Μείωση δεδομένων με διαμερισμό του χώρου χρησιμοποιώντας αλγόριθμους **Convex Hull** για την εύρεση των πιο απομακρυσμένων αντικειμένων

Γκιοργκίνης Θωμάς
ΑΜ 134038

Τμήμα Μηχανικών Πληροφορικής ΑΤΕΙΘ
Αλεξάνδρειο Τεχνολογικό Εκπαιδευτικό Ίδρυμα Θεσσαλονίκης

Επίβλεψη
Ουγιάρογλου Στέφανος
Δέρβος Δημήτριος

Φεβρουάριος 2019

Περίληψη

Στο συνεχές εξελισσόμενο επιστημονικό πεδίο της κατηγοριοποίησης δεδομένων, ο αλγόριθμος των εγγύτερων γειτόνων αποτελεί μια σταθερή, αποτελεσματική μέθοδο. Διαθέτει όμως αδυναμίες που τον καθιστούν ακατάλληλο σε ορισμένες περιπτώσεις συνόλων δεδομένων. Τα βασικά μειονεκτήματά του είναι: υψηλό κόστος κατηγοριοποίησης του κάθε αντικειμένου λόγω πολλαπλών υπολογισμών που μεσολαβούν, υψηλές απαιτήσεις σε αποθηκευτικό χώρο και η σχέση εξάρτησης της ακρίβειας των αποτελεσμάτων με την ποιότητα των δεδομένων εκπαίδευσης. Στην προσπάθεια αντιμετώπισης των αδυναμιών του έχουν υλοποιηθεί διάφοροι αλγόριθμοι μείωσης του πλήθους των δεδομένων, που στοχεύουν στην όσο πιο δυνατή ελάττωση του φόρτου επεξεργασίας του κατηγοριοποιητή χωρίς να επηρεάζεται η ακρίβειά του. Ο αλγόριθμος Reduction by Space Partitioning είναι ένας από τους πιο γνωστούς αλγόριθμους prototype generation για την επιτάχυνση instance based κατηγοριοποιητών. Ο RSP3 βασίζεται σε μια επαναληπτική διαδικασία διαχωρισμού του αρχικού συνόλου εκπαίδευσης. Ως κριτήριο για το πιο υποσύνολο θα διαιρεθεί πρώτο, ο RSP3 υιοθετεί το κριτήριο της μεγαλύτερης διαμέτρου. Δηλαδή, το υποσύνολο με τη μεγαλύτερη διάμετρο που ορίζεται από τα δύο πιο απομακρυσμένα αντικείμενα διαιρείται πρώτο. Η διαδικασία συνεχίζεται μέχρις ότου τα υποσύνολα που προκύπτουν είναι ομοιογενή, δηλαδή να περιλαμβάνουν αντικείμενα της ίδια κλάσης. Η αναζήτηση των δυο πιο απομακρυσμένων αντικειμένων κάθε

υποσυνόλου αποτελεί μια διαδικασία μεγάλου κόστους, αφού προϋποθέτει τον υπολογισμό όλων των αποστάσεων μεταξύ των αντικειμένων του κάθε υποσυνόλου. Έτσι, ο RSP3 έχει υψηλό υπολογιστικό κόστος προ-επεξεργασίας. Στα πλαίσια της εργασίας θα μελετηθούν οι αλγόριθμοι υπολογιστικής γεωμετρίας για την εύρεση του Convex Hull του κάθε υποσυνόλου. Το Convex Hull είναι τα αντικείμενα του συνόλου δεδομένων που ορίζουν το περίγραμμα του. Το κίνητρο για την μελέτη αυτών των αλγορίθμων πηγάζει από την εξής παρατήρηση: Αν βρεθούν αυτά τα αντικείμενα που ορίζουν το Convex Hull, η αναζήτηση των πιο απομακρυσμένων αντικειμένων σε κάθε υποσύνολο θα γίνεται γρήγορα αφού δεν θα απαιτείται η εύρεση όλων των πιθανών αποστάσεων στο υποσύνολο αλλά μόνο όλων των πιθανών αποστάσεων μεταξύ των αντικείμενων του Convex Hull. Στο πειραματικό κομμάτι της εργασίας δύο αλγόριθμοι Convex Hull θα υλοποιηθούν και ενσωματωθούν στον RSP3. Επίσης, θα μελετηθεί το κατά πόσο είναι δυνατό, οι αλγόριθμοι εύρεσης του Convex Hull να εκτελεστούν σε σύνολα δεδομένων με πολλές διαστάσεις. Οι νέες παραλλαγές του RSP3 θα δοκιμαστούν σε 15 σύνολα δεδομένων κατηγοριοποίησης πραγματικών δεδομένων και η ταχύτητα τους θα συγκριθεί με τον "συμβατικό" RSP3.

Abstract

In the ever evolving scientific field of data categorization, the nearest neighbors algorithm is a stable, efficient method. However, it has several weaknesses that deem it inappropriate in some cases of data sets. Its main drawbacks are: high cost of categorizing each object due to multiple calculations, high storage requirements, and dependence of the accuracy of the results on the quality of the training data. In order to address its weaknesses, several data reduction algorithms have been implemented, aiming at minimizing the processing burden of the categorizer without affecting its accuracy. The Reduction by Space Partitioning algorithm is one of the most well-known prototype generation algorithms for accelerating instance based categorists. RSP3 is based on a repetitive separation process of the original training set. As the criterion for the most subset will be divided first, RSP3 adopts the larger diameter criterion. That is, the subset with the largest diameter defined by the two most distant objects is divided first. The process continues until the resulting subsets are homogeneous, that is, they include objects of the same class. The search for the two most remote objects of each subset is a costly process, since it requires the calculation of all the distances between the objects of each sub-set. Thus, RSP3 has a high computational pre-processing cost. In the framework of the thesis, computational geometry algorithms will be studied to find the Convex Hull of each subset. Convex

Hull is composed by the objects of the data set that define its contour. The motivation for studying these algorithms stems from the following observation: If these objects that define Convex Hull are found, searching for the most remote objects in each subset will be done quickly since it will not be necessary to calculate all possible distances to the subset but only all possible distances between the objects of Convex Hull. In the experimental part of the thesis, two Convex Hull algorithms will be implemented and incorporated into RSP3. It will also be investigated whether it is possible to use Convex Hull algorithms in multi-dimensional datasets. The new variants of RSP3 will be tested in 15 sets of real data categorization data and their speed will be compared to that of the "conventional" RSP3.

Ευχαριστίες

Για τη διεκπεραίωση της παρούσας πτυχιακής εργασίας, θα ήθελα να ευχαριστήσω αρχικά τους επιβλέποντες, κύριο Στέφανο Ουγιάρογλου και Δημήτριο Δέρβο για τη συνεργασία και την πολύτιμη συμβολή τους στην ολοκλήρωση της, αλλά και τον κύριο Γεώργιο Ευαγγελίδη για το ενδιαφέρον και τις υποδείξεις του. Ο ρόλος τους ήταν καθοριστικός για την ολοκλήρωσή της.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

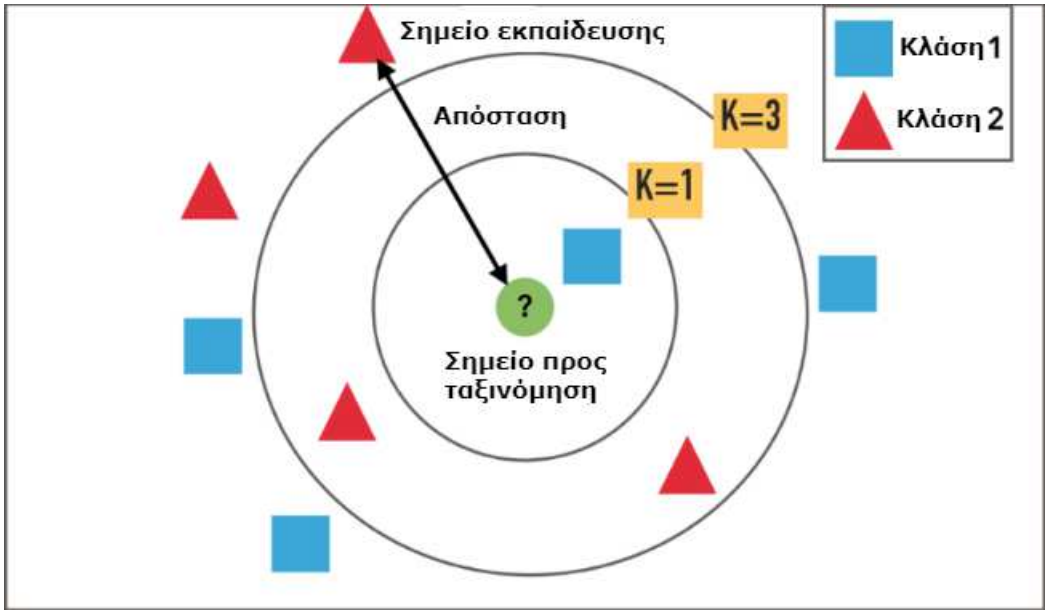
Κεφάλαιο 1

Εισαγωγή

1.1 Ο αλγόριθμος εγγύτερων γειτόνων

Ο ταξινομητής k-Nearest Neighbors (k-NN) [11; 12] είναι ένας αποτελεσματικός και ευρέως χρησιμοποιούμενος αλγόριθμος ταξινόμησης. Είναι απλός και εύκολος στο να εφαρμοστεί, μπορεί να εκμεταλλευτεί σε πολλούς τομείς εφαρμογής και μπορεί να ενσωματωθεί σε πολλά συστήματα.

Δεδομένου ότι ο ταξινομητής k-NN ανήκει στην κατηγορία των lazy ταξινομητών, δεν δημιουργεί κανένα μοντέλο ταξινόμησης. Ο αλγόριθμος χρησιμοποιεί τα δεδομένα εκπαίδευσης όποτε χρειάζεται να ταξινομηθεί ένα νέο στοιχείο. Συγκεκριμένα, ταξινομεί ένα στοιχείο x με αναζήτηση στα διαθέσιμα δεδομένα εκπαίδευσης και με ανάκτηση των πλησιέστερων στοιχείων k (γείτονες) του x σύμφωνα με μια μετρική απόστασης (π.χ. Ευκλείδεια). Στην συνέχεια, το x ανατίθεται στην πιο κοινή κλάση μεταξύ των κλάσεων των εγγεγραμμένων πλησιέστερων γειτόνων που ανακτήθηκαν. Η κλάση αυτή ονομάζεται μείζων κλάση και καθορίζεται μέσω μιας διαδικασίας που είναι γνωστή ως η ψήφος των πλησιέστερων γειτόνων. Να σημειωθεί ότι όταν $k = 1$, ο αλγόριθμος είναι επίσης γνωστός ως ταξινομητής πλησιέστερης γειτονίας (ή κανόνας 1-NN).



Σχήμα 1.1: Διαδικασία ταξινόμησης γειτόνων για $k = 1$ και $k = 3$

Το σχήμα 1.1 απεικονίζει ένα δισδιάστατο παράδειγμα της διαδικασίας ταξινόμησης. Συγκεκριμένα, παρουσιάζει ένα σύνολο δεδομένων με δύο κλάσεις, τετράγωνα και τρίγωνα και ένα στοιχείο που χρειάζεται να ταξινομηθεί σε μία από αυτές. Αν θέσουμε το k να είναι ίσο με ένα, το στοιχείο παίρνει την κλάση του κοντινότερου γείτονα σε αυτό, δηλαδή τετράγωνο (Κλάση 1). Εάν το k είναι ίσο με τρία, έχουμε ψήφο των τριών κοντινότερων γειτόνων με αποτέλεσμα να υπερισχύουν τα τρίγωνα (Κλάση 2). Οι κύκλοι προσομοιώνουν προσεγγιστικά τα όρια του χώρου που περιβάλλουν όλα τα στοιχεία που παίρνουν μέρος στην ψηφοφορία, ανάλογα με την τιμή του k.

Η απόδοση της ταξινόμησης εξαρτάται σίγουρα από την επιλογή της τιμής της παραμέτρου k. Η τιμή του k που επιτυγχάνει την υψηλότερη ακρίβεια εξαρτάται από το σύνολο δεδομένων και ο προσδιορισμός του συνήθως προέρχεται μέσω δαπανηρών εργασιών δοκιμής και σφάλματος (trial and error). Αν και ο προσδιορισμός του k δεν μπορεί να ακολουθήσει κανέναν γενικό κανόνα και το βέλτιστο k μπορεί να είναι τελείως διαφορετικό για διαφορετικά σύνολα δεδομένων, μεγαλύτερες τιμές

k είναι κατάλληλες για σύνολα δεδομένων με θόρυβο αφού εξετάζουν μεγαλύτερες γειτονίες. Ωστόσο, δεν καθορίζουν σαφώς τα όρια μεταξύ διακριτών κλάσεων. Αντίθετα, οι μικρές τιμές του k καθιστούν τον ταξινομητή περισσότερο ευαίσθητο στο θόρυβο. Επομένως, σε περιπτώσεις δεδομένων που περιέχουν θόρυβο, η ταξινόμηση είναι πιθανώς λιγότερο ακριβής. Αξίζει να σημειωθεί ότι ακόμη και η καλύτερη τιμή k μπορεί να μην είναι η βέλτιστη.

Σε περιπτώσεις δυαδικών προβλημάτων ταξινόμησης (σύνολα δεδομένων με δύο κλάσεις), το k πρέπει να έχει περιττή τιμή για την αποφυγή ισοψηφιών (οι δύο κλάσεις παρουσιάζονται με την ίδια συχνότητα) κατά τη διάρκεια των ψηφοφοριών των πλησιέστερων γειτόνων. Σε περιπτώσεις μη δυαδικών προβλημάτων, το k μπορεί να έχει οποιαδήποτε τιμή. Εδώ, πιθανές ισοψηφίες κατά την ψηφοφορία επιλύονται επιλέγοντας είτε μια τυχαία πιο συνηθισμένη κλάση είτε την κλάση του πλησιέστερου γείτονα. Το δημοφιλές λογισμικό Weka [30] και πολλά άλλα εργαλεία λογισμικού εξόρυξης δεδομένων/μηχανικής μάθησης επιλύουν το πρόβλημα επιλέγοντας τυχαία. Εδώ να σημειωθεί ότι στις πειραματικές μελέτες αυτής της εργασίας γίνεται χρήση του k-NN με $k = 1$.

Ένα άλλο σημαντικό ζήτημα είναι η επιλογή της μετρικής που χρησιμοποιείται για τον υπολογισμό των αποστάσεων μεταξύ των στοιχείων. Προφανώς πρέπει να ληφθούν υπ' όψιν οι τύποι δεδομένων του συνόλου δεδομένων (χαρακτηριστικά). Σε περιπτώσεις πραγματικών και/ή ακέραιων χαρακτηριστικών, η Ευκλείδεια απόσταση είναι η κοινώς χρησιμοποιούμενη. Ωστόσο, μπορούν να υιοθετηθούν άλλες μετρήσεις απόστασης (π.χ. Mahalanobis, Manhatan, Minkowski, Chebyshev) [18]. Όλα τα πειράματα της εργασίας διεξάγονται σε σύνολα δεδομένων με πραγματικά και/ή ακέραια χαρακτηριστικά, ως εκ τούτου η επιλογή της Ευκλείδειας απόστασης είναι

η βέλτιστη. Ο τύπος της δίνεται ως:

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

όπου p, q δύο σημεία στο χώρο, d η Ευκλείδεια απόσταση μεταξύ τους και n ο αριθμός των διαστάσεων.

1.2 Τα μειονεκτήματα του k-NN

Παρόλο που ο ταξινομητής k-NN θεωρείται αποτελεσματική μέθοδος, έχει κάποιες αδυναμίες που μπορούν να καταστήσουν τη χρήση του μη αποδοτική. Το πρώτο αδύναμο σημείο είναι το υψηλό υπολογιστικό κόστος. Ο ταξινομητής k-NN πρέπει να υπολογίζει όλες τις αποστάσεις μεταξύ κάθε μη ταξινομημένου στοιχείου και όλων των αντικειμένων που είναι αποθηκευμένα στο σετ εκπαίδευσης. Σε περιπτώσεις μεγάλων συνόλων δεδομένων, αυτό το μειονέκτημα καθιστά τη χρήση του ιδιαίτερα απαιτητική χρονικά και σε ορισμένες περιπτώσεις μια απαγορευτική διαδικασία. Για παράδειγμα, υποθέστε ότι ένα σύστημα ταξινόμησης αποθηκεύει 100.000 είδη εκπαίδευσης. Επιπλέον, υποθέστε ότι το σύστημα πρέπει να ταξινομήσει περίπου 50.000 μη ταξινομημένα αντικείμενα χρησιμοποιώντας τον ταξινομητή k-NN στα δεδομένα εκπαίδευσης. Αυτό σημαίνει ότι το σύστημα πρέπει να υπολογίζει πέντε δισεκατομμύρια αποστάσεις. Αν και σήμερα τα συστήματα είναι εξοπλισμένα με ισχυρούς επεξεργαστές, αυτοί οι υπολογισμοί είναι χρονοβόροι και απαράδεκτοι σε περιπτώσεις περιβαλλόντων χρονικού περιορισμού. Να αναφερθεί ότι, εκτός από το μέγεθος του εκπαιδευτικού συνόλου, το υπολογιστικό κόστος της διαδικασίας ταξινόμησης εξαρτάται επίσης από τα δεδομένα των διαστάσεων. Όσο υψηλότερη είναι η διαστασιολόγηση των δεδομένων, τόσο περισσότεροι υπολογισμοί πραγματοποιούνται για τον υπολογισμό των αποστάσεων.

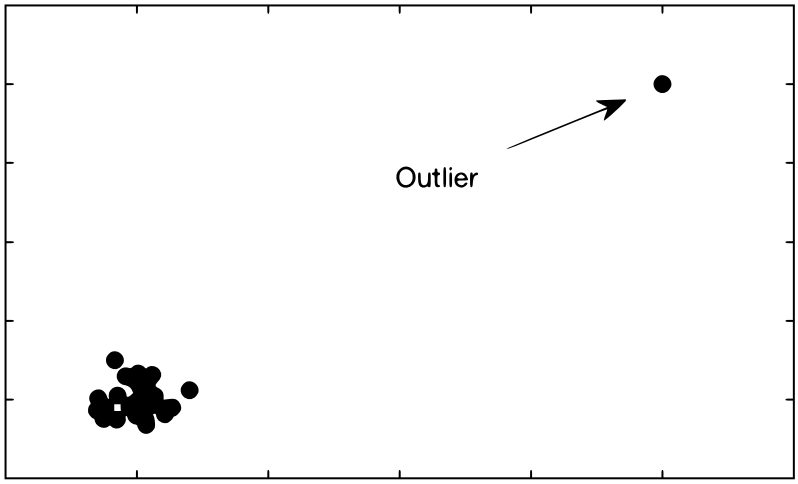
Μια άλλη αδυναμία του ταξινομητή k-NN είναι οι μεγάλες απαιτήσεις αποθήκευσης για τα δεδομένα εκπαίδευσης. Σε αντίθεση με άλλους ταξινομητές που μπορούν να απορρίψουν τα δεδομένα εκπαίδευσης μετά την κατασκευή του μοντέλου ταξινόμησης, ο ταξινομητής k-NN χρειάζεται τα δεδομένα εκπαίδευσης να είναι πάντα διαθέσιμα. Συνεπώς, πρέπει να εκτελεστεί σε συστήματα υπολογιστών εξοπλισμένα με αρκετή κύρια μνήμη για την αποθήκευση των δεδομένων εκπαίδευσης.

Η τελευταία αδυναμία είναι ότι ο ταξινομητής k-NN, όπως και πολλές άλλες μέθοδοι ταξινόμησης, είναι ευαίσθητος στον θόρυβο. Ειδικότερα η ακρίβεια ταξινόμησης εξαρτάται σε μεγάλο βαθμό από την ποιότητα των δεδομένων κατάρτισης. Θόρυβος και λανθασμένα δεδομένα, καθώς και απόκλιση και επικαλύψεις μεταξύ περιοχών δεδομένων διαφορετικών κατηγοριών, οδηγούν σε λιγότερο ακριβή ταξινόμηση. Η χρήση υψηλών τιμών k επεκτείνει την εξεταζόμενη γειτονιά και, κατά συνέπεια, μπορεί εν μέρει να αποκαταστήσει αυτό το μειονέκτημα. Ωστόσο, αυτό συνεπάγεται σε μεγάλο αριθμό δοκιμών και σφαλμάτων για τον καθορισμό της κατάλληλης τιμής k και ο θόρυβος κατανέμεται ομοιόμορφα στο σύνολο εκπαίδευσης.

1.3 Προεπεξεργασία των δεδομένων

Οι επιδόσεις των περισσότερων ταξινομητών βελτιώνονται σημαντικά μέσω διαφόρων τεχνικών/διαδικασιών που εκτελούνται πριν την ταξινόμηση, οι οποίες εκφράζονται ως προεπεξεργασία των δεδομένων. Στόχος αυτών των ενεργειών είναι αρχικά η εύρεση του βαθμού του θορύβου που ενδεχομένως υπάρχει και εάν εφικτό, τη μείωση του. Αυτό μπορεί για παράδειγμα να επιτευχθεί με την αφαίρεση των διπλότυπων στοιχείων ή τη διαγραφή στοιχείων που θεωρούνται ως outliers (υπερβολικές τιμές/ακραίες περιπτώσεις), δηλαδή δεδομένα που αποτελούν αποκλειστικότητα ή μειονότητα και δεν αντιπροσωπεύουν το ευρύτερο σύνολο όπως στο σχήμα 1.2.

Ιδιαίτερα σημαντική πριν την ταξινόμηση είναι η μελέτη του εύρους των τιμών για



Σχήμα 1.2: Outlier σε ένα σύνολο δεδομένων

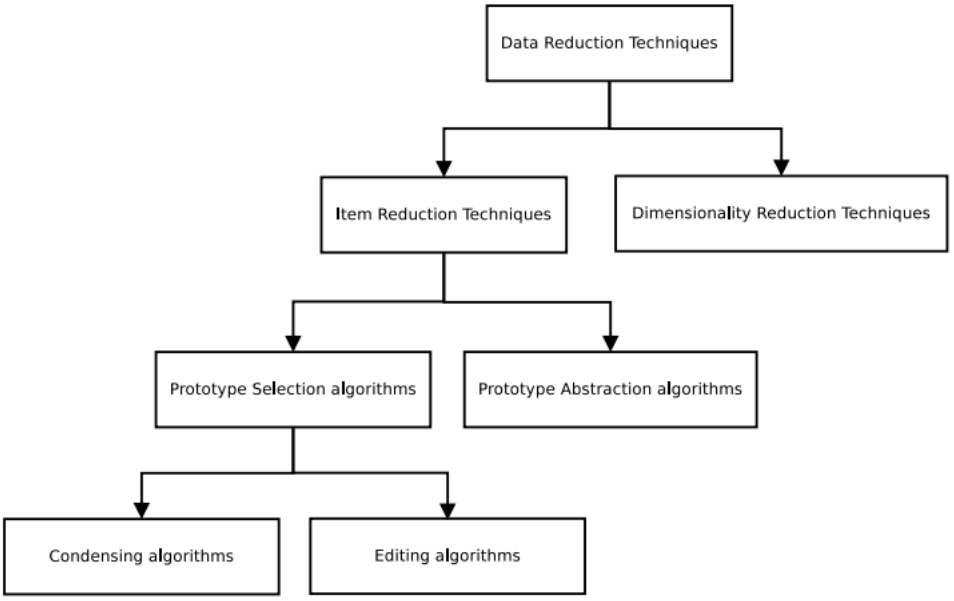
κάθε χαρακτηριστικό του συνόλου. Εάν θεωρήσουμε τα χαρακτηριστικά "μισθός" με εύρος μεταξύ 800-5000 και "αριθμός παιδιών" με τιμές μεταξύ 0-6 και επίσης υποθέσουμε ότι τα δύο χαρακτηριστικά έχουν την ίδια σημασία, είναι προφανές πως το πρώτο θα επισκιάσει τη βαρύτητα του δεύτερου καθώς έχει μεγαλύτερο αντίκτυπο στον υπολογισμό αποστάσεων. Επομένως, το εύρος των παραμέτρων πρέπει να κανονικοποιηθεί σε ένα συγκεκριμένο εύρος διαστημάτων. Στα πειράματα της εργασίας όλα τα σύνολα δεδομένων έχουν κανονικοποιηθεί στο εύρος [0, 1] μέσω του τύπου:

$$Normalized(e_i) = \frac{e_i - A_{min}}{A_{max} - A_{min}}$$

όπου e το i-οστό στοιχείο του χαρακτηριστικού A, με $A_{\min}$ και $A_{\max}$ τα στοιχεία με την ελάχιστη και μέγιστη τιμή του χαρακτηριστικού αντίστοιχα.

