



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

Μέθοδοι ανάπτυξης Νευρωνικών Δικτύων για
αποτελεσματική ταξινόμηση συνόλων δεδομένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Νικολαΐδη Β. Κωνσταντίνου ΑΜ:03109206

Επιβλέπων:
Κόλλιας Στέφανος,
Καθηγητής Ε.Μ.Π

Αθήνα, Φεβρουάριος 2016



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

Μέθοδοι ανάπτυξης Νευρωνικών Δικτύων για
αποτελεσματική ταξινόμηση συνόλων δεδομένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Νικολαΐδη Κωνσταντίνου ΑΜ:03109206

Επιβλέπων:

Κόλλιας Στέφανος,
Καθηγητής Ε.Μ.Π

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 26η Φεβρουαρίου 2016.

.....
Στέφανος Κόλλιας

.....
Α.Γ Σταφυλοπάτης

.....
Γεώργιος Στάμου

Καθηγητής Ε.Μ.Π

Καθηγητής Ε.Μ.Π

Λέκτορας Ε.Μ.Π

Αθήνα ,Φεβρουάριος 2016

.....
Νικολαΐδης Κωνσταντίνος

Copyright © Νικολαΐδης Κωνσταντίνος, 2016
Με επιφύλαξη παντός δικαιώματος. All rights reserved.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα.

Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσόβιου Πολυτεχνείου

Περίληψη

Σκοπός της εργασίας αυτής είναι ο σχεδιασμός και στην συνέχεια η υλοποίηση τεχνικών ταξινόμησης με την αποτελεσματική αξιοποίηση ομάδων νευρωνικών δικτύων. Οι τεχνικές που μελετώνται σχετίζονται με την κατάτμηση του δεδομένου συνόλου εκπαίδευσης σε επιμέρους τμήματα κάθε φορά με διαφορετικό τρόπο και κατόπιν στον διαμοιρασμό σε διαφορετικά δίκτυα, διαφορετικών τμημάτων του συνόλου αυτού. Τα κριτήρια τμηματοποίησης είναι άλλοτε καθαρά χωρικά, στον χώρο των χαρακτηριστικών, ενώ σε άλλες περιπτώσεις σχετίζονται και με τις κλάσεις των προτύπων.

Λέξεις κλειδιά:

Εκπαίδευση , Νευρωνικό δίκτυο , Ταξινόμηση, Μέθοδος , Κλάση

Abstract

The aim of this work is to design and then implement different classification techniques through the effective use of groups of neural networks. The techniques studied divide the training data set in different subsections ,each time with a different method, and then they share these subsections in different neural networks. The segmentation criteria are sometimes purely spatial, in the field of features, while in other cases associated with the output classes of the given data.

Key words:

Training , Neural Network , Classification, Method , Class

Περιεχόμενα:

- 1.Εισαγωγή 8
- 2.Θεωρητικό Μέρος 10
 - 2.0. Γενικές έννοιες 10
 - 2.0.1 Μηχανική Μάθηση 10
 - 2.0.2 Στατιστική ταξινόμηση 11
 - 2.1 Τεχνητά Νευρωνικά Δίκτυα 13
 - 2.2 Ταξινομητής KNN 25
 - 2.3 Ταξινομητής *K-means* 28
 - 2.4 Χάρτες Αυτοοργάνωσης 31
 - 2.4.1 Παραλλαγές αυξανομένων *SOM* 34
 - 2.4.2 *ESOINN* 36
 - 2.5 Ο αλγόριθμος EM 37
 - 2.6 Support Vector Machines(SVM) 41
 - 2.7 Naive Bayesian Classifier 47
 - 2.8 Classifier Ensembles 50
 - 2.8.1 Εισαγωγή 50
 - 2.8.2 *Bagging* 53
 - 2.8.2.1 *RandomForests* 54
 - 2.8.3 *Boosting* 56
 - 2.8.3.1 *Adaboost* 58
3. Πρακτικό Μέρος 63
 - 3.1 Συναρτήσεις και διαδικασίες που προηγήθηκαν της εφαρμογής 63
 - 3.2 Μελέτη συμπεριφοράς ομάδας εξειδικευμένων νευρωνικών δικτύων 66
 - 3.2.1 Γενικά 66
 - 3.2.2 Υλοποίηση 69
 - 3.2.2.1 Περίπτωση Νευρωνικού σαν ταξινομητή υπερομάδων 71
 - 3.2.2.2 Περίπτωση *knn* σαν ταξινομητή υπερομάδων 73
 - 3.2.3 Αποτελέσματα 73
 - 3.2.4 Επεκτάσεις 81
 - 3.3 Μελέτη συμπεριφοράς συστήματος νευρωνικών δικτύων που βασίζεται στο *boosting* 86
 - 3.3.1 Γενικά 86
 - 3.3.2 Υλοποίηση 89
 - 3.3.3 Αποτελέσματα 91
 - 3.3.4 Επεκτάσεις 102

| | | |
|-------|--|-----|
| 3.4 | Μελέτη συστήματος ξεχωριστής εκπαίδευσης επι μέρους συστάδων | 105 |
| 3.4.1 | Γενικά | 105 |
| 3.4.2 | Υλοποίηση | 107 |
| 3.4.3 | Αποτελέσματα | 109 |
| 3.4.4 | Επεκτάσεις/Τροποποιήσεις | 118 |
| 3.5 | Μελέτη συστήματος προσαρμογής των <i>clusters</i> προτύπων εκπαίδευσης στα πρότυπα ελέγχου | 122 |
| 3.5.1 | Γενικά | 122 |
| 3.5.2 | Υλοποίηση | 124 |
| 3.5.3 | Αποτελέσματα | 126 |
| 3.6 | Σχολιασμός αποτελεσμάτων για τα δυσκολότερα σύνολα | 136 |
| 3.7 | Συγκρίσεις /Τελικά αποτελέσματα | 138 |
| 4. | Επίλογος /Προτεινόμενες επεκτάσεις | 145 |

1. Εισαγωγή

Η παρούσα εργασία έχει ως σκοπό να ασχοληθεί με την ανάπτυξη και μελέτη συστημάτων μηχανικής μάθησης τα οποία στοχεύουν στην ταξινόμηση προτύπων. Το σχεπτικό είναι να χρησιμοποιήσουμε με αποτελεσματικό τρόπο πολλούς διαφορετικούς απλούς ταξινομητές σε διάφορες μεθόδους για την σύσταση μιας ομάδας, η οποία κάθε φορά για τους διαφορετικούς αλγόριθμους που εξετάζουμε θα φτιάχνεται και με διαφορετικό τρόπο, ώστε τελικά να παρέχει ισχυρότερα αποτελέσματα σε σχέση με οποιονδήποτε από τους μεμονωμένους ταξινομητές της, αλλά και σε σύγκριση με πιο απλές περιπτώσεις συστημάτων.

Στην εργασία αυτή θα ασχοληθούμε με την χρήση νευρωνικών δικτύων ως βασικού ταξινομητή σε κάθε περίπτωση διότι προσομοιάζει -σε πολύ θεμελιώδες επίπεδο βέβαια- ορισμένες λειτουργίες του ανθρώπινου εγκεφάλου. Επίσης, αν και υπάρχουν πολύ πιο απλά και σε ορισμένες περιπτώσεις αποτελεσματικά μοντέλα απλών ταξινομητών (όπως πχ τα δέντρα απόφασης), είχε ενδιαφέρον να μελετηθεί η περίπτωση των νευρωνικών λόγω του ότι αποτελούν ένα πιο φυσικό μοντέλο, το οποίο όμως έχει μια φορμαλιστική μαθηματική αποτύπωση.

Η εργασία χωρίζεται σε δύο τμήματα : το θεωρητικό και το πρακτικό. Στο θεωρητικό θα ασχοληθούμε γενικότερα με τις έννοιες της ταξινόμησης και της μηχανικής μάθησης πιο συνολικά, ενώ επίσης θα δούμε με μία συνοπτική παρουσίαση -που έγινε και για την καλύτερη κατανόηση των συστημάτων από τον συγγραφέα-, διάφορους ταξινομητές και συστήματα ομαδοποίησης καθώς και την βασική θεωρία/ λογική της σχεδίασης του. Θα δοθεί πιο μεγάλη έμφαση σε κάποιους απότι σε κάποιους άλλους είτε λόγω του ότι αυτοί χρησιμοποιούνται στην εργασία είτε γιατί αυτοί θα μπορούσαν να 'ταιριάξουν' καλύτερα σε συστήματα τα οποία βασίζονται στην ομαδική απόφαση έναντι της μεμονωμένης.

Στο πρακτικό τμήμα της εργασίας θα παρουσιαστούν οι αλγόριθμοι που σχεδιάστηκαν και κατόπιν υλοποιήθηκαν. Παρουσιάζονται ξεχωριστά υποτμήματα, καθένα από τα οποία ουσιαστικά αποτελεί μια αναφορά πάνω στον συγκεκριμένο αλγόριθμο και στην υλοποίηση του. Πιο συγκεκριμένα, σε καθένα από τα τμήματα αυτά παρουσιάζεται μια σύντομη περιγραφή του συγκεκριμένου αλγορίθμου, ή για την ακρίβεια του σχεπτικού λειτουργίας του, στην συνέχεια αναφερόμαστε πιο αναλυτικά στην υλοποίηση του αλγορίθμου και περιγράφουμε ουσιαστικά τον κώδικά που χρησιμοποιήθηκε για την υλοποίηση του συγκεκριμένου τμήματος. Κατόπιν, εμφανίζουμε και σχολιάζουμε τα διάφορα αποτελέσματα από τα σύνολα στα οποία εφαρμόσαμε τους αλγορίθμους

μας και τέλος προτείνουμε όπου υπήρξε κάποια πρόσθετη ιδέα πιθανές επεκτάσεις/τροποποιήσεις για τον κάθε αλγόριθμο.

Αξίζει να σημειωθεί ότι αν και κατα πάσα πιθανότητα αντίστοιχα συστήματα ήδη υπάρχουν και έχουν αναπτυχθεί και μάλιστα σε μεγαλύτερο βάθος, δεδομένου ότι η διαδικασία αναπτυξής τους στην παρούσα εργασία ήταν ανεξάρτητη δεν εμπεριέχονται στο πρακτικό κομμάτι σχετικές βιβλιογραφικές αναφορές εφόσον δεν χρησιμοποιήθηκαν.

Ακόμη σε αυτό το κομμάτι υπάρχει ένα μικρό τμήμα για τις built in συναρτήσεις που χρησιμοποιήθηκαν στον κώδικα, καθώς και για τις διάφορες παραδοχές που χρησιμοποιήθηκαν κατά την εκπαίδευση για τα διαφορετικά σύνολα αλλά και συνολικά.

Τέλος, παρουσιάζονται τα καταληκτικά αποτελέσματα και οι συγκρίσεις των διαφόρων αλγορίθμων συνολικά σε μια προσπάθεια να προσδιορίσουμε ποιος από αυτούς είναι ο πιο γενικός, καθώς και αν τελικά αξίζει τον κόπο η χρήση ενός ομαδικού συστήματος σε σχέση με κάποιο πιο απλό. Από την παραπάνω διαδικασία οδηγούμαστε σε κάποια ενδιαφέροντα τελικά συμπεράσματα.

Εν κατακλείδι ένα πολύ γενικό συμπέρασμα που βγήκε από την εργασία και που είναι αξιο αναφοράς στην εισαγωγή αν και ομολογουμένως καπως απλοϊκό, είναι ότι η ποικιλία προβλημάτων οδηγεί και σε ποικιλία λύσεων.

2. Θεωρητικό μέρος:

Η μάθηση αποτελεί ένα πολυδιάστατο φαινόμενο, το οποίο περιλαμβάνει την απόκτηση νέας (δηλωτικής) γνώσης, την ανάπτυξη κινησιολογικών και γνωστικών ικανοτήτων μέσω καθοδήγησης ή εξάσκησης, την οργάνωση νέας γνώσης σε γενικότερες αποτελεσματικές αποτυπώσεις και την ανακάλυψη νέων γεγονότων και θεωριών μέσω παρατήρησης και πειραματισμού.

Απο την έναρξη της εποχής των υπολογιστών οι ερευνητές προσπαθούν να μεταδώσουν προαναφερθείσες ιδιότητες της μάθησης στους υπολογιστές. Η επίλυση αυτού του προβλήματος αποτελεί έναν από τους πιο σημαντικούς και ενδιαφέροντες μακροπρόθεσμους στόχους στο πεδίο της τεχνητής νοημοσύνης. Η μελέτη και η μοντελοποίηση των διαδικασιών μάθησης στις ποικίλες εκδηλώσεις τους συνιστούν το αντικείμενο μελέτης της μηχανικής μάθησης[10].

2.0 Γενικές έννοιες:

2.0.1 Μηχανική Μάθηση:

Η Μηχανική μάθηση είναι ένα πεδίο της επιστήμης των υπολογιστών που συσχετίζεται με τη μελέτη της αναγνώρισης προτύπων και τη θεωρία της υπολογιστικής μάθησης στην τεχνητή νοημοσύνη[3]. Η Μηχανική μάθηση διερευνά την μελέτη και κατασκευή αλγορίθμων που μπορούν να μάθουν από και να κάνουν προβλέψεις σε δεδομένα [4].

Στόχος της είναι να κάνουμε τους υπολογιστές να δρουν χωρίς να έχει προηγηθεί σε αυτούς κάποιος ρητός προγραμματισμός. Κατά την τελευταία δεκαετία, η μηχανική μάθηση μας έχει δώσει αυτο-οδηγούμενα αυτοκίνητα, έμπρακτη αναγνώριση ομιλίας, αποτελεσματική αναζήτηση στο διαδίκτυο και μια σημαντικά βελτιωμένη κατανόηση του ανθρώπινου γονιδιώματος. Η Μηχανική Μάθηση είναι τόσο διάχυτη σήμερα που κατα πάσα πιθανότητα τη χρησιμοποιούμε δεκάδες φορές την ημέρα χωρίς να το καταλαβαίνουμε. Υπάρχουν διάφορα είδη Μηχανικής Μάθησης τα οποία θα μελετηθούν παρακάτω.

