αυτομάτως να μειωθεί και το υπολογιστικό κόστος κατα την κατηγοριοποίηση. Απο την άλλη, φυσικά θα πρέπει η ακρίβεια να παραμένει σε υψηλά επίπεδα και αυτό μπορεί να επιτευχθεί, αν κατα την εφαρμογή των μεθόδων, δεν αφαιρείται χρήσιμη πληροφορία και παράγονται αντιπροσωπευτικά στιγμιότυπα απο το αρχικό σύνολο.

Έτσι, η παρούσα εργασία παρουσίασε τους αλγόριθμους μείωσης δεδομένων MRHC1 και MRHC2 καθώς και τους κατηγοριοποιητές MKNN1 και MKNN2 που επιτυγχάνουν τους προαναφερθέντες στόχους. Οι αλγόριθμοι MRHC1 και RHC2 μπορούν να θεωρηθούν παραλλαγές του αλγορίθμου μείωσης δεδομένων μονής ετικέτας RHC για προβλήματα πολλαπλών ετικετών. Ο RHC απο τη φύση του εφαρμόζεται χωρίς καμία παραμετροποίηση. Η βασική λειτουργία των αλγορίθμων αυτών βασίζεται στη συσταδοποίηση K μέσων, ο οποίος όμως εκτελείται επαναληπτικά, στις μη ομοιογενείς συστάδες που παράγονται. Οι MRHC1 και MRHC2 θεωρούν μια συστάδα ως ομοιογενή όταν όλα τα στιγμιότυπα της συστάδας έχουν τουλάχιστον μια κοινή ετικέτα. Στο τέλος της επαναληπτικής διαδικασίας συσταδοποίησης όλες οι συστάδες γίνονται ομοιογενείς και τα κέντρα των συστάδων αποτελούν τα πρότυπα που συνθέτουν το συμπυκνωμενο σύνολο δεδομένων.

Η πειραματική μελέτη που εκπονήθηκε στα πλαίσια της παρούσας διπλωματικής εργασίας βασίστηκε σε εννέα σύνολα δεδομένων εκπαίδευσης πολλαπλών ετικετών και για την αξιολόγηση της ακρίβειας που επιτυγχάνουν, μετρήθηκε η απώλεια Hamming Loss. Τα πειραματικά αποτελέσματα έδειξαν οτι οι προτεινόμενοι αλγόριθμοι επιτυγχάνουν υψηλούς λόγους μείωσης δεδομένων ενώ παράλληλα, η απώλεια Hamming Loss δεν επηρεάζεται αρνητικά ενώ, σε κάποιες περιπτώσεις παρατηρείται και μείωση της απώλειας. Όλα αυτά επιτεύχθηκαν χωρίς την εμπλοκή διαδικασιών για προσδιορισμό τιμών σε παραμέτρους. Συνεπώς, οι MRHC1 και

MRHC2 καθώς και οι αντίστοιχοι κατηγοριοποιητές MKNN1 και MKNN2 επιτύγχαν τους στόχους για τους οποίους αναπτύχθηκαν.

Όπως έχει πολλές φορές αναφερθεί, ενώ υπάρχουν δεκάδες αλγόριθμοι μείωσης δεδομένων που μπορούν να συνδυαστούν με τον αλγόριθμο κατηγοριοποίησης εγγύτερων γειτόνων, ελάχιστοι απο αυτούς αφορούν προβλήματα κατηγοριοποίησης πολλαπλών ετικετών. Συνεπώς, ο συγκεκριμένος χώρος θεωρείται ένα ελεύθερο πεδίο έρευνας. Οι κατεύθύνσεις για μελλοντική έρευνα περιλαμβάνουν την προσαρμογή γνωστών αλγορίθμων μείωσης δεδομένων, ώστε αυτοί να μπορούν να χειριστούν, δεδομένα πολλαπλών ετικετών χωρίς να απαιτείται μετασχηματισμός προβλήματος. Φυσικά, στις μελλοντικές κατευθύνσεις περιλαμβάνεται η ανάπτυξη νέων αλγορίθμων επιλογής και παραγωγής προτύπων για προβλήματα πολλαπλών ετικετών.

Βιβλιογραφία

[1] M. Zhang and Z. Zhou, "A review on multi-label learning algorithms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, pp. 1819–1837, 2014.

[2] S. García, J. Luengo, and F. Herrera, *Introduction*, pp. 1–17. Cham: Springer International Publishing, 2015.

[3] F. Herrera, F. Charte, A. J. Rivera, and M. J. del Jesus, *Multilabel Classification: Problem Analysis, Metrics and Techniques*. Springer Publishing Company, Incorporated, 1st ed., 2016.

[4] P.-N. Tan, M. Steinbach, A. Karpatne, and V. Kumar, *Introduction to Data Mining (2nd Edition)*. Pearson, 2nd ed., 2018.

[5] A.-G. Alvar, D.-P. Jose-Francisco., J. Juan, and G.-O. Cesar, "Study of data transformation techniques for adapting single-label prototype selection algorithms to multi-label learning," *Elsevier*, vol. 109, pp. 114–130, 2018.

[6] L. Rokach and O. Maimon, *Data Mining With Decision Trees: Theory and Applications*. USA: World Scientific Publishing Co., Inc., 2nd ed., 2014.

[7] S. Haykin, *Neural Networks: A Comprehensive Foundation*. USA: Prentice Hall PTR, 2nd ed., 1998.

[8] G. P. Zhang, "Neural networks for classification: A survey," *Trans. Sys. Man Cyber Part C*, vol. 30, pp. 451–462, Nov. 2000.

[9] P. Domingos and M. Pazzani, "On the optimality of the simple bayesian classifier under zero-one loss," *Mach. Learn.*, vol. 29, pp. 103–130, Nov. 1997.

[10] H. Zhang, "The optimality of naive bayes," in *Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference, Miami Beach, Florida, USA* (V. Barr and Z. Markov, eds.), pp. 562–567, AAAI Press, 2004.

[11] F. Thabtah, "A review of associative classification mining," *Knowl. Eng. Rev.*, vol. 22, pp. 37–65, Mar. 2007.

[12] S. Ougiaroglou, *Algorithms and Techniques for Efficient and Effective Nearest Neighbours Classification*. PhD thesis, University of Macedonia, 2014.

[13] M. Zhang and Z. Zhou, "A k-nearest neighbor based algorithm for multi-label classification," *2005 IEEE International Conference on Granular Computing*, vol. 2, pp. 718–721 Vol. 2, 2005.

[14] L. Tao and O. Mitsunori, "Detecting emotion in music.," *Johns Hopkins*, 2003.

[15] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Learning multi-label scene classification," *Pattern Recognition*, vol. 37, no. 9, pp. 1757–1771, 2004.

[16] O. Harrison, "Machine learning basics with the k-nearest neighbors algorithm."

[17] S. Ougiaroglou, A. Nanopoulos, A. Papadopoulos, Y. Manolopoulos, and T. Welzer-Druzovec, "Adaptive k-nearest-neighbor classification using a dynamic number of nearest neighbors," in *Advances in Databases and Information Systems* (Y. Ioannidis, B. Novikov, and B. Rachev, eds.), vol. 4690 of *Lecture Notes in Computer Science*, pp. 66–82, Springer Berlin Heidelberg, 2007.

[18] S. Ougiaroglou, G. Evangelidis, and K. I. Diamantaras, "Dynamic k-nn classification based on region homogeneity," in *New Trends in Databases and Information Systems* (J. Darmont, B. Novikov, and R. Wrembel, eds.), (Cham), pp. 27–37, Springer International Publishing, 2020.

[19] H. Mark, F. Eibe, H. Geoffrey, P. Bernhard, R. Peter, and H. W. Ian, "The weka data mining software: An update.," *SIGKDD Explor. Newsl.*, vol. 11, pp. 10–18, 2009.

[20] D. Elena and D. Michel M., *Encyclopedia of Distances*. Springer,Berlin,Heidelberg, 2009.

[21] K. Sawsan, A. Fahed, D. Tierry, and T. Kifah, "Editing training data for multi-label classification with the k-nearest neighbor rule)," *Spinger Link*, vol. 19, pp. 145–161, 2016.

[22] G. Ougiaroglou, Stefanosand Evangelidis, "Rhc: non-parametric cluster-based data reduction for efficient k-nn classification," *Pattern Analysis and Applications*, vol. 19, no. 1, pp. 93–109, 2014.

[23] T. Grigorios and K. Ioannis, "Multi-label classification: An overview," *International Journal of Data Warehousing and Mining*, Thessaloniki, 2009.

[24] R. Jesse, H. Geoffrey, B. Albert, and P. Bernhard, "Streaming multi-label classification," *JMLR.org*, vol. 17, pp. 19–25, 2011.

[25] J. Read, *Scalable Multi-label Classification*. PhD thesis, University of Waikato, Department of Computer Science, 2010.

[26] T. Grigorios and K. Ioannis, "Multi-label classification: An overview," *International Journal of Data Warehousing and Mining*, vol. 3, pp. 1–13, 2009.

[27] T. Grigorios, I. Katakis, and V. I, "Random k-labelsets for multi-label classification," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, pp. 1079–1089, 2011.

[28] T. Grigorios and Z. Min-Ling, "Learning from multi-label data," *1st International Workshop on Learning from Multi-Label Data (MLD'09)*, 2009.

