ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

ΕΥΦΥΕΙΣ ΤΕΧΝΟΛΟΓΙΕΣ ΔΙΑΔΙΚΤΥΟΥ – WEB INTELLIGENCE

Μηχανική μάθηση βασισμένη σε στιγμιότυπα μέσω τεχνικών μείωσης δεδομένων για μη μετρικούς χώρους

(Instance-based machine learning through data reduction techniques for non-metric spaces)

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Φωτιάδη Γεώργιου
Α.Μ. : 22/2021

Επιβλέπων : Δρ. Στέφανος Ουγιάρογλου
Επίκουρος καθηγητής, ΔΙ.ΠΑ.Ε.

Θεσσαλονίκη, Σεπτέμβριος 2023

ΔΙΕΘΝΕΣ ΠΑΝΕΠΙΣΤΗΜΙΟ ΤΗΣ ΕΛΛΑΔΟΣ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ ΕΥΦΥΕΙΣ ΤΕΧΝΟΛΟΓΙΕΣ ΔΙΑΔΙΚΤΥΟΥ – WEB INTELLIGENCE

Μηχανική μάθηση βασισμένη σε στιγμιότυπα μέσω τεχνικών μείωσης δεδομένων για μη μετρικούς χώρους

(Instance-based machine learning through data reduction techniques for non-metric spaces)

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Φωτιάδη Γεώργιου

Επιβλέπων : Δρ. Στέφανος Ουγιάρογλου
Επίκουρος Καθηγητής, ΔΙ.ΠΑ.Ε.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή στις 30 Σεπτεμβρίου 2023.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Στέφανος Ουγιάρογλου Επίκουρος Καθηγητής ΔΙ.ΠΑ.Ε.	Χρήστος Ηλιούδης Καθηγητής ΔΙ.ΠΑ.Ε.	Περικλής Χατζημίσιος Καθηγητής ΔΙ.ΠΑ.Ε.

Θεσσαλονίκη, Σεπτέμβριος 2023

(Υπογραφή)

.....

Η παρούσα εργασία αποτελεί πνευματική ιδιοκτησία του μεταπτυχιακού φοιτητή Γεώργιου Φωτιάδη που την εκπόνησε. Στο πλαίσιο της πολιτικής ανοικτής πρόσβασης, ο συγγραφέας/δημιουργός εκχωρεί στο Διεθνές Πανεπιστήμιο της Ελλάδος άδεια χρήσης του δικαιώματος αναπαραγωγής, δανεισμού, παρουσίασης στο κοινό και ψηφιακής διάχυσης της εργασίας διεθνώς, σε ηλεκτρονική μορφή και σε οποιοδήποτε μέσο, για διδακτικούς και ερευνητικούς σκοπούς, άνευ ανταλλάγματος. Η ανοικτή πρόσβαση στο πλήρες κείμενο της εργασίας, δεν σημαίνει καθ' οποιονδήποτε τρόπο παραχώρηση δικαιωμάτων διανοητικής ιδιοκτησίας του συγγραφέα/δημιουργού, ούτε επιτρέπει την αναπαραγωγή, αναδημοσίευση, αντιγραφή, πώληση, εμπορική χρήση, διανομή, έκδοση, μεταφόρτωση (downloading), ανάρτηση (uploading), μετάφραση, τροποποίηση με οποιονδήποτε τρόπο, τμηματικά ή περιληπτικά της εργασίας, χωρίς τη ρητή προηγούμενη έγγραφη συναίνεση του συγγραφέα/δημιουργού. Η έγκριση της μεταπτυχιακής εργασίας από το Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου της Ελλάδος, δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του συγγραφέα, εκ μέρους του Τμήματος.

Γεώργιος Φωτιάδης

Απόφοιτος Πληροφορικής ΤΕΙ

Ευχαριστίες

Κατά τη διάρκεια των μεταπτυχιακών μου σπουδών και ιδιαίτερα κατά την υλοποίηση αυτής της μεταπτυχιακής εργασίας μου δόθηκε η ευκαιρία να αποκτήσω νέες γνώσεις και ιδέες σε μια περιοχή αρκετά εξελισσόμενη, αυτή της Εξόρυξης Δεδομένων. Για αυτόν τον λόγο θα ήθελα να ευχαριστήσω τους ανθρώπους που με βοήθησαν σε αυτή την προσπάθεια μου.

Αρχικά, θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου κύριο Στέφανο Ουγιάρογλου για την εμπιστοσύνη που μου έδειξε αλλά και για την ουσιαστική βοήθεια του με τις πολύτιμες συμβουλές και κατευθύνσεις που μου έδινε καθ' όλη τη διάρκεια της μεταπτυχιακής μου εργασίας.

Τέλος, οφείλω να ευχαριστήσω όλους εκείνους τους ανθρώπους, οι οποίοι, αν και δεν χρειάστηκε να κάνουν κάτι για αυτή την εργασία, παρ' όλα αυτά στάθηκαν δίπλα μου και με στήριξαν με περίσσεια υπομονή και αγάπη σε όλη μου την πορεία μέχρι σήμερα. Επίσης, ένα μεγάλο ευχαριστώ οφείλω στους γονείς μου Φώτη και Σουλτάνα που με στηρίζουν όλα αυτά τα χρόνια μέχρι σήμερα.

Γεώργιος Φωτιάδης

Θεσσαλονίκη, Σεπτέμβριος 2023

Περίληψη

Στην εποχή της πληροφορίας και της ταχύτατης ανάπτυξης του Διαδικτύου, η διαχείριση και επεξεργασία μεγάλου όγκου δεδομένων εκπαίδευσης αποκτά όλο και μεγαλύτερη σημασία. Ο μεγάλος όγκος δεδομένων, συνήθως δεν είναι εφικτό να χρησιμοποιηθεί από τους αλγόριθμους κατηγοριοποίησης εξαιτίας του υψηλού υπολογιστικού κόστους καθώς και των υψηλών απαιτήσεων αποθήκευσης στη μνήμη. Συνεπώς, τα δεδομένα αυτά προ-επεξεργάζονται από τεχνικές μείωσης του πληθυσμού των δεδομένων εκπαίδευσης (Data Reduction Techniques) με στόχο τη μείωση του υπολογιστικού κόστους αλλά και των απαιτήσεων σε μνήμη. Οι περισσότερες τεχνικές μείωσης του πληθυσμού των δεδομένων που έχουν προταθεί και είναι διαθέσιμες στη βιβλιογραφία αφορούν κυρίως τον κατηγοριοποιητή των κ εγγύτερων γειτόνων (k-Nearest Neighbor classifier). Αυτός ο κατηγοριοποιητής αποτελεί την πιο απλή μέθοδο μηχανικής μάθησης και λειτουργεί μέσω μάθησης βασισμένης σε στιγμιότυπα (instance-based learning). Στις περισσότερες πρακτικές εφαρμογές της επιστήμης των δεδομένων, τα σύνολα δεδομένων θα περιέχουν κατηγορικές μεταβλητές. Ωστόσο, ο κατηγοριοποιητής k-NN δεν μπορεί να διαχειριστεί τα κατηγορικά δεδομένα. Επομένως, πριν από την επεξεργασία από μια τεχνική μείωσης του πληθυσμού των δεδομένων, είναι απαραίτητη η εφαρμογή ενός ακόμη βήματος προ-επεξεργασίας για την μετατροπή των κατηγορικών δεδομένων σε αριθμητικά δεδομένα. Στη βιβλιογραφία, συναντάμε διάφορες τέτοιες μεθόδους και η παρούσα εργασία παρουσιάζει τις σημαντικότερες. Εντούτοις, η εφαρμογή ενός ακόμη βήματος προ-επεξεργασίας είναι ένα αρνητικό σημείο, επειδή προσθέτει υπολογιστικό κόστος. Αυτό το σημείο αποτελεί το κίνητρο εκπόνησης της παρούσας διπλωματικής εργασίας. Ο σκοπός της παρούσας εργασίας είναι να αντιμετωπίσει την πρόκληση της αποτελεσματικής κατηγοριοποίησης δεδομένων που περιέχουν κατηγορικά χαρακτηριστικά, χωρίς να απαιτείται το επιπρόσθετο βήμα προ-επεξεργασίας για την μετατροπή τους. Η μεθοδολογία που χρησιμοποιήθηκε περιλαμβάνει την ανάπτυξη νέων παραλλαγών του αλγορίθμου CNN-rule (Condensed Nearest Neighbour rule), οι οποίες χρησιμοποιούν μετρικές απόστασης για μη μετρικούς χώρους. Εκτελώντας πειράματα σε 8 σύνολα δεδομένων, συγκρίθηκαν οι τρείς παραλλαγές του αλγορίθμου CNN-rule με τον αλγόριθμο των κ εγγύτερων γειτόνων χωρίς μείωση του πληθυσμού των δεδομένων, αξιολογώντας την ακρίβεια (accuracy) και το ποσοστό μείωσης (reduction rate). Τα πειραματικά αποτελέσματα δείχνουν αξιοσημείωτη απόδοση και για τις τρεις παραλλαγές του αλγορίθμου CNN-rule.

Λέξεις Κλειδιά: τεχνικές μείωσης του πληθυσμού των δεδομένων εκπαίδευσης, Data Reduction Techniques, κατηγοριοποιητής κ εγγύτερων γειτόνων, Instance-Based Classification, Instance-Based Learning, μετρικοί χώροι, μη μετρικοί χώροι, μετρικές απόστασης, μη μετρικές απόστασης, συναρτήσεις απόστασης, Hamming απόσταση, Edit απόσταση, Jaccard απόσταση, CNN-rule.

Abstract

In the era of information and rapid internet growth, the management and processing of large volumes of training data becomes increasingly important. Handling such large datasets is not feasible by classification algorithms due to high computational costs and memory storage requirements. Therefore, this data is pre-processed using Data Reduction Techniques to reduce computational costs and the memory storage. Most data reduction techniques which have been proposed and are available in the literature primarily focus on the k-Nearest Neighbor (k-NN) classifier. The k-NN classifier is the simplest instance-based learning method in machine learning. In most practical data science applications, datasets contain categorical variables. However, the k-NN classifier cannot handle categorical data, thus a preprocessing step is necessary to convert categorical data into numerical data. Various methods for this purpose can be found in the literature, and this work presents the most important ones. However, applying an additional preprocessing step is a drawback because it adds computational cost. This issue is the motivation behind this thesis. The purpose of this thesis is to address the challenge of effective classification of data containing categorical features without requiring additional preprocessing steps for their conversion. The methodology used includes the development of new variations of the CNN-rule algorithm (Condensed Nearest Neighbor rule), which use distance metrics for non-metric spaces. By conducting experiments on eight datasets, the three variations of the CNN-rule algorithm were compared to the k-Nearest Neighbor algorithm without data reduction, evaluating accuracy and reduction rate. The experimental results demonstrate remarkable performance in all three variations of the CNN-rule algorithm.

Keywords: Data Reduction Techniques, k-Nearest Neighbor, Instance-Based Classification, Instance-Based Learning, Metric Spaces, Non-Metric Spaces, Distance Metrics, Distance Non-Metrics, Distance Functions, Hamming Distance, Edit Distance, Jaccard Distance, CNN-rule.

Πίνακας περιεχομένων

Πίνακας εικόνων

Πίνακας πινάκων

1

Εισαγωγικές έννοιες

Στο κεφάλαιο αυτό, γίνεται μια εισαγωγή στην διαδικασία της κατηγοριοποίησης (classification), η οποία εμφανίζεται σε πολλά ερευνητικά πεδία της πληροφορικής. Αρχικά, παρουσιάζονται κάποια εισαγωγικά θέματα σχετικά με την έννοια και τις βασικές τεχνικές της κατηγοριοποίησης, ενώ στη συνέχεια παρουσιάζονται ορισμένοι από τους γνωστούς κατηγοριοποιητές που χρησιμοποιούνται για την επίλυση προβλημάτων κατηγοριοποίησης. Πραγματοποιείται ανάλυση του κατηγοριοποιητή των k-Εγγύτερων Γειτόνων (k-NN), ο οποίος χρησιμοποιείται ευρέως από τις περισσότερες τεχνικές μείωσης του πληθυσμού των δεδομένων εκπαίδευσης. Ακολουθεί εκτενής αναφορά στις τεχνικές μείωσης του πληθυσμού των δεδομένων εκπαίδευσης και στο τέλος του κεφαλαίου παρουσιάζονται το κίνητρο που οδήγησε στην εκπόνηση της παρούσας διπλωματικής εργασίας, η συνεισφορά της, καθώς και η δομή που ακολουθείται στα επόμενα κεφάλαια της.

1.1 Κατηγοριοποίηση

Η κατηγοριοποίηση (classification) είναι μια από τις βασικές εργασίες της Εξόρυξης Δεδομένων, με μεγάλο αριθμό εφαρμογών τόσο στον ακαδημαϊκό χώρο, όσο και στον χώρο της οικονομίας και της βιομηχανίας [1]. Ο όρος κατηγοριοποίηση συναντάται συχνά στη βιβλιογραφία και ως ταξινόμηση. Επίσης, η κατηγοριοποίηση συγχέεται με το γενικό όρο της

πρόβλεψης. Επομένως, πρόκειται για μια προγνωστική (προβλεπτική) μέθοδο. Στην κατηγοριοποίηση, το αποτέλεσμα που θέλουμε να προβλέψουμε είναι η κλάση των δειγμάτων [2]. Η μέθοδος της κατηγοριοποίησης ωστόσο βασίζεται στην εξέταση των χαρακτηριστικών ενός νέου αντικειμένου (μη κατηγοριοποιημένο), το οποίο με βάση τα χαρακτηριστικά αυτά, αντιστοιχίζεται σε ένα προκαθορισμένο σύνολο κατηγοριών (κλάσεων). Γενικότερα, ο στόχος της διαδικασίας αυτής είναι η δημιουργία ενός μοντέλου – κατηγοριοποιητή (classifier) με βάση τα διαθέσιμα δεδομένα εκπαίδευσης, δηλαδή ένα σύνολο ήδη ταξινομημένων περιπτώσεων (ή στοιχείων), το οποίο αργότερα θα μπορεί να χρησιμοποιηθεί για την κατηγοριοποίηση μελλοντικών δεδομένων [3].

Ο σκοπός των τεχνικών κατηγοριοποίησης είναι η ομαδοποίηση δεδομένων στην απαιτούμενη δομή με βάση τα κοινά χαρακτηριστικά. Η διαδικασία της κατηγοριοποίησης βασίζεται στα χαρακτηριστικά τους για να είναι σε θέση να προβλέψει τη μελλοντική τους συμπεριφορά [10]. Η λειτουργία της διαδικασίας αυτής καθιστά δυνατό τον προσδιορισμό εκείνων των δεδομένων των οποίων η συνδεδεμένη κλάση είναι άγνωστη. Οι μέθοδοι κατηγοριοποίησης ορίζονται ως μοντέλα που παράγουν διαφορετικά αποτελέσματα. Όλες αυτές οι μέθοδοι περιλαμβάνουν ανάλυση και κατηγοριοποίηση βάσει των στιγμιότυπων στο σύνολο δεδομένων εκπαίδευσης [11]. Κάποια παραδείγματα εφαρμογών του τομέα συναντάμε στην ιατρική με την πρόβλεψη καρκινικών κυττάρων χαρακτηρίζοντάς τα ως καλοήθη ή κακοήθη [12], στον χώρο των οικονομικών με την κατηγοριοποίηση των πελατών μιας τράπεζας ανάλογα με την πιστωτική τους ικανότητα [5], στην βιομηχανία με τη βελτιστοποίηση και την ποιοτική ανάλυση των διαδικασιών παραγωγής σε διάφορες ενεργειακές εφαρμογές, τον προσδιορισμό της αντοχής των δομικών υλικών ακόμη και στον διαχωρισμό των μηνυμάτων ηλεκτρονικού ταχυδρομείου με βάση την επικεφαλίδα τους ή το περιεχόμενό τους είτε σε κατηγορίες τύπου "spam" ή κατηγορία "μη- spam" [13]. Ένας άλλος τρόπος κατηγοριοποίησης του προβλήματος μηχανικής μάθησης πραγματοποιείται με βάση το αποτέλεσμα που περιμένουμε, δηλαδή του στόχου που έχει ο αλγόριθμος. Η πιο απλή μορφή κατηγοριοποίησης είναι η δυαδική κατηγοριοποίηση (Binary Classification). Σε ορισμένες περιπτώσεις, η κατηγορία στόχος έχει μόνο δύο τιμές. Ωστόσο, όταν οι κατηγορίες δεν είναι δύο, τότε αντιμετωπίζουμε προβλήματα πολλαπλών κατηγοριών (Multiclass Classification). Σε αυτά τα προβλήματα, είναι αναγκαίο να καταταχθούν δείγματα σε περισσότερες από δύο κλάσεις [14].

Η κατηγοριοποίηση μπορεί να περιγραφεί ως μια διαδικασία δύο κύριων βημάτων. Αναλυτικότερα, οι εργασίες που λαμβάνουν χώρα σε κάθε βήμα είναι οι ακόλουθες [9]:

1. **Εκμάθηση** (Learning): Στο πρώτο βήμα της διαδικασίας, πραγματοποιείται η δημιουργία του μοντέλου κατηγοριοποίησης χρησιμοποιώντας ένα σύνολο προ-κατηγοριοποιημένων στιγμιότυπων, που ονομάζονται δεδομένα εκπαίδευσης (training data). Η επιλογή των

δεδομένων εκπαίδευσης είναι καθοριστικής σημασίας, γιατί το μοντέλο που θα προκύψει θα αποτυπώνει σχέσεις που υπάρχουν στο σύνολο εκπαίδευσης [5]. Τα δεδομένα εκπαίδευσης αναλύονται από ένα αλγόριθμο κατηγοριοποίησης, προκειμένου να σχηματιστεί το μοντέλο. Λόγω του ότι τα δεδομένα εκπαίδευσης ανήκουν σε μία προκαθορισμένη κατηγορία, η οποία είναι γνωστή, η κατηγοριοποίηση αποτελεί μέθοδος εποπτευομένης μάθησης (supervised learning). Το μοντέλο, που λέγεται και αλλιώς κατηγοριοποιητής (classifier), αναπαρίσταται με τη μορφή κανόνων κατηγοριοποίησης (classification rules), δέντρων απόφασης (decision trees), νευρωνικών δικτύων ή μαθηματικών τύπων [4].

2. **Κατηγοριοποίηση** (Classification): Μετά την δημιουργία του μοντέλου, το επόμενο βήμα είναι η αξιολόγηση του. Για να επιτευχθεί αυτό, χρησιμοποιούμε τα δοκιμαστικά δεδομένα (test data) για να υπολογίσουν την ακρίβεια του μοντέλου. Το μοντέλο κατηγοριοποιεί τα δοκιμαστικά δεδομένα. Κατά την διάρκεια της κατηγοριοποίησης, το μοντέλο χρησιμοποιείται για να αναγνωρίσει τα χαρακτηριστικά των νέων αντικειμένων και να τα κατατάξει στις κατηγορίες που έχει μάθει κατά τη διάρκεια της εκπαίδευσης. Έπειτα, η κατηγορία που σχηματίστηκε με βάση τα δοκιμαστικά δεδομένα συγκρίνεται με την πρόβλεψη που έγινε για τα δεδομένα εκπαίδευσης, τα οποία είναι ανεξάρτητα από αυτά της δοκιμής [6]. Όπως γίνεται κατανοητό, η εκτίμηση της απόδοσης ενός μοντέλου-αλγορίθμου κατηγοριοποίησης δεν κρίνεται μόνο από την ικανότητα του να ταξινομεί σωστά το σύνολο εκπαίδευσης, αλλά και να ταξινομεί ορθά δεδομένα τα οποία ήταν άγνωστα σε αυτό [7]. Αν τα δεδομένα εισόδου είναι ανεπαρκή ή μη-συναφή αυτό θα αντικατοπτριστεί στην περιγραφή του θέματος από τον αλγόριθμο και θα έχει αποτέλεσμα την εσφαλμένη κατηγοριοποίηση όταν δοκιμαστεί σε νέα δεδομένα [8].

Στην περίπτωση που το μοντέλο κριθεί αποδεκτό, τότε μπορεί να χρησιμοποιηθεί για την κατηγοριοποίηση μελλοντικών δειγμάτων δεδομένων, των οποίων η κατηγοριοποίηση είναι άγνωστη.

Ανάλογα με το είδος της μηχανικής μάθησης (επιβλεπόμενης ή μη επιβλεπόμενης) και τη μέθοδο που χρησιμοποιούν, υπάρχουν διάφορα παραδείγματα κατηγοριοποιητών. Μερικοί από τους πιο γνωστούς κατηγοριοποιητές που χρησιμοποιούνται σε διάφορα προβλήματα κατηγοριοποίησης αναφέρονται παρακάτω [9]:

Ο κατηγοριοποιητής **k εγγύτερων γειτόνων** (k-Nearest Neighbor, k-NN) είναι ένας από τους αλγόριθμους επιβλεπόμενης μάθησης, ο οποίος αποτελεί μη-παραμετρική μέθοδο ταξινόμησης [14]. Η κεντρική ιδέα είναι πως "όμοια αντικείμενα ανήκουν στην ίδια κλάση". Αρχικά ορίζουμε ένα μέτρο ομοιότητας ή απόστασης μεταξύ των στιγμιότυπων και θεωρούμε ότι αυτά τα στιγμιότυπα που βρίσκονται αρκετά κοντά μεταξύ τους θα μοιράζονται την ίδια κλάση. Επομένως, υπολογίζουμε τις αποστάσεις μεταξύ τους, βρίσκουμε τους κ

κοντινότερους γείτονες και συμβουλευόμαστε τη δική τους γνωστή κλάση με απώτερο σκοπό να προβλέψουμε την κλάση του νέου δείγματος. Ένα πρακτικό θέμα κατά την εφαρμογή του κατηγοριοποιητή k εγγύτερων γειτόνων είναι ότι ο συγκεκριμένος αλγόριθμος δεν κατασκευάζει κάποιο μοντέλο κατηγοριοποίησης. Αντίθετα, το σύνολο δεδομένων εκπαίδευσης παίζει τον ρόλο του μοντέλου κατηγοριοποίησης. Συνεπώς, η μέθοδος των k εγγύτερων γειτόνων αποτελεί μια πολύ απλή και εύκολη, ως προς την κατανόηση εφαρμογή [15][16]. Ο κατηγοριοποιητής με βάση τους k εγγύτερους γείτονες είναι το ερευνητικό αντικείμενο μελέτης της παρούσας διπλωματικής εργασίας.

Τα Δ**έντρα Απόφασης** (Decision Trees) ή Δέντρα Ταξινόμησης (Classification Trees) αποτελούν ένα απλό μοντέλο κατηγοριοποίησης και ενδεχομένως είναι ένα από τα πιο χρησιμοποιούμενα μοντέλα της επιβλεπόμενης μηχανικής μάθησης. Τα δομικά στοιχεία ενός δέντρου απόφασης είναι οι κόμβοι (κόμβοι) και τα φύλλα (leaves), τα οποία χρησιμοποιούνται για το διαχωρισμό του συνόλου δεδομένων σε κλάσεις, που ανήκουν στη μεταβλητή στόχο [17]. Ένα δέντρο απόφασης κατασκευάζεται σύμφωνα με ένα σύνολο εκπαίδευσης δεδομένων, έτσι ώστε κάθε εσωτερικός κόμβος να προσδιορίζει τον έλεγχο των γνωρισμάτων και κάθε φύλλο να αντιστοιχεί σε μια από τις προκαθορισμένες κλάσεις. Στη συνέχεια πρέπει να το διασχίσουν με βάση τα χαρακτηριστικά των νέων στιγμιότυπων, τα οποία και ταξινομούνται ανάλογα με το φύλλο του δένδρου στο οποίο θα καταλήξουν [18]. Η μέθοδος κατηγοριοποίησης των δέντρων αποφάσεων έχει εφαρμοστεί με επιτυχία σε πολλούς τομείς όπως είναι η ιατρική, η τεχνολογία, ο στρατηγικός σχεδιασμός του marketing και άλλα [19].

Ο κατηγοριοποιητής **Bayes** αποτελεί μια απλή, γρήγορη και αρκετά αποτελεσματική μέθοδο κατηγοριοποίησης, η οποία χρησιμοποιεί πιθανοτικά μοντέλα τα οποία στηρίζονται στη στατιστική θεωρία κατηγοριοποίησης του Bayes [20]. Η Bayesian κατηγοριοποίηση εφαρμόζεται για την εκτίμηση της πιθανότητας ενός στιγμιότυπου να ανήκει σε μια από τις προκαθορισμένες κλάσεις [21]. Ο κατηγοριοποιητής Bayes [22][23] υποθέτει ότι η επίδραση ενός γνωρίσματος σε μία κατηγορία είναι ανεξάρτητη από τις τιμές των υπόλοιπων γνωρισμάτων. Η υπόθεση της ανεξαρτησίας των χαρακτηριστικών δεν ισχύει πάντα, όμως απλοποιεί κατά πολύ τους υπολογισμούς οδηγώντας σε καλή εκτίμηση της πιθανότητας χωρίς να απαιτεί μεγάλο σύνολο εκπαίδευσης δεδομένων [20]. Η τεχνική κατηγοριοποίησης του Bayes [24] είναι αποτελεσματική σε πολλές πρακτικές εφαρμογές, συμπεριλαμβανομένης της ταξινόμησης κειμένων, της ιατρικής διάγνωσης και της διαχείρισης απόδοσης συστημάτων.

Στη Μηχανική Μάθηση, η μέθοδος κατηγοριοποίησης βασισμένη στις **Μηχανές** Δ**ιανυσμάτων Υποστήριξης** (Support Vector Machines, SVM) είναι κατάλληλη για σύνολα δεδομένων πολλών διαστάσεων, ενώ μπορεί κάλλιστα να χρησιμοποιηθεί και για κατηγορικές μεταβλητές [25]. Η μέθοδος των μηχανών διανυσμάτων υποστήριξης προτάθηκε για πρώτη

φορά το 1995 από τον Vapnik και την ομάδα του, και γρήγορα προσέλκυσε το ενδιαφέρον πολλών ερευνητών αφού παρουσίασε μεγάλη ικανότητα γενίκευσης σε σχέση με άλλες παραδοσιακές μεθόδους κατηγοριοποίησης [26][9]. Ο διαχωρισμός των δεδομένων αυτής της κατηγοριοποίησης πραγματοποιείται με τη βοήθεια γραμμικής ή μη γραμμικής λειτουργίας [27]. Η βασική ιδέα των SVM είναι η κατασκευή ενός βέλτιστου υπερ-επιπέδου (hyperplane), το οποίο μέσω μιας ειδικής διαχωριστικής γραμμής χωρίζει τις κλάσεις μεταξύ τους [10]. Στόχος αυτής της μεθόδου είναι η επιλογή ενός μικρού αριθμού στιγμιότυπων εκπαίδευσης από κάθε κλάση, τα οποία ονομάζονται διανύσματα υποστήριξης (support vectors). Αυτά τα διανύσματα υποστήριξης ορίζουν τα όρια μεταξύ των διαφορετικών κλάσεων και χρησιμοποιούνται για την εύρεση ενός υπερ-επιπέδου με το μεγαλύτερο περιθώριο γραμμικού διαχωρισμού τους [28]. Οι μηχανές διανυσμάτων υποστήριξης έχουν πλήθος εφαρμογών όπως αναγνώριση γραφής (handwriting recognition), ταξινόμηση κειμένων (text categorization), στον κλάδο της ιατρικής και της οικονομίας [29].

Τα **Νευρωνικά Δίκτυα** (Neural Networks) μπορούν επίσης να χρησιμοποιηθούν ως κατηγοριοποιητές. Εμπνευσμένα από το βιολογικό νευρικό σύστημα, και ειδικότερα από τον τρόπο με τον οποίο λειτουργεί ο ανθρώπινος εγκέφαλος, διαθέτουν αξιοσημείωτα χαρακτηριστικά, όπως τη δυνατότητα τους να αναπαριστούν σύνθετες εξαρτήσεις ή την ικανότητα τους να προβλέπουν την κλάση άγνωστων παρατηρήσεων [30]. Ενα Τεχνητό Νευρωνικό Δίκτυο (Artificial Neural Networks) αποτελεί μια χαρακτηριστική μέθοδο μοντελοποίησης σύνθετων προβλημάτων πρόβλεψης [28], και βασίζεται σε μια συλλογή συνδεδεμένων μονάδων ή κόμβων που ονομάζονται Τεχνητοί Νευρώνες (Artificial Neurons). Οι νευρώνες συνδέονται μεταξύ τους με διάφορους τρόπους, με σκοπό να επιτρέψουν την έξοδο ορισμένων νευρώνων να γίνει η είσοδος άλλων. Επομένως, το δίκτυο σχηματίζει ένα κατευθυνόμενο, σταθμισμένο γράφημα [30]. Χάρη στη στιβαρή θεωρητική τους θεμελίωση και στις ιδιαίτερες δυνατότητες τους έχουν καταστεί ιδιαίτερα δημοφιλή σε προβλήματα που δεν μπορούν να γίνουν προβλέψεις και έχουν εφαρμοστεί σε πολλούς τομείς, όπως είναι η ιατρική, η οικονομία και η άμυνα [5].

Παραγωγή κανόνων κατηγοριοποίησης. Η γνώση που αποκτούμε κατά την διαδικασία της κατηγοριοποίησης μπορεί να αναπαρασταθεί και με τη χρήση κανόνων κατηγοριοποίησης. Οι κανόνες κατηγοριοποίησης σε σχέση με τα δέντρα απόφασης, γίνονται ευκολότερα κατανοητοί όταν το δέντρο που παράχθηκε είναι μεγάλο. Έτσι μπορούμε να μετατρέψουμε ένα δέντρο απόφασης σε ένα σύνολο κανόνων κατηγοριοποίησης. Αυτό μπορεί να επιτευχθεί εάν θεωρήσουμε ότι κάθε κανόνας αντιστοιχεί σε ένα μονοπάτι του δέντρου από τη ρίζα μέχρι ένα φύλλο [31]. Επομένως, για κάθε φύλλο δημιουργείται ένας κανόνας, ο οποίος έχει τη μορφή IF-THEN και περιλαμβάνει τις λογικές συνθήκες όλων των ελέγχων. Οι έλεγχοι συνήθως χρησιμοποιούν τις λογικές συζεύξεις με συχνότερη να

εμφανίζεται η AND [5]. Οι συνθήκες που θα μας οδηγήσουν στο φύλλο (υπόθεση) αποτελούν το αριστερό μέρος του κανόνα, ενώ το φύλλο (αποτέλεσμα) αντιστοιχεί στο δεξιό μέρος του κανόνα [31]. Στον τομέα της ιατρικής, η παραγωγή κανόνων κατηγοριοποίησης μπορεί να βοηθήσει στη διάγνωση ασθενειών ή στην πρόβλεψη του κινδύνου ανάπτυξης τους, λαμβάνοντας υπόψη την ατομική ιστορία και τα χαρακτηριστικά του ασθενούς [32].

Όταν μιλάμε για την επίτευξη της μάθησης στους ανθρώπους, αναφερόμαστε συχνά στην ικανότητα τους να κατανοούν ένα αντικείμενο. Η χρήση τέτοιων εννοιών, μπορεί να μας οδηγήσει σε εσφαλμένο συμπέρασμα ότι η κατανόηση είναι απαραίτητη για τη μάθηση. Υπάρχουν και περιπτώσεις που δεν ισχύει αυτό πλήρως. Στους ζωντανούς οργανισμούς πολύ συχνά η μάθηση προέρχεται από την εμπειρία και όχι από τη κατανόηση.

Έτσι και στη μηχανική μάθηση, μπορούμε να διακρίνουμε τους τρόπους που μπορεί να μάθει η μηχανή, σε μεθόδους που βασίζονται στη κατανόηση και δημιουργούν μοντέλα ταξινόμησης και σε μεθόδους που βασίζονται στις εμπειρίες τους, τις οποίες αποθηκεύουν για μελλοντική χρήση. Οι πρώτες ονομάζονται μέθοδοι Πρόθυμης Μάθησης (Eager Learning) και οι δεύτερες τεχνικές Οκνηρής Μάθησης (Lazy Learning) [33]. Τα περισσότερα μοντέλα κατηγοριοποίησης στην Εξόρυξη Δεδομένων που έχουμε περιγράψει παραπάνω με εξαίρεση του k εγγύτερων γειτόνων (k-NN), είναι όλα μοντέλα πρόθυμης όπως λέγεται μάθησης και αφορούν τους πρόθυμους κατηγοριοποιητές (eager classifiers) [9]. Αυτό σημαίνει ότι όταν δίνεται ένα σύνολο δεδομένων για εκπαίδευση θα κατασκευαστεί ένα γενικευμένο μοντέλο πριν ακόμη γίνουν δεκτά νέα στιγμιότυπα προς κατηγοριοποίηση. Μπορούμε να θεωρήσουμε ότι το μοντέλο είναι έτοιμο και πρόθυμο να κατηγοριοποιήσει τα άγνωστα μέχρι πριν στιγμιότυπα [34]. Η διαφορά με την οκνηρή μάθηση είναι ότι στην πρόθυμη μάθηση το σύστημα κατά την διάρκεια της εκπαίδευσης του προσπαθεί να κατασκευάσει έναν ανεξάρτητο στόχο εισόδου [37]. Αντιθέτως στην οκνηρή προσέγγιση (lazy approach) περιμένουμε μέχρι την τελευταία στιγμή, πριν γίνει κατασκευή οποιουδήποτε μοντέλου, να κατηγοριοποιηθεί το σύνολο των δοκιμαστικών στιγμιότυπων. Ένας οκνηρός λοιπόν κατηγοριοποιητής (ή τεμπέλης, όπως συναντάται στην βιβλιογραφία - lazy classifier) απλά αποθηκεύει ένα στιγμιότυπο με γνωστή κατηγορία, ή κάνει μια πολύ μικρή επεξεργασία του και στη συνέχεια περιμένει μέχρι να λάβει ένα δοκιμαστικό στιγμιότυπο. Μόνο όταν δει το δοκιμαστικό στιγμιότυπο εφαρμόζει γενίκευση για να κατηγοριοποιηθεί με βάση την ομοιότητα του με τα αποθηκευμένα εκπαιδευμένα στιγμιότυπα.