Το τελευταίο και σημαντικότερο κομμάτι προεπεξεργασίας με στόχο τη βελτίωση του ταξινομητή k-NN είναι η χρήση τεχνικών μείωσης δεδομένων. Οι τεχνικές αυτές έχουν δύο οπτικές γωνίες: (i) τη μείωση των στοιχείων και (ii) τη μείωση των διαστάσεων. Μπορούν να αντιμετωπίζουν αποτελεσματικά τις αδυναμίες του ταξινο-

μητή και να επιταχύνουν σημαντικά τις επιδόσεις του. Ομαδοποιούνται σε δύο κύριες κατηγορίες αλγορίθμων: αλγόριθμοι επιλογής [27], και αλγόριθμοι γενίκευσης [35] πρωτοτύπων. Οι πρώτοι επιλέγουν αντιπροσωπευτικά αντικείμενα (ή πρωτότυπα) από την αρχική εκπαίδευση ενώ οι δεύτεροι δημιουργούν αντικείμενα, γενικεύοντας τα ήδη υπάρχοντα δεδομένα εκπαίδευσης και τα χρησιμοποιούν ως πρωτότυπα. Στην πράξη, κάθε πρωτότυπο αντιπροσωπεύει μια συγκεκριμένη περιοχή δεδομένων του πολυδιάστατου χώρου.



Σχήμα 1.3: Ιεραρχική ταξινόμηση των κατηγοριών των τεχνικών μείωσης

Οι αλγόριθμοι επιλογής πρωτοτύπων χωρίζονται σε δύο υποκατηγορίες: συμπύκνωσης και επεξεργασίας. Οι αλγόριθμοι αφαίρεσης και σύνοψης πρωτοτύπων έχουν το ίδιο κίνητρο. Σκοπεύουν στην δημιουργία ενός μικρού αντιπροσωπευτικού συνόλου των αρχικών δεδομένων. Αυτό το σύνολο ονομάζεται συνήθως το σύνολο συμπύκνωσης. Η χρήση ενός συνόλου συμπύκνωσης έχει τα πλεονεκτήματα του χαμηλού υπολογιστικού κόστους και απαιτήσεις αποθήκευσης, ενώ η ακρίβεια ταξινόμησης δεν είναι επηρεάζεται αρνητικά. Από την άλλη πλευρά, οι αλγόριθμοι επεξεργασίας στοχεύουν στη βελτίωση της ακρίβειας παρά στην επίτευξη υψηλών

ποσοστών μείωσης. Για να επιτευχθεί αυτό, προσπαθούν να βελτιώσουν την ποιότητα των δεδομένων εκπαίδευσης με την αφαίρεση του θορύβου, των ακραίων τιμών και των λανθασμένων στοιχείων και με την εξομάλυνση των ορίων απόφασης μεταξύ κλάσεων. Στην ιδανική περίπτωση, ένας αλγόριθμος επεξεργασίας δημιουργεί ένα εκπαιδευμένο σύνολο εκπαίδευσης χωρίς επικαλύψεις μεταξύ των κλάσεων. Το σχήμα 1.3 συνοψίζει τις προαναφερόμενες κατηγορίες σε μια ιεραρχική ταξινόμηση.

1.4 Συνεισφορά/Οργάνωση της εργασίας

Η εργασία αυτή εστιάζει στο προαναφερόμενο κομμάτι της προεπεξεργασίας των δεδομένων και επικεντρώνεται στους αλγορίθμους σύνοψης (prototype abstraction ή prototype generation) και συγκεκριμένα στην οικογένεια αλγορίθμων που ονομάζεται αλγόριθμοι μείωσης μέσω διαμερισμού χώρου (Reduction by Space Partitioning). Τελικός στόχος της εργασίας είναι η δραματική μείωση του χρόνου που απαιτείται από τον ταξινομητή k-NN για την παραγωγή αποτελεσμάτων μέσω ενός αντιπροσωπευτικού σετ συμπύκνωσης του σετ εκπαιδεύσεως, χωρίς όμως να επηρεάζεται η ποιότητα τους. Η βελτίωση του χρόνου εκτέλεσης του αλγορίθμου RSP3 της προαναφερόμενης οικογένειας μέσω τεχνικών που θα παρουσιαστούν στα επόμενα κεφάλαια, αποτελεί τον πυρήνα μελέτης και έρευνας την παρούσας πτυχιακής.

Στο κεφάλαιο 2 θα αναλυθεί η μεθοδολογία των τεχνικών μείωσης δεδομένων και θα παρουσιαστούν οι σημαντικότεροι αλγόριθμοι κάθε κατηγορίας, με τις υλοποιήσεις τους και την περιγραφή της λειτουργίας τους.

Στο κεφάλαιο 3 αναλύεται διεξοδικά η τεχνική μείωσης μέσω διαμερισμού του χώρου. Θα παρουσιαστεί το ιστορικό της μεθοδολογίας και οι υλοποιήσεις της, με ιδιαίτερη βαρύτητα στην τελευταία έκδοση (RSP3), η οποία χρησιμοποιήθηκε ως το βασικό αντικείμενο μελέτης της εργασίας.

Στο κεφάλαιο 4 εισάγεται η ιδέα του Convex Hull το οποίο προτείνεται ως ένας

εναλλακτικός τρόπος για την πολύ ταχύτερη εύρεση μεγίστων αποστάσεων σε σύνολα δεδομένων. Περιγράφεται η βασική ιδέα και το ιστορικό των βασικότερων αλγορίθμων ως προς τον υπολογισμό του. Στην συνέχεια, παρουσιάζεται ένας νέος αλγόριθμος που προσεγγίζει το Convex Hull και αποσκοπεί στον ακόμα ταχύτερο υπολογισμό αποστάσεων.

Στο κεφάλαιο 5 παρουσιάζονται δύο νέες παραλλαγές του αλγόριθμου RSP3 που δημιουργήθηκαν από την τεχνογνωσία και τις ιδέες του προηγούμενου κεφαλαίου. Περιγράφεται η υλοποίηση τους, οι διαφορές τους καθώς και η ανάλυση των πειραματικών αποτελεσμάτων που προέκυψαν από τη δοκιμή τους με πραγματική δεδομένα.

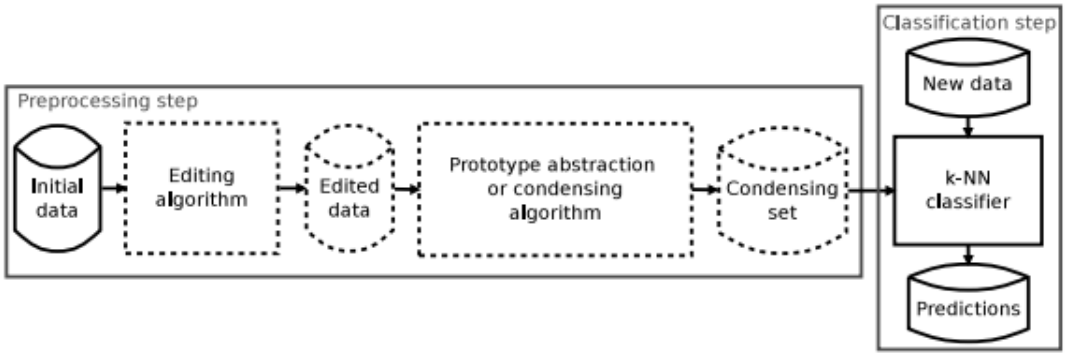
Στο τελευταίο κεφάλαιο παρατίθενται τα συμπεράσματα της μελέτης μέσω συγκριτικών γραφημάτων και επεξηγούνται οι διαφορές μεταξύ όλων των υλοποιήσεων καθώς επίσης δίνονται προτάσεις για πιθανές μελλοντικές έρευνες πάνω στο επιστημονικό πεδίο.

Κεφάλαιο 2

Τεχνικές μείωσης του πληθυσμού των δεδομένων

2.1 Εισαγωγή

Όπως ήδη αναφέρθηκε στην εισαγωγική ενότητα, υπάρχουν δύο κατηγορίες τεχνικών μείωσης δεδομένων (DRT) [7; 29; 49]: επιλογή πρωτοτύπων και σύνοψη πρωτοτύπων. Επιπλέον, οι αλγόριθμοι επιλογής πρωτοτύπων διακρίνονται σε συμπύκνωσης και αλγορίθμους επεξεργασίας. Οι αλγόριθμοι DRT μπορούν να αντεπεξέλθουν στις αδυναμίες του ταξινομητή k-NN. Αυτό επιτυγχάνεται με την οικοδόμηση ενός μικρού αντιπροσωπευτικού συνόλου των αρχικών δεδομένων. Το σύνολο ονομάζεται σύνολο συμπύκνωσης (Condensing Set) και περιέχει μόνο βασικά στοιχεία. Εφαρμόζοντας τον ταξινομητή k-NN χρησιμοποιώντας το σύνολο συμπύκνωσης, υπάρχουν τα πλεονεκτήματα πολύ χαμηλότερου υπολογιστικού κόστους και απαιτήσεις αποθήκευσης, ενώ η ακρίβεια παραμένει υψηλή ή δεν υποβαθμίζεται σημαντικά. Από την άλλη πλευρά, οι αλγόριθμοι επεξεργασίας βελτιώνουν την ακρίβεια, αφαιρώντας δεδομένα που είναι θόρυβος και εξομαλύνουν τα όρια απόφασης μεταξύ των κλάσεων.



Σχήμα 2.1: Διαδικασία ταξινόμησης k-NN μέσω τεχνικών μείωσης

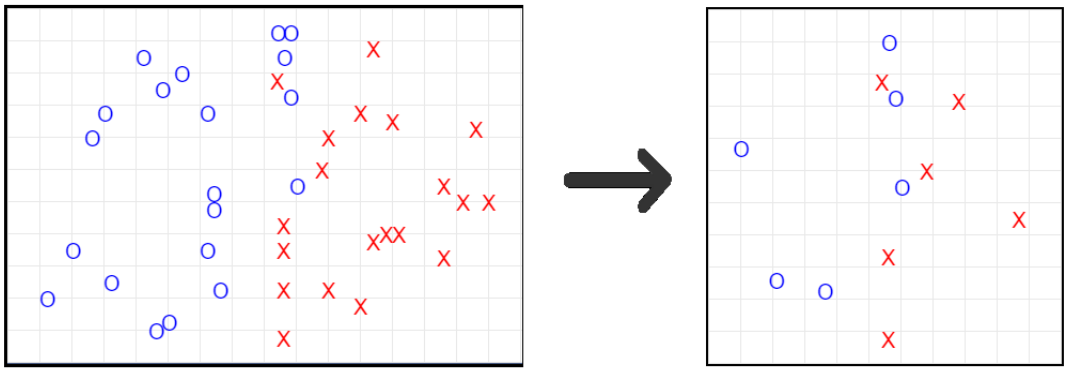
Οι DRT μπορούν να αξιολογηθούν χρησιμοποιώντας τρία κριτήρια. Το πρώτο είναι ο ρυθμός μείωσης που δείχνει πόσο μικρότερο είναι το μέγεθος του συνόλου συμπύκνωσης σε σχέση με το μέγεθος του αρχικού συνόλου εκπαίδευσης. Πρακτικά, είναι ο λόγος του αριθμού των απορριφθέντων αντικειμένων σε σχέση με τον αριθμό των αρχικών στοιχείων του σετ εκπαίδευσης. Προφανώς, όσο υψηλότερος είναι ο ρυθμός μείωσης, τόσο ταχύτερη είναι η ταξινόμηση k-NN. Ένα άλλο σημαντικό κριτήριο είναι η ακρίβεια ταξινόμησης που επιτυγχάνεται με τον ταξινομητή k-NN όταν αυτός εκτελείται πάνω από το σύνολο συμπύκνωσης. Το τρίτο κριτήριο είναι το υπολογιστικό κόστος προεπεξεργασίας που πρόκειται για το κόστος που απαιτείται για την κατασκευή του συνόλου συμπύκνωσης.

Το σχήμα 2.1 συνοψίζει τη διαδικασία ταξινόμησης k-NN μέσω των τεχνικών της μείωσης των δεδομένων. Η διαδικασία περιλαμβάνει δύο φάσεις, προεπεξεργασία και ταξινόμηση. Βεβαίως, η φάση προεπεξεργασίας είναι προαιρετική. Γενικά, υπάρχουν τέσσερις πιθανές μορφές προεπεξεργασίας: (i) χωρίς προεπεξεργασία, (ii) μόνο επεξεργασία, (iii) μόνο συμπύκνωση, και (iv) επεξεργασία και συμπύκνωση. Αν το σετ εκπαίδευσης δεν περιέχει θόρυβο και παραπλανητικά δεδομένα και το μέγεθός του είναι μικρό, δεν απαιτείται προεπεξεργασία. Όταν το μέγεθος του σετ εκπαίδευσης είναι μικρό, αλλά περιέχει θόρυβο, μόνο ένας αλγόριθμος επεξεργασίας θα πρέπει να εκτελείται κατά την προεπεξεργασία. Από την άλλη πλευρά, σε περι-

πτώσεις μεγάλων και χωρίς θόρυβο συνόλων εκπαίδευσης, προτείνεται η μείωση των δεδομένων χωρίς επεξεργασία. Τέλος, σε περιπτώσεις μεγάλων σετ εκπαίδευσης με θόρυβο, και τα δύο είδη αλγορίθμων προεπεξεργασίας είναι απαραίτητα.

2.2 Σχετική έρευνα

2.2.1 Αλγόριθμοι επεξεργασίας



Σχήμα 2.2: Μείωση θορύβου και εξομάλυνση ορίων των κλάσεων

Οι αλγόριθμοι επεξεργασίας βελτιώνουν την ποιότητα των δεδομένων εκπαίδευσης με την αφαίρεση των ακραίων σημείων, του θορύβου και των λανθασμένων στοιχείων καθώς επίσης εξομαλύνουν τα όρια της απόφασης κλάσης. Σε μια ιδανική περίπτωση, μια διαδικασία επεξεργασίας προσπαθεί να δημιουργήσει ένα σύνολο εκπαίδευσης χωρίς επικαλύψεις μεταξύ των κλάσεων. Το σχήμα 2.1 παρουσιάζει τον τύπο δεδομένων που προσπαθούν να αφαιρέσουν οι αλγόριθμοι επεξεργασίας.

2.2.1.1 The Edited Nearest Neighbor (ENN) rule

Ο κανόνας του επεξεργασμένου εγγύτερου γείτονα (ENN) του Wilson [16] αποτελεί τη βάση όλων των άλλων αλγορίθμων επεξεργασίας. Ο κανόνας ENN είναι πολύ απλός. Στο σχήμα 2.2 δίνεται ο ψευδοκώδικας του κανόνα ENN. Αρχικά, το σύνολο

ENN-rule

Input: TS, k
Output: ES

1: $ES \leftarrow TS$
2: **for each** $x \in TS$ **do**
3: $NNs \leftarrow$ find the k nearest to x neighbors in $TS - \{x\}$
4: $majorClass \leftarrow$ find the most common class of NNs
5: **if** $x_{class} \neq majorClass$ **then**
6: $ES \leftarrow ES - \{x\}$
7: **end if**
8: **end for**
9: **return** ES

Σχήμα 2.3: Ψευδοκώδικας του ENN-rule

επεξεργασίας (ES) έχει οριστεί να είναι ίσο με το σύνολο εκπαίδευσης (TS) (γραμμή 1). Για κάθε στοιχείο x του ES, ο αλγόριθμος σαρώνει το TS και ανιχνεύει τους k πλησιέστερους γείτονές του (γραμμή 3). Αν το x έχει ταξινομηθεί εσφαλμένα με την πλειοψηφία των εγγεγραμμένων πλησιέστερων γειτόνων, αφαιρείται από το ES (γραμμές 4-7). Ο κανόνας ENN θεωρεί τα λανθασμένα ταξινομημένα αντικείμενα ως θόρυβο ή οριακά σημεία και, κατά συνέπεια, τα αφαιρεί. Να σημειωθεί πως, σε κάθε επανάληψη του αλγορίθμου, ο κανόνας ENN αναζητά τους πλησιέστερους γείτονες στο αρχικό σετ εκπαίδευσης και όχι στο υπό κατασκευή επεξεργασμένο σύνολο.

Προφανώς, το κόστος της επεξεργασίας εξαρτάται από το μέγεθος του σετ εκπαίδευσης. Σε περιπτώσεις μεγάλων συνόλων δεδομένων, ο κανόνας ENN είναι ένας χρονοβόρος αλγόριθμος, αφού πρέπει να υπολογίζει όλες τις αποστάσεις μεταξύ των αντικειμένων εκπαίδευσης. Αν λοιπόν N είναι ο αριθμός αντικειμένων στο σύνολο εκπαίδευσης, πρέπει να υπολογιστούν $\frac{N \times (N-1)}{2}$ αποστάσεις.

Ένα κρίσιμο ζήτημα είναι ο προσδιορισμός της τιμής του κ που καθορίζει το μέγεθος της εξεταζόμενης γειτονιάς. [25; 50; 61] θεωρούν το $k = 3$ ως μια τυπικά αποδεκτή τιμή. Σε πολλές περιπτώσεις, οι ερευνητές καθορίζουν την τιμή του k που επιτυγχάνει την καλύτερη απόδοση μέσω διαδικασιών δοκιμής και σφάλματος (π.χ. [59]). Αποδεικνύεται ότι η καλύτερη τιμή του k εξαρτάται από το σύνολο δεδομένων

και θα πρέπει να καθοριστεί λαμβάνοντας υπόψη την κατανομή των αντικειμένων στο πολυδιάστατο χώρο. Ακόμη και η καλύτερη τιμή του k μπορεί να μην είναι βέλτιστη καθώς μπορεί να αφαιρέσει στοιχεία που δεν είναι θόρυβος [25] ή να διατηρήσει άλλα που είναι θόρυβος. Αυτό συμβαίνει επειδή ο κανόνας ENN χρησιμοποιεί μια μοναδική τιμή k για ολόκληρο το σετ εκπαίδευσης, ενώ διαφορετικές τιμές k μπορεί να είναι βέλτιστες για διαφορετικές περιοχές στο διάστημα.

2.2.1.2 Ο αλγόριθμος επεξεργασίας Multi Edit

Multi Edit

Input: TS, n, R
Output: ES

1: $ES \leftarrow TS$
2: $r \leftarrow 0$
3: **repeat**
4: $flag \leftarrow$ FALSE
5: $S \leftarrow$ set of n random subsets, $s_1, s_2, \ldots, s_n$ of TS
6: **for each** $s_i \in S$ **do**
7: **for each** $x \in s_i$ **do**
8: $nn \leftarrow$ find the nearest neighbor in $s_{(i+1) \bmod n}$
9: **if** $x_{class} \neq nn_{class}$ **then**
10: $ES \leftarrow ES - \{x\}$
11: $flag \leftarrow$ TRUE
12: **end if**
13: **end for**
14: **end for**
15: **if** $flag ==$ FALSE **then**
16: $r \leftarrow r + 1$
17: **else**
18: $r \leftarrow 0$
19: **end if**
20: $TS \leftarrow ES$
21: **until** $r == R$ {until none of the last R iterations edit data}
22: **return** ES

Σχήμα 2.4: Ψευδοκώδικας του Multi Edit

Το Multi Edit [17] είναι μια άλλη πολύ γνωστή προσέγγιση επεξεργασίας. Ο ψευδοκώδικας του παρουσιάζεται στο σχήμα 2.3. Αρχικά, το επεξεργασμένο σύνολο

(ES) είναι ίσο με το σύνολο εκπαίδευσης (TS) (γραμμή 1). Στην συνέχεια, το TS διαιρείται σε n τυχαία υποσύνολα, s_1, s_2, ..., s_n (γραμμή 5). Ο αλγόριθμος συνεχίζεται εφαρμόζοντας τον κανόνα ENN πάνω σε κάθε στοιχείο $x \in s_i$ (γραμμή 7) κάθε υποσυνόλου (γραμμή 6), αλλά ψάχνοντας το μοναδικό εγγύτερο γείτονα (1-NN) στο επόμενο υποσύνολο υπολοίπου της Ευκλείδειας διαίρεσης (modulo) με το n, π.χ. $s_{(i+1) \bmod n}$ (γραμμή 8). Τα λάθος ταξινομημένα στοιχεία αφαιρούνται από το ES (γραμμή 10). Εάν αφαιρεθεί τουλάχιστον ένα στοιχείο, το TS ορίζεται ως ES (γραμμή 20) και όλη η διαδικασία επαναλαμβάνεται. Το Multi Edit συνεχίζει μέχρι οι τελευταίες R επαναλήψεις να μην επηρεάζουν τα δεδομένα (γραμμές 11,15-16,21).

Εδώ, η παράμετρος k δεν χρησιμοποιείται αφού το Multi Edit χρησιμοποιεί τον ταξινομητή 1-NN. Ωστόσο, οι παράμετροι n και R επηρεάζουν το προκύπτον επεξεργασμένο σύνολο. Η παράμετρος $n \geq 3$ καθορίζει τον αριθμό των υποσυνόλων. Σε πολλά έγγραφα (π.χ. [25; 39]), το $n = 3$ είτε υιοθετείται είτε προτείνεται. Η παράμετρος P καθορίζει τον αριθμό των επαναλήψεων μη επεξεργασίας. Παρ΄όλα αυτά, οι καλύτερες τιμές για αυτές τις παραμέτρους δεν μπορούν να προσδιοριστούν χωρίς συντονισμό μέσω μιας διαδικασίας δοκιμής μέσω σφάλματος.

Το Multi Edit συνήθως επιτυγχάνει υψηλότερα ποσοστά μείωσης από τον κανόνα ENN. Μπορεί να αφαιρέσει με επιτυχία το θόρυβο, τα ακραία σημεία και τα οριακά σημεία στα σύνορα των κλάσεων. Ωστόσο, μπορεί επίσης να αφαιρέσει στοιχεία που δεν είναι θόρυβος. Αν τα αντικείμενα δύο ή περισσοτέρων κλάσεων είναι κοντά μεταξύ τους, μπορεί να εξαλείψει ολόκληρες κατηγορίες [25]. Ένα άλλο μειονέκτημα του Multi Edit είναι ότι βασίζεται σε τυχαίο σχηματισμό των υποσυνόλων, δηλαδή οι επαναλαμβανόμενες εφαρμογές του μπορούν να δημιουργήσουν ένα εντελώς διαφορετικό επεξεργασμένο σύνολο χρησιμοποιώντας το ίδιο σετ εκπαίδευσης.