[29] D. Topourtis, "Development mechanisms automatic categorization legal texts." Hellenic Open University, School of Science and Technology, 2016.

[30] H. Afiontzi, "Evaluation of semantic highlighting tools." Economic University Of Athens,Information Section, 2013.

[31] T. Grigorios, K. Ioannis, and V. . Ioannis, "Mining multi-label data," *Data Mining and Knowledge Discovery Handbook, O. Maimon,L. Rokach (Ed.)*, Springer, 2nd edition, 2010.

[32] E. Spyromitros, G. Tsoumakas, and I. Vlahavas, "An empirical study of lazy multilabel classification algorithms," in *Artificial Intelligence: Theories, Models and Applications* (J. Darzentas, G. A. Vouros, S. Vosinakis, and A. Arnellos, eds.), (Berlin, Heidelberg), pp. 401–406, Springer Berlin Heidelberg, 2008.

[33] A. Clare and R. D. King, "Knowledge discovery in multi-label phenotype data," in *Principles of Data Mining and Knowledge Discovery, 5th European Conference* (L. D. Raedt and A. Siebes, eds.), vol. 2168 of *Lecture Notes in Computer Science*, pp. 42–53, Springer, 2001.

[34] R. E. Schapire and Y. Singer, "BoosTexter: A Boosting-based System for Text Categorization," *Machine Learning*, vol. 39, no. 2/3, pp. 135–168, 2000.

[35] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[36] X. Luo and A. N. Zincir-Heywood, "Evaluation of two systems on multi-class multi-label document classification," in *Foundations of Intelligent Systems* (M.-S. Hacid, N. V. Murray, Z. W. Raś, and S. Tsumoto, eds.), (Berlin, Heidelberg), pp. 161–169, Springer Berlin Heidelberg, 2005.

[37] A. McCallum, "Multi-label text classification with a mixture model trained by em," in *AAAI'99 Workshop on Text Learning*, 1999.

[38] A. Elisseeff and J. Weston, "A kernel method for multi-labelled classification," in *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]* (T. G. Dietterich, S. Becker, and Z. Ghahramani, eds.), pp. 681–687, MIT Press, 2001.

[39] S. Godbole and S. Sarawagi, "Discriminative methods for multi-labeled classification," in *Advances in Knowledge Discovery and Data Mining* (H. Dai, R. Srikant, and C. Zhang, eds.), (Berlin, Heidelberg), pp. 22–30, Springer Berlin Heidelberg, 2004.

[40] D. H. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, no. 2, pp. 241–259, 1992.

[41] F. A. Thabtah, P. Cowling, and Yonghong Peng, "Mmac: a new multi-class, multi-label associative classification approach," in *Fourth IEEE International Conference on Data Mining (ICDM'04)*, pp. 217–224, 2004.

[42] Y. Yang, "An evaluation of statistical approaches to text categorization," *Information Retrieval*, vol. 1, pp. 69–90, 1999.

[43] G. Tsoumakas, I. Katakis, and I. Vlahavas, *Mining Multi-label Data*, pp. 667–685. Boston, MA: Springer US, 2010.

[44] I. Jolliffe, *Principal component analysis*. Springer series in statistics, Springer,New York,NY, 2002.

[45] I. Dabbura, "K-means clustering: Algorithm, applications, evaluation methods, and drawbacks."

[46] C.-Z. Jorge, V.-M. Jose J, and R.-J. Juan R, "Improving knn multi-label classification in prototype selection scenarios using class proposals," *Pattern Recognition*, vol. 48, pp. 1608–1622, Spain,2014.

[47] A.-G. Álvar, D.-P. José-Francisco, R. Juan J., and G.-O. César, "Study of data transformation techniques for adapting single-label prototype selection algorithms to multi-label learning," *International Journal of Data Warehousing and Mining*, vol. 109, pp. 114–130, Spain,2018.

[48] E. Leyva, A. González, and R. Pérez, "Three new instance selection methods based on local sets: A comparative study with several approaches from a bi-objective perspective," *Pattern Recognition*, vol. 48, no. 4, pp. 1523–1537, 2015.

[49] H. Brighton and C. Mellish, "On the consistency of information filters for lazy learning algorithms," in *Principles of Data Mining and Knowledge Discovery* (J. M. Żytkow and J. Rauch, eds.), (Berlin, Heidelberg), pp. 283–288, Springer Berlin Heidelberg, 1999.

[50] A.-G. Álvar, D.-P. José-Francisco, J. R. Juan, and G.-O. César, "Local sets for multi-label instance selection," *International Journal of Data Warehousing and Mining*, vol. 68, pp. 651–666, Spain,2018.

[51] M.-L. Zhang and Z.-H. Zhou, "Ml-knn: A lazy learning approach to multi-label learning," *Pattern Recognition*, vol. 40, no. 7, pp. 2038–2048, 2007.

[52] W. Cheng and E. Hüllermeier, "Combining instance-based learning and logistic regression for multilabel classification," *Machine Learning*, vol. 76, pp. 211–225, 09 2009.

[53] N. Ingela, H. H. Yanio, and M. N. Vladimir, *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. Springer, Berlin, Heidelberg, 2019.

[54] K. Sawsan, A. Fahed, D. Thierry, and T. Kifah, "Editing training data for multi-label classification with the k-nearest neighbor rule," *Springer-Verlag*, vol. 19, pp. 145–161, London, 2015.

[55] T. Denœux, Z. Younes, and F. Abdallah, "Representing uncertainty on set-valued variables using belief functions," *Artificial Intelligence*, vol. 174, no. 7, pp. 479–499, 2010.

[56] A. Elisseeff and J. Weston, "A kernel method for multi-labelled classification," in *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, NIPS'01, (Cambridge, MA, USA), p. 681–687, MIT Press, 2001.

[57] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[58] Jose M. Moyano, "Multi-label classification dataset repository," 2020.

[59] B. Forrest, H. Yonghong, R. Raviv, E. Konstantinos, L. Zhong, and C. William, "New methods for acoustic classification of multiple simultaneous bird species in a noisy enviroment," *2013 IEEE International workshop on machine learning for signal processing*, 2013,Southampton, UK.

[60] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 5, pp. 293–302, 2002.

[61] B. Logan, "Mel frequency cepstral coefficients for music modeling," in *In International Symposium on Music Information Retrieval*, 2000.

[62] D. Yang and W.-S. Lee, "Disambiguating music emotion using software agents.," in *ISMIR*, 2004.

[63] T. Konstantinos, T. Grigorios, K. George, and V. Ioannis, "Multi-label classification of music by emotion," *EURASIP Journal on Audio, Speech, and Music Processing volume*, 2011.

[64] B. Hendrik, D. Sašo, and G. Jasna, "Multi-label classification of music by emotion," *Simultaneous Prediction of Multiple Chemical Parameters of River Water Quality with TILDE,Principles of Data Mining and Knowledge Discovery*, vol. 1704, 1999,Berlin, Heidelberg.

[65] Z. Zhou and M. Zhang, "Multi-instance multi-label learning with application to scene classification," in *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 4-7, 2006* (B. Schölkopf, J. C. Platt, and T. Hofmann, eds.), pp. 1609–1616, MIT Press, 2006.

[66] N. Gulisong and K. Abbas, "Comparative evaluation of multi-label classification methods," *2012 9th International Conference on Fuzzy Systems and Knowledge Discovery*, 2012,Sichuan,China.

[67] B. Forrest, H. Yonghong, R. Raviv, E. Konstantinos, L. Zhong, and et, "New methods for acoustic classification of multiple simultaneous bird species in a noisy environment," *2013 IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2013,Southampton, UK.

[68] S. Huan, L. Guo-Zheng, L. GuoPing, and W. YiQin, "Symptom selection for multi-label data of inquiry diagnosis in traditional chinese medicine," *Science China Information Sciences volume*, vol. 56, pp. 1–13, 2012.

[69] Z. Min-Ling and Z. Zhi-Hua, "Multi-label learning by instance differentiation.," *Conference: Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence*, pp. 669–674, 2007,Vancouver, British Columbia, Canada.

[70] S. Cees G. M., W. Marcel, V. G. Jan C, G. Jan Mark, and S. Arnold W M, "The challenge problem for automated detection of 101 semantic concepts in multimedia," *MM '06: Proceedings of the 14th ACM international conference on Multimedia*, pp. 421–430, 2006.

[71] Y. Yang, "A study on thresholding strategies for text categorization," *Acm.org*, Pittsburgh 2001.