Σε αντίθεση με τους πρόθυμους κατηγοριοποιητές, οι οκνηροί κατηγοριοποιητές κάνουν λιγότερη δουλειά όταν εμφανίζεται ένα στιγμιότυπο προς εκπαίδευση και περισσότερη όταν είναι να γίνει κατηγοριοποίηση ή αριθμητική πρόβλεψη. Αυτό σημαίνει ότι οι μέθοδοι οκνηρής μάθησης είναι πιο γρήγοροι στην εκπαίδευση με μικρό υπολογιστικό κόστος. Στην εξόρυξη δεδομένων επειδή οι οκνηροί κατηγοριοποιητές αποθηκεύουν τα

στιγμιότυπα αναφέρονται και ως κατηγοριοποιητές βασισμένοι σε στιγμιότυπα, διότι σε αυτά βασίζεται όλη η διαδικασία της μάθησης. Για να γίνει μια κατηγοριοποίηση ή μια αριθμητική πρόβλεψη, η χρήση οκνηρών κατηγοριοποιητών μπορεί να έχει μεγάλο υπολογιστικό κόστος. Απαιτούν αποδοτικές τεχνικές αποθήκευσης και θα πρέπει να προσαρμόζονται σε εφαρμογές με υλικό που λειτουργεί με παραλληλοποίηση, ενώ δεν δίνουν πολλές λεπτομέρειες για την εσωτερική δομή των δεδομένων. Παρόλα αυτά όμως υποστηρίζουν από τη φύση τους τη στοιχειώδη εκπαίδευση. Μπορούν να μοντελοποιήσουν πολύπλοκους χώρους αποφάσεων που έχουν υπερπολυγωνικά σχήματα, τα οποία δεν είναι εύκολα να περιγραφούν από άλλους αλγόριθμους [34][35].

Συνοψίζοντας, τα δένδρα αποφάσεων κατηγοριοποίησης [36] αποτελούν μια πολύ γνωστή υποκατηγορία πρόθυμων κατηγοριοποιητών. Άλλοι πρόθυμοι κατηγοριοποιητές βασίζονται στις μηχανές διανυσμάτων υποστήριξης και στα τεχνητά νευρωνικά δίκτυα. Ένα χαρακτηριστικό παράδειγμα ενός πιθανοτικού κατηγοριοποιητή είναι ο αφελής μπαϋεσιανός κατηγοριοποιητής (naive bayes classifier). Από την άλλη πλευρά, η κατηγορία των οκνηρών κατηγοριοποιητών περιλαμβάνει τον γνωστό κατηγοριοποιητή k-Εγγύτερων Γειτόνων (k-Nearest Neighbors) [15][16].

1.2 Κατηγοριοποίηση με βάση τα στιγμιότυπα (instance-based classification)

Οι κατηγοριοποιητές βασισμένοι σε στιγμιότυπα (instance-based classifiers), γνωστοί επίσης ως οκνηροί κατηγοριοποιητές ή κατηγοριοποιητές που βασίζονται στην μνήμη, είναι μια κατηγορία αλγορίθμων μηχανικής μάθησης που χρησιμοποιούνται για την κατηγοριοποίηση δεδομένων σε διάφορες κατηγορίες [38]. Σε αντίθεση με τους παραδοσιακούς αλγόριθμους κατηγοριοποίησης που χρησιμοποιούν επεκτατικές περιγραφές εννοιών ή κανόνες [39], οι μέθοδοι μάθησης βασισμένες σε στιγμιότυπα (instance-based methods) στηρίζονται στην ομοιότητα μεταξύ των δεδομένων, η οποία υπολογίζεται με τη χρήση κατάλληλης μετρικής απόστασης (distance metric) [35]. Ο βασικός τρόπος λειτουργίας των κατηγοριοποιητών είναι να χρησιμοποιούν μετρικές απόστασης για να εντοπίσουν τις πλησιέστερες κατηγορίες ή στιγμιότυπα στο χώρο των δεδομένων. Όταν πρέπει να ταξινομήσουν ένα νέο στιγμιότυπο, αναζητούν τους κοντινότερους γείτονες του και βασίζονται στις κατηγορίες τους για να καθορίσουν την κατηγορία του νέου στιγμιότυπου. Επομένως, οι μετρικές απόστασης και το πλήθος των πλησιέστερων γειτόνων (στιγμιότυπων) είναι οι παράμετροι που παίζουν κρίσιμο ρόλο στην εκτέλεση αυτών των κατηγοριοποιητών.

Το σύνολο των κατηγοριοποιητών βασισμένων σε στιγμιότυπα αναγνωρίζεται ως ασταθές, διότι μια αλλαγή στις παραμέτρους τους μπορεί να αλλάξει τη συνολική τους απόδοση [38].

Η μέθοδος των κατηγοριοποιητών με βάση τα στιγμιότυπα αποτελεί ουσιαστικά μια εφαρμογή της μάθησης που βασίζεται σε στιγμιότυπα (instance-based learning) για την κατηγοριοποίηση νέων στιγμιότυπων. Η διαδικασία αυτής της μάθησης σχετίζεται, αλλά δεν είναι ακριβώς η ίδια με τη συλλογιστική βασισμένη στις περιπτώσεις (case-based reasoning), στην οποία μπορούν να χρησιμοποιηθούν προηγούμενα στιγμιότυπα για να γίνουν προβλέψεις σχετικά με συγκεκριμένες δοκιμές στιγμιότυπων. Τέτοια συστήματα μπορούν να τροποποιήσουν περιπτώσεις ή να χρησιμοποιήσουν τμήματα περιπτώσεων για να κάνουν προβλέψεις. Οι μέθοδοι που βασίζονται σε στιγμιότυπα μπορούν να θεωρηθούν ως ένα συγκεκριμένο είδος προσέγγισης που βασίζεται σε περιπτώσεις, το οποίο χρησιμοποιεί συγκεκριμένα είδη αλγορίθμων για κατηγοριοποίηση με βάση τα στιγμιότυπα. Το πλαίσιο των αλγορίθμων που βασίζονται σε στιγμιότυπα είναι πιο επιδεκτικό για τη μείωση των απαιτήσεων υπολογισμού και αποθήκευσης, του θορύβου και των άσχετων χαρακτηριστικών. Ωστόσο, αυτές οι ορολογίες δεν διαφέρουν σαφώς η μία από την άλλη, επειδή πολλοί ερευνητές χρησιμοποιούν στην βιβλιογραφία τον όρο «μάθηση βασισμένη σε περιπτώσεις» για να αναφέρονται σε αλγόριθμους μάθησης που βασίζονται σε στιγμιότυπα [39].

Η μάθηση βασισμένη στα στιγμιότυπα είναι μια από τις πολλές μεθόδους εκπαίδευσης της μηχανικής μάθησης. Η μέθοδος όμως αυτή έχει μια ιδιαιτερότητα. Γενικά στις περισσότερες μεθόδους κατηγοριοποίησης κατασκευάζεται εξαρχής ένα μοντέλο εκπαίδευσης από τα υπάρχοντα δεδομένα και στη συνέχεια απλώς πραγματοποιείται η κατηγοριοποίηση για κάθε νέο στιγμιότυπο βασιζόμενο σε αυτό το μοντέλο [39]. Η διαδικασία της μάθησης βασισμένη σε στιγμιότυπα όπως έχει ήδη επισημανθεί είναι διαφορετική. Τα αρχικά δεδομένα αποθηκεύονται ως έχουν, και κάθε φορά που ένα καινούργιο στιγμιότυπο έρχεται προς κατηγοριοποίηση, τότε αυτή γίνεται ανάλογα με τα στιγμιότυπα που έχουν αποθηκευτεί έως εκείνη τη στιγμή. Για αυτόν τον λόγο πολλές φορές οι μέθοδοι μάθησης βασισμένες στα στιγμιότυπα αναφέρονται στην βιβλιογραφία και ως «τεμπέλικες» (Lazy) [40]. Η τεμπέλικη πτυχή αυτής της διαδιδκασίας της μάθησης είναι το μεγαλύτερο πλεονέκτημά της.

Είναι προφανές, ότι στην πρώτη διαδικασία κατηγοριοποίησης ο υπολογιστικός χρόνος είναι αρκετά μικρότερος, αφού η εκπαίδευση γίνεται μόνο μια φορά στην αρχή, ενώ αντίθετα στην δεύτερη διαδικασία κατηγοριοποίησης, κάθε φορά που έρχεται ένα νέο στιγμιότυπο γίνεται εκπαίδευση εξαρχής. Αν και η μέθοδος κατηγοριοποίησης με βάση τα στιγμιότυπα μπορεί να είναι απλή, ο υπολογιστικός χρόνος μπορεί να αυξηθεί σημαντικά καθώς αυξάνεται το μέγεθος του συνόλου δεδομένων. Ως αποτέλεσμα, μπορεί συχνά να μην είναι δυνατή η δημιουργία πολύπλοκων μοντέλων λόγω των υπολογιστικών απαιτήσεων. Σε

ορισμένες περιπτώσεις, αυτό μπορεί να οδηγήσει σε υπεραπλούστευση. Από την άλλη πλευρά, η μάθηση βασισμένη στα στιγμιότυπα έχει το πλεονέκτημα ότι η εκπαίδευση γίνεται κάθε φορά με όλα τα υπάρχοντα στιγμιότυπα και όχι μόνο με τα αρχικά, οπότε η κατηγοριοποίηση είναι συνήθως καλύτερη και αποφεύγεται η πιθανότητα της υπερπροσαρμογής (overfitting) [39].

Ένα άλλο πλεονέκτημα της μάθησης βασισμένη στα στιγμιότυπα είναι η ικανότητά των αλγορίθμων που βασίζονται σε στιγμιότυπα να χειρίζονται αριθμητικά δεδομένα και να προβλέπουν αποτελεσματικά κλάσεις με αριθμητική αξία. Αυτό συμβαίνει επειδή αυτοί οι αλγόριθμοι μηχανικής μάθησης διατηρούν κάθε στιγμιότυπο εκπαίδευσης ως μια ξεχωριστή έννοια και δεν προσπαθούν να οικοδομήσουν ένα γενικευμένο μοντέλο που αντιπροσωπεύει το πρόβλημα χρησιμοποιώντας έναν μικρό αριθμό εννοιών ή κατηγοριών. Αντίθετα, αποθηκεύουν και χρησιμοποιούν ολόκληρο το σύνολο δεδομένων εκπαίδευσης κατά τη φάση της δοκιμής ή της πρόβλεψης [43].

Η μάθηση με βάση τα στιγμιότυπα περιλαμβάνει τους αλγόριθμους του κοντινότερου γείτονα (Nearest Neighbor), π.χ. την μάθηση με βάση τους k εγγύτερους γείτονες (k-Nearest Neighbors), τους αλγόριθμους της τοπικά σταθμισμένης παλινδρόμησης (Locally Weighed Regression), τις συναρτήσεις ακτινικής βάσης (Radial Basis Functions) και τη συλλογιστική βασισμένη στις περιπτώσεις (Case Based Reasoning) [41][42].

Οι αλγόριθμοι που βασίζονται σε στιγμιότυπα αντιμετωπίζουν πολλές προκλήσεις που αφορούν την απόδοση και τις σημαντικές απαιτήσεις αποθήκευσης [39]. Ανάλογα με το μέγεθος του συνόλου δεδομένων εκπαίδευσης, οι κατηγοριοποιητές βασισμένοι σε στιγμιότυπα μπορεί να απαιτούν μεγάλο χώρο μνήμης για να αποθηκεύσουν όλα τα στιγμιότυπα. Αυτό μπορεί να είναι ιδιαίτερα πρόβλημα όταν έχουμε να διαχειριστούμε μεγάλα σύνολα δεδομένων. Μια άλλη πρόκληση είναι η απόδοση της κατηγοριοποίησης. Σε κάποιες περιπτώσεις, οι κατηγοριοποιητές βασισμένοι σε στιγμιότυπα μπορεί να μην έχουν την ίδια ακρίβεια με κάποιες πιο προηγμένες τεχνικές μηχανικής μάθησης. Αυτό συμβαίνει ειδικά όταν το σύνολο δεδομένων είναι μεγάλο και αρκετά πολύπλοκο. Η απόδοση της κατηγοριοποίησης συχνά υπερβαίνει το 75% του μέγιστου δυνατού μετά την αποδοχή μόνο του 25% ενός πλήρους συνόλου δεδομένων. Επομένως, η προσέγγιση λειτουργεί καλά όταν υπάρχουν περιορισμένα δεδομένα [43].

Για να αντιμετωπίσουμε αυτές τις προκλήσεις, είναι δυνατόν να χρησιμοποιήσουμε τεχνικές όπως η επιλογή μόνο των σημαντικών χαρακτηριστικών εισόδου, η επιλογή κατάλληλων μετρικών αξιολόγησης του μοντέλου [44] και η εφαρμογή τεχνικών συμπίεσης δεδομένων για μείωση της μνήμης [45].

Συνοψίζοντας, η μάθηση βασισμένη σε στιγμιότυπα και οι κατηγοριοποιητές με βάση τα στιγμιότυπα είναι σημαντικές εργασίες στον τομέα της εξόρυξης δεδομένων, παρέχοντας

ευελιξία και απλότητα στην υλοποίηση [46]. Ωστόσο, πρέπει να επιλυθούν ορισμένα ζητήματα πριν από την ευρεία τους εφαρμογή σε πιο προηγμένα προβλήματα μηχανικής μάθησης.

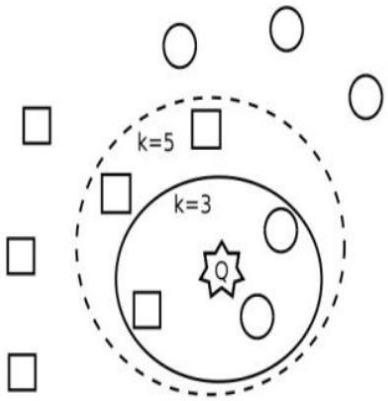
1.3 Κατηγοριοποίηση με βάση τους k-Εγγύτερους Γείτονες (k-Nearest Neighbors)

Ο αλγόριθμος κατηγοριοποίησης των k-Εγγύτερων (ή κοντινότερων) Γειτόνων (k-Nearest Neighbor, k-NN) αποτελεί την πιο απλή μέθοδο μηχανικής μάθησης και λειτουργεί μέσω μάθησης βασισμένης σε στιγμιότυπα (instance-based learning) [15][16]. Στην αναγνώριση προτύπων, ο κατηγοριοποιητής k-Εγγύτερων Γειτόνων είναι ένας ακόμη αλγόριθμος επιβλεπόμενης μάθησης, ο οποίος αποτελεί μη-παραμετρική μέθοδο κατηγοριοποίησης που βασίζεται στην απόσταση των δεδομένων για την ετικετοποίηση (κατηγοριοποίηση) ενός νέου στιγμιότυπου. Ιστορικά, πρωτοεμφανίστηκε από την Fix και τον Hodges το 1951 και αργότερα επεκτάθηκε από τον Cover [47]. Η κεντρική ιδέα του αλγορίθμου εγγύτερων γειτόνων είναι πως "όμοια αντικείμενα ανήκουν στην ίδια κλάση". Αυτή η αρχή της ομοιότητας μεταξύ δειγμάτων μπορεί να θεωρηθεί ως μια παρατήρηση, η οποία αποτελεί τη βάση μιας αρκετά δημοφιλούς προσέγγισης στην κατηγοριοποίηση [48]. Επίσης, θεωρείται ένας από τους πιο απλούς και εύχρηστους κατηγοριοποιητές, καθώς μπορεί να χρησιμοποιηθεί σε πολλούς τομείς εφαρμογών και να ενσωματωθεί εύκολα σε πολλά συστήματα [46].

Ο κατηγοριοποιητής k-Εγγύτερων Γειτόνων ανήκει στους οκνηρούς αλγόριθμους επειδή απομνημονεύει το σύνολο δεδομένων εκπαίδευσης. Ένας οκνηρός λοιπόν κατηγοριοποιητής δεν δημιουργεί κανένα μοντέλο κατηγοριοποίησης αλλά παράγει τη βάση της γνώσης του αποθηκεύοντας όλα τα δεδομένα εκπαίδευσης [34]. Στην κατηγοριοποίηση παρέχεται στον αλγόριθμο αρχικά ένα επισημασμένο (labeled) σύνολο δεδομένων εκπαίδευσης όπου τα δεδομένα κατηγοριοποιούνται σε κατηγορίες με σκοπό να μπορεί να πραγματοποιηθεί η πρόβλεψη της κατηγορίας των μη επισημασμένων δεδομένων (unlabeled data).

Τα δεδομένα εκπαίδευσης τυγχάνουν επεξεργασίας όταν εμφανιστεί ένα νέο στιγμιότυπο. Κάθε φορά που ένα νέο στιγμιότυπο πρόκειται να κατηγοριοποιηθεί, υπολογίζεται η ομοιότητα του με κάθε ένα από τα πλησιέστερα ή τα γειτονικά δεδομένα εκπαίδευσης, σύμφωνα με μια μετρική απόστασης [49]. Οι εγγύτεροι γείτονες ενός στιγμιότυπου ορίζονται συνήθως με την έννοια της Ευκλείδειας απόστασης, χωρίς όμως αυτό να αποκλείει και χρήση κάποιας παραλλαγής της ή και άλλης μετρικής [34]. Υπολογίζονται οι k πλησιέστεροι γείτονες και η πλειοψηφία μεταξύ των γειτονικών δεδομένων καθορίζει

την κατηγοριοποίηση των νέων δεδομένων. Η τιμή του k είναι σημαντική [50] για την κατηγοριοποίηση των μη επισημασμένων δεδομένων (unlabeled data). Για τον υπολογισμό της καλύτερης τιμής του k πρέπει να εκτελεστεί ο αλγόριθμος k-Εγγύτεροι Γείτονες αρκετές φορές με διαφορετική τιμή στο k. Με αυτόν τον τρόπο θα παρατηρήσουμε ποια τιμή στο k δίνει το αποτελεσματικότερο αποτέλεσμα. Το νέο στιγμιότυπο ανατίθεται στην κατηγορία που είναι επικρατέστερη μεταξύ των κοντινότερων γειτόνων του. Ένα παράδειγμα φαίνεται στην Εικόνα 1. Τα δεδομένα εκπαίδευσης είναι χωρισμένα σε δύο κατηγορίες, οι κύκλοι και τα τετράγωνα. Ο αστερίσκος είναι το αντικείμενο προς ταξινόμηση. Στην Εικόνα 1 φαίνεται πως για k = 3, το οποίο υποδεικνύει στο σχήμα ο εσωτερικός από τους δύο ομόκεντρους κύκλους, η πρόβλεψη του αλγορίθμου είναι η κατηγορία κύκλος. Ενώ για k = 5, το οποίο υποδεικνύει ο εξωτερικός κύκλος με τις διακεκομμένες γραμμές, η πρόβλεψη είναι η κατηγορία τετράγωνο. Στην περίπτωση που το k = 1, ο αλγόριθμος είναι επίσης γνωστός ως κατηγοριοποιητής πλησιέστερου γείτονα (ή κανόνας 1-NN) [51].



Εικόνα 1: Καθορισμός κλάσης νέου στιγμιότυπου για k = 3 και k = 5

Ο αλγόριθμος των k-Εγγύτερων Γειτόνων (k-Nearest Neighbor, k-NN) συνοψίζεται παρακάτω:

1. Καθορισμός της παραμέτρου k και της μετρικής.
2. Εύρεση των k πλησιέστερων γειτόνων με βάση τη μετρική.
3. Καταμέτρηση του αριθμού των δεδομένων σε κάθε κατηγορία.
4. Αντιστοίχιση των νέων δεδομένων στην κατηγορία με τον μέγιστο αριθμό γειτόνων.

Καθορισμός της παραμέτρου k. Αν παρατηρήσουμε την Εικόνα 1, θα συμπεράνουμε πως ο κατάλληλος καθορισμός της παραμέτρου k, αποτελεί ένα από τα

κυριότερα ζητήματα της μεθόδου των k-Εγγύτερων Γειτόνων. Η παράμετρος k επηρεάζει τα αποτελέσματα της κατηγοριοποίησης, και δεν είναι κάποια τυχαία τιμή, αλλά εξαρτάται απαραίτητα από τα δεδομένα του κάθε προβλήματος.

Η επιλογή της παραμέτρου k παίζει πολύ σημαντικό ρόλο στην αποδοτικότητα του αλγορίθμου και είναι δύσκολο να προσδιοριστεί [44]. Συγκεκριμένα, η επιλογή μεγάλων τιμών της k αυξάνουν την επίδραση του θορύβου στο μοντέλο, αλλά κάνουν ασαφή τα όρια μεταξύ των κατηγοριών [52]. Από την άλλη, αν η k παίρνει μικρές τιμές, τότε το μοντέλο καταλήγει να είναι ασταθές ως προς τον θόρυβο. Επομένως, θα πρέπει να βρεθεί η βέλτιστη τιμή της k, η οποία θα προσδίδει ισορροπία στο μοντέλο. Μια καλή τιμή για το k μπορεί να βρεθεί χρησιμοποιώντας διάφορες ευριστικές μεθόδους. Η τιμή της k μπορεί δηλαδή να καθοριστεί μέσω της τεχνικής Διασταυρούμενης Επικύρωσης (Cross Validation) [44]. Στη συνηθισμένη περίπτωση των δύο κατηγοριών, είναι προτιμότερο να χρησιμοποιείται περιττός αριθμός για την τιμή της k ώστε να αποφεύγονται οι ισοψηφίες [53]. Επίσης, έχουν υπάρξει και άλλες παραλλαγές του κλασικού αλγορίθμου k-Εγγύτερων Γειτόνων, όπως είναι ο Τροποποιημένος αλγόριθμος k-Εγγύτερων Γειτόνων (Modified k-Nearest Neighbor, MkNN). Η μέθοδος καθορισμού της k είναι πολύ ευαίσθητη τόσο στο θόρυβο όσο και στα δεδομένα του προβλήματος. Σε αυτόν τον τροποποιημένο αλγόριθμο, αποδίδεται ένα βάρος σε κάθε στιγμιότυπο που χρησιμοποιείται για τον υπολογισμό του αποτελέσματος. Αυτό το βάρος υπολογίζεται με βάση δύο παράγοντες, την εγκυρότητα του δεδομένου εκπαίδευσης και την Ευκλείδεια απόσταση ανάμεσα στο δεδομένο εκπαίδευσης και το στιγμιότυπο που αξιολογείται. Αυτή η τροποποίηση επιτρέπει στον MkNN να αντιμετωπίσει καλύτερα το θόρυβο και τις ανεπιθύμητες επιπτώσεις των δεδομένων εκπαίδευσης, καθιστώντας τον πιο ευέλικτο σε διάφορα προβλήματα κατηγοριοποίησης [54].

Καθορισμός της μετρικής. Ο κανόνας των k-Εγγύτερων Γειτόνων κατηγοριοποιεί κάθε μη ετικετοποιημένο (κατηγοριοποιημένο) στιγμιότυπο βάσει της πλειοψηφίας των k εγγύτερων γειτόνων του συνόλου εκπαίδευσης. Συνεπώς, η απόδοση του αλγορίθμου εξαρτάται σε μεγάλο βαθμό από την απόσταση που μετριέται ώστε να εντοπιστούν οι k εγγύτεροι γείτονες [55]. Με άλλα λόγια, ένας σημαντικός παράγοντας για τη λειτουργία του αλγορίθμου των k-Εγγύττερων Γειτόνων είναι ο καθορισμός της κατάλληλης μετρικής. Για το συγκεκριμένο ζήτημα υπάρχουν αρκετές δυνατές επιλογές [56]. Η απόφαση εξαρτάται από τα ειδικά χαρακτηριστικά του χώρου στιγμιότυπων του προβλήματος. Ιδιαίτερη σημασία έχει αν στην αναπαράσταση των στιγμιότυπων περιλαμβάνονται αριθμητικά ή συμβολικά χαρακτηριστικά. Επιπλέον, όταν τα δεδομένα είναι πυκνά ή συνεχόμενα, η Ευκλείδεια απόσταση είναι το καλύτερο μέτρο εγγύτητας. Στην περίπτωση που τα χαρακτηριστικά είναι ονομαστικά, η Ευκλείδια απόσταση δεν είναι δυνατόν να χρησιμοποιηθεί, επειδή δεν έχει νόημα η αφαίρεση συμβολικών ποσοτήτων [57]. Ο αλγόριθμος των k-Εγγύττερων Γειτόνων

υποθέτει ότι όλα τα στιγμιότυπα απεικονίζονται σε σημεία του n-διάστατου χώρου [44]. Για λόγους απλότητας θεωρούμε αρχικά ένα πρόβλημα κατηγοριοποίησης, όπου οι παρατηρήσεις αποτελούνται από δύο αριθμητικά πεδία και το γνώρισμα της κλάσης. Κάθε παρατήρηση μπορεί να θεωρηθεί ως ένα σημείο στον χώρο των δύο διαστάσεων. Μια παρατήρηση X απέχει από μια άλλη παρατήρηση Y, απόσταση d(X, Y) μέσα στον δισδιάστατο χώρο. Η απόσταση d(X, Y) μπορεί να υπολογιστεί ως η Ευκλείδεια απόσταση σύμφωνα με την εξίσωση:

$$d(X,Y) = \sqrt{((x_1 - y_1)^2 + (x_2 - y_2)^2)}$$

όπου x_1, y_1 οι τιμές των X και Y για την πρώτη διάσταση και x_2, y_2 οι τιμές των X και Y για τη δεύτερη διάσταση [9].

Ένας περιορισμός του υπολογισμού της ομοιότητας με βάση την Ευκλείδεια απόσταση ή κάποια παραλλαγή της είναι το γεγονός ότι αυτές οι προσεγγίσεις προϋποθέτουν γνωρίσματα αριθμητικών τιμών. Για να αντιμετωπιστεί αυτό το πρόβλημα, έχουν προταθεί συναρτήσεις που υπολογίζουν την απόσταση παρατηρήσεων που αποτελούνται από ονομαστικές τιμές. Στην απλούστερη εκδοχή τους οι συναρτήσεις αυτές επιστρέφουν την τιμή 0 εάν οι τιμές του ίδιου ονομαστικού γνωρίσματος δύο διαφορετικών παρατηρήσεων είναι ίδιες, αλλιώς επιστρέφουν την τιμή 1. Επίσης, έχουν προταθεί αλγόριθμοι που υπολογίζουν την απόσταση αντικειμένων που αποτελούνται και από αριθμητικές και από ονομαστικές τιμές [58].

Όπως αναφέρθηκε παραπάνω ο κατηγοριοποιητής k-NN διαθέτει το πλεονέκτημα ότι μπορεί εύκολα να υλοποιηθεί. Απαιτείται μόνο μία ακέραια τιμή για την παράμετρο k, ένα σύνολο εκπαίδευσης και ένα μέτρο απόστασης. Επιπλέον, ο κατηγοριοποιητής k-NN είναι ανθεκτικός στα θορυβώδη δεδομένα εκπαίδευσης και μπορεί σε πολλές περιπτώσεις να επιτυγχάνει υψηλές επιδόσεις κατηγοριοποίησης εάν τα δεδομένα εκπαίδευσης είναι πολύ μεγάλα [5]. Ωστόσο παρουσιάζει ορισμένα μειονεκτήματα όπως είναι ότι στερείται ενός βασικού τρόπου επιλογής της παραμέτρου k. Ο καθορισμός της τιμής του k πρέπει πάντα να πραγματοποιείται από την αρχή της κατηγοριοποίησης. Επίσης το υπολογιστικό κόστος είναι πολύ υψηλό έως απαγορευτικό κατά τη διάρκεια της πρόβλεψης. Για κάθε νέο στιγμιότυπο που πρέπει να κατηγοριοποιηθεί ή να προβλεφθεί, ο αλγόριθμος των k-Εγγύτερων Γειτόνων πρέπει να υπολογίσει όλες τις αποστάσεις προς όλα τα σημεία δεδομένων που έχουν αποθηκευτεί στο σύνολο εκπαίδευσης. Αυτό μπορεί να γίνει ασύμφορο για την ολοκλήρωση της διαδικασίας [44].

Ο τρόπος λειτουργίας του κατηγοριοποιητή k-NN θα μπορούσε να χαρακτηριστεί μια ακόμη αδυναμία γιατί υποδηλώνει υψηλές απαιτήσεις αποθήκευσης για τα δεδομένα

εκπαίδευσης. Η βάση δεδομένων εκπαίδευσης πρέπει να είναι πάντα διαθέσιμη, με αποτέλεσμα να δεσμεύεται μόνιμα μεγάλος αποθηκευτικός χώρος. Κατά συνέπεια για να εκτελεστεί αποτελεσματικά ο κατηγοριοποιητής k-NN, απαιτούνται υπολογιστικά συστήματα που διαθέτουν μεγάλη κύρια μνήμη, ώστε να μπορούν να αποθηκεύονται τα δεδομένα εκπαίδευσης.

Μια τελευταία αδυναμία του κατηγοριοποιητή k-NN είναι η υψηλή ευαισθησία στον θόρυβο και τα λανθασμένα δεδομένα. Εάν τα δεδομένα περιέχουν θόρυβο ή αν υπάρχουν λανθασμένα δεδομένα στο σύνολο δεδομένων εκπαίδευσης, ο αλγόριθμος μπορεί να προκαλέσει λανθασμένες κατηγοριοποιήσεις. Επιπλέον, ο κατηγοριοποιητής k-NN είναι ευαίσθητος στις αποκλίσεις και τις αλληλοεπικαλύψεις μεταξύ περιοχών δεδομένων που ανήκουν σε διαφορετικές κατηγορίες. Αυτές οι αδυναμίες μπορούν να οδηγήσουν σε εσφαλμένες κατηγοριοποιήσεις και να προκαλέσουν μείωση της ακρίβειας του αλγορίθμου. Συνεπώς, η προ-επεξεργασία των δεδομένων και η επιλογή σωστής τιμής της παραμέτρου k, μπορούν να βελτιώσουν σημαντικά την ακρίβεια και την αξιοπιστία των κατηγοριοποιήσεων που πραγματοποιεί ο κατηγοριοποιητής k-NN [46].

Συνοψίζοντας, η μέθοδος των k-Εγγύτερων Γειτόνων αποτελεί μία απλή και εύκολη, ως προς την κατανόηση εφαρμογή. Δεδομένου ότι αυτή η μέθοδος είναι μη-παραμετρική μέθοδος κατηγοριοποίησης, αυτό σημαίνει πως δεν χρειάζεται να πληρούνται υποθέσεις για να εφαρμοστεί [47]. Ακόμα, ο κατηγοριοποιητής k-NN μπορεί να εφαρμοστεί εύκολα σε προβλήματα πολλών κλάσεων. Παρόλο που έχει τις περιορισμένες του αδυναμίες, όπως την υπολογιστική πολυπλοκότητα, τις υψηλές απαιτήσεις αποθήκευσης [39] και την υψηλή ευαισθησία στον θόρυβο και τα λανθασμένα δεδομένα [46], μπορεί να είναι χρήσιμος σε απλά προβλήματα κατηγοριοποίησης με λίγες διαστάσεις. Η επιλογή της κατάλληλης παραμέτρου k και η προ-επεξεργασία των δεδομένων είναι σημαντικοί παράγοντες για τη βελτιστοποίηση της απόδοσης του αλγορίθμου [44].

1.4 Τεχνικές μείωσης του πληθυσμού των δεδομένων (Data Reduction Techniques)

Είναι γεγονός ότι ο κατηγοριοποιητής k-NN είναι μια από τις πιο χρησιμοποιούμενες και γνωστές τεχνικές για την εκτέλεση εργασιών αναγνώρισης. Έχει αποδειχθεί επίσης ότι είναι ένας από τους πιο χρήσιμους αλγόριθμους στην εξόρυξη δεδομένων παρά την απλότητα του. Όπως κάθε κατηγοριοποιητής έχει τα πλεονεκτήματα και τα μειονεκτήματα του, έτσι λοιπόν και ο κατηγοριοποιητής k-NN έχει αρκετά μειονεκτήματα. Ένα βασικό μειονέκτημα θα μπορούσαμε να αναφέρουμε ότι είναι το υψηλό υπολογιστικό κόστος που απαιτεί για όλες

τις αποστάσεις και πως αυτό μπορεί να μειωθεί. Τα αδύνατα σημεία του κατηγοριοποιητή k-NN έχουν αποτελέσει αντικείμενο μελέτης πολλών ερευνητών και έχουν προταθεί πολλές λύσεις. Μία από τις πιο υποσχόμενες λύσεις είναι η ανάπτυξη τεχνικών μείωσης του πληθυσμού των δεδομένων (Data Reduction Techniques – DRTs).

Οι τεχνικές μείωσης του πληθυσμού των δεδομένων στον τομέα της εξόρυξης δεδομένων αναφέρονται σε μια σειρά από μεθόδους και πρακτικές, οι οποίες οδηγούν σε πιο ευέλικτη παρουσίαση των δεδομένων, καθώς μειώνεται κατά πολύ ο όγκος, διατηρώντας ταυτόχρονα την ακεραιότητα των αρχικών δεδομένων. Καθώς οι πηγές δεδομένων γίνονται όλο και πιο μεγάλες και πολύπλοκες, η ανάγκη για αποτελεσματικές τεχνικές διαχείρισης και ανάλυσης των δεδομένων αυξάνεται. Οι τεχνικές μείωσης του πληθυσμού των δεδομένων αποτελούν σημαντικό εργαλείο για τη διαχείριση αυτής της πρόκλησης [59].

Οι τεχνικές μείωσης του πληθυσμού των δεδομένων μπορούν να χωριστούν σε δύο μεγάλες κατηγορίες: α) **αλγόριθμοι επιλογής προτύπων** (prototype selection algorithms – PS) και β) **αλγόριθμοι παραγωγής προτύπων** (prototype generation algorithms – PG) [60].