Είναι συνήθως πιο χρονοβόρο από τον κανόνα ENN. Ωστόσο, μπορεί να υπολογίσει λιγότερες από $\frac{N \times (N-1)}{2}$ αποστάσεις. Μια εφαρμογή του Multi Edit που δεν υπολογίζει μια απόσταση περισσότερες από μία φορές θα πρέπει να έχει τις α-

ποστάσεις που έχουν ήδη υπολογιστεί διαθέσιμες μέχρι το τέλος της εκτέλεσης. Επομένως, μια τέτοια εφαρμογή απαιτεί περισσότερη μνήμη. Σε περίπτωση απλής εκτέλεσης όπου κάθε απόσταση μπορεί να υπολογιστεί περισσότερες από μία φορές, το το υπολογιστικό κόστος του αλγορίθμου εξαρτάται σε μεγάλο βαθμό από την τιμή του R.

2.2.1.3 Άλλοι αλγόριθμοι επεξεργασίας

Πολλές ακόμα προσεγγίσεις επεξεργασίας έχουν προταθεί στο επιστημονικό πεδίο. Οι EENProb και ENNth [60] είναι επεκτάσεις του κανόνα ENN. Και οι δύο ανακτούν τους πλησιέστερους γείτονες k, και στη συνέχεια εκτελούν επεξεργασία βάσει εκτιμήσεων πιθανότητας. Ο επαναλαμβανόμενος ENN (RENN) [33] είναι επίσης μια παραλλαγή του κανόνα ENN. Στην πραγματικότητα, είναι αρκετά παρόμοιος με το All-kNN. Ο κανόνας RENN εφαρμόζει τον κανόνα ENN κατά επαναληπτικό τρόπο, μέχρις ότου η πλειονότητα των k πλησιέστερων αντικειμένων του στοιχείου έχει την ίδια κλάση. Στο [40] προτείνεται μια άλλη απλή παραλλαγή του κανόνα ENN που τοποθετεί ένα στοιχείο στο σύνολο επεξεργασίας, μόνο εάν όλοι οι πλησιέστεροι γείτονές του k έχουν την ίδια κλάση με αυτό.

Η επεξεργασία k-NCN και η επαναληπτική της έκδοση [56] βασίζονται επίσης στον κανόνα ENN. Κάνουν χρήση του ταξινομητή πλησιέστερου κεντροειδούς [38] αντί του ταξινομητή k-NN. Και οι δυο βασίζονται στην ακόλουθη απλή ιδέα: η κατάλληλη περιοχή του χώρου που πρέπει να εξεταστεί για κάθε αντικείμενο καθορίζεται λαμβάνοντας υπόψη όχι μόνο τους πλησιέστερους γείτονές της αλλά και τη συμμετρική κατανομή των γειτόνων γύρω από αυτό.

2.2.2 Αλγόριθμοι συμπύκνωσης

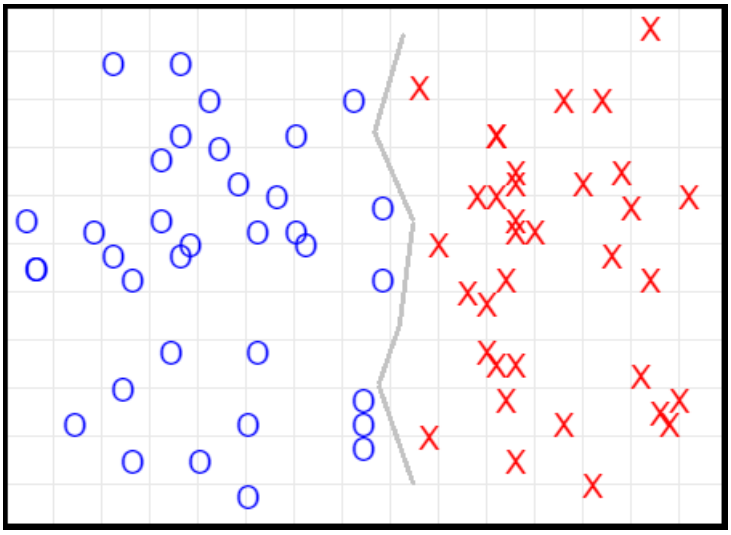
Η συμπύκνωση δεδομένων αναφέρεται στη διαδικασία επιλογής, εστίασης, απλούστευσης, την αφαίρεση ή/και τη μετατροπή των δεδομένων που εμφανίζονται σε ένα

σύνολο δεδομένων. Με τη συμπύκνωση, κάνουμε τα δεδομένα ισχυρότερα. Ακόμα και πριν τα δεδομένα συγκεντρωθούν, προληπτικά η συμπύκνωση δεδομένων συμβαίνει καθώς ο ερευνητής αποφασίζει (συχνά χωρίς πλήρη επίγνωση) ποιο εννοιολογικό πλαίσιο, ποιες περιπτώσεις, ποιες ερευνητικές ερωτήσεις και ποιες προσεγγίσεις συλλογής δεδομένων να επιλέξει. Στην υποενότητα αυτή θα παρουσιαστούν δύο βασικοί αλγόριθμοι συμπύκνωσης με τις υλοποιήσεις τους.

2.2.2.1 The Condensing Nearest Neighbour (CNN) rule

Ο κανόνας συμπύκνωσης πλησιέστερων γειτόνων (CNN) [32] είναι ο πρώτος αλγόριθμος συμπύκνωσης. Ομοίως με πολλά άλλα DRT χτίζει ένα συμπυκνωμένο σύνολο με βάση την ακόλουθη απλή ιδέα: τα στοιχεία που βρίσκονται στην εσωτερική περιοχή δεδομένων μιας κλάσης (μακριά από τα όρια απόφασης κλάσης) είναι άχρηστα κατά τη διάρκεια της διαδικασία ταξινόμησης. Μπορούν λοιπόν να αφαιρεθούν χωρίς απώλεια ακρίβειας. Υιοθετώντας αυτή την ιδέα, ο κανόνας CNN προσπαθεί να τοποθετήσει στο σύνολο συμπύκνωσης μόνο τα στοιχεία που βρίσκονται στις ζώνες δεδομένων των κοντινών κλάσεων. Αυτό είναι το μόνο βασικό κριτήριο για τη διαδικασία ταξινόμησης. Στο σχήμα 2.4 απεικονίζεται αυτή η στρατηγική. Η ιδέα είναι ότι ο ταξινομητής k-NN θα είναι σε θέση να έχει παρόμοια ακρίβεια χρησιμοποιώντας είτε το σύνολο εκπαίδευσης είτε το σύνολο συμπύκνωσης με πολύ χαμηλότερο κόστος υπολογισμών και απαιτήσεις αποθήκευσης από το αρχικό σετ εκπαίδευσης.

Στο σχήμα 2.5 δίνεται ο ψευδοκώδικας του κανόνα CNN. Ο αλγόριθμος προσπαθεί να κρατήσει τα συνοριακά στοιχεία των κλάσεων ως εξής. Αρχικά, ένα στοιχείο του σετ εκπαίδευσης (TS) μετακινείται στο σύνολο συμπύκνωσης (CS) (γραμμή 2). Στην συνέχεια, ο κανόνας CNN εφαρμόζει τον κανόνα 1-NN και ταξινομεί τα στοιχεία του TS με σάρωση των στοιχείων του CS (γραμμή 6). Αν ένα στοιχείο έχει ταξινομηθεί εσφαλμένα, μετακινείται από το TS στο CS (γραμμές 7-11). Ο αλγόριθ-



Σχήμα 2.5: Τα όρια που καθορίζουν τα σημεία του συνόλου συμπύκνωσης

μος συνεχίζεται μέχρι να μην υπάρχουν μετακινήσεις κατά τη διάρκεια ενός πλήρους περάσματος του TS (γραμμή 13). Έτσι εξασφαλίζεται ότι το περιεχόμενο του TS είναι σωστά ταξινομημένο από το περιεχόμενο του CS. Το υπόλοιπο περιεχόμενο του TS απορρίπτεται (γραμμή 14).

Ο κανόνας CNN θεωρεί ότι τα λάθος ταξινομημένα στοιχεία είναι πιθανώς κοντά στα όρια της απόφασης και έτσι πρέπει να τοποθετηθούν στο σύνολο συμπύκνωσης. Προφανώς, τα στοιχεία που είναι θόρυβος είναι παραπλανητικά και τοποθετούνται λανθασμένα με τη γειτονιά τους (στοιχεία που είναι κοντά σε αυτά) στο σύνολο συμπύκνωσης. Ως εκ τούτου, ο ρυθμός μείωσης (και το κόστος προεπεξεργασίας) επηρεάζονται από υψηλά επίπεδα θορύβου. Φυσικά, ο αριθμός των διακεκριμένων κλάσεων μπορεί επίσης να επηρεάσει το ρυθμό μείωσης καθώς υπάρχουν περισσότερα όρια και ως επακόλουθο να συλλέγονται περισσότερα πρωτότυπα.

Ένα πλεονέκτημα του κανόνα CNN είναι ότι πρόκειται για μια μή παραμετρική προσέγγιση. Καθορίζει τον αριθμό των πρωτοτύπων αυτόματα, χωρίς παραμέτρους που ορίζονται από το χρήστη. Μια άλλη επιθυμητή ιδιότητα είναι ότι ο κανόνας CNN μέσω των πολλαπλών περασμάτων πάνω από τα δεδομένα εγγυάται ότι τα αφαιρο-

CNN-Rule

Input: TS **Output:** CS

1: $CS \leftarrow \varnothing$
2: pick an item of TS and move it to CS
3: **repeat**
4: $stop \leftarrow TRUE$
5: **for each** $x \in TS$ **do**
6: $NN \leftarrow$ Nearest Neighbour of x in CS
7: **if** $NN_{class} \neq x_{class}$ **then**
8: $CS \leftarrow CS \cup \{x\}$
9: $TS \leftarrow TS - \{x\}$
10: $stop \leftarrow FALSE$
11: **end if**
12: **end for**
13: **until** $stop == TRUE$ {no move during a pass of TS}
14: discard TS
15: **return** CS

Σχήμα 2.6: Ψευδοκώδικας του CNN-Rule

ύμενα αντικείμενα κατάρτισης μπορούν να ταξινομηθούν σωστά με την εκτέλεση του κανόνα 1-NN στο πλαίσιο του τελικού σετ συμπύκνωσης. Ένα μειονέκτημα είναι ότι το σύνολο συμπύκνωσης που θα προκύψει εξαρτάται από τη σειρά των αντικειμένων εκπαίδευσης, πράγμα που σημαίνει ότι ο κανόνας CNN δημιουργεί διαφορετικό σύνολο συμπύκνωσης εξετάζοντας τα ίδια δεδομένα κατάρτισης με διαφορετική σειρά. Επιπλέον, ο κανόνας CNN δεν μπορεί να χειριστεί νέα δεδομένα εκπαίδευσης. Αυτό σημαίνει ότι τα νέα δεδομένα εκπαίδευσης δεν μπορούν να ενημερώσουν ένα ήδη κατασκευασμένο σύνολο συμπύκνωσης. Για να κατασκευαστεί ένα ενημερωμένο σύνολο συμπύκνωσης χρειάζεται να εκτελεστεί ο CNN από το μηδέν σε όλο το σύνολο εκπαίδευσης. Επομένως, τα στοιχεία εκπαίδευσης που δεν εισέρχονται στο σύνολο συμπυκνωμάτων θα πρέπει να διατηρηθούν.

2.2.2.2 Ο αλγόριθμος IB2

Ο IB2 ανήκει στους πολύ γνωστούς αλγόριθμους της οικογένειας εκπαίδευσης βάση περίστασης (Instance-Based Learning, IBL) [5, 4] και βασίζεται στον κανόνα CNN. Στην πραγματικότητα, ο IB2 αποτελεί μια απλή παραλλαγή ενός περάσματος του κανόνα CNN. Στο σχήμα 2.6 παρουσιάζεται ο ψευδοκώδικας του. Κάθε αντικείμενο εκπαίδευσης x ∈ TS ταξινομείται χρησιμοποιώντας το 1-NN ταξινομητή στο τρέχων CS (γραμμή 4). Εάν το x έχει ταξινομηθεί σωστά, απορρίπτεται (γραμμή 8). Διαφορετικά, το x μεταφέρεται στο CS (γραμμή 6).

IB2

Input: TS **Output:** CS

1: $CS \leftarrow \varnothing$
2: pick an item of TS and move it to CS
3: **for each** $x \in TS$ **do**
4: $NN \leftarrow$ Nearest Neighbour of x in CS
5: **if** $NN_{class} \neq x_{class}$ **then**
6: $CS \leftarrow CS \cup \{x\}$
7: **end if**
8: $TS \leftarrow TS - \{x\}$
9: **end for**
10: **return** CS

Σχήμα 2.7: Ψευδοκώδικας του IB2

Παρομοίως με τον κανόνα CNN, ο IB2 είναι μη παραμετρικός και το σύνολο συμπύκνωσης εξαρτάται σε μεγάλο βαθμό από τη σειρά των αντικειμένων στο σετ εκπαίδευσης. Σε αντίθεση με τον κανόνα CNN, ο IB2 δεν εξασφαλίζει ότι όλα τα απορριφθέντα αντικείμενα μπορούν να ταξινομηθούν σωστά από την τελική έκδοση του συνόλου συμπύκνωσης. Ωστόσο, επειδή είναι ένας αλγόριθμος ενός περάσματος, είναι πολύ γρήγορος καθώς περιλαμβάνει υπολογισμούς χαμηλού κόστους προεπεξεργασίας.

Επιπλέον, ο IB2 κατασκευάζει το σύνολο συμπύκνωσης του σταδιακά. Μπορούν να εισαχθούν νέα αντικείμενα εκπαίδευσης μετέπειτα της δημιουργίας του συνόλου

συμπύκνωσης. Με άλλα λόγια, νέα δεδομένα εκπαίδευσης μπορούν να ενημερώσουν ένα ήδη υπάρχον σύνολο συμπύκνωσης με απλό τρόπο και χωρίς να πρέπει να ληφθούν υπόψη τα "παλαιά" (αφαιρούμενα) δεδομένα που είχαν χρησιμοποιηθεί για την κατασκευή του αρχικού συνόλου. Κάθε νέο αντικείμενο εκπαίδευσης μπορεί να εξετασθεί για το εάν θα τοποθετηθεί ή όχι στο σύνολο συμπύκνωσης και στη συνέχεια να αφαιρεθεί. Επιπλέον, ο IB2 μπορεί να χειριστεί την εισαγωγή νέων κλάσεων. Επομένως, ο IB2 είναι μια κατάλληλη DRT για δυναμικά περιβάλλοντα, όπου νέα δεδομένα εκπαίδευσης μπορεί να είναι συστηματικά διαθέσιμα.

Τελευταίο και αρκετά σημαντικό, σε αντίθεση με τον κανόνα CNN και πολλά άλλα DRT, ο IB2 δεν απαιτεί την αποθήκευση όλων των δεδομένων εκπαίδευσης στην κύρια μνήμη, πράγμα που σημαίνει πως είναι εφικτό να εκτελεστεί από συσκευές των οποίων η χωρητικότητα μνήμης δεν επαρκεί για την αποθήκευση ολόκληρου του συνόλου εκπαίδευσης.

2.2.2.3 Άλλοι αλγόριθμοι συμπύκνωσης

Υπάρχουν πολλοί άλλοι αλγόριθμοι συμπύκνωσης οι οποίοι είτε επεκτείνουν τον κανόνα του CNN είτε βασίζονται στην ίδια ιδέα, δηλαδή στην αφαίρεση των δεδομένων που δεν είναι κοντά στα σύνορα. Μερικοί από αυτούς τους αλγόριθμους είναι ο κανόνας του μειωμένου πλησιέστερου γείτονα (RNN) [24], ο κανόνας του επιλεκτικού πλησιέστερου γείτονα [23], ο κανόνας του τροποποιημένου CNN [15], ο γενικευμένος κανόνας CNN [10], οι ταχύς Fast CNN αλγόριθμοι [4; 5], ο κανόνας CNN του Tomek [34], ο αλγόριθμος (POP) [2; 36], και η πρόσφατα προτεινόμενη μείωση προτύπων για k-NN (TRkNN) [21].

Με την ολοκλήρωση αυτής της σύντομης ανασκόπησης, θα πρέπει να αναφέρουμε ότι ορισμένοι αλγόριθμοι συμπύκνωσης ενσωματώνουν την ιδέα της επεξεργασίας. Αυτοί οι αλγόριθμοι ονομάζονται υβριδικοί [26]. Η υβριδοποίηση επιτυγχάνεται συνδυάζοντας μηχανισμούς συμπύκνωσης και επεξεργασίας σε μια διαδικασία

μείωσης δεδομένων. Αυτοί οι αλγόριθμοι επιλογής πρωτότυπων προσπαθούν να καταργήσουν τα μη κοντινά στοιχεία συνόρων και, ταυτόχρονα, να εξομαλύνουν τις περιοχές των συνόρων αλλά και να απομακρύνουν το θόρυβο και τα λανθασμένα στοιχεία.

2.2.3 Αλγόριθμοι σύνοψης

Οι αλγόριθμοι σύνοψης πρωτοτύπων έχουν τον ίδιο στόχο με τους αλγορίθμους συμπύκνωσης, διαφέρουν όμως ως προς τον τρόπο κατασκευής του τελικού συνόλου. Σε αντίθεση με τους αλγόριθμους συμπύκνωσης που επιλέγουν ορισμένα "πραγματικά" αντικείμενα εκπαίδευσης ως πρωτότυπα, οι αλγόριθμοι σύνοψης δημιουργούν νέα πρωτότυπα, συνοψίζοντας τα δεδομένα εκπαίδευσης σε παρόμοια αντικείμενα. Στην πραγματικότητα, ένας k-NN ταξινομητής που υιοθετεί την ιδέα της σύνοψης ταξινομεί ένα τεχνητό σύνολο εκπαίδευσης.

2.2.3.1 Ο αλγόριθμος των Chen και Jozwik

Οι Chen και Jozwik πρότειναν έναν πολύ γνωστό και αποτελεσματικό αλγόριθμο σύνοψης πρωτοτύπων. Ο αλγόριθμος Chen και Jozwik (CJA) [8] αρχικά ανακτά τα πιο απομακρυσμένα στοιχεία, έστω x και y, στο σετ εκπαίδευσης. Η απόσταση μεταξύ των δύο καθορίζει τη διάμετρο του συνόλου δεδομένων. Στην συνέχεια, με βάση αυτά τα δύο στοιχεία, ο CJA χωρίζει το σετ εκπαίδευσης σε δύο υποσύνολα: τα αντικείμενα που βρίσκονται πιο κοντά στο x τοποθετούνται στο S_x ενώ αυτά που βρίσκονται πλησιέστερα στο y τοποθετούνται στο S_y αντίστοιχα. Ο CJA προχωρά διαιρώντας τα υποσύνολα που περιέχουν στοιχεία περισσότερων από μίας κλάσης (μη ομοιογενή υποσύνολα). Το μη ομοιογενές υποσύνολο με τη μεγαλύτερη διάμετρο διαιρείται πάντα πρώτο. Εάν όλα τα υποσύνολα είναι ομοιογενή, ο CJA συνεχίζει διαιρώντας τα ομοιογενή υποσύνολα. Η διαδικασία επαναλαμβάνεται μέχρις ότου ο αριθμός των υποσυνόλων ισούται με μια τιμή που καθορίζεται από το χρήστη. Στο

τέλος, για κάθε δημιουργημένο υποσύνολο S, ο CJA υπολογίζει το μέσο όρο των χαρακτηριστικών του κάθε αντικειμένου στο S και δημιουργεί ένα μέσο στοιχείο στο οποίο ορίζεται ως κλάση η πλειοψηφία του S. Τα στοιχεία που δημιουργούνται αποτελούν το τελικό σύνολο συμπύκνωσης.

Στο σχήμα 2.7 παρουσιάζεται μια πιθανή υλοποίηση του CJA σε ψευδοκώδικα. Ως είσοδο, δέχεται ένα σετ εκπαίδευσης (TS) και τον αριθμό των πρωτοτύπων, n, που θα παραχθούν. Ο αλγόριθμος χρησιμοποιεί μια δομή δεδομένων για την αποθήκευση των υποσυνόλων που έχουν δημιουργηθεί. Αρχικά, ολόκληρο το TS αποτελεί ένα υποσύνολο και αποθηκεύεται στο S (γραμμές 1-2). Στην συνέχεια, το μη ομοιογενές υποσύνολο C με τη μεγαλύτερη διάμετρο διαιρείται σε δύο υποσύνολα (γραμμές 4,8). Εάν όλα τα υποσύνολα είναι ομοιογενή, ο CJA διαιρεί το ομοιογενές υποσύνολο C με την υψηλότερη διάμετρο (γραμμές 5-7). Και τα δύο υποσύνολα προστίθενται στο S, ενώ το C διαγράφεται (γραμμές 9-11). Η διαδικασία της κατασκευής υποσυνόλων συνεχίζεται έως ότου δημιουργηθούν n υποσύνολα (γραμμή 3). Το τελευταίο βήμα είναι ο υπολογισμός του μέσου στοιχείου (ή η παραγωγή πρωτότυπου) για κάθε υποσύνολο και η ένταξή του στο σύνολο συμπύκνωσης CS (γραμμές 13-18).

Ο CJA επιλέγει το επόμενο υποσύνολο που θα διαιρεθεί εξετάζοντας τη διάμετρο του. Η ιδέα είναι πως ένα υποσύνολο με μεγάλη διάμετρο πιθανότατα περιέχει περισσότερα αντικείμενα εκπαίδευσης και αν αυτό διαιρεθεί αρχικά, θα επιτευχθεί υψηλότερος ρυθμός μείωσης. Θετικό είναι πως ο CJA δημιουργεί το ίδιο υποσύνολο συμπύκνωσης ανεξάρτητα από τη σειρά των δεδομένων στο σύνολο εκπαίδευσης. Έχει όμως δύο αδύνατα σημεία. Πρώτον, ο αλγόριθμος είναι παραμετρικός. Ο χρήστης πρέπει να καθορίσει τον αριθμό των πρωτοτύπων. Σε ορισμένες περιπτώσεις, αυτή η ιδιότητα είναι επιθυμητή, αφού επιτρέπει τον έλεγχο του μεγέθους του συνόλου συμπύκνωσης. Η δεύτερη αδυναμία είναι ότι τα στοιχεία που δεν ανήκουν στην πιο κοινή κλάση του υποσυνόλου δεν αντιπροσωπεύονται, καθώς το κάθε μέσο στοιχε-

CJA

Input: TS, n
Output: CS

1: $S \leftarrow \varnothing$
2: add(S, TS)
3: **for** $i = 2$ to n **do**
4: $C \leftarrow$ select the non-homogeneous subset $\in S$ with the largest diameter
5: **if** $C == \varnothing$ {All subsets are homogeneous} **then**
6: $C \leftarrow$ select the homogeneous subset $\in S$ with the largest diameter
7: **end if**
8: $(S_x, S_y) \leftarrow$ divide C into two subsets
9: add(S, S_x)
10: add(S, S_y)
11: remove(S, C)
12: **end for**
13: $CS \leftarrow \varnothing$
14: **for each** subset $T \in S$ **do**
15: $r \leftarrow$ compute the mean item by averaging the items in T
16: $r.label \leftarrow$ find the most common class label in T
17: $CS \leftarrow CS \cup \{r\}$
18: **end for**
19: **return** CS

Σχήμα 2.8: Ψευδοκώδικας του CJA

ίο αποδίδεται στην πιο κοινή κλάση με αποτέλεσμα τα αντικείμενα που ανήκουν σε άλλες κλάσεις πρακτικά αγνοούνται.