Κώδικας Προτεινόμενων Αλγορίθμων και Λοιπών Τεχνικών

Algorithm 1: Αλγόριθμος Κατηγοριοποίησης Πολλαπλών Ετικετών MKNN1

```
# -*- coding: utf-8 -*-
"""
Created on Fri Oct 16 14:28:44 2020

@author: pit filippakis
"""
import math
import operator
import threading
import matplotlib.pyplot as plt
import numpy
import numpy as np
import unicodecsv
from sklearn.metrics import hamming_loss
from time import time
start = time()#metraei o xronos
# getdata() function definition
def getdata(filename):
    with open(filename, 'rb') as f:
        reader = unicodecsv.reader(f)
        return list(reader)


# **edo allazo kayhe fora ta stoixeia tou dataset**
number_of_classes = 101  # arithmos klaseon ??
number_of_attr = 40  # arithmos xaraktiristikon ??
number_of_colums = 141  # arithmos stilon ??
K = 102 #arithmos kontinoteron gitonon  ??


def euclideanDist(x, xi):
    d = 0.0
    for i in range(len(x) - number_of_classes):
        d += pow((float(x[i]) - float(xi[i])), 2)
    d = math.sqrt(d)
    return d


# KNN prediction and model training
def knn_predict(test_data, train_data, k_value):
    knn_new = []
    results = []

    for i in test_data:
        eu_Distance = []
        knn = []

```

```
48
49          for j in train_data:
50              eu_dist = euclideanDist(i, j)
51              eu_Distance.append((*j[number_of_attr:
52              (number_of_attr + number_of_classes)],eu_dist))
53              eu_Distance.sort(key=operator.itemgetter(
54                  number_of_colums - number_of_attr))
55              knn = eu_Distance[
56                    :k_value]
57          knn_new.append(knn)
58          print(knn_new)
59      for i, a in enumerate(knn_new):
60          classes = [
61                        0] * number_of_classes
62          found = False
63          for k, b in enumerate(a):
64              if found:
65                  break
66              for index in range(
67                      number_of_classes):
68                  if b[
69                      index] == '1':
70                      classes[index] += 1
71
72                  if classes[
73                      index] == 2:
74                      found = True
75                      break
76
77          if found:
78              results.append(get_classes(classes))
79
80      return {'knn_new': knn_new, 'results': results}
81
82
83  def get_classes(results):
84      result_tuple = []
85      for index, result in enumerate(results):
86          if result > 0:
87              result_tuple.append(index)
88
89      return result_tuple
90
91
92  def formalize_hamming_loss(hresults):
93      formalized_class = []
94      for k, result in enumerate(hresults):
95          formalized_class.append([0] * number_of_classes)
```

```
        for j in result:
            formalized_class[k][j] = 1

    return formalized_class


def calc_hamming_loss(test_file, train_file, index):
    train_dataset = getdata(test_file)

    test_dataset = getdata(train_file)

    test_data_new = []
    knn_results = knn_predict(test_dataset, train_dataset, K)

    for i in test_dataset:
        test_data_new.append(i[number_of_attr:])
    test_data_arr = numpy.array(test_data_new)
    test_data_list = [[int(float(j)) for j in i] for i in
       test_data_arr]

    if not knn_results['results']:
        return None
    else:
        hl = hamming_loss(np.array(formalize_hamming_loss
        (knn_results['results'])), test_data_list)
        total_results.insert(index, hl)


threads = []
total_results = []
threads.append(threading.Thread(target=calc_hamming_loss,
 args=('mediamill_tr1.csv', 'mediamill_ts1.csv', 1)))
threads.append(threading.Thread(target=calc_hamming_loss,
args=('mediamill_tr2.csv', 'mediamill_ts2.csv', 2)))
threads.append(threading.Thread(target=calc_hamming_loss,
args=('mediamill_tr3.csv', 'mediamill_ts3.csv', 3)))
threads.append(threading.Thread(target=calc_hamming_loss,
 args=('mediamill_tr4.csv', 'mediamill_ts4.csv', 4)))
threads.append(threading.Thread(target=calc_hamming_loss,
 args=('mediamill_tr5.csv', 'mediamill_ts5.csv', 5)))

for t in threads:
    t.start()
    t.join()

print("Hamming Loss: ")
print(total_results)

```

```
print("Total Hamming Loss: ")
hl_sum = 0
for r in total_results:
    hl_sum += r

average_hamming_loss = hl_sum/5
print(average_hamming_loss)

#---CHART-------------------------------------------------------------------------------
# x-coordinates of left sides of bars
left = [1, 2, 3, 4, 5, 6]

# heights of bars
height = total_results
height.append(average_hamming_loss)

# labels for bars
tick_label = ['HL_1', 'HL_2', 'HL_3', 'HL_4', 'HL_5','Average HL']

# plotting a bar chart
plt.bar(left, height, tick_label = tick_label,
        width = 0.8, color = ['red',
            'green','Yellow','Blue','Orange','cyan'])

# naming the x-axis
plt.xlabel('Pairs of 5-folds')
# naming the y-axis
plt.ylabel('Accurancy of Hamming Loss')
# plot title
plt.title('Hamming Loss chart!')

# function to show the plot
plt.show()
print(f'Time taken to run: {time()/60 - start/60} minutes')
```

Algorithm 2: Αλγόριθμος Κατηγοριοποίησης Πολλαπλών Ετικετών MKNN2

```
# -*- coding: utf-8 -*-
"""
Created on Fri Oct 16 14:28:44 2020

@author: pit filippakis
"""
import math
import operator
import threading
import matplotlib.pyplot as plt
import numpy
import numpy as np
import unicodecsv
from sklearn.metrics import hamming_loss
from time import time

start = time()  # metraei o xronos


# getdata() function definition
def getdata(filename):
    with open(filename, 'rb') as f:
        reader = unicodecsv.reader(f)
        return list(reader)


# **edo allazo kayhe fora ta stoixeia tou dataset***
number_of_classes = 6  # arithmos klaseon ??????????????????
number_of_attr = 294 # arithmos xaraktiristikon ???????????????
number_of_colums = 300  # arithmos stilon ????????????????????????
K = 1  # arithmos kontinoteron gitonon
    ????????????????????????????????


def euclideanDist(x, xi):
    d = 0.0
    for i in range(len(x) - number_of_classes):
        d += pow((float(x[i]) - float(xi[i])), 2)
    d = math.sqrt(d)
    return d


# KNN prediction and model training
def knn_predict(test_data, train_data, k_value):
    knn_new = []  # lista knn
    results = []

```

```
47      for i in test_data:
48          eu_Distance = []  # lista apothikeuei euklidies apostaseis
49          knn = []  # lista apothikeuo klasi kai apostasi gia kathe
                instance
50
51          # bad = 0
52          for j in train_data:
53              eu_dist = euclideanDist(i, j)
54
55              eu_Distance.append((*j[number_of_attr:
56              (number_of_attr + number_of_classes)],eu_dist))
57              eu_Distance.sort(key=operator.itemgetter(
58                  number_of_colums - number_of_attr))
59              knn = eu_Distance[
60                    :k_value]
61          knn_new.append(knn)
62
63
64      for i, a in enumerate(knn_new):
65          classes = [0] * number_of_classes
66          for k, b in enumerate(a):
67              for index in range(
68                      number_of_classes):
69                  if b[
70                      index] == '1':
71                      classes[index] += 1
72
73          results.append(get_classes(classes))
74
75      return {'knn_new': knn_new, 'results': results}
76
77
78  def get_classes(results):
79      result_tuple = []
80      for index, result in enumerate(results):
81          if result > 0:
82              result_tuple.append(index)
83
84      return result_tuple
85
86
87  def formalize_hamming_loss(hresults):
88      formalized_class = []
89      for k, result in enumerate(hresults):
90          formalized_class.append([0] * number_of_classes)
91          for j in result:
92              formalized_class[k][j] = 1
93
```

```
94      return formalized_class
95
96
97  def calc_hamming_loss(test_file, train_file, index):
98      train_dataset = getdata(test_file)
99
100     test_dataset = getdata(train_file)
101
102     test_data_new = []
103     knn_results = knn_predict(test_dataset, train_dataset, K)
104
105     for i in test_dataset:
106         test_data_new.append(i[number_of_attr:])
107     test_data_arr = numpy.array(test_data_new)  # kane ti lista
            pinaka
108     test_data_list = [[int(float(j)) for j in i] for i in
            test_data_arr]
109
110     if not knn_results['results']:
111         return None
112     else:
113         hl = hamming_loss(np.array(formalize_hamming_loss
114         (knn_results['results'])),test_data_list)
115         total_results.insert(index, hl)
116
117
118 threads = []  # lista me threads
119 total_results = []  # lista me apotelesmata tou hamming loss
120 total_results = []#lista me apotelesmata tou hamming loss
121 threads.append(threading.Thread(target=calc_hamming_loss,
122  args=('scene_norm_tr1.csv', 'scene_norm_ts1.csv', 1)))
123 threads.append(threading.Thread(target=calc_hamming_loss,
124  args=('scene_norm_tr2.csv', 'scene_norm_ts2.csv', 2)))
125 threads.append(threading.Thread(target=calc_hamming_loss,
126  args=('scene_norm_tr3.csv', 'scene_norm_ts3.csv', 3)))
127 threads.append(threading.Thread(target=calc_hamming_loss,
128  args=('scene_norm_tr4.csv', 'scene_norm_ts4.csv', 4)))
129 threads.append(threading.Thread(target=calc_hamming_loss,
130  args=('scene_norm_tr5.csv', 'scene_norm_ts5.csv', 5)))
131
132 for t in threads:  # jekinoun ta threads
133     t.start()
134     t.join()  # sinexizei o kodikas afou teleiosoun ola ta threads
135
136 print("Hamming Loss: ")
137 print(total_results)  # print tis listas me hamming loss
138
139 print("Total Hamming Loss: ")
```

```
hl_sum = 0
for r in total_results:  # athroisma ton hamming loss
    hl_sum += r

average_hamming_loss = hl_sum / 5  # mesos oros ton hamming loss
print(average_hamming_loss)


#    ---CHART----------------------------------------------------------------------------
# x-coordinates of left sides of bars
left = [1, 2, 3, 4, 5, 6]

# heights of bars
height = total_results
height.append(average_hamming_loss)

# labels for bars
tick_label = ['HL_1', 'HL_2', 'HL_3', 'HL_4', 'HL_5', 'Average HL']

# plotting a bar chart
plt.bar(left, height, tick_label=tick_label,
        width=0.8, color=['red', 'green', 'Yellow', 'Blue',
           'Orange', 'cyan'])

# naming the x-axis
plt.xlabel('Pairs of 5-folds')
# naming the y-axis
plt.ylabel('Accurancy of Hamming Loss')
# plot title
plt.title('Hamming Loss chart!')

# function to show the plot
plt.show()

print(f'Time taken to run: {time() / 60 - start / 60} minutes')
```