Στην πρώτη κατηγορία, οι αλγόριθμοι επιλογής προτύπων (PS algorithms) επιλέγουν κάποια αντιπροσωπευτικά στιγμιότυπα ή αλλιώς συγκεκριμένα πρότυπα, από το σύνολο δεδομένων για να αποτελέσουν ένα νέο μειωμένο σύνολο. Ο στόχος είναι να μειωθεί η πολυπλοκότητα και ο υπολογιστικός φόρτος που απαιτούνται για την εκπαίδευση ενός μοντέλου, διατηρώντας παράλληλα την αποτελεσματικότητα του. Αυτοί οι αλγόριθμοι εξετάζουν διάφορες μετρικές και κριτήρια για να αποφασίσουν ποια πρότυπα θα διατηρηθούν και ποια θα απορριφθούν [59].

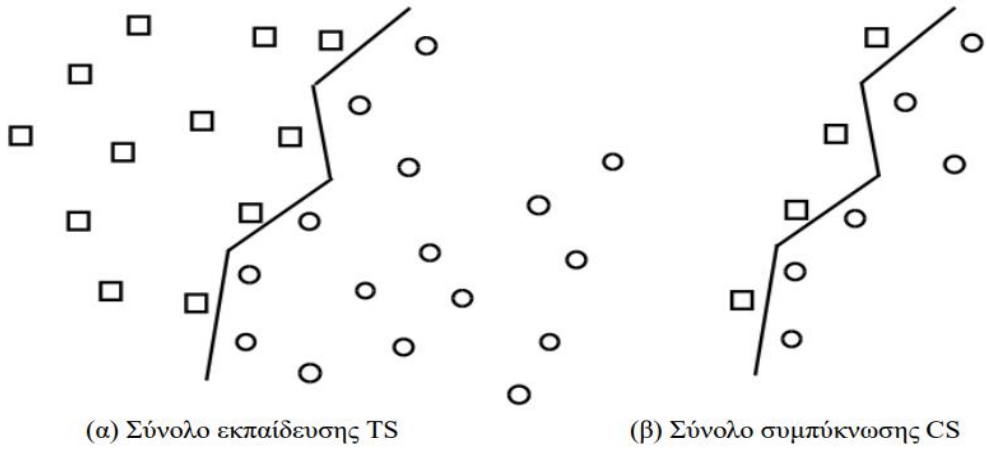
Στην δεύτερη κατηγορία, οι αλγόριθμοι παραγωγής προτύπων (PG algorithms) δημιουργούν καινούργια στιγμιότυπα συνενώνοντας παρόμοια στιγμιότυπα εκπαίδευσης και έτσι δημιουργούν ένα νέο μειωμένο σύνολο. Συνεπώς, τα παραγόμενα πρότυπα μπορεί να είναι συνδυασμοί υπαρχόντων προτύπων ή να δημιουργούνται από κάποιο μοντέλο. Στην πραγματικότητα κάθε πρότυπο αντιπροσωπεύει μία συγκεκριμένη περιοχή δεδομένων του πολυδιάστατου χώρου [60].

Γίνεται αντιληπτό λοιπόν, ότι και οι δύο κατηγορίες μοιράζονται το ίδιο κίνητρο. Έχουν στόχο τη δημιουργία ενός μικρού αντιπροσωπευτικού συνόλου δεδομένων, αλλά διαφέρουν στην μεθοδολογία που χρησιμοποιούν για να το κατασκευάσουν.

Οι περισσότερες τεχνικές μείωσης του πληθυσμού των δεδομένων προσπαθούν να εντοπίσουν τα σύνορα μεταξύ των διαφορετικών κλάσεων στα δεδομένα. Στο συμπυκνωμένο σύνολο τοποθετούνται περισσότερα αντικείμενα από τις περιοχές που βρίσκονται κοντά σε σύνορα των κλάσεων και όχι από τις εσωτερικές περιοχές αυτών. Η λειτουργία αυτής της διαδικασίας βασίζεται στην απλή ιδέα ότι τα δεδομένα που δεν ορίζουν τα όρια απόφασης, δεν επηρεάζουν την πρόβλεψη κατηγοριοποίησης και έτσι μπορούν να απομακρυνθούν.

Όπως γίνεται εύκολα αντιληπτό, μόνο τα στιγμιότυπα που ορίζουν τα όρια απόφασης επηρεάζουν το αποτέλεσμα της κατηγοριοποίησης και έτσι πρέπει να τοποθετηθούν στο συμπυκνωμένο σύνολο [61].

Ας υποθέσουμε ότι έχουμε ένα σύνολο δεδομένων εκπαίδευσης με δύο κλάσεις. Ένα παράδειγμα φαίνεται στην Εικόνα 2. Συγκεκριμένα, έχουμε τα στιγμιότυπα που ανήκουν στην κλάση «τετράγωνο» και αυτά που ανήκουν στην κλάση «κύκλος». Στην αριστερή πλευρά της εικόνας παρουσιάζεται το αρχικό σύνολο (training set), ενώ στα δεξιά παρουσιάζεται το συμπυκνωμένο σύνολο (condensing set) που προέκυψε μετά την εφαρμογή μιας τεχνικής μείωσης του πληθυσμού των δεδομένων. Ένα νέο, μη κατηγοριοποιημένο αντικείμενο θα κατηγοριοποιηθεί στην κλάση «τετράγωνο» αν βρεθεί από την αριστερή πλευρά του συνόρου απόφασης. Αυτή η αντιστοίχιση σε κλάση πραγματοποιείται είτε χρησιμοποιηθεί το αρχικό είτε το συμπυκνωμένο σύνολο δεδομένων. Αντίθετα, αν βρεθεί στην δεξιά πλευρά θα κατηγοριοποιηθεί στην κλάση «κύκλος». Με αυτόν τον τρόπο, όπως φαίνεται, επιτυγχάνεται μείωση του πληθυσμού των δεδομένων χωρίς να ελαττώνεται σημαντικά η ακρίβεια στην κατηγοριοποίηση [62].



Εικόνα 2: Στιγμιότυπα του συνόλου εκπαίδευσης και στιγμιότυπα στα όρια των κλάσεων [62]

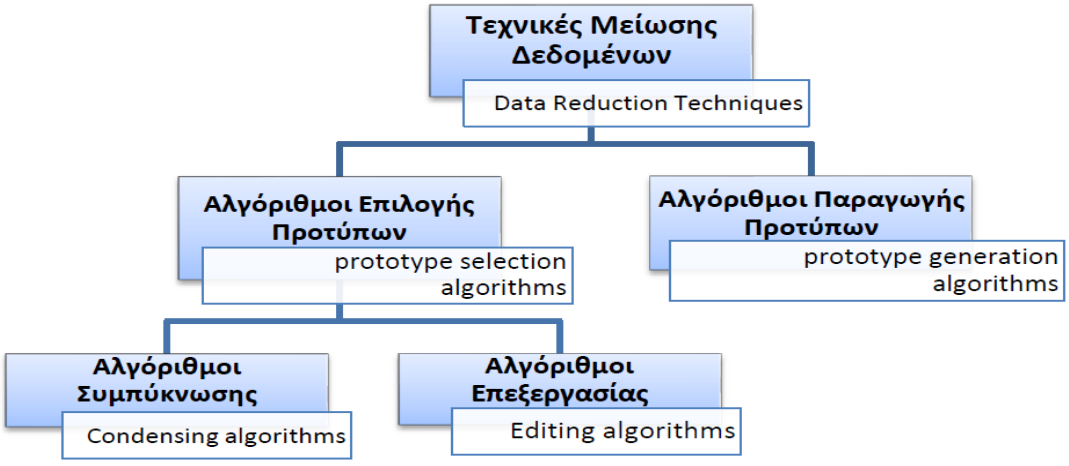
Οι αλγόριθμοι επιλογής προτύπων διασπώνται σε δύο υποκατηγορίες αλγορίθμων. Η πρώτη υποκατηγορία είναι οι **αλγόριθμοι συμπύκνωσης** (condensing algorithms) και η δεύτερη υποκατηγορία είναι οι **αλγόριθμοι επεξεργασίας** (editing algorithms) [59].

Οι αλγόριθμοι συμπύκνωσης έχουν στόχο να δημιουργήσουν ένα μικρό αντιπροσωπευτικό σύνολο δεδομένων από το αρχικό σύνολο εκπαίδευσης, το οποίο ονομάζεται σύνολο συμπύκνωσης (condensing set). Με αυτή την τεχνική επιτυγχάνεται αρχικά χαμηλότερο υπολογιστικό κόστος, ενώ ταυτόχρονα δεν υπάρχουν μεγάλες απαιτήσεις

αποθήκευσης και επίσης ταυτόχρονα διατηρείται η ακρίβεια της κατηγοριοποίησης σε υψηλά επίπεδα.

Οι αλγόριθμοι επεξεργασίας, σε αντίθεση με τους αλγόριθμους συμπύκνωσης, έχουν σαν βασικό στόχο την βελτίωση της ακρίβειας κατηγοριοποίησης παρά την επίτευξη υψηλών ποσοστών μείωσης του κόστους. Οι αλγόριθμοι που ανήκουν σε αυτή την υποκατηγορία προσπαθούν να βελτιώσουν την ποιότητα των δεδομένων εκπαίδευσης με διάφορες τεχνικές, όπως την αφαίρεση του θορύβου [63], την αφαίρεση των ακραίων τιμών και την εξομάλυνση των ορίων απόφασης μεταξύ των κλάσεων [59]. Επομένως, η αποδοτική εφαρμογή μιας τεχνικής μείωσης του πληθυσμού των δεδομένων προϋποθέτει την απομάκρυνση θορύβου από τα δεδομένα από έναν αλγόριθμό επεξεργασίας. Σαν τελικό αποτέλεσμα, αυτός ο αλγόριθμος δημιουργεί ένα επεξεργασμένο σύνολο εκπαίδευσης χωρίς επικαλύψεις μεταξύ των κλάσεων [64]. Στις περισσότερες περιπτώσεις είναι θεμιτή η χρήση και των δύο τεχνικών (Condensation και Editing) για να συνεχιστεί ορθότερα η Διαδικασία Ανακάλυψης Γνώσης, ωστόσο υπάρχει πάντα ο κίνδυνος να χαθεί πιθανώς χρήσιμη πληροφορία. Η χρήση των τεχνικών μείωσης του πληθυσμού των δεδομένων πρέπει να γίνεται βάσει των προτεραιοτήτων που θέτει ο ερευνητής. Είναι σημαντικό να αναφερθεί, ότι το στάδιο της προ-επεξεργασίας των δεδομένων όπου πραγματοποιείται και η μείωση του πληθυσμού τους, αναφέρεται και ως Καθαρισμός των Δεδομένων [65].

Οι **υβριδικοί αλγόριθμοι** όπως υποδηλώνει και το όνομα βρίσκονται μεταξύ των δύο ανωτέρω τεχνικών και ο στόχος τους είναι να βρούνε το μικρότερο και ταυτόχρονα το πιο ακριβές υποσύνολο των δεδομένων. Διαγράφουν κεντρικά αλλά και συνοριακά στιγμιότυπα [59]. Στην Εικόνα 3 που ακολουθεί παρουσιάζονται οι κατηγορίες που αναφέρθηκαν παραπάνω σε ιεραρχική σχεδίαση.



Εικόνα 3: Κατηγορίες τεχνικών μείωσης του πληθυσμού των δεδομένων

Μια άλλη κατηγορία τεχνικών για τη μείωση του πληθυσμού των δεδομένων που εμφανίζεται στην βιβλιογραφία είναι οι **επαυξητικοί αλγόριθμοι** (incremental algorithms) και οι **μη επαυξητικοί αλγόριθμοι** (decremental algorithms). Οι επαυξητικοί αλγόριθμοι ξεκινούν έχοντας ένα μικρό υποσύνολο των δεδομένων, που σταδιακά το αυξάνουν μέχρι να ικανοποιήσουν κάποια κριτήρια (π.χ. ποιότητα των δεδομένων) [66]. Αντίθετα, οι μη επαυξητικοί αλγόριθμοι αρχικά λαμβάνουν υπόψη όλα τα δεδομένα εκπαίδευσης και σταδιακά τα μειώνουν μέχρι να ικανοποιηθούν συγκεκριμένα κριτήρια [67].

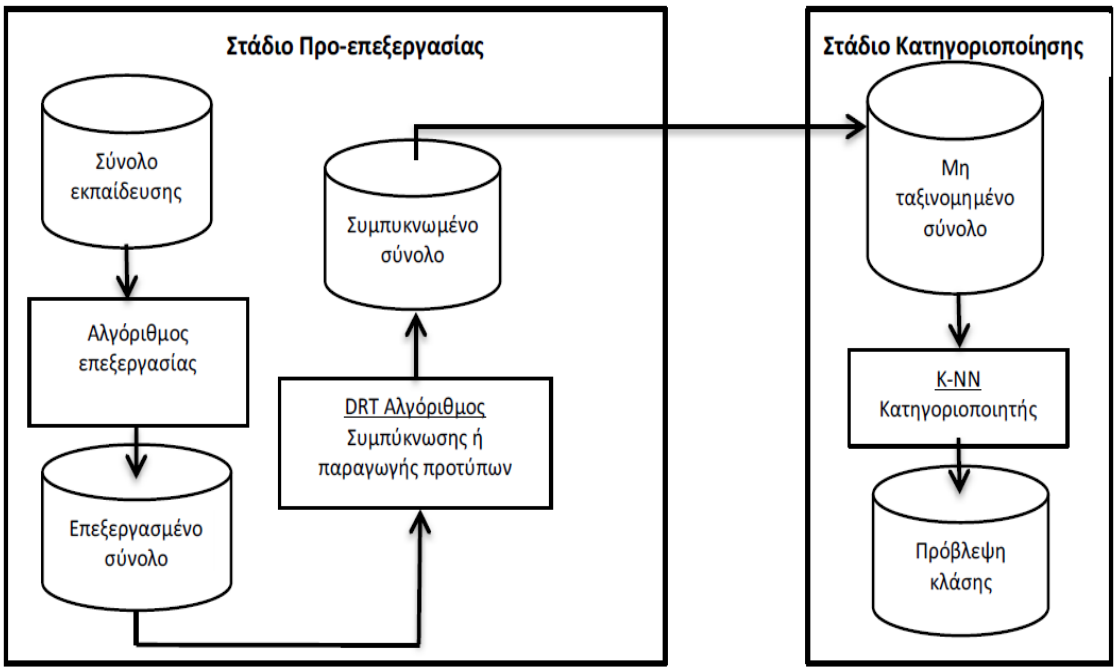
Οι τεχνικές μείωσης του πληθυσμού των δεδομένων μπορούν να συγκριθούν μεταξύ τους χρησιμοποιώντας τρία κριτήρια αξιολόγησης.

Το πρώτο είναι η ακρίβεια κατηγοριοποίησης που επιτυγχάνεται από τον κατηγοριοποιητή k-NN όταν εκτελείται στο σύνολο συμπύκνωσης.

Το δεύτερο είναι το ποσοστό μείωσης (reduction rate) του πληθυσμού των δεδομένων που επιτυγχάνεται μετά την εφαρμογή κάθε αλγορίθμου. Όπως γίνεται αντιληπτό, υψηλότερα ποσοστά μείωσης του πληθυσμού των δεδομένων είναι προτιμότερο να προκύπτουν από κάποια τεχνική, καθώς έτσι μειώνεται το υπολογιστικό κόστος επεξεργασίας και οι απαιτήσεις σε μνήμη, οπότε αυξάνεται η ταχύτητα εκτέλεσης του κατηγοριοποιητή k-Εγγύτερων Γειτόνων που θα ακολουθήσει.

Το τρίτο είναι το υπολογιστικό κόστος προ-επεξεργασίας δεδομένων που απαιτεί η εκτέλεση κάθε αλγορίθμου. Κάποιοι αλγόριθμοι απαιτούν μόνο "ένα πέρασμα" των δεδομένων (one-pass algorithms), γεγονός που τους καθιστά προτιμότερους βάσει αυτού του κριτηρίου, ενώ άλλοι με χρήση πολλών επαναλήψεων υστερούν πολύ σε αυτό το κομμάτι.

Στην Εικόνα 4 που ακολουθεί παρουσιάζεται η διαδικασία που απαιτείται για το στάδιο της προ-επεξεργασίας κάνοντας χρήση κάποιας τεχνικής μείωσης του πληθυσμού των δεδομένων και η διαδικασία που απαιτείται για το στάδιο της κατηγοριοποίησης κάνοντας χρήση του k-NN κατηγοριοποιητή [62].



Εικόνα 4: Διαδικασία κατηγοριοποίησης μέσω της μείωσης του πληθυσμού των δεδομένων [62]

Ο στόχος των τεχνικών μείωσης του πληθυσμού των δεδομένων (DRTs) είναι να δημιουργήσουν ένα σύνολο συμπύκνωσης χωρίς θόρυβο και να κρατήσουν ή να παράγουν για κάθε κλάση επαρκή πρότυπα για να κάνουν τη δουλειά του k-NN κατηγοριοποιητή πιο εύκολη και πιο γρήγορη [46].

Ορισμένες τεχνικές μείωσης του πληθυσμού των δεδομένων μπορούν να επιτύχουν καλή απόδοση, αλλά εμφανίζουν μία ή περισσότερες από τις ακόλουθες αδυναμίες [46]:

1. Συνήθως περιλαμβάνουν μια δαπανηρή και χρονοβόρα προ-επεξεργασία στο αρχικό σύνολο, το οποίο μπορεί να είναι απαγορευτικό για μεγάλα σύνολα δεδομένων.
2. Πολλές τεχνικές μείωσης του πληθυσμού των δεδομένων είναι παραμετρικές. Ο χρήστης πρέπει να παρέχει τις τιμές για έναν αριθμό παραμέτρων εκ των προτέρων. Αυτό συνήθως περιλαμβάνει συντονισμό μέσω μιας επαναληπτικής εκτέλεσης μιας διαδικασίας δοκιμής και σφάλματος.
3. Το τελικό συμπυκνωμένο σύνολο πολλών τεχνικών μείωσης του πληθυσμού των δεδομένων εξαρτάται από τη σειρά των αντικειμένων στο αρχικό σύνολο. Αυτό σημαίνει ότι ορισμένες τεχνικές μείωσης του πληθυσμού των δεδομένων μπορεί να δημιουργήσουν ένα διαφορετικό συμπυκνωμένο σύνολο όταν επεξεργάζονται τα στοιχεία ενός συγκεκριμένου αρχικού συνόλου με διαφορετική σειρά.
4. Συνήθως, υπάρχει μια αντιστάθμιση μεταξύ της μείωσης των δεδομένων και της ακρίβειας της κατηγοριοποίησης. Αν και ορισμένες τεχνικές μείωσης του πληθυσμού των δεδομένων μπορούν να επιτύχουν υψηλούς ρυθμούς μείωσης, η ακρίβεια του

κατηγοριοποιητή επηρεάζεται αρνητικά. Από την άλλη πλευρά, υπάρχουν τεχνικές μείωσης του πληθυσμού των δεδομένων που παράγουν συμπυκνωμένο σύνολο που επιτυγχάνουν ακρίβεια κοντά σε αυτά που επιτυγχάνονται από τα μη μειωμένα αρχικά σύνολα, αλλά τα ποσοστά μείωσης τους δεν είναι υψηλά.

5. Οι περισσότερες τεχνικές μείωσης του πληθυσμού των δεδομένων βασίζονται στη μνήμη. Αυτό σημαίνει ότι ολόκληρο το αρχικό σύνολο πρέπει να βρίσκεται στην κύρια μνήμη. Επομένως, είναι ακατάλληλες τεχνικές για πολύ μεγάλα σύνολα δεδομένων που δεν μπορούν να χωρέσουν στην κύρια μνήμη ή για συσκευές με περιορισμένη κύρια μνήμη.
6. Οι περισσότερες τεχνικές μείωσης του πληθυσμού των δεδομένων δεν μπορούν να εξετάσουν νέα αντικείμενα εκπαίδευσης μετά την κατασκευή του συμπυκνωμένου συνόλου. Αυτές οι τεχνικές μείωσης του πληθυσμού των δεδομένων είναι ακατάλληλες για δυναμικά περιβάλλοντα/συνεχούς ροής όπου σταδιακά είναι διαθέσιμα νέα αντικείμενα εκπαίδευσης.

Συνοψίζοντας, η μείωση δεδομένων είναι μια κρίσιμη εργασία στον τομέα της εξόρυξης δεδομένων. Οι τεχνικές μείωσης του πληθυσμού των δεδομένων (Data Reduction Techniques – DRTs) συνεισφέρουν σημαντικά τόσο στην μείωση του υπολογιστικού κόστους αναζήτησης των εγγύτερων γειτόνων όσο και στην μείωση των απαιτήσεων αποθήκευσης [46]. Ειδικά στην περίπτωση του κατηγοριοποιητή k-NN, η εφαρμογή ορισμένων τεχνικών μείωσης του πληθυσμού των δεδομένων κρίνεται απαραίτητη, διότι μπορεί να οδηγήσει σε βελτιωμένα αποτελέσματα κατηγοριοποίησης, χωρίς ωστόσο να θυσιάζεται η σημαντική πληροφορία που περιέχεται στα δεδομένα [59].

1.5 Κίνητρο και συνεισφορά

Η κατηγοριοποίηση με βάση τους k-Εγγύτερους Γείτονες (k-Nearest Neighbors) είναι μια από τις πιο θεμελιώδεις και απλές μεθόδους κατηγοριοποίησης και θα πρέπει να είναι μία από τις πρώτες επιλογές για μια μελέτη κατηγοριοποίησης όταν υπάρχει λίγη ή καθόλου προηγούμενη γνώση σχετικά με τη διανομή των δεδομένων. Ένας αποτελεσματικός και ευρέως χρησιμοποιούμενος οκνηρός αλγόριθμος κατηγοριοποίησης είναι ο k-NN. Ανήκει στον τομέα της επιβλεπόμενης μάθησης και βρίσκει έντονη εφαρμογή στην αναγνώριση προτύπων και στην εξόρυξη δεδομένων.

Το αρνητικό του σημείο όμως είναι πως προϋποθέτει ότι τα σύνολα δεδομένων, τα οποία χειρίζεται, πρέπει να μην περιλαμβάνουν κατηγορικά χαρακτηριστικά. Όταν όμως τα σύνολα δεδομένων περιέχουν κατηγορικά χαρακτηριστικά, που σχεδόν πάντα εμφανίζονται στα πραγματικά σύνολα δεδομένων εκπαίδευσης, τότε ο αλγόριθμος κατηγοριοποίησης k-

ΝΝ αντιμετωπίζει δυσκολίες στον υπολογισμό των αποστάσεων μεταξύ των στιγμιότυπων, καθιστώντας τη χρήση του απαγορευτική για αυτόν τον τύπο δεδομένων. Είναι σημαντικό να αναφερθεί, ότι και η εφαρμογή των τεχνικών μείωσης του πληθυσμού των δεδομένων προϋποθέτει την απουσία κατηγορικών χαρακτηριστικών στο σύνολο δεδομένων. Πριν από την επεξεργασία από μια τεχνική μείωσης του πληθυσμού των δεδομένων, είναι απαραίτητη η εφαρμογή ενός ακόμη βήματος προ-επεξεργασίας για την μετατροπή των κατηγορικών δεδομένων σε αριθμητικά δεδομένα με τη χρήση του μετασχηματισμού των δεδομένων και συγκεκριμένα με την τεχνική της κατηγορικής κωδικοποίησης. Εντούτοις, η εφαρμογή ενός ακόμη βήματος προ-επεξεργασίας είναι ένα αρνητικό σημείο, επειδή προσθέτει υπολογιστικό κόστος. Αυτό το σημείο αποτελεί το κίνητρο για την εκπόνηση της παρούσας διπλωματικής εργασίας, δηλαδή πως μπορούμε να αντιμετωπίσουμε τα κατηγορικά δεδομένα ώστε να επιτυγχάνουμε παράλληλα καλύτερες επιδόσεις του κατηγοριοποιητή και πως θα μπορούσε μία τεχνική μείωσης του πληθυσμού των δεδομένων να μειώσει το απαιτούμενο υπολογιστικό κόστος που προσθέτουν οι τεχνικές μετασχηματισμού των δεδομένων.

Η παρούσα διπλωματική εργασία προτείνει μια νέα παραλλαγή μιας τεχνικής μείωσης του πληθυσμού των δεδομένων που μπορεί να διαχειριστεί τα κατηγορικά δεδομένα χωρίς να απαιτείται το επιπρόσθετο βήμα προ-επεξεργασίας για την μετατροπή τους σε αριθμητικά δεδομένα. Η τεχνική που προτείνεται είναι ένας αλγόριθμος επιλογής προτύπων (Prototype Selection) και ονομάζεται Κανόνας Συμπύκνωσης Εγγύτερου Γείτονα (Condensed Nearest Neighbour rule – CNN-rule). Συγκεκριμένα, οι νέες παραλλαγές του CNN-rule, τις οποίες ονομάζουμε CNN-rule Hamming Distance, CNN-rule Levenshtein Distance και CNN-rule Jaccard Distance, αξιοποιούν τις τεχνικές των μετρικών αποστάσεων (metric distances) για μη μετρικούς χώρους. Με αυτόν τον τρόπο αποφεύγεται η επιπλέον επιβάρυνση υπολογιστικού κόστους που θα απαιτούσε η μετατροπή των κατηγορικών δεδομένων σε αριθμητικά δεδομένα και συνεπώς επεξεργάζεται απευθείας τα σύνολα δεδομένων που περιέχουν κατηγορικά χαρακτηριστικά.

Η συνεισφορά μας με την εργασία αυτή συνίσταται στην ανάπτυξη μιας σημαντικής προσέγγισης στον χώρο της μηχανικής μάθησης βασισμένης σε στιγμιότυπα. Με την μελέτη των μετρικών απόστασης για μη μετρικούς χώρους και την εφαρμογή τεχνικών μείωσης του πληθυσμού των δεδομένων, προτείνουμε νέες παραλλαγές του αλγορίθμου CNN-rule που ανοίγουν νέους ορίζοντες στην κατηγοριοποίηση κατηγορικών δεδομένων.

Με πειραματικό έλεγχο δοκιμάζεται η λειτουργία και η απόδοση των αλγορίθμων CNN Hamming, CNN Levenshtein και CNN Jaccard στα σύνολα δεδομένων που περιλαμβάνουν κατηγορικά χαρακτηριστικά κάνοντας χρήση του υπολογισμού των τροποποιημένων συναρτήσεων απόστασης. Οι τρείς παραλλαγές του αλγορίθμου CNN-rule συγκρίνονται μεταξύ τους αλλά και με τον αλγόριθμο των k-Εγγύτερων Γειτόνων, χωρίς

μείωση του πληθυσμού των δεδομένων, εκτελώντας πειράματα σε 8 σύνολα δεδομένων και εκτιμώντας την ακρίβεια κατηγοριοποίησης (accuracy) αλλά και το ποσοστό μείωσης (reduction rate) που επιτυγχάνουν. Τα πειραματικά αποτελέσματα δείχνουν αξιοσημείωτη απόδοση και για τις τρεις παραλλαγές του αλγορίθμου CNN-rule.

1.6 Οργάνωση της διπλωματικής εργασίας

Αυτή η μεταπτυχιακή εργασία επικεντρώνεται σε δύο ερευνητικές διαστάσεις της εξόρυξης δεδομένων που αναφέρονται στην κατηγοριοποίηση. Η μία διάσταση είναι οι μετρικές απόστασης για μη μετρικούς χώρους και η άλλη διάσταση μελετά τις τεχνικές μείωσης του πληθυσμού των δεδομένων. Η εργασία συνεισφέρει νέες παραλλαγές μείωσης του πληθυσμού των δεδομένων, προτείνοντας βελτιώσεις σε μία υπάρχουσα τεχνική μείωσης του πληθυσμού των δεδομένων. Στο πρώτο κεφάλαιο της εργασίας συνοψίζουμε τα κύρια σημεία που χρειάζονται για να εξερευνήσουμε τον χώρο της κατηγοριοποίησης. Στα πλαίσια της μελέτης για την κατηγοριοποίηση με βάση τα στιγμιότυπα παρουσιάζεται αναλυτικά ο κατηγοριοποιητής k-NN, ο οποίος μας έδωσε το κίνητρο για να δημιουργήσουμε και να συνεισφέρουμε στην υλοποίηση της προτεινόμενης βελτίωσης τεχνικής μείωσης προτύπων (συμπύκνωσης) καθώς και στην πειραματική μελέτη της. Κλείνοντας το κεφάλαιο, παρουσιάζουμε την οργάνωση της εργασίας μας.

Στο κεφάλαιο 2 γίνεται μία μελέτη ανασκόπησης για τους μετρικούς και μη μετρικούς χώρους, οι οποίοι είναι έννοιες που αναφέρονται στην μαθηματική ανάλυση και τη θεωρία των τοπολογικών χώρων. Εν συνεχεία, εξετάζεται η δυνατότητα προσέγγισης της έννοιας της μετρικής απόστασης σε χώρους που δεν είναι μετρικοί. Επιπρόσθετα, γίνεται ανάλυση των μετρικών απόστασης Hamming, Levenshtein και Jaccard, οι οποίες είναι βασικά δομικά στοιχεία για την τεχνική μείωσης του πληθυσμού των δεδομένων που εστιάζει η παρούσα εργασία και κατά συνέπεια για τις νέες προτεινόμενες παραλλαγές του αλγορίθμου.

Στο κεφάλαιο 3 γίνεται αναφορά στον κανόνα συμπύκνωσης εγγύτερου γείτονα (CNN-rule). Ο αλγόριθμος CNN-rule είναι μια τεχνική μείωσης του πληθυσμού των δεδομένων που βασίζεται στην έννοια των εγγύτερων γειτόνων για την κατηγοριοποίηση δεδομένων. Τα δεδομένα εκπαίδευσης, δηλαδή τα δείγματα που χρησιμοποιούνται ως Σύνολο Εκπαίδευσης (Training Set), αποτελούν τη βάση πάνω στην οποία βασίζεται ο αλγόριθμος CNN-rule. Αυτά τα δεδομένα εκπαίδευσης χρησιμοποιούνται για την εκπαίδευση του αλγορίθμου και για τον καθορισμό των συνόρων απόφασης. Ο αλγόριθμος CNN-rule είναι ένας αποδοτικός αλγόριθμος επιλογής προτύπων, ο οποίος έχει σχετικά χαμηλό κόστος προ-επεξεργασίας. Παράγοντας προβληματισμού αποτελεί το γεγονός ότι αυτός ο αλγόριθμος δεν

είναι κατάλληλος για σύνολα δεδομένων που περιέχουν κατηγορικά χαρακτηριστικά. Για την επίλυση του συγκεκριμένου προβλήματος, προτείνουμε αλγορίθμους χρησιμοποιώντας τροποποιημένες συναρτήσεις απόστασης. Οι αλγόριθμοι CNN Hamming, CNN Levenshtein και CNN Jaccard, είναι παραλλαγές του αλγορίθμου CNN-rule. Επιπρόσθετα, προτείνουμε παραλλαγές του κατηγοριοποιητή k-NN χρησιμοποιώντας τροποποιημένες μετρικές για την αποτελεσματική κατηγοριοποίηση των κατηγορικών δεδομένων. Αυτές οι παραλλαγές είναι οι κατηγοριοποιητές k-NN Hamming, k-NN Levenshtein και k-NN Jaccard.

Στο κεφάλαιο 4 παρουσιάζονται όλες οι ρυθμίσεις και οι παράμετροι που χρησιμοποιήθηκαν και πως αξιοποιήθηκαν για την πειραματική μελέτη που έγινε πάνω σε 8 γνωστά σύνολα δεδομένων. Εν συνεχεία, παρουσιάζονται και αναλύονται τα πειραματικά αποτελέσματα που προέκυψαν από τις μετρήσεις των προτεινόμενων αλγορίθμων όσον αφορά την ακρίβεια κατηγοριοποίησης αλλά και το ποσοστό μείωσης του πληθυσμού των δεδομένων που επιτυγχάνουν.

Τέλος, στο κεφάλαιο 5 παρουσιάζονται τα συμπεράσματα και γίνονται προτάσεις για μελλοντική εργασία.

2

Μετρικές απόστασης για μη μετρικούς χώρους

Οι μη μετρικοί χώροι αποτελούν ένα σημαντικό πεδίο στην επιστήμη της μηχανικής μάθησης και της εξόρυξης δεδομένων. Το πρόβλημα της κατηγοριοποίησης σε μη μετρικούς χώρους είναι ένα ερευνητικό πρόβλημα, το οποίο οι ερευνητές συχνά καλούνται να επιλύσουν. Στο κεφάλαιο αυτό γίνεται αναφορά στο πρόβλημα της έλλειψης μετρικών αποστάσεων στους μη μετρικούς χώρους. Παρουσιάζεται μία μελέτη ανασκόπησης για τους μετρικούς και μη μετρικούς χώρους. Εν συνεχεία, εξετάζεται η ύπαρξη μεθόδων που μπορούν να προσεγγίσουν την έννοια της μετρικής απόστασης σε μη μετρικούς χώρους. Τέλος, παρουσιάζονται οι μετρικές απόστασης Hamming, Levenshtein και Jaccard, οι οποίες είναι μερικές από τις πιο σημαντικές μετρικές απόστασης που χρησιμοποιήθηκαν στην παρούσα εργασία και παρουσιάζονται στο κεφάλαιο 3.

2.1 Μετρικοί και μη μετρικοί χώροι

Οι **μετρικοί** (metric) και **μη μετρικοί χώροι** (non-metric spaces) είναι έννοιες που αναφέρονται στην μαθηματική ανάλυση και τη θεωρία των τοπολογικών χώρων [68][69]. Η τοπολογία είναι ένα ισχυρό εργαλείο έρευνας και έκφρασης σε όλους τους κλάδους της μαθηματικής επιστήμης. Τα τελευταία μάλιστα χρόνια η τοπολογία χρησιμοποιείται όλο και περισσότερο στη δημιουργία μαθηματικών μοντέλων που εξυπηρετούν ερευνητικά εφαρμοσμένους κλάδους των θετικών επιστημών, όπως η οικονομία, η πληροφορική κ.τ.λ.