2.2.3.2 Ο αλγόριθμος AIB2

Ο αλγόριθμος AIB2 (Abstraction IB2) [51] αποτελεί μια παραλλαγή του αλγορίθμου IB2 που υιοθετεί όλα τα χαρακτηριστικά του. Ο AIB2 θεωρεί πως τα πρωτότυπα θα πρέπει να βρίσκονται κοντά στο κέντρο της περιοχής των δεδομένων που αντιπροσωπεύουν. Σε αντίθεση με τον IB2 ο AIB2 δεν αγνοεί της περιπτώσεις που ταξινομήθηκαν σωστά, αλλά τις χρησιμοποιεί για την κατασκευή του σετ συμπύκνωσης επανατοποθετώντας το πλησιέστερο πρωτότυπο. Αυτό επιτυγχάνεται αναθέτοντας μια τιμή "βαρύτητας" σε κάθε πρωτότυπο που δηλώνει τον αριθμό των περιπτώσεων που αντιπροσωπεύει.

Στα αρχικά βήματα, μια τυχαία τιμή εκπαίδευσης τοποθετείται στο σετ συμπύκνωσης και η βαρύτητα της ισούται με ένα. Για κάθε περίπτωση του σετ εκπαίδευσης x, ο AIB2 βρίσκει το πλησιέστερο πρωτότυπο της P από το παρών σετ συμπύκνωσης. Εάν η κλάση του x διαφέρει από αυτήν του P, τότε μεταφέρεται στο σετ συμπύκνωσης όπου του ανατίθεται ο ρόλος ενός καινούργιου πρωτοτύπου με βαρύτητα ίση με ένα. Διαφορετικά, τα χαρακτηριστικά του P τροποποιούνται σύμφωνα με αυτά του x και της βαρύτητάς του. Πιο συγκεκριμένα κάθε χαρακτηριστικό $attr(i)$ του P γίνεται $P_{attr(i)} \leftarrow \frac{P_{attr(i)} \times P_{weight} + x_{attr(i)}}{NN_{weight}+1}$. Έτσι, το P μετακινείται ελαφρώς προς το x. Η βαρύτητα του P αυξάνεται κατά ένα και το x απορρίπτεται.

2.2.3.3 Ο αλγόριθμος RHC

Ο αλγόριθμος RHC (Reduction through Homogeneous Clusters) [52; 55] βασίζεται στην έννοια της ομοιογένειας αλλά χρησιμοποιεί τη μέθοδο k-means clustering [31] προς αποφυγή δαπανηρών υπολογισμών που απαιτούνται σε απομακρυσμένες περιπτώσεις. Αρχικά, τα δεδομένα εκπαίδευσης θεωρούνται ως μια μη ομοιογενή συστάδα και ο αλγόριθμος υπολογίζει μια μέση τιμή για κάθε κλάση σε αυτήν. Στην συνέχεια ο RHC εκτελεί τον αλγόριθμο k-means στην παρούσα μη ομοιογενή συστάδα χρησιμοποιώντας τη μέση τιμή των κλάσεων ως αρχική τιμή. Το αποτέλεσμα είναι η δημιουργία συστάδων πλήθους ίσο με τον αριθμό των διακριτών κλάσεων. Η διαδικασία συσταδοποίησης επαναλαμβάνεται για κάθε μη ομοιογενή συστάδα έως ότου όλες οι συστάδες να είναι ομοιογενής. Κάθε συστάδα συμβάλει στο σετ συμπύκνωσης με ένα πρωτότυπο που προκύπτει από τη μέση τιμή των χαρακτηριστικών των στοιχείων της.

Ο RHC παράγει πολλά πρωτότυπα για τις περιοχές κοντά στα σύνορα των κλάσεων και λιγότερα για τις "εσωτερικές" περιοχές. Η μέση τιμή των κλάσεων χρησιμοποιείται ως αρχική τιμή με σκοπό τη γρήγορη εύρεση μεγάλων ομοιογενών συστάδων. Αυτό έχει το πλεονέκτημα επίτευξης υψηλών ποσοστών μείωσης, καθώς,

όσο μεγαλύτερη είναι συστάδα που θα βρεθεί τόσο υψηλότερο και το ποσοστό μείωσης. Προφανώς ο θόρυβος έχει αντίκτυπο στο ποσοστό μείωσης. Ο RHC είναι γρήγορος διότι η σειρά των δεδομένων στο σετ εκπαίδευσης δεν επηρεάζει το σετ συμπύκνωσης. Σχετικές πειραματικές μελέτες [] απέδειξαν πως ο RHC είναι ταχύτερος από τον CNN χωρίς να επηρεάζεται το ποσοστό ευστοχίας.

2.2.3.4 Άλλοι αλγόριθμοι σύνοψης

Η οικογένεια των αλγορίθμων RSP (Reduction by Space Partitioning) ανήκει στην κατηγορία των αλγορίθμων σύνοψης και θα αναλυθεί εκτενώς στο επόμενο κεφάλαιο. Επιπλέον, οι αλγόριθμοι που βασίζονται στο Quantization Vector Learning (LVQ) είναι παραδοσιακοί αλγόριθμοι αυτού του τύπου. Αρχικά, ο Kohonen πρότεινε ένα σύνολο τεσσάρων αλγορίθμων βασισμένων σε LVQ [43] (Επίσης, βλ. Kohonen et al [44]). Όπως και ο αλγόριθμος των Chen Jozwik, ο RSP1 και ο RSP2, επιτρέπουν στο χρήστη να επιλέξει το αντιστάθμισμα μεταξύ ακρίβειας και ρυθμού μείωσης, καθορίζοντας το μέγεθος του συνόλου συμπύκνωσης μέσω παραμέτρων εισόδου. Στη βιβλιογραφία υπάρχουν διαθέσιμα πέντε άλλοι αλγόριθμοι αφαίρεσης πρωτότυπου βασισμένοι σε LVQ που βασίζονται στο έργο του Kohonen: VQ [62], LVQ με εκπαιδευτικό αριθμό (LVQTC) [54], προσαρμοστικό LVQ και υβριδικός LVQ3 [57] και LVQ με μείωση (prunning) (LVQPRU) [46].

Ορισμένοι άλλοι γνωστοί αλγόριθμοι αφαίρεσης πρωτότυπων βασίζονται σε διαδικασίες προεπεξεργασίας σε ομάδες. Αυτοί που παράγουν πρωτότυπα όπως ο ταξινομητής (SNMC) [13] και ο αλγόριθμος γενικευμένου τροποποιημένου Chang (GMCA) [53] είναι χαρακτηριστικά παραδείγματα. Ο αλγόριθμος βασίζεται στην ιδέα του παλαιότερου και γνωστού αλγόριθμου αφαίρεσης πρωτοτύπου που εισήγαγε ο Chang [9].

Κεφάλαιο 3

Μείωση δεδομένων με διαμερισμό του χώρου

Περίληψη

Ο αλγόριθμος των Chen και Jozwik αποτελεί τον πρόγονο της οικογένειας μείωσης δεδομένων με διαμερισμό του χώρου. Οι αλγόριθμοι RSP [37] είναι ένα δημοφιλές σύνολο τριών αλγορίθμων γενίκευσης προτύπων που είναι γνωστού ως RSP1, RSP2 και RSP3. Ιδιαίτερα η τρίτη έκδοση (RSP3) αποτελεί το βασικό έναυσμα μελέτης της εργασίας και θα δοθεί ιδιαίτερη βαρύτητα στην ανάλυση του αλγορίθμου και της μεθοδολογίας του, ώστε με το κατάλληλο υπόβαθρο να είναι εφικτή η πρόταση νέων τεχνικών με σκοπό την βελτίωση του.

3.1 Η πρώτη έκδοση (RSP1)

Ένα πρόβλημα που εμφανίζεται στο μοντέλο που εισήγαγαν οι Chen και Jozwik είναι το γεγονός πως σε ορισμένες περιπτώσεις, όλα τα στοιχεία που ανήκουν σε μία από τις κλάσεις του συνόλου δεδομένων μπορούν να αφαιρεθούν από το TS

(σύνολο εκπαίδευσης) λόγω του τρόπου με τον οποίο καθορίζονται οι εκπρόσωποι των κλάσεων (νέα παραγόμενα πρωτότυπα). Συνεπώς, εάν μια κλάση αφαιρεθεί εξ ολοκλήρου, ο ταξινομητής θα αποτύχει σε όλες τις περιπτώσεις των στοιχείων που ανήκουν σε αυτήν.

RSP1

1: $b_c \leftarrow 1$ and $i \leftarrow 1$ where b_c is the current number of subsets in T
2: $B = T$
3: Find the two farthest points p_1 and p_2, in B
4: Divide the set B into two subsets B_1 and B_2, where

$$B_1 = \{p \in B : d(p, p_1) \leq d(p, p_2)\},$$
$$B_2 = \{p \in B : d(p, p_2) < d(p, p_1)\}$$

5: $b_c \leftarrow b_c + 1$, $C(i) = B_1$ and $C(b_c) = B_2$
6: $I_1 = \{i : C(i)$ contains instances from two different classes at least$\}$, and $I_2 = \{i : i \leq b_c\}$ - I_1
7: $I = I_1$ **if** $I_1 \neq 0$ **else** $I = I_2$
8: Find the two farthest points, $q_1(i)$ and $q_2(i)$, in each $C(i)$, for $i \in I$
9: Find the set $C(j)$ with the largest diameter Θ_j
10: $B = C(j)$, $p_1 = q_1(j)$, and $p_2 = q_2(j)$
11: **if** $b_c < b$ (where b is the number of final subsets) **go to** 4
12: Find the centroids $c(l, i)$ for each class l in subset $C(i)$, $i = 1, 2, \ldots, b$
13: Put all $c(l, i)$ in the resulting set S

Σχήμα 3.1: Ψευδοκώδικας του RSP1

Μεριμνώντας για το πρόβλημα αυτό, ο πρώτος αλγόριθμος που προτείνεται από τον Sánchez ξεκινά με τον υπολογισμό της διαμέτρου του αρχικού συνόλου δεδομένων T, η οποία δίνεται από τα δύο πιο απομακρυσμένα σημεία του, έστω p_1 και p_2 και στην συνέχεια διαιρώντας το σύνολο εκπαίδευσης (TS) σε δύο υποσύνολα. Ένα υποσύνολο περιέχει εκείνες τις περιπτώσεις των στοιχείων που είναι κοντινότερα στο p_1, ενώ το άλλο υποσύνολο περιέχει τις υπόλοιπες περιπτώσεις. Αυτή η διαδικασία επαναλαμβάνεται μέχρις ότου παραχθούν b υποσύνολα. Στη συνέχεια, για κάθε υποσύνολο, ο RSP1 υπολογίζει ένα κεντροειδές για κάθε κλάση που συμπεριλαμβάνεται σε αυτό. Ως αποτέλεσμα, θα παραχθούν c πρωτότυπα ανά υποσύνολο, με

τον αριθμό c να εκφράζει τον αριθμό των κλάσεων που καλύπτονται από ένα τέτοιο διαμέρισμα.

Στο σχήμα 3.1 δίνεται η υλοποίηση σε ψευδοκώδικα του RSP1. Για αυτή την πρώτη τεχνική μείωσης, ο αριθμός των τελικών αντιπροσώπων στο S δεν μπορεί να καθοριστεί εκ των προτέρων. Αντίθετα, ο αριθμός των υποσυνόλων είναι γνωστός, που έχει ως αποτέλεσμα ένα αριθμό πρωτοτύπων $b \leq S \leq bm$, όπου b υποδηλώνει τον αριθμό των υποσυνόλων που δημιουργούνται και m είναι ο αριθμός των κλάσεων που υπάρχουν στο αρχικό TS.

Είναι σημαντικό να σημειωθεί ότι με τον ορισμό του κεντρικού πεδίου (centroid) των περιπτώσεων από κάθε κλάση, αναμένεται ότι τα όρια απόφασης που σχετίζονται με το προκύπτων σύνολο πρωτοτύπων δεν θα μετατοπιστούν όσο στην περίπτωση του αλγορίθμου των Chen και Jozwik. Από την άλλη πλευρά, ακόμα πιο σημαντικό, η υλοποίηση αυτή εγγυάται ότι καμία κλάση δεν είναι κενή μετά την εφαρμογή του αλγορίθμου.

3.2 Η δεύτερη έκδοση (RSP2)

Στην προηγούμενη έκδοση, το κριτήριο διαίρεσης μπορεί να δηλωθεί ως εξής: διαίρεση του υποσυνόλου με τη μεγαλύτερη διάμετρο (γραμμές 9 και 10 του RSP1). Η ιδέα είναι ότι το μεγαλύτερο υποσύνολο θα πρέπει πιθανότατα να περιέχει περισσότερες περιπτώσεις από οποιοδήποτε άλλο και ως εκ τούτου θα πρέπει επίσης να επιτύχουμε το υψηλότερο ποσοστό μείωσης. Ωστόσο, η θεωρία υπαγορεύει ότι τα στοιχεία μιας κλάσης πρέπει να είναι όσο το δυνατόν πλησιέστερα μεταξύ τους, ενώ αντίθετα τα στοιχεία που ανήκουν σε διαφορετικές κλάσεις πρέπει να βρίσκονται όσο το δυνατόν πιο μακριά μεταξύ τους. Συνεπώς, από θεωρητική άποψη, θα ήταν πιο ενδεδειγμένο να χωριστεί το υποσύνολο με τον υψηλότερο βαθμό επικαλύψεως μεταξύ περιπτώσεων από διαφορετικές κλάσεις [37].

Ο βαθμός επικάλυψης ενός συνόλου δεδομένων X, έστω Φ_x, ορίζεται ως ο λόγος της μέσης απόστασης μεταξύ των στοιχείων που ανήκουν σε διαφορετικές κλάσεις, $D^{\neq}$, και της μέσης απόστασης μεταξύ των στοιχείων που ανήκουν στην ίδια κλάση, $D^{=}$ και αποτελεί το κριτήριο διαίρεσης που χρησιμοποιείται στην δεύτερη έκδοση (RSP2).

3.3 Η τρίτη έκδοση (RSP3)

Η τρίτη έκδοση συνίσταται στην εκτέλεση κατατμήσεων έως ότου όλα τα υποσύνολα είναι ομοιογενή από πλευράς κλάσης. Ο λόγος πίσω από αυτό είναι ότι κάθε υποσύνολο πρέπει να αντιπροσωπεύει ένα σύμπλεγμα στοιχείων που ανήκουν σε μία μόνο κλάση. Σε αυτή την περίπτωση, δεν είναι απαραίτητο να ορίζεται στον αλγόριθμο κάποια παράμετρος συντονισμού (αριθμός υποσυνόλων ή αριθμός τελικών πρωτοτύπων).

Ο RSP3 μπορεί να χρησιμοποιήσει και τα δύο κριτήρια διαίρεσης που είχαν αναφερθεί στις προηγούμενες εκδόσεις: διαίρεση του υποσυνόλου με τη μεγαλύτερη διάμετρο ή εκείνο με τον υψηλότερο βαθμό επικαλύψεως. Στην πραγματικότητα, το κριτήριο διαίρεσης δεν είναι σημαντικό εδώ, διότι όλα τα μη-ομοιογενή υποσύνολα πρέπει να διαιρούνται. Ο αριθμός των πρωτότυπων που παράγονται από αυτήν την προσέγγιση θα είναι γενικά πολύ υψηλότερος από αυτόν των άλλων εκδόσεων RSP, ειδικά εάν δεν έχουν αφαιρεθεί προηγουμένως από το σύνολο εκπαίδευσης ο θόρυβος και τα ακραία σημεία. Κατά συνέπεια, ο RSP3 εξαλείφει και τα δύο αδύνατα σημεία του CJA. Αξίζει να αναφερθεί ότι όπως οι CJA, RSP1 και RSP2, το σετ συμπύκνωσης που κατασκευάζεται από τον RSP3 δεν εξαρτάται από την σειρά των δεδομένων στο σετ εκπαίδευσης.

Στο σχήμα 3.2 παραθέτεται ο ψευδοκώδικας του RSP3. Χρησιμοποιεί μια απλή δομή δεδομένων S για να αποθηκεύει τα μη επεξεργασμένα υποσύνολα. Αρχικά,

RSP3

Input: TS
Output: CS

1: $S \leftarrow \varnothing$
2: add(S, TS)
3: $CS \leftarrow \varnothing$
4: **repeat**
5: $C \leftarrow$ select the subset $\in S$ with the highest splitting criterion value
6: **if** C is homogeneous **then**
7: $r \leftarrow$ calculate the mean item by averaging the items in C
8: $r.label \leftarrow$ class of items in C
9: $CS \leftarrow CS \cup \{r\}$
10: **else**
11: $(D_1, D_2) \leftarrow$ divide C into two subsets
12: add(S, D_1)
13: add(S, D_2)
14: **end if**
15: remove(S, C)
16: **until** IsEmpty(S)
17: **return** CS

Σχήμα 3.2: Ψευδοκώδικας του RSP3

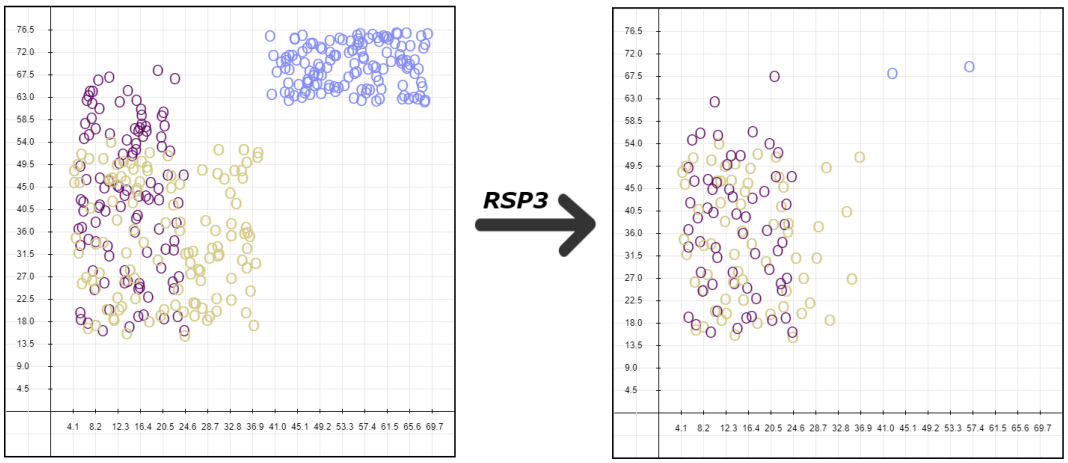
ολόκληρο το σετ εκπαίδευσης (TS) θεωρείται ως ένα μη επεξεργασμένο υποσύνολο και τοποθετείται στο S (γραμμή 2). Σε κάθε επανάληψη μέχρι την περάτωση του αλγορίθμου, ο RSP3 επιλέγει το υποσύνολο C με το υψηλότερο κριτήριο διαίρεσης (γραμμή 5) και ελέγχει εάν αυτό είναι ομοιογενές ή όχι. Αν είναι ομοιογενές, το μέσο στοιχείο υπολογίζεται από τον μέσο όρο των στοιχείων στο C και ακολουθεί η εισαγωγή του στο σύνολο συμπύκνωσης (CS) ως παραγόμενο πρωτότυπο (γραμμές 6-9). Διαφορετικά, το C διαιρείται σε δύο υποσύνολα D1 και D2 (γραμμή 11) παρομοίως με τον CJA. Τα νέα υποσύνολα προστίθενται στο S (γραμμές 12, 13). Έπειτα, ασχέτως εάν υπήρξε διαίρεση υποσυνόλου ή όχι το C αφαιρείται από το S (γραμμή 15). Ο βρόχος επαναλαμβάνεται έως ότου το S γίνει κενό (γραμμή 16), δηλαδή όλα τα υποσύνολα είναι ομοιογενή.

Μελετώντας τον RSP3, παρατηρείται ότι παράγει λίγα πρωτότυπα για την αντιπροσώπευση μη κοντινών συνοριακών περιοχών και πολλά πρωτότυπα για την

εκπροσώπηση κοντινών συνοριακών περιοχών. Σίγουρα, ο ρυθμός μείωσης που επιτυγχάνεται με τον RSP3 εξαρτάται σε μεγάλο βαθμό από το επίπεδο θορύβου στα δεδομένα. Όσο υψηλότερο είναι το επίπεδο θορύβου στα δεδομένα, τόσο μικρότερα τα υποσύνολα που δημιουργούνται και, κατά συνέπεια, χαμηλότερος ρυθμός μείωσης. Αξίζει να σημειωθεί ότι η εύρεση των πιο απομακρυσμένων αντικειμένων στο κάθε υποσύνολο συνεπάγεται στον υπολογισμό όλων των αποστάσεων μεταξύ των στοιχείων του υποσυνόλου. Πρόκειται δηλαδή για δαπανηρές διαδικασίες που υποβαθμίζουν το συνολικό κόστος προεπεξεργασίας του αλγορίθμου. Σε περιπτώσεις μεγάλων συνόλων δεδομένων, το μειονέκτημα αυτό μπορεί να καταστήσει αδύνατη την εκτέλεσή του.

3.3.1 Εφαρμογή του RSP3 σε παράδειγμα

Έστω ένα σύνολο δεδομένων εκπαίδευσης που απαρτίζεται από δεδομένα τριών διακριτών κλάσεων A,B,C. Για ευκολία προσομοίωσης του παραδείγματος αποδίδουμε



Κλάση	Σύμβολο	Σημεία	Όρια x	Όρια y	Τύπος σημείων
A	o	133	39 - 68	61 - 75	Πραγματικοί
B	o	108	4 - 24	15 - 68	Πραγματικοί
C	o	120	3 - 37	14 - 53	Πραγματικοί

Σχήμα 3.3: Συμπύκνωση ενός συνόλου εκπαίδευσης τριών κλάσεων με τον RSP3

στα δεδομένα των τριών κλάσεων από δύο χαρακτηριστικά ώστε να είναι δυνατή η αναπαράσταση τους στον δισδιάστατο χώρο (σχήμα 3.3).

Το κριτήριο διαίρεσης που χρησιμοποιείται σε όλες τις υλοποιήσεις του RSP3 στην εργασία είναι η μέγιστη απόσταση. Είναι ιδιαίτερα εμφανής η μετατόπιση σημείων στο σύνολο συμπύκνωσης που δικαιολογείται από τον υπολογισμό των μέσων σημείων (παραγωγή πρωτότυπων) του αλγορίθμου μετά την εύρεση ομοιογενών υποσυνόλων . Ιδιαίτερα η κλάση A που είναι γραμμικά διαχωρίσιμη από τις υπόλοιπες, δηλαδή κανένα στοιχείο της δεν συμπίπτει με στοιχεία άλλης κλάσης, έχει μειωθεί σε επίπεδα άνω του 95% με μόνο δύο στοιχεία να παραμένουν στο σετ.