Πρίν προχωρήσουμε πρέπει να δώσουμε ιδιαίτερη έμφαση στον όρο της αναγ-

νώρισης προτύπων δεδομένου ότι η παρούσα εργασία πραγματεύεται σε μεγάλο βαθμό συστήματα που εκτελούν ακριβώς αυτή τη διεργασία.

Η Αναγνώριση προτύπων επομένως ορίζεται ως η διαδικασία δια της οποίας ένα προσλαμβανόμενο σήμα/πρότυπο αντιστοιχίζεται σε μια κλάση που ανήκει σε ένα προκαθορισμένο σύνολο κλάσεων. Οι άνθρωποι τα καταφέρνουν πολύ καλά στην αναγνώριση προτύπων. Είναι ιδιαίτερα ικανοί στην αναγνώριση της προέλευσης των δεδομένων που λαμβάνουν μέσω των αισθήσεων τους. Συχνά μαλιστα, αυτό γίνεται με πρακτικά μηδενική προσπάθεια. Για παράδειγμα είναι πολύ εύκολο για κάποιον να αναγνωρίσει ένα οικείο πρόσωπο ακόμα και αν αυτό έχει αλλάξει από την τελευταία φορά που το είδε.[1] Βέβαια, για να επιτευχθεί αυτό το αποτέλεσμα, το άτομο πρέπει να περάσει από μια διαδικασία μάθησης (στο παράδειγμα να δει το πρόσωπο του άλλου ατόμου πολλές φορές).

Βλέπουμε επομένως ότι σε μεγάλο βαθμό η επιστήμη της μηχανικής μάθησης έχει την τάση να μιμνεί μηχανικά (και ίσως μακροπρόθεσμα να βελτιστοποιήσει) ιδιότητες στις οποίες ο άνθρωπος είναι πολύ ικανός. Παρακάτω θα περιγράψουμε συστήματα τα οποία υλοποιούν μεθόδους αναγνώρισης προτύπων μέσω της ταξινόμησης και θα διερευνήσουμε πιο αναλυτικά διάφορα είδη μηχανικής μάθησης.

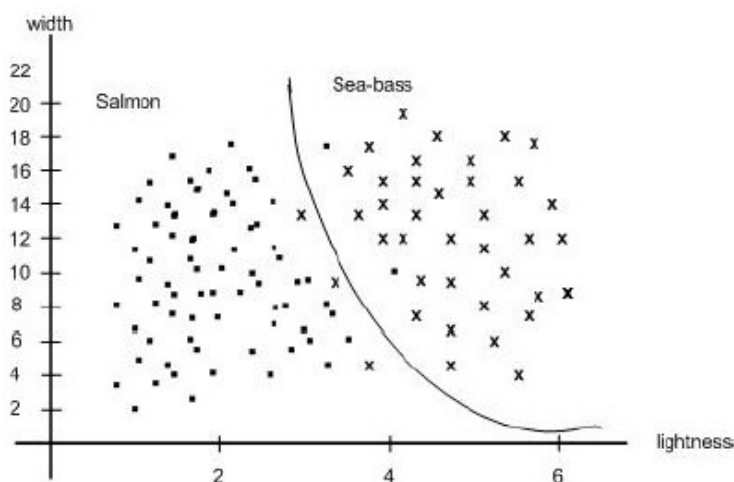
2.0.2 Στατιστική Ταξινόμηση, Είδη Μάθησης και Ταξινομητές:

Στο πεδίο της Μηχανικής Μάθησης και της Στατιστικής, ορίζουμε ως ταξινόμηση το πρόβλημα του προσδιορισμού σε ποιο από ένα σύνολο κατηγοριών (υποπληθυσμών) ανήκει μια νέα παρατήρηση με βάση ένα σύνολο εκπαίδευσης των δεδομένων που περιέχει τις παρατηρήσεις (ή περιπτώσεις) των οποίων η κατηγορία είναι γνωστή. Βλέπουμε λοιπόν ότι η ταξινόμηση είναι άμεσα συνυφασμένη με την αναγνώριση προτύπων. Για παράδειγμα, η τοποθέτηση ενός email σε κάποια από τις κατηγορίες 'spam' ή 'non-spam' με βάση κάποια παρατηρούμενα χαρακτηριστικά αποτελεί μια διαδικασία ταξινόμησης.

Η ταξινόμηση, σε όρους Μηχανικής Μάθησης, αποτελεί μια διαδικασία επιβλεπόμενης μάθησης. Η επιβλεπόμενη μάθηση αποτελεί ουσιαστικά μάθηση υπό την καθοδήγηση ενός εκπαιδευτή. Με εννοιολογικούς όρους, μπορούμε να θεωρήσουμε για την επιβλεπόμενη μάθηση ότι ο εκπαιδευτής έχει γνώση του περιβάλλοντος και αυτή η γνώση αντιπροσωπεύεται από ένα σύνολο παραδειγμάτων εισόδου-εξόδου. Ωστόσο το περιβάλλον είναι άγνωστο στον ταξινομητή.[1]

Εδώ πρέπει να σημειωθεί ότι ταξινομητής είναι ένα σύστημα το οποίο πραγ-

ματοποιεί μια διαδικασία κατηγοριοποίησης . Πιο συγκεκριμένα, εκτελεί μια χαρτογράφηση από ένα χώρο χαρακτηριστικών X σε ένα σύνολο ετικετών Y . Στο παραπάνω παράδειγμα εκτελεί την ταξινόμηση σε 'spam' ή 'non-spam' . Αξίζει να αναφερθεί η διαφορά με την συσταδοποίηση (*clustering*) όπου ο αλγόριθμος χωρίζει τα δεδομένα αυτόνομα σε συστάδες (*clusters*) έτσι ώστε τα δεδομένα κάθε συστάδας να έχουν κοινά στο χώρο χαρακτηριστικών.[2]



2.1. Παράδειγμα ταξινόμησης προτύπων στις κλάσεις σολομός και λαβράκι με βάση τα χαρακτηριστικά πλάτος και φωτεινότητα

Υπάρχουν και άλλοι τρόποι μάθησης. Πιο συγκεκριμένα, υπάρχει η μάθηση χωρίς εκπαιδευτή η οποία χωρίζεται σε μη ελεγχόμενη μάθηση και σε ενισχυτική μάθηση.

Στην ενισχυτική μάθηση η εκμάθηση μιας αντιστοιχίας εισόδου εξόδου εκτελείται μέσω συνεχούς αλληλεπίδρασης με το περιβάλλον με στόχο την ελαχιστοποίηση ενός βαθμωτού δείκτη απόδοσης . Ο δείκτης απόδοσης αυτός, ορίζεται ως πρόβλεψη του συνολικού κόστους ενεργειών που εκτελούνται σε μια αλληλουχία βημάτων αντί απλώς του άμεσου κόστους μιας ενέργειας. Ενδεχομένως ορισμένες απο τις ενέργειες που έχουν εκτελεστεί σε αυτή την αλληλουχία χρονικών βημάτων να είναι καλύτερες ορίζουσες της συνολικής συμπεριφοράς του συστήματος . Η λειτουργία λοιπόν του συστήματος μάθησης είναι να ανακαλύψει αυτές τις ενέργειες και να τις επανατροφοδοτήσει στο περιβάλλον.

Στη μη επιβλεπόμενη ή αυτο-οργανούμενη μάθηση, δεν υπάρχει εξωτερικός εκπαιδευτής ή κριτής που να επιβλέπει τη διαδικασία μάθησης . Αντ' αυτού , υπάρχει ένα ανεξάρτητο απο την εργασία μέτρο της ποιότητας της αναπαράσ-

τασης που καλείται να μάθει το δίκτυο και οι ελεύθερες παράμετροι του δικτύου βελτιστοποιούνται με βάση αυτό το μέτρο. Για ένα συγκεκριμένο ανεξάρτητο από την εργασία μέτρο, αφού το δίκτυο συντονιστεί στις στατιστικές κανονικότητες των δεδομένων εισόδου, αναπτύσσει την δυνατότητα να σχηματίζει εσωτερικές αναπαραστάσεις για την κωδικοποίηση χαρακτηριστικών εισόδου και μέσω αυτών, να δημιουργεί νέες κλάσεις αυτόματα.

Για την εκτέλεση της μη επιβλεπόμενης μάθησης μπορεί να χρησιμοποιηθεί ένας κανόνας ανταγωνιστικής μάθησης. Για παράδειγμα, μπορεί να χρησιμοποιηθεί ένα νευρωνικό δίκτυο το οποίο θα αποτελείται από δύο επίπεδα- ένα επίπεδο εισόδου και ένα ανταγωνιστικό επίπεδο. Το επίπεδο εισόδου θα λαμβάνει τα διαθέσιμα δεδομένα ενώ το ανταγωνιστικό επίπεδο θα αποτελείται από νευρώνες οι οποίοι θα ανταγωνίζονται ο ένας τον άλλο (σύμφωνα με ένα κανόνα μάθησης) για την ευκαιρία να αποκριθούν σε χαρακτηριστικά που περιέχονται σε δεδομένα εισόδου. Στην απλούστερη δυνατή μορφή του, το δίκτυο λειτουργεί σύμφωνα με μια στρατηγική 'ο νικητής τα παίρνει όλα'. Βάσει μιας τέτοιας στρατηγικής, ο νευρώνας με τη μεγαλύτερη συνολική είσοδο νικά στον ανταγωνισμό και ενεργοποιείται. Όλοι οι άλλοι νευρώνες του δικτύου απενεργοποιούνται.[1]

Παρακάτω θα εξετάσουμε πιο αναλυτικά διάφορους δημοφιλείς ταξινομητές και θα περιγράψουμε την λειτουργία τους, προκειμένου να μελετηθούν πιο διεξοδικά διάφορες τεχνικές ταξινόμησης. Θα δοθεί ιδιαίτερη έμφαση σε αυτούς που χρησιμοποιήθηκαν στην παρούσα εργασία.

2.1 Τεχνητά Νευρωνικά Δίκτυα (Artificial Neural Networks):

Στη μηχανική μάθηση, τα τεχνητά νευρωνικά δίκτυα αποτελούν μια οικογένεια στατιστικών μοντέλων μάθησης εμπνευσμένη από τα βιολογικά νευρωνικά δίκτυα, (γενικότερα το κεντρικό νευρικό σύστημα των θηλαστικών και κυρίως από τον εγκέφαλο) και χρησιμοποιούνται για την εκτίμηση ή την προσέγγιση λειτουργιών οι οποίες μπορεί να εξαρτώνται από ένα μεγάλο αριθμό εισόδων και είναι γενικά άγνωστες. Τα τεχνητά νευρωνικά δίκτυα παρουσιάζονται συνήθως ως συστήματα διασυνδεδεμένων νευρώνων τα οποία ανταλλάσσουν μηνύματα μεταξύ τους. Οι μεταξύ τους συνδέσεις έχουν αριθμητικά βάρη τα οποία συντονίζονται με βάση την απόκτηση εμπειρίας από την εκπαίδευση του συστήματος, πράγμα που καθιστά τα νευρωνικά δίκτυα ικανά να μαθαίνουν.

Η επινόηση του πρώτου τεχνητού νευρωνικού δικτύου, το οποίο ονομαζόταν *perceptron* έγινε από έναν ψυχολόγο, τον *Rosenblatt* και αυτή ήταν που έδωσε την ώθηση για μετέπειτα μελέτη του μοντέλου των νευρωνικών δικτύων από

μηχανικούς, μαθηματικούς και φυσικούς κατά τις δεκαετίες του '60 και του '70.

Το *perceptron* είναι η απλούστερη δυνατή μορφή ενός νευρωνικού δικτύου που χρησιμοποιείται για την ταξινόμηση προτύπων τα οποία είναι γραμμικά διαχωρίσιμα. Ουσιαστικά αποτελείται από ένα μεμονωμένο νευρώνα με προσαρμόζομενα συναπτικά βάρη και 'προδιάθεση' (πόλωση). Ο αλγόριθμος που χρησιμοποιείται για τη προσαρμογή των ελεύθερων παραμέτρων αυτού του νευρωνικού πρωτοεμφανίστηκε σε μια διαδικασία μάθησης που αναπτύχθηκε από τον Rosenblatt για το βασιζόμενο στο *perceptron* μοντέλο εγκεφάλου που πρότεινε. Πράγματι ο Ροσενβλαττ απέδειξε ότι εάν τα πρότυπα που χρησιμοποιούνται για την εκπαίδευση του *perceptron* αντλούνται από δύο γραμμικά διαχωρίσιμες κλάσεις, τότε ο αλγόριθμος του *perceptron* συγκλίνει και τοποθετεί την επιφάνεια απόφασης με την μορφή ενός υπερεπιπέδου μεταξύ των δύο κλάσεων. Η δεδομένη απόδειξη είναι γνωστή ως θεώρημα σύγκλισης του *perceptron*.

Εδώ αξίζει να σημειωθεί ότι το *perceptron* που βασίζεται σε έναν μεμονωμένο νευρώνα περιορίζεται στην ταξινόμηση προτύπων που ανήκουν μόνο σε δύο κλάσεις. Επεκτείνοντας το επίπεδο εξόδου του *perceptron* ώστε να περιλαμβάνει περισσότερους του ενός νευρώνες μπορούμε να εκτελούμε ταξινόμηση προτύπων που ανήκουν σε περισσότερες από δύο κλάσεις. Ωστόσο, οι κλάσεις αυτές πρέπει να είναι γραμμικά διαχωρίσιμες για να δουλέψει σωστά το *perceptron*.

Το *perceptron* βασίζεται σε ένα μη γραμμικό νευρώνα, συγκεκριμένα στο μοντέλο νευρώνα McCulloch-Pitts. Το συγκεκριμένο μοντέλο αποτελείται από ένα γραμμικό συνδυαστή ο οποίος ακολουθείται από έναν απότομο περιοριστή που εκτελεί τη συνάρτηση προσίμου ή κάποια σιγμοειδή συνάρτηση. Ο κόμβος άθροισης του νευρωνικού μοντέλου υπολογίζει τον γραμμικό συνδυασμό εισόδων που εφαρμόζονται στις συνάψεις του και ενσωματώνει επίσης, μια εξωτερικά παραγόμενη 'προδιάθεση' ή πόλωση. Το προκύπτον άθροισμα δηλαδή το παραγόμενο τοπικό πεδίο (induced local field), εφαρμόζεται σε έναν απότομο περιοριστή. Ως απόκριση ο νευρώνας παράγει έξοδο ίση με +1 εάν η είσοδος του απότομου περιοριστή είναι θετική και -1 εάν είναι αρνητική.