Η διεξοδική μελέτη των μετρικών χώρων συντελεί στην καλύτερη κατανόηση της δομής του Ευκλείδειου χώρου R^n. Η έννοια της απόλυτης τιμής στο σύνολο R των πραγματικών αριθμών και ειδικότερα, η δυνατότητα που μας δίνει η έννοια αυτή να μιλάμε για την απόσταση |x-y| δύο πραγματικών αριθμών x και y, παίζει θεμελιακό ρόλο στη μελέτη του συνόλου R. Η έννοια αυτής της απόστασης στο R μας επιτρέπει, εκτός των άλλων, να ορίσουμε σύγκλιση μέσα στο σύνολο των πραγματικών αριθμών, πάνω στην οποία στηρίζεται ουσιαστικά το οικοδόμημα του Απειροστικού Λογισμού (μαθηματική μελέτη της συνεχούς μεταβολής των τιμών).

Από την παραπάνω απλή παρατήρηση και μόνο, γίνεται φανερό το πόσο σκόπιμο είναι να εξεταστεί η δυνατότητα ορισμού μιας έννοιας αντίστοιχης με την απόσταση στο R, σε ένα τυχαίο σύνολο καθώς και οι συνέπειες ενός τέτοιου ορισμού. Αυτή ακριβώς είναι η σκοπιμότητα του ορισμού των εννοιών της "μετρικής" και του "μετρικού χώρου" που ακολουθούν [69].

Ορισμός. Έστω X ένα μη κενό σύνολο. **Μετρική** στο X λέγεται κάθε συνάρτηση $d : X \times X \to R$ που ικανοποίει τις παρακάτω ιδιότητες [70]:

- $d(x, y) \geq 0$ για κάθε $x, y \in X$ και $d(x, y) = 0$ αν και μόνο αν $x = y$ (θετική οριστικότητα).
- $d(x, y) = d(y, x)$ για κάθε $x, y \in X$ (συμμετρική ιδιότητα).
- $d(x, y) \leq d(x, z) + d(y, z)$ για κάθε $x, y, z \in X$ (τριγωνική ανισότητα).

Αν d είναι μια μετρική στο X, τότε το ζεύγος (X, d) καλείται **μετρικός χώρος**. Στην περίπτωση αυτή τα στοιχεία του X ονομάζονται και σημεία. Επίσης, την τιμή d(x, y) στο ζευγάρι (x, y) την ονομάζουμε **απόσταση** των x, y.

Πιο απλά λέμε ότι ο μετρικός χώρος είναι ένα μη κενό σύνολο X και ένας συγκεκριμένος τρόπος μέτρησης αποστάσεων ανάμεσα στα στοιχεία του. Επομένως, μετρικός χώρος είναι δύο πράγματα μαζί, ένα μη κενό σύνολο X και μια μετρική $d : X \times X \to R$. Αυτό σημαίνει πως όταν έχουμε ένα σύνολο X δεν μπορούμε να μιλάμε για μετρικό χώρο X παρά μόνο όταν είναι ήδη καθορισμένη και εννοείται από τα συμφραζόμενα μια συγκεκριμένη μετρική d στο σύνολο X. Στην ιδανική περίπτωση, η μετρική πρέπει να ελαχιστοποιεί την απόσταση μεταξύ δύο παρόμοιων κατηγοριοποιημένων στιγμιότυπων, ενώ μεγιστοποιεί την απόσταση μεταξύ των στιγμιότυπων διαφορετικών κλάσεων. Έχουν παρουσιαστεί πολλές διαφορετικές μετρικές απόστασης (metric distances) [44]. Σε περιπτώσεις πραγματικών ή ακέραιων χαρακτηριστικών, η Ευκλείδεια απόσταση (Euclidean distance) είναι η μετρική απόστασης που χρησιμοποιείται συνήθως σε μετρικούς χώρους. Ο διανυσματικός χώρος R^n εφοδιασμένος με μετρική απόστασης την εισαγόμενη από την Ευκλείδεια στάθμη | |, ονομάζεται Ευκλείδειος μετρικός χώρος (euclidean metric space) και συμβολίζεται με $(R^n, |\ |)$. Ωστόσο, μπορούν να υιοθετηθούν και άλλες μετρικές απόστασης π.χ. Mahalanobis,

Manhatan, Minkowski, Chebyshev [69][71]. Οι μετρικές απόστασης Hamming, Levenshtein (Edit) και Jaccard, είναι από τις πιο σημαντικές για τους μετρικούς χώρους.

Μια άλλη βασική κατηγορία χώρων στον κλάδο της μαθηματικής ανάλυσης είναι οι μη μετρικοί χώροι, προσφέροντας μια διαφορετική προσέγγιση για την κατανόηση των σχέσεων μεταξύ των σημείων ενός χώρου. Οι μη μετρικοί χώροι είναι η γενίκευση των μετρικών χώρων που δεν πληρούν την ιδιότητα της τριγωνικής ανισότητας [72]. Ο ορισμός του **μη μετρικού χώρου** δεν είναι καθορισμένος στην βιβλιογραφία με τον ίδιο τρόπο όπως στον μετρικό χώρο. Οι μη μετρικοί χώροι είναι ποικίλοι και διαφορετικοί μεταξύ τους, και αυτή η ποικιλομορφία καθιστά δύσκολη τη δημιουργία ενός σαφούς και γενικά αποδεκτού ορισμού.

Οι μετρικές απόστασης που χρησιμοποιούνται σε μετρικούς χώρους δεν είναι εφαρμόσιμες σε μη μετρικούς χώρους. Για αυτό πρέπει να χρησιμοποιούνται διάφορες άλλες συναρτήσεις απόστασης που να ταιριάζουν με το ανάλογο πλαίσιο του προβλήματος. Στους μη μετρικούς χώρους, κάθε συνάρτηση απόστασης που δεν ικανοποιεί ορισμένες από τις μετρικές ιδιότητες θεωρείται **μη μετρική απόσταση** (non-metric distance). Υπάρχουν διάφοροι λόγοι που δικαιολογούν τη χρήση μη μετρικών αποστάσεων, όπως η ευρωστία σε ακραίες τιμές, η δυσκολία πρόβλεψης της συμπεριφοράς των μοντέλων μηχανικής μάθησης και η ανάγκη για βελτιωμένη μοντελοποίηση της ανθρώπινης αντίληψης [73].

Παρουσιάζουμε διάφορες μεθόδους που μπορούν να προσεγγίσουν την έννοια της απόστασης σε μη μετρικούς χώρους, χωρίς να ικανοποιούν τις ιδιότητες που συνήθως απαιτούνται από μια μετρική, όπως η ιδιότητα της τριγωνικής ανισότητας. Αυτές οι μη μετρικές αποστάσεις μας επιτρέπουν να υπολογίζουμε την απόσταση μεταξύ σημείων και να αναπτύσσουμε αλγόριθμους που λειτουργούν αποτελεσματικά σε αυτούς τους χώρους.

Bregman divergence

Η **απόκλιση Bregman** (Bregman divergence), που όρισε ο μαθηματικός Lev M. Bregman το 1967, αποτελεί ένα παράδειγμα τυπικά μη μετρικής συνάρτησης απόστασης [74]. Στα μαθηματικά, συγκεκριμένα στη στατιστική και στη γεωμετρία πληροφοριών, η απόκλιση Bregman ή αλλιώς η απόσταση Bregman είναι ένα μέτρο της διαφοράς μεταξύ δύο σημείων, που ορίζεται από μια κυρτή συνάρτηση.

Οι αποκλίσεις Bregman, παρόμοιες με τις μετρικές των μετρικών χώρων, δεν ικανοποιούν ούτε τη συμμετρική ιδιότητα ούτε την τριγωνική ανισότητα. Ωστόσο, ικανοποιούν μια γενίκευση του Πυθαγόρειου θεωρήματος και στη γεωμετρία πληροφοριών η αντίστοιχη στατιστική πολλαπλότητα ερμηνεύεται ως μια (διπλή) επίπεδη πολλαπλότητα. Αυτό επιτρέπει σε πολλές τεχνικές της θεωρίας βελτιστοποίησης να γενικεύονται στις

αποκλίσεις Bregman, γεωμετρικά ως γενικεύσεις ελαχίστων τετραγώνων. Η απόκλιση Bregman $d_f(x, y)$ για τα δύο σημεία x και y ορίζεται ως εξής:

$$d_f(x, y) = f(x) - f(y) - (f(y) \cdot (x - y))$$

Οι αποκλίσεις Bregman έχουν ορισμένες ιδιότητες που μοιάζουν με αυτές της μετρικής απόστασης, αλλά ισχύουν και σε μη μετρικούς χώρους. Η απόκλιση πρέπει να είναι μη αρνητική, δηλαδή $d_f(x, y) \geq 0$ για κάθε ζευγάρι (x, y). Η απόκλιση θα πρέπει να είναι μηδέν μόνο όταν τα δύο σημεία x και y είναι τα ίδια, δηλαδή $d_f(x, y) = 0$ αν και μόνο αν x = y. Εάν δύο αποκλίσεις d_f και d_g είναι ίσες, δηλαδή $d_f(x, y) = d_g(x, y)$, τότε οι δύο αποκλίσεις είναι ισοδύναμες.

Στην εξίσωση, το f(x) είναι η τιμή της συνάρτησης f στο σημείο x και το f(y) είναι η τιμή της συνάρτησης f στο σημείο y. Το (x - y) είναι η διαφορά των δύο σημείων, δηλαδή ένα διάνυσμα που δείχνει από το y προς το x. Το f(y) · (x - y) είναι το εσωτερικό γινόμενο (scalar product) του διανύσματος (x - y) με τον οριακό ρυθμό μεταβολής (gradient) της συνάρτησης f στο σημείο y.

Η απόκλιση Bregman μετρά το "κόστος" της μετάβασης από το σημείο y στο σημείο x, λαμβάνοντας υπόψη την κυρτότητα της συνάρτησης f. Αυτό το κόστος μπορεί να είναι διαφορετικό αν μετακινηθούμε από το x στο y αντίστροφα, και αυτό καθιστά τη απόκλιση Bregman μη μετρική απόσταση. Ωστόσο, αυτή η διαφοροποιησιμότητα της απόκλισης Bregman την καθιστά χρήσιμη σε πολλές εφαρμογές, όπως η μηχανική μάθηση και η βελτιστοποίηση, όπου χρειάζεται να μετρήσουμε την απόσταση μεταξύ διαφόρων σημείων σε κυρτούς χώρους [75].

Οι αποκλίσεις Bregman περιλαμβάνουν την τετραγωνισμένη Ευκλείδεια απόσταση ή αλλιώς KL-απόκλιση (Kullbac-Leibler divergence), η οποία υπολογίζεται με τον μαθηματικό τύπο [76]:

$$d(x, y) = \sum x_i \log(x_i / y_i)$$

και την απόσταση Itakura-Saito:

$$d(x, y) = \sum x_i / y_i - \log(x_i / y_i) - 1$$

Cosine distance

Η **Απόσταση Συνημιτόνου** (Cosine Distance) είναι μια μη μετρική απόστασης που χρησιμοποιείται συχνά στην ανάλυση και τον υπολογισμό της ομοιότητας μεταξύ δύο διανυσμάτων σε πολυδιάστατους χώρους. Συνήθως, χρησιμοποιείται σε περιπτώσεις όπου τα διανύσματα αναπαριστούν τα χαρακτηριστικά ή τα γνωρίσματα δύο αντικειμένων, όπως κείμενα, εικόνες, ή άλλα δεδομένα. Η απόσταση συνημιτόνου δεν ικανοποιεί τις τρεις βασικές ιδιότητες της μετρικής. Αυτό σημαίνει ότι δεν είναι θετικά οριστική, καθώς μπορεί να λάβει αρνητικές τιμές ανάλογα με τη σχετική κατεύθυνση των διανυσμάτων. Επίσης, δεν ικανοποιεί την τριγωνική ανισότητα, καθώς η απόσταση μεταξύ διανυσμάτων δεν είναι απαραίτητα μικρότερη ή ίση με το άθροισμα των αποστάσεων από ένα τρίτο διάνυσμα. Η απόσταση συνημιτόνου είναι μια δημοφιλής συνάρτηση απόστασης που χρησιμοποιείται ευρέως στην ανάλυση δεδομένων, την εξόρυξη γνώσης και τη μηχανική μάθηση [73].

Για να υπολογίσουμε την απόσταση συνημιτόνου μεταξύ των διανυσμάτων A και B, μπορούμε να χρησιμοποιήσουμε την ακόλουθη σχέση:

Cosine Distance(A, B) = 1 – Cosine Similarity(A, B)

Συγκεκριμένα, αφαιρούμε την ομοιότητα συνημιτόνου από το 1 για να υπολογίσουμε την απόσταση συνημιτόνου. Αυτή η απόσταση θα είναι μεταξύ 0 έως 2. Όταν τα δύο διανύσματα είναι απόλυτα ίδια, η απόσταση συνημιτόνου είναι 0 (υψηλή ομοιότητα). Αντίθετα, όταν τα δύο διανύσματα είναι απόλυτα αντίθετα, η απόσταση συνημιτόνου είναι 2 (μεγαλύτερη απόκλιση). Συνεπώς, η απόσταση συνημιτόνου μπορεί να χρησιμοποιηθεί τόσο για τη μέτρηση της ομοιότητας όσο και για τη μέτρηση της απόκλισης μεταξύ διανυσμάτων.

Ο τύπος που ακολουθεί, χρησιμοποιείται για τη μέτρηση της απόστασης συνημιτόνου μεταξύ δύο διανυσμάτων σε έναν πολλαπλασιαστικό χώρο. Η συνάρτηση απόσταση συνημιτόνου d(x, y) μεταξύ των διανυσμάτων x και y ορίζεται ως εξής:

$$d(x,y) = 1 - \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}}$$

Για την συνάρτηση απόστασης συνημιτόνου, τα x_i και y_i είναι οι συνιστώσες των δύο διανυσμάτων στην θέση i και n είναι ο αριθμός των διαστάσεων των διανυσμάτων [74].

Dynamic Partial Function

Η Δυναμική Μερική Συνάρτηση (Dynamic Partial Function – DPF) είναι μια μη μετρική συνάρτηση απόστασης, η οποία λειτουργεί καλύτερα από τις συναρτήσεις τύπου Minkowski για τη μέτρηση της αντιληπτικής ομοιότητας [73]. Η δυναμική μερική συνάρτηση δεν υπόκειται στους περιορισμούς που ισχύουν για τις μετρικές συναρτήσεις απόστασης. Αυτό σημαίνει ότι δεν περιορίζεται από τους κοινούς κανόνες που διέπουν την απόσταση μεταξύ αντικειμένων, όπως η τριγωνική ανισότητα και η συμμετρία.

Η μέθοδος της δυναμικής μερικής συνάρτησης λειτουργεί δυναμικά, ενεργοποιώντας διαφορετικά χαρακτηριστικά για τη μέτρηση της ομοιότητας ανάλογα με τα αντικείμενα που συγκρίνονται. Αυτό σημαίνει ότι η δυναμική μερική συνάρτηση δεν υποθέτει αυθαίρετα ότι όλα τα αντικείμενα πρέπει να έχουν τα ίδια χαρακτηριστικά ή να μοιάζουν μεταξύ τους σε ένα σταθερό σύνολο χαρακτηριστικών. Αντίθετα, επιτρέπει τη δυναμική ενεργοποίηση χαρακτηριστικών που είναι σημαντικά για την συγκεκριμένη σύγκριση των αντικειμένων. Αυτό επιτρέπει στην δυναμική μερική συνάρτηση να αντιμετωπίσει περισσότερο πολύπλοκες και πολυδιάστατες σχέσεις ομοιότητας μεταξύ αντικειμένων. Η δυναμική μερική συνάρτηση έχει εφαρμοστεί ευρέως σε διάφορους τομείς, μεταξύ των οποίων η ανάκτηση εικόνων, η αναγνώριση προτύπων και η αναζήτηση πληροφοριών. Ο τύπος υπολογισμού για τη μη μετρική απόσταση δυναμικής μερικής συνάρτησης $d_{DPF}(X, Y)$ μεταξύ των αντικειμένων X και Y ορίζεται ως εξής:

$$d_{DPF}(X,Y) = \left(\sum_{\delta_i \in \Delta_m} {\delta_i}^r \right)^{\frac{1}{r}}$$

Για την δυναμική μερική συνάρτηση, ο όρος r αντιπροσωπεύει τον παράγοντα Minkowski, ο οποίος καθορίζει τον τύπο της απόστασης, ενώ ο όρος Δ_m αναφέρεται στο σύνολο των m χαμηλότερων τιμών των δi, που αντιστοιχούν στα χαρακτηριστικά των X και Y που διαφέρουν λιγότερο μεταξύ τους [77].

Dynamic Time Warping

Η συνάρτηση απόστασης της **Δυναμικής Χρονικής Στρέβλωσης** (Dynamic Time Warping – DTW) είναι ένα μέτρο που χρησιμοποιείται συνήθως για τη σύγκριση της απόστασης σε ακολουθίες. Αυτές οι ακολουθίες μπορεί να είναι χρονικές σειρές (time series) ή γενικότερα, χαρακτηριστικά ακολουθιών που δειγματοληπτούνται σε ίσες χρονικές στιγμές. Αυτή η μέθοδος αρχικά εφαρμόστηκε για την αναγνώριση ομιλίας από τον Berndt και Clifford [78], αλλά μπορεί επίσης να χρησιμοποιηθεί για τον υπολογισμό της ομοιότητας

μεταξύ δύο χρονικών ακολουθιών που διαφέρουν σε μήκος ή ταχύτητα. Αν δύο χρονικές ακολουθίες είναι παρόμοιες, το γεγονός ότι δεν είναι ευθυγραμμισμένες στο χρόνο, έχει ως αποτέλεσμα η Ευκλείδεια απόσταση να ταιριάζει ένα-ένα τα σημεία των δύο σειρών και εσφαλμένα να καταλήγει στο συμπέρασμα ότι υπάρχει μεγάλη απόσταση μεταξύ τους. Αυτό το πρόβλημα διορθώνεται από τη συνάρτηση απόστασης δυναμικής χρονικής στρέβλωσης με γραμμική αντιστοίχιση [79]. Αυτή η συνάρτηση απόστασης είναι μία βελτίωση των μετρικών συναρτήσεων, αλλά και αυτή είναι ευαίσθητη στο θόρυβο, καθώς όλα τα σημεία των χρονικών ακολουθιών πρέπει να αντιστοιχηθούν, ακόμα και οι ακραίες τιμές. Η συνάρτηση απόστασης δυναμικής χρονικής στρέβλωσης σε αντίθεση με τις προαναφερθείσες μετρικές, ταιριάζει κάθε σημείο της πρώτης ακολουθίας με το κοντινότερο της δεύτερης, και τελικά επιλέγει τη συντομότερη απόσταση.

Η απόσταση δυναμικής χρονικής στρέβλωσης μεταξύ δύο χρονικών ακολουθιών δεν είναι συμμετρική και δεν ικανοποιεί την ιδιότητα της τριγωνικής ανισότητας. Αυτές οι αποκλίσεις από τις ιδιότητες των μετρικών καθιστούν αυτή την απόσταση μη μετρική. Η απόσταση δυναμικής χρονικής στρέβλωσης DTW(x, y) μεταξύ δύο χρονικών ακολουθιών x και y ορίζεται ως εξής:

$$\mathrm{DTW}(x,y) = \min_{i_1,i_2,\ldots,i_{k-\ell}} \sum_{j=1}^{k-\ell} w_{i_j}$$

Για την απόσταση δυναμικής χρονικής στρέβλωσης ισχύουν οι συνθήκες $i_1 < i_2 < \cdots < i_{k-\ell}$ και όπου $i_j + 1 \neq i_{j+1}$ για κάθε j. Αυτές οι συνθήκες εξασφαλίζουν ότι ο αλγόριθμος επιλέγει μια ευθυγράμμιση που έχει φυσικό νόημα και αποφεύγει τις ανεπιθύμητες επιλογές σημείων. Ο τύπος της απόστασης είναι χρήσιμος σε πολλές εφαρμογές, κυρίως εκεί όπου οι χρονικές σειρές έχουν διαφορετικά μήκη και ρυθμούς [80].

Συνοψίζοντας, στον τομέα της μαθηματικής ανάλυσης και της θεωρίας των τοπολογικών χώρων, υπάρχουν δύο βασικές κατηγορίες: οι μετρικοί χώροι και οι μη μετρικοί χώροι [68][69]. Οι μετρικοί χώροι ακολουθούν τους κανόνες της μετρικής απόστασης και προσφέρουν μια προσεγγιστική μέθοδο για τη μέτρηση αποστάσεων μεταξύ των σημείων [44]. Αντίθετα, οι μη μετρικοί χώροι είναι πιο γενικοί και δεν υπόκεινται στους περιορισμούς της μετρικής απόστασης. Επιπρόσθετα, παρουσιάζονται μερικές μη μετρικές απόστασης, περιλαμβάνοντας την απόκλιση Bregman, την απόσταση Συνημιτόνου (Cosine distance), την Δυναμική Μερική Συνάρτηση (DPF) και την Δυναμική Χρονική Στρέβλωση (DTW). Εξετάζοντας αναλυτικά κάθε μία από αυτές τις μεθόδους, διαφέρουν στον τρόπο υπολογισμού της απόστασης ή της ομοιότητας μεταξύ των σημείων ή αντικειμένων.

2.2 *Hamming distance*

Η απόσταση Hamming μεταξύ δύο συμβολοσειρών (string) ίσου μήκους, είναι ο αριθμός των θέσεων στις οποίες τα αντίστοιχα σύμβολα είναι διαφορετικά. Με άλλα λόγια, μετρά τον ελάχιστο αριθμό αντικαταστάσεων που απαιτούνται για την αλλαγή μιας συμβολοσειράς στην άλλη ή τον ελάχιστο αριθμό σφαλμάτων που θα μπορούσαν να έχουν μετατρέψει τη μία συμβολοσειρά στην άλλη. Σε ένα πιο γενικό πλαίσιο, η απόσταση Hamming είναι ένας από τους πολλούς αλγόριθμους ομοιότητας συμβολοσειρών για τη μέτρηση της απόστασης επεξεργασίας μεταξύ δύο ακολουθιών [81].

Η απόσταση Hamming πήρε το όνομά της το 1950 από τον Richard Hamming. Εισήγαγε την έννοια στη θεμελιώδη εργασία του σχετικά με τους κώδικες Hamming, ανίχνευση σφαλμάτων και κωδικούς διόρθωσης σφαλμάτων [82]. Η ανάλυση βάρους Hamming των bit χρησιμοποιείται σε διάφορους κλάδους, όπως στη θεωρία πληροφοριών, στη θεωρία κωδικοποίησης και στην κρυπτογραφία.

Ο υπολογισμός της απόστασης Hamming υλοποιείται με τον μαθηματικό τύπο:

$$\mathcal{H}_d(x, y) = \sum_{i=1}^{k} |x_i - y_i|$$

Στην εξίσωση, το αξιολογούμενο $x_i - y_i$ δηλαδή η απόσταση Hamming δύο λέξεων είναι ίσο με 0, όταν οι δύο λέξεις είναι ίδιες. Σε αντίθετη περίπτωση η απόσταση Hamming είναι ίσο 1. Όπως αναφέρθηκε, η απόσταση Hamming είναι ένα μέτρο που χρησιμοποιείται για να μετρήσει τη διαφορά μεταξύ δύο συμβολοσειρών ίσου μήκους. Αυτή η υλοποίηση της απόστασης Hamming είναι γνωστή ως κύβος Hamming, μια παραλλαγή της κλασικής απόστασης Hamming εφαρμοσμένη σε συμβολοσειρές που είναι συγκρίσιμες με τις κορυφές ενός υπερκύβου [81].

Στα παρακάτω παραδείγματα που ακολουθούν, φαίνεται ακριβώς πως υπολογίζεται η απόσταση Hamming.

Παράδειγμα 1

Στο δυαδικό σύστημα θέλοντας να συγκρίνουμε τις συμβολοσειρές s1 και s2, θα έχουμε:

s1: 0 1 1 0 0 1 0 0 1 0

s2: 1 0 0 0 1 1 0 0 1 1

Για να υπολογίσουμε την απόσταση Hamming μεταξύ αυτών των δύο συμβολοσειρών, συγκρίνουμε τα στοιχεία τους ένα προς ένα:

- Τα πρώτα στοιχεία είναι διαφορετικά (0 έναντι 1), οπότε απαιτείται μία αντικατάσταση.
- Τα δεύτερα στοιχεία είναι διαφορετικά (1 έναντι 0), οπότε απαιτείται μία ακόμη αντικατάσταση.
- Τα τρίτα στοιχεία είναι διαφορετικά (1 έναντι 0), οπότε απαιτείται μία τρίτη αντικατάσταση.
- Τα τέταρτα στοιχεία είναι τα ίδια (0 έναντι 0), οπότε δεν απαιτείται αντικατάσταση.
- Τα πέμπτα στοιχεία είναι διαφορετικά (0 έναντι 1), οπότε απαιτείται μία ακόμη αντικατάσταση.
- Τα έκτα στοιχεία είναι τα ίδια (1 έναντι 1), οπότε δεν απαιτείται αντικατάσταση.
- Τα έβδομα στοιχεία είναι τα ίδια (0 έναντι 0), οπότε δεν απαιτείται αντικατάσταση.
- Τα όγδοα στοιχεία είναι τα ίδια (0 έναντι 0), οπότε δεν απαιτείται αντικατάσταση.
- Τα ένατα στοιχεία είναι τα ίδια (1 έναντι 1), οπότε δεν απαιτείται αντικατάσταση.
- Τα δέκατα στοιχεία είναι διαφορετικά (0 έναντι 1), οπότε απαιτείται μία τελευταία αντικατάσταση.

Με κόκκινο χρώμα φαίνονται τα διαφορετικά στοιχεία των δύο συμβολοσειρών. Το συνολικό πλήθος των αντικαταστάσεων που απαιτούνται είναι 5. Επομένως, η απόσταση Hamming μεταξύ των δύο συμβολοσειρών είναι h(s1, s2) = 5.

Παράδειγμα 2

Υπολογίζοντας αντίστοιχα την απόσταση Hamming μεταξύ των συμβολοσειρών s3 και s4, έχουμε:

s3: 1 0 0 0 1 1 0 1 1 0

s4: 1 0 0 0 1 1 0 0 1 1

Διαπιστώνουμε ότι η απόσταση Hamming μεταξύ των δύο συμβολοσειρών είναι h(s3, s4) = 2. Γίνεται αντιληπτό, ότι όσο πιο μικρή είναι η απόσταση, τόσο πιο κοντά βρισκόμαστε στην ομοιότητα [83].

Παράδειγμα 3

Υπολογίζοντας αντίστοιχα την απόσταση Hamming μεταξύ των δύο συμβολοσειρών s5 και s6, έχουμε:

s5: k a r o l i n

s6: k a t h r i n

Σε αυτό το παράδειγμα, το συνολικό πλήθος των αντικαταστάσεων που απαιτούνται είναι 3. Συνεπώς, η απόσταση Hamming μεταξύ των δύο συμβολοσειρών είναι h(s5, s6) = 3.

Συνοψίζοντας, η απόσταση Hamming (Hamming distance) είναι πιο κοντά στην φιλοσοφία των κατηγορικών δεδομένων αφού λαμβάνει υπόψη της μόνο τα σημεία στα οποία διαφέρουν τα δύο διανύσματα χωρίς να ασχολείται με την αριθμητική της αντιστοίχησης των κατηγορικών δεδομένων σε φυσικούς αριθμούς [82]. Αυτή η μετρική απόστασης ωστόσο μπορεί να εφαρμοστεί σε συμβολοσειρές οποιουδήποτε είδους (γράμματα, bits ή δεκαδικά ψηφία), αρκεί να έχουν το ίδιο μήκος [84].

2.3 *Levenshtein distance (Edit distance)*

Ο Vladimir Levenshtein το 1966, διατύπωσε έναν αλγόριθμο ο οποίος εξετάζει την ομοιότητα μεταξύ δύο συμβολοσειρών, και είναι ευρύτερα γνωστός με τον όρο Edit distance [85], ο οποίος σε ελεύθερη μετάφραση σημαίνει απόσταση σύνταξης. Αυτός ο αλγόριθμος εξετάζει την ομοιότητα μεταξύ δύο συμβολοσειρών, υπολογίζοντας τον ελάχιστο αριθμό πράξεων που χρειάζονται προκειμένου να επιτευχθεί η μετατροπή της μιας συμβολοσειράς στην άλλη [86].

Η απόσταση Levenshtein μεταξύ δύο συμβολοσειρών α, b (μήκους |α| και |b| αντίστοιχα) δίνεται από το $lev_{\alpha,b}(|\alpha|, |b|)$ όπου:

$$lev_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ \min \begin{cases} lev_{a,b}(i-1,j) + 1 \\ lev_{a,b}(i,j-1) + 1 \\ lev_{a,b}(i-1,j-1) + 1_{a(i) \neq b(j)} \end{cases} & \text{otherwise.} \end{cases}$$

Ο δείκτης συνάρτησης $1(\alpha_i \neq b_i)$ είναι ίσο με 0, όταν οι χαρακτήρες α_i και b_i διαφέρουν μεταξύ τους. Σε αντίθετη περίπτωση ο δείκτης συνάρτησης είναι 1. Η $lev_{\alpha,b}(i, j)$ είναι η

απόσταση μεταξύ των πρώτων i χαρακτήρων της συμβολοσειράς α και των πρώτων j χαρακτήρων της συμβολοσειράς b. Είναι σημαντικό να αναφερθεί, ότι το πρώτο στοιχείο στο ελάχιστο αντιστοιχεί σε διαγραφή (από το "α" στο "b"), το δεύτερο σε εισαγωγή, και το τρίτο σε αντιστοίχιση ή αναντιστοιχία, ανάλογα με το εάν οι αντίστοιχοι χαρακτήρες είναι ίδιοι.

Η ομοιότητα των δύο συμβολοσειρών μπορεί να αναπαρασταθεί από την απόσταση Levenshtein. Προφανώς, όσο μεγαλύτερη είναι η απόσταση Levenshtein, τόσο μικρότερη είναι η ομοιότητα. Εάν δύο συμβολοσειρές S και T έχουν μήκη m και n αντίστοιχα, και χρησιμοποιείται η απόσταση Levenshtein για να αναπαρασταθεί η απόσταση επεξεργασίας τους και η Sim(S, T) για να υποδειχθεί η ομοιότητα τους, τότε ισχύει:

$$\mathrm{Sim}(S, T) = 1 - \frac{\mathrm{LD}}{\max(m, n)}$$

Ο όρος απόσταση επεξεργασίας αποτελεί έναν άλλο όρο για την απόσταση Levenshtein. Η προσέγγιση της απόστασης Levenshtein έχει χρησιμοποιηθεί σε ορθογραφικό έλεγχο, αναγνώριση ομιλίας, ανάλυση DNA και ανακάλυψη λογοκλοπής.

Ο πιο συνηθισμένος τρόπος υπολογισμού του αλγορίθμου είναι η προσέγγιση δυναμικού προγραμματισμού (dynamic programming). Ο δυναμικός προγραμματισμός είναι η τεχνική που συχνά χρησιμοποιείται για τον υπολογισμό της απόστασης Levenshtein μεταξύ δύο ακολουθιών χαρακτήρων. Η ιδέα πίσω από τον δυναμικό προγραμματισμό στην απόσταση Levenshtein είναι να χρησιμοποιηθεί ένας πίνακας (ή πίνακες) για να αποθηκευθούν τα ενδιάμεσα αποτελέσματα των υπολογισμών και να χρησιμοποιηθούν ξανά αυτά τα αποτελέσματα για να υπολογιστεί η τελική απόσταση [87].

Ο βασικός αλγόριθμος για τον υπολογισμό της απόστασης Levenshtein με δυναμικό προγραμματισμό περιλαμβάνει τα εξής βήματα [87]:

Στάδιο 1: Αρχικοποίηση

(α) Ορίζουμε τη μέτρηση της συμβολοσειράς S σε n και τη μέτρηση της συμβολοσειράς T σε m.

(β) Δημιουργία ενός πίνακα με στήλες από 0 έως m και γραμμές από 0 έως n, όπου m και n είναι το μήκος των δύο ακολουθιών που εξετάζονται. Αυτός ο πίνακας χρησιμοποιείται για την αποθήκευση των ενδιάμεσων αποτελεσμάτων και αρχικοποιείται με τιμές για να υποδείξει την απόσταση επεξεργασίας για τα διαδοχικά στοιχεία των ακολουθιών.

Στάδιο 2: Υπολογισμός

(α) Εξετάζουμε τα διαδοχικά στοιχεία της συμβολοσειράς S (i από 1 έως n).

(β) Εξετάζουμε τα διαδοχικά στοιχεία της συμβολοσειράς T (j από 1 έως m)

(γ) Το κόστος είναι 0 εάν ο χαρακτήρας S[i] ταιριάζει με τον χαρακτήρα T[j].

(δ) Το κόστος είναι 1 εάν ο χαρακτήρας S[i] δεν ταιριάζει με τον χαρακτήρα T[j].

(ε) Υπολογίζουμε την αξία του κελιού d[i, j] του πίνακα ως το ελάχιστο από τις τρεις γειτονικές τιμές, όπου:

- d[i-1, j] + 1 αντιπροσωπεύει το κόστος της διαγραφής ενός χαρακτήρα από τη συμβολοσειρά S.
- d[i, j-1] + 1 αντιπροσωπεύει το κόστος της εισαγωγής ενός χαρακτήρα στη συμβολοσειρά S.
- d[i-1, j-1] + "κόστος", αντιπροσωπεύει το κόστος της αντικατάστασης ενός χαρακτήρα στη συμβολοσειρά S με έναν χαρακτήρα από τη συμβολοσειρά T, όπου το κόστος είναι 0 αν οι χαρακτήρες ταιριάζουν, και 1 αν δεν ταιριάζουν.

Στάδιο 3: Τελικό αποτέλεσμα

Η απόσταση προκύπτει στο κελί d[n, m] όταν ολοκληρωθεί η επανάληψη στο στάδιο (2).