Algorithm 3: Αλγόριθμος Κατηγοριοποίησης KNN με Μετασχηματισμό Δυαδικής Συνάφειας (BR-KNN)

```
# Importing the libraries

#import matplotlib.pyplot as plt
import pandas as pd
import matplotlib.pyplot as plt
import sklearn.metrics as metrics
from skmultilearn.problem_transform import BinaryRelevance
from sklearn.neighbors import KNeighborsClassifier
from time import time

start = time()  # metraei o xronos

#metablhtes gia allagi----------------------------------------
number_of_attr = 120 # arithmos xaraktiristikon ???????????????
number_of_colums = 221  # arithmos stilon ??????????????????????
k=1 #metabliti K gia KNN?????????????????????????????????????
#1o
    DATASET-------------------------------------------------------------
# Importing the dataset

dataset = pd.read_csv('mediamill_tr1.csv', header=None)

X_train= dataset.iloc[:,80:number_of_attr].values
y_train = dataset.iloc[:,number_of_attr:number_of_colums].values

dataset_test = pd.read_csv('mediamill_ts1.csv', header=None)

X_test= dataset_test.iloc[:,80:number_of_attr].values
y_test = dataset_test.iloc[:,number_of_attr:number_of_colums].values

print("x_train_1:\n", X_train)
print("y_train_1:\n", y_train)
print("x_test_1:\n", X_test)
print("y_test_1:\n", y_test)

#classifier with binary relevance and Knn----

classifier = BinaryRelevance(
    classifier = KNeighborsClassifier(k)
)

classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)
#-------Hamming loss----------------------------
br_hamm_1=metrics.hamming_loss(y_test,y_pred)

```

```
#CLEAR DATA-------------------------------------

del X_train
del y_train
del X_test
del y_test
del y_pred

#2o DATASET-----------------------------------
# Importing the dataset

dataset = pd.read_csv('mediamill_tr2.csv', header=None)

X_train= dataset.iloc[:,80:number_of_attr].values
y_train = dataset.iloc[:,number_of_attr:number_of_colums].values

dataset_test = pd.read_csv('mediamill_ts2.csv', header=None)

X_test= dataset_test.iloc[:,80:number_of_attr].values
y_test = dataset_test.iloc[:,number_of_attr:number_of_colums].values

print("x_train_2:\n", X_train)
print("y_train_2:\n", y_train)
print("x_test_2:\n", X_test)
print("y_test_2:\n", y_test)


#Binary relevance with KNN
classifier = BinaryRelevance(
    classifier = KNeighborsClassifier(k)
)
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)

#-------Hamming loss----------------------------
br_hamm_2=metrics.hamming_loss(y_test,y_pred)


#CLEAR DATA-------------------------------------
del X_train
del y_train
del X_test
del y_test
del y_pred

#3o DATASET-------------------------------------
# Importing the dataset

```

```
dataset = pd.read_csv('mediamill_tr3.csv', header=None)

X_train= dataset.iloc[:,80:number_of_attr].values
y_train = dataset.iloc[:,number_of_attr:number_of_colums].values

dataset_test = pd.read_csv('mediamill_ts3.csv', header=None)

X_test= dataset_test.iloc[:,80:number_of_attr].values
y_test = dataset_test.iloc[:,number_of_attr:number_of_colums].values

print("x_train_3:\n", X_train)
print("y_train_3:\n", y_train)
print("x_test_3:\n", X_test)
print("y_test_3:\n", y_test)


#Binary relevance with KNN
classifier = BinaryRelevance(
    classifier = KNeighborsClassifier(k)
)
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)

#-------Hamming loss-----------------------------
br_hamm_3=metrics.hamming_loss(y_test,y_pred)


#CLEAR DATA-------------------------------------
del X_train
del y_train
del X_test
del y_test
del y_pred

#4o DATASET-------------------------------------
# Importing the dataset

dataset = pd.read_csv('mediamill_tr4.csv', header=None)

X_train= dataset.iloc[:,80:number_of_attr].values
y_train = dataset.iloc[:,number_of_attr:number_of_colums].values

dataset_test = pd.read_csv('mediamill_ts4.csv', header=None)

X_test= dataset_test.iloc[:,80:number_of_attr].values
y_test = dataset_test.iloc[:,number_of_attr:number_of_colums].values

print("x_train_4:\n", X_train)
```

```
print("y_train_4:\n", y_train)
print("x_test_4:\n", X_test)
print("y_test_4:\n", y_test)


#Binary relevance with KNN
classifier = BinaryRelevance(
    classifier = KNeighborsClassifier(k)
)
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)

#-------Hamming loss-----------------------------
br_hamm_4=metrics.hamming_loss(y_test,y_pred)


#CLEAR DATA-------------------------------------
del X_train
del y_train
del X_test
del y_test
del y_pred

#5o DATASET-------------------------------------
# Importing the dataset

dataset = pd.read_csv('mediamill_tr5.csv', header=None)

X_train= dataset.iloc[:,80:number_of_attr].values
y_train = dataset.iloc[:,number_of_attr:number_of_colums].values

dataset_test = pd.read_csv('mediamill_ts5.csv', header=None)

X_test= dataset_test.iloc[:,80:number_of_attr].values
y_test = dataset_test.iloc[:,number_of_attr:number_of_colums].values

print("x_train_5:\n", X_train)
print("y_train_5:\n", y_train)
print("x_test_5:\n", X_test)
print("y_test_5:\n", y_test)


#Binary relevance with KNN
classifier = BinaryRelevance(
    classifier = KNeighborsClassifier(k)
)
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)
```

```
190
191 #-------Hamming loss-----------------------------
192 br_hamm_5=metrics.hamming_loss(y_test,y_pred)
193
194
195 #CLEAR DATA-------------------------------------
196 del X_train
197 del y_train
198 del X_test
199 del y_test
200 del y_pred
201
202 #Total Hamming Loss-----------------------------
203 average_hamming_loss=(br_hamm_1 + br_hamm_2 + br_hamm_3
204 + br_hamm_4 + br_hamm_5)/5
205
206 #---CHART----------------------------------------
207 # x-coordinates of left sides of bars
208 left = [1, 2, 3, 4, 5, 6]
209
210 # heights of bars
211 height = [br_hamm_1, br_hamm_2, br_hamm_3, br_hamm_4,
212  br_hamm_5, average_hamming_loss]
213
214 # labels for bars
215 tick_label = ['HL_1', 'HL_2', 'HL_3', 'HL_4', 'HL_5','Average HL']
216
217 # plotting a bar chart
218 plt.bar(left, height, tick_label = tick_label,
219         width = 0.8, color = ['red',
              'green','Yellow','Blue','Orange','cyan'])
220
221 # naming the x-axis
222 plt.xlabel('Pairs of 5-folds')
223 # naming the y-axis
224 plt.ylabel('Accurancy of Hamming Loss')
225 # plot title
226 plt.title('Hamming Loss chart!')
227
228 # function to show the plot
229 plt.show()
230 print('Binary Relevance Hamming Loss_1:',round(br_hamm_1,10))
231 print('Binary Relevance Hamming Loss_2:',round(br_hamm_2,10))
232 print('Binary Relevance Hamming Loss_3:',round(br_hamm_3,10))
233 print('Binary Relevance Hamming Loss_4:',round(br_hamm_4,10))
234 print('Binary Relevance Hamming Loss_5:',round(br_hamm_5,10))
235 print("Average Hamming Loss :",average_hamming_loss)
236
```

```
237 print(f'Time taken to run: {time() / 60 - start / 60} minutes')
```