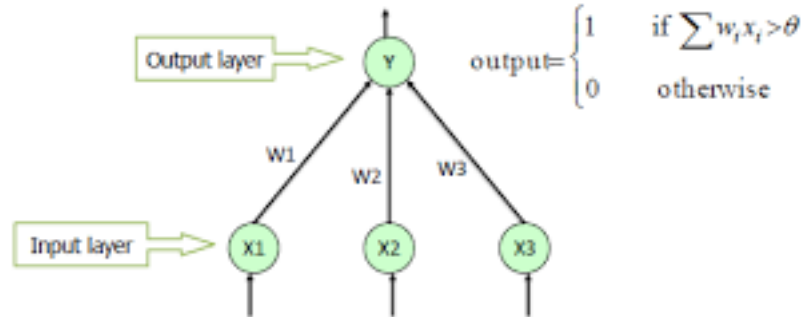
Τα συναπτικά βάρη του *perceptron* συμβολίζονται ως w_1, w_2, \dots, w_m . Αντίστοιχα, οι εισοδοί που εφαρμόζονται στο *perceptron* συμβολίζονται ως x_1, x_2, \dots, x_m . Η εξωτερικά εφαρμοζόμενη πόλωση συμβολίζεται ως b . Από το μοντέλο βρίσκουμε ότι η είσοδος του απότομου περιοριστή ισούται με :

$$u = \sum_{i=1}^m w_i x_i + b \quad (1)$$

Ο στόχος του *perceptron* είναι να ταξινομήσει σωστά το σύνολο των εξω-

τερικά εφαρμοζόμενων διεγέρσεων $x_1 \dots x_m$ σε μια απο τις κλάσεις έστω $C1$ ή $C2$. Ο κανόνας απόφασης για την ταξινόμηση υπαγορεύει την αντιστοίχιση του σημείου που αντιπροσωπεύουν οι είσοδοι στην κλάση $C1$ αν η έξοδος ψ του *perceptron* είναι +1 και στην $C2$ αν είναι -1.

Single Layer Perceptron



2.2. Το perceptron του Rosenblatt

Για τη κατανόηση της συμπεριφοράς του ταξινομητή προτύπων συνηθίζεται να απεικονίζουμε ένα χάρτη των περιοχών απόφασης στον n -διάστατο χώρο σημάτων που καταλαμβάνουν οι k μεταβλητές εισόδου. Στην απλούστερη δυνατή μορφή του *perceptron* υπάρχουν δύο περιοχές απόφασης διαχωριζόμενες από ένα υπερεπίπεδο το οποίο ορίζεται από την

$$\sum_{i=1}^m w_i x_i + b = 0 \quad (2)$$

Στη περίπτωση που έχουμε μόνο δύο μεταβλητές έστω (x_1, x_2) το υπερεπίπεδο απόφασης εκφυλίζεται σε γραμμή απόφασης. Εδώ αξίζει να αναφερθεί ότι η επίδραση της πόλωσης b σχετίζεται με τη μετακίνηση του υπερεπιπέδου απόφασης από την αρχή των αξόνων.

Τα συναπτικά βάρη του *perceptron* μπορούν να προσαρμόζονται με μια επαναληπτική διαδικασία. Για την προσαρμογή μπορούμε να χρησιμοποιήσουμε ένα κανόνα διόρθωσης σφαλμάτων γνωστό και ως αλγόριθμο σύγκλισης του *perceptron*.

Προκειμένου να κατανοήσουμε την λειτουργία μάθησης του *perceptron* θα εξετάσουμε τον αλγόριθμο σύγκλισης του. Θα αντιμετωπίσουμε την πόλωση του νευρωνικού ως ένα συναπτικό βάρος το οποίο οδηγείται από σταθερή είσοδο ίση με +1. Μπορούμε επομένως να ορίσουμε το διάνυσμα εισόδων $(m+1)$ -επι

1 ως:

$$\mathbf{x}(n) = [+1, x_1(n), x_2(n), \dots, x_m(n)]^T$$

όπου το n συμβολίζει το χρονικό βήμα εφαρμογής του αλγορίθμου. Αντίστοιχα , ορίζουμε το διάνυσμα βαρών $(m + 1)$ -επι-1 ως

$$\mathbf{w}(n) = [b, w_1(n), w_2(n), \dots, w_m(n)]^T$$

Κατα συνέπεια ,η έξοδος του γραμμικού συνδυαστή μπορεί να γραφεί ως:

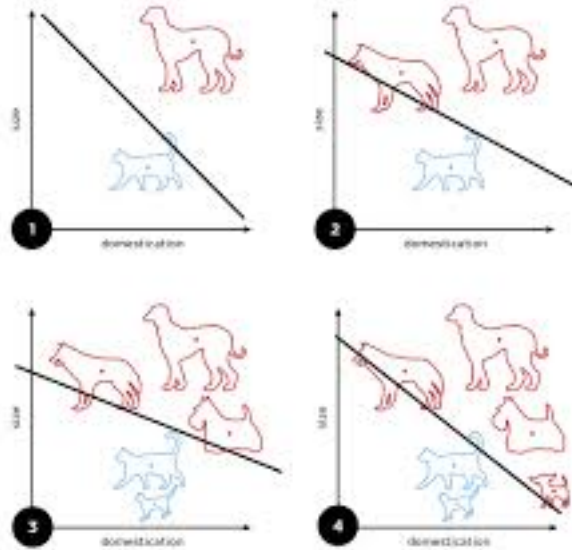
$$u(n) = \sum_{i=0}^m w_i(n)x_i(n) = \mathbf{w}^T(n)\mathbf{x}(n) \quad (3)$$

όπου στην πρώτη γραμμή το $w_0(n)$,που αντιστοιχεί στο $i = 0$ αντιπροσωπεί την πόλωση b . Για σταθερό n η εξίσωση $\mathbf{w}^T(n)\mathbf{x}(n) = 0$ εάν απεικονισθεί σε έναν m -διάστατο χώρο με συντεταγμένες x_1, x_2, \dots, x_m ορίζει ένα υπερεπίπεδο ως διαχωριστική επιφάνεια απόφασης μεταξύ δύο διαφορετικών κλάσεων εισόδων.

Για να λειτουργήσει σωστά το perceptron οι δύο κλάσεις C_1, C_2 πρέπει να είναι γραμμικά διαχωρίσιμες και επομένως ,πρέπει τα προς ταξινόμηση πρότυπα να είναι επαρκώς διαχωρισμένα το ένα απο το άλλο ωστε να διασφαλισθεί ότι η επιφάνεια απόφασης θα είναι ένα υπερεπίπεδο. Εάν οι δύο κλάσεις πλησιάσουν πολύ η μια με την άλλη , τότε το υπερεπίπεδο δεν θα είναι αρκετο για να τις διαχωρίσει, θα γίνουν μη γραμμικά διαχωρίσιμες και η ταξινόμηση τους θα περνάει τις υπολογιστικές δυνατότητες του απλού perceptron.

Θα υποθέσουμε λοιπον ότι οι μεταβλητές εισόδου προέρχονται απο δύο γραμμικά διαχωρίσιμες κλάσεις. Εστω ότι H_1 είναι ο υποχώρος των διανυσμάτων εκπαίδευσης $x_1(1), x_1(2) \dots$ που ανήκουν στη κλάση C_1 και αντίστοιχα H_2 είναι ο υποχώρος των διανυσμάτων εκπαίδευσης $x_2(1), x_2(2) \dots$ που ανήκουν στη κλάση C_2 . Παρατηρούμε ότι η ένωση των H_1, H_2 συνιστά τον πλήρη χώρο εκπαίδευσης(έστω H) δεδομένου ότι έχουμε μόνο δύο κλάσεις. Δοθέντων των H_1 και H_2 για την εκπαίδευση του ταξινομητή , η διαδικασία εκπαίδευσης απαιτεί τη προσαρμογή του διανύσματος βαρών \mathbf{w} με τρόπο ωστε οι δύο κλάσεις να είναι γραμμικά διαχωρίσιμες.Δηλαδή απαιτεί την διαμόρφωση ενός διανύσματος \mathbf{w} ετσι ώστε να μπορούμε να δηλώσουμε:

$$\begin{aligned} \mathbf{w}^T \mathbf{x} &> 0 \forall \mathbf{x} \in C_1 \\ \mathbf{w}^T \mathbf{x} &\leq 0 \forall \mathbf{x} \in C_2 \end{aligned} \quad (4)$$



2.3. Παράδειγμα γραμμικά διαχωρίσιμων προτύπων

Στη δεύτερη γραμμή της εξίσωσης 4 έχουμε αποφασίσει αυθαίρετα ότι το διάνυσμα εισόδων \mathbf{x} θα ανήκει στη κλάση C_2 αν ανήκει στο υπερεπίπεδο. Δοθέντων των υποσυνόλων των διανυσμάτων εκπαίδευσης H_1 και H_2 , το πρόβλημα εκπαίδευσης του perceptron συνίσταται στο να βρούμε ένα διάνυσμα βαρών τέτοιο ώστε να ικανοποιούνται οι ανισότητες της εξίσωσης 4

Μπορούμε επομένως τώρα να διατυπώσουμε τον αλγόριθμο για την προσαρμογή του διανύσματος βαρών του στοιχειώδους perceptron ως εξής:

1. Εάν το n -οστό μέλος του συνόλου εκπαίδευσης, $\mathbf{x}(n)$ ταξινομείται σωστά από το διάνυσμα βαρών $\mathbf{w}(n)$ που υπολογίζεται στη n -οστή επανάληψη του αλγορίθμου δεν γίνεται καμία διόρθωση στο διάνυσμα βαρών του perceptron σύμφωνα με τον κανόνα:

$$\mathbf{w}(n+1) = \mathbf{w}(n) \quad \text{if} \quad \mathbf{w}^T \mathbf{x} > 0 \quad \text{and} \quad \mathbf{x}(n) \in C_1$$

ενώ

$$\mathbf{w}(n+1) = \mathbf{w}(n) \quad \text{if} \quad \mathbf{w}^T \mathbf{x} \leq 0 \quad \text{and} \quad \mathbf{x}(n) \in C_2 \quad (5)$$

2. Διαφορετικά, το διάνυσμα βαρών του perceptron ενημερώνεται σύμφωνα με τον κανόνα :

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \eta(n)\mathbf{x}(n) \quad \text{if} \quad \mathbf{w}(n)^T \mathbf{x}(n) > 0 \quad \text{and} \quad \mathbf{x}(n) \in C_2$$

και

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \eta(n)\mathbf{x}(n) \quad \text{if} \quad \mathbf{w}(n)^T \mathbf{x}(n) \leq 0 \quad \text{and} \quad \mathbf{x}(n) \in C_1 \quad (6)$$

όπου η παράμετρος του ρυθμού μάθησης $\eta(n)$ ελέγχει τη προσαρμογή που εφαρμόζεται στο διάνυσμα βαρών στην επανάληψη n

Η τελική έξοδος του perceptron θα δίνεται ως χβαντισμένη απόκριση $y(n)$ με

$$y(n) = \text{sgn}[\mathbf{w}^T(n)\mathbf{x}(n)] \quad (7)$$

όπου

$$\text{sgn}(u) = \begin{cases} +1 & \text{αν } u > 0 \\ -1 & \text{αν } u < 0 \end{cases} \quad (8)$$

Γενικά παρατηρούμε ότι το διάνυσμα εισόδων $\mathbf{x}(n)$ είναι ένα διάνυσμα $(m+1)$ -επι 1 του οποίου το πρώτο στοιχείο είναι σταθερό και ίσο με +1 καθ'όλη τη διάρκεια του υπολογισμού του αλγορίθμου.

Αντίστοιχα, το διάνυσμα βαρών $\mathbf{w}(n)$ είναι ένα διάνυσμα του οποίου το πρώτο στοιχείο καθόλο τον υπολογισμό του αλγορίθμου θα ισούται με b .

Τελικά η προσαρμογή των βαρών με βάση των αλγόριθμο, από την επανάληψη n στην επανάληψη $n+1$ συνοψίζεται (για σταθερό ρυθμό μάθησης) στη ακόλουθη μορφή:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta[d(n) - y(n)]\mathbf{x}(n) \quad (9)$$

όπου $d(n)$ η χβαντισμένη επιθυμητή απόκριση η οποία ορίζεται ως:

$$d(n) = \begin{cases} +1 & \text{αν } \mathbf{x}(n) \in C_1 \\ -1 & \text{αν } \mathbf{x}(n) \in C_2 \end{cases} \quad (10)$$

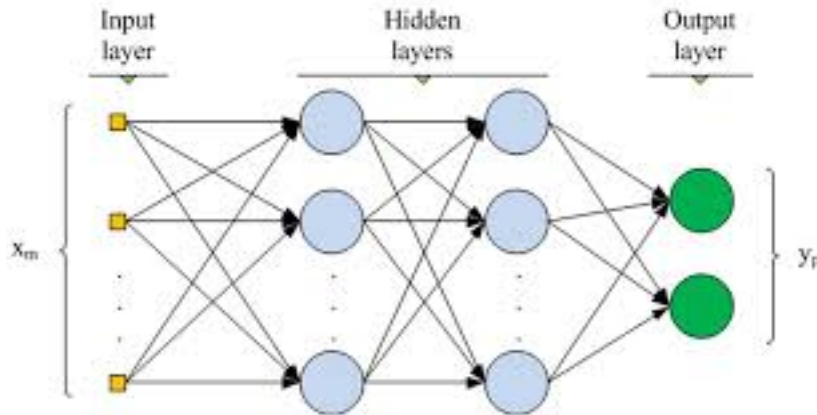
Παρατηρούμε ότι η διαφορά $d(n) - y(n)$ παίζει τον ρόλο ενός σήματος σφάλματος. Η παράμετρος του ρυθμού μάθησης είναι μια θετική σταθερά με $0 < \eta \leq 1$. Όταν της αναθετούμε μια τιμή από αυτό το πεδίο πρέπει να έχουμε υπόψιν δύο αντικρουόμενες απαιτήσεις:

1. υπολογισμό του μεσου όρου των παρελθόντων εισόδων για την παροχή σταθερών εκτιμήσεων για τα βάρη ο οποίος απαιτεί μικρή η
2. γρήγορη προσαρμογή αναφορικά με τις πραγματικές μεταβολές στις υποκείμενες κατανομές της διαδικασίας που είναι που είναι υπεύθυνη για την παραγωγή του διανύσματος εισόδων η οποία απαιτεί μεγάλο η . [1]

Φυσικά θα μπορούσαμε να χρησιμοποιήσουμε ρυθμό μάθησης εξαρτώμενο από το n έτσι ώστε να μπορέσουμε να συνδυάσουμε τα πλεονεκτήματα του μεγάλου συντελεστή (κοντά στον 1, πχ γρήγορη προσαρμογή) με αυτά του μικρού (κοντά στο 0, πιο σταθερή και ομαλή συγκλιση).