Παράδειγμα 1

Όταν η αρχική συμβολοσειρά είναι "CART" και η λέξη προορισμού είναι "MARCH", σύμφωνα με τον αλγόριθμο υπολογισμού της απόστασης Levenshtein, πραγματοποιείται η δημιουργία και η αρχικοποίηση του πίνακα. Το στάδιο 1 του αλγορίθμου παρουσιάζεται στην Εικόνα 5. Εάν υπάρχει αντιστοιχία, δηλαδή οι χαρακτήρες στις ίδιες θέσεις των δύο συμβολοσειρών είναι ίδιοι, αντιγράφουμε τη διαγώνια τιμή στο LevDist[i][j] (αναφέρεται στην τιμή του πίνακα LevDist). Εάν υπάρχει αναντιστοιχία, δηλαδή οι χαρακτήρες στις ίδιες θέσεις των δύο συμβολοσειρών είναι διαφορετικοί, τότε επιλέγουμε την ελάχιστη από τις τρεις γειτονικές τιμές του πίνακα και προσθέτουμε 1 σε αυτήν για να υπολογίσουμε την απόσταση.

Η απόσταση Levenshtein εμφανίζεται στο κελί d[n, m] του πίνακα όταν ολοκληρωθεί η επανάληψη στο στάδιο (2). Με αυτόν τον τρόπο ο αριθμός στην κάτω δεξιά γωνία είναι η απόσταση μεταξύ των λέξεων "CART" και "MARCH". Όπως φαίνεται στην Εικόνα 6 η απόσταση Levenshtein είναι 3.

ED	Ø	M	A	R	C	H
Ø	0	1	2	3	4	5
C	1					
A	2					
R	3					
T	4					

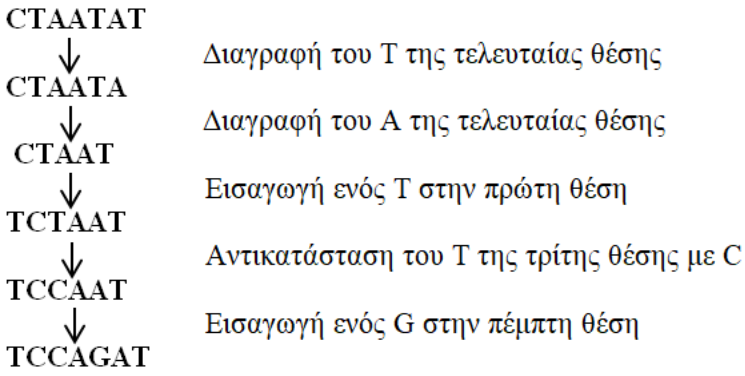
Εικόνα 5: Στάδιο αρχικοποίησης [87]

ED	ϕ	M	A	R	C	H
ϕ	0	1	2	3	4	5
C	1	1 D(n, m)	2	3	3	4
A	2	2	1	2	3	4
R	3	3	2	1	2	3
T	4	4	3	2	2	3

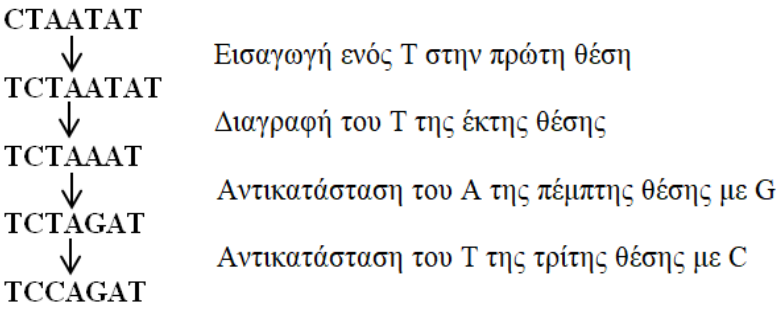
Εικόνα 6: Αποτέλεσμα παραδείγματος απόστασης Levenshtein [87]

Παράδειγμα 2

Ο όρος πράξη που χρειάζεται ο αλγόριθμος προκειμένου να επιτευχθεί η μετατροπή της μιας συμβολοσειράς στην άλλη, αντιστοιχεί σε τρεις βασικές λειτουργίες, στην εισαγωγή, διαγραφή ή στην αντικατάσταση ενός μοναδικού χαρακτήρα στη συμβολοσειρά [87]. Για παράδειγμα, η αλληλουχία CTAATAT μπορεί να μετατραπεί στην TCCAGAT με πέντε πράξεις όπως φαίνεται στην Εικόνα 7. Αυτό σημαίνει ότι η απόσταση σύνταξης μεταξύ τους είναι το πολύ 5. Στην πραγματικότητα, η απόσταση επεξεργασίας μεταξύ τους είναι 4, επειδή είναι εφικτό να μετασχηματίσουμε τη μια συμβολοσειρά στην άλλη με μια πράξη λιγότερη, όπως φαίνεται στην Εικόνα 8 [88].



Εικόνα 7: Πέντε πράξεις για την μετατροπή CTAATAT στην TCCAGAT



Εικόνα 8: Τέσσερις πράξεις για την μετατροπή CTAATAT στην TCCAGAT

Συνοψίζοντας, η μέθοδος υπολογισμού απόστασης Levenshtein αποτελεί μια σημαντική τεχνική βελτίωσης που συμβάλλει στη μείωση της ανακρίβειας κατά τον υπολογισμό της απόστασης μεταξύ δύο συμβολοσειρών [87]. Σε αντίθεση με την απόσταση Hamming, η απόσταση Levenshtein επιτρέπει εισαγωγές, διαγραφές και αντικαταστάσεις χαρακτήρων και μπορεί να χρησιμοποιηθεί για τη σύγκριση συμβολοσειρών διαφορετικού μήκους. Αυτό την καθιστά πολύ πιο ευέλικτη και χρήσιμη σε πολλές εφαρμογές, όπου οι συμβολοσειρές που πρέπει να συγκριθούν διαφέρουν στο μήκος τους [89]. Επίσης, όσο μικρότερη είναι η απόσταση Levenshtein, τόσο πιο όμοιες είναι οι συμβολοσειρές [87]. Συνολικά, αυτή η μέθοδος ανοίγει νέους ορίζοντες για τη βελτίωση της ακρίβειας και της απόδοσης στον τομέα της ανάλυσης κειμένου.

2.4 *Jaccard distance*

Η απόσταση Jaccard, γνωστή επίσης ως συντελεστής ομοιότητας Jaccard, συγκρίνει τις ομοιότητες και τις διαφορές σε σύνολα δεδομένων [90]. Αναπτύχθηκε από τον Grove Karl Gilbert το 1884 ως αναλογία επαλήθευσης [91]. Αργότερα αναπτύχθηκε ανεξάρτητα από τον Paul Jaccard, δίνοντας αρχικά το όνομα "coefficient of community" [92] και διατυπώθηκε ξανά ανεξάρτητα από τον T. Tanimoto [93]. Έτσι, ο δείκτης Tanimoto ή ο συντελεστής Tanimoto χρησιμοποιούνται επίσης σε ορισμένα πεδία. Ο συντελεστής Jaccard μετρά την ομοιότητα μεταξύ πεπερασμένων συνόλων δειγμάτων και ορίζεται ως το μέγεθος της τομής διαιρούμενο με το μέγεθος της ένωσης των συνόλων δεδομένων. Μαθηματικά, αναπαριστάται ως εξής:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

Ο δείκτης ομοιότητας κυμαίνεται από 0 έως 1. Εάν δύο σύνολα δεδομένων μοιράζονται τα ίδια ακριβώς στοιχεία, ο δείκτης ομοιότητας Jaccard θα είναι 1. Επίσης, όσο πιο κοντά στο 1 είναι ο δείκτης ομοιότητας Jaccard, τόσο πιο όμοια είναι τα δύο σύνολα δεδομένων. Εάν μια τομή B είναι κενή, δηλαδή δεν έχουν κοινά στοιχεία τότε η ομοιότητα τους θα είναι 0. Η απόσταση Jaccard χρησιμοποιείται ευρέως στην ανάκτηση πληροφορίας, την εξόρυξη δεδομένων, στη μηχανική μάθηση [94] και σε άλλες επιστήμες, όπου χρησιμοποιούνται δυαδικά δεδομένα. Τόσο η ακριβής λύση όσο και οι μέθοδοι προσέγγισης είναι διαθέσιμες για τον έλεγχο υποθέσεων με τον συντελεστή Jaccard [93].

Η απόσταση Jaccard μετρά την ανομοιότητα μεταξύ δύο συνόλων δεδομένων και είναι συμπληρωματική με τον συντελεστή Jaccard. Προκύπτει αφαιρώντας τον συντελεστή Jaccard από το 1, ή ισοδύναμα, διαιρώντας τη διαφορά των μεγεθών της ένωσης και της τομής δύο συνόλων με το μέγεθος της ένωσης. Η απόσταση Jaccard ορίζεται ως εξής:

$$d_J(A,B) = 1 - J(A,B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|}$$

Η απόσταση Jaccard είναι γνωστό ότι πληροί όλες τις ιδιότητες που απαιτούνται για να θεωρείται μια μετρική απόστασης. Ιδιαίτερα, αποδεικνύεται ότι ικανοποιεί την ιδιότητα της τριγωνικής ανισότητας. Η ικανότητα της απόστασης Jaccard να ικανοποιεί την τριγωνική

ανισότητα έχει παρατηρηθεί σε διάφορα πεδία, συμπεριλαμβανομένων των μετρικών μετασχηματισμών (metric transforms), των ενσωματώσεων σε διανυσματικούς χώρους (embeddings in vector spaces), των ελαχίστων ανεξάρτητων μεταθέσεων (min-wise independent permutations) ή των πολύπλοκων αριθμητικών υπολογισμών (cumbersome arithmetics) [94]. Αυτή η ιδιότητα ενισχύει ακόμη περισσότερο την αξιοπιστία και την ευελιξία της απόστασης Jaccard ως μετρικής για την αξιολόγηση της ομοιότητας ή της διαφορετικότητας μεταξύ των συνόλων δεδομένων [95].

Η απόσταση Jaccard δεν είναι η κατάλληλη μετρική για όλα τα είδη συνόλων δεδομένων. Σε σύνολα δεδομένων με ισορροπημένη κατανομή των δεδομένων σε κάθε κατηγορία, η μετρική Jaccard αποδίδει χαμηλή ακρίβεια στην κατηγοριοποίηση σε σύγκριση με άλλες μετρικές απόστασης. Αντίθετα, σε σύνολα δεδομένων με ανισορροπία στην κατανομή των δεδομένων σε κάθε κατηγορία, η μετρική Jaccard φαίνεται να βελτιώνει την ακρίβεια της κατηγοριοποίησης. Αυτό υποδηλώνει ότι η απόδοση της απόστασης Jaccard εξαρτάται από τον τρόπο με τον οποίο τα δεδομένα κατανέμονται σε διάφορες κατηγορίες [96].

Στα παρακάτω παραδείγματα που ακολουθούν, φαίνεται ακριβώς πως υπολογίζεται η απόσταση Jaccard.

Παράδειγμα 1:

Ας υποθέσουμε ότι έχουμε δύο σύνολα. Το σύνολο Α και το σύνολο Β.

$A = \{1,3,5,7,9\}$

$B = \{1,2,3,4,5,6,7,8\}$

Στη συνέχεια, για να υπολογίσουμε την ομοιότητα Jaccard μεταξύ τους, μετράμε πρώτα τον συνολικό αριθμό των παρατηρήσεων και στα δύο σύνολα, μετά διαιρούμε αυτόν τον συνολικό αριθμό με τον συνολικό αριθμό παρατηρήσεων σε κάθε σύνολο.

Αριθμός παρατήρησης και στα δύο: $(A \cap B) = \{1,2,3,5,7,6,9\} = 7$

Αριθμός παρατήρησης σε οποιοδήποτε από τα δύο: $(A \cup B) = \{1,2,3,4,5,6,7,8,9\} = 9$

Επομένως, $J(A, B) = |A \cap B| / |A \cup B| = |7| / |9| = 0,78$

Καθώς αυτός ο αριθμός είναι κοντά στο ένα μπορούμε να πούμε ότι τα δύο σύνολα είναι αρκετά παρόμοια.

Απόσταση Jaccard = 1 – 0,78 = 0,22

Παράδειγμα 2:

Όπως και στο παράδειγμα 1, αν έχουμε κείμενο αντί για αριθμούς,

A = {‘Lion’ , ‘Tiger’ , ‘Cheetah’ , ‘Leopard’ , ‘Rhino’}

B = {‘Lion’ , ‘Monkey’ , ‘Cheetah’ , ‘Cat’ , ‘Dog’}

Αριθμός παρατήρησης και στα δύο: (A∩B) = {'Lion' , 'Cheetah'} = 2

Αριθμός παρατήρησης σε οποιοδήποτε από τα δύο: (A∪B) = {‘Lion’ , ‘Tiger’ , ‘Cheetah’ , ‘Leopard’ , ‘Rhino’ , ‘Monkey’ , ‘Cat’ , ‘Dog’} = 8

Επομένως, J(A, B) = |A∩B| / |A∪B| = |2| / |8| = 0,25

Δεδομένου ότι αυτός ο αριθμός είναι αρκετά χαμηλός και κοντά στο 0, δείχνει ότι τα δύο σύνολα είναι αρκετά ανόμοια.

Απόσταση Jaccard = 1 – 0,25 = 0,75

Συνοψίζοντας, η απόσταση Jaccard είναι ένα μέτρο ομοιότητας που χρησιμοποιείται για να αξιολογήσει την ομοιότητα ή την διαφορά μεταξύ δύο ή περισσότερων συνόλων. Η τιμή της απόστασης Jaccard κυμαίνεται από 0 έως 1. Όταν η τιμή είναι 0, αυτό σημαίνει ότι τα δύο σύνολα δεν έχουν κοινά στοιχεία. Ενώ όταν η τιμή είναι 1, τα δύο σύνολα έχουν ακριβώς τα ίδια στοιχεία [94]. Η τριγωνική ανισότητα και η επεκτασιμότητά της καθιστούν την απόσταση Jaccard ένα ιδιαίτερα χρήσιμο εργαλείο για την εξαγωγή πληροφοριών από τα δεδομένα μας [95]. Η απόδοση της απόστασης Jaccard εξαρτάται σημαντικά από τον τρόπο με τον οποίο τα σύνολα δεδομένων κατανέμονται στις κατηγορίες [96].

3

Κανόνας Συμπύκνωσης Εγγύτερου Γείτονα (Condensed Nearest Neighbour rule, CNN-rule)

Οι τεχνικές μείωσης του πληθυσμού των δεδομένων (DRTs) έχουν στόχο τη μείωση των στιγμιότυπων ενός συνόλου δεδομένων. Στο κεφάλαιο αυτό παρουσιάζεται ο αλγόριθμος γνωστός με το όνομα κανόνας συμπύκνωσης εγγύτερου γείτονα (CNN-rule), ο οποίος είναι μια τεχνική μείωσης του πληθυσμού που βασίζεται στην έννοια των εγγύτερων γειτόνων για την κατηγοριοποίηση δεδομένων. Ένας περιορισμός του αλγορίθμου CNN-rule, είναι το γεγονός ότι δεν μπορεί να εφαρμοστεί σε σύνολα δεδομένων που περιέχουν κατηγορικά χαρακτηριστικά. Για τον λόγο αυτό μετά από έρευνα στην βιβλιογραφία, παρουσιάζονται εναλλακτικές συναρτήσεις απόστασης για κατηγορικά δεδομένα. Επιπρόσθετα, εξετάζουμε παραλλαγές του αλγορίθμου CNN-rule για κατηγορικά δεδομένα, χρησιμοποιώντας τροποποιημένες συναρτήσεις απόστασης όπως η Hamming απόσταση, η Levenshtein (Edit) απόσταση και η Jaccard απόσταση. Τέλος, προτείνουμε παραλλαγές του κατηγοριοποιητή k-NN για την αποτελεσματική κατηγοριοποίηση των κατηγορικών δεδομένων.

3.1 Αλγόριθμος CNN-rule

Ο αλγόριθμος γνωστός στη βιβλιογραφία με το όνομα κανόνας συμπύκνωσης εγγύτερου γείτονα (Condensed Nearest Neighbours rule – CNN-rule) θεωρείται η πρώτη

επίσημη πρόταση της τεχνικής επιλογής στιγμιότυπων (prototype selection algorithms – PS) για τον εγγύτερο γείτονα. Ο Hart, το 1968, ήταν ο πρώτος που παρουσίασε την έννοια της μείωσης του πληθυσμού των δεδομένων εκπαίδευσης με βασικό στόχο την γρήγορη εφαρμογή του k-NN κατηγοριοποιητή. Επιπλέον, παρουσίασε για πρώτη φόρα το γεγονός, ότι οποιοδήποτε από τα αντικείμενα του συνόλου εκπαίδευσης που δεν βρίσκονται κοντά στα σύνορα απόφασης των κλάσεων μπορούν να απομακρυνθούν με ασφάλεια, με αποτέλεσμα το κόστος της σειριακής αναζήτησης των γειτόνων να μειωθεί σε αρκετά μεγάλο βαθμό [97]. Ο αλγόριθμος CNN-rule έχει υιοθετηθεί ως μέτρο σύγκρισης και αποτελεί σημείο αναφοράς στις περισσότερες έρευνες που παρουσιάζουν νέες τεχνικές μείωσης του πληθυσμού των δεδομένων (Data Reduction Techniques – DRTs) [62]. Ακόμα και σήμερα αποτελεί έναν από τους πιο αποδοτικούς αλγόριθμους στον τομέα της εξόρυξης δεδομένων. Επίσης, πολλοί μεταγενέστεροι αλγόριθμοι αποτελούν παραλλαγές ή έχουν βασιστεί στην απλή ιδέα του CNN-rule [46]. Ο κανόνας μείωσης εγγύτερου γείτονα (Reduced Nearest Neighbor Rule – RNN-rule) [98], ο κανόνας επιλογής εγγύτερου γείτονα (Selective Nearest Neighbor Rule – SNN-rule) [99], ο τροποποιημένος κανόνας συμπύκνωσης εγγύτερου γείτονα (Modified Nearest Neighbour Rule – MCNN-rule) [66], ο γρήγορος κανόνας συμπύκνωσης εγγύτερου γείτονα (Fast Nearest Neighbor Rule – FCNN-rule) [66] και η οικογένεια αλγορίθμων IB [100] είναι παραδείγματα αυτής της κατηγορίας αλγορίθμων.

Η ιδέα στην οποία βασίζεται ο αλγόριθμος CNN-rule, όπως περιγράφηκε αναλυτικά στο κεφάλαιο 1, είναι η εξής: Αν η κλάση ενός αντικειμένου δεν συμφωνεί με την κλάση ενός γειτονικού αντικειμένου, τότε βρίσκεται κοντά στα σύνορα απόφασης και έτσι πρέπει να συμπεριληφθεί στο συμπυκνωμένο σύνολο εκπαίδευσης. Σε διαφορετική περίπτωση, το αντικείμενο βρίσκεται σε εσωτερική περιοχή της κλάσης και έτσι δεν συμπεριλαμβάνεται στο συμπυκνωμένο σύνολο εκπαίδευσης. Η Εικόνα 2 απεικονίζει αυτή τη στρατηγική.

Ο αλγόριθμος CNN-rule είναι ένας αλγόριθμος επιλογής που προσπαθεί να αποθηκεύσει στο συμπυκνωμένο σύνολο εκπαίδευσης μόνο εκείνα τα στοιχεία που βρίσκονται κοντά στα σύνορα απόφασης. Αυτό επιτυγχάνεται χρησιμοποιώντας ουσιαστικά δύο σύνολα δεδομένων. Ο αλγόριθμος δέχεται ως είσοδο το αρχικό σύνολο δεδομένων ως Σύνολο Εκπαίδευσης (Training Set) και ο σκοπός του είναι να δημιουργήσει ως έξοδο το Συμπυκνωμένο Σύνολο Δεδομένων (Condensing Set).

Ξεκινώντας η εκτέλεση και αφού το Συμπυκνωμένο Σύνολο Δεδομένων (CS) είναι κενό, μεταφέρεται σε αυτό ένα στοιχείο από το Σύνολο Εκπαίδευσης (TS) (γραμμή 2). Στη συνέχεια, ο αλγόριθμος CNN-rule προσπαθεί να κατηγοριοποιήσει τα στοιχεία του Συνόλου Εκπαίδευσης (TS) αναζητώντας τον εγγύτερο γείτονα (1-NN) στο Συμπυκνωμένο Σύνολο Δεδομένων (CS) (γραμμή 6). Εάν ένα στοιχείο του Συνόλου Εκπαίδευσης (TS) δεν έχει κατηγοριοποιηθεί σωστά, μετακινείται από το ένα σύνολο στο άλλο, δηλαδή από το Σύνολο

Εκπαίδευσης (TS) στο Συμπυκνωμένο Σύνολο Δεδομένων (CS) (γραμμές 7–11). Αντίθετα, τα στοιχεία που κατηγοριοποιούνται σωστά δεν μεταφέρονται. Η διαδικασία αυτή συνεχίζει μέχρι να σταματήσουν οι μεταφορές από το Σύνολο Εκπαίδευσης (TS) στο Συμπυκνωμένο Σύνολο Δεδομένων (CS), δηλαδή όλα τα στιγμιότυπα που ανήκουν στο Σύνολο Εκπαίδευσης (TS) να κατηγοριοποιούνται σωστά εφαρμόζοντας τον 1-NN κατηγοριοποιητή στα δεδομένα του Συμπυκνωμένου Συνόλου Δεδομένων (CS) (γραμμή 13). Αυτό διασφαλίζει ότι το περιεχόμενο του Συνόλου Εκπαίδευσης (TS) κατηγοριοποιείται σωστά από το περιεχόμενο του Συμπυκνωμένου Συνόλου Δεδομένων (CS). Το υπόλοιπο περιεχόμενο του Συνόλου Εκπαίδευσης (TS) απορρίπτεται (γραμμή 14). Ο αλγόριθμος CNN-rule θεωρεί ότι τα εσφαλμένα κατηγοριοποιημένα στοιχεία είναι πιθανότατα κοντά στα σύνορα απόφασης και επομένως πρέπει να τοποθετηθούν στο Συμπυκνωμένο Σύνολο Δεδομένων (CS). Η διαδικασία αυτή συνοψίζεται στον αλγόριθμο που παρουσιάζεται στην Εικόνα 9.

Algorithm 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

Εικόνα 9: Ο αλγόριθμος CNN-rule [62]

Ένα από τα πλεονεκτήματα του CNN-rule είναι ότι αποτελεί μια μη-παραμετρική (non-parametric) προσέγγιση, γεγονός που τον καθιστά εύκολο στην υλοποίηση και την εκτέλεση. Συγκεκριμένα, ο CNN-rule καθορίζει αυτόματα το μέγεθος του συμπυκνωμένου συνόλου δεδομένων. Αυτό σημαίνει ότι δεν απαιτείται από το χρήστη να εισάγει κάποια παράμετρο που να ορίζει το επιθυμητό μέγεθος. Η ύπαρξη τέτοιου είδους παραμέτρων καθιστά τη διαδικασία της προ-επεξεργασίας δεδομένων ως μία δύσκολη και χρονοβόρα

διαδικασία, αφού πρέπει αναζητηθούν και να τους ανατεθούν οι κατάλληλες τιμές, δηλαδή αυτές οι τιμές που επιτυγχάνουν την υψηλότερη απόδοση της κατηγοριοποίησης.

Αντίθετα, ο αλγόριθμος CNN-rule έχει τρία κύρια αρνητικά στοιχεία. Το πρώτο μειονέκτημα είναι η εξάρτηση του από τη σειρά με την οποία εξετάζει τα δεδομένα, είναι δηλαδή αλγόριθμος που βασίζεται στη σειρά (order dependent algorithm). Αυτό σημαίνει ότι έχει ως αποτέλεσμα την πιθανή δημιουργία διαφορετικών συμπυκνωμένων συνόλων με πολλαπλές εκτελέσεις του στα ίδια δεδομένα εκπαίδευσης, αν αυτά αποθηκευτούν με διαφορετική σειρά. Το δεύτερο μειονέκτημα είναι η εκτέλεση του αλγορίθμου που προϋποθέτει την παρουσία όλου του συνόλου δεδομένων στην κύρια μνήμη του υπολογιστή. Επειδή το μέγεθος των συνόλων δεδομένων που χρησιμοποιούνται στις εφαρμογές όλο και περισσότερο αυξάνεται, θεωρείται αυτό το μειονέκτημα σημαντικό. Τρίτο και τελευταίο σημαντικό μειονέκτημα είναι ότι ο CNN-rule είναι αλγόριθμος μη-σταδιακός (non-incremental). Τα σημαντικά αντικείμενα που βρίσκονται κοντά στα σύνορα απόφασης, τα οποία ο αλγόριθμος τελικά κρατάει, προστίθενται στο συμπυκνωμένο σύνολο δεδομένων στο τέλος της εκτέλεσης και όχι σταδιακά, οπότε η χρήση του δεν είναι δυνατή για περιβάλλοντα δυναμικά ή συνεχούς ροής (streaming) [62].

Έχοντας αναλύσει την παραπάνω διαδικασία λειτουργίας του αλγορίθμου CNN-rule, γίνεται εύκολα αντιληπτό το πρόβλημα που αποτελεί ο θόρυβος στα δεδομένα εκπαίδευσης. Όσο περισσότερες είναι οι κλάσεις και ο θόρυβος στα δεδομένα, τόσο περισσότερα σύνορα αποφάσεων θα υπάρχουν και συνεπώς θα επιτυγχάνεται χαμηλότερο ποσοστό μείωσης του πληθυσμού των δεδομένων. Η ανάγκη εφαρμογής μιας διαδικασίας απομάκρυνσης θορύβου, πολλές φορές είναι επιβεβλημένη [46].

Συνοψίζοντας, ο αλγόριθμος CNN-rule είναι ένας αλγόριθμος μηχανικής μάθησης που χρησιμοποιείται για την απλοποίηση και βελτίωση της απόδοσης των κατηγοριοποιητών με βάση τους k-Εγγύτερους Γείτονες (k-Nearest Neighbors). Συγκεκριμένα, ο αλγόριθμος CNN-rule προσπαθεί να μειώσει τον αριθμό των στιγμιότυπων του συνόλου εκπαίδευσης, διατηρώντας παράλληλα ακέραια την πληροφορία των δεδομένων. Παρόλα αυτά ο αλγόριθμος CNN-rule έχει δεχθεί κριτική ότι είναι ευαίσθητος στην σειρά με την οποία εξετάζει τα δεδομένα [62] και στον θόρυβο [46]. Ένας επιπλέον περιορισμός του αλγορίθμου CNN-rule, είναι το γεγονός ότι δεν μπορεί να εφαρμοστεί σε σύνολα δεδομένων που περιέχουν κατηγορικά χαρακτηριστικά. Για την αντιμετώπιση αυτών των προβλημάτων πραγματοποιήθηκαν τροποποιήσεις του αλγορίθμου CNN-rule [62].

3.2 Παραλλαγές CNN-rule για κατηγορικά δεδομένα

Οι παραλλαγές του αλγορίθμου CNN-rule αποτελούν μια σημαντική προσέγγιση για τη διαχείριση κατηγορικών (ονομαστικών) δεδομένων. Η σημασία αυτών των παραλλαγών αναδεικνύεται από την ανάγκη να αντιμετωπιστούν τα χαρακτηριστικά των κατηγορικών (categorical attributes) δεδομένων στην επιστήμη της μηχανικής μάθησης. Τα κατηγορικά δεδομένα είναι διαφορετικά από τα αριθμητικά, καθώς τα χαρακτηριστικά τους παίρνουν τιμές από ένα προκαθορισμένο, πεπερασμένο σύνολο. Αυτά τα χαρακτηριστικά, σύμφωνα με τον τρόπο που μετρούν την πληροφορία διαχωρίζονται σε ονομαστικά χαρακτηριστικά (nominal attributes) και τακτικά χαρακτηριστικά (ordinal attributes). Τα κατηγορικά δεδομένα δεν μπορούν να υποβληθούν απευθείας στους αλγόριθμους μηχανικής μάθησης που συχνά απαιτούν αριθμητικές τιμές για την εκπαίδευση και την κατηγοριοποίηση [101].

Ο αλγόριθμος CNN-rule βασίζεται στην έννοια των εγγύτερων γειτόνων για την κατηγοριοποίηση δεδομένων. Ο όρος "εγγύτεροι γείτονες" αναφέρεται στα δείγματα που είναι πιο παρόμοια μεταξύ τους, χρησιμοποιώντας μια συνάρτηση απόστασης. Η συνάρτηση απόστασης χρησιμοποιείται για να αποφασίσει ποιοι γείτονες είναι πιο κοντά σε ένα διάνυσμα εισόδου, η οποία μπορεί να έχει δραματική επίδραση σε ένα σύστημα μάθησης που βασίζεται σε στιγμιότυπα (instance-based learning). Στην περίπτωση του CNN-rule, μία συνήθης συνάρτηση απόστασης που χρησιμοποιείται είναι η Ευκλείδεια απόσταση. Ο αλγόριθμος του πλησιέστερου γείτονα συνήθως χρησιμοποιεί παραλλαγές της συνάρτησης της Ευκλείδειας απόστασης, η οποία ορίζεται ως εξής:

$$E(\vec{x}, \vec{y}) = \sqrt{\sum_{i=1}^{m} (x_i - y_i)^2}$$

Για την συνάρτηση της Ευκλείδειας απόστασης, όπου x και y είναι τα δύο διανύσματα εισόδου, m είναι ο αριθμός των χαρακτηριστικών εισόδου και x_i και y_i είναι οι τιμές εισόδου για το χαρακτηριστικό εισόδου i. Αυτή η συνάρτηση απόστασης είναι κατάλληλη μόνο για αριθμητικά δεδομένα όπου μπορούμε να υπολογίσουμε την απόσταση μεταξύ δύο σημείων. Η Ευκλείδεια απόσταση εφαρμόζεται και σε κατηγορικά δεδομένα, αλλά συνήθως σε αυτές τις περιπτώσεις η ερμηνεία είναι διαφορετική. Για παράδειγμα, σε δυαδικά δεδομένα, είναι εύκολο να χαρακτηριστούν όμοια. Στην περίσταση αυτή, αν η διαφορά των τιμών τους είναι 0 αναφέρονται ως όμοια ενώ το αντίθετο ισχύει αν η διαφορά

των τιμών τους είναι 1. Αλλά στην πλειονότητα των πειραμάτων οι μεταβλητές μπορεί να είναι αριθμητικές, κατηγορικές, δυαδικές ή ονομαστικές [45].

Μια εναλλακτική συνάρτηση απόστασης για τη χρήση σε δεδομένα που περιέχουν ονομαστικά χαρακτηριστικά είναι η **Μετρική Διαφοράς Τιμής** (Value Difference Metric – VDM). Η απόσταση $VDM_\alpha(x, y)$ μεταξύ δύο τιμών x και y ενός μοναδικού χαρακτηριστικού α ορίζεται ως εξής:

$$\mathrm{VDM}_a(x, y) = \sqrt{\sum_{a=1}^{C} \left(\frac{N_{a,x,c}}{N_{a,x}} - \frac{N_{a,y,c}}{N_{a,y}} \right)^2}$$

Για την συνάρτηση απόστασης Value Difference Metric, το C είναι ο συνολικός αριθμός των κλάσεων εξόδου. Το $N_{\alpha,x}$ είναι ο αριθμός των εμφανίσεων (ή παρατηρήσεων) που το χαρακτηριστικό "α" είχε την τιμή x στο σύνολο δεδομένων. Το $N_{\alpha,x,c}$ είναι ο αριθμός των εμφανίσεων που το χαρακτηριστικό "α" είχε την τιμή x και η κλάση εξόδου ήταν c. Το $N_{\alpha,y}$ είναι ο αριθμός των εμφανίσεων που το χαρακτηριστικό "α" είχε την τιμή y στο σύνολο δεδομένων. Το $N_{\alpha,y,c}$ είναι ο αριθμός των εμφανίσεων που το χαρακτηριστικό "α" είχε την τιμή y και η κλάση εξόδου ήταν c [102].

Μια ετερογενή συνάρτηση απόστασης, όπως η **Μετρική Διαφοράς Ετερογενούς Τιμής** (Heterogeneous Value Difference Metric – HVDM), είναι κατάλληλη για τη χρήση σε δεδομένα που περιέχουν τόσο αριθμητικά όσο και ονομαστικά χαρακτηριστικά. Η κεντρική ιδέα της συνάρτησης μετρικής διαφοράς ετερογενούς τιμής (HVDM) είναι πως "δύο στιγμιότυπα (δείγματα) θεωρούνται πιο κοντινά εάν έχουν περισσότερες παρόμοιες κατηγοριοποιήσεις, ανεξάρτητα από τη σειρά των τιμών τους". Η απόσταση HVDM(x, y) μεταξύ δύο σημείων δεδομένων που συμβολίζονται ως x και y ορίζεται ως εξής:

$$\mathrm{HVDM}(\vec{x}, \vec{y}) = \sqrt{\sum_{a=1}^{m} d_a^2(x_a, y_a)}$$

Η συνάρτηση $d_\alpha(x, y)$ υπολογίζει την απόσταση για ένα μεμονωμένο χαρακτηριστικό "α" μεταξύ δύο δειγμάτων δεδομένων x και y και ορίζεται ως εξής:

$$d_a(x, y) = \begin{cases} 1, & \text{if } x \text{ or } y \text{ is unknown; otherwise...} \\ \text{VDM}_a(x, y), & \text{if } a \text{ is nominal, else} \\ \frac{|x - y|}{4\sigma_a}. & \text{if } a \text{ is numeric.} \end{cases}$$

Για την συνάρτηση $d_\alpha(x, y)$ εάν το x ή το y έχουν άγνωστη τιμή για το χαρακτηριστικό "α", η απόσταση ορίζεται σε 1. Άγνωστες τιμές μπορούν να προκύψουν όταν τα δεδομένα λείπουν ή είναι απροσδιόριστα. Εάν το χαρακτηριστικό "α" είναι ονομαστικό (κατηγορικό), τότε χρησιμοποιείται η συνάρτηση μετρικής διαφοράς τιμής $VDM_\alpha(x, y)$ για τον υπολογισμό της απόστασης. Εάν το χαρακτηριστικό "α" είναι αριθμητικό, η απόσταση υπολογίζεται με βάση την απόλυτη διαφορά μεταξύ των τιμών ($|x - y|$), κανονικοποιημένη με το τετραπλάσιο της τυπικής απόκλισης (σ_α) των τιμών για το χαρακτηριστικό "α" στο σύνολο δεδομένων εκπαίδευσης.