Algorithm 4: Αλγόριθμος Παραγωγής Προτύπων MRHC1

```
#include <stdlib.h>
#include <fstream>
#include <iostream>
#include <math.h>
#include <string>
#include <cstring>
#include <cstdlib>


struct TrainItem{
   int *classAttr;
   double *attr;
   int attrP1;
   double dist;
};

struct Representative{
   int *labels;
   double *attr;
   double *sumR;
   TrainItem *data;
   int dataCounter;
   bool homogeneous;
   bool noneItemFlag;
};

unsigned long long int kmeansComputations;
int ATTRIBUTES;
int CLASSES;
int *mx;

using namespace std;


int findMaxMX();
void readTrainData(TrainItem[], char[], int);
void readTrainData2(TrainItem[], char[], int);
bool checkIfHomogenous(Representative);
int DistancesComputationKMeans(TrainItem&, Representative[], int,
   int);
void initial_kMeans(TrainItem[], Representative[], int);
void kMeans(Representative, Representative[], int&);
void setInitialClassesCentroid(TrainItem[], Representative[], int);
int countLinesAttrs(char[], int&, int&);


int main(int argc, char *argv[]){
```

```
47    TrainItem *TrD;
48    Representative *Repr;
49    int f, i, j, c, back;
50    int dataNumV;
51    int sumc=0, sumd=0;
52
53    if (argc != 5){
54       cout<<"ERROR. Number of parameters"<<endl;
55       return 1;
56    }
57
58    char *fileName = (char*) malloc( sizeof(char) *
          strlen(argv[1])+2);
59    char *fileName2 = (char*) malloc( sizeof(char) *
          strlen(argv[2])+2);
60
61    strcpy(fileName,argv[1]);
62    strcat(fileName,"X");
63    strcpy(fileName2,argv[2]);
64    strcat(fileName2,"X");
65
66    if (!isdigit(*argv[3])){
67       cout<<"ERROR. Folds number must be numeric"<<endl;
68       return 1;
69    }
70    int FOLDS=atoi(argv[3]);
71    if (FOLDS < 1){
72        cout<<"Error: number of FOLDS"<<endl;
73        return 1;
74    }
75    if (!isdigit(*argv[4])){
76       cout<<"ERROR. labels must be numeric"<<endl;
77       return 1;
78    }
79    CLASSES=atoi(argv[4]);
80
81    if (CLASSES < 1){
82        cout<<"Error: number of labels"<<endl;
83        return 1;
84    }
85
86    for (f=1; f<=FOLDS; f++){
87       cout<<"Folds:"<<f<<endl;
88
89       fileName[strlen(fileName)-1]=f+'0';
90       fileName2[strlen(fileName2)-1]=f+'0';
91
92       if (countLinesAttrs(fileName,dataNumV,ATTRIBUTES)){
```

```
93          cout<<"File "<<fileName<<" does not exist"<<endl;
94          return 1;
95        }
96
97        kmeansComputations=0;
98
99      TrD = new TrainItem[dataNumV];
100       Repr = new Representative[dataNumV];
101
102     for (i=0; i<dataNumV; i++){
103     TrD[i].attr = new double[ATTRIBUTES];
104     TrD[i].classAttr = new int[CLASSES];
105     }
106
107       readTrainData2(TrD, fileName, dataNumV);
108       setInitialClassesCentroid(TrD, Repr, dataNumV);
109       initial_kMeans(TrD, Repr, dataNumV);
110       back=CLASSES;
111
112       ofstream o;
113       o.open(fileName2);
114
115       i=0;
116       c=0;
117
118       while (i < back){
119          if ((!Repr[i].homogeneous) && (Repr[i].dataCounter>1)){
120             kMeans(Repr[i], Repr, back);
121          }
122          else if (Repr[i].homogeneous){
123             if (Repr[i].dataCounter>0){
124                c++;
125                for (j=0; j<ATTRIBUTES; j++){
126                   o<<Repr[i].attr[j]<<",";
127                }
128                for (j=0; j<CLASSES; j++){
129          if (j < CLASSES-1)
130           o<<Repr[i].labels[j]<<",";
131         else
132             o<<Repr[i].labels[j]<<endl;
133          }
134             }
135          }
136          i++;
137       }
138
139       o.close();
140
```

```
141     int j;
142       for (j=0; j<dataNumV; j++){
143       delete TrD[j].classAttr;
144       delete TrD[j].attr;
145     }
146       delete []TrD;
147       for (j=0; j<i; j++){
148       delete[] Repr[j].labels;
149       delete[] Repr[j].attr;
150       delete[] Repr[j].data;
151     }
152       delete []Repr;
153
154
155
156       if (f == 1){
157          cout<<"Classes: "<<CLASSES<<endl;
158          cout<<"Attributes: "<<ATTRIBUTES<<endl;
159          cout<<"Data counter: "<<dataNumV<<endl;
160       }
161       cout<<"Prototypes: "<<c<<endl;
162       cout<<"Distance computations: "<<kmeansComputations<<endl;
163       sumc += c;
164       sumd += kmeansComputations;
165    }
166    cout.precision(3);
167    cout.setf(ios::fixed);
168    cout<<endl;
169    cout<<"Avg. prototypes: "<<(double)sumc/(double)FOLDS<<endl;
170    cout<<"Avg. Reduction rate: "<<(double)(1-((double)sumc/
171    (double)FOLDS/(double)dataNumV))*100<<endl;
172    cout<<"Avg. distance computations:
          "<<(double)sumd/(double)FOLDS<<endl;
173
174    return 0;
175 }
176
177
178 void setInitialClassesCentroid(TrainItem TrD[], Representative R[],
179 int dataNumV){
180    int c, i, j;
181    double **sumAttr;
182
183    mx = new int[CLASSES];
184    sumAttr = new double* [CLASSES];
185
186    for (i=0; i<CLASSES; i++){
187       sumAttr[i] = new double[ATTRIBUTES];
```

```
188       mx[i]=0;
189       for (j=0; j<ATTRIBUTES; j++){
190          sumAttr[i][j]=0;
191       }
192    }
193
194    for (i=0; i<dataNumV; i++){
195     for (c=0; c<CLASSES; c++){
196      if (TrD[i].classAttr[c] == 1){
197       for (j=0; j<ATTRIBUTES; j++){
198         sumAttr[c][j] += TrD[i].attr[j];
199       }
200      }
201      mx[c] += TrD[i].classAttr[c];
202     }
203    }
204
205    for (i=0; i<CLASSES; i++){
206       if (mx[i] > 0){
207          R[i].noneItemFlag = false;
208          R[i].data = new TrainItem[dataNumV];
209          R[i].attr = new double[ATTRIBUTES];
210          R[i].sumR = new double[ATTRIBUTES];
211          for (j=0; j<ATTRIBUTES; j++){
212             R[i].attr[j] = (double)sumAttr[i][j] / (double)mx[i];
213          }
214       }
215       else{
216         R[i].noneItemFlag = true;
217       }
218    }
219
220    delete []sumAttr;
221 }
222
223
224 void kMeans(Representative R, Representative *Repr, int &back){
225    int c, i, j, beg, pos;
226    bool kmf;
227
228    static double **sumAttr;
229    static int *mx2;
230
231    mx2 = new int[CLASSES];
232    sumAttr = new double* [CLASSES];
233
234    beg=back;
235
```

```
236    for (i=0; i<CLASSES; i++){
237       sumAttr[i] = new double[ATTRIBUTES];
238       mx2[i]=0;
239       for (j=0; j<ATTRIBUTES; j++){
240          sumAttr[i][j]=0;
241       }
242    }
243
244    for (i=0; i<R.dataCounter; i++){
245     for (c=0; c<CLASSES; c++){
246      if (R.data[i].classAttr[c] == 1){
247       for (j=0; j<ATTRIBUTES; j++){
248         sumAttr[c][j] += R.data[i].attr[j];
249       }
250      }
251      mx2[c] += R.data[i].classAttr[c];
252     }
253    }
254
255    for (i=0; i<CLASSES; i++){
256
257       if (mx2[i] > 0){
258          Repr[back].noneItemFlag = false;
259          Repr[back].data = new TrainItem[R.dataCounter];
260          Repr[back].attr = new double[ATTRIBUTES];
261          Repr[back].sumR = new double[ATTRIBUTES];
262          for (j=0; j<ATTRIBUTES; j++){
263             Repr[back].attr[j] = (double)sumAttr[i][j] /
                  (double)mx2[i];
264          }
265          back++;
266       }
267    }
268
269
270   delete []sumAttr;
271   delete [] mx2;
272
273    for (i=0; i<R.dataCounter; i++){
274       R.data[i].attrP1=-1;
275    }
276
277    do{
278       kmf=0;
279       for (i=beg; i<back; i++){
280          Repr[i].dataCounter=0;
281          for (j=0; j<ATTRIBUTES; j++){
282             Repr[i].sumR[j]=0;
```

```
283          }
284       }
285
286       for (i=0; i<R.dataCounter; i++){
287          pos = DistancesComputationKMeans(R.data[i], Repr, beg,
                back);
288          if (pos == -1){cout<<"-1"<<endl;continue;}
289
290          Repr[pos].data[Repr[pos].dataCounter] = R.data[i];
291          Repr[pos].dataCounter++;
292          if (R.data[i].attrP1 != pos){
293             R.data[i].attrP1=pos;
294             kmf=1;
295          }
296          for (j=0; j<ATTRIBUTES; j++){
297             Repr[pos].sumR[j]+=(double)R.data[i].attr[j];
298          }
299       }
300
301       for (i=beg; i<back; i++){
302          if (Repr[i].dataCounter>0){
303             for (j=0; j<ATTRIBUTES; j++){
304                Repr[i].attr[j]=((double)Repr[i].sumR[j] /
305          (double)Repr[i].dataCounter);
306             }
307          }
308       }
309    }
310    while (kmf != 0);
311
312    for (i=beg; i<back; i++){
313     Repr[i].labels = new int[CLASSES];
314     for (j = 0; j<CLASSES; j++){
315     Repr[i].labels[j]=0;
316     }
317       Repr[i].homogeneous = checkIfHomogenous(Repr[i]);
318    }
319 }
320
321 void initial_kMeans(TrainItem TrD[], Representative Repr[], int
       dataNumV){
322    bool kmf;
323    int i, j, pos;
324
325    for (i=0; i<dataNumV; i++){
326       TrD[i].attrP1=-1;
327    }
328
```

```
329    do{
330       kmf=0;
331       for (i=0; i<CLASSES; i++){
332          Repr[i].dataCounter=0;
333          if (Repr[i].noneItemFlag){
334             continue;
335          }
336          for (j=0; j<ATTRIBUTES; j++){
337             Repr[i].sumR[j]=0;
338          }
339       }
340
341       for (i=0; i<dataNumV; i++){
342          pos = DistancesComputationKMeans(TrD[i], Repr, 0, CLASSES);
343          Repr[pos].data[Repr[pos].dataCounter] = TrD[i];
344          Repr[pos].dataCounter++;
345          if (TrD[i].attrP1 != pos){
346             TrD[i].attrP1=pos;
347             kmf=1;
348          }
349          for (j=0; j<ATTRIBUTES;j++){
350             Repr[pos].sumR[j]+=TrD[i].attr[j];
351          }
352       }
353
354       for (i=0; i<CLASSES; i++){
355          if (Repr[i].dataCounter>0){
356             for (j=0; j<ATTRIBUTES; j++){
357                Repr[i].attr[j]=((double)Repr[i].sumR[j]/
358          (double)Repr[i].dataCounter);
359             }
360          }
361       }
362
363    } while (kmf != 0);
364
365    for (i=0; i<CLASSES; i++){
366     Repr[i].labels = new int[CLASSES];
367     for (j = 0; j<CLASSES; j++){
368     Repr[i].labels[j]=0;
369     }
370       Repr[i].homogeneous = checkIfHomogenous(Repr[i]);
371    }
372
373 }
374
375 int DistancesComputationKMeans(TrainItem &ti, Representative Repr[],
376 int b, int e){
```

```
377     int pos=-1, k, j, m;
378     float min, u, x;
379     float *dist;
380     bool minInit = false;
381
382     dist = new float[e-b+1];
383
384     for (k=b; k<e; k++){
385        if (Repr[k].noneItemFlag){
386           continue;
387        }
388        kmeansComputations++;
389        x=0;
390        for (j=0; j<ATTRIBUTES; j++){
391           u = (float)ti.attr[j] - (float)Repr[k].attr[j];
392           x=x + u * u;
393        }
394        dist[k-b]=x;
395        if (!minInit){
396           min = dist[k-b];
397           pos=k;
398           minInit = true;
399        }
400
401        if (dist[k-b]<min){
402           min=dist[k-b];
403           pos=k;
404        }
405     }
406     ti.dist = min;
407
408     if (e-b > 2){
409      delete [] dist;
410        return pos;
411     }
412     else {
413        m=0;
414        for (k=0; k<e-b; k++){
415            if (dist[k] == min) {
416               m++;
417            }
418        }
419        delete [] dist;
420        if (m==1){
421           return pos;
422        }
423        else{
424           int n = rand() % 2;
```

```
         if (n == 0)
            return pos;
         else
            return pos+1;
      }
   }
}



void readTrainData2(TrainItem trainData[], char fileName[], int n){
   int i,j;

   ifstream dat;
   dat.open(fileName);

   for (i=0; i<n; i++){
      for (j=0; j<ATTRIBUTES; j++){
         dat>>trainData[i].attr[j];
      }
      for (j=0; j<CLASSES; j++){
    dat>>trainData[i].classAttr[j];
    }
   }
   dat.close();
}

bool checkIfHomogenous(Representative R){
   int i,p;
   if (R.dataCounter <= 1){
    p = findMaxMX();
      R.labels[p] = 1;
      return true;
   }

   int j;
   int *cl = new int[CLASSES];
   for (i = 0; i<CLASSES; i++){
     cl[i] = 0;
   }

   for (i=0; i<R.dataCounter; i++){
     for (j = 0; j<CLASSES; j++){
       cl[j] += R.data[i].classAttr[j];
     }
   }