Οι κλάσεις B και C αναπτύσσονται ως ένα βαθμό στο ίδιο συγκρότημα του χώρου, πράγμα που υποδηλώνει πολύ χαμηλότερα ποσοστά μείωσης σε σύγκριση με την κλάση A. Παρατηρούμε πως υπάρχουν γειτονίες των δύο κλάσεων που δεν επικαλύπτονται η μια από την άλλη, οι οποίες και συμπυκνώθηκαν σημαντικά (συντεταγμένες y: 54-70 {B}, x: 24-41 {C}). Στα υπόλοιπα στοιχεία των δύο κλάσεων υπάρχει μικρή μείωση και ελάχιστες μετατοπίσεις, αφού ο αλγόριθμος προσπαθεί να αποτρέψει την αφαίρεση ή αλλοίωση δεδομένων που θα έχουν ως συνέπεια μικρότερα ποσοστά επιτυχής ταξινόμησης από τον k-NN.

3.4 Σύγκριση / Κίνητρο για βελτίωση του αλγορίθμου

Ο RSP3 συνδυάζει ένα αρκετά υψηλό ποσοστό μείωσης χωρίς σημαντική υποβάθμιση της ακρίβειας ταξινόμησης. Στην πραγματικότητα, για πολλές πρακτικές εφαρμογές, ο αλγόριθμος RSP3 θα μπορούσε να είναι ο καλύτερος αλγόριθμος μείωσης όσον αφορά την εξισορρόπηση της ακρίβειας με τη μείωση της αποθήκευσης. Ωστόσο, αν μια συγκεκριμένη εφαρμογή χρειάζεται να επιτύχει το υψηλότερο ρυθμό μείωσης, τότε οι εκδόσεις RSP1, RSP2 και ο CJA παρέχουν μια κατάλληλη λύση

επειδή παρουσιάζουν εξαιρετικά υψηλή ποσοστιαία μείωση, αν και αποφέρουν μέτρια απόδοση ταξινόμησης. Από τις τρεις προσεγγίσεις που προτείνονται, ο RSP3 δείχνει σαφώς την υψηλότερη ακρίβεια ταξινόμησης και παράλληλα αφαιρεί αρκετά στοιχεία εκπαίδευσης (78,49% κατά μέσο όρο), αν και τρέχει ουσιαστικά πιο αργά από οποιονδήποτε άλλον αλγόριθμο RSP (πρέπει να επαναληφθεί μέχρι όλα τα υποσύνολα να γίνουν ομοιογενή). Από την άλλη πλευρά, οι RSP1 και RSP2 εξαλείφουν το μεγαλύτερο ποσοστό των πρωτοτύπων και η μέση ακρίβεια είναι πάνω από 68%, καθώς επιτρέπουν στον χρήστη να επιλέξει το ποοστό της μείωσης που θα επιτευχθεί αλλά με αντίκτυπο την ακρίβεια. Όλα τα ποσοστά αντλούνται από την μελέτη του Sánchez [37].

Εάν λοιπόν ήταν εφικτό να μειωθεί δραστικά ο χρόνος επεξεργασίας που απαιτείται από τον RSP3, θα αποτελούσε ένα σημαντικό εύρημα τόσο για τον τομέα της μείωσης δεδομένων αλλά και της ταξινόμησης πρωτοτύπων. Στο επόμενο κεφάλαιο θα εισαχθούν δύο βασικές μεθοδολογίες που αποτελούν το πυλώνα της προσπάθειας ελαχιστοποίησης του φόρτου υπολογισμού μεγίστων αποστάσεων, που αποτελεί το βασικό αίτιο της συγκριτικά αργής εκτέλεσης του RSP3.

Κεφάλαιο 4

Τεχνικές εύρεσης μεγίστων αποστάσεων σε σύνολα δεδομένων

Περίληψη

Το ουσιαστικό βήμα για την ελαχιστοποίηση του χρόνου εκτέλεσης του αλγορίθμου RSP3 είναι η μείωση των Ευκλείδειων αποστάσεων που πρέπει να υπολογιστούν σε κάθε επανάληψη διαίρεσης υποσυνόλων, μέχρι την περάτωση του αλγορίθμου. Στο κεφάλαιο αυτό θα παρουσιαστεί αρχικά ο συμβατικός αλγόριθμος που χρησιμοποιείται από την κλασσική υλοποίηση του RSP3. Στην συνέχεια θα ενταχθούν δύο νέες τεχνικές: εύρεση μεγίστων αποστάσεων μέσω του Convex Hull και στην συνέχεια, μέσω της προσέγγισης του Convex Hull.

4.1 Ο συμβατικός αλγόριθμος

Έστω ένα υποψήφιο υποσύνολο δεδομένων προς διαίρεση μέσω του RSP3 που απαρτίζεται από N στοιχεία. Για την εύρεση των δύο πιο απομακρυσμένων στοιχείων του με το συμβατικό αλγόριθμο θα χρειαστούν υπολογισμοί Ευκλείδειων αποστάσεων $k = N-1+N-2+N-3+\ldots$. Πιο γενικευμένα, ο τύπος περιγράφεται από την παρακάτω σχέση:

$$k = \frac{N(N-1)}{2}$$

Στο σχήμα 4.1 παρουσιάζεται σε ψευδοκώδικα η δομή του αλγορίθμου. Εάν συμπεριλάβουμε όλα τα σημεία του υποσυνόλου σε δύο πανομοιότυπες παράλληλες στήλες και με απλές γραμμές ενώσουμε όλες τις Ευκλείδειες αποστάσεις μεταξύ των σημείων που πρέπει να υπολογιστούν για κάθε επανάληψη, προκύπτει ένα πλέγμα (grid) από όπου και προέρχεται η ονομασία του.

```
Grid
 1: D_max ← 0                                  ▷ Variable to store max distance
 2: i ← 0
 3: while i < SS do                                          ▷ SS: Subset Size
 4:     j ← i + 1
 5:     while j < SS do
 6:         D_curr ← ED(p_i, p_j)                ▷ ED: Euclid Distance procedure
 7:         if D_curr > D_max then
 8:             D_max ← D_curr
 9:             pos_1 ← i
10:             pos_2 ← j
11:         end if
12:         j ← j + 1
13:     end while
14:     i ← i + 1
15: end while
16: return pos_1, pos_2, D_max          ▷ Return positions of elements and distance
```

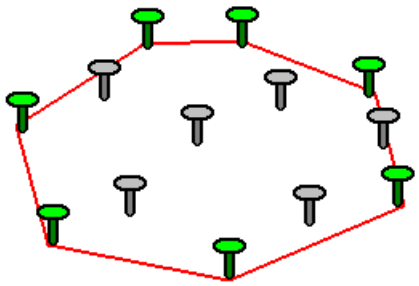
Σχήμα 4.1: Ψευδοκώδικας του συμβατικού αλγόριθμου Grid

Τα πλεονεκτήματα του αλγορίθμου είναι κυρίως η απλότητα του που καθιστά δυνατή την ενσωμάτωση του σε οποιαδήποτε υπολογιστική συσκευή, αλλά και η δυνατότητα εφαρμογής του σε κάθε σύνολο δεδομένων ασχέτως τον αριθμό των διαστάσεων του, που προκύπτει από τον αριθμό των χαρακτηριστικών των στοιχείων του. Επιπλέον, τα σημεία που επιστρέφονται από τον αλγόριθμο είναι πάντοτε αυτά με τη μεγαλύτερη απόσταση στο υποσύνολο που σημαίνει πως δεν είναι προσεγγιστικός. Η πολυπλοκότητα της εκτέλεσης του όμως είναι υψηλή με αποτέλεσμα να είναι πολύ αργός και να επιβραδύνει σημαντικά τη διαδικασία διαίρεσης υποσυνόλων του RSP3.

4.2 Convex Hull

4.2.1 Η βασική έννοια

Στα μαθηματικά, το Convex Hull (κυρτό περίβλημα) ενός συνόλου S σημείων στο ευκλείδειο επίπεδο ή σε έναν ευκλείδειο χώρο είναι το μικρότερο κυρτό σύνολο που περιέχει το S. Για παράδειγμα, όταν το S είναι ένα οριοθετημένο υποσύνολο του επιπέδου, το Convex Hull του μπορεί να απεικονιστεί ως το σχήμα που περικλείεται από μια λαστιχένια ζώνη που εκτείνεται γύρω από το S [14]. Τυπικά, το Convex



Σχήμα 4.2: Απεικόνιση του Convex Hull

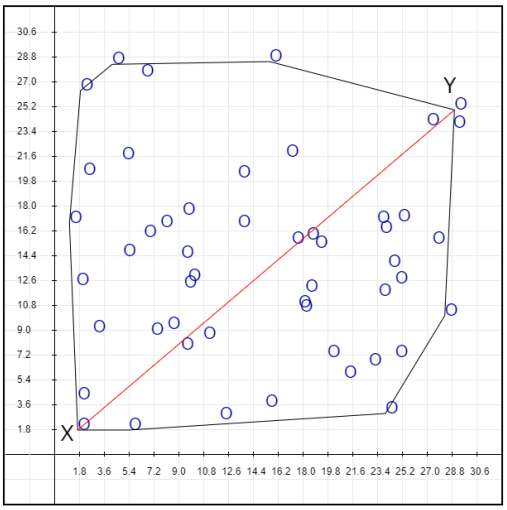
Hull μπορεί να οριστεί ως η τομή όλων των κυρτών συνόλων που περιέχουν το X ή ως σύνολο όλων των κυρτών συνδυασμών σημείων στο X. Με τον τελευταίο

ορισμό, τα Convex Hull μπορούν να επεκταθούν από τους Ευκλείδειους χώρους και να γενικευτούν σε αυθαίρετους πραγματικούς διανυσματικούς χώρους [41].

Μια ιδιαίτερα σημαντική ιδιότητα του Convex Hull ενός συνόλου δεδομένων είναι το γεγονός πως συμπεριλαμβάνει τα σημεία που σχηματίζουν τη διάμετρο του συνόλου. Εάν x, y τα δύο πιο απομακρυσμένα σημεία εντός ενός συνόλου, τα x, y είναι αυστηρά κυρτά (convex) σημεία, δηλαδή ανήκουν στο Convex Hull. Συνεπώς η εκτέλεση του συμβατικού αλγορίθμου (Grid) σε ένα υποσύνολο S και στο Convex Hull του S (S_{CH}) θα επιστρέψει τα ίδια σημεία, σε πολύ ταχύτερο χρόνο. Για να αξίζει η εισαγωγή του επιπλέον βήματος εύρεσης του Convex Hull στον RSP3 και στη συνέχεια η αναζήτηση μέσω Grid των πιο απομακρυσμένων σημείων σε αυτό αντί για το S, θα πρέπει να ισχύει:

$$T_{S_{CH}} + T_{GRID(S_{CH})} << T_{GRID(S)}$$

που ουσιαστικά σημαίνει πως το άθροισμα του χρόνου για την εύρεση του Convex Hull ενός συνόλου S με το χρόνο εκτέλεσης του Grid σε αυτό, πρέπει να είναι σημαντικά μικρότερο από το χρόνο εκτέλεσης του Grid σε ολόκληρο το S.



Σχήμα 4.3: Τα x, y σημεία της διαμέτρου είναι κυρτά σημεία

4.2.2 Μεθοδολογίες εύρεσης

Το αλγοριθμικό πρόβλημα της εύρεσης του Convex Hull ενός πεπερασμένου συνόλου σημείων στο επίπεδο ή σε άλλους ευ μεγέθους χώρους χαμηλών διαστάσεων είναι ένα από τα θεμελιώδη προβλήματα της υπολογιστικής γεωμετρίας. Στην υποενότητα αυτή, θα παρουσιαστούν γνωστοί αλγόριθμοι για την αντιμετώπιση του προβλήματος. Για κάθε έναν θα δοθεί η υλοποίηση του σε ψευδοκώδικα και η πολυπλοκότητα των εντολών του.

4.2.2.1 Graham Scan

Η σάρωση του Graham είναι μια μέθοδος για την εύρεση του Convex Hull ενός πεπερασμένου συνόλου σημείων στο επίπεδο. Ονομάστηκε από τον Ronald Graham, ο οποίος δημοσίευσε τον αρχικό αλγόριθμο το 1972 [28]. Ο αλγόριθμος βρίσκει όλες τις κορυφές του Convex Hull που διατάσσονται κατά μήκος των ορίων του. Χρησιμοποιεί μια στοίβα για την ανίχνευση και την αποτελεσματική απομάκρυνση των κοιλοτήτων στα όρια. Στο σχήμα 4.4 παρουσιάζεται ο ψευδοκώδικας του.

Graham Scan

```
1: procedure CCW(p_1, p_2, p_3)          ▷ Check if p_1, p_2 and p_3 are clockwise
2:     return (p_2x − p_1x) × (p_3y − p_1y) − (p_2y − p_1y) × (p_3x − p_1x)
3: end procedure

4: let N be number of points
5: let points[N] be the array of points
6: let S be a stack to store points

7: points[0] ← p_{y_min}
8: sort points[N] by polar angle with points[0]
9: S ← ∅
10: push points[0], points[1] to S
11: for i = 2 to N − 1 :
12:     while S_size ≥ 2 and CCW(S_{top+1}, S_top, points[i]) ≤ 0 : pop S
13:     push points[i] to S
14: end for
15: return S
```

Σχήμα 4.4: Ο ψευδοκώδικας της σάρωσης του Graham

Το πρώτο βήμα του αλγορίθμου είναι η εύρεση του σημείου με τη χαμηλότερη συντεταγμένη y. Εάν η χαμηλότερη συντεταγμένη y υπάρχει σε περισσότερα από ένα σημεία του συνόλου, επιλέγεται το σημείο με τη χαμηλότερη συντεταγμένη x από τους υποψηφίους. Το σημείο αυτό ονομάζεται P. Αυτό το βήμα χρειάζεται $O(n)$ χρόνο, όπου n είναι ο αριθμός των εν λόγω σημείων.

Στη συνέχεια, το σύνολο των σημείων πρέπει να ταξινομηθεί με αυξανόμενη σειρά ανάλογα της γωνίας που σχηματίζει το κάθε σημείο με το σημείο P στον άξονα x. Οποιοσδήποτε αλγόριθμος ταξινόμησης γενικού σκοπού είναι κατάλληλος. Η ταξινόμηση κατά σειρά γωνίας δεν απαιτεί υπολογισμό της γωνίας. Είναι δυνατή η χρήση οποιασδήποτε συνάρτησης της γωνίας που είναι μονοτονική στο διάστημα $[0, \pi]$.

Ο αλγόριθμος προχωρά εξετάζοντας κάθε ένα από τα σημεία του ταξινομημένου πίνακα σε σειρά. Για κάθε σημείο, καθορίζεται πρώτα κατά πόσον η μετακίνηση από τα δύο σημεία που προηγούνται αμέσως του σημείου αυτού αποτελεί μια αριστερή στροφή ή μια δεξιά στροφή. Εάν η στροφή είναι δεξιά, το δεύτερο μέχρι το τελευταίο σημείο δεν είναι μέρος του Convex Hull, και βρίσκονται «μέσα» του. Ο ίδιος προσδιορισμός επαναλαμβάνεται για το σύνολο του τελευταίου σημείου και για τα δύο σημεία που προηγούνται αμέσως του σημείου που βρέθηκε ότι ήταν μέσα στο Convex Hull και επαναλαμβάνεται μέχρις ότου συναντήσουμε ένα «αριστερόστροφο» σύνολο, οπότε ο αλγόριθμος κινείται στο επόμενο σημείο του ταξινομημένου συνόλου μείον των σημείων που βρέθηκαν να βρίσκονται μέσα στο Convex Hull. Δεν χρειάζεται να επανεξετάσουμε αυτά τα σημεία. Εάν σε οποιοδήποτε στάδιο τα τρία σημεία είναι συγγραμικά, μπορεί κάποιος να επιλέξει να τα απορρίψει ή να τα αναφέρει, αφού σε ορισμένες εφαρμογές απαιτείται να βρεθούν όλα τα σημεία στο όριο του Convex Hull. Η συνολική πολυπλοκότητα του αλγορίθμου είναι $O(n\ log\ n)$ όπου n ο αριθμός των στοιχείων στο σύνολο δεδομένων.

4.2.2.2 Ο αλγόριθμος του Chan

Ο αλγόριθμος του Chan, ονομάζεται από τον Timothy M. Chan [58] και είναι ένας βέλτιστος αλγόριθμος με ευαισθησία εξόδου για τον υπολογισμό του Convex Hull ενός συνόλου δεδομένων. Αυτό σημαίνει πως η ταχύτητα της εκτέλεσης του εξαρτάται από το μέγεθος της εξόδου που θα παράγει και όχι από αυτό της εισόδου ως συνήθως. Ο αλγόριθμος παίρνει χρόνο $O(n\ log\ h)$, όπου h είναι ο αριθμός των κορυφών της εξόδου ενώ n τα δεδομένα του συνόλου. Συνδυάζει τη σάρωση του Graham με την πορεία του Jarvis [45] για να επιτύχει τον προαναφερόμενο χρόνο. Ο αλγόριθμος του Chan είναι αξιοσημείωτος επειδή είναι απλούστερος και φυσικά εκτείνεται στον τρισδιάστατο χώρο. Ο ψευδοκώδικας του σχήματος 4.4 αναπτύχθηκε ανεξάρτητα από τον Frank Nielsen [22].

Chan's Algorithm

Input: a list S of bidimensional points
Output: the convex hull of the set
for $i = 1, 2, \ldots$ **do**
    L = ChanHullStep(S, m, H), where $m = H = min\{|S|, 2^{2^i}\}$
    if $L \neq$ *incomplete* **then** return L

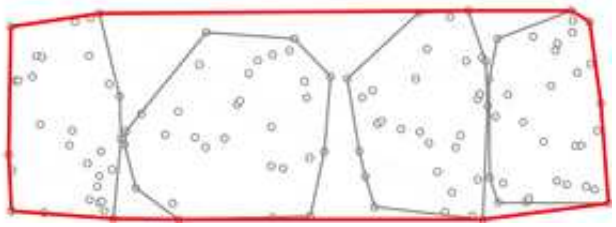
procedure ChanHullStep(S, m, H)
    Input: a list S of bidimensional points, the parameters m, H
    Output: the convex hull of the set, sorted counterclockwise, or an empty list, if H is $< h$
    Partition S into subsets $S_1, \ldots, S_{\lceil n/m \rceil}$.
    for $i = 1, \ldots, \lceil n/m \rceil$ **do**
        Compute the convex hull of S_i by using Graham Scan, store the output in a counter-clockwise sorted list.
    $p_0 = (0, -\infty)$
    $p_1 =$ the leftmost point of S.
    for $k = 1, \ldots, H$ **do**
        for $i = 1, \ldots, \lceil n/m \rceil$ **do**
            Compute the points $q_i \in S$ that maximizes $\angle p_{k-1} p_k q_i$, with $q_i \neq p_k$, by performing binary search on the vertices of the partial hull S_i.
        $p_{k+1} =$ the point $q \in \{q_1, \ldots, q_{\lceil n/m \rceil}\}$.
        if $p_{k+1} = p_t$ **then** return $\{p_1, \ldots, p_k\}$
    return *incomplete*
end procedure

Σχήμα 4.5: Ο ψευδοκώδικας του αλγόριθμου του Chan

Ένα μόνο πέρασμα του αλγορίθμου απαιτεί μια παράμετρο $m \geq h$ για να τερματίσει επιτυχώς. Η τιμή του h δεν είναι γνωστή εκ των προτέρων και πολλαπλά περάσματα με αυξανόμενες τιμές m θα χρησιμοποιηθούν, όπως φαίνεται και στο σχήμα. Ο αλγόριθμος ξεκινά με την αυθαίρετη κατάτμηση του συνόλου σημείων S σε $K \leq 1 + n/m$ υποσύνολα Q_k όπου $k = 1, 2, \ldots, K$ με το πολύ m σημεία το καθένα. Να σημειωθεί πως $K = O(n/m)$.

Στην πρώτη φάση για κάθε υποσύνολο Q_k υπολογίζεται το Convex Hull του (C_k) χρησιμοποιώντας τη σάρωση του Graham. Στην δεύτερη φάση εκτελείται η πορεία του Jarvis η οποία κάνει χρήση των Convex Hull που δημιουργήθηκαν στο προηγούμενο βήμα. Σε κάθε επανάληψη, υπάρχει ένα σημείο p_i (το οποίο αρχικά μπορεί να είναι το σημείο με το μικρότερο y καθώς είναι σίγουρο πως ανήκει στο Convex Hull) και πρέπει να βρεθεί το σημείο p_{i+1} ώστε όλα τα υπόλοιπα σημεία στο S να είναι δεξιά της γραμμής που σχηματίζουν τα p_i, p_{i+1}. Με την εύρεση του σημείου αυτού είναι εφικτό μέσω δυαδικής αναζήτησης να βρεθούν τα υπόλοιπα σημεία του ευρύτερου Convex Hull.

Εάν μια αυθαίρετη τιμή έχει οριστεί για το m, μπορεί να ισχύει ότι $m < h$. Σε αυτή την περίπτωση, μετά από m βήματα στη δεύτερη φάση, διακόπτεται η πορεία του Jarvis καθώς η εκτέλεση μέχρι το τέλος θα χρειαζόταν πολύ χρόνο. Σε μια τέτοια περίπτωση, θα έχει δαπανηθεί χρόνος $O(n\ log\ m)$ χωρίς να υπολογιστεί το Convex Hull. Η ιδέα είναι να κάνουμε πολλαπλά περάσματα του αλγορίθμου με αυξανόμενες τιμές m έτσι ώστε να βρεθεί η βέλτιστη χωρίς να χαθεί χρόνος.



Σχήμα 4.6: Οπτικό παράδειγμα του αλγόριθμου του Chan

4.2.2.3 Η μέθοδος Quick Hull

Η μέθοδος της γρήγορης (quick) εύρεσης του Convex Hull ενός συνόλου δεδομένων αποτελεί μια ευρέως διαδεδομένη λύση που προτάθηκε από τον Barber [6]. Αναβαθμίζεται και συντηρείται ως βιβλιοθήκη στην γλώσσα C από το Geometry Center στο qhull.org όπου υπάρχει επεξηγηματικό υλικό και αναλυτικές οδηγίες χρήσης. Χρησιμοποιείται σε πολλές εφαρμογές όπως στο MatLab ή το Octave [20; 48]. Επιπλέον, όπως θα παρουσιαστεί και στο επόμενο κεφάλαιο, μια υλοποίηση του αλγορίθμου χρησιμοποιήθηκε για την πρώτη παραλλαγή του RSP3 στο πλαίσιο της εργασίας.