Αυτή η συνάρτηση απόστασης παρέχει την κατάλληλη κανονικοποίηση μεταξύ αριθμητικών και ονομαστικών χαρακτηριστικών, καθώς και μεταξύ αριθμητικών χαρακτηριστικών διαφορετικής κλίμακας. Επιπλέον, χειρίζεται άγνωστες τιμές εισόδου, εκχωρώντας τους μια μεγάλη απόσταση, έτσι ώστε τα στιγμιότυπα με χαρακτηριστικά που λείπουν να είναι λιγότερο πιθανό να χρησιμοποιηθούν ως γείτονες σε σύγκριση με τα στιγμιότυπα που έχουν καθορισμένα όλα τα χαρακτηριστικά. Η χρήση μιας σταθερής τιμής για την απόσταση προς ένα άγνωστο χαρακτηριστικό χρησιμεύει στην αποτελεσματική παράβλεψη αυτών των χαρακτηριστικών όταν απουσιάζουν τα δεδομένα σε ένα στιγμιότυπο που χρειάζεται να κατηγοριοποιηθεί [58].

Στον τομέα της μηχανικής μάθησης, υπάρχουν διάφορες συναρτήσεις απόστασης που είναι διαθέσιμες κυρίως για αριθμητικά δεδομένα. Ένα παράδειγμα από αυτές είναι η απόσταση Hamming (Hamming, 1950) [82], η Edit απόσταση ή αλλιώς Levenshtein απόσταση (Levenshtein, 1966) [85] και η Jaccard απόσταση (Gilbert, 1884) [91]. Ο στόχος και η αναλυτική περιγραφή λειτουργίας των τριών συναρτήσεων απόστασης για μετρικούς χώρους περιγράφονται στο κεφάλαιο 2. Όπως αναφέρθηκε, μια αδυναμία του αλγορίθμου CNN-rule, είναι το γεγονός ότι δεν μπορεί να εφαρμοστεί σε σύνολα δεδομένων που περιέχουν κατηγορικά χαρακτηριστικά. Για να ενσωματώσουμε τις προαναφερθείσες συναρτήσεις απόστασης σε έναν αλγόριθμο CNN-rule για κατηγορικά δεδομένα, τροποποιήσαμε τη λειτουργία των συναρτήσεων απόστασης για να υπολογίζει αποστάσεις σε δεδομένα που περιέχουν κατηγορικά χαρακτηριστικά. Έπειτα, τροποποιήσαμε τους κανόνες και την διαδικασία εκτέλεσης του αλγορίθμου. Οι κανόνες του αλγορίθμου CNN-rule πρέπει να τροποποιηθούν για να λαμβάνουν υπόψη την εκάστοτε συνάρτηση απόστασης. Στην

παρούσα εργασία, η κωδικοποίηση του αλγορίθμου CNN-rule πραγματοποιήθηκε στην γλώσσα προγραμματισμού C++.

Ανάλογα με τον τύπο της συνάρτησης απόστασης που θέλουμε να χρησιμοποιήσουμε, προκύπτουν οι ακόλουθες παραλλαγές αλγορίθμου CNN-rule για κατηγορικά δεδομένα:

1. Ο αλγόριθμος **CNN Hamming** χρησιμοποιείται για τη σύγκριση συμβολοσειρών κατηγορικών δεδομένων. Υπολογίζει τον αριθμό των διαφορετικών χαρακτήρων μεταξύ των δύο συμβολοσειρών ως απόσταση μεταξύ τους.

Αλγόριθμος CNN Hamming

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

using namespace std;

struct TrainItem{
  int classAttr;
  string *attr;
  int aa;
  char SX;
};

unsigned long long computations;

int ATTRIBUTES;
int CLASSES;

void readTrainData(TrainItem[], char[], int);
int NearestNeighbor(TrainItem, TrainItem[], int);
int countLinesAttrs(char[], int&, int&);

int main(int argc, char *argv[]){
  static TrainItem *trainData;
  int i, j, nn, c, g, f;
  int dataNumV;
```

```
bool fl;

unsigned long long sumd = 0;
int sumc = 0;

if (argc != 3){
  cout<<"ERROR. Number of parameters"<<endl;
  return 1;
}

char *fileName = argv[1];
char *fileName2 = argv[2];

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

trainData = new TrainItem[dataNumV];
readTrainData(trainData, fileName, dataNumV);

computations = 0;

g=0;
//Harts Condensing Nearest Neighbor Rule
trainData[0].SX='C';

//C:CS, T:TS
//dataNumV: number of elements
do{
  fl=false;
  g++;
  for (i=0; i<dataNumV; i++){
    if (trainData[i].SX == 'T'){
      nn=NearestNeighbor(trainData[i], trainData, dataNumV);
      if (nn!=-1){
                                    if (trainData[nn].classAttr != trainData[i].classAttr){
                                            trainData[i].SX='C';
                                            fl=true;
                                    }
                            }
                    }
  }
  c=0;
```

```
    for (i=0; i<dataNumV; i++){
      if (trainData[i].SX == 'T'){
        c++;
      }
    }
  }
  while ((fl == true) && (c != 0));

  ofstream out1;
  out1.open(fileName2);

  c=0;
  for (i=0; i<dataNumV; i++){
    if (trainData[i].SX == 'C'){
      for (j=0; j<ATTRIBUTES; j++){
        out1<<trainData[i].attr[j]<<"\t";
      }
      out1<<trainData[i].classAttr<<endl;
      c++;
    }
  }
  out1.close();

  cout<<"Classes: "<<CLASSES<<endl;
  cout<<"Attributes: "<<ATTRIBUTES<<endl;
  cout<<"Items:"<<dataNumV<<endl;
  cout<<"Prototypes: "<<c<<endl;
  cout<<"Computations: "<<computations<<endl;
  cout<<"Reduction Rate: "<<(float)(1-((float)c/(float)dataNumV))*100<<endl;

  return 0;
}

int NearestNeighbor(TrainItem ti, TrainItem trainData[], int dataNumV){
  int nn, k, j;
  float u, sum, min;
  bool flag = false;
  float dist;

  nn=-1;

  for (k=0; k<dataNumV; k++){
    //search for the NN in CS, not in TS
    if (trainData[k].SX == 'T'){
```

```
      continue;
    }
    if (ti.aa == k){
      continue;
    }
    sum=0;
    computations++;
    //Υπολογισμός Hamming απόστασης μεταξύ των πολλαπλών συμβολοσειρών
    for (j=0; j<ATTRIBUTES; j++){
      if (ti.attr[j] != trainData[k].attr[j]) {
        sum += 1;
      }
    }
    dist=sqrt(sum);

    if (flag == false){
                    min = dist;
                    nn = k;
                    flag = true;
    }

    if (dist<min){
      min=dist;
      nn=k;
    }
  }
  return nn;
}

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

  ifstream dat;
  dat.open(fileName);
  CLASSES=0;
  bool fl;

  for (i=0; i<n; i++){
    trainData[i].attr = new string[ATTRIBUTES];
    for (j = 0; j < ATTRIBUTES; j++) {
      getline(dat, trainData[i].attr[j], '\t');
    }
    dat>>trainData[i].classAttr ;
    trainData[i].SX = 'T';
```

```
    trainData[i].aa=i;
    fl = false;

    for (j=0; j<i; j++){
      if (trainData[i].classAttr == trainData[j].classAttr){
        fl = true;
        break;
      }
    }
    if (!fl){
      CLASSES++;
    }
  }
  dat.close();
}

int countLinesAttrs(char fileName[], int &lines, int &attrs){
  unsigned int i, c;

  ifstream dat;
  dat.open(fileName);

  if (!dat){
    return 1;
  }

  string lline;

  getline(dat, lline);
  c=0;
  for (i=0; i<lline.length(); i++){
    if (lline[i] == '\t'){
      c++;
    }
  }
  dat.close();
  attrs = c;

  dat.open(fileName);
  i=0;
  while (!dat.eof()){
    getline(dat, lline);
    i++;
  }
```

```
  lines = i-1;
  dat.close();
  return 0;
}
```

2. Ο αλγόριθμος **CNN Levenshtein** χρησιμοποιείται για τον υπολογισμό του ελάχιστου αριθμού επεξεργασιών (προσθήκη, διαγραφή ή αντικατάσταση χαρακτήρων) που απαιτούνται για να μετατραπεί η μια συμβολοσειρά κατηγορικών χαρακτήρων στην άλλη.

Αλγόριθμος CNN Levenshtein

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

using namespace std;

struct TrainItem{
  int classAttr;
  string *attr;
  int aa;
  char SX;
};

unsigned long long computations;

int ATTRIBUTES;
int CLASSES;

void readTrainData(TrainItem[], char[], int);
int NearestNeighbor(TrainItem, TrainItem[], int);
int countLinesAttrs(char[], int&, int&);

int main(int argc, char *argv[]){
  static TrainItem *trainData;
```

```
int i, j, nn, c, g, f;
int dataNumV;
bool fl;

unsigned long long sumd = 0;
int sumc = 0;

if (argc != 3){
  cout<<"ERROR. Number of parameters"<<endl;
  return 1;
}

char *fileName = argv[1];
char *fileName2 = argv[2];

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

trainData = new TrainItem[dataNumV];
readTrainData(trainData, fileName, dataNumV);

computations = 0;

g=0;
//Harts Condensing Nearest Neighbor Rule
trainData[0].SX='C';

//C:CS, T:TS
//dataNumV: number of elements
do{
  fl=false;
  g++;
  for (i=0; i<dataNumV; i++){
    if (trainData[i].SX == 'T'){
      nn=NearestNeighbor(trainData[i], trainData, dataNumV);
      if (nn!=-1){
                                    if (trainData[nn].classAttr != trainData[i].classAttr){
                                          trainData[i].SX='C';
                                          fl=true;
                                    }
                              }
                  }
```

```
      }
      c=0;
      for (i=0; i<dataNumV; i++){
        if (trainData[i].SX == 'T'){
          c++;
        }
      }
    }
    while ((fl == true) && (c != 0));

    ofstream out1;
    out1.open(fileName2);

    c=0;
    for (i=0; i<dataNumV; i++){
      if (trainData[i].SX == 'C'){
        for (j=0; j<ATTRIBUTES; j++){
          out1<<trainData[i].attr[j]<<"\t";
        }
        out1<<trainData[i].classAttr;
        c++;
      }
    }
    out1.close();

    cout<<"Classes: "<<CLASSES<<endl;
    cout<<"Attributes: "<<ATTRIBUTES<<endl;
    cout<<"Items:"<<dataNumV<<endl;
    cout<<"Prototypes: "<<c<<endl;
    cout<<"Computations: "<<computations<<endl;
    cout<<"Reduction Rate: "<<(float)(1-((float)c/(float)dataNumV))*100<<endl;

    return 0;
}

int LevenshteinDistance(string s1, string s2) {
    const int m = s1.length();
    const int n = s2.length();

    int dp[m + 1][n + 1];

    for (int i = 0; i <= m; i++) {
        for (int j = 0; j <= n; j++) {
            if (i == 0) {
```

```
            dp[i][j] = j;
          } else if (j == 0) {
            dp[i][j] = i;
          } else if (s1[i - 1] == s2[j - 1]) {
            dp[i][j] = dp[i - 1][j - 1];
          } else {
            dp[i][j] = 1 + min(dp[i - 1][j], min(dp[i][j - 1], dp[i - 1][j - 1]));
          }
        }
      }

    return dp[m][n];
}

int NearestNeighbor(TrainItem ti, TrainItem trainData[], int dataNumV){
   int nn, k, j;
   float u, sum, min;
   bool flag = false;
   float dist;

   nn=-1;

   for (k=0; k<dataNumV; k++){

     if (trainData[k].SX == 'T'){
       continue;
     }
     if (ti.aa == k){
       continue;
     }
     sum=0;
     computations++;
     //Υπολογισμός Levenshtein απόστασης μεταξύ των πολλαπλών συμβολοσειρών
     for (j=0; j<ATTRIBUTES; j++){
                    sum += LevenshteinDistance(ti.attr[j], trainData[k].attr[j]);
     }
     dist=sqrt(sum);

     if (flag == false){
                     min = dist;
                     nn = k;
                     flag = true;
     }
```

```
    if (dist<min){
      min=dist;
      nn=k;
    }
  }
  return nn;
}

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

  ifstream dat;
  dat.open(fileName);
  CLASSES=0;
  bool fl;

  for (i=0; i<n; i++){
    trainData[i].attr = new string[ATTRIBUTES];
    for (j = 0; j < ATTRIBUTES; j++) {
      getline(dat, trainData[i].attr[j], '\t');
    }
    dat>>trainData[i].classAttr ;
    trainData[i].SX = 'T';
    trainData[i].aa=i;
    fl = false;

    for (j=0; j<i; j++){
      if (trainData[i].classAttr == trainData[j].classAttr){
        fl = true;
        break;
      }
    }
    if (!fl){
      CLASSES++;
    }
  }
  dat.close();
}

int countLinesAttrs(char fileName[], int &lines, int &attrs){
  unsigned int i, c;

  ifstream dat;
  dat.open(fileName);
```

```
  if (!dat){
    return 1;
  }

  string lline;

  getline(dat, lline);
  c=0;
  for (i=0; i<lline.length(); i++){
    if (lline[i] == '\t'){
      c++;
    }
  }
  dat.close();
  attrs = c;

  dat.open(fileName);
  i=0;
  while (!dat.eof()){
    getline(dat, lline);
    i++;
  }

  lines = i-1;
  dat.close();
  return 0;
}
```

3. Ο αλγόριθμος **CNN Jaccard** χρησιμοποιείται για τη μέτρηση της ομοιότητας ή της απόστασης μεταξύ συμβολοσειρών ή χαρακτήρων. Υπολογίζει τη Jaccard απόσταση βασιζόμενη στον αριθμό των κοινών κατηγοριών και τον συνολικό αριθμό των χαρακτήρων στις δύο συμβολοσειρές.

Αλγόριθμος CNN Jaccard

```
#include <stdlib.h>
#include <fstream>
#include <iostream>
#include <math.h>
```

```
#include <string>
#include <cstring>
#include <cstdlib>

using namespace std;

struct TrainItem{
  int classAttr;
  string *attr;
  int aa;
  char SX;
};

unsigned long long computations;

int ATTRIBUTES;
int CLASSES;

void readTrainData(TrainItem[], char[], int);
int NearestNeighbor(TrainItem, TrainItem[], int);
int countLinesAttrs(char[], int&, int&);

int main(int argc, char *argv[]){
  static TrainItem *trainData;
  int i, j, nn, c, g, f;
  int dataNumV;
  bool fl;

  unsigned long long sumd = 0;
  int sumc = 0;

  if (argc != 3){
    cout<<"ERROR. Number of parameters"<<endl;
    return 1;
  }

  char *fileName = argv[1];
  char *fileName2 = argv[2];


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

```
trainData = new TrainItem[dataNumV];
readTrainData(trainData, fileName, dataNumV);

computations = 0;

g=0;
//Harts Condensing Nearest Neighbor Rule
trainData[0].SX='C';

//C:CS, T:TS
//dataNumV: number of elements
do{
  fl=false;
  g++;
  for (i=0; i<dataNumV; i++){
    if (trainData[i].SX == 'T'){
      nn=NearestNeighbor(trainData[i], trainData, dataNumV);
      if (nn!=-1){
                                    if (trainData[nn].classAttr != trainData[i].classAttr){
                                            trainData[i].SX='C';
                                            fl=true;
                                    }
                            }
                    }
  }
  c=0;
  for (i=0; i<dataNumV; i++){
    if (trainData[i].SX == 'T'){
      c++;
    }
  }
}
while ((fl == true) && (c != 0));

ofstream out1;
out1.open(fileName2);

c=0;
for (i=0; i<dataNumV; i++){
  if (trainData[i].SX == 'C'){
    for (j=0; j<ATTRIBUTES; j++){
      out1<<trainData[i].attr[j]<<"\t";
    }
```

```
        out1<<trainData[i].classAttr;
        c++;
      }
    }
    out1.close();

    cout<<"Classes: "<<CLASSES<<endl;
    cout<<"Attributes: "<<ATTRIBUTES<<endl;
    cout<<"Items:"<<dataNumV<<endl;
    cout<<"Prototypes: "<<c<<endl;
    cout<<"Computations: "<<computations<<endl;
    cout<<"Reduction Rate: "<<(float)(1-((float)c/(float)dataNumV))*100<<endl;

    return 0;
}

double jaccardDistance(string s1, string s2) {
    size_t m = s1.length();
    size_t n = s2.length();
    size_t intersection = 0;
    size_t unionSize = m + n;

    //Check for empty strings
    if (m == 0 && n == 0)
        return 0.0;

    //Iterate over characters in s1
    for (size_t i = 0; i < m; i++) {
        // Check if character exists in s2
        if (strchr(s2.c_str(), s1[i]) != NULL) {
            intersection++;
            unionSize--;
        }
    }

    //Iterate over characters in s2
    for (size_t i = 0; i < n; i++) {
        //Check if character exists in s1
        if (strchr(s1.c_str(), s2[i]) == NULL) {
            unionSize++;
        }
    }
    //Calculate Jaccard distance
          double jaccardSimilarity = (double)(intersection) / unionSize;
```

```
    double jaccardDistance = 1.0 - jaccardSimilarity;

    return jaccardDistance;
}

int NearestNeighbor(TrainItem ti, TrainItem trainData[], int dataNumV){
  int nn, k, j;
  double minDistance = 1.0;
  bool flag = false;
  double sum,dist;
  double min;

  nn = -1;

  for (k = 0; k < dataNumV; k++) {
    if (trainData[k].SX == 'T') {
      continue;
    }
    if (ti.aa == k) {
      continue;
    }
          sum=0;
    computations++;
    //Υπολογισμός Jaccard απόστασης ανάμεσα στα σύνολα γνωρισμάτων
    for (j=0; j<ATTRIBUTES; j++){
                  sum += jaccardDistance(ti.attr[j], trainData[k].attr[j]);
    }
    dist = sum/ATTRIBUTES;

    if (flag == false){
                  min = dist;
                  nn = k;
                  flag = true;
    }

    if (dist<min){
      min=dist;
      nn=k;
    }
  }
  return nn;
}

void readTrainData(TrainItem trainData[], char fileName[], int n){
```

```
  int i,j;

  ifstream dat;
  dat.open(fileName);
  CLASSES=0;
  bool fl;

  for (i=0; i<n; i++){
    trainData[i].attr = new string[ATTRIBUTES];
    for (j = 0; j < ATTRIBUTES; j++) {
      getline(dat, trainData[i].attr[j], '\t');
    }
    dat>>trainData[i].classAttr ;
    trainData[i].SX = 'T';
    trainData[i].aa=i;
    fl = false;

    for (j=0; j<i; j++){
      if (trainData[i].classAttr == trainData[j].classAttr){
        fl = true;
        break;
      }
    }
    if (!fl){
      CLASSES++;
    }
  }
  dat.close();
}

int countLinesAttrs(char fileName[], int &lines, int &attrs){
  unsigned int i, c;

  ifstream dat;
  dat.open(fileName);

  if (!dat){
    return 1;
  }

  string lline;

  getline(dat, lline);
  c=0;
```

```
  for (i=0; i<lline.length(); i++){
    if (lline[i] == '\t'){
      c++;
    }
  }
  dat.close();
  attrs = c;

  dat.open(fileName);
  i=0;
  while (!dat.eof()){
    getline(dat, lline);
    i++;
  }

  lines = i-1;
  dat.close();
  return 0;
}
```

Συνοψίζοντας, οι παραλλαγές του αλγορίθμου CNN-rule που χρησιμοποιούν τις Hamming, Levenshtein (Edit) και Jaccard αποστάσεις παρέχουν νέες δυνατότητες για την αντιμετώπιση κατηγορικών δεδομένων. Η επιλογή της κατάλληλης συνάρτησης απόστασης εξαρτάται από τον χαρακτήρα των δεδομένων και τις απαιτήσεις του εκάστοτε προβλήματος κατηγοριοποίησης. Επιπλέον, η Μετρική Διαφοράς Τιμής (VDM) [102] και η Μετρική Διαφοράς Ετερογενούς Τιμής (HVDM) [58] είναι εξίσου χρήσιμες για την επεξεργασία κατηγορικών δεδομένων. Η συνεχής ανάπτυξη και προσαρμογή του αλγορίθμου CNN-rule σε νέες συναρτήσεις απόστασης συμβάλλει στην εξέλιξη της επιστήμης της μηχανικής μάθησης και στην αντιμετώπιση πρακτικών προβλημάτων που αφορούν κατηγορικά δεδομένα.

3.3 Κατηγοριοποίηση εγγύτερων γειτόνων και κατηγορικά δεδομένα

Η κατηγοριοποίηση των k-Εγγύτερων (ή κοντινότερων) Γειτόνων (k-NN) είναι μια μέθοδος μηχανικής μάθησης που χρησιμοποιείται για την κατηγοριοποίηση δεδομένων. Όπως αναφέρθηκε στο κεφάλαιο 1, ο κατηγοριοποιητής k-Εγγύτερων Γειτόνων (k-NN) είναι ένας μη παραμετρικός και μη γραμμικός κατηγοριοποιητής, ο οποίος στηρίζεται στην έννοια της

εγγύτητας. Οι παρατηρήσεις του χώρου των χαρακτηριστικών κατηγοριοποιούνται στην κλάση που είναι η πιο κοινή μεταξύ των k πλησιέστερων παρατηρήσεων εκπαίδευσης. Στη κατηγοριοποίηση των k-Εγγύτερων η δυσκολία είναι στην προσέγγιση των γειτόνων της νέας παρατήρησης. Αυτό μπορεί να επιτευχθεί μέσω ενός κριτηρίου που να αποφασίζει αν μια παρατήρηση αποτελεί γείτονα ή όχι της νέας παρατήρησης. Όπως γίνεται εύκολα αντιληπτό, μια τέτοια λειτουργία προϋποθέτει μια έννοια απόστασης. Σε αυτό βοηθάει η επιλογή της κατάλληλης συνάρτησης απόστασης ή αλλιώς μετρικής όπως αναπτύχθηκε στο κεφάλαιο 2, η οποία θα αποφασίζει αν ένα σημείο είναι γειτονικό, κάτι που ικανοποιείται όταν απέχει ελάχιστα ως προς την υποκείμενη μετρική. Στο σημείο αυτό, πρέπει να δοθεί ιδιαίτερη προσοχή στο πως ορίζεται μια γειτονιά, δηλαδή ποια θα είναι η χρήση της κατάλληλης μετρικής. Η πιο δημοφιλής μέτρηση απόστασης είναι η Ευκλείδεια απόσταση [34]. Η περιοχή της γειτονιάς συνήθως ελέγχεται από μια παράμετρο k, η οποία μπορεί να επιλεγεί μέσω της τεχνικής της Διασταυρούμενης Επικύρωσης (Cross Validation), δηλαδή πραγματοποιούνται διάφορες δοκιμές ώστε να καταλήξουμε στον αριθμό γειτόνων που μεγιστοποιεί την ακρίβεια του μοντέλου [44]. Με αυτόν τον τρόπο, το μοντέλο k-NN μπορεί να προβλέψει την τιμή ενός χαρακτηριστικού για μια νέα παρατήρηση με το ελάχιστο δυνατό σφάλμα.

Η κατηγοριοποίηση των k-Εγγύτερων Γειτόνων μπορεί να είναι μια πανίσχυρη μέθοδος και απλή στις διαδικασίες της, ειδικά για εκείνα τα σύνολα δεδομένων που εισέρχονται σε ορισμένους τύπους ειδικής τοπικής δομής. Από την άλλη πλευρά, παρουσιάζονται διάφορα μειονεκτήματα. Στην ενότητα 1.3 έγινε εκτενής αναφορά για τις αδυναμίες του αλγορίθμου κατηγοριοποίησης k-NN. Επιπρόσθετα, η μέθοδος της κατηγοριοποίησης k-NN μπορεί να αποτύχει πλήρως, όταν χρειάζεται να μελετηθούν κατηγορικά δεδομένα. Το συγκεκριμένο πρόβλημα επιλύεται μέσω του μετασχηματισμού δεδομένων ή μέσω του ορισμού μιας κατάλληλης μη μετρικής απόστασης.

Η δημιουργία ενός συνόλου δεδομένων σπάνια γίνεται μόνο για προβλέψεις. Τις περισσότερες φορές τα δεδομένα δεν είναι στη σωστή μορφή ή απαιτούν κάποιους μετασχηματισμούς για να γίνουν πιο χρήσιμα [103]. Οι τεχνικές μετασχηματισμού των δεδομένων περιλαμβάνουν την κατηγορική κωδικοποίηση (categorical encoding).

Κατηγορική κωδικοποίηση (categorical encoding). Συχνά τα σύνολα δεδομένων περιλαμβάνουν και κατηγορικά δεδομένα, δηλαδή τα δεδομένα τα οποία δεν είναι αριθμητικά. Οι περισσότεροι αλγόριθμοι μηχανικής μάθησης δεν δέχονται χαρακτηριστικά τα οποία δεν είναι αριθμητικά. Επομένως, είναι αναγκαία η μετατροπή των κατηγορικών δεδομένων σε αριθμητικά δεδομένα. Οι δύο πιο συνηθισμένες μέθοδοι για την μετατροπή αυτή είναι η Διατάξιμη Κωδικοποίηση (Ordinal Encoding) και η Κωδικοποίηση One-Hot (One-Hot Encoding). Υπάρχουν μεταβλητές των οποίων οι κατηγορίες τους σχετίζονται με κάποιου είδους διάταξη.

Διατάξιμη Κωδικοποίηση (Ordinal Encoding). Η Διατάξιμη Κωδικοποίηση είναι μια τεχνική που χρησιμοποιείται στην επεξεργασία δεδομένων όταν έχουμε κατηγορικές μεταβλητές (categorical variables) που έχουν μια συγκεκριμένη διάταξη ή σειρά μεταξύ των κατηγοριών. Συγκεκριμένα, η Ordinal Encoding μετατρέπει τις κατηγορικές τιμές σε αριθμητικές τιμές, λαμβάνοντας υπόψη τη διάταξη τους. Για παράδειγμα, ας υποθέσουμε ότι έχουμε μια κατηγορική μεταβλητή "Μέγεθος Ρούχων" με τις ακόλουθες κατηγορίες: "Μικρό", "Μεσαίο", "Μεγάλο". Αν χρησιμοποιήσουμε τη διατάξιμη κωδικοποίηση, μπορούμε να αναθέσουμε αριθμητικές τιμές σε αυτές τις κατηγορίες, π.χ. "Μικρό" = 1, "Μεσαίο" = 2, "Μεγάλο" = 3. Για τις μεταβλητές εκείνες που οι κατηγορίες τους δεν υπαινίσσονται κάποιου είδους διάταξη ονομάζονται ονομαστικές (nominal). Χαρακτηριστικό παράδειγμα μιας ονομαστικής μεταβλητής είναι το φύλο του συμμετέχοντα σε μια έρευνα ή ο τόπος κατοικίας. Στις ονομαστικές μεταβλητές εφαρμόζουμε την κωδικοποίηση One-Hot, εφόσον οι κατηγορίες της μεταβλητής δεν είναι πολλές.

Κωδικοποίηση One-Hot (One-Hot Encoding). Η κωδικοποίηση One-Hot είναι μια από τις πιο κοινές μεθόδους κωδικοποίησης στη μηχανική μάθηση. Η συγκεκριμένη κωδικοποίηση χρησιμοποιείται για να μπορούμε να διαχειριστούμε καλύτερα τα κατηγορικά χαρακτηριστικά (categorical features). Η γενική λογική αυτής της μεθοδολογίας είναι, αντί το χαρακτηριστικό να καταλαμβάνει μια στήλη μέσα στο σύνολο δεδομένων, να καταλαμβάνει τόσες στήλες όσες και οι διαφορετικές τιμές που έχει το κατηγορικό χαρακτηριστικό. Η κάθε νέα στήλη πλέον έχει το όνομα μίας τιμής του χαρακτηριστικού. Στη συνέχεια, για κάθε εγγραφή συμπληρώνεται 1 στη νέα στήλη που έχει για όνομα την τιμή που υπήρχε σε αυτή την εγγραφή και σε όλες τις υπόλοιπες στήλες μπαίνει μηδέν. Αυτή η μέθοδος αλλάζει τα κατηγορικά χαρακτηριστικά, τα οποία είναι δύσκολο να αντιληφθούν οι αλγόριθμοι, σε αριθμητικά χαρακτηριστικά, χωρίς να χάσουμε καμία πληροφορία [104]. Για παράδειγμα αν στο αρχικό σύνολο δεδομένων υπάρχει η μεταβλητή “Χρώμα” και έχει τρεις κατηγορίες: “Κόκκινο”, “Κίτρινο”, χαρακτηριστικού σε αριθμητικό “Πράσινο”, τότε με την εφαρμογή της προσέγγισης του one-hot encoding γίνεται ο μετασχηματισμός που φαίνεται στην Εικόνα 10 [104].

Χρώμα
Κόκκινο
Κόκκινο
Κίτρινο
Πράσινο
Κίτρινο

Κόκκινο	Κίτρινο	Πράσινο
1	0	0
1	0	0
0	1	0
0	0	1
0	1	0

Εικόνα 10: Μετασχηματισμός κατηγορικής μεταβλητής με την μέθοδο one-hot encoding

Μέθοδος μη μετρικής απόστασης

Ανάλογα με τον τύπο της μετρικής που θέλουμε να χρησιμοποιήσουμε, προκύπτουν οι ακόλουθες παραλλαγές του κατηγοριοποιητή k-NN για κατηγορικά δεδομένα:

1. Ο κατηγοριοποιητής **k-NN Hamming:** Σε αυτήν την παραλλαγή, χρησιμοποιούμε τη μετρική απόστασης Hamming για τον υπολογισμό της απόστασης μεταξύ διανυσμάτων κατηγορικών μεταβλητών. Η απόσταση Hamming μετρά τον αριθμό των διαφορετικών στοιχείων σε διανύσματα δυαδικών τιμών.

Αλγόριθμος k-NN Hamming

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

using namespace std;

struct TrainItem{
  int classAttr;
  string *attr;
};

struct TestItem{
  int classAttr;
  string *attr;
  int pred;
};

int convKNN;
int NumberOfAttributes;
int classes;

int findClassConv(int votingArray[], TrainItem trainData[], int pos[], int max, int p, int U){
  int i, j, cb=0;
  int *snov;

  snov = new int[classes];
```

```
  for (i=0; i<classes; i++){
    if (votingArray[i] == max){
      snov[cb]= i;
      cb++;
    }
  }
  if (cb == 1){
    return p;
  }
  else{
    for (i=0; i<U; i++){
      for (j=0; j<cb; j++){
        if (trainData[pos[i]].classAttr == snov[j]){
          return trainData[pos[i]].classAttr;
        }
      }
    }
  }
  return -1;
}

int findMax(int votingArray[]){
  int max, p, i;
  max=votingArray[0];
  p=0;
  for (i=1; i<classes; i++){
    if (votingArray[i]>max){
      max=votingArray[i];
      p=i;
    }
  }
  return p;
}

void readTestDataFile(TestItem testData[], char fileName[], int n){
  int i, j;
  ifstream dat;
  dat.open(fileName);

  i=0;
  for (i=0; i<n; i++){
    testData[i].attr = new string[NumberOfAttributes];
    for (j=0; j<NumberOfAttributes; j++){
```

```
      dat>>testData[i].attr[j];
    }
    dat>>testData[i].classAttr;
  }
  dat.close();
}

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

  ifstream dat;
  dat.open(fileName);
  classes=0;
  bool fl;

  for (i=0; i<n; i++){
    trainData[i].attr = new string[NumberOfAttributes];
    for (j=0; j<NumberOfAttributes; j++){
      getline(dat, trainData[i].attr[j], '\t');
    }
    dat>>trainData[i].classAttr ;
    fl = false;

    for (j=0; j<i; j++){
      if (trainData[i].classAttr == trainData[j].classAttr){
        fl = true;
        break;
      }
    }
    if (!fl){
      classes++;
    }
  }

  dat.close();
}

int conventialKNNSearch(TrainItem trainData[], TestItem queryPoint, int n){
  int h, g, k, j, pos, *posMinKArray;
  float u,x, *min;
  float *dist;
  int sum=0;

  min = new float[convKNN];
```

```
posMinKArray = new int[convKNN];
dist = new float[n];

for (h=0; h<convKNN; h++){
  min[h]=999999999999999999;
  posMinKArray[h]=-1;
}

for (k=0; k<n; k++){
  sum=0;
//Υπολογισμός Hamming απόστασης μεταξύ των πολλαπλών συμβολοσειρών
  for (j=0; j<NumberOfAttributes; j++){
     if (queryPoint.attr[j].compare(trainData[k].attr[j]) != 0) {
                  sum ++;
    }
  }
  //cout<<sum<<endl;
  dist[k]=sqrt((float)sum);

  for (h=0; h<convKNN; h++) {
    if (dist[k]<min[h]){
      for (g=convKNN-1; g>h; g--){
        min[g]=min[g-1];
        posMinKArray[g]=posMinKArray[g-1];
      }
      min[h]=dist[k];
      posMinKArray[h]=k;
      break;
    }
  }
}

free(dist);

if (convKNN==1){
  return trainData[posMinKArray[0]].classAttr;
}
else{
  int max, *votingArray;

  votingArray = new int[classes];

  for (k=0; k<classes; k++){
    votingArray[k]=0;
```

```
    }
    for (k=0; k<convKNN; k++){
      votingArray[trainData[posMinKArray[k]].classAttr]++;
    }
    pos=findMax(votingArray);
    max=votingArray[pos];

    return findClassConv(votingArray, trainData, posMinKArray, max, pos,
convKNN);
  }
}

int countLinesAttrs(char fileName[], int &lines, int &attrs){
  int i, c;

  ifstream dat;
  dat.open(fileName);

  if (!dat){
    return 1;
  }

  string lline;

  getline(dat, lline);
  c=0;
  for (i=0; i<lline.length(); i++){
    if (lline[i] == '\t'){
      c++;
    }
  }
  dat.close();
  attrs = c;

  dat.open(fileName);
  i=0;
  while (!dat.eof()){
    getline(dat, lline);
    i++;
  }

  lines = i-1;

  dat.close();
```

```
  return 0;
}

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

  static TrainItem *trainData;
  static TestItem *testData;

  unsigned long long int comp;

  int i, n, n2, f;
  int correctclassified;
  double acc;

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

  if (!isdigit(*argv[3])){
    cout<<"ERROR. parameter must be numeric"<<endl;
    return 1;
  }

  convKNN=atoi(argv[3]);

  char *fileName = argv[1];
  char *fileName2 = argv[2];

  if (countLinesAttrs(fileName,n,NumberOfAttributes)){
    cout<<"File "<<fileName<<" dos not exist"<<endl;
    return 1;
  }

  if (countLinesAttrs(fileName2,n2,NumberOfAttributes)){
    cout<<"File "<<fileName2<<" does not exist"<<endl;
    return 1;
  }

  trainData = new TrainItem[n];
  readTrainData(trainData, fileName, n);

  testData = new TestItem[n2];
  readTestDataFile(testData, fileName2 ,n2);
```

```
  cout<<"Classes: "<<classes<<endl;
  cout<<"Attributes: "<<NumberOfAttributes<<endl;

  correctclassified=0;
  for (i=0; i<n2; i++){
    testData[i].pred=conventialKNNSearch(trainData, testData[i], n);
    if (testData[i].pred == testData[i].classAttr){
      correctclassified++;
    }
  }

  free(trainData);
  free(testData);

  acc = (double)correctclassified/(double)(n2)*100;

  comp = n * n2;

  cout<<"Accuracy: "<<acc<<" Computations: "<<comp<<endl;
}
```

2. Ο κατηγοριοποιητής **k-NN Leveshtein:** Σε αυτήν την παραλλαγή, χρησιμοποιούμε τη μετρική απόστασης Edit (ή Levenshtein) για τον υπολογισμό της απόστασης μεταξύ διανυσμάτων κατηγορικών μεταβλητών. Η απόσταση Edit μετρά τον αριθμό των επεξεργασιών που απαιτούνται για να μετατραπεί ένα διάνυσμα σε ένα άλλο.