```

```
473     int plithos = 0;
474
475     for (i = 0; i<CLASSES; i++){
476        if (cl[i] == R.dataCounter){
477        p = i;
478        plithos++;
479      }
480     }
481
482     if (plithos == 0){
483       return false;
484     }
485     if (plithos == 1) {
486      R.labels[p] = 1;
487      return true;
488     }
489
490     p = findMaxMX();
491     R.labels[p] = 1;
492
493     return true;
494 }
495
496 int findMaxMX(){
497     int i, max = mx[0];
498     int p = 0;
499     for (i = 1; i<CLASSES; i++){
500      if (mx[i] > max){
501      max = mx[i];
502      p=i;
503      }
504     }
505     return p;
506 }
507
508 int countLinesAttrs(char fileName[], int &lines, int &attrs){
509     int i, c;
510
511     ifstream dat;
512     dat.open(fileName);
513
514     if (!dat){
515        return 1;
516     }
517
518     string lline;
519
520     getline(dat, lline);
```

```
521     c=0;
522     for (i=0; i<(int)lline.length(); i++){
523        if (lline[i] == '\t'){
524           c++;
525        }
526     }
527     dat.close();
528     attrs = c+1;
529 
530     attrs = attrs - CLASSES;
531 
532     dat.open(fileName);
533     i=0;
534     while (!dat.eof()){
535        getline(dat, lline);
536        i++;
537     }
538 
539     lines = i-1;
540     dat.close();
541     return 0;
542 }
```