Στη συνέχεια θα εξετάσουμε την πιο σύνθετη περίπτωση νευρωνικού δικτύου πολλών επιπέδων η οποία όμως αποτελεί μια πολύ πιο ισχυρή εκδοχή του συστήματος και επομένως αξίζει μελετηθεί διεξοδικά.

Υπάρχουν πολλά προβλήματα στα οποία τα πρότυπα δεν είναι γραμμικά διαχωρίσιμα και επομένως δεν μπορούν να ταξινομηθούν από ένα στοιχειώδες perceptron (όπως για παράδειγμα το πρόβλημα της XOR). Χρησιμοποιώντας πολυεπίεδα νευρωνικά μπορούμε να ξεπεράσουμε αυτό το πρόβλημα, καθώς δίνουν τη δυνατότητα για αυθαίρετες χαρτογραφήσεις στο χώρο των προτύπων.



2.4. Παράδειγμα ενός πολυεπίεδου perceptron

Γενικά, η φύση του ορίου απόφασης εξαρτάται από την τοπολογία του νευρωνικού δικτύου. Επίσης, πρέπει να σημειωθεί ότι ανάλογα με το πόσα επίπεδα έχει ένα πολυεπίεδο νευρωνικό δίκτυο έχει και κλιμακούμενες δυνατότητες ταξινόμησης. Πιο συγκεκριμένα διακρίνουμε τρεις περιπτώσεις [7]

1. Ένα επίπεδο: Είναι ικανό να τοποθετήσει ένα υπερεπίεδο στο χώρο των προτύπων εισόδου
2. Δύο επίπεδα (ένα κρυφό επίπεδο): Είναι ικανό να περιγράψει όριο απόφασης το οποίο περικλύει μια μεμονωμένη κυρτή περιοχή των προτύπων εισόδου.
3. Τρία επίπεδα και πάνω (τουλάχιστον δύο κρυφά επίπεδα): Είναι ικανό να παράγει αυθαίρετα όρια απόφασης.

| STRUCTURE | TYPES OF DECISION REGIONS | EXCLUSIVE OR PROBLEM | CLASSES WITH MESHED REGIONS | MOST GENERAL REGION SHAPES |
|------------------|---|----------------------|-----------------------------|----------------------------|
| SINGLE-LAYER | HALF PLANE BOUNDED BY HYPERPLANE | | | |
| TWO-LAYER | CONVEX OPEN OR CLOSED REGIONS | | | |
| THREE-LAYER | ARBITRARY (Complexity Limited By Number of Nodes) | | | |

2.5. Παραδείγματα ορίων απόφασης για διάφορο αριθμό επιπέδων για πολυεπίπεδα perceptron

Απο τα προηγούμενα καταλαβαίνουμε πως ο αριθμός των επιπέδων του νευρωνικού δικτύου επηρεάζει άμεσα τα όρια απόφασης που θα παραχθούν. Στην επιλογή του αριθμού επιπέδων πρέπει να ληφθούν υπόψη τα παρακάτω:

Τα πολυεπίπεδα perceptron είναι πιο δύσκολο να εκπαιδευτούν απο τα απλούστερα νευρωνικά ενός επιπέδου

Ενα νευρωνικό δίκτυο δυο επιπέδων με σιγμοειδείς συναρτήσεις ενεργοποίησης (στο απλό perceptron πιο πριν χρησιμοποιήσαμε την $sgn(u)$) είναι ικανό να μοντελοποιήσει οποιοδήποτε όριο απόφασης.

Ενας ακόμα προβληματισμός σχετίζεται με τον αριθμό των νευρώνων ανα επίπεδο. Γενικά ο αριθμός των νευρώνων του επιπέδου εξόδου σχετίζεται με τον αριθμό των κλάσεων εξόδου. Αντίστοιχα ο αριθμός των νευρώνων του επιπέδου εισόδου εξαρτάται απο τον αριθμό της διάστασης των προτύπων τα οποία θα μπούν ως είσοδοι στο σύστημα. Τέλος ο αριθμός των ενδιάμεσων κρυφών νευρώνων, είναι σχεδιαστική απόφαση. Εδώ πρέπει να θυμόμαστε οτι αν έχουμε λίγους νευρώνες το σύστημα δεν θα μπορεί να δημιουργήσει περίπλοκα όρια απόφασης ,ενώ αν είναι πάρα πολλοί το σύστημα δεν θα μπορεί να γενικεύει ικανοποιητικά.[7] Γενικά πάντως οι κρυφοί νευρώνες δρουν ως ανιχνευτές χαρακτηριστικών και ως τέτοιοι παίζουν σημαντικό ρόλο στη λειτουργία ενος πολυεπίπεδου νευρωνικού δικτύου. Καθώς προχωρά η διαδικασία μάθησης οι κρυφοί νευρώνες αρχίζουν σταδιακά να ανακαλύπτουν τα εξέχοντα χαρακτηριστικά -αυτά δηλαδή που χαρακτηρίζουν τα δεδομένα εκπαίδευσης. Αυτό το κάνουν εκτελώντας μη γραμμικό μετασχηματισμό στα δεδομένα εισόδου,σε ένα νεο χώρο που αποκαλείται χώρος χαρακτηριστικών. Σε αυτό το

νέο χώρο οι κλάσεις χαρακτηριστικών που ενδιαφέρουν μια συγκεκριμένη εργασία ταξινόμησης προτύπων μπορούν να ξεχωρίσουν από οτιδήποτε άλλο θα μπορούσε να υπάρχει στον αρχικό χώρο των δεδομένων εισόδου.

Σε κάθε περίπτωση, το μεγαλύτερο ζήτημα για το πολυεπίπεδο perceptron είναι όπως προαναφέρθηκε η εκπαίδευση του. Δεδομένου ότι παρουσιάζει μια κατανεμημένη μορφή μη γραμμικότητας με συνήθως υψηλή διασυνδεσημότητα, η θεωρητική ανάλυση του καθίσταται δύσκολη υπόθεση.

Μια δημοφιλής μέθοδος για την εκπαίδευση των perceptron πολλών επιπέδων είναι ο αλγόριθμος *BP* (*backpropagation*). Κατά τον αλγόριθμο αυτό η εκπαίδευση λαμβάνει χώρα σε δύο φάσεις:

1. Στη φάση που εξελίσσεται προς τα εμπρός, τα συναπτικά βάρη του δικτύου είναι σταθερά και το σήμα εισόδου διαδίδεται διαμέσου του δικτύου επίπεδο προς επίπεδο μέχρι να φτάσει στη έξοδο. Συνεπώς σε αυτή τη φάση οι αλλαγές περιορίζονται στα δυναμικά ενεργοποίησης (διαμέσου των συναρτήσεων ενεργοποίησης του κάθε νευρώνα) και στις εξόδους των νευρώνων του δικτύου.
2. Στη φάση που εξελίσσεται προς τα πίσω, όπου παράγεται ένα σήμα σφάλματος μέσω της σύγκρισης της εξόδου του δικτύου με μια επιθυμητή απόκριση. Το σήμα σφάλματος διαδίδεται διαμέσου του δικτύου ξανά επίπεδο προς επίπεδο αλλά αυτή τη φορά η διάδοση γίνεται με κατεύθυνση προς τα πίσω. Σε αυτή τη δεύτερη φάση γίνονται διαδοχικές προσαρμογές στα συναπτικά βάρη του δικτύου. Ο υπολογισμός των προσαρμογών για το επίπεδο εξόδου είναι απλή υπόθεση ενώ για τα κρυφά επίπεδα γίνεται πολύ πιο δύσκολος.
Σε αυτή τη φάση όπως αναφέρθηκε, το προκύπτον σήμα ονομάζεται σήμα σφάλματος και αυτό γιατί ο υπολογισμός του καθώς διαδίδεται από νευρώνα σε νευρώνα απαιτεί τη χρήση μιας εξαρτώμενης από το αρχικό σφάλμα συνάρτησης.

Αξίζει να αναφερθεί ότι η χρήση του όρου *backpropagation* εμφανίστηκε στο προσκήνιο μετά το 1985, όταν ο όρος απέκτησε ευρεία απήχηση χάρις το βιβλίο με τίτλο *Parallel Distributed Processing*. Η αναπτυξη του αλγορίθμου *BP* στα μέσα της δεκαετίας του '80 αποτέλεσε ένα ορόσημο στην εξέλιξη του κλάδου των νευρωνικών δικτύων διότι παρείχε μια υπολογιστικά αποτελεσματική μέθοδο για την εκπαίδευση των πολυεπίπεδων νευρωνικών αίροντας έτσι τον έως τότε πεσιμισμό που υπήρχε σχετικά με τις δυνατότητες μάθησης τους.

Περιγράφουμε τώρα πιο συγκεκριμένα τα βήματα του αλγορίθμου *BP*. Υποθέτουμε ότι ο αλγόριθμος διατρέχει το δείγμα εκπαίδευσης $[(\mathbf{x}(n), \mathbf{d}(n))]_{n=1}^N$:

1. *Αρχικοποίηση.* Υποθέτοντας ότι δεν είναι διαθέσιμη πρότερη πληροφορία επιλέγουμε τα συναπτικά βάρη και τα κατώφλια από μία ομοιόμορφη κατανομή, της οποίας ο μέσος να είναι μηδέν και της οποίας η διακύμανση επιλέγεται με τρόπο ώστε να κάνει την τυπική απόκλιση των πεδίων των νευρώνων να βρίσκεται στο όριο μεταξύ του γραμμικού και του σταθερού μέρους της σιγμοειδούς συνάρτησης ενεργοποίησης.
2. *Παρουσιάσεις παραδειγμάτων εκπαίδευσης.* Παρουσιάζουμε στο δίκτυο μια εποχή παραδειγμάτων εκπαίδευσης. Για κάθε παράδειγμα που περιλαμβάνει το δείγμα (οργανωμένο με κάποιο τρόπο), εκτελούμε την ακολουθία των προς τα εμπρός και προς τα πίσω υπολογισμών που περιλαμβάνονται στα βήματα 3 και 4 αντίστοιχα.
3. *Υπολογισμός προς τα εμπρός.* Εστω ότι ένα παράδειγμα εκπαίδευσης στη συγκεκριμένη εποχή συμβολίζεται ως $(\mathbf{x}(n), \mathbf{d}(n))$ με το διάνυσμα εισόδων $\mathbf{x}(n)$ να εφαρμόζεται στο επίπεδο εισόδου των αισθητηριακών κόμβων και το διάνυσμα των επιθυμητών αποκρίσεων $\mathbf{d}(n)$ να παρουσιάζεται στο επίπεδο εξόδου των υπολογιστηκών κόμβων. Υπολογίζουμε τα τοπικά πεδία και τα λειτουργικά σήματα του δικτύου προχωρώντας προς τα εμπρός, σε όλη την έκταση του δικτύου, επίπεδο προς επίπεδο. Το τοπικό πεδίο $u_j^{(l)}(n)$ για τον νευρώνα j στο επίπεδο l είναι:

$$u_j^{(l)}(n) = \sum_i w_{ji}^{(l)}(n) y_i^{(l-1)}(n) \quad (11)$$

όπου $y_i^{(l-1)}(n)$ είναι το σήμα εξόδου (λειτουργικό) του νευρώνα i του προηγούμενου επιπέδου $l - 1$ στην επανάληψη n και $w_{ji}^{(l)}(n)$ είναι το συναπτικό βάρος του νευρώνα j του επιπέδου l που τροφοδοτείται από τον νευρώνα i στο επίπεδο $l - 1$. Για $i = 0$, έχουμε $y_0^{(l-1)}(n) = +1$ και $w_{j0}^{(l)}(n) = b_j^{(l)}(n)$ είναι η πόλωση που εφαρμόζεται στο νευρώνα j στο επίπεδο l . Υποθέτουμε ότι χρησιμοποιούμε μια σιγμοειδή συνάρτηση το σήμα εξόδου του νευρώνα j στο επίπεδο l είναι:

$$y_j^{(l)} = \varphi_j(u_j(n)) \quad (12)$$

Εάν ο νευρώνας j είναι στο πρώτο κρυφό επίπεδο (δηλαδή $l = 1$) θέτουμε:

$$y_j^{(0)} = x_j(n) \quad (13)$$

όπου $x_j(n)$ είναι το j -οστό στοιχείο του δεδομένου διανύσματος εισόδου $\mathbf{x}(n)$. Αν ο νευρώνας j είναι στο επίπεδο εξόδου (δηλαδή $l = L$ όπου το

L αναφέρεται αναφέρεται ως βάθος του δικτύου) θεωρούμε:

$$y_j^{(L)} = o_j(n) \quad (14)$$

όπου $o_j(n)$ είναι το λειτουργικό σήμα στην έξοδο του νευρώνα j (στο τελευταίο επίπεδο του νευρωνικού-σήμα εξόδου). Κατόπιν υπολογίζουμε το σήμα σφάλματος:

$$e_j(n) = d_j(n) - o_j(n) \quad (15)$$

όπου $d_j(n)$ είναι το j -οστο στοιχείο του διανύσματος ιδανικών αποκρίσεων $\mathbf{d}(n)$.