Αλγόριθμος k-NN Leveshtein

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

using namespace std;

struct TrainItem{
```

```
  int classAttr;
  string *attr;
};

struct TestItem{
  int classAttr;
  string *attr;
  int pred;
};

int convKNN;
int NumberOfAttributes;
int classes;

int findClassConv(int votingArray[], TrainItem trainData[], int pos[], int max, int p, int U){
  int i, j, cb=0;
  int *snov;

  snov = new int[classes];

  for (i=0; i<classes; i++){
    if (votingArray[i] == max){
      snov[cb]= i;
      cb++;
    }
  }
  if (cb == 1){
    return p;
  }
  else{
    for (i=0; i<U; i++){
      for (j=0; j<cb; j++){
        if (trainData[pos[i]].classAttr == snov[j]){
          return trainData[pos[i]].classAttr;
        }
      }
    }
  }
  return -1;
}

int findMax(int votingArray[]){
  int max, p, i;
```

```
  max=votingArray[0];
  p=0;
  for (i=1; i<classes; i++){
    if (votingArray[i]>max){
      max=votingArray[i];
      p=i;
    }
  }
  return p;
}

void readTestDataFile(TestItem testData[], char fileName[], int n){
  int i, j;
  ifstream dat;
  dat.open(fileName);

  i=0;
  for (i=0; i<n; i++){
    testData[i].attr = new string[NumberOfAttributes];
    for (j=0; j<NumberOfAttributes; j++){
      dat>>testData[i].attr[j];
    }
    dat>>testData[i].classAttr;
  }
  dat.close();
}

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

  ifstream dat;
  dat.open(fileName);
  classes=0;
  bool fl;

  for (i=0; i<n; i++){
    trainData[i].attr = new string[NumberOfAttributes];
    for (j=0; j<NumberOfAttributes; j++) {
      getline(dat, trainData[i].attr[j], '\t');
    }
    dat>>trainData[i].classAttr ;
    fl = false;

  for (j=0; j<i; j++){
```

```
      if (trainData[i].classAttr == trainData[j].classAttr){
        fl = true;
        break;
     }
   }
   if (!fl){
     classes++;
   }
  }
  dat.close();
}

int LevenshteinDistance(string s1, string s2) {
    const int m = s1.length();
    const int n = s2.length();

    int dp[m + 1][n + 1];

    for (int i = 0; i <= m; i++) {
        for (int j = 0; j <= n; j++) {
            if (i == 0) {
                dp[i][j] = j;
            } else if (j == 0) {
                dp[i][j] = i;
            } else if (s1[i - 1] == s2[j - 1]) {
                dp[i][j] = dp[i - 1][j - 1];
            } else {
                dp[i][j] = 1 + min(dp[i - 1][j], min(dp[i][j - 1], dp[i - 1][j - 1]));
            }
        }
    }

    return dp[m][n];
}

int conventialKNNSearch(TrainItem trainData[], TestItem queryPoint, int n){
   int h, g, k, j, pos, *posMinKArray;
   double u,x, *min;
   static double *dist = new double[n];
   double sum=0;

   min = new double[convKNN];
   posMinKArray = new int[convKNN];
   dist = new double[n];
```

```
for (h=0; h<convKNN; h++){
  min[h]=999999999999999999;
  posMinKArray[h]=-1;
}

for (k=0; k<n; k++){
  sum=0;
//Υπολόγισμος Levenshtein απόστασης μεταξύ των πολλαπλών συμβολοσειρών
  for (j=0; j<NumberOfAttributes; j++){
                sum += LevenshteinDistance(queryPoint.attr[j], trainData[k].attr[j]);
  }
  dist[k]=sqrt(sum);

  for (h=0; h<convKNN; h++) {
    if (dist[k]<min[h]){
      for (g=convKNN-1; g>h; g--){
        min[g]=min[g-1];
        posMinKArray[g]=posMinKArray[g-1];
      }
      min[h]=dist[k];
      posMinKArray[h]=k;
      break;
    }
  }
}

free(dist);

if (convKNN==1){
  return trainData[posMinKArray[0]].classAttr;
}
else{
  int max, *votingArray;

  votingArray = new int[classes];

  for (k=0; k<classes; k++){
    votingArray[k]=0;
  }
  for (k=0; k<convKNN; k++){
    votingArray[trainData[posMinKArray[k]].classAttr]++;
  }
  pos=findMax(votingArray);
```

```
    max=votingArray[pos];

    return findClassConv(votingArray, trainData, posMinKArray, max, pos,
convKNN);
  }
}

int countLinesAttrs(char fileName[], int &lines, int &attrs){
  int i, c;

  ifstream dat;
  dat.open(fileName);

  if (!dat){
    return 1;
  }

  string lline;

  getline(dat, lline);
  c=0;
  for (i=0; i<lline.length(); i++){
    if (lline[i] == '\t'){
      c++;
    }
  }
  dat.close();
  attrs = c;

  dat.open(fileName);
  i=0;
  while (!dat.eof()){
    getline(dat, lline);
    i++;
  }

  lines = i-1;

  dat.close();
  return 0;
}

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

```
static TrainItem *trainData;
static TestItem *testData;

unsigned long long int comp;

int i, n, n2, f;
int correctclassified;
double acc;

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

if (!isdigit(*argv[3])){
  cout<<"ERROR. parameter must be numeric"<<endl;
  return 1;
}

convKNN=atoi(argv[3]);

char *fileName = argv[1];
char *fileName2 = argv[2];

if (countLinesAttrs(fileName,n,NumberOfAttributes)){
  cout<<"File "<<fileName<<" dos not exist"<<endl;
  return 1;
}

if (countLinesAttrs(fileName2,n2,NumberOfAttributes)){
  cout<<"File "<<fileName2<<" does not exist"<<endl;
  return 1;
}

trainData = new TrainItem[n];
readTrainData(trainData, fileName, n);

testData = new TestItem[n2];
readTestDataFile(testData, fileName2 ,n2);

cout<<"Classes: "<<classes<<endl;
cout<<"Attributes: "<<NumberOfAttributes<<endl;

correctclassified=0;
```

```
  for (i=0; i<n2; i++){
    testData[i].pred=conventialKNNSearch(trainData, testData[i], n);
    if (testData[i].pred == testData[i].classAttr){
      correctclassified++;
    }
  }

  free(trainData);
  free(testData);

  acc = (double)correctclassified/(double)(n2)*100;

  comp = n * n2;

  cout<<"Accuracy: "<<acc<<" Computations: "<<comp<<endl;
}
```

3. Ο κατηγοριοποιητής **k-NN Jaccard:** Σε αυτήν την παραλλαγή, χρησιμοποιούμε τη μετρική απόστασης Jaccard για τον υπολογισμό της ομοιότητας μεταξύ διανυσμάτων κατηγορικών μεταβλητών. Η απόσταση Jaccard μετρά την ομοιότητα μεταξύ δύο συνόλων.

Αλγόριθμος k-NN Jaccard

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

using namespace std;

struct TrainItem{
  int classAttr;
  string *attr;
};

struct TestItem{
```

```
  int classAttr;
  string *attr;
  int pred;
};

int convKNN;
int NumberOfAttributes;
int classes;

int findClassConv(int votingArray[], TrainItem trainData[], int pos[], int max, int p, int U){
  int i, j, cb=0;
  int *snov;

  snov = new int[classes];

  for (i=0; i<classes; i++){
    if (votingArray[i] == max){
      snov[cb]= i;
      cb++;
    }
  }
  if (cb == 1){
    return p;
  }
  else{
    for (i=0; i<U; i++){
      for (j=0; j<cb; j++){
        if (trainData[pos[i]].classAttr == snov[j]){
          return trainData[pos[i]].classAttr;
        }
      }
    }
  }
  return -1;
}

int findMax(int votingArray[]){
  int max, p, i;
  max=votingArray[0];
  p=0;
  for (i=1; i<classes; i++){
    if (votingArray[i]>max){
      max=votingArray[i];
```

```
        p=i;
      }
    }
    return p;
}

void readTestDataFile(TestItem testData[], char fileName[], int n){
    int i, j;
    ifstream dat;
    dat.open(fileName);

    i=0;
    for (i=0; i<n; i++){
      testData[i].attr = new string[NumberOfAttributes];
      for (j=0; j<NumberOfAttributes; j++){
        dat>>testData[i].attr[j];
      }
      dat>>testData[i].classAttr;
    }
    dat.close();
}

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

    ifstream dat;
    dat.open(fileName);
    classes=0;
    bool fl;

    for (i=0; i<n; i++){
      trainData[i].attr = new string[NumberOfAttributes];
      for (j=0; j<NumberOfAttributes; j++) {
        getline(dat, trainData[i].attr[j], '\t');
      }
      dat>>trainData[i].classAttr ;
      fl = false;

    for (j=0; j<i; j++){
      if (trainData[i].classAttr == trainData[j].classAttr){
        fl = true;
        break;
      }
    }
```

```
  if (!fl){
    classes++;
  }
 }
 dat.close();
}

double jaccardDistance(string s1, string s2) {
   size_t m = s1.length();
   size_t n = s2.length();
   size_t intersection = 0;
   size_t unionSize = m + n;

   //Check for empty strings
   if (m == 0 && n == 0)
      return 0.0;

   //Iterate over characters in s1
   for (size_t i = 0; i < m; i++) {
      // Check if character exists in s2
      if (strchr(s2.c_str(), s1[i]) != NULL) {
         intersection++;
         unionSize--;
      }
   }

   //Iterate over characters in s2
   for (size_t i = 0; i < n; i++) {
   //Check if character exists in s1
      if (strchr(s1.c_str(), s2[i]) == NULL) {
         unionSize++;
      }
   }

   //Calculate Jaccard distance
         double jaccardSimilarity = (double)(intersection) / unionSize;
   double jaccardDistance = 1.0 - jaccardSimilarity;

   return jaccardDistance;
}

int conventialKNNSearch(TrainItem trainData[], TestItem queryPoint, int n){
  int h, g, k, j, pos, *posMinKArray;
  double u,x, *min;
```

```
static double *dist = new double[n];
double sum=0;

min = new double[convKNN];
posMinKArray = new int[convKNN];
dist = new double[n];

for (h=0; h<convKNN; h++){
  min[h]=999999999999999999;
  posMinKArray[h]=-1;
}

for (k=0; k<n; k++){
  sum=0;
//Υπολογισμός Jaccard απόστασης ανάμεσα στα σύνολα γνωρισμάτων
 for (j=0; j<NumberOfAttributes; j++){
                sum += jaccardDistance(queryPoint.attr[j], trainData[k].attr[j]);
  }
  dist[k]= sum/NumberOfAttributes;

  for (h=0; h<convKNN; h++) {
    if (dist[k]<min[h]){
      for (g=convKNN-1; g>h; g--){
        min[g]=min[g-1];
        posMinKArray[g]=posMinKArray[g-1];
      }
      min[h]=dist[k];
      posMinKArray[h]=k;
      break;
    }
  }
}

free(dist);

if (convKNN==1){
  return trainData[posMinKArray[0]].classAttr;
}
else{
  int max, *votingArray;

  votingArray = new int[classes];

  for (k=0; k<classes; k++){
```

```
      votingArray[k]=0;
    }
    for (k=0; k<convKNN; k++){
      votingArray[trainData[posMinKArray[k]].classAttr]++;
    }
    pos=findMax(votingArray);
    max=votingArray[pos];

    return findClassConv(votingArray, trainData, posMinKArray, max, pos,
convKNN);
  }
}

int countLinesAttrs(char fileName[], int &lines, int &attrs){
  int i, c;

  ifstream dat;
  dat.open(fileName);

  if (!dat){
    return 1;
  }

  string lline;

  getline(dat, lline);
  c=0;
  for (i=0; i<lline.length(); i++){
    if (lline[i] == '\t'){
      c++;
    }
  }
  dat.close();
  attrs = c;

  dat.open(fileName);
  i=0;
  while (!dat.eof()){
    getline(dat, lline);
    i++;
  }

  lines = i-1;
```

```
  dat.close();
  return 0;
}

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

  static TrainItem *trainData;
  static TestItem *testData;

  unsigned long long int comp;

  int i, n, n2, f;
  int correctclassified;
  double acc;

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

  if (!isdigit(*argv[3])){
    cout<<"ERROR. parameter must be numeric"<<endl;
    return 1;
  }

  convKNN=atoi(argv[3]);

  char *fileName = argv[1];
  char *fileName2 = argv[2];

  if (countLinesAttrs(fileName,n,NumberOfAttributes)){
    cout<<"File "<<fileName<<" dos not exist"<<endl;
    return 1;
  }

  if (countLinesAttrs(fileName2,n2,NumberOfAttributes)){
    cout<<"File "<<fileName2<<" does not exist"<<endl;
    return 1;
  }

  trainData = new TrainItem[n];
  readTrainData(trainData, fileName, n);

  testData = new TestItem[n2];
```

```
  readTestDataFile(testData, fileName2 ,n2);

  cout<<"Classes: "<<classes<<endl;
  cout<<"Attributes: "<<NumberOfAttributes<<endl;

  correctclassified=0;
  for (i=0; i<n2; i++){
    testData[i].pred=conventialKNNSearch(trainData, testData[i], n);
    if (testData[i].pred == testData[i].classAttr){
      correctclassified++;
    }
  }

  free(trainData);
  free(testData);

  acc = (double)correctclassified/(double)(n2)*100;

  comp = n * n2;