Algorithm 5: Αλγόριθμος Παραγωγής Προτύπων MRHC2

```
#include <stdlib.h>
#include <fstream>
#include <iostream>
#include <math.h>
#include <string>
#include <cstring>
#include <cstdlib>


struct TrainItem{
   int *classAttr;
   double *attr;
   int attrP1;
   double dist;
};

struct Representative{
   int *labels;
   double *attr;
   double *sumR;
   TrainItem *data;
   int dataCounter;
   bool homogeneous;
   bool noneItemFlag;
};

unsigned long long int kmeansComputations;
int ATTRIBUTES;
int CLASSES;
int *mx;

using namespace std;

void readTrainData(TrainItem[], char[], int);
void readTrainData2(TrainItem[], char[], int);
bool checkIfHomogenous(Representative);
int DistancesComputationKMeans(TrainItem&, Representative[], int,
    int);
void initial_kMeans(TrainItem[], Representative[], int);
void kMeans(Representative, Representative[], int&);
void setInitialClassesCentroid(TrainItem[], Representative[], int);
int countLinesAttrs(char[], int&, int&);


int main(int argc, char *argv[]){
   TrainItem *TrD;
   Representative *Repr;
```

```
47    int f, i, j, c, back;
48    int dataNumV;
49    int sumc=0, sumd=0;
50
51    if (argc != 5){
52       cout<<"ERROR. Number of parameters"<<endl;
53       return 1;
54    }
55
56    char *fileName = (char*) malloc( sizeof(char) *
          strlen(argv[1])+2);
57    char *fileName2 = (char*) malloc( sizeof(char) *
          strlen(argv[2])+2);
58
59    strcpy(fileName,argv[1]);
60    strcat(fileName,"X");
61    strcpy(fileName2,argv[2]);
62    strcat(fileName2,"X");
63
64    if (!isdigit(*argv[3])){
65       cout<<"ERROR. Folds number must be numeric"<<endl;
66       return 1;
67    }
68    int FOLDS=atoi(argv[3]);
69    if (FOLDS < 1){
70        cout<<"Error: number of FOLDS"<<endl;
71        return 1;
72    }
73    if (!isdigit(*argv[4])){
74       cout<<"ERROR. labels must be numeric"<<endl;
75       return 1;
76    }
77    CLASSES=atoi(argv[4]);
78
79    if (CLASSES < 1){
80        cout<<"Error: number of labels"<<endl;
81        return 1;
82    }
83
84    for (f=1; f<=FOLDS; f++){
85       cout<<"Folds:"<<f<<endl;
86
87       fileName[strlen(fileName)-1]=f+'0';
88       fileName2[strlen(fileName2)-1]=f+'0';
89
90       if (countLinesAttrs(fileName,dataNumV,ATTRIBUTES)){
91          cout<<"File "<<fileName<<" does not exist"<<endl;
92          return 1;
```

```
93        }
94
95        kmeansComputations=0;
96
97      TrD = new TrainItem[dataNumV];
98        Repr = new Representative[dataNumV];
99
100     for (i=0; i<dataNumV; i++){
101     TrD[i].attr = new double[ATTRIBUTES];
102     TrD[i].classAttr = new int[CLASSES];
103     }
104
105       readTrainData2(TrD, fileName, dataNumV);
106       setInitialClassesCentroid(TrD, Repr, dataNumV);
107       initial_kMeans(TrD, Repr, dataNumV);
108       back=CLASSES;
109
110       ofstream o;
111       o.open(fileName2);
112
113       i=0;
114       c=0;
115
116       while (i < back){
117          if ((!Repr[i].homogeneous) && (Repr[i].dataCounter>1)){
118             kMeans(Repr[i], Repr, back);
119          }
120          else if (Repr[i].homogeneous){
121             if (Repr[i].dataCounter>0){
122                c++;
123                for (j=0; j<ATTRIBUTES; j++){
124                   o<<Repr[i].attr[j]<<",";
125                }
126                for (j=0; j<CLASSES; j++){
127          if (j < CLASSES-1)
128           o<<Repr[i].labels[j]<<",";
129         else
130             o<<Repr[i].labels[j]<<endl;
131          }
132             }
133          }
134          i++;
135       }
136
137       o.close();
138
139
140
```

```
141       if (f == 1){
142          cout<<"Classes: "<<CLASSES<<endl;
143          cout<<"Attributes: "<<ATTRIBUTES<<endl;
144          cout<<"Data counter: "<<dataNumV<<endl;
145       }
146       cout<<"Prototypes: "<<c<<endl;
147       cout<<"Distance computations: "<<kmeansComputations<<endl;
148       sumc += c;
149       sumd += kmeansComputations;
150    }
151    cout.precision(3);
152    cout.setf(ios::fixed);
153    cout<<endl;
154    cout<<"Avg. prototypes: "<<(double)sumc/(double)FOLDS<<endl;
155    cout<<"Avg. Reduction rate: "<<(double)(1-((double)sumc/
156    (double)FOLDS/(double)dataNumV))*100<<endl;
157    cout<<"Avg. distance computations:
          "<<(double)sumd/(double)FOLDS<<endl;
158
159    return 0;
160 }
161
162
163 void setInitialClassesCentroid(TrainItem TrD[], Representative R[],
164 int dataNumV){
165    int c, i, j;
166    double **sumAttr;
167
168    mx = new int[CLASSES];
169    sumAttr = new double* [CLASSES];
170
171    for (i=0; i<CLASSES; i++){
172       sumAttr[i] = new double[ATTRIBUTES];
173       mx[i]=0;
174       for (j=0; j<ATTRIBUTES; j++){
175          sumAttr[i][j]=0;
176       }
177    }
178
179    for (i=0; i<dataNumV; i++){
180     for (c=0; c<CLASSES; c++){
181      if (TrD[i].classAttr[c] == 1){
182       for (j=0; j<ATTRIBUTES; j++){
183         sumAttr[c][j] += TrD[i].attr[j];
184       }
185      }
186      mx[c] += TrD[i].classAttr[c];
187     }
```

```
188    }
189
190    for (i=0; i<CLASSES; i++){
191       if (mx[i] > 0){
192          R[i].noneItemFlag = false;
193          R[i].data = new TrainItem[dataNumV];
194          R[i].attr = new double[ATTRIBUTES];
195          R[i].sumR = new double[ATTRIBUTES];
196          for (j=0; j<ATTRIBUTES; j++){
197             R[i].attr[j] = (double)sumAttr[i][j] / (double)mx[i];
198          }
199       }
200       else{
201         R[i].noneItemFlag = true;
202       }
203    }
204
205    delete []sumAttr;
206 }
207
208
209 void kMeans(Representative R, Representative *Repr, int &back){
210    int c, i, j, beg, pos;
211    bool kmf;
212
213    static double **sumAttr;
214    static int *mx2;
215
216    mx2 = new int[CLASSES];
217    sumAttr = new double* [CLASSES];
218
219    beg=back;
220
221    for (i=0; i<CLASSES; i++){
222       sumAttr[i] = new double[ATTRIBUTES];
223       mx2[i]=0;
224       for (j=0; j<ATTRIBUTES; j++){
225          sumAttr[i][j]=0;
226       }
227    }
228
229    for (i=0; i<R.dataCounter; i++){
230     for (c=0; c<CLASSES; c++){
231      if (R.data[i].classAttr[c] == 1){
232       for (j=0; j<ATTRIBUTES; j++){
233         sumAttr[c][j] += R.data[i].attr[j];
234       }
235      }
```

```
236       mx2[c] += R.data[i].classAttr[c];
237      }
238     }
239
240     for (i=0; i<CLASSES; i++){
241
242        if (mx2[i] > 0){
243           Repr[back].noneItemFlag = false;
244           Repr[back].data = new TrainItem[R.dataCounter];
245           Repr[back].attr = new double[ATTRIBUTES];
246           Repr[back].sumR = new double[ATTRIBUTES];
247           for (j=0; j<ATTRIBUTES; j++){
248              Repr[back].attr[j] = (double)sumAttr[i][j] /
                     (double)mx2[i];
249           }
250           back++;
251        }
252     }
253
254
255   delete []sumAttr;
256   delete [] mx2;
257
258     for (i=0; i<R.dataCounter; i++){
259        R.data[i].attrP1=-1;
260     }
261
262     do{
263        kmf=0;
264        for (i=beg; i<back; i++){
265           Repr[i].dataCounter=0;
266           for (j=0; j<ATTRIBUTES; j++){
267              Repr[i].sumR[j]=0;
268           }
269        }
270
271        for (i=0; i<R.dataCounter; i++){
272           pos = DistancesComputationKMeans(R.data[i], Repr, beg,
                  back);
273           if (pos == -1){cout<<"-1"<<endl;continue;}
274
275           Repr[pos].data[Repr[pos].dataCounter] = R.data[i];
276           Repr[pos].dataCounter++;
277           if (R.data[i].attrP1 != pos){
278              R.data[i].attrP1=pos;
279              kmf=1;
280           }
281           for (j=0; j<ATTRIBUTES; j++){
```

```
282             Repr[pos].sumR[j]+=(double)R.data[i].attr[j];
283          }
284       }
285
286       for (i=beg; i<back; i++){
287          if (Repr[i].dataCounter>0){
288             for (j=0; j<ATTRIBUTES; j++){
289                Repr[i].attr[j]=((double)Repr[i].sumR[j] /
290          (double)Repr[i].dataCounter);
291             }
292          }
293       }
294    }
295    while (kmf != 0);
296
297    for (i=beg; i<back; i++){
298     Repr[i].labels = new int[CLASSES];
299     for (j = 0; j<CLASSES; j++){
300     Repr[i].labels[j]=0;
301     }
302       Repr[i].homogeneous = checkIfHomogenous(Repr[i]);
303    }
304 }
305
306 void initial_kMeans(TrainItem TrD[], Representative Repr[], int
     dataNumV){
307    bool kmf;
308    int i, j, pos;
309
310    for (i=0; i<dataNumV; i++){
311       TrD[i].attrP1=-1;
312    }
313
314    do{
315       kmf=0;
316       for (i=0; i<CLASSES; i++){
317          Repr[i].dataCounter=0;
318          if (Repr[i].noneItemFlag){
319             continue;
320          }
321          for (j=0; j<ATTRIBUTES; j++){
322             Repr[i].sumR[j]=0;
323          }
324       }
325
326       for (i=0; i<dataNumV; i++){
327          pos = DistancesComputationKMeans(TrD[i], Repr, 0, CLASSES);
328          Repr[pos].data[Repr[pos].dataCounter] = TrD[i];
```

```
329          Repr[pos].dataCounter++;
330          if (TrD[i].attrP1 != pos){
331             TrD[i].attrP1=pos;
332             kmf=1;
333          }
334          for (j=0; j<ATTRIBUTES;j++){
335             Repr[pos].sumR[j]+=TrD[i].attr[j];
336          }
337       }
338
339       for (i=0; i<CLASSES; i++){
340          if (Repr[i].dataCounter>0){
341             for (j=0; j<ATTRIBUTES; j++){
342                Repr[i].attr[j]=((double)Repr[i].sumR[j]/
343          (double)Repr[i].dataCounter);
344             }
345          }
346       }
347
348    } while (kmf != 0);
349
350    for (i=0; i<CLASSES; i++){
351     Repr[i].labels = new int[CLASSES];
352     for (j = 0; j<CLASSES; j++){
353     Repr[i].labels[j]=0;
354     }
355       Repr[i].homogeneous = checkIfHomogenous(Repr[i]);
356    }
357
358 }
359
360 int DistancesComputationKMeans(TrainItem &ti, Representative Repr[],
361 int b, int e){
362    int pos=-1, k, j, m;
363    float min, u, x;
364    float *dist;
365    bool minInit = false;
366
367    dist = new float[e-b+1];
368
369    for (k=b; k<e; k++){
370       if (Repr[k].noneItemFlag){
371          continue;
372       }
373       kmeansComputations++;
374       x=0;
375       for (j=0; j<ATTRIBUTES; j++){
376          u = (float)ti.attr[j] - (float)Repr[k].attr[j];
```

```
377             x=x + u * u;
378          }
379          dist[k-b]=x;
380          if (!minInit){
381             min = dist[k-b];
382             pos=k;
383             minInit = true;
384          }
385
386          if (dist[k-b]<min){
387             min=dist[k-b];
388             pos=k;
389          }
390       }
391       ti.dist = min;
392
393       if (e-b > 2){
394        delete [] dist;
395          return pos;
396       }
397       else {
398          m=0;
399          for (k=0; k<e-b; k++){
400              if (dist[k] == min) {
401                 m++;
402              }
403          }
404          delete [] dist;
405          if (m==1){
406             return pos;
407          }
408          else{
409             int n = rand() % 2;
410             if (n == 0)
411                return pos;
412             else
413                return pos+1;
414          }
415       }
416    }
417
418
419
420
421    void readTrainData2(TrainItem trainData[], char fileName[], int n){
422       int i,j;
423
424       ifstream dat;
```

```
425     dat.open(fileName);
426
427     for (i=0; i<n; i++){
428        for (j=0; j<ATTRIBUTES; j++){
429           dat>>trainData[i].attr[j];
430        }
431        for (j=0; j<CLASSES; j++){
432      dat>>trainData[i].classAttr[j];
433      }
434     }
435     dat.close();
436 }
437
438 bool checkIfHomogenous(Representative R){
439     int i;
440     if (R.dataCounter == 0){
441       return true;
442     }
443
444     int j;
445     int *cl = new int[CLASSES];
446     for (i = 0; i<CLASSES; i++){
447       cl[i] = 0;
448     }
449
450     for (i=0; i<R.dataCounter; i++){
451       for (j = 0; j<CLASSES; j++){
452         cl[j] += R.data[i].classAttr[j];
453       }
454     }
455
456     bool flag = false;
457     for (i = 0; i<CLASSES; i++){
458        if (cl[i] == R.dataCounter){
459        flag = true;
460        break;
461      }
462     }
463
464     if (!flag){
465       return false;
466     }
467
468     if (flag){
469      for (i = 0; i<CLASSES; i++){
470        if (cl[i] > R.dataCounter/2){
471        R.labels[i] = 1;
472        }
```

```
473      }
474    }
475    return true;
476 }
477
478 int countLinesAttrs(char fileName[], int &lines, int &attrs){
479    int i, c;
480
481    ifstream dat;
482    dat.open(fileName);
483
484    if (!dat){
485       return 1;
486    }
487
488    string lline;
489
490    getline(dat, lline);
491    c=0;
492    for (i=0; i<(int)lline.length(); i++){
493       if (lline[i] == '\t'){
494          c++;
495       }
496    }
497    dat.close();
498    attrs = c+1;
499
500    attrs = attrs - CLASSES;
501
502    dat.open(fileName);
503    i=0;
504    while (!dat.eof()){
505       getline(dat, lline);
506       i++;
507    }
508
509    lines = i-1;
510    dat.close();
511    return 0;
512 }
```