Algorithm 1 Quick Hull

Input: a set S of n points, where $n \geq 2$

1: $CH \leftarrow \varnothing$
2: $A \leftarrow$ left most point
3: $B \leftarrow$ right most point
4: add A, B to CH
5: $S_1 \leftarrow$ points on the right side of line AB
6: $S_2 \leftarrow$ points on the right side of line BA
7: FindHull(S_1,A,B)
8: FindHull(S_2,B,A)
9: **return** CH

Find points on convex hull from the set S_k of points that are on the right side of the oriented line from P to Q

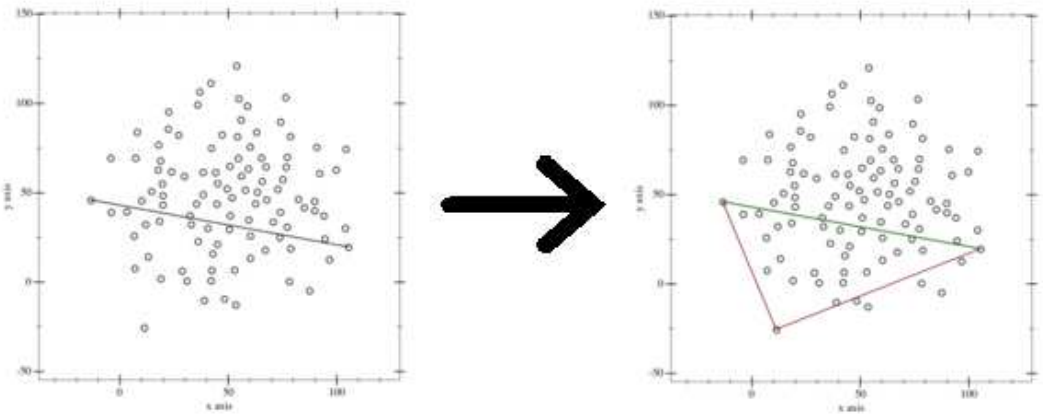
10: **procedure** FINDHULL(S_k, P, Q)
11: **if** $S_k = \varnothing$ **return**
12: $C \leftarrow$ farthest point from segment PQ
13: add C to CH between P and Q
14: $S_0 \leftarrow$ points inside triangle PCQ
15: $S_1 \leftarrow$ points on the right side of line PC
16: $S_2 \leftarrow$ points on the right side of line CQ
17: FindHull(S_1,P,C)
18: FindHull(S_2,C,Q)
19: **end procedure**

Σχήμα 4.7: Ο ψευδοκώδικας της μεθόδου Quick Hull

Αλγοριθμικά η μέθοδος παρουσιάζεται σε ψευδοκώδικα στο σχήμα 4.7. Χρησι-

μοποιεί μια προσέγγιση «διαίρει και βασίλευε» (divide and conquer) παρόμοια με εκείνη της quicksort, από την οποία προέρχεται το όνομά της. Η μέση πολυπλοκότητα της θεωρείται ότι είναι $O(n\ log\ n)$, ενώ στη χειρότερη περίπτωση παίρνει $O(n^2)$. Ωστόσο, σε αντίθεση με την quicksort, δεν υπάρχει προφανής τρόπος για τη μετατροπή της σε τυχαιοποιημένο αλγόριθμο. Έτσι, η μέση πολυπλοκότητα του χρόνου δεν μπορεί να υπολογιστεί εύκολα.

Υπό κανονικές συνθήκες, ο αλγόριθμος λειτουργεί ικανοποιητικά, αλλά η επεξεργασία συνήθως γίνεται αργή σε περιπτώσεις υψηλής συμμετρίας ή σημείων που βρίσκονται στην περιφέρεια ενός κύκλου. Ο αλγόριθμος μπορεί να αναλυθεί στα παρακάτω βήματα: Αρχικά την εύρεση των σημείων με τις ελάχιστες και μέγιστες συντεταγμένες x, καθώς αυτά πάντοτε ανήκουν στο Convex Hull.Έπειτα χρησιμοποιείται η γραμμή που σχηματίζεται από τα δύο σημεία για να διαιρεθεί το σετ σε δύο υποσύνολα σημείων, τα οποία θα επεξεργαστούν αναδρομικά. Στην συνέχεια πρέπει να προσδιοριστεί το σημείο, στη μία πλευρά της γραμμής, με τη μέγιστη απόσταση από τη γραμμή. Αυτό το σημείο σχηματίζει ένα τρίγωνο με τα σημεία της γραμμής. Τα σημεία που βρίσκονται στο εσωτερικό αυτού του τριγώνου δεν μπορο-



Σχήμα 4.8: Οπτικό παράδειγμα κατασκευής της Quick Hull

ύν να αποτελούν μέρος του Convex Hull και επομένως μπορούν να αγνοηθούν στα

επόμενα βήματα. Ακολουθεί επανάληψη των προηγούμενων δύο βημάτων στις δύο γραμμές που σχηματίζονται από το τρίγωνο (όχι την αρχική γραμμή). Η διαδικασία επαναλαμβάνεται έως ότου δεν μείνουν άλλα σημεία, η αναδρομή περατώνεται και τα επιλεγμένα σημεία αποτελούν το Convex Hull.

4.2.3 Πρόβλημα γενίκευσης στον πολυδιάστατο χώρο

Εάν και υπάρχουν αρκετοί αλγόριθμοι για την εύρεση του Convex Hull ενός συνόλου, λίγοι από αυτούς επεκτείνονται στον τρισδιάστατο χώρο και ακόμα λιγότεροι στον πολυδιάστατο [1]. Ως ένα σύνθετο πρόβλημα υπολογιστικής γεωμετρίας το οποίο ταυτοχρόνως είναι ευαίσθητο σε ιδιαίτερες περιπτώσεις συνόλων, η εισαγωγή του σε έναν αλγόριθμο μείωσης δεδομένων γενικής χρήσεως για κάθε πλήθους πραγματικών χαρακτηριστικών συνόλου είναι αρκετά δύσκολη.

Σχήμα 4.9: 3D Αναπαράσταση του Convex Hull

Το Geometry Center έχει γενικεύσει τον αλγόριθμο Quick Hull σε πολλαπλές διαστάσεις, αλλά τίθενται πολλαπλοί περιορισμοί για το σύνολο προς εισαγωγή ώστε να είναι αποδεκτό από την εφαρμογή. Παράλληλα, είναι άγνωστο εάν ο υπολογισμός του Convex Hull θα είναι επιτυχής, ακόμα και εάν το σύνολο τηρεί τις κατάλληλες

προϋποθέσεις. Υψηλό φόρτο επεξεργασίας προσθέτει και η σειριοποίηση των σημείων που ανήκουν στο Convex Hull, δηλαδή όχι απλά η εύρεση τους αλλά και η αποθήκευση με τη σωστή σειρά που είναι διατεταγμένα σε αυτό. Να σημειωθεί πως η διαδικασία αυτή δεν έχει κανένα όφελος για την εύρεση μεγίστων αποστάσεων. Τέλος, ακόμα και εάν δεν υπήρχαν τα προβλήματα που προαναφέρθηκαν, οι δημιουργοί τις βιβλιοθήκης αναφέρουν πως οι απαιτήσεις μνήμης όσο αυξάνονται οι διαστάσεις είναι ιδιαίτερα υψηλές και η συνολική πολυπλοκότητα είναι άγνωστη καθώς εξαρτάται σημαντικά από το σύνολο δεδομένων.

Προκύπτει λοιπόν πως η προσπάθεια δημιουργίας μιας παραλλαγής του RSP3 κάνοντας χρήση του αλγορίθμου Quick Hull θα είναι προβληματική σε σύνολα δεδομένων με πολλαπλά χαρακτηριστικά αφού εκτός των πιθανών προβλημάτων κατά τη φάση των υπολογισμών είναι πιθανό η συνολική ταχύτητα επεξεργασίας και οι απαιτήσεις μνήμης να ξεπεράσουν αυτές του συμβατικού αλγορίθμου, χρήζοντας την απόπειρα ανώφελη. Για την αντιμετώπιση των ζητημάτων αυτών, θα παρουσιαστούν στο επόμενο κεφάλαιο τεχνικές επιλογής χαρακτηριστικών από σύνολα δεδομένων που αποσκοπούν στην τεράστια μείωση του πλήθους των διαστάσεων ενός συνόλου με στόχο την εύρυθμη λειτουργία του Quick Hull.

4.3 Προσέγγιση (Approximation) του Convex Hull

Η επιλογή των διαστάσεων του συνόλου δεδομένων προς μείωση αποτέλεσε μια καλή ιδέα για την αντιμετώπιση του προβλήματος της γενίκευσης του Quick Hull αλλά δεν έπαψε να αποτελεί ουσιαστικά έναν συμβιβασμό στην φύση του αλγορίθμου. Η ανάγκη για μια τεχνική που μπορεί να εφαρμοστεί σε κάθε πλήθος χαρακτηριστικών-διαστάσεων ικανοποιήθηκε από μια νέα μεθοδολογία που θα παρουσιαστεί σε αυτήν την ενότητα, η οποία βασίζεται στα συμπεράσματα που προκύπτουν από τη μελέτη

του RSP3 και των τεχνικών εύρεσης του Convex Hull.

4.3.1 Κατευθυντήρια πορίσματα

Δύο είναι τα θεμελιώδη πορίσματα που οδήγησαν στην γέννηση του αλγορίθμου προσέγγισης του Convex Hull και κατ΄ επέκταση της μέγιστης απόστασης ενός συνόλου δεδομένων.

Πόρισμα Α

Στο κεφάλαιο 3 όπου μελετήθηκε η οικογένεια αλγορίθμων RSP, αναφέρθηκε το κριτήριο διαίρεσης των υποσυνόλων, το οποίο είναι είτε η μέγιστη απόσταση είτε ο βαθμός επικάλυψης του υποσυνόλου. Μια ιδιαίτερη ιδιότητα του αλγορίθμου RSP3 είναι πως η ακέραια περάτωση του αλγορίθμου δεν εξαρτάται αυστηρά από το εάν η απόσταση που δίνεται ως μέγιστη είναι πράγματι αυτή. Σε ένα υποθετικό σενάριο όπου η διαίρεση υποσυνόλων γίνεται τυχαία χωρίς κριτήριο, κατά μέσο όρο, ο αλγόριθμος θα επιστρέψει σετ συμπύκνωσης, αλλά θα χρειαστεί περισσότερο χρόνο λόγω επιπλέον διαιρέσεων, με πολύ μικρότερο ποσοστό μείωσης. Εμπειρικά όμως διαψεύδεται ότι η διαίρεση υποσυνόλων με γνώμονα τη μέγιστη απόσταση επιστρέφει το βέλτιστο αποτέλεσμα μείωσης, αλλά διασφαλίζει ότι θα τείνει προς αυτό. Άρα, προκύπτει πως δεν είναι αναγκαίο να υπολογιστεί η μέγιστη απόσταση ενός υποσυνόλου, αλλά όσο πλησιέστερη στο μέγεθος της είναι η απόσταση που εν τέλει θα χρησιμοποιηθεί, τόσο πιο κοντά στο βέλτιστο συνδυασμό χρόνου / ποσοστού μείωσης θα είναι το τελικό αποτέλεσμα.

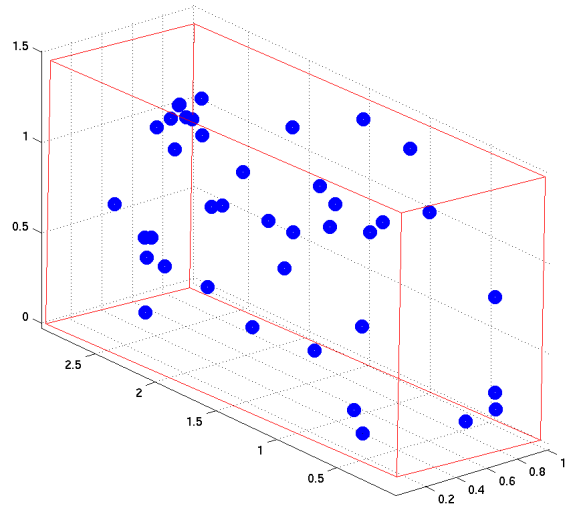
Πόρισμα Β

Σε κάθε αλγόριθμο εύρεσης του Convex Hull παρατηρήθηκε η ανάγκη ενός ή περισσοτέρων σημείων αφετηρίας τα οποία κατά κανόνα ανήκουν σε αυτό. Σε όλες τις περιπτώσεις τα σημεία αυτά ήταν εκείνα με την ελάχιστη ή μέγιστη τιμή σε μία από

τις διαστάσεις του συνόλου (συνήθως το x). Προκύπτει λοιπόν, πως η εύρεση των σημείων με την ελάχιστη και μέγιστη τιμή για κάθε χαρακτηριστικό-διάσταση έχει ως αποτέλεσμα την παραγωγή ενός υποσυνόλου το οποίο μπορεί να αντικαταστήσει το Convex Hull στην προσπάθεια εύρεσης μιας κατάλληλης απόστασης. Στην καλύτερη περίπτωση η απόσταση αυτή θα είναι και η μέγιστη, ενώ στην χειρότερη δεν θα είναι πολύ μικρότερη της, δηλαδή θα την προσεγγίζει.

4.3.2 Ο αλγόριθμος Margin Hull

Οι προεκτάσεις των μεγίστων και ελαχίστων σημείων ενός συνόλου δεδομένων προς όλες τις διαστάσεις του ορίζουν το "περιθώριο" (margin) του, δηλαδή το παραλληλόγραμο ορθογώνιο που περιβάλει το χώρο μέσα στον οποίο βρίσκονται όλα τα σημεία του. Ως εκ τούτου και η ονομασία του αλγορίθμου εύρεσης των σημείων αυτών (σχ. 4.10). Να σημειωθεί πως το περιθώριο ενός συνόλου δεδομένων με το περιθώριο του Convex Hull του συνόλου αυτού ταυτίζονται.



Σχήμα 4.10: Το ορθογώνιο περίβλημα ενός συνόλου

Ο ψευδοκώδικας του αλγορίθμου παρουσιάζεται στο σχήμα 4.11. Ως είσοδο, δέχεται μια δομή δεδομένων SS που περιέχει τα σημεία ενός υποσυνόλου, με κάθε

Margin Hull

Input: Subset SS of d dimensions

1: $MMI[d][2] \leftarrow 0$ ▷ Array holding minimum and maximum data indexes

Set first element (index 0) as min and max for all dimensions
2: **for** $i = 0\ i < d\ i{+}{+}$ **do** $MMI[i][0] = MMI[i][1] \leftarrow 0$

Find and store min and max indexes for every dimension
3: **for** $i = 1\ i < SS_{size}\ i{+}{+}$ **do**
4: **for** $j = 0\ j < d\ j{+}{+}$ **do**
5: **if** $SS[i][j] < MMI[j][0]$ **then** $MMI[j][0] = i$
6: **else if** $SS[i][j] > MMI[j][1]$ **then** $MMI[j][1] = i$
7: **end if**
8: **end for**
9: **end for**

Remove duplicate instances and return the margin hull
10: **return** $min \cup max$ from MMI

Σχήμα 4.11: Ο ψευδοκώδικας του Margin Hull

σημείο να αποτελείται από d χαρακτηριστικά-διαστάσεις. Σε μια άλλη δομή MMI (min max indexes) με πλήθος d γραμμών και 2 στηλών αποθηκεύονται σε κάθε γραμμή η θέση στο SS του ελάχιστου και μέγιστου σημείου για την αντίστοιχη διάσταση. Το MMI αρχικοποιείται θέτοντας το πρώτο σημείου του SS ως ελάχιστο και μέγιστο για κάθε διάσταση. Στην συνέχεια, με ένα πέρασμα (one pass) του SS ολοκληρώνεται η διαδικασία ανάθεσης των σημείων που ανήκουν στο περιθώριο και συμπληρώνεται η δομή MMI.

Πριν επιστραφεί το MMI από τον αλγόριθμο, προτείνεται η χρήση μιας οποιασδήποτε γενικού τύπου μεθόδου αφαίρεσης διπλοτύπων, καθώς υπάρχει πάντα η πιθανότητα ένα σημείο να αποτελεί ελάχιστο ή μέγιστο για περισσότερες από μια διαστάσεις, ιδιαίτερα όσο μικραίνει το πλήθος των στοιχείων του υποσυνόλου. Αφαιρώντας τα διπλότυπα μειώνεται σημαντικά ο αριθμός των αποστάσεων που που θα χρειαστεί να υπολογιστούν στην συνέχεια από το συμβατικό αλγόριθμο Grid.

Η φιλοσοφία του αλγορίθμου είναι απλή και στοχεύει στην όσο ταχύτερη ολοκλήρωση των διαδικασιών του. Η πολυπλοκότητα του αλγορίθμου είναι $O(n)$ καθώς πρωτεύοντα ρόλο έχει το μέγεθος του υποσυνόλου που δίνεται ως είσοδο, ενώ ο αριθμός των διαστάσεων έχει μικρή επίπτωση στο πλήθος των εντολών που θα εκτελεστούν. Ιδιαίτερο πλεονέκτημα του αλγορίθμου πέρα από την απλότητα είναι η επεκτασιμότητα του σε υποσύνολα οποιονδήποτε διαστάσεων, αλλά και το σταθερό μέγεθος του συνόλου σημείων που επιστρέφει, το οποίο θα είναι πάντα το διπλάσιο των διαστάσεων της εισόδου $(2 \times d)$ ασχέτως του πλήθους των στοιχείων της.

4.4 Ενσωμάτωση νέων τεχνικών στον RSP3

Απο τους αλγορίθμους που παρουσιάστηκαν στο κεφάλαιο αυτό δύο χρησιμοποιήθηκαν για τη δημιουργία νέων παραλλαγών του RSP3. Η πρώτη παραλαγή κάνει χρήση της μεθοδολογίας του Convex Hull με τον αλγόριθμο Quick Hull και ονομάζεται RSPQ (Reduction By Space Partitioning using Quick Hull). Αντίστοιχα, η δεύτερη παραλλαγή χρησιμοποιεί την τεχνική προσέγγισης του Convex Hull με τον αλγόριθμο Margin Hull και ονομάζεται RSPA (Reduction by Space Partitioning using Approximation). Στα ακόλουθα κεφάλαια θα παρουσιαστούν τα πειραματικά αποτελέσματα των δύο νέων παραλλαγών και θα συγκριθούν με αυτά του συμβατικού αλγορίθμου αλλά και μεταξύ τους.

Κεφάλαιο 5

Πειραματική μελέτη των δύο νέων παραλλαγών του **RSP3**

Περίληψη

Στο κεφάλαιο αυτό θα εισαχθούν δύο νέες παραλλαγές του RSP3. Για την εύρεση μεγίστων αποστάσεων η πρώτη χρησιμοποιεί τον αλγόριθμο Quick Hull, ενώ η δεύτερη τον Margin Hull. Και οι δύο έχουν υλοποιηθεί στην γλώσσα C++ και δοκιμάστηκαν με πραγματικά σύνολα δεδομένων. Θα αναλυθεί ο τρόπος της διεξαγωγής των πειραμάτων και θα παρουσιαστούν τα αποτελέσματα της κάθε μίας.

5.1 Τα δεδομένα του πειράματος

Τα σύνολα δεδομένων τα οποία χρησιμοποιήθηκαν στις πειραματικές μελέτες αποτελούνται από πραγματικά δεδομένα και αντλήθηκαν από το KEEL-Dataset repository [3]. Στον πίνακα 5.1 περιέχεται το όνομα του κάθε συνόλου πραγματικών δεδομένων, με το πλήθος των στοιχείων, των χαρακτηριστικών του και των κλάσεων του. Όπως παρατηρείται, υπάρχει ποικιλία στο είδος των συνόλων τόσο στο πλήθος στοιχείων,

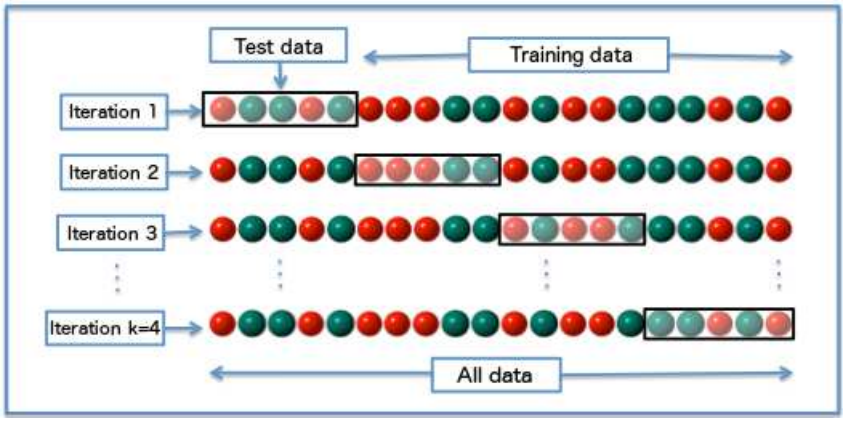
Όνομα	Στοιχεία	Διαστάσεις	Κλάσεις
$Banana(BN)$	5300	2	2
$EyeState(EEG)$	14980	14	2
$KDDCup(KDD)$	494020	40	23
$LetterImageRecognition(LIR)$	20000	16	26
$LandsatSatellite(LS)$	6435	36	7
$MagicGammaTelescope(MGT)$	19020	11	2
$PenDigits(PD)$	10992	16	10
$Phoneme(PH)$	5404	5	2
$Ring(RNG)$	7400	20	2
$Segment(SG)$	2310	19	7
$Shuttle(SH)$	58000	9	7
$Twonorm(TN)$	7400	20	2
$Textrue(TXR)$	5500	40	11
$Waveform(WF)$	5000	21	3
$WineQualityWhite(WQW)$	4898	11	11
$Yeast(YST)$	1484	8	10

Πίνακας 5.1: Τα σύνολα δεδομένων του πειράματος

χαρακτηριστικών αλλά και κλάσεων. Όλα τα σύνολα αποτελούνται από αριθμητικά (ποσοτικά) χαρακτηριστικά και έχουν κανονικοποιηθεί στο εύρος $[0, 1]$. Στα σύνολα δεν περιλαμβάνονται άγνωστες τιμές ή κενά καθώς δεν υπάρχει συμβατότητα του RSP3 για παρόμοιες περιπτώσεις.

5.2 Επικύρωση με k-Fold Cross validation

Για να είναι ουσιαστική η δοκιμή ενός πειραματικού μοντέλου προβλέψεων που παράχθηκε από ένα σετ εκπαίδευσης μέσω οποιουδήποτε αλγορίθμου είναι θεμιτό να χρησιμοποιηθούν άγνωστα δεδομένα που δεν συμπεριλήφθηκαν κατά την διαδικασία της εκπαίδευσης. Με τον τρόπο αυτό επικυρώνεται το πείραμα, αξιολογείται αντικειμενικά η ικανότητα του μοντέλου να γενικεύσει το πρόβλημα και αποφεύγεται το ενδεχόμενο της υπερεκπαίδευσης.