  cout<<"Accuracy: "<<acc<<" Computations: "<<comp<<endl;
}
```

Συνοψίζοντας, κάθε παραλλαγή του κατηγοριοποιητή k-Εγγύτερων Γειτόνων (k-NN) χρησιμοποιεί διαφορετική μετρική για τον υπολογισμό της απόστασης μεταξύ κατηγορικών δεδομένων. Ο κατηγοριοποιητής K-NN Hamming χρησιμοποιεί τη μη μετρική Hamming, που μετρά τον αριθμό των διαφορετικών στοιχείων σε διανύσματα δυαδικών τιμών. Ο κατηγοριοποιητής K-NN Levenshtein χρησιμοποιεί τη μη μετρική Levenshtein για τον υπολογισμό της απόστασης, μετρώντας τον αριθμό των επεξεργασιών που απαιτούνται για να μετατραπεί ένα διάνυσμα σε ένα άλλο. Τέλος, ο κατηγοριοποιητής K-NN Jaccard χρησιμοποιεί τη μη μετρική Jaccard, που μετρά την ομοιότητα μεταξύ διανυσμάτων κατηγορικών μεταβλητών με βάση την ομοιότητα των συνόλων τους. Κάθε παραλλαγή προσφέρει διαφορετική προσέγγιση για την κατηγοριοποίηση των κατηγορικών δεδομένων και μπορεί να επιλεγεί ανάλογα με τα χαρακτηριστικά του προβλήματος και την απόδοση που επιδιώκουμε.

4

Πειραματική μελέτη

Σε αυτό το κεφάλαιο παρουσιάζεται η πειραματική μελέτη που εκπονήθηκε στα πλαίσια της παρούσας διπλωματικής εργασίας. Η πειραματική μελέτη βασίστηκε σε 8 γνωστά σύνολα δεδομένων. Παρουσιάζονται όλες οι απαραίτητες ρυθμίσεις και οι παράμετροι που χρησιμοποιήθηκαν για την εκτέλεση των πειραμάτων. Επιπρόσθετα, παρουσιάζονται και αναλύονται τα πειραματικά αποτελέσματα που προέκυψαν και επιχειρείται μία σύγκριση μεταξύ των προτεινόμενων μεθόδων με τις διαφορετικές προσεγγίσεις κατηγοριοποίησης που βασίζονται στην αναζήτηση εγγύτερων γειτόνων. Η σύγκριση πραγματοποιήθηκε ως προς την ακρίβεια κατηγοριοποίησης και το ποσοστό μείωσης του πληθυσμού των δεδομένων που επιτυγχάνουν.

4.1 Πειραματικές ρυθμίσεις

Οι αλγόριθμοι που δημιουργήθηκαν για τις ανάγκες της διπλωματικής εργασίας γράφτηκαν στην γλώσσα προγραμματισμού C++ και συμμετείχαν στα πειράματα μας με τις παρακάτω συντομογραφίες:

K-NN Hamming: ο κατηγοριοποιητής k-NN με εφαρμογή σε σύνολο δεδομένων που περιέχει κατηγορικά χαρακτηριστικά κάνοντας χρήση της μη μετρικής Hamming.

K-NN Levenshtein: ο κατηγοριοποιητής k-NN με εφαρμογή σε σύνολο δεδομένων που περιέχει κατηγορικές μεταβλητές με την μέθοδο της μη μετρικής Levenshtein.

K-NN Jaccard: ο κατηγοριοποιητής k-NN με εφαρμογή σε σύνολο δεδομένων που περιέχει κατηγορικά χαρακτηριστικά κάνοντας χρήση της μη μετρικής Jaccard.

CNN Hamming: η προτεινόμενη παραλλαγή του αλγορίθμου CNN-rule με εφαρμογή σε σύνολο δεδομένων που περιέχει κατηγορικές μεταβλητές με την μέθοδο της συνάρτησης απόστασης Hamming.

CNN Levenshtein: η προτεινόμενη παραλλαγή του αλγορίθμου CNN-rule με εφαρμογή σε σύνολο δεδομένων που περιέχει κατηγορικά χαρακτηριστικά κάνοντας χρήση της συνάρτησης απόστασης Levenshtein.

CNN Jaccard: η προτεινόμενη παραλλαγή του αλγορίθμου CNN-rule με εφαρμογή σε σύνολο δεδομένων που περιέχει κατηγορικές μεταβλητές με την μέθοδο της συνάρτησης απόστασης Jaccard.

Ο υπολογιστής στον οποίο εκτελέστηκαν οι αλγόριθμοι και όλα τα πειράματα ήταν ένας φορητός υπολογιστής, ο οποίος χρησιμοποιούσε έναν i5-Intel επεξεργαστή, μνήμη RAM 12GB και λειτουργικό σύστημα Windows 10.

Για τον έλεγχο της απόδοσης των αλγορίθμων που μελετήσαμε καθώς και για τον υπολογισμό της ακρίβειας τους, χρησιμοποιήθηκαν οχτώ σύνολα δεδομένων που διανεμήθηκαν τα περισσότερα από το αποθετήριο KEEL [105] καθώς και από το αποθετήριο UC Irvine Machine Learning Repository (UCI) [106] και παρουσιάζονται συνοπτικά στον Πίνακα 1. Οι τιμές που μετρήθηκαν είναι η ακρίβεια (ACC - Accuracy) και το ποσοστό μείωσης (RR – Reduction Rate). Οι τιμές ACC και RR μετρήθηκαν ως ποσοστά (%). Για λόγους σύγκρισης, χρησιμοποιούμε την συμβατική κατηγοριοποίηση k-NN με ορισμό του k στο ένα (k=1).

Σε όλα τα σύνολα δεδομένων (datasets) που χρησιμοποιήθηκαν για την εκτέλεση των πειραμάτων έχει προηγηθεί η διαδικασία του μετασχηματισμού δεδομένων. Συγκεκριμένα, πραγματοποιήσαμε μία επεξεργασία κανονικοποίησης (normalization) κατά την οποία όλες οι τιμές του συνόλου δεδομένων έχουν μετατραπεί στην κλίματα [0 - 1].

Περιγραφή Datasets

Στο σημείο αυτό θα γίνει μία σύντομη αναφορά στο τι αντιπροσωπεύει κάθε ένα σύνολο δεδομένων που έχουμε χρησιμοποιήσει στα πειράματα μας. Αλφαβητικά λοιπόν είναι τα εξής:

Adult Data Set [107]: Αυτό το σύνολο δεδομένων προήλθε το 1994 από τα απογραφικά δεδομένα των Ηνωμένων Πολιτειών. Περιλαμβάνει συνεχείς και ονομαστικές

(κατηγορικές) μεταβλητές που περιγράφουν κοινωνικές πληροφορίες (ηλικία, εθνικότητα, φύλο, οικογενειακή κατάσταση κ.λπ.) για τους εγγεγραμμένους πολίτες.

Car Evaluation Data Set [108]: Αυτό το σύνολο δεδομένων προέκυψε από ένα απλό ιεραρχικό μοντέλο απόφασης. Το μοντέλο αξιολογεί αυτοκίνητα βάσει έξι χαρακτηριστικών εισόδου: αγορά, συντήρηση, πόρτες, άτομα, χώρος αποσκευών και ασφάλεια.

Chess (King-Rook vs. King) Data Set [109]: Αυτό το σύνολο δεδομένων αναπαριστά για το τέλος του παιχνιδιού σκακιού τις θέσεις στο ταμπλό με τον λευκό βασιλιά, τον λευκό πύργο και τον μαύρο βασιλιά. Ο στόχος είναι να προβλεφθεί ο βέλτιστος αριθμός επαναλήψεων που απαιτούνται για να κερδίσει ο παίκτης με τον λευκό στρατό. Εάν αυτό δεν είναι δυνατόν και το παιχνίδι διαρκέσει περισσότερες από δεκαέξι επαναλήψεις, τότε το αποτέλεσμα μπορεί να είναι ισοπαλία.

Chess (King-Rook vs. King-Pawn) Data Set [110]: Ένα σύνολο δεδομένων που αναπαριστά για το τέλος του παιχνιδιού σκακιού, όπου ένα πιόνι στη θέση a7 απέχει ένα τετράγωνο από τη βασίλισσα. Ο στόχος είναι να προβλεφθεί εάν ο παίκτης με τον λευκό στρατό μπορεί να κερδίσει ή όχι.

Connect-4 Data Set [111]: Αυτή η βάση δεδομένων περιλαμβάνει όλες τις νόμιμες θέσεις στο παιχνίδι του "Connect-4" για ένα πλέγμα 6x7, στο οποίο κανένας από τους παίκτες δεν έχει κερδίσει ακόμη, και η επόμενη κίνηση δεν είναι υποχρεωτική. Έτσι, κάθε χαρακτηριστικό περιέχει μια ονομαστική τιμή που περιγράφει εάν μια δεδομένη θέση είναι κενή ή αν έχει καταληφθεί από έναν παίκτη. Ο στόχος είναι να προβλεφθεί ποιοι παίκτες είναι πιθανό να κερδίσουν τον αγώνα.

Fatality Analysis Reporting System Data Set [112]: Αυτό το σύνολο δεδομένων είναι μια συλλογή στατιστικών στοιχείων σχετικά με τροχαία ατυχήματα που δημιουργήθηκε από το Εθνικό Κέντρο Στατιστικής και Ανάλυσης των Ηνωμένων Πολιτειών Αμερικής. Το συγκεκριμένο σύνολο δεδομένων περιέχει πληροφορίες για όλα τα άτομα που ενεπλάκησαν σε τροχαία ατυχήματα στις Ηνωμένες Πολιτείες Αμερικής κατά τη διάρκεια του έτους 2001, όπου η πλειονότητα των χαρακτηριστικών του αναπαρίστανται με ονομαστικές τιμές. Το χαρακτηριστικό class περιγράφει το επίπεδο τραυματισμού που υπέστησαν.

Nursery Data Set [113]: Αυτή η βάση δεδομένων προήλθε από ένα ιεραρχικό μοντέλο απόφασης που αρχικά αναπτύχθηκε για να κατατάσσει αιτήσεις για νηπιαγωγεία. Χρησιμοποιήθηκε για αρκετά χρόνια τη δεκαετία του 1980, όταν υπήρχαν υπερβολικές εγγραφές σε αυτά τα νηπιαγωγεία στη Λιουμπλιάνα, την πρωτεύουσα της Σλοβενίας και οι απορριφθείσες αιτήσεις χρειάζονταν συχνά μια αντικειμενική εξήγηση. Ο στόχος είναι η κατάταξη των αιτήσεων για ένα συγκεκριμένο νηπιαγωγείο.

Molecular Biology (Splice-junction Gene Sequences) Data Set [114]: Τα σημεία ματίσματος (splice junctions) είναι σημεία σε μια ακολουθία DNA στα οποία το "περιττό"

DNA αφαιρείται κατά τη διαδικασία δημιουργίας πρωτεϊνης σε ανώτερους οργανισμούς. Το πρόβλημα που τίθεται σε αυτό το σύνολο δεδομένων είναι να αναγνωριστούν, δεδομένης μιας ακολουθίας DNA, τα όρια μεταξύ των εξόνιων (τα τμήματα της ακολουθίας DNA που διατηρούνται μετά τη διαδικασία ματίσματος) και των ιντρονίων (τα τμήματα της ακολουθίας DNA που αφαιρούνται με τη διαδικασία ματίσματος).

Συνοπτική παρουσίαση των συνόλων δεδομένων με τις βασικές πληροφορίες τους φαίνεται στον Πίνακα 1 που ακολουθεί.

Σύνολο Δεδομένων		**Μέγεθος**	**Χαρακτηριστικά**	**Κλάσεις**
1.	Adult	48842	14	2
2.	Car Evaluation	1728	6	4
3.	Chess (Kr-vs-k)	28056	6	17
4.	Chess	3196	36	2
5.	Connect-4	67557	42	3
6.	Fatality Analysis Reporting System	100968	29	8
7.	Nursery	12690	8	5
8.	Molecular Biology	3190	60	3

Πίνακας 1: Συνοπτική παρουσίαση των συνόλων δεδομένων

4.2 Πειραματικές μετρήσεις και αποτελέσματα

Η απόδοση του κατηγοριοποιητή k-NN συγκρίθηκε με τις βελτιστοποιημένες παραλλαγές του CNN-rule που προτείναμε, τον CNN Hamming, CNN Levenshtein και τον CNN Jaccard, χρησιμοποιώντας τα 8 σύνολα κατηγορικών δεδομένων που παρουσιάστηκαν παραπάνω. Μετρήθηκε η ακρίβεια του κατηγοριοποιητή στα αρχικά σύνολα, καθώς και σε σύνολα δεδομένων όπου έχει εφαρμοστεί η τεχνική μείωσης του πληθυσμού (Data Reduction Technique). Τα αποτελέσματα που προέκυψαν παρουσιάζονται στον Πίνακα 2. Παράλληλα μετρήθηκε και το ποσοστό μείωσης που επιτυγχάνεται μετά από κάθε πειραματική μέτρηση.

		K-NN Hamming	K-NN Levenshtein	K-NN Jaccard	CNN Hamming	CNN Levenshtein	CNN Jaccard
Adult	ACC	78.293	79.252	79.091	75.626	76.035	75.673
	RR	0	0	0	64.482	64.423	62.476
Car Evaluation	ACC	72.056	80.843	78.558	70.650	82.073	75.922
	RR	0	0	0	65.427	66.032	66.378
Chess (kr-vs-k)	ACC	42.573	37.398	44.604	40.898	37.625	43.480
	RR	0	0	0	33.510	33.430	32.984
Chess	ACC	89.184	89.848	89.373	87.571	88.330	90.512
	RR	0	0	0	76.962	77.523	77.289
Connect-4	ACC	67.312	66.222	67.456	62.876	62.647	62.737
	RR	0	0	0	54.074	54.352	50.576
Fatality Analysis Reporting System	ACC	72.870	73.464	74.224	70.577	72.831	72.306
	RR	0	0	0	63.166	63.894	62.081
Nursery	ACC	77.169	92.888	85.964	75.695	91.695	86.198
	RR	0	0	0	69.702	79.721	71.602
Molecular Biology	ACC	75.000	75.760	74.809	72.053	72.148	72.053
	RR	0	0	0	58.661	58.614	57.724

Πίνακας 2: Αποτελέσματα πειραματικών μετρήσεων

Αναλύουμε τις πειραματικές μετρήσεις από τον παραπάνω πίνακα για κάθε σύνολο δεδομένων. Στο Adult σύνολο δεδομένων, οι αλγόριθμοι K-NN φαίνεται να έχουν καλύτερη απόδοση σε σχέση με τους αντίστοιχους CNN για όλες τις μετρήσεις. Ωστόσο ο αλγόριθμος CNN Levenshtein έχει την υψηλότερη ακρίβεια, ενώ οι άλλοι δύο αλγόριθμοι CNN έχουν παρόμοια απόδοση. Στο Car Evaluation σύνολο δεδομένων, ο αλγόριθμος CNN Levenshtein έχει και πάλι την υψηλότερη ακρίβεια, παρουσιάζοντας σημαντικά καλύτερη απόδοση σε σύγκριση με τους άλλους δύο αλγορίθμους CNN.

Στο Chess (kr-vs-k) σύνολο δεδομένων, κανένας αλγόριθμος δεν εμφανίζει ιδιαίτερα υψηλή ακρίβεια. Ωστόσο, ο αλγόριθμος K-NN Jaccard έχει την υψηλότερη ακρίβεια. Ο αλγόριθμος CNN Jaccard έχει την υψηλότερη ακρίβεια σε σύγκριση με τους άλλους δύο αλγορίθμους CNN, αν και οι διαφορές μεταξύ των αλγορίθμων δεν είναι μεγάλες. Στο Chess σύνολο δεδομένων, ο αλγόριθμος CNN Jaccard φαίνεται να έχει την υψηλότερη ακρίβεια από όλους τους αλγορίθμους. Στο Connect-4 σύνολο δεδομένων, ο αλγόριθμος K-NN Jaccard έχει την υψηλότερη ακρίβεια. Οι τρεις αλγόριθμοι CNN έχουν παρόμοια απόδοση, χωρίς σημαντικές διαφορές. Στο Fatality Analysis Reporting System σύνολο δεδομένων, ο αλγόριθμος K-NN Jaccard φαίνεται να έχει την υψηλότερη ακρίβεια. Σε ότι αφορά τους αλγορίθμους CNN, ο αλγόριθμος CNN Levenshtein έχει την υψηλότερη ακρίβεια. Στο Nursery σύνολο δεδομένων, ο αλγόριθμος K-NN Levenshtein έχει την υψηλότερη ακρίβεια.

Ωστόσο, ο αλγόριθμος CNN Levenshtein έχει την υψηλότερη ακρίβεια σε σύγκριση με τους άλλους δύο αλγορίθμους CNN. Στο Molecular Biology σύνολο δεδομένων, ο αλγόριθμος K-NN Levenshtein φαίνεται να έχει καλύτερη απόδοση σε σχέση με τους υπόλοιπους αλγορίθμους CNN για όλες τις μετρήσεις. Ο αλγόριθμος CNN Levenshtein έχει υψηλότερη ακρίβεια σε σύγκριση με τους άλλους δύο αλγορίθμους CNN.

Αφού διερευνήσαμε τις πειραματικές μετρήσεις, παρατηρήσαμε ότι ο αλγόριθμος CNN Levenshtein εμφανίζει υψηλή απόδοση στα σύνολα δεδομένων Adult, Car Evaluation, Fatality Analysis Reporting System, Nursery και Molecular Biology. Αυτό υποδηλώνει ότι ο αλγόριθμος CNN Levenshtein είναι εξαιρετικά αποτελεσματικός στην κατηγοριοποίηση αυτών των κατηγορικών δεδομένων. Αντίστοιχα, όσον αφορά την παράμετρο του ποσοστού μείωσης (RR), παρατηρούμε ότι ο αλγόριθμος CNN Levenshtein έχει το υψηλότερο ποσοστό μείωσης σε πολλά από τα σύνολα δεδομένων, όπως Chess, Connect-4, Fatality Analysis Reporting System και Nursery.

Συνοψίζοντας, οι παραλλαγές του αλγορίθμου CNN σε σύνολα κατηγορικών δεδομένων υφίστανται μικρή απώλεια ακρίβειας, αλλά ταυτόχρονα επιτυγχάνουν σημαντική μείωση του μεγέθους των δεδομένων. Σύμφωνα με τα αποτελέσματα των πειραματικών μετρήσεων, ο αλγόριθμος CNN Levenshtein είναι μια αποτελεσματική μέθοδος για την κατηγοριοποίηση κατηγορικών δεδομένων, διατηρώντας σταθερά υψηλά επίπεδα ακρίβειας και απόδοσης.

4.3 Wilcoxon test

Τα αποτελέσματα του τεστ Wilcoxon για τις μετρήσεις της ακρίβειας (ACC) και του ποσοστού μείωσης (RR) εμφανίζονται στον Πίνακα 3. Η στήλη με την ένδειξη "w/l/t" παρουσιάζει τον αριθμό των νικών, των ηττών και των ισοπαλιών για κάθε δοκιμή σύγκρισης ζεύγους. Η στήλη, με την ένδειξη "Wilcoxon", υποδεικνύει μια αριθμητική τιμή που ποσοτικοποιεί τη σημασία της διαφοράς μεταξύ των δύο συγκριτικών αλγορίθμων. Εάν αυτή η τιμή είναι μικρότερη από 0,05, μπορεί να συναχθεί το συμπέρασμα ότι η διαφορά είναι στατιστικά σημαντική. Τα αποτελέσματα δείχνουν ότι δεν υπάρχει στατιστική διαφορά στην ακρίβεια μεταξύ των αλγορίθμων CNN Levenshtein και K-NN Levenshtein, των αλγορίθμων CNN Jaccard και K-NN Jaccard και μεταξύ των αλγορίθμων CNN Jaccard vs CNN Levenshtein.

Επιπρόσθετα, τα αποτελέσματα του τεστ Wilcoxon επιβεβαιώνουν ότι υπάρχει στατιστική διαφορά ως προς την ακρίβεια (accuracy) μεταξύ του αλγορίθμου CNN Hamming και του αλγορίθμου K-NN Hamming. Αυτό σημαίνει ότι ο αλγόριθμος CNN Hamming

θεωρείται στατιστικά καλύτερος από τον αλγόριθμο K-NN Hamming. Επιπρόσθετα, υπάρχει στατιστική διαφορά ως προς την ακρίβεια μεταξύ του αλγορίθμου CNN Jaccard και του αλγορίθμου CNN Hamming. Αυτό σημαίνει ότι ο αλγόριθμος CNN Jaccard θεωρείται στατιστικά καλύτερος από τον αλγόριθμο CNN Hamming.

Σύμφωνα με το τεστ Wilcoxon, υπάρχει στατιστική διαφορά ως προς το ποσοστό μείωσης (reduction rate) μεταξύ του αλγορίθμου CNN Jaccard και του αλγορίθμου CNN Levenshtein.

Μέθοδοι	**Accuracy**		**Reduction Rate**	
	w/l/t	**Wilc.**	**w/l/t**	**Wilc.**
CNN Hamming vs K-NN Hamming	8/0/0	**0.012**	-	-
CNN Levenshtein vs K-NN Levenshtein	6/2/0	0.069	-	-
CNN Jaccard vs K-NN Jaccard	6/2/0	0.050	-	-
CNN Jaccard vs CNN Levenshtein	5/3/0	0.484	1/7/0	**0.025**
CNN Levenshtein vs CNN Hamming	2/6/0	0.161	5/3/0	0.093
CNN Jaccard vs CNN Hamming	1/6/1	**0.043**	3/5/0	0.327

Πίνακας 3: Αποτελέσματα δοκιμών Wilcoxon για μετρήσεις ACC και RR

Συνοψίζοντας, τα αποτελέσματα από το τεστ Wilcoxon δείχνουν ότι ο αλγόριθμος CNN Hamming είναι στατιστικά καλύτερος από τον αλγόριθμο K-NN Hamming όσον αφορά την ακρίβεια, ενώ ο αλγόριθμος CNN Jaccard είναι στατιστικά καλύτερος από τον CNN Hamming όσον αφορά την ακρίβεια. Ωστόσο, δεν υπάρχει στατιστική διαφορά στην ακρίβεια μεταξύ των αλγορίθμων CNN Levenshtein και CNN Jaccard [115].

4.4 Friedman test

Τα αποτελέσματα του τεστ Friedman για τις μετρήσεις της ακρίβειας (ACC) και του ποσοστού μείωσης (RR) εμφανίζονται στον Πίνακα 4. Για τη μέτρηση της ακρίβειας, ο αλγόριθμος CNN Hamming βρίσκεται στην πρώτη θέση με τη χαμηλότερη μέση κατάταξη, ενώ οι αλγόριθμοι CNN Levenshtein και CNN Jaccard βρίσκονται στις δεύτερες θέσεις. Για τη μέτρηση του ποσοστού μείωσης, ο αλγόριθμος CNN Hamming βρίσκεται στην δεύτερη

θέση, ενώ ο αλγόριθμος CNN Levenshtein και ο αλγόριθμος CNN Jaccard βρίσκονται στην πρώτη και τρίτη θέση.

Οι κατατάξεις αυτές χρησιμοποιούνται για τον υπολογισμό του τεστ Friedman, το οποίο είναι ένα μη παραμετρικό τεστ που χρησιμοποιείται για τη σύγκριση των αλγορίθμων σε πολλαπλές μετρήσεις. Το τεστ αξιολογεί αν οι διαφορές μεταξύ των αλγορίθμων είναι στατιστικά σημαντικές.

Από το τεστ Friedman προκύπτει ότι υπάρχει στατιστικά σημαντική διαφορά μεταξύ των αλγορίθμων σε τουλάχιστον μία από τις μετρήσεις (ακρίβεια ή ποσοστό μείωσης). Συγκεκριμένα, ο αλγόριθμος CNN Hamming διαφέρει στατιστικά σημαντικά από τους αλγορίθμους CNN Levenshtein και CNN Jaccard στη μέτρηση του ποσοστού μείωσης.

Γενικά, αυτά τα αποτελέσματα υποδεικνύουν ότι ο αλγόριθμος CNN Hamming είναι η καλύτερη επιλογή για τη μέτρηση της ακρίβειας, ενώ οι αλγόριθμοι CNN Levenshtein και CNN Jaccard είναι καλύτεροι για τη μέτρηση του ποσοστού μείωσης [115].

Αλγόριθμος	**Μέση κατάταξη**	
	ACC	**RR**
k-NN Hamming	3.63	-
k-NN Levenshtein	4.75	-
k-NN Jaccard	4.75	-
CNN Hamming	1.56	2
CNN Levenshtein	3.13	2.50
CNN Jaccard	3.19	1.50

Πίνακας 4: Αποτελέσματα δοκιμών Friedman για μετρήσεις ACC και RR

5

Συμπεράσματα και μελλοντική έρευνα

Η παρούσα διπλωματική εργασία επικεντρώνεται στο πρόβλημα της κατηγοριοποίησης των κατηγορικών δεδομένων που σχεδόν πάντα εμφανίζονται στα πραγματικά σύνολα δεδομένων εκπαίδευσης. Λαμβάνοντας υπόψη ότι το μέγεθος των συνόλων δεδομένων που χρησιμοποιούνται στις εφαρμογές αυξάνεται με μεγάλο ρυθμό, κρίνεται αναγκαία η επεξεργασία τους ώστε να περιέχουν μόνο τις χρήσιμες πληροφορίες. Οι μεγάλες ποσότητες δεδομένων δεν είναι εφικτό να χρησιμοποιηθούν από τους αλγόριθμους κατηγοριοποίησης εξαιτίας του υψηλού υπολογιστικού κόστους καθώς και των υψηλών απαιτήσεων αποθήκευσης στη μνήμη. Για την επίλυση αυτών των προβλημάτων υπεύθυνες είναι οι τεχνικές μείωσης του πληθυσμού των δεδομένων εκπαίδευσης (Data Reduction Techniques). Αναζητώντας στη σχετική βιβλιογραφία, έγινε αντιληπτό ότι οι περισσότερες τεχνικές αφορούν κυρίως τον κατηγοριοποιητή κ εγγύτερων γειτόνων (k-Nearest Neighbor classifier). Ωστόσο, οι τεχνικές αυτές δεν μπορούν να διαχειριστούν κατηγορικά δεδομένα. Για αυτόν τον λόγο κρίνεται απαραίτητη η εφαρμογή ενός ακόμη βήματος προ-επεξεργασίας για την μετατροπή των κατηγορικών δεδομένων σε αριθμητικά δεδομένα. Παράγοντας προβληματισμού είναι το γεγονός ότι αυτό το επιπλέον βήμα προ-επεξεργασίας θα προσθέτει επιπλέον υπολογιστικό κόστος. Η δική μας συνεισφορά συνίσταται στην ανάπτυξη μιας σημαντικής προσέγγισης στον χώρο της μηχανικής μάθησης βασισμένης σε στιγμιότυπα. Για την επίτευξη του στόχου μας, προτείνουμε νέες παραλλαγές του αλγορίθμου CNN-rule, μιας τεχνικής μείωσης του πληθυσμού των δεδομένων που μπορεί να διαχειριστεί τα κατηγορικά δεδομένα χωρίς να απαιτείται κάποιο επιπρόσθετο βήμα προ-επεξεργασίας για την μετατροπή

τους σε αριθμητικά δεδομένα. Προτείνουμε τους αλγόριθμους CNN Hamming, CNN Levenshtein και CNN Jaccard, οι οποίοι χρησιμοποιούν μετρικές απόστασης για μη μετρικούς χώρους.

Στην πειραματική μας μελέτη, εξετάσαμε την απόδοση του αλγορίθμου k-NN (k-Nearest Neighbors) σε σύγκριση με τις βελτιστοποιημένες παραλλαγές του αλγορίθμου CNN-rule. Χρησιμοποιήσαμε οκτώ διαφορετικά σύνολα κατηγορικών δεδομένων και μετρήσαμε την ακρίβεια (Accuracy - ACC) του κατηγοριοποιητή σε αυτά τα σύνολα, τόσο στην αρχική τους μορφή όσο και μετά από την εφαρμογή της τεχνικής μείωσης του πληθυσμού (Data Reduction Technique). Παράλληλα, μετρήσαμε το ποσοστό μείωσης (Reduction Rate - RR) του μεγέθους των δεδομένων μετά από κάθε πειραματική μέτρηση.

Ολοκληρώνοντας την πειραματική διαδικασία, προκύπτει ότι ο αλγόριθμος CNN Levenshtein εμφανίζει υψηλή απόδοση σε πολλά από τα σύνολα δεδομένων, διατηρώντας σταθερά υψηλά επίπεδα ακρίβειας και απόδοσης, ενώ παράλληλα επιτυγχάνει σημαντική μείωση του μεγέθους των δεδομένων. Αυτό υποδηλώνει ότι ο αλγόριθμος CNN Levenshtein είναι μια αποτελεσματική μέθοδος για την κατηγοριοποίηση κατηγορικών δεδομένων.

Στα πλαίσια της πειραματικής μελέτης πραγματοποιήσαμε επίσης το τεστ Wilcoxon για να αξιολογήσουμε τη στατιστική σημαντικότητα των διαφορών μεταξύ των αλγορίθμων σε ό,τι αφορά την ακρίβεια (Accuracy - ACC) και το ποσοστό μείωσης (Reduction Rate - RR). Σύμφωνα με τα αποτελέσματα του τεστ Wilcoxon, συμπεραίνουμε ότι δεν υπάρχει στατιστική διαφορά στην ακρίβεια (ACC) μεταξύ του αλγορίθμου CNN Levenshtein και των αλγορίθμων CNN Jaccard και CNN Hamming. Αυτό σημαίνει ότι οι αλγόριθμοι αυτοί έχουν παρόμοια απόδοση όσον αφορά την ακρίβεια.

Ωστόσο, υπάρχει στατιστική διαφορά στο ποσοστό μείωσης (RR) μεταξύ του αλγορίθμου CNN Levenshtein και του αλγορίθμου CNN Jaccard. Αυτό σημαίνει ότι ο αλγόριθμος CNN Levenshtein επιτυγχάνει μεγαλύτερη μείωση του μέγεθους των κατηγορικών δεδομένων σε σύγκριση με τον αλγόριθμο CNN Jaccard.

Συνεπώς, ο αλγόριθμος CNN Levenshtein έχει στατιστικά διαφορετική απόδοση από τον αλγόριθμο CNN Jaccard όσον αφορά το ποσοστό μείωσης, αλλά δεν υπάρχει στατιστική διαφορά στην ακρίβεια μεταξύ αυτών των δύο αλγορίθμων.

Η πειραματική μελέτη ολοκληρώνεται με το τεστ Friedman προκειμένου να συγκρίνουμε επιπλέον την στατιστική σημαντικότητα των διαφορών μεταξύ των αλγορίθμων. Σύμφωνα με τα αποτελέσματα του τεστ Friedman, προκύπτει το συμπέρασμα ότι ο αλγόριθμος CNN Hamming βρίσκεται στην πρώτη θέση με τη χαμηλότερη μέση κατάταξη για τη μέτρηση της ακρίβειας (Accuracy - ACC). Ο αλγόριθμος CNN Levenshtein βρίσκεται στην πρώτη θέση για τη μέτρηση του ποσοστού μείωσης (Reduction Rate - RR), ενώ ο

αλγόριθμος CNN Hamming βρίσκεται στη δεύτερη θέση και ο αλγόριθμος CNN Jaccard βρίσκεται στη τρίτη θέση.

Το περιεχόμενο της παρούσας διπλωματικής εργασίας θα μπορούσε σίγουρα να επεκταθεί. Στο πεδίο της μείωσης του πληθυσμού των δεδομένων προτείνονται συχνά νέοι αλγόριθμοι με βελτιωμένα χαρακτηριστικά. Ερευνητές και φοιτητές μπορούν να ασχοληθούν με την υλοποίηση των υπολοίπων αλγορίθμων Data Reduction.

6

Βιβλιογραφία

[1] Tan, P. N., Steinbach, M., Karpatne, A., & Kumar, V. (2018). Introduction to Data Mining (What's New in Computer Science).

[2] Βερύκιος, Β., Καγκλής, Β., & Σταυρόπουλος, Η. (2015). Η επιστήμη των δεδομένων μέσα από τη γλώσσα R [Προπτυχιακό εγχειρίδιο]. Κάλλιπος, Ανοικτές Ακαδημαϊκές Εκδόσεις.

[3] James, M. (1985). Classification algorithms. Wiley-Interscience.

[4] Jain, A., Somwanshi, D., Joshi, K., & Bhatt, S. S. (2022, April). A Review: Data Mining Classification Techniques. In 2022 3rd International Conference on Intelligent Engineering and Management (ICIEM) (pp. 636-642). IEEE.

[5] Κύρκος, Ε. (2015). Επιχειρηματική ευφυΐα και εξόρυξη δεδομένων [Προπτυχιακό εγχειρίδιο]. Κάλλιπος, Ανοικτές Ακαδημαϊκές Εκδόσεις.

[6] Furdík, K., Paralic, J., & Tutoky, G. (2008, September). Meta-learning method for automatic selection of algorithms for text classification. In Proc. of the Central European Conference on Information and Intelligent Systems (CECIIS 2008) (pp. 24-26).

[7] Cichosz, P. (2014). Data mining algorithms: explained using R. John Wiley & Sons.

[8] Kwon, O., & Sim, J. M. (2013). Effects of data set features on the performances of classification algorithms. Expert Systems with Applications, 40(5), 1847-1857.

[9] Gorade, S. M., Deo, A., & Purohit, P. (2017). A study of some data mining classification

techniques. International Research Journal of Engineering and Technology, 4(4), 3112-3115.

[10] García-Peñalvo, F. J., Peraković, D., Mishra, A., & Gupta, B. B. (2021). A Survey on Data mining classification approaches.

[11] Azuaje, F. (2006). Witten ih, frank e: Data mining: Practical machine learning tools and techniques 2nd edition.

[12] Mandadi, S. R. A. V. Y. A., Tejashwini, B., & Vijayakumar, S. A. N. J. A. N. (2020). Breast cancer classification using data mining. International Journal of Research in Engineering and Science, 8(11), 43-7.

[13] Wernick, M. N., Yang, Y., Brankov, J. G., Yourganov, G., & Strother, S. C. (2010). Machine learning in medical imaging. IEEE signal processing magazine, 27(4), 25-38.

[14] Aly, M. (2005). Survey on multiclass classification methods. Neural Netw, 19(1-9), 2.

[15] Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. IEEE transactions on information theory, 13(1), 21-27.

[16] Dasarathy, B. V. (1991). Nearest neighbor (NN) norms: NN pattern classification techniques. IEEE Computer Society Tutorial.

[17] Breiman, L., & Friedman, J. (1984). Charles J Stone, and Richard A Olshen. Classification and regression trees.

[18] Thabtah, F. (2007). A review of associative classification mining. The Knowledge Engineering Review, 22(1), 37-65.

[19] Lee, C. S., Cheang, P. Y. S., & Moslehpour, M. (2022). Predictive analytics in business analytics: decision tree. Advances in Decision Sciences, 26(1), 1-29.

[20] Hanson, R., Stutz, J., & Cheeseman, P. (1991). Bayesian classification theory (No. NAS 1.15: 107885).

[21] Langley, P., Iba, W., & Thompson, K. (1992, July). An analysis of Bayesian classifiers. In Aaai (Vol. 90, pp. 223-228).

[22] Domingos, P., & Pazzani, M. (1997). On the optimality of the simple Bayesian classifier under zero-one loss. Machine learning, 29, 103-130.

[23] Zhang, H. (2004). The optimality of naive bayes. In Valerie Barr and Zdravko Markov, editors, Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference, Miami Beach, Florida, USA, pages 562–567. AAAI Press

[24] Rish, I. (2001, August). An empirical study of the naive Bayes classifier. In IJCAI 2001 workshop on empirical methods in artificial intelligence (Vol. 3, No. 22, pp. 41-46).

[25] Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine learning, 20, 273-

297.

[26] Vapnik, V. N. (1995). The nature of statistical learning theory. 840 Springer-Verlag New York. Inc., New York, NY, USA, 841, 842.

[27] Özkan, Y. (2008). Data mining methods. İstanbul: Papatya Publishing.

[28] Gorunescu, F. (2011). Data Mining: Concepts, models and techniques (Vol. 12). Springer Science & Business Media.

[29] Moguerza, J. M., & Muñoz, A. (2006). Support vector machines with applications.

[30] Zell, A. (2003). Simulation Neuronaler Netze (Simulation with Neuronal Networks). Wissenschaftsverlag: Oldenbourg, Germany.

[31] Thabtah, F. (2007). A review of associative classification mining. The Knowledge Engineering Review, 22(1), 37-65.

[32] Chen, S., & Liu, B. (2001, April). Generating classification rules according to user's existing knowledge. In Proceedings of the 2001 SIAM International Conference on Data Mining (pp. 1-15). Society for Industrial and Applied Mathematics.

[33] Wei, C. C. (2015). Comparing lazy and eager learning models for water level forecasting in river-reservoir basins of inundation regions. Environmental Modelling & Software, 63, 137-155.

[34] Neelamegam, S., & Ramaraj, E. (2013). Classification algorithm in data mining: An overview. International Journal of P2P Network Trends and Technology (IJPTT), 4(8), 369-374.

[35] Aha, D. W. (1997). Artificial intelligence review. Editorial: Lazy Learning, 11(1-5), 7-10.

[36] Maimon, O. Z., & Rokach, L. (2014). Data mining with decision trees: theory and applications (Vol. 81). World scientific.

[37] Hendrickx, I., & Van Den Bosch, A. (2005). Hybrid algorithms with instance-based classification. In Machine Learning: ECML 2005: 16th European Conference on Machine Learning, Porto, Portugal, October 3-7, 2005. Proceedings 16 (pp. 158-169). Springer Berlin Heidelberg.

[38] Khan, I., Ahamd, A. R., Jabeur, N., & Mahdi, M. N. (2021). Towards an implementation of instance-based classifiers in pedagogical environment. Journal of Engineering Science and Technology, 16(5), 3757-3771.

[39] Aggarwal, C. C. (2014). Instance-Based Learning: A Survey. Data classification: algorithms and applications, 157.

[40] Herrera, F., Ventura, S., Bello, R., Cornelis, C., Zafra, A., Sánchez-Tarragó, D., ... &

Vluymans, S. (2016). Multiple instance learning (pp. 17-33). Springer International Publishing.

[41] Russell, S. J., Norvig, P., Canny, J. F., Malik, J. M., & Edwards, D. D. (2003). Artificial intelligence: a modern approach. vol. 2 Prentice hall. Upper Saddle River.

[42] Mitchell, T. M. (1997). Artificial neural networks. Machine learning, 45(81), 127.

[43] Martin, B.(1995) Instance-based learning: nearest neighbour with generalisation. (Working paper 95/18). Hamilton, New Zealand: University of Waikato, Department of Computer Science.

[44] Kotsiantis, S. B., Zaharakis, I., & Pintelas, P. (2007). Supervised machine learning: A review of classification techniques. Emerging artificial intelligence applications in computer engineering, 160(1), 3-24.

[45] Wilson, D. R., & Martinez, T. R. (2000). Reduction techniques for instance-based learning algorithms. Machine learning, 38, 257-286.

[46] Ougiaroglou, S., & Evangelidis, G. (2016). RHC: a non-parametric cluster-based data reduction for efficient k-NN classification. Pattern Analysis and Applications, 19(1), 93-109.

[47] Fix, E., & Hodges, J. L. (1989). Discriminatory analysis. Nonparametric discrimination: Consistency properties. International Statistical Review/Revue Internationale de Statistique, 57(3), 238-247.

[48] Kubat, M., & Kubat, J. A. (2017). An introduction to machine learning (Vol. 2, pp. 321-329). Cham, Switzerland: Springer International Publishing.

[49] Taunk, K., De, S., Verma, S., & Swetapadma, A. (2019). A Brief Review of Nearest Neighbor Algorithm for Learning and Classification. In 2019 International Conference on Intelligent Computing and Control Systems (ICCS). 2019 International Conference on Intelligent Computing and Control Systems (ICCS). IEEE.

[50] Vaquero Barnadas, M. (2016). Machine learning applied to crime prediction (Bachelor's thesis, Universitat Politècnica de Catalunya).

[51] Shi, Z. (2020). Improving k-nearest neighbors algorithm for imbalanced data classification. In IOP Conference Series: Materials Science and Engineering (Vol. 719, No. 1, p. 012072). IOP Publishing.

[52] Everitt, B., Landau, S., Leese, M., & Stahl, D. (2011). Cluster analysis. (5th ed.) Wiley.

[53] Hall, P., Park, B. U., & Samworth, R. J. (2008). Choice of neighbor order in nearest-neighbor classification.

[54] Aprilia, R. Y. D., Latuconsina, R., & Purboyo, T. W. (2018). A review of several

algorithms for data mining. J. Eng. Appl. Sci, 13, 6157-6161.

[55] K. Q. Weinberger, J. Blitzer, and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," in Advances in neural information processing systems, 2006, pp. 1473–1480.

[56] Chomboon, K., Chujai, P., Teerarassamee, P., Kerdprasop, K., & Kerdprasop, N. (2015, March). An empirical study of distance metrics for k-nearest neighbor algorithm. In Proceedings of the 3rd international conference on industrial application engineering (Vol. 2).

[57] Cost, S., & Salzberg, S. (1993). A weighted nearest neighbor algorithm for learning with symbolic features. Machine learning, 10, 57-78.

[58] Wilson, D. R., & Martinez, T. R. (1997). Improved heterogeneous distance functions. Journal of artificial intelligence research, 6, 1-34.

[59] Garcia, S., Derrac, J., Cano, J., & Herrera, F. (2012). Prototype selection for nearest neighbor classification: Taxonomy and empirical study. IEEE transactions on pattern analysis and machine intelligence, 34(3), 417-435.

[60] Triguero, I., Derrac, J., Garcia, S., & Herrera, F. (2011). A taxonomy and experimental study on prototype generation for nearest neighbor classification. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 42(1), 86-100.

[61] Ougiaroglou, S., Karamitopoulos, L., Tatoglou, C., Evangelidis, G., & Dervos, D. A. (2015). Applying prototype selection and abstraction algorithms for efficient time-series classification. In Artificial Neural Networks: Methods and Applications in Bio-/Neuroinformatics (pp. 333-348). Springer International Publishing.

[62] Ougiaroglou, S. (2014). Algorithms and techniques for efficient and effective nearest neighbours classification.

[63] Wilson, D. L. (1972). Asymptotic properties of nearest neighbor rules using edited data. IEEE Transactions on Systems, Man, and Cybernetics, (3), 408-421.

[64] Lozano Albalate, M. T. (2007). Data reduction techniques in classification processes (Phd Thesis). Universitat Jaume I

[65] Brown, M. S. (2014). Data mining for dummies. John Wiley & Sons.

[66] Devi, V. S., & Murty, M. N. (2002). An incremental prototype set building technique. Pattern Recognition, 35(2), 505-513.

[67] Wilson, D. R., & Martinez, T. R. (1997, July). Instance pruning techniques. In ICML (Vol. 97, No. 1997, pp. 400-411).

[68] Tepavčević, B., & Stojaković, V. (2014). Representation of non-metric concepts of space

in architectural design theories. Nexus Network Journal, 16, 285-297.

[69] Τσαμάτος, Π. Χ. (1999). Εισαγωγή στην τοπολογία: μετρικοί χώροι.

[70] O'Searcoid, M. (2006). Metric spaces. Springer Science & Business Media.

[71] Deza, E., Deza, M. M., Deza, M. M., & Deza, E. (2009). Encyclopedia of distances (pp. 1-583). Springer Berlin Heidelberg.

[72] Vermorel, J. (2005). Near neighbor search in metric and nonmetric space.

[73] Sepulveda, V., & Bustos, B. (2010, September). CP-Index: using clustering and pivots for indexing non-metric spaces. In Proceedings of the Third International Conference on SImilarity Search and APplications (pp. 75-82).

[74] Naidan, B., Boytsov, L., Malkov, Y., & Novak, D. (2015). Non-metric space library manual. arXiv preprint arXiv:1508.05470.

[75] Bregman, L. M. (1967). The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. USSR computational mathematics and mathematical physics, 7(3), 200-217.

[76] Boytsov, L., & Naidan, B. (2013). Engineering efficient and effective non-metric space library. In Similarity Search and Applications: 6th International Conference, SISAP 2013, A Coruña, Spain, October 2-4, 2013, Proceedings 6 (pp. 280-293). Springer Berlin Heidelberg.

[77] Goh, K. S., Li, B., & Chang, E. (2002, December). Dyndex: a dynamic and non-metric space indexer. In Proceedings of the tenth ACM international conference on Multimedia (pp. 466-475).

[78] Berndt, D. J., & Clifford, J. (1994, July). Using dynamic time warping to find patterns in time series. In Proceedings of the 3rd international conference on knowledge discovery and data mining (pp. 359-370).

[79] Rakthanmanon, T., Campana, B., Mueen, A., Batista, G., Westover, B., Zhu, Q., ... & Keogh, E. (2012, August). Searching and mining trillions of time series subsequences under dynamic time warping. In Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 262-270).

[80] Abboud, A., Backurs, A., & Williams, V. V. (2015, October). Tight hardness results for LCS and other sequence similarity measures. In 2015 IEEE 56th Annual Symposium on Foundations of Computer Science (pp. 59-78). IEEE.

[81] Stabili, D., Marchetti, M., & Colajanni, M. (2017, September). Detecting attacks to internal vehicle networks through Hamming distance. In 2017 AEIT International Annual Conference (pp. 1-6). IEEE.

[82] Hamming, R. W. (1950). Error detecting and error correcting codes. The Bell system technical journal, 29(2), 147-160.

[83] Afrati, F. N., Sarma, A. D., Menestrina, D., Parameswaran, A., & Ullman, J. D. (2012, April). Fuzzy joins using mapreduce. In 2012 IEEE 28th International Conference on Data Engineering (pp. 498-509). IEEE.

[84] Waggener, B., & Waggener, W. N. (1995). Pulse code modulation techniques. Springer Science & Business Media.

[85] Navarro, G. (2001). A guided tour to approximate string matching. ACM computing surveys (CSUR), 33(1), 31-88.

[86] Levenshtein, V. I. (1966, February). Binary codes capable of correcting deletions, insertions, and reversals. In Soviet physics doklady (Vol. 10, No. 8, pp. 707-710).

[87] Arockiya Jerson, J., & Preethi, N. (2023, February). An Analysis of Levenshtein Distance Using Dynamic Programming Method. In Proceedings of 3rd International Conference on Recent Trends in Machine Learning, IoT, Smart Cities and Applications: ICMISC 2022 (pp. 525-532). Singapore: Springer Nature Singapore.

[88] Berger, B., Waterman, M. S., & Yu, Y. W. (2020). Levenshtein distance, sequence comparison and biological database search. IEEE transactions on information theory, 67(6), 3287-3294.

[89] Wilson, R. C., & Hancock, E. R. (2004, August). Levenshtein Distance for Graph Spectral Features. In ICPR (2) (pp. 489-492).

[90] Raff, E., & Nicholas, C. (2018). Lempel-Ziv Jaccard Distance, an effective alternative to ssdeep and sdhash. Digital Investigation, 24, 34-49.

[91] Murphy, A. H. (1996). The Finley affair: A signal event in the history of forecast verification. Weather and forecasting, 11(1), 3-20.

[92] Jaccard, P. (1912). The distribution of the flora in the alpine zone. 1. New phytologist, 11(2), 37-50.

[93] Chung, N. C., Miasojedow, B., Startek, M., & Gambin, A. (2019). Jaccard/Tanimoto similarity test and estimation methods for biological presence-absence data. BMC bioinformatics, 20(15), 1-11.

[94] Kosub, S. (2019). A note on the triangle inequality for the Jaccard distance. Pattern Recognition Letters, 120, 36-38.

[95] Ouyang, M. (2016, September). KNN in the Jaccard space. In 2016 IEEE High Performance Extreme Computing Conference (HPEC) (pp. 1-7). IEEE.

[96] Chomboon, K., Chujai, P., Teerarassamee, P., Kerdprasop, K., & Kerdprasop, N. (2015,

March). An empirical study of distance metrics for k-nearest neighbor algorithm. In Proceedings of the 3rd international conference on industrial application engineering (Vol. 2).

[97] Hart, P. (1968). The condensed nearest neighbor rule (corresp.). IEEE transactions on information theory, 14(3), 515-516.

[98] Gates, G. (1972). The reduced nearest neighbor rule (corresp.). IEEE transactions on information theory, 18(3), 431-433.

[99] Ritter, G., Woodruff, H., Lowry, S., & Isenhour, T. (1975). An algorithm for a selective nearest neighbor decision rule (corresp.). IEEE Transactions on Information Theory, 21(6), 665-669.

[100] Aha, D. W., Kibler, D., & Albert, M. K. (1991). Instance-based learning algorithms. Machine learning, 6, 37-66.

[101] Kaur, S. P. (2013). Variables in research. Indian Journal of Research and Reports in Medical Sciences, 3(4), 36-38.

[102] Stanfill, C., & Waltz, D. (1986). Toward memory-based reasoning. Communications of the ACM, 29(12), 1213-1228.

[103] J. Boyer. "Prepare data for machine learning: Improve data quality with data cleansing, data transformation, and feature engineering." IBM.

https://www.ibm.com/garage/method/practices/reason/prepare-data-for-machine-learning/ (accessed 19 September, 2023).

[104] Zheng, A., & Casari, A. (2018). Feature engineering for machine learning: principles and techniques for data scientists. " O'Reilly Media, Inc.".

[105] KEEL: A software tool to assess evolutionary algorithms for Data Mining problems (regression, classification, clustering, pattern mining and so on). (n.d.). Retrieved June 7, 2023, from sci2s.ugr.es website: http://sci2s.ugr.es/keel/datasets.php

[106] UCI Machine Learning Repository: Data Sets. (2009). Retrieved June 7, 2023, from Uci.edu website: https://archive.ics.uci.edu/datasets

[107] KEEL: A software tool to assess evolutionary algorithms for Data Mining problems (regression, classification, clustering, pattern mining and so on). (n.d.). Retrieved June 7, 2023, from sci2s.ugr.es website: https://sci2s.ugr.es/keel/dataset.php?cod=192

[108] KEEL: A software tool to assess evolutionary algorithms for Data Mining problems (regression, classification, clustering, pattern mining and so on). (n.d.). Retrieved June 7, 2023, from sci2s.ugr.es website: https://sci2s.ugr.es/keel/dataset.php?cod=56

[109] KEEL: A software tool to assess evolutionary algorithms for Data Mining problems

(regression, classification, clustering, pattern mining and so on). (n.d.). Retrieved June 7, 2023, from sci2s.ugr.es website: https://sci2s.ugr.es/keel/dataset.php?cod=197

[110] KEEL: A software tool to assess evolutionary algorithms for Data Mining problems (regression, classification, clustering, pattern mining and so on). (n.d.). Retrieved June 7, 2023, from sci2s.ugr.es website:

https://archive.ics.uci.edu/dataset/22/chess+king+rook+vs+king+pawn

[111] KEEL: A software tool to assess evolutionary algorithms for Data Mining problems (regression, classification, clustering, pattern mining and so on). (n.d.). Retrieved June 7, 2023, from sci2s.ugr.es website: https://sci2s.ugr.es/keel/dataset.php?cod=193

[112] KEEL: A software tool to assess evolutionary algorithms for Data Mining problems (regression, classification, clustering, pattern mining and so on). (n.d.). Retrieved June 7, 2023, from sci2s.ugr.es website: https://sci2s.ugr.es/keel/dataset.php?cod=191

[113] KEEL: A software tool to assess evolutionary algorithms for Data Mining problems (regression, classification, clustering, pattern mining and so on). (n.d.). Retrieved June 7, 2023, from sci2s.ugr.es website: https://sci2s.ugr.es/keel/dataset.php?cod=103

[114] KEEL: A software tool to assess evolutionary algorithms for Data Mining problems (regression, classification, clustering, pattern mining and so on). (n.d.). Retrieved June 7, 2023, from sci2s.ugr.es website: https://sci2s.ugr.es/keel/dataset.php?cod=181

[115] Ougiaroglou, S., Filippakis, P., & Evangelidis, G. (2021). Prototype generation for multi-label nearest neighbours classification. In Hybrid Artificial Intelligent Systems: 16th International Conference, HAIS 2021, Bilbao, Spain, September 22–24, 2021, Proceedings 16 (pp. 172-183). Springer International Publishing.