Algorithm 6: Τεχνική 5-fold cross validation (CROSS-VALIDATION)

```
#include <stdlib.h>
#include <fstream>
#include <iostream>
#include <math.h>
#include <string>
#include <cstring>
#include <cstdlib>

using namespace std;

struct Item{
   int classAttr;
   float *attr;
};


int ATTRIBUTES;

void readData(Item[], char[], int);
int countLinesAttrs(char[], int&, int&);

int main(int argc, char *argv[]){
    static Item *data;
    int i, n, f, j, inef, tebe, teen, yp;
    ofstream trset;
    ofstream teset;

    if (argc != 3){
      cout<<"ERROR. Number of parameters"<<endl;
      return 1;
    }
    char *fileName = (char*) malloc( sizeof(char) *
       strlen(argv[1])+1);
    char *trainFile = (char*) malloc( sizeof(char) *
       strlen(argv[1])+5);
    char *testFile = (char*) malloc( sizeof(char) *
       strlen(argv[1])+5);

    strcpy(fileName,argv[1]);

    strcpy(trainFile,argv[1]);
    strcpy(testFile,argv[1]);
    strcat(trainFile,"_trX");
    strcat(testFile,"_tsX");

    if (!isdigit(*argv[2])){
       cout<<"ERROR. Folds number must be numeric"<<endl;
```

```
45          return 1;
46      }
47      int FOLDS=atoi(argv[2]);
48
49      if (countLinesAttrs(fileName,n,ATTRIBUTES)){
50        cout<<"File "<<fileName<<" does not exist"<<endl;
51        return 1;
52      }
53
54      data = (Item*) malloc (n * sizeof(Item));
55      readData(data, fileName, n);
56
57      inef = ceil((float)n / (float)FOLDS);
58
59     // cout<<inef<<endl;
60
61      yp = n % FOLDS;
62
63      tebe = 0;
64      teen = inef;
65
66      for (f=1; f<=FOLDS; f++){
67          trainFile[strlen(trainFile)-1]=f+'0';
68          testFile[strlen(testFile)-1]=f+'0';
69          trset.open(trainFile);
70          teset.open(testFile);
71
72          cout<<"Creating files: "<<trainFile<<" "<<testFile<<endl;
73
74
75          for (i=0; i<n; i++){
76             if ((i >= tebe) && (i<teen)){
77                for (j=0; j<ATTRIBUTES; j++){
78                   teset<<data[i].attr[j]<<"\t";
79                }
80                teset<<data[i].classAttr<<endl;
81             }
82             else {
83                for (j=0; j<ATTRIBUTES; j++){
84                   trset<<data[i].attr[j]<<"\t";
85                }
86                trset<<data[i].classAttr<<endl;
87             }
88          }
89
90          tebe += inef;
91          yp--;
92          if (yp == 0){
```

```
93              inef--;
94          }
95
96
97          teen += inef;
98
99
100
101         trset.close();
102         teset.close();
103     }
104     return 0;
105 }
106
107
108 void readData(Item data[], char fileName[], int n){
109    int i,j;
110
111    ifstream dat;
112    dat.open(fileName);
113    bool fl;
114
115    for (i=0; i<n; i++){
116       data[i].attr = new float[ATTRIBUTES];
117       for (j=0; j<ATTRIBUTES; j++){
118          dat>>data[i].attr[j];
119       }
120       dat>>data[i].classAttr ;
121    }
122
123    dat.close();
124 }
125
126 int countLinesAttrs(char fileName[], int &lines, int &attrs){
127    unsigned int i, c;
128
129    ifstream dat;
130    dat.open(fileName);
131
132    if (!dat){
133       return 1;
134    }
135
136    string lline;
137
138    getline(dat, lline);
139    c=0;
140    for (i=0; i<lline.length(); i++){
```

```
141         if (lline[i] == '\t'){
142            c++;
143         }
144      }
145      dat.close();
146      attrs = c;
147
148      dat.open(fileName);
149      i=0;
150      while (!dat.eof()){
151         getline(dat, lline);
152         i++;
153      }
154
155      lines = i-1;
156      dat.close();
157      return 0;
158   }
```

Algorithm 7: Κώδικας Κανονικοποίησης Δεδομένων (NORMALIZATION)

```
#include <stdlib.h>
#include <fstream>
#include <iostream>
#include <math.h>
#include <string>
#include <cstring>
#include <cstdlib>

using namespace std;

struct Item{
   int *classAttr;
   float *attr;
};


int ATTRIBUTES;

void readData(Item[], char[], int,int);
int countLinesAttrs(char[], int&, int&,int);
void maxmin(Item[], float[], float[], int);

int main(int argc, char *argv[]){
    static Item *data;
    float *max, *min;
    int i, j, n;
    ofstream out;

    if (argc != 4){
      cout<<"ERROR. Number of parameters"<<endl;
      return 1;
    }

    char *fileName = argv[1];//arxiko onoma
    char *fileNameOut = argv[2];//teliko arxeio
    int labels=atoi(argv[3]);//arithmos labels

    if (countLinesAttrs(fileName,n,ATTRIBUTES,labels)){
      cout<<"File "<<fileName<<" does not exist"<<endl;
      return 1;
    }

    data = (Item*) malloc (n * sizeof(Item));
    readData(data, fileName, n,labels);

    max = (float*) malloc (ATTRIBUTES * sizeof(float));
    min = (float*) malloc (ATTRIBUTES * sizeof(float));
```

```
48
49      maxmin(data, max, min, n);
50
51      out.open(fileNameOut);
52      for (i=0; i<n; i++){
53         for (j=0; j<ATTRIBUTES; j++){
54            out<<(data[i].attr[j]-min[j]) / (max[j]-min[j])<<"\t";
55         }
56         for(j=0;j<labels;j++){
57           out<<data[i].classAttr[j]<<endl;
58         }
59
60      }
61
62      out.close();
63      return 0;
64  }
65
66  void maxmin(Item data[], float max[], float min[], int n){
67     int i,j;
68
69     for (j=0; j<ATTRIBUTES; j++){
70        max[j] = data[0].attr[j];
71        min[j] = data[0].attr[j];
72
73        for (i=1; i<n; i++){
74           if (data[i].attr[j] > max[j]){
75               max[j] = data[i].attr[j];
76           }
77           if (data[i].attr[j] < min[j]){
78               min[j] = data[i].attr[j];
79           }
80        }
81     }
82  }
83
84
85  void readData(Item data[], char fileName[], int n,int labels){
86     int i,j;
87
88     ifstream dat;
89     dat.open(fileName);
90     bool fl;
91
92     for (i=0; i<n; i++){
93        data[i].attr = new float[ATTRIBUTES];
94        for (j=0; j<ATTRIBUTES; j++){
95           dat>>data[i].attr[j];
```

```
96        }
97        data[i].classAttr=new int[labels];//pinakas labels
98        for(j=0; j<labels;j++){
99          dat>>data[i].classAttr[j] ;
100       }
101
102    }
103
104    dat.close();
105 }
106
107 int countLinesAttrs(char fileName[], int &lines, int &attrs,int
       labels){
108    unsigned int i, c;
109
110    ifstream dat;
111    dat.open(fileName);
112
113    if (!dat){
114       return 1;
115    }
116
117    string lline;
118
119    getline(dat, lline);
120    c=0;
121    for (i=0; i<lline.length(); i++){
122       if (lline[i] == '\t'){
123          c++;
124       }
125    }
126    dat.close();
127    attrs = c-labels+1;
128
129
130    dat.open(fileName);
131    i=0;
132    while (!dat.eof()){
133       getline(dat, lline);
134       i++;
135    }
136
137    lines = i-1;
138    dat.close();
139    return 0;
140 }
```