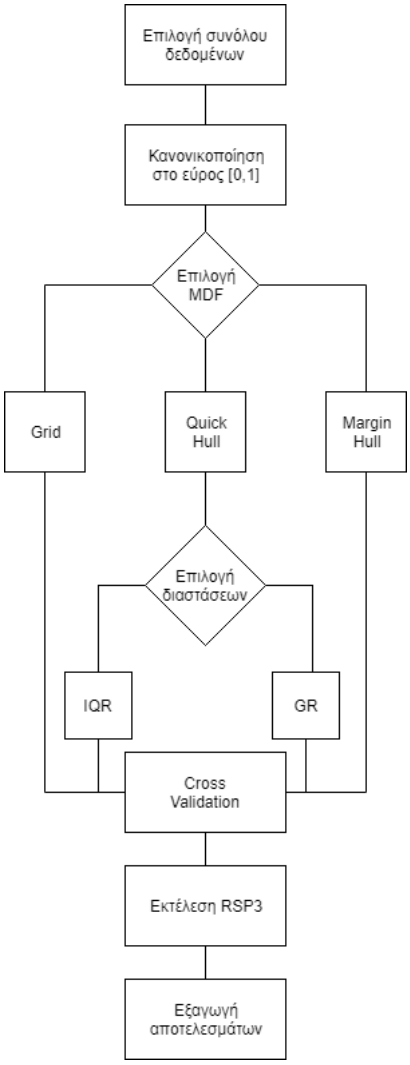
Σχήμα 5.1: Παράδειγμα επικύρωσης με k-Fold Cross Validation

Με την τεχνική επικύρωσης k-Fold Cross validation [42], το αρχικό δείγμα χωρίζεται τυχαία σε δείγματα k ίσου μεγέθους. Από τα δείγματα k, διατηρείται ένα δείγμα ως δεδομένα επικύρωσης για τη δοκιμή του μοντέλου και τα υπόλοιπα $k-1$ δείγματα χρησιμοποιούνται ως δεδομένα εκπαίδευσης. Στη συνέχεια, η διαδικασία επαναλαμβάνεται k φορές, με καθένα από τα δείγματα k να χρησιμοποιείται ακριβώς μία φορά ως δεδομένα επικύρωσης. Τα αποτελέσματα k μπορούν στη συνέχεια να υπολογιστούν κατά μέσο όρο για να παραχθεί μια ενιαία εκτίμηση. Το πλεονέκτημα αυτής της μεθόδου είναι ότι όλα τα στοιχεία χρησιμοποιούνται τόσο για την εκπαίδευση όσο και για την επικύρωση και κάθε δείγμα χρησιμοποιείται για την επαλήθευση ακριβώς μία φορά. Για την εργασία επιλέχτηκε η τιμή k να ισούται με 5.

Τα πειραματικά αποτελέσματα κάθε υλοποίησης του RSP3 που θα παρουσιαστούν στην συνέχεια περιλαμβάνουν τα ποσοστά ακρίβειας και μείωσης που επιτεύχθηκαν, το χρόνο που χρειάστηκε μόνο για την εκτέλεση του RSP3 και το πλήθος των αποστάσεων που υπολογίστηκαν. Όλες οι τιμές προκύπτουν από το μέσο όρο των αποτελεσμάτων των 5 δειγμάτων που παράχθηκαν με την τεχνική επικύρωσης. Ο χρόνος ορίζεται ως χρόνος επεξεργασίας (CPU Time) και δεν είναι πραγματικός. Επιπλέον, δεν συμπεριλαμβάνεται σε αυτόν ο χρόνος ταξινόμησης.

5.3 Η πειραματική διαδικασία

Όλα τα βήματα της διαδικασίας του πειράματος μπορούν να αναπαρισταθούν απλοϊκά με ένα διάγραμμα ροής (σχ. 5.2). Η διαδικασία αυτή επαναλήφθηκε για κάθε ένα από τα σύνολα δεδομένων του πειράματος προς εξαγωγή των αποτελεσμάτων που θα παρουσιαστούν παρακάτω.



Σχήμα 5.2: Διάγραμμα ροής του πειράματος

5.4 Τα αποτελέσματα του αλγορίθμου GRID

Στον πίνακα 5.2 παρουσιάζονται τα αποτελέσματα για κάθε ένα από τα σύνολα δεδομένων του πειράματος, με την εκτέλεση του RSP3 σε συνδυασμό με το συμβατικό αλγόριθμο GRID. Κάθε πίνακας αποτελεσμάτων που θα ακολουθήσει στην συνέχεια θα έχει παρόμοια μορφή. Πρόκειται ουσιαστικά για την δοκιμή της αρχικής έκδοσης του RSP3 όπως προτάθηκε από τον Sanchez. Ο κυρίαρχος στόχος των νέων παραλλαγών είναι η επίτευξη σημαντικά μικρότερου χρόνου επεξεργασίας από αυτόν του GRID χωρίς να υπάρξει αλλοίωση του ποσοστού ακρίβειας ή μείωσης. Επομένως τα πειραματικά αποτελέσματα του GRID είναι απαραίτητα για να γίνει σύγκριση μεταξύ των νέων μεθοδολογιών και της κλασσικής υλοποίησής του RSP3.

Εφόσον ο συμβατικός αλγόριθμος βρίσκει την πραγματική μέγιστη απόσταση για κάθε υποσύνολο, τα ποσοστά ακρίβειας και μείωσης είναι πολύ κοντά στα βέλτιστα. Επομένως, όσο μικρότερη η απόκλιση των αποτελεσμάτων των νέων παραλλαγών

Σύνολο	Ακρίβεια(%)	Μείωση(%)	Χρόνος(s)	Αποστάσεις
BN	84.42	75.13	5.76	18,762,868
EEG	47.31	53.69	465.63	498,735,010
KDD	99.60	98.54	41021.00	20,895,641,705
LIR	95.45	61.81	389.20	326,319,162
LS	90.10	73.20	119.30	33,856,789
MGT	77.82	58.44	315.63	382,553,638
PD	99.10	89.22	139.17	86,727,126
PH	87.10	69.09	12.46	20,640,854
RNG	81.84	57.15	94.29	46,908,442
SG	95.97	81.95	9.77	5,012,489
SH	99.39	99.38	3125.62	1,755,937,353
TN	93.00	84.58	75.03	37,775,520
TXR	98.54	82.60	99.60	25,929,238
WF	77.62	57.02	16.74	16,939,820
WQW	61.68	35.34	39.82	33,014,949
YST	50.00	28.35	2.10	1,960,959

Πίνακας 5.2: Τα αποτελέσματα του RSP3 με GRID

από αυτά, τόσο πιο εύστοχα μπορούν να τον αντικαταστήσουν. Ιδιαίτερα εμφανή είναι τα τεράστιας τάξης νούμερα των αποστάσεων που υπολογίζονται για κάθε σύνολο ιδιαίτερα σε αυτά με μεγάλο αριθμό στοιχείων (KDD, SH). Η μεγάλη χρονική καθυστέρηση του αλγορίθμου οφείλεται σε αυτό, καθώς υπάρχει αναλογία μεταξύ του χρόνου επεξεργασίας και των αποστάσεων που υπολογίστηκαν.

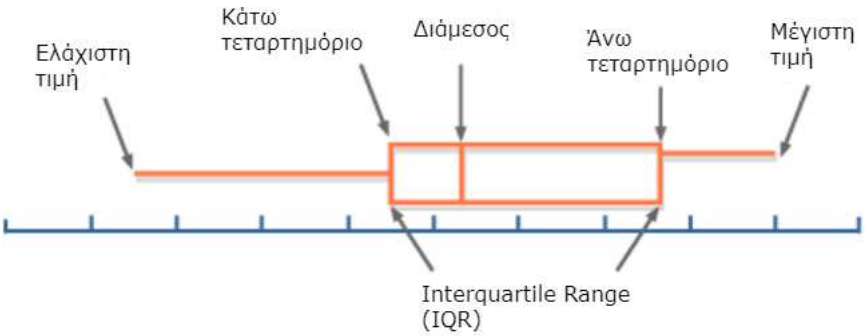
5.5 RSPQ: Μείωση με διαμερισμό του χώρου χρησιμοποιώντας τον Quick Hull

Η πρώτη παραλλαγή του RSP3 χρησιμοποιεί μια υλοποίηση του Quick Hull των τριών διαστάσεων. Στα σύνολα δεδομένων που αποτελούνται το πολύ από τρία χαρακτηριστικά η παραλλαγή μπορεί να αντικαταστήσει ολοκληρωτικά την κλασσική υλοποίηση. Σε σύνολα δεδομένων όμως με περισσότερες διαστάσεις είναι αναγκαία η χρήση μιας τεχνικής επιλογής τριών χαρακτηριστικών για την διεξαγωγή του πειράματος. Επιλέχτηκαν δύο τεχνικές με σκοπό την επίλυση του προβλήματος των διαστάσεων. Η πρώτη επιχειρεί να διαχωρίσει τα χαρακτηριστικά βάση της πυκνότητας των τιμών τους (Interquartile Range), ενώ η δεύτερη βάση της σημασίας τους στην διαμόρφωση της κλάσης (Gain Ratio).

5.5.1 Interquartile Range (IQR)

Το IQR είναι ένα μέτρο μεταβλητότητας, που βασίζεται στη διαίρεση ενός συνόλου δεδομένων σε τεταρτημόρια. Τα τεταρτημόρια χωρίζουν ένα ταξινομημένο σύνολο δεδομένων σε τέσσερα ίσα μέρη. Οι τιμές που χωρίζουν τα μέρη του συνόλου ονομάζονται πρώτο, δεύτερο και τρίτο τεταρτημόριο και συμβολίζονται με τα $Q1$, $Q2$ και $Q3$, αντίστοιχα. Υπολογιστικά, το IQR είναι το πρώτο τεταρτημόριο που αφαιρείται από το τρίτο τεταρτημόριο ($IQR = Q_3 - Q_1$).

Ο υπολογισμός του IQR για κάθε χαρακτηριστικό και στην συνέχεια η επιλογή των τριών χαρακτηριστικών με τις υψηλότερες τιμές του αποτελεί την πρώτη τεχνική επιλογής διαστάσεων της παραλλαγής RSP3. Η ιδέα είναι πως όσο μεγαλύτερο είναι το εύρος του όγκου των τιμών για ένα χαρακτηριστικό τόσο πιο μεγάλη είναι η διασπορά της διάστασης αυτής. Επομένως, θεωρητικά θα συμβάλει περισσότερο στον



Σχήμα 5.3: Αναπαράσταση του Interquartile Range

Σύνολο	Ακρίβεια(%)	Μείωση(%)	Χρόνος(s)	Αποστάσεις
BN	84.42	75.18	0.56	20,063
EEG	47.26	53.70	8.39	57,499
KDD	99.50	98.19	12.06	17,686
LIR	95.37	61.57	13.93	185,418
LS	90.35	73.47	1.58	79,779
MGT	77.55	58.85	13.56	364,086
PD	99.06	88.77	1.19	74,436
PH	86.89	69.40	1.01	88,380
RNG	82.22	56.80	3.11	116,817
SG	96.23	82.45	0.32	39,473
SH	99.48	99.29	2.22	18,490
TN	92.86	83.87	0.98	79,221
TXR	98.67	82.72	1.06	67,111
WF	78.00	56.61	1.32	79,680
WQW	61.86	35.17	2.96	83,874
YST	50.14	27.72	0.45	27,692

Πίνακας 5.3: Τα αποτελέσματα του RSPQ με IQR

διαμερισμό του χώρου των δεδομένων σε σύγκριση με τις υπόλοιπες. Στον πίνακα 5.3 παρουσιάζονται τα αποτελέσματα του RSP3 με χρήση IQR για την επιλογή των τριών χαρακτηριστικών. Οι αποστάσεις που υπολογίζονται είναι πολύ λιγότερες από αυτές του συμβατικού αλγορίθμου με συνέπεια τον επιθυμητό δραματικά μικρότερο χρόνο επεξεργασίας. Επιπλέον, ιδιαίτερα θετικό στοιχείο αποτελεί το ότι οι διακυμάνσεις των ποσοστών ακρίβειας και μείωσης είναι αμελητέες. Να σημειωθεί πως σε κάθε υπολογισμό απόστασης λαμβάνονται υπ΄ όψιν όλα τα χαρακτηριστικά των σημείων και όχι μόνο τα τρία χαρακτηριστικά που επιλέχθηκαν με κριτήριο το IQR.

5.5.2 Κατάταξη χαρακτηριστικών με Gain Ratio

Ένα συχνά εμφανιζόμενο πρόβλημα στην προσπάθεια ταξινόμησης δεδομένων εκπαίδευσης είναι η μεγάλη υπολογιστική πολυπλοκότητα που απαιτείται, ιδιαίτερα σε σύνολα δεδομένων πολλών διαστάσεων. Για την αντιμετώπιση του προβλήματος δημιουργήθηκαν διάφορες τεχνικές κατάταξης που αποσκοπούν στην αξιολόγηση του κάθε χαρακτηριστικού και στην εύρεση της σημασίας τους στην διαμόρφωση της κλάσης με στόχο τη μείωση του φόρτου ταξινόμησης [19; 47; 63].

Το περιβάλλον Weka [30] προτείνει διάφορες τεχνικές κατάταξης για την επίλυση του προβλήματος. Στα πλαίσια της εργασίας και εφόσον υπάρχει η ανάγκη επιλογής τριών χαρακτηριστικών για την εκτέλεση του Quick Hull χρησιμοποιήθηκε η τεχνική Gain Ratio με σκοπό την επιλογή των τριών χαρακτηριστικών με το υψηλότερο αντίκτυπο στην διαμόρφωση της κλάσης του συνόλου.

Ο Gain Ratio αποτελεί έναν αλγόριθμο δέντρου αποφάσεων του οποίου οι διαδικασίες μπορούν να συνοψιστούν ως εξής: Αρχικά επιλέγει το χαρακτηριστικό που διαφοροποιεί καλύτερα τις τιμές των χαρακτηριστικών εξόδου. Ένας ξεχωριστός κόμβος δημιουργείται για κάθε τιμή του επιλεγμένου χαρακτηριστικού. Στην συνέχεια, όλες οι περιπτώσεις διαχωρίζονται σε υποομάδες ώστε να ανταναικλούν τις τιμές χαρακτηριστικών του επιλεγμένου κόμβου. Για κάθε υποομάδα η διαδικασία

Σύνολο	Ακρίβεια(%)	Μείωση(%)	Χρόνος(s)	Αποστάσεις
BN	84.42	75.18	0.56	20,063
EEG	46.46	53.73	8.98	56,689
KDD	99.30	98.63	9.01	12,382
LIR	95.34	62.17	13.38	147,890
LS	90.07	74.10	1.51	78,098
MGT	77.59	58.77	13.62	347,700
PD	98.94	88.66	1.07	52,416
PH	87.30	69.35	1.00	78,874
RNG	81.85	56.17	3.18	117,140
SG	96.40	82.44	0.28	25,874
SH	99.56	99.32	2.37	21,479
TN	92.79	83.75	0.99	80,214
TXR	98.80	82.70	1.06	67,111
WF	78.00	56.61	1.04	79,696
WQW	61.88	34.69	3.00	73,182
YST	49.86	27.94	0.31	4,280

Πίνακας 5.4: Τα αποτελέσματα του RSPQ με Gain Ratio

επιλογής τερματίζεται εάν α) τα μέλη μιας υποομάδας έχουν την ίδια σημασία για το χαρακτηριστικό εξόδου, περατώνεται η διαδικασία επιλογής για την τρέχουσα διαδρομή και ως ετικέτα του κόμβου ορίζεται η τιμή της σημασίας ή β) η υποομάδα περιέχει έναν μοναδικό κόμβο ή δεν μπορούν να προσδιοριστούν άλλα διακριτικά χαρακτηριστικά. Όπως και στο α), στην ετικέτα του κλάδου ορίζεται η τιμή εξόδου που παρατηρείται από την πλειοψηφία των υπολοίπων περιπτώσεων. Στον πίνακα 5.4 παρουσιάζονται τα αποτελέσματα του RSPQ με χρήση του Gain Ratio. Υπάρχει ομοιότητα με τα αποτελέσματα του IQR και αποδεικνύεται πως και αυτή η μέθοδος μπορεί να αντικαταστήσει επιτυχώς την κλασσική υλοποίηση του RSP3.

5.6 RSPA: Μείωση με διαμερισμό του χώρου χρησιμοποιώντας τον Margin Hull

Σε αντίθεση με την RSPQ η παραλλαγή RSPA δεν χρειάζεται καμία τεχνική επιλογής διαστάσεων καθώς ο αλγόριθμος Margin Hull είναι γενικεύσιμος για οποιοδήποτε αριθμό χαρακτηριστικών. Στον πίνακα 5.5 παραθέτονται τα αποτελέσματα του πειράματος. Παρατηρείται πως και αυτή η παραλλαγή εκτελείται σε συντριπτικά λιγότερο χρόνο σε σχέση με την κλασσική υλοποίηση και διαφέρει ελάχιστα σε ποσοστά ακρίβειας και μείωσης. Αποτελεί ακόμα μια εναλλακτική μέθοδο αντικατάστασης της κλασσικής υλοποίησης.

Σύνολο	Ακρίβεια(%)	Μείωση(%)	Χρόνος(s)	Αποστάσεις
BN	84.53	75.32	0.42	4,778
EEG	47.56	53.79	8.70	259,890
KDD	99.58	98.56	16.71	150,270
LIR	95.38	61.56	14.62	276,038
LS	90.21	73.11	2.24	207,111
MGT	77.58	58.88	12.73	217,924
PD	99.04	89.10	1.06	99,658
PH	87.47	69.30	0.70	22,824
RNG	81.30	55.75	3.44	206,321
SG	96.49	82.24	0.23	20,944
SH	99.69	99.37	2.16	21,486
TN	92.52	84.07	1.01	140,266
TXR	98.58	82.97	1.37	125,003
WF	78.02	57.14	1.37	150,659
WQW	62.42	35.65	2.85	64,071
YST	50.54	27.45	0.38	12,963

Πίνακας 5.5: Τα αποτελέσματα του RSPA

Κεφάλαιο 6

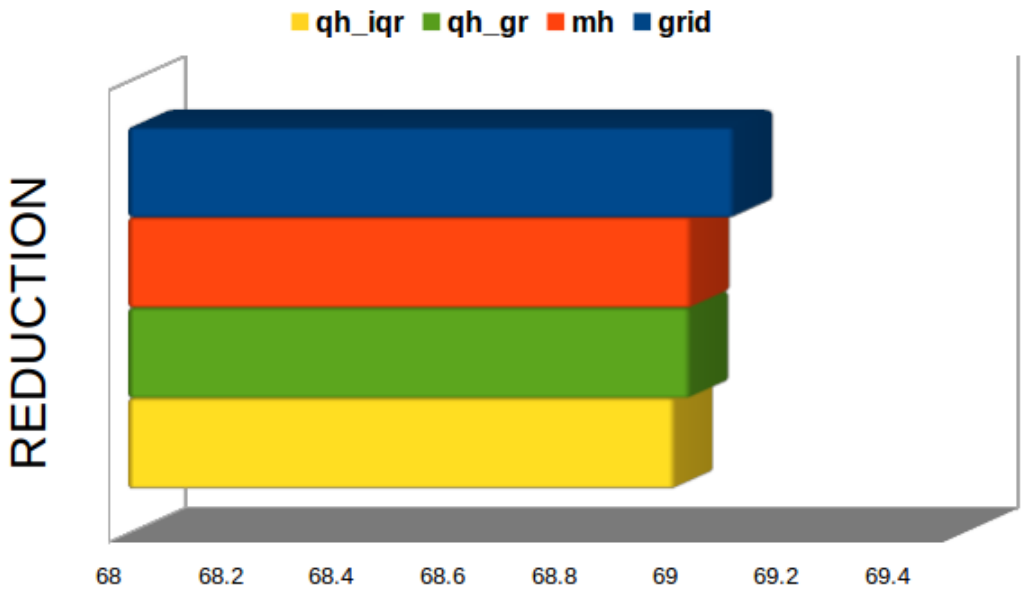
Συμπεράσματα

Στο τελευταίο αυτό κεφάλαιο θα παρουσιαστούν σε μορφή γραφημάτων τα προηγούμενα αποτελέσματα και θα ερμηνευτεί η σημασία τους. Στην συνέχεια, θα γίνει σύγκριση μεταξύ των νέων παραλλαγών και θα ερμηνευτούν τα αποτελέσματα των μεθόδων για κάθε κριτήριο. Στον επίλογο, θα παρατεθούν ορισμένες συστάσεις και προβληματισμοί πάνω στο αντικείμενο της έρευνας με σκοπό την παραγωγή τροφής για μελλοντική έρευνα.

6.1 Σύγκριση μέσω γραφημάτων

Προκειμένου να είναι πιο κατανοητή και ευπαρουσίαστη η σύγκριση όλων των μεθόδων μεταξύ τους, δημιουργήθηκαν κατάλληλα διαμορφωμένα γραφήματα για κάθε ένα από τα τέσσερα κριτήρια αξιολόγησης του πειράματος (μείωση, ακρίβεια, χρόνος εκτέλεσης, πλήθος αποστάσεων). Για κάθε κριτήριο υπολογίστηκε ο μέσος όρος των τιμών του σε όλα τα σύνολα δεδομένων, για κάθε έναν από τους τέσσερις αλγορίθμους (Grid, Margin Hull, Quick Hull με IQR και Gain Ratio). Για τα κριτήρια του χρόνου εκτέλεσης και τον αποστάσεων, λόγω της υπερμεγέθης διαφοράς των αποτελεσμάτων μεταξύ του συμβατικού Grid και των υπολοίπων, τα εύρη των γραφημάτων τροποποιήθηκαν καταλλήλως. Παρακάτω ακολουθούν αναλυτικά τα συμπεράσματα της σύγκρισης των μεθόδων για τα κριτήρια του πειράματος.

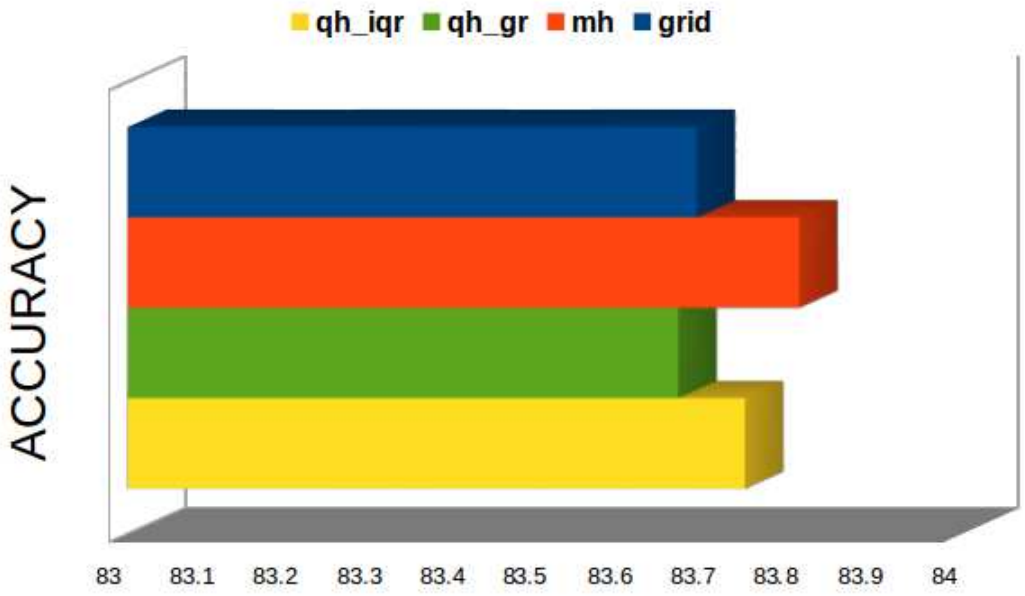
α) Ποσοστό μείωσης



Σχήμα 6.1: Γράφημα σύγκρισης του ποσοστού μείωσης

Από το γράφημα προκύπτει πως οι διαφορές στο ποσοστό μείωσης είναι αμελητέες της τάξης των δεκαδικών ψηφίων. Αυτό είναι ένα ιδιαίτερα θετικό στοιχείο καθώς αποτελούσε έναν από τους πρωταρχικούς στόχους της έρευνας. Ο συμβατικός αλγόριθμος Grid επιτυγχάνει ένα ελάχιστα υψηλότερο ποσοστό συγκριτικά με τους υπόλοιπους αλγορίθμους. Ο μέσος όρος των ποσοστών του Margin Hull είναι πανομοιότυπος με αυτόν του Quick Hull με χρήση της τεχνικής Gain Ratio ενώ μέσω της τεχνικής Interquartile Range επιφέρει έναν ελάχιστα χαμηλότερο μέσο όρο ποσοστού μείωσης σε σχέση με τους υπόλοιπους αλγορίθμους. Μπορούμε από το γράφημα να συμπεράνουμε πως η προσέγγιση της μέγιστης απόστασης ή η εύρεση αυτής μέσω της ταχείας τεχνικής Quick Hull των τριών διαστάσεων έχει σχεδόν μηδαμινό αντίκτυπο στο κριτήριο αυτό και αποτελούν και οι δύο παραλλαγές ιδανικές λύσεις.