4. Υπολογισμός προς τα πίσω. Υπολογίζουμε τα δ του δικτύου (δηλαδή τις τοπικές κλίσεις) που ορίζονται ως:

$$\delta_j^{(l)}(n) = \begin{cases} e_j^{(L)}(n) \varphi_j'(u_j^{(L)}(n)) & \text{για το νευρώνα } j \text{ στο επίπεδο εξόδου } L \\ \varphi_j'(u_j^{(l)}(n)) \sum_k \delta_k^{(l+1)}(n) w_{kj}^{(l+1)}(n) & \text{για το νευρώνα } j \text{ στο κρυφό επίπεδο } l \end{cases} \quad (16)$$

όπου το σύμβολο πρώτης παραγώγου στην $\varphi_j^{(l)}(\cdot)$ συμβολίζει διαφορίσιμη αναφορικά ως προς το όρισμα. Στη συνέχεια προσαρμόζουμε τα συναπτικά βάρη στο επίπεδο l σύμφωνα με τη σχέση:

$$w_{ji}^{(l)}(n+1) = w_{ji}^{(l)}(n) + \eta \delta_j^{(l)}(n) y_i^{(l-1)}(n) \quad (17)$$

όπου η είναι η παράμετρος του ρυθμού μάθησης.

5. Επανάληψη. Επανάλαβε τους υπολογισμούς των βημάτων 3 και 4 παρουσιάζοντας νέες εποχές παραδειγμάτων εκπαίδευσης στο δίκτυο μέχρι να ικανοποιηθεί το επιλεγμένο κριτήριο τερματισμού.

Γενικά, η σειρά παρουσίασης των παραδειγμάτων εκπαίδευσης θα πρέπει να είναι τυχαία απο εποχή σε εποχή. Επίσης, συνήθως η παράμετρος του ρυθμού μάθησης προσαρμόζεται (συνήθως μειώνεται) καθώς αυξάνεται ο αριθμός των επαναλήψεων.

Σχετικά με τον κανόνα 4, ο τρόπος με τον οποίο υπολογίζεται η σχέση για τις κλίσεις σχετίζεται με το γεγονός ότι η διόρθωση $\Delta w_{ji}(n)$ στο συναπτικό βάρος $w_{ji}(n)$ είναι ανάλογη με τη μερική παράγωγο $\frac{\partial E(n)}{\partial w_{ji}(n)}$ έτσι ώστε :

$$\Delta w_{ji}(n) = -\eta \frac{\partial E(n)}{\partial w_{ji}(n)} \quad (18)$$

όπου το E αποτελεί τη στιγμιαία ενέργεια σφάλματος ολόκληρου του δικτύου και ορίζεται ως:

$$E(n) = \sum_{j \in C} E_j(n) \quad (19)$$

με E_j να αποτελεί τη στιγμιαία ενέργεια σφάλματος για τον νευρώνα j με C να είναι το σύνολο των νευρώνων εξόδου. Το E_j ορίζεται ως

$$E_j(n) = \frac{1}{2} e_j^2(n)$$

με

$$e_j(n) = d_j(n) - y_j(n) \quad (20)$$

Από όλα τα παραπάνω και αξιοποιώντας τις σχέσεις (11),(12),(19),(20) και εφαρμόζοντας τον κανόνα αλυσίδας καταλήγουμε στη σχέση:

$$\frac{\partial E(n)}{\partial w_{ji}(n)} = \frac{\partial E(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial u_j(n)} \frac{\partial u_j(n)}{\partial w_{ji}(n)} \quad (21)$$

Χρησιμοποιώντας τις παραπάνω σχέσεις και υπολογίζοντας τη κάθε μια από τις παραπάνω μερικές παραγώγους, καταλήγουμε στην ακόλουθη σχέση:

$$\frac{\partial E(n)}{\partial w_{ji}(n)} = -e_j(n) \varphi_j'(u_j(n)) y_j(n) \quad (22)$$

και επομένως λόγω της (18) καταλήγουμε:

$$\Delta w_{ji}(n) = \eta e_j(n) \varphi_j'(u_j(n)) y_j(n) \quad (23)$$

Ορίζουμε την τοπική κλίση $\delta_j(n)$ ως:

$$\delta_j(n) = \frac{\partial E(n)}{\partial u_j(n)} = e_j(n) \varphi_j'(u_j(n)) \quad (24)$$

όπου το πρώτο μέρος της σχέσης αφορά την πιο γενική περίπτωση και το δεύτερο τη συγκεκριμένη περίπτωση. Παίρνοντας την παραπάνω σχέση και κατόπιν επεξεργασίας για την περίπτωση των κρυφών νευρώνων καταλήγουμε στη σχέση (16) που αποτελεί σε συνδυασμό με τη (17) τον πυρήνα της οπισθοδιάδοσης σφάλματος στην οποία βασίζεται ο αλγόριθμος *BP*. [1]

Μια ακόμα ιδιαίτερα σημαντική ιδιότητα των νευρωνικών δικτύων είναι αυτή της γενίκευσης. Ένα δίκτυο λέγεται ότι γενικεύει καλά όταν η αντιστοίχιση εισόδου εξόδου υπολογίζεται από ένα δίκτυο είναι σωστή (ή σχεδόν σωστή) για δεδομένα ελέγχου τα οποία δεν χρησιμοποιήθηκαν ποτέ κατά την εκπαίδευση ή την δημιουργία του δικτύου. Η ιδιότητα αυτή αποτελεί και μια από τις

ουσιαστικότερες αιτίες χρήσης νευρωνικών δικτύων δεδομένου ότι σαν συστημα γενικεύουν ικανοποιητικά. Αξίζει να σημειωθεί ότι τα δεδομένα ελέγχου υποθέτουμε ότι αντλούνται από τον ίδιο πλυθησμό από όπου προήλθαν τα δεδομένα εκπαίδευσης.

Η διαδικασία μάθησης (η εκπαίδευση ενός νευρωνικού δικτύου) μπορεί να αντιμετωπισθεί σαν ένα πρόβλημα προσαρμογής καμπύλης. Το ίδιο το δίκτυο μπορεί να θεωρηθεί ως μια μη γραμμική αντιστοίχιση εισόδου - εξόδου. Μια τέτοια θεώρηση μας επιτρέπει να εξετάσουμε την γενίκευση όχι σαν μια μυστηριώδη ιδιότητα αλλά απλώς σαν το αποτέλεσμα της εφαρμογής μιας καλής μη γραμμικής παρεμβολής στα δεδομένα εισόδου. Το δίκτυο εκτελεί μια χρησιμη λειτουργία παρεμβολής κυρίως επειδή τα *perceptron* πολλών επιπέδων με συνεχείς συναρτήσεις ενεργοποίησης έχουν επίσης συναρτήσεις εξόδου που είναι συνεχείς.

Ένα νευρωνικό δίκτυο που είναι σχεδιασμένο ώστε να γενικεύει καλά, θα παράγει σωστή αντιστοίχιση εισόδου - εξόδου ακόμα και όταν η είσοδος είναι ελαφρώς διαφορετική από τα παραδείγματα που χρησιμοποιήθηκαν κατά την εκπαίδευση του. Ωστόσο όταν ένα νευρωνικό δίκτυο μαθαίνει υπερβολικά πολλά παραδείγματα εισόδου - εξόδου μπορεί να οδηγηθεί σε μια κατάσταση όπου θα αποθηκεύει τα δεδομένα εκπαίδευσης και θα λειτουργεί μόνο βάσει αυτών. Αυτό μπορεί να συμβεί εάν βρεί κάποιο χαρακτηριστικό (πχ οφειλόμενο σε θόρυβο) το οποίο υπάρχει στα δεδομένα εκπαίδευσης, αλλά όχι στην υποκείμενη συνάρτηση που πρέπει να μοντελοποιήσει. Αυτό το φαινόμενο είναι γνωστό και ως υπερπροσαρμογή (*overfit*) ή υπερεκπαίδευση (*overtrain*) του δικτύου. Όταν το δίκτυο είναι υπερεκπαιδευμένο (εστιασμένο αποκλειστικά στα δεδομένα εκπαίδευσης) χάνει τη δυνατότητα να γενικεύει βασιζόμενο σε παρόμοια πρότυπα εισόδου-εξόδου.

Συνήθως η φορτώση δεδομένων κατά αυτό τον τρόπο σε ένα *perceptron* πολλών επιπέδων απαιτεί τη χρήση περισσότερων νευρώνων από όσους είναι πραγματικά αναγκαίοι, με αποτέλεσμα, οι ανεπιθύμητες συνεισφορές του θορύβου στο χώρο εισόδου να αποθηκεύονται σε συναπτικά βάρη του δικτύου.

2.2. Ταξινομητής K-NN

Ο ταξινομητής KNN είναι ένας από εκείνους τους αλγόριθμους που είναι πολύ απλός για να τον καταλάβουμε, αλλά που λειτουργεί πάρα πολύ καλά στην πράξη. Επίσης, είναι εκπληκτικά ευέλικτος και οι εφαρμογές του κυμαίνονται από όραση μέχρι πρωτεΐνες και από υπολογιστική γεωμετρία μέχρι γραφικές παραστάσεις. Οι περισσότεροι άνθρωποι μαθαίνουν τον αλγόριθμο και δεν το χρησιμοποιούν πολύ, πράγμα το οποίο είναι κρίμα, αφού η έξυπνη χρήση του

KNN μπορεί να κάνει τα πράγματα πολύ απλά. Επίσης, είναι εντυπωσιακό ότι ο KNN έχει εκλεγεί από την IEEE ως ένας από τους δεκα καλύτερους αλγόριθμους εξόρυξης δεδομένων.

Ο KNN είναι ένας μη παραμετρικός, σκληρός αλγόριθμος μάθησης. Όταν αναφερόμαστε σε μια τεχνική που είναι μη παραμετρική, αυτό σημαίνει ότι δεν κάνουμε οποιεσδήποτε υποθέσεις σχετικά με την υποκείμενη κατανομή των δεδομένων. Αυτό είναι αρκετά χρήσιμο, καθώς στον πραγματικό κόσμο, τα περισσότερα από τα πρακτικά δεδομένα δεν υπακούνε στις τυπικές θεωρητικές παραδοχές (π.χ γκαουσιανές κατανομές, γραμμική διαχωρισιμότητα κλπ). Σε τέτοιες περιπτώσεις μη παραμετρικοί αλγόριθμοι όπως ο KNN μπορούν να σώσουν την κατάσταση.

Ο KNN επίσης ένας σκληρός αλγόριθμος. Αυτό σημαίνει ότι δεν χρησιμοποιεί τα δεδομένα εκπαίδευσης για να κάνει οποιαδήποτε γενίκευση. Με άλλα λόγια, δεν υπάρχει ρητή φάση εκπαίδευσης ή είναι πολύ ελάχιστη. Αυτό σημαίνει ότι η φάση αυτή είναι αρκετά γρήγορη. Έλλειψη γενίκευσης σημαίνει ότι ο KNN διατηρεί όλα τα δεδομένα εκπαίδευσης. Πιο συγκεκριμένα, όλα τα δεδομένα εκπαίδευσης είναι απαραίτητα κατά τη διάρκεια της φάσης των δοκιμών. Αυτό έρχεται σε αντίθεση με άλλες τεχνικές όπως των SVM, όπου μπορούμε να απορρίψουμε κάποια δεδομένα που δεν αποτελούν, για παράδειγμα διανύσματα υποστήριξης, χωρίς πρόβλημα. Οι περισσότεροι από τους σκληρούς αλγόριθμους - και ειδικά ο KNN - παίρνουν την απόφαση με βάση το σύνολο των διαθέσιμων δεδομένων εκπαίδευσης (στην καλύτερη περίπτωση ένα υποσύνολο τους).

Η ανταλλαγή είναι αρκετά προφανής εδώ - Υπάρχει μια ανύπαρκτη ή ελάχιστη φάση εκπαίδευσης, αλλά μια δαπανηρή φάση ελέγχου. Το κόστος σχετίζεται με χρόνο και μνήμη. Μπορεί να χρειαστεί πολύ χρόνο, καθώς στη χειρότερη περίπτωση, όλα τα σημεία δεδομένων μπορεί να συμμετέχουν στην απόφαση. Επίσης είναι απαραίτητη μεγάλη μνήμη καθώς χρειάζεται να αποθηκευτούν όλα τα δεδομένα εκπαίδευσης.

Πριν δούμε συγκεκριμένα τον αλγόριθμο KNN, ας επανεξετάσουμε κάποιες από τις παραδοχές για αυτόν οι οποίες σχετίζονται και με τους υπόλοιπους ταξινομητές που έχουν εξετασθεί μέχρι τώρα.

Ο KNN προϋποθέτει ότι τα δεδομένα βρίσκονται σε ένα χώρο χαρακτηριστικών. Επομένως, τα σημεία των δεδομένων είναι σε ένα μετρικό χώρο πράγμα που σημαίνει ότι υπάρχουν κάποιες έννοιες απόστασης - με την Ευκλείδεια να είναι η πιο συχνά χρησιμοποιούμενη.

Κάθε ένα από τα δεδομένα εκπαίδευσης αποτελείται από ένα σύνολο χαρακτηριστικών, τα οποία συνιστούν ένα διάνυσμα και την αντίστοιχη ετικέτα κλάσης που σχετίζεται με κάθε διάνυσμα. Στην απλούστερη περίπτωση η ετικέτα αυτή, θα είναι είτε + ή - (για θετική ή αρνητική κλάση) αν και ο KNN, μπορεί να

λειτουργήσει εξίσου καλά με αυθαίρετο αριθμό κλάσεων.

Μας δίνεται επίσης ένας μοναδικός αριθμός k . Ο αριθμός αυτός αποφασίζει πόσοι γείτονες (όπου ο γείτονας ορίζεται με βάση την μετρική απόστασης) θα επηρεάσουν τη ταξινόμηση του κάθε προτύπου ελέγχου . Αν το πλήθος των κλάσεων είναι άρτιο χρησιμοποιείται συνήθως περριτός αριθμος για k . Αν $k = 1$, τότε ο αλγόριθμος καλείται απλά ο αλγόριθμος πλησιέστερου γείτονα.