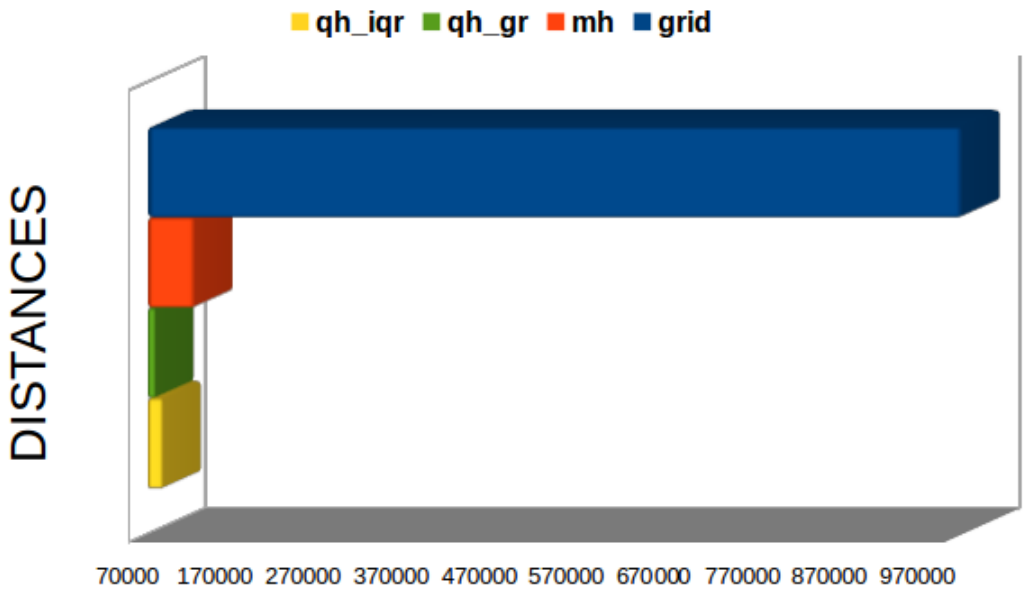
β) **Ποσοστό ακρίβειας**



Σχήμα 6.2: Γράφημα σύγκρισης του ποσοστού ακρίβειας

Παρομοίως με το ποσοστό μείωσης οι διακυμάνσεις του ποσοστού ακρίβειας είναι μικρές της τάξης δεκαδικών ψηφίων μεταξύ των αλγορίθμων του πειράματος, εκπληρώνοντας άλλον έναν σημαντικό στόχο της έρευνας. Ενδιαφέρον παρουσιάζει το γεγονός πως η προσέγγιση της μέγιστης απόστασης με Margin Hull ξεπερνάει οριακά τους μέσους όρους ακριβείας των υπόλοιπων αλγορίθμων, οι οποίοι χρησιμοποιούν πραγματικές μέγιστες αποστάσεις. Το εύρημα αυτό επαληθεύει τη θεωρία πως η χρήση της μέγιστης απόστασης ενός υποσυνόλου δεν παράγει αναγκαστικά το βέλτιστο αποτέλεσμα για τον RSP3. Σε γενικές γραμμές όσον αφορά τα ποσοστά μείωσης και ακρίβειας, δεν υπάρχει ουσιαστική διαφορά μεταξύ των αποτελεσμάτων με συνέπεια να θεωρούνται όλοι οι αλγόριθμοι επάξιοι στις επιδόσεις τους στα κριτήρια αυτά.

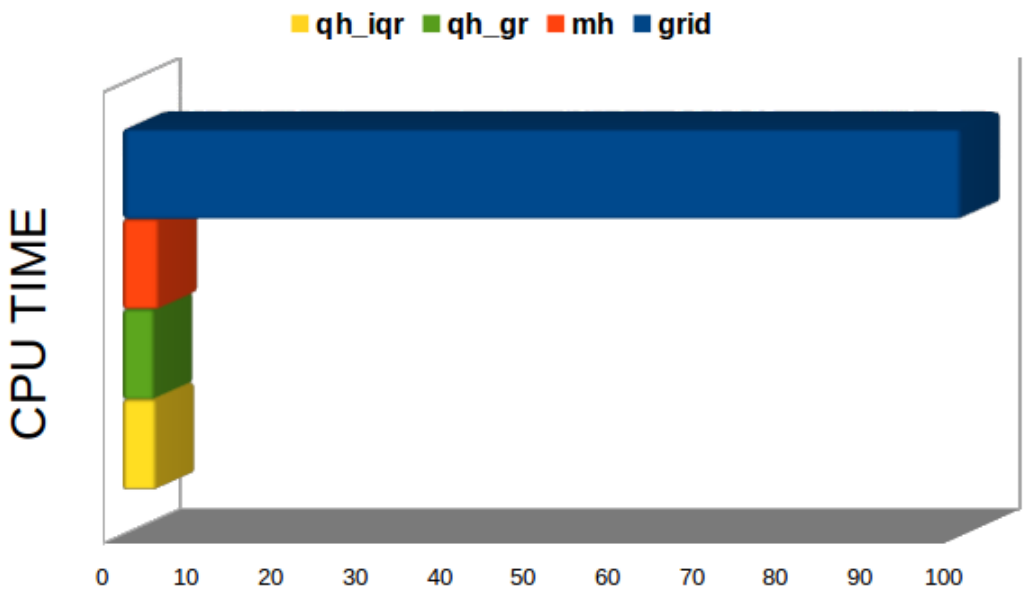
γ) Πλήθος αποστάσεων



Σχήμα 6.3: Γράφημα σύγκρισης του πλήθους αποστάσεων

Η τεράστια διαφορά στο πλήθος των αποστάσεων που υπολογίζονται μεταξύ του συμβατικού Grid και των υπολοίπων αλγορίθμων είναι ιδιαίτερα εμφανής στο γράφημα. Στην πραγματικότητα, ο πραγματικός μέσος όρος των αποστάσεων του συμβατικού αλγορίθμου είναι 100 φορές μεγαλύτερος από την απεικονιζόμενη τιμή. Παρατηρείται πως ο Margin Hull υπολογίζει περισσότερες αποστάσεις από τον Quick Hull, εφόσον χρησιμοποιεί όλες τις διαστάσεις του συνόλου δεδομένων σε αντίθεση με το δεύτερο, που χρησιμοποιεί τρεις. Ενδιαφέρον προκαλεί το γεγονός ότι σε σύνολα δεδομένων χαμηλών διαστάσεων (π.χ. BN - 2 διαστάσεις) ο Margin Hull υπολογίζει λιγότερες αποστάσεις από τον Quick Hull. Μπορούμε λοιπόν να συμπεράνουμε ότι σε ένα υποθετικό σενάριο όπου ο Quick Hull θα χρησιμοποιούσε όλες τις διαστάσεις του συνόλου, ο μέσος όρος του πλήθους των αποστάσεων που θα υπολόγιζε θα ήταν πολύ μεγαλύτερος από αυτόν του Margin Hull. Αυτό συμβαίνει καθώς το πλήθος των σημείων του Convex Hull ενός υποσυνόλου είναι κατά κανόνα μεγαλύτεο απο το πλήθος των σημείων Margin Hull του.

δ) Χρόνος επεξεργασίας



Σχήμα 6.4: Γράφημα σύγκρισης του χρόνου επεξεργασίας

Καθώς το πλήθος τον αποστάσεων καθορίζει σχεδόν εξ' ολοκλήρου το χρόνο επεξεργασίας που θα χρειαστεί ο κάθε αλγόριθμος, είναι φυσικό το γράφημα του χρόνου επεξεργασίας να είναι όμοιο με αυτό των αποστάσεων. Η μείωση του εύρους της μέγιστης τιμής ήταν αναγκαία και σε αυτό το γράφημα, καθώς ο μέσος όρος χρόνου επεξεργασίας του Grid είναι πολύ μεγαλύτερος των υπολοίπων. Παρατηρείται πως αν και κατά μέσο όρο ο Margin Hull υπολογίζει περισσότερες αποστάσεις από τον Quick Hull, αυτό δεν φαίνεται να επηρεάζει σημαντικά τη διαφορά στον χρόνο επεξεργασίας του. Το γεγονός δικαιολογείται με την απλότητα των εσωτερικών διεργασιών του αλγορίθμου και τη σταθερότητα του συνόλου που επιστρέφει σε αντίθεση με τον Quick Hull όπου η πολυπλοκότητα του αλγορίθμου εξαρτάται από τα δεδομένα που εισάγονται και όχι απλά από το πλήθος των χαρακτηριστικών.

6.2 Επίλογος / Κατευθύνσεις για μελλοντική έρευνα

Από τα πειραματικά αποτελέσματα προκύπτει πως υπάρχουν πολλά περιθώρια βελτίωσης του αλγορίθμου RSP3. Οι δύο νέες παραλλαγές που προτάθηκαν κατάφεραν να μειώσουν δραματικά το χρόνο που απαιτείται για την παραγωγή του σετ συμπύκνωσης χωρίς να αλλοιωθεί η ποιότητα των ποσοστών μείωσης και ακρίβειας και είναι ασφαλές να εννοηθεί πως μπορούν να αντικαταστήσουν ολικά την κλασσική υλοποίηση.

Εφόσον ο χρόνος που αποτελούσε το μεγαλύτερο εμπόδιο για την ευρεία χρήση του RSP3 έχει αντιμετωπιστεί, το επόμενο βήμα για τη μελέτη και βελτίωση του αλγορίθμου είναι η αύξηση των ποσοστών μείωσης και ακρίβειας. Η περαιτέρω μελέτη του κριτηρίου επιλογής της σειράς των υποσυνόλων προς διαίρεση αποτελεί μια καλή αρχή. Όπως και αποδείχτηκε απο τα πειράματα της εργασίας, η μέγιστη απόσταση δεν αποτελεί αναγκαστικά τη βέλτιστη επιλογή και ίσως μια διαφορετική μεθοδολογία επιλογής να βελτίωνε σημαντικά τα ποσοστά απόδοσης του RSP3.

Τέλος, με τις νέες παραλλαγές του RSP3 ίσως είναι εφικτό να δημιουργεί μια τεχνική για μείωση δεδομένων σε ζωντανή συνεχόμενη ροή (data streaming) όπου τα δεδομένα ενός συνόλου μεταβάλλονται συνεχόμενα σε πραγματικό χρόνο.

Αναφορές

[1] Adam Jóźwik. A method for solving the n-dimensional convex hull problem. *Pattern Recognition Letters*, 2(1):23 – 25, 1983. 45

[2] Aguilar, Jesús S. and Riquelme, José C. and Toro, Miguel. Data Set Editing by Ordered Projection. *Intell. Data Anal.*, 5(5), 10. 2001. 21

[3] Alcalá-Fdez, Jesús and Fernández, Alberto and Luengo, Julián and Derrac, Joaquín and García, Salvador. KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework. *Multiple-Valued Logic and Soft Computing*, 17(2-3):255–287, 2011. 51

[4] Angiulli, Fabrizio. Fast condensed nearest neighbor rule. Iν *Proceedings of the 22nd international conference on Machine learning*, ICML '05, New York, NY, USA, 2005. ACM. 21

[5] Angiulli, Fabrizio. Fast Nearest Neighbor Condensation for Large Data Sets Classification. *IEEE Trans. on Knowl. and Data Eng.*, 19(11), 11. 2007. 21

[6] Barber, C. Bradford and Dobkin, David P. and Dobkin, David P. and Huhdanpaa, Hannu. The Quickhull Algorithm for Convex Hulls. *ACM Trans. Math. Softw.*, 22(4):469–483, 12. 1996. 43

[7] Brighton, Henry and Mellish, Chris. Advances in Instance Selection for Instance-Based Learning Algorithms. *Data Min. Knowl. Discov.*, 6(2), 4. 2002. 10

[8] Chen, C. H. and Jóźwik, Adam. A sample set condensation algorithm for the class sensitive artificial neural network. *Pattern Recogn. Lett.*, 17(8):819–823, 8. 1996. 22

[9] Chin-Liang Chang. Finding Prototypes For Nearest Neighbor Classifiers. *IEEE Trans. Comput.*, 23(11), 11. 1974. 26

[10] Chou, Chien-Hsing and Kuo, Bo-Han and Chang, Fu. The Generalized Condensed Nearest Neighbor Rule as A Data Reduction Method. Iν *Proceedings of the 18th International Conference on Pattern Recognition - Volume 02*, ICPR '06, Washington, DC, USA, 2006. IEEE Computer Society. 21

[11] Cover, T. and Hart, P. Nearest Neighbor Pattern Classification. *IEEE Trans. Inf. Theor.*, 13(1), 9. 1

[12] Dasarathy, B. V. *Nearest neighbor (NN) norms : NN pattern classification techniques*. IEEE Computer Society Press, 1991. 1

[13] Datta, Piew and Kibler, Dennis F. Learning Symbolic Prototypes. Iν *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc. 26

[14] De Berg, M. and Cheong, O. and Van Kreveld, M. *Computational geometry: algorithms and applications*. Springer-Verlag New York Inc, 2008. 37

[15] Demšar, Janez. Statistical Comparisons of Classifiers over Multiple Data Sets. *J. Mach. Learn. Res.*, 7, 12. 2006. 21

[16] Dennis L. Wilson. Asymptotic Properties of Nearest Neighbor Rules Using Edited Data. *IEEE trans. on systems, man, and cybernetics*, 2(3):408–421, 7. 1972. 12

[17] Devijver, P. A. and Kittler, J. On the edited nearest neighbor rule. Iν *Proceedings of the Fifth International Conference on Pattern Recognition*. The Institute of Electrical and Electronics Engineers, 1980. 14

[18] Deza, Michel Marie and Deza, Elena. *Encyclopedia of distances*. Springer, 2009. 3

[19] Doraisamy, Shyamala and Golzari, Shahram and Mohd. Norowi, Noris and Sulaiman, md nasir and Udzir, Nur. A Study on Feature Selection and Classification Techniques for Automatic Genre Classification of Traditional Malay Music. παγες 331–336, 1. 2008. 58

[20] Eaton, John W and Bateman, David and Hauberg, Soren. *GNU Octave Manual Version 3*. Network Theory Ltd., 2008. 43

[21] Fayed, Hatem A. and Atiya, Amir F. A Novel Template Reduction Approach for the K-nearest Neighbor Method. *Trans. Neur. Netw.*, 20(5), 5. 2009. 21

[22] Frank Nielsen. *Algorithmes géométriques adaptatifs*. Thèse de doctorat en sciences, Université de Nice-Sophia Antipolis, France, 1996. 41

[23] G. Ritter and H. Woodruff and S. Lowry and T. Isenhour . An algorithm for a selective nearest neighbor decision rule. *IEEE Trans. on Inf. Theory*, 21(6), 1975. 21

[24] G. W. Gates. The reduced nearest neighbor rule. IEEE Transactions on Information Theory. *IEEE Transactions on Information Theory*, 18(3), 1972. 21

[25] García-Borroto, Milton and Villuendas-Rey, Yenny and Carrasco-Ochoa, Jesús Ariel and Martínez-Trinidad, José Fco. Using Maximum Similarity Graphs to Edit Nearest Neighbor Classifiers. Iν *Proceedings of the 14th Iberoamerican Conference on Pattern Recognition: Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, CIARP '09, Berlin, Heidelberg, 2009. Springer-Verlag. 13, 14, 15

[26] Garcia, Salvador and Derrac, Joaquin and Cano, Jose and Herrera, Francisco. Prototype Selection for Nearest Neighbor Classification: Taxonomy and Empirical Study. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(3), 3. 2012. 21

[27] García, Salvador and Derrac, Joaquín and Cano, José Ramón and Herrera, Francisco. Prototype selection for nearest neighbor classification: taxonomy and empirical study. *IEEE transactions on pattern analysis and machine intelligence*, 34(3):417–35, 3. 2012. 7

[28] Graham, Ronald L. An Efficient Algorithm for Determining the Convex Hull of a Finite Planar Set. *Inf. Process. Lett.*, 1(4):132–133, 1972. 39

[29] Grochowski, Marek and Jankowski, Norbert. Comparison of Instance Selection Algorithms II. Results and Comments. Iν *Artificial Intelligence and Soft Computing - ICAISC 2004*. 10

[30] Hall, M. and Frank, E. and Holmes, G. and Pfahringer, B. and Reutemann, P. and Witten, I.H. The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18, 2009. 3, 58

[31] Hamerly, Greg and Elkan, Charles. Alternatives to the K-means Algorithm That Find Better Clusterings. Iν *Proceedings of the Eleventh International*

Conference on Information and Knowledge Management, CIKM '02, παγες 600–607, New York, NY, USA, 2002. A͡M. 25

[32] Hart, P E. The condensed nearest neighbor rule. *IEEE Transactions on Information Theory*, 14(3):515–516, 1968. 17

[33] I. Tomek. An experiment with the edited nearest-neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics*, 6, 1976. 16

[34] I. Tomek. Two Modifications of CNN. *Systems, Man and Cybernetics, IEEE Transactions on*, SMC-6(11), 11. 1976. 21

[35] Isaac Triguero, Joaquín Derrac, Salvador Garcia, and Francisco Herrera. A taxonomy and experimental study on prototype generation for nearest neighbor classification. *Trans. Sys. Man Cyber Part C*, 1. 2012. 7

[36] Jose C. Riquelme and Jesus S. Aguilar-Ruiz and Miguel Toro. Finding representative patterns with ordered projections . *Pattern Recognition* , 36(4), 2003. . 21

[37] José Salvador Sánchez. High training set size reduction by space partitioning and prototype abstraction. *Pattern Recognition*, 37(7):1561–1564, 2004. 27, 29, 34

[38] J.S. Sanchez and F. Pla and F.J. Ferri. On the use of neighbourhood-based non-parametric classifiers . *Pattern Recognition Letters* , 18(11–13), 1997. 16

[39] J.S. Sánchez and F. Pla and F.J. Ferri. Prototype selection for the nearest neighbour rule through proximity graphs. *Pattern Recognition Letters*, 18(6):507 – 513, 1997. 15

[40] Kazuo Hattori and Masahito Takahashi. A new edited k-nearest neighbor rule in the pattern classification problem . *Pattern Recognition* , 33(3), 2000. 16

[41] Knuth, Donald E. *Axioms and hulls, Lecture Notes in Computer Science*. Springer-Verlag, 2008. 38

[42] Kohavi, Ron. A Study of Cross-validation and Bootstrap for Accuracy Estimation and Model Selection. Iν *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'95, παγες 1137–1143, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc. 53

[43] Kohonen, T. The self-organizing map. *Proceedings of the IEEE*, 78(9), 1990. 26

[44] Kohonen, T. and Schroeder, M. R. and Huang, T. S., εδιτορ. *Self-Organizing Maps*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 3rd εδιτιον, 2001. 26

[45] L. Cinque and C. Di Maggio. A BSP realisation of Jarvis's algorithm. Iν *Proceedings 10th International Conference on Image Analysis and Processing*, παγες 247–252, 9. 1999. 41

[46] Li, Jiang and Manry, Michael T. and Yu, Changhua and Wilson, D. Randall. Prototype Classifier Design with Pruning. *International Journal on Artificial Intelligence Tools*, 14(1-2), 2005. 26

[47] Mark A. Hall. Correlation-based Feature Selection for Machine Learning. 58

[48] MATLAB. *version 7.10.0 (R2010a)*. The MathWorks Inc., Natick, Massachusetts, 2010. 43

[49] Olvera-López, J. Arturo and Carrasco-Ochoa, J. Ariel and Martínez-Trinidad, J. Francisco and Kittler, Josef. A review of instance selection methods. *Artif. Intell. Rev.*, 34(2):133–143, 8. 2010. 10

[50] Olvera-López, J. Arturo and Martínez-Trinidad, J. Francisco and Carrasco-Ochoa, J. Ariel. Mixed data object selection based on clustering and border objects. Iν *Proceedings of the Congress on pattern recognition 12th Iberoamerican conference on Progress in pattern recognition, image analysis and applications*, CIARP'07, παγες 674–683, Berlin, Heidelberg, 2007. Springer-Verlag. 13

[51] Ougiaroglou Stefanos, Evangelidis Georgios. Efficient data abstraction using weighted IB2 prototypes. 2014. 24

[52] Ougiaroglou Stefanos, Evangelidis Georgios. RHC:a non-parametric cluster-based data reduction for efficient k-nn classification. 2014. 25

[53] R.A. Mollineda and F.J. Ferri and E. Vidal. Αν εφφιςιεντ προτοτψπε μεργινγ στρατεγψ φορ τηε ςονδενσεδ 1-νν ρυλε τηρουγη ςλασσ-ςονδιτιοναλ ηιεραρςηιςαλ ςλυστερινγ. *Pattern Recognition*, 35(12), 2002. 26

[54] Roberto Odorico. Learning Vector Quantization with Training Count (LVQTC) . *Neural Networks* , 10:1083 – 1088, 1997. . 26

[55] S. Ougiaroglou and G. Evangelidis. Fast and Accurate k-Nearest Neighbor Classification Using Prototype Selection by Clustering. Iν *2012 16th Panhellenic Conference on Informatics*, 10. 2012. 25

[56] Sánchez, J. S. and Barandela, R. and Marqués, A. I. and Alejo, R. and Badenas, J. Analysis of new techniques to obtain quality training sets. *Pattern Recogn. Lett.*, 24(7), 4. 2003. 16

[57] Sang-Woon Kim and B.J. Oommen. Enhancing prototype reduction schemes with LVQ3-type algorithms . *Pattern Recognition* , 36(5), 2003. . 26

[58] Timothy M. Chan. Optimal output-sensitive convex hull algorithms in two and three dimensions. 1996. 41

[59] Vázquez, Fernando and Sánchez, J. Salvador and Pla, Filiberto. A stochastic approach to wilson's editing algorithm. Iν *Proceedings of the Second Iberian conference on Pattern Recognition and Image Analysis - Volume Part II*, IbPRIA'05, παγες 35–42, Berlin, Heidelberg, 2005. Springer-Verlag. 13

[60] Vázquez, Fernando and Sánchez, J. Salvador and Pla, Filiberto. A stochastic approach to wilson's editing algorithm. Iν *Proceedings of the Second Iberian conference on Pattern Recognition and Image Analysis - Volume Part II*, IbPRIA'05, Berlin, Heidelberg, 2005. Springer-Verlag. 16

[61] Wilson, D. Randall and Martinez, Tony R. Reduction Techniques for Instance-Based Learning Algorithms. *Mach. Learn.*, 38(3):257–286, 3. 2000. 13

[62] Xie, Qiaobing and Laszlo, Charles A. and Ward, Rabab K. Vector Quantization Technique for Nonparametric Classifier Design. *IEEE Trans. Pattern Anal. Mach. Intell.*, 15(12), 1993. 26

[63] Y.Saeys, I.Inza, and P. LarrANNaga. A Review of Feature Selection Techniques in Bioinformatics. 2007. 58