Ας δούμε τώρα πώς μπορούμε να χρησιμοποιήσουμε τον KNN για ταξινόμηση προτύπων. Μας δίνονται κάποια σημεία δεδομένων για την εκπαίδευση, αλλά και ένα νέο μη ταξινομημένο πρότυπο για δοκιμή . Στόχος μας είναι να βρούμε την κλάση για το νέο σημείο . Ο αλγόριθμος έχει διαφορετική συμπεριφορά ανάλογα με τα διαφορετικά k .

1. $k = 1$ ή κανόνας πλησιέστερου γείτονα:

Αυτή είναι η απλούστερη περίπτωση. Έστω ξ είναι το σημείο που πρέπει να ταξινομηθεί. Βρισκουμε το σημείο που βρίσκεται πλησιέστερα προς χ . Έστω ψ . Τώρα με βάση τον κανόνα πλησιέστερου γείτονα ορίζουμε την ετικέτα του ψ στο χ .

Εάν ο αριθμός των σημείων δεδομένων είναι αρκετά μεγάλος, τότε υπάρχει πολύ μεγάλη πιθανότητα οι ετικέτες του χ και ψ να είναι ίδιες. Ένα παράδειγμα θα μπορούσε να βοηθήσει: Ας πούμε ότι έχουμε ένα (δυσνητικά) προκατειλημμένο κέρμα. Έστω ότι το πετάμε ένα εκατομμύριο φορές και τις 900000 έρχεται κορώνα. Τότε είναι πιθανότερο ότι αν ξαναρίξουμε το κέρμα θα έρθει πάλι κορώνα. Μπορούμε να χρησιμοποιήσουμε ένα παρόμοιο επιχείρημα εδώ.

Ας προσπαθήσουμε να αιτιολογήσουμε πιο αναλυτικά αυτή τη συμπεριφορά. Ας υποθέσουμε ότι όλα τα σημεία βρίσκονται σε ένα χώρο με επαρκώς μεγάλο αριθμό σημείων. Αυτό σημαίνει ότι η πυκνότητα του χώρου σε οποιοδήποτε υποχώρο του είναι αρκετά υψηλή. Με άλλα λόγια, σε οποιοδήποτε υπόχωρο υπάρχει επαρκής αριθμός σημείων. Σκεφτείτε ένα σημείο χ στο υποχώρο που έχει επίσης πολλούς γείτονες. Τώρα έστω ψ ο πλησιέστερος γείτονας. Αν χ και ψ είναι αρκετά κοντά, τότε μπορούμε να υποθέσουμε ότι η πιθανότητα ότι τα χ και ψ ανήκουν στην ίδια κατηγορία είναι αρκετά μεγάλη, πράγμα το οποίο προκύπτει από τη θεωρία απόφασης.

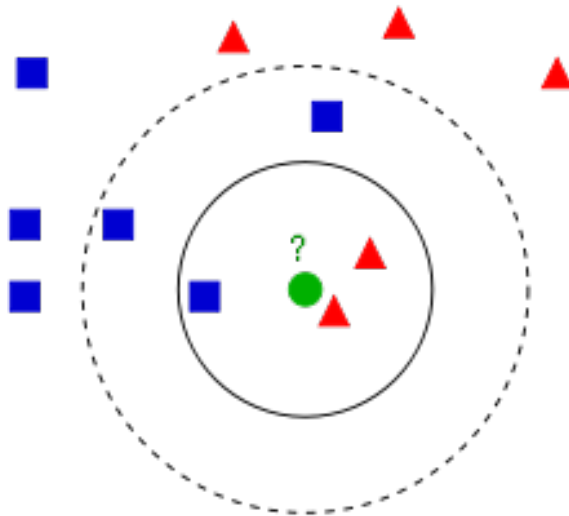
2. $k = K$. Κανόνας πλησιέστερου γείτονα.

Αυτή είναι μια απλή επέκταση της πρώτης περίπτωσης. Ουσιαστικά αυτό που κάνουμε είναι να προσπαθήσουμε να βρούμε τους k πλησιέστερους γείτονες και να κάνουν μια ψηφοφορία. Συνήθως ο k είναι περιττός όταν ο αριθμός των κλάσεων είναι 2. Ας υποθέσουμε ότι $k = 5$ και υπάρχουν

3 πρότυπα που ανήκουν στη κλάση $C1$ και 2 που ανήκουν στη $C2$. Σε αυτήν την περίπτωση ,ο KNN λέει ότι το νέο σημείο πρέπει να επισημανθεί ως $C1$, καθώς αυτή η κατηγορία έχει τα περισσότερα πρότυπα. Ακολουθούμε ένα παρόμοιο επιχείρημα όταν υπάρχουν πολλαπλές κατηγορίες .

Μία ακόμα επέκταση είναι να δώσουμε πάνω απο μία ψήφο σε κάποιους γείτονες. Ένα πολύ συνηθισμένο πράγμα που θα μπορούσαμε να κάνουμε είναι σταθμισμένο KNN όπου κάθε σημείο έχει ένα βάρος το οποίο συνήθως υπολογίζεται με βάση την απόστασή του . Για παράδειγμα σύμφωνα με την στάθμιση αντίστροφης απόστασης , κάθε σημείο έχει ένα βάρος ίσο με το αντίστροφο της απόστασης του απο σημείο που πρέπει να ταξινομηθεί. Αυτό σημαίνει ότι οι γειτονικές μονάδες έχουν μεγαλύτερη βαρύτητα από τα μακρύτερα σημεία .

Επίσης αξίζει να σημειωθεί ότι η ευστοχία του ταξινομητή ενδέχεται να βελτιωθεί με την αύξηση του k , αλλά επίσης αυξάνεται και ο χρόνος υπολογισμού.[8]



2.6.Παραδειγμα εφαρμογής του αλγορίθμου KNN. Αν $k = 3$ τότε ανήκει στη κοκκινη κατηγορία,ενώ αν $k = 5$ στη μπλε

2.3. Ταξινομητής K- means

Ο αλγόριθμος $k - means$ είναι μια απλή επαναληπτική μέθοδος για να διαιρεθεί ένα ορισμένο σύνολο δεδομένων από το χρήστη σε ένα καθορισμένο αριθμό συστάδων k . Αυτός ο αλγόριθμος έχει ανακαλυφθεί από αρκετούς ερευνητές από διαφορετικές ειδικότητες και γενικά τοποθετείται στην ευρύτερη κατηγορία αλγορίθμων τύπου *hill climbing*.

Ο αλγόριθμος λειτουργεί σε ένα σύνολο από d -διάστατα διανύσματα

$$D = \{\mathbf{x}_i | i = 1 \dots N\}$$

όπου το $\mathbf{x}_i \in R^d$ υπονοεί το i -οστό διάνυσμα και σημείο του χώρου. Ο αλγόριθμος ξεκινά με την επιλογή k σημείων που ανήκουν στον R^d ως τα αρχικά k σημεία το καθένα από τα οποία εκπροσωπεί μία συστάδα (*cluster*). Τεχνικές για την επιλογή αυτών των αρχικών τιμών περιλαμβάνουν τυχαία δειγματοληψία από το σύνολο δεδομένων και στη συνέχεια ομαδοποίηση ενός μικρού υποσυνόλου των δεδομένων, καθώς και μεταβολή του μέσου όρου των δεδομένων k φορές και τη λήψη των εκπροσώπων από αυτούς τους μέσους όρους. Στη συνέχεια, ο αλγόριθμος επαναλαμβάνει τα δύο ακόλουθα βήματα μέχρι τη σύγκλιση:

1. Βήμα ανάθεσης δεδομένων. Κάθε σημείο ανατίθεται στον κοντινότερο εκπροσωπο με δεσμούς οι οποίοι αλλάζουν αυθαίρετα. Αυτό έχει σαν αποτέλεσμα τον διαχωρισμό των δεδομένων
2. Βήμα μετεγκατάστασης των μέσων (αντιπροσώπων). Κάθε αντιπρόσωπος σύστάδας μεταφέρεται στο κέντρο (μέσος όρος) όλων των σημείων δεδομένων που του έχουν ανατεθεί. Αν τα σημεία δεδομένων έρχονται με ένα μέτρο πιθανότητας (βάρη), τότε η μετεγκατάσταση σχετίζεται με το σταθμισμένο μέσο όρο δεδομένων της συστάδας.

Ο αλγόριθμος συγκλίνει όταν οι αναθέσεις (έστω από εδώ και πέρα $c_j, j \in [1, k]$) δεν αλλάζουν περαιτέρω. Αξίζει να σημειωθεί ότι η κάθε επανάληψη χρειάζεται $N \times k$ συγκρίσεις, πράγμα το οποίο καθορίζει εν μέρει τη συνολική χρονική πολυπλοκότητα του αλγορίθμου δεδομένου ότι ο αριθμός των επαναλήψεων για σύγκλιση μπορεί να ποικίλει και γενικά εξαρτάται από το N . Πάντως σαν μια πρώτη προσέγγιση ο αλγόριθμος μπορεί να θεωρηθεί γραμμικός ως προς το μέγεθος των δεδομένων εισόδου.

Ένα ζήτημα προς επίλυση είναι το πώς να ποσοτικοποιηθεί το 'πιο κοντά' στο στάδιο ανάθεσης. Το πιο συνήθες μέτρο εγγύτητας είναι η Ευκλείδεια απόσταση

, οπότε μπορεί κανείς να δείξει αμέσως ότι η μη αρνητική συνάρτηση κόστους

$$\sum_{i=1}^N \operatorname{argmin}_j (\|\mathbf{x}_i - \mathbf{c}_j\|) \quad (25)$$

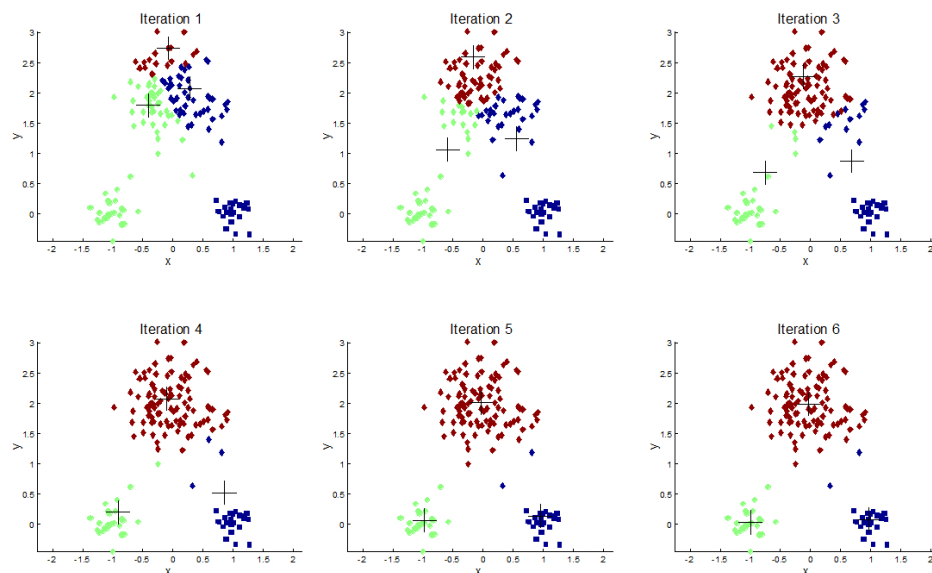
θα μειώνεται όποτε υπάρχει αλλαγή στα βήματα εκχώρησης ή μετεγκατάστασης και ως εκ τούτου η σύγκλιση είναι εγγυημένη σε έναν πεπερασμένο αριθμό επαναλήψεων. Η άπληστη, καθοδική φύση του αλγορίθμου πάνω σε ένα μή κυρτό κόστος συνεπάγεται επίσης ότι η σύγκλιση είναι μόνο σε ένα τοπικό βέλτιστο και επομένως ο αλγόριθμος είναι τυπικά αρκετά ευαίσθητος στις αρχικές τοποθεσίες των αντιπροσώπων καθώς αυτές θα προσδιορίσουν το τοπικό βέλτιστο όπου αυτός θα συγκλίνει. Αυτό το πρόβλημα μπορεί εν μέρει να αντιμετωπισθεί είτε με το να τρέξουμε τον αλγόριθμο πολλές φορές με διαφορετικούς αρχικούς εκπροσώπους είτε με το να κάνουμε κάποιου είδους περιορισμένη τοπική αναζήτηση για την συγκλινομένη λύση.

Εκτός του ότι είναι ευαίσθητος στην αρχικοποίηση, ο k -means υποφέρει από αρκετά άλλα προβλήματα. Αρχικά, παρατηρούμε ότι ο αλγόριθμος είναι μια περιοριστική περίπτωση εκπαίδευσης δεδομένων με μία μίξη από k γκαουσιανές με πανομοιότυπους, πίνακες ιστροπικής συμμεταβλητότητας όπου οι αναθέσεις των δεδομένων τείνουν να δεσμεύσουν το κάθε σημείο από τα δεδομένα στο πιο πιθανό στοιχείο. Έτσι ο αλγόριθμος δεν θα είναι αποτελεσματικός όταν τα δεδομένα δεν είναι προσδιορισμένα ως ικανοποιητικά σφαιρικές συστάδες, σε περίπτωση που για παραδειγμα υπάρχουν μη-κυρτές συστάδες. Αυτό το πρόβλημα μπορεί να λυθεί εάν επανακλιμακώσουμε τα δεδομένα (πχ αλλάζοντας το σύστημα συντεταγμένων) ή χρησιμοποιήσουμε άλλο μέτρο απόστασης πιο κατάλληλο στα συγκεκριμένα δεδομένα.

Ο k -means μπορεί να συνδυαστεί με έναν άλλο αλγόριθμο για να περιγράψει μη κυρτές συστάδες. Με βάση αυτήν την ιδέα, αρχικά ομαδοποιούνται τα δεδομένα σε μεγάλο αριθμό συστάδων με τη χρήση του k -means. Αυτές οι συστάδες στη συνέχεια συσσωματώνονται σε μεγαλύτερες συστάδες χρησιμοποιώντας την μέθοδο single link hierarchical clustering η οποία μπορεί να εντοπίζει περίπλοκες μορφές. Η προσέγγιση αυτή κάνει επίσης την επίλυση λιγότερο ευαίσθητη στην αρχικοποίηση και δεδομένου ότι η ιεραρχική μέθοδος παρέχει αποτελέσματα σε πολλαπλές αναλύσεις, δεν χρειάζεται να προπροδιορισθεί το k .

Τέλος, ο k means είναι επίσης ευαίσθητος στην παρουσία ακραίων τιμών, δεδομένου ότι ο μέσος δεν είναι ένα ισχυρό στατιστικό μέτρο. Ένα βήμα προεπεξεργασίας για την αφαίρεση των ακραίων τιμών μπορεί να είναι χρήσιμο σε τέτοιες περιπτώσεις. Μετεπεξεργασία των αποτελεσμάτων, για παράδειγμα, για την εξάλειψη μικρών συστάδων, ή για συγχώνευση κοντινών συστάδων σε

μεγαλύτερες συστάδες , είναι επίσης επιθυμητή[9].



2.7. Παραδειγμα εφαρμογής του αλγορίθμου $k - means$. Παρατηρούμε ότι σε 6 επαναλήψεις έχει συγκλίνει.

2.4. Χάρτες αυτοοργάνωσης (self organized map - SOM)

Ο χάρτης αυτοοργάνωσης (*SOM*), κοινώς γνωστός και ως δίκτυο *Kohonen* αποτελεί μια υπολογιστική μέθοδο για την οπτικοποίηση και ανάλυση δεδομένων μεγάλων διαστάσεων.

Ο χάρτης αυτός ορίζει μια διατεταγμένη χαρτογράφηση. Αποτελεί ένα είδος προβολής από ένα σύνολο δεδομένων στοιχείων σε ένα συνήθως διδιάστατο πλέγμα. Ένα μοντέλο m_i ανατίθεται σε κάθε κόμβο του πλέγματος και αυτά τα μοντέλα υπολογίζονται από τον αλγόριθμο του *SOM*. Ένα στοιχείο θα αντιστοιχιστεί στον κόμβο του οποίου το μοντέλο που είναι πιο παρόμοιο με το στοιχείο, π.χ. , έχει τη μικρότερη απόσταση από το στοιχείο δεδομένων σε κάποια μετρική.

Το κάθε μοντέλο είναι συνήθως ένας ορισμένος σταθμισμένος τοπικός μέσος όρος των στοιχείων στο χώρο των δεδομένων. Επιπροσθέτως, όταν τα μοντέλα υπολογίζονται από τον αλγόριθμο του *SOM* που θα δούμε παρακάτω , έχουν την τάση να είναι πιο πανομοιότυπα για κοντινούς κόμβους απότι για

κόμβους που βρίσκονται μακριά στο πλέγμα. Με τον τρόπο αυτό το σύνολο των μοντέλων(κόμβων) μπορεί να θεωρηθεί ότι αποτελεί ένα γράφημα ομοιότητας , και δομημένο «σκελετό» της κατανομής των δεδομένων στοιχείων.

Το *SOM* αναπτύχθηκε αρχικά για την απεικόνιση των κατανομών μετρικών διανυσμάτων , όπως διατεταγμένα σύνολα τιμών μετρήσεων ή στατιστικών ιδιοτήτων , αλλά μπορεί να αποδειχθεί ότι η χαρτογράφηση τύπου *SOM* μπορεί να οριστεί για οποιαδήποτε στοιχεία, των οποίων μπορούν να ορισθούν οι μεταξύ τους αποστάσεις. Παραδείγματα μη διανυσματικών δεδομένων στα οποία είναι εφικτό να εφαρμοσθεί αυτή η μέθοδος είναι οι συμβολοσειρές χαρακτήρων καθώς και οι αλληλουχίες τμημάτων σε οργανικά μόρια.

Ο αλγόριθμος *SOM* αναπτύχθηκε από τα πρώτα μοντέλα περιγραφής νευρωνικών δικτύων και ειδικά από μοντέλα συνειρμικής μνήμης και προσαρμοστικής μάθησης. Ένα κίνητρο αποτελούσε η ερμηνεία της χωρικής οργάνωσης των λειτουργιών του εγκεφάλου και ειδικά του εγκεφαλικού φλοιού. Παρόλα αυτά το *SOM* δεν αποτελούσε το πρώτο βήμα προς αυτή τη κατεύθυνση. Υπήρχαν ήδη άλλα μοντέλα (όπως πχ οι ανιχνευτές του von der Malsburg), των οποίων όμως οι δυνατότητες αυτοοργάνωσης ήταν περιορισμένες.

Η κρίσιμη εφεύρεση του *Kohonen* ήταν να εισαγάγει ένα μοντέλο συστήματος που αποτελείτο από τουλάχιστον δύο αλληλεπιδρούντα υποσυστήματα διαφορετικής φύσης. Το ένα από αυτά τα υποσυστήματα είναι ένα ανταγωνιστικό νευρωνικό δίκτυο που υλοποιεί τη λειτουργία ό νικητής τα παίρνει όλα , αλλά υπάρχει επίσης ένα άλλο υποσύστημα που ελέγχεται από το νευρωνικό δίκτυο και το οποίο τροποποιεί την τοπική συναπτική πλαστικότητα των νευρώνων στη μάθηση. Η μάθηση περιορίζεται χωρικά στη γειτονιά των πιο ενεργών νευρώνων. Το υποσύστημα ελέγχου πλαστικότητας θα μπορούσε να βασισθεί σε μη συγκεκριμένες νευρικές αλληλεπιδράσεις ,αλλά το πιθανότερο είναι ότι αποτελεί μια χημική επίδραση ελέγχου. Μόνο μέσω του διαχωρισμού της μεταφοράς του νευρικού σήματος και του ελέγχου πλαστικότητας έχει γίνει δυνατή η υλοποίηση ενός αποτελεσματικού αυτο οργανουμένου συστήματος.

Παρ'όλα αυτά , η βάση του *SOM* μπορεί επίσης να εκφραστεί μαθηματικά σε μια καθαρή μορφή , χωρίς αναφορά σε κάποιο υποκείμενο συστατικό (είτε νευρολογικής είτε χημικής φύσης).

Αρχικά έστω το δεδομένο στοιχείο του n -διάστατου Ευκλείδειου χώρου:

$$x(t) = [\xi_1(t), \xi_2(t), \dots, \xi_n(t)]$$

όπου t είναι ο δείκτης του στοιχείου σε μια δεδομένη ακολουθία. Έστω το i -οστό μοντέλο είναι

$$m_i(t) = [\mu_{i1}(t), \mu_{i2}(t), \dots, \mu_{in}(t)]$$

όπου το t υποδηλώνει εδώ τη σειρά στην οποία τα μοντέλα παράγονται. Αυτή η σειρά ορίζεται σαν μια διεργασία στην οποία η νέα τιμή $m_i(t+1)$ υπολογίζεται επαναληπτικά από την παλιά τιμή $m_i(t)$ και το τωρινό στοιχείο $x(t)$ ως:

$$m_i(t+1) = m_i(t) + a(t)h_{ci}(t)[x(t) - m_i(t)] \quad (26)$$

Εδώ το $a(t)$ αποτελεί ένα βαθμωτό παράγοντα που ορίζει το μέγεθος της διόρθωσης: οι τιμές του μικράνουν καθώς μεγαλώνει ο δείκτης t . Ο δείκτης i αναφέρεται στο μοντέλο υπό επεξεργασία και c είναι ο δείκτης του μοντέλου που έχει τη μικρότερη απόσταση από το $x(t)$ στον Ευκλείδιο χώρο του σήματος. Ο συντελεστής $h_{ci}(t)$ αποτελεί ένα είδος πυρήνα εξομάλυνσης ο οποίος ονομάζεται και συνάρτηση γειτονίας. Είναι ίσος με 1 όταν $i = c$ και η αξία του μειώνεται καθώς αυξάνεται η απόσταση από του δεδομένου μοντέλου i από το μοντέλο c πάνω στο πλέγμα. Επιπλέον, είναι συνηθες το χωρικό πλάτος του πυρήνα να μειώνεται με την αύξηση του βήματος t . Οι λειτουργίες αυτές του δείκτη βήματος t καθορίζουν σε μεγάλο βαθμό τη σύγκλιση του αλγορίθμου και επομένως πρέπει να διαλέγονται με μεγάλη προσοχή. Επίσης ένα άλλο ζήτημα αποτελεί η αρχικοποίηση του πλέγματος $m_i(1)$.

Αξίζει εδώ να σημειωθεί ότι αν και ο παραπάνω επαναληπτικός αλγόριθμος έχει χρησιμοποιηθεί με επιτυχία σε πολλές εφαρμογές, έχει αποδειχθεί ότι υπάρχει σύστημα - το οποίο ονομάζεται *batch map* - που παράγει ουσιαστικά παρόμοια αποτελέσματα, αλλά μια τάξη μεγέθους πιο γρήγορα. Η βασική ιδέα είναι ότι για κάθε κόμβο j στο δίκτυο, ο μέσος όρος \bar{x}_j όλων αυτών των στοιχείων εισόδου $x(t)$ που έχουν σχηματιστεί με το m_j ως το πλησιέστερο μοντέλο. Μετά από αυτό, τα νέα μοντέλα υπολογίζονται ως

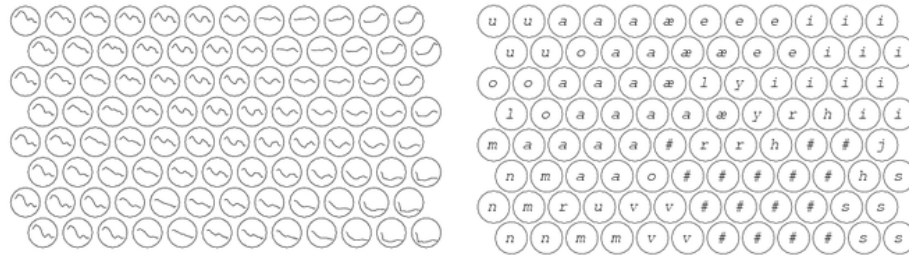
$$m_i = \frac{\sum_j n_j h_{ji} \bar{x}_j}{\sum_j n_j h_{ji}} \quad (27)$$

όπου το n_j είναι ο αριθμός των στοιχείων εισόδου που χαρτογραφούνται στον κόμβο j και ο δείκτης j κινείται σε όλη τη γειτονία του κόμβου i . Για το ανανεόμενο m_i ο αλγόριθμος επαναλαμβάνεται μερικές φορές πάντα χρησιμοποιώντας το ίδιο σύνολο δεδομένων εισόδου για να υπολογίσει το ανανεόμενο \bar{x}_j .

Γενικά η μαθηματική θεωρία ανάλυσης του *SOM* είναι ιδιαίτερα περίπλοκη και μόνο η περίπτωση της μίας διάστασης έχει αναλυθεί πλήρως.

Πριν προχωρήσουμε πρέπει να διευκρινισθεί ότι ο αλγόριθμος του *SOM* ουσιαστικά αποτελεί περίπτωση μη επιβλεπόμενης μάθησης δεδομένου ότι δεν εκπαιδεύεται με κάποιον τρόπο από εκπαιδευτή, αλλά μαθαίνει και προσαρμόζεται στο περιβάλλον του από το ίδιο το περιβάλλον βάσει ενός κριτηρίου (πχ εδώ μικρότερη απόσταση, μεγαλύτερη επιβράβευση).

Επίσης, ο αλγόριθμος αυτός, ουσιαστικά επικεντρώνεται στην χαρτογράφηση και όχι στην ταξινόμηση δεδομένων. Παρόλα αυτά εμπεριέχεται στη παρούσα διπλωματική η θεωρητική ανάλυση του λόγω του ότι αποτελεί μια ενδιαφέρουσα περίπτωση νευρωνικού δικτύου η οποία αξίζει να μελετηθεί, καθώς και επειδή θα χρησιμοποιηθεί σαν αλγόριθμος ομαδοποίησης παρακάτω.



2.8. Παράδειγμα χαρτογράφησης με SOM όπου σήματα ακουστικών συχνοτήτων φινλανδικών φωνηέντων μετατρέπονται από ένα SOM στα αντίστοιχα γράμματα

Εδώ έχει ενδιαφέρον να εξετάσουμε ορισμένες πιο σύγχρονες προσεγγίσεις χαρτών αυτοοργάνωσης, οι οποίες έχουν αναπτυχθεί τα τελευταία χρόνια, προκειμένου να κατανοήσουμε καλύτερα την ανταγωνιστική μάθηση.

2.4.1 Παραλλαγές αυξανόμενων SOM (Fritzke)

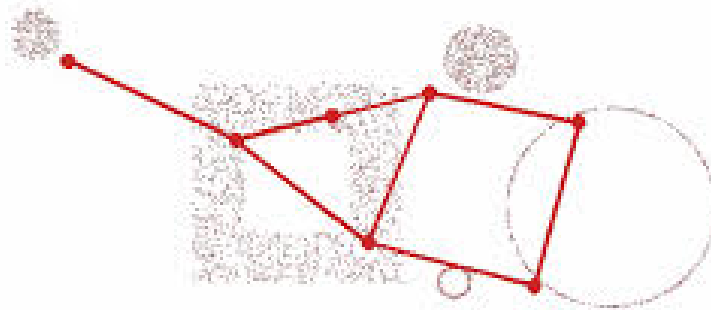
Οι χάρτες αυτοοργάνωσης όπως παρουσιάστηκαν από τον Kohonen αποτελούν πλέγματα στατικού μεγέθους. Το προφανές μειονέκτημα ενός δεδομένου αριθμού νευρώνων που δεν αλλάζει είναι ότι είτε θα είναι πολύ μικρός για να χαρτογραφήσει ικανοποιητικά την είσοδο, είτε θα είναι πολύ μεγάλος, με αποτέλεσμα να υπάρχουν νευρώνες που υποχρησιμοποιούνται. Το SOM δεδομένου ότι εκπαιδεύεται μή επιβλεπόμενα, δεν αναμένεται να ξέρει τα χαρακτηριστικά του χώρου εισόδου εκ των προτέρων. Επομένως, μια τοπολογία η οποία επιτρέπει την προσθήκη-αφαίρεση νευρώνων από το πλέγμα είναι ένα λογικό βήμα στην ανάπτυξη της αυτοοργάνωσης. Ο Fritzke εισήγαγε τρεις αρχιτεκτονικές στη δεκαετία του 1990, την growing grid (GG), τη growing cell (GC) και τη growing neural gas (GNG). Και οι τρεις ξεκινάνε με ελάχιστο αριθμό νευρώνων και προσθέτουν νευρώνες ανάλογα με το που τους χρειάζονται. Το κατά πόσο χρειάζονται εξαρτάται από μία ηλικιακή παράμετρο ενώ μια παράμετρος σφάλματος καθορίζει ποιος νευρώνας θα αποκτήσει νέο γείτονα (GC, GNG) ή νέο σύνολο γειτόνων (GG).

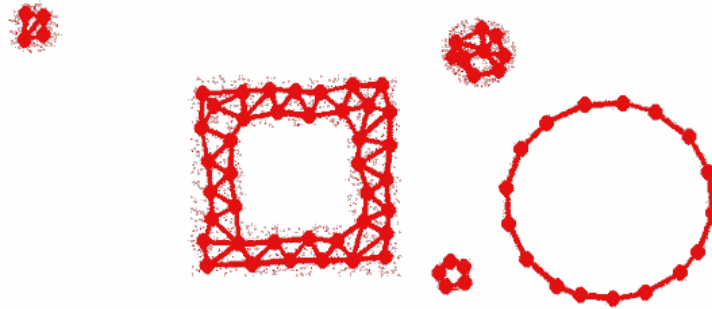
Στη περίπτωση του GC από τη στιγμή που ο νευρώνας με τη μεγαλύτερη τιμή σφάλματος επιλεγεί, η μακρύτερη ακμή του γίνεται δύο ακμές και ένας

νευρώνας προστίθεται για να διευκολύνει τη σύνδεση τους.

Το *GG* χρησιμοποιεί επίσης μια παράμετρο ηλικίας για τους νευρώνες. Κάθε φορά που κάποιος νευρώνας νικά η ηλικία του αυξάνεται. Στα μεσοδιαστήματα, ο νευρώνας με τις περισσότερες νίκες αναμένεται να λάβει καινούργιους γείτονες και επίσης εντοπίζεται ο πιο μακρινός άμεσος τοπολογικός γείτονας του επιλεγμένου νευρώνα. Το *GG* χρησιμοποιεί ένα ορθογώνιο πλέγμα το οποίο διατηρείται κατά την μεγένθυση. Αν αυτοί οι δύο νευρώνες είναι στην ίδια σειρά, θα προστεθεί μεταξύ τους μια νέα στήλη, με αποτέλεσμα να επηρεασθούν νευρώνες από άλλες γραμμές. Αν είναι στην ίδια στήλη, προστίθεται μια νέα σειρά μεταξύ τους, με αποτέλεσμα να επηρεασθούν νευρώνες από άλλες στήλες.

Στη περίπτωση του *GNG* παρατηρούμε έναν αντίστοιχο τρόπο σκέψης με την περίπτωση του *GC*. Παρατάτα, σε αυτή τη περίπτωση η ηλικιακή παράμετρος ανατίθεται στις συνδέσεις μεταξύ των νευρώνων. Επομένως, το *GNG* προσθέτει και αφαιρεί συνδέσεις με βάση την επιλογή των νευρώνων. Το *GNG* επίσης έχει τη δυνατότητα να επιτυγχάνει διαμορφώσεις μεταξύ πολλαπλών δικτύων (δεδομένου ότι το πλέγμα μπορεί να διαχωρισθεί)





2.9. Παράδειγμα χαρτογράφησης με *GNG* (αρχικό τελικό). Παρατηρούμε ότι το πλέγμα διαχωρίζεται σε πολλά πλέγματα προκειμένου να πάρει τη μορφή της εισόδου και ότι ο αριθμός των νευρώνων δεν είναι σταθερός

Ένα από τα μεγαλύτερα μειονεκτήματα του *GNG* όπως επίσης και για τις άλλες αυξητικές μεθόδους είναι ότι ο αριθμός των νευρώνων συνεχώς αυξάνεται. Αυτό εν μέρει αντισταθμίζεται από μία επέκταση του *GNG*, το *GNG-U* (growing neural gas with utility) το οποίο εμπεριέχει την αφαίρεση κόμβων το οποίο βασίζεται σε κριτήρια χαμηλής πυκνότητας προτύπων εισόδου 'κάτω' από τον δεδομένο (προς αφαίρεση) νευρώνα. Το πρόβλημα αυτών των μεθόδων είναι ότι δεν παρουσιάζουν αυξητική μάθηση.

2.4.2 ESOINN

Το ESOINN (enhanced Self organizing Incremental Neural Network) (Furao, Ogura and Hasegawa 2007) αντιπροσωπεύει αυξητικές αρχιτεκτονικές, οι οποίες έχουν εν μέρει επηρεασθεί από το *GNG* και το *GNG-U*. Με βάση τους συγγραφείς του ESOINN, το συγκεκριμένο σύστημα αντιμετωπίζει το δίλημμα σταθερότητας-ελαστικότητας με το να παρέχει τη δυνατότητα διατήρησης της γνώσης από πρότυπα τα οποία έχει ήδη μάθει (σταθεροτητα) ενώ ταυτόχρονα μπορεί να προσαρμοσθεί σε και να μάθει νέα πρότυπα στα οποία δεν έχει ακόμα εκτεθεί (ελαστικότητα). Το ESOINN προσδιορίζει συστάδες κατά την εκτέλεση με το να ενώνει μαζί υποομάδες για να δημιουργήσει μεγαλύτερες συστάδες. Αυτό έχει σαν αποτέλεσμα όταν υπάρχει επικάλυψη σε πολλαπλές συστάδες, αυτή μπορεί να εντοπισθεί και να διαχωρισθεί αποτελεσματικά.

Συνήθως με τις αυξανόμενες αρχιτεκτονικές , το δίκτυο μεγαλώνει με την προσθήκη των νευρώνων σε επαρκώς πυκνές περιοχές του χώρου εισόδου . Στο *ESOINN* , προστίθενται νευρώνες όταν το το παρών πρότυπο εισόδου είναι επαρκώς μακριά από τον πλησιέστερο σε αυτό νευρώνα. Ένας νέος νευρώνας προστίθεται στο δίκτυο που έχει το ίδιο (όχι παρόμοιο) διάνυσμα αναφοράς με την είσοδο.

Το *ESOINN* αποφασίζει για την πρόσθεση ενός νεου νευρώνα βασιζόμενο σε ένα κατώφλι ομοιότητας. Ουσιαστικά αποτελεί ένα δυναμικό μέτρο απόστασης το οποίο υπολογίζεται με βάση την απόσταση των γειτόνων του νευρώνα ή εάν δεν υπάρχουν γείτονες, από των υπόλοιπων νευρώνων του δικτύου. Όταν ένα δεδομένο εισόδου παρουσιάζεται στο δίκτυο ο πρώτος και ο δεύτερος νικητής (ή αλλιώς η μονάδα βέλτιστου ταιριάσματος και δεύτερου καλύτερου ταιριάσματος) προσδιορίζονται. Κατόπιν το δίκτυο αποφασίζει αν μια σύνδεση μεταξύ του νικητή και του δεύτερου πρέπει να δημιουργηθεί ή αν υπάρχει ήδη.

Στο σύστημα αυτό η γνώση της πυκνότητας νευρώνων σε μια δεδομένη περιοχή του χώρου εισόδου είναι κρίσιμης σημασίας για την εκτέλεση καθηκόντων , όπως η δημιουργία συνδέσεων μεταξύ των νευρώνων και η ανίχνευση επικαλύψεων σε συστάδες. Με το να είναι σε θέση να μετρήσει την πυκνότητα , το δίκτυο μπορεί να καθορίσει αν ένα συγκεκριμένο τμήμα του χώρου εισόδου είναι μέρος ενός ενιαίου σύμπλεγματος ή ενός επικαλυπτόμενου τμήματος. Μετά την ανίχνευση των επικαλυπτόμενων περιοχών, οι συνδέσεις μεταξύ των νευρώνων διαφορετικών υποκατηγοριών αφαιρούνται.

Αυτό διαχωρίζει τις υποκλάσεις που ανήκουν σε διαφορετικές σύνθετες κλάσεις. Αυτή η διαδικασία εκτελείται σε τακτά διαστήματα όπου ο αριθμός των εισόδων που παρουσιάζεται στο δίκτυο είναι ομοιόμορφα διαχωρίσιμος από μια δεδομένη τιμή ενός ακεραίου.

2.5. Ο αλγόριθμος EM (Expectation - Maximization)

Ο αλγόριθμος EM είναι μια πολύ γενική μέθοδος για την επίλυση προβλημάτων υπολογισμού μέγιστης πιθανότητας (Maximum Likelihood (ML)). Αποτελεί μια αποτελεσματική επαναληπτική διαδικασία για τον υπολογισμό τέτοιου είδους προβλημάτων παρουσία κρυμμένων δεδομένων. Στην εκτίμηση *ML* θέλουμε να προσδιορίσουμε το μοντέλο παραμέτρων στο οποίο τα παρατηρούμενα δεδομένα είναι πιο πιθανό να ανήκουν. Κάθε επανάληψη του αλγορίθμου EM αποτελείται από δύο διαδικασίες : Το E - βήμα και το M- βήμα.

Στο βήμα E (-expectation) εκτιμούνται τα κρυμμένα δεδομένα με βάση

τα παρατηρούμενα δεδομένα και την τρέχουσα εκτίμηση των παραμέτρων του μοντέλου. Αυτό επιτυγχάνεται χρησιμοποιώντας την υπο συνθηκη παρούσα προσδοχία.

Κατα το βήμα M (- maximization) η συνάρτηση πιθανότητας μεγιστοποιείται υπο την υπόθεση ότι τα δεδομένα που λείπουν (κρυφά) είναι γνωστά. Η εκτίμηση των ελλειπόντων δεδομένων από το E - βήμα χρησιμοποιείται στη θέση των πραγματικών δεδομένων που λείπουν. Η σύγκλιση είναι εξασφαλισμένη δεδομένου ότι ο αλγόριθμος είναι εγγυημένο ότι θα αυξήσει τη πιθανότητα σε κάθε επανάληψη.

Για κάθε κρυφό δεδομένο δηλαδή ο αλγόριθμος προσδιορίζει με βάση κάποια πιθανότητα την (ή τις) κατανομή στην οποία αυτό είναι πιθανότερο να ανήκει (βήμα E), και κατόπιν με βάση αυτά τα δεδομένα, επαναπροσδιορίζονται οι αντιστοιχες νέες θέσεις της κάθε κατανομής (βήμα M). Παρακάτω εξετάζουμε τον αλγόριθμο με μεγαλύτερη λεπτομέρεια.

Εστω \mathbf{X} ένα τυχαίο διάνυσμα. Θέλουμε να βρούμε θ τέτοιο ώστε να μεγιστοποιούμε τη πιθανότητα $P(\mathbf{X}|\theta)$. Αυτό αποτελεί την εκτίμηση μέγιστης πιθανότητας για το θ . Προκείμενου να υπολογίσουμε το θ , είναι συνηθισμένο να εισάγουμε την λογαριθμική συνάρτηση πιθανότητας η οποία ορίζεται ως:

$$L(\theta) = \ln P(\mathbf{X}|\theta) \quad (28)$$

Η συνάρτηση πιθανότητας θεωρείται ότι είναι μια συνάρτηση της παραμέτρου θ με δεδομένο το διάνυσμα \mathbf{X} . Εφόσον η $\ln(x)$ είναι μια γνησιως αύξουσα συνάρτηση, η τιμή του θ η οποία μεγιστοποιεί το $P(\mathbf{X}|\theta)$ θα μεγιστοποιεί επίσης και το $L(\theta)$.

Ο αλγόριθμος EM αποτελεί μια επαναληπτική μέθοδο για να μεγιστοποιήσουμε το $L(\theta)$. Υποθέτουμε ότι μετά την n -οστή επανάληψη η παρούσα εκτίμηση για το θ δίνεται από το θ_n . Εφόσον στόχος είναι η μεγιστοποίηση του $L(\theta)$, θέλουμε να υπολογίσουμε μια νέα εκτίμηση για το θ έτσι ώστε

$$L(\theta) > L(\theta_n) \quad (29)$$

Ισοδύναμα θέλουμε να μεγιστοποιήσουμε τη διαφορά

$$L(\theta) - L(\theta_n) = P(\mathbf{X}|\theta) - P(\mathbf{X}|\theta_n) \quad (30)$$

Μέχρι στιγμής δέν έχουμε συμπεριλάβει καθόλου τυχών μη παρατηρούμενα ή ελλειπόντα δεδομένα. Σε προβλήματα όπου υπάρχουν τέτοια δεδομένα, ο EM παρέχει ένα 'φυσικό' περιβάλλον για να συμπεριληφθούν. Εναλλακτικά, οι κρυφές μεταβλητές μπορούν να εισαχθούν καθαρά σαν ένα τέχνασμα για να μπορέσουμε να προσδιορίσουμε την εκτίμηση μέγιστης πιθανότητας του θ .

Σε κάθε περίπτωση, ορίζουμε το κρυμμένο τυχαίο διάνυσμα \mathbf{Z} και έστω μια εκδοχή του \mathbf{z} . Η συνολική πιθανότητα $P(\mathbf{X}|\theta)$ μπορεί να γραφει ως προς τις κρυφές μεταβλητές σαν :

$$P(\mathbf{X}|\theta) = \sum_z P(\mathbf{X}|\mathbf{z}, \theta)P(\mathbf{z}|\theta) \quad (31)$$

Εδώ μπορούμε να ξαναγράψουμε την (30) ως εξής:

$$L(\theta) - L(\theta_n) = \ln\left(\sum_z P(\mathbf{X}|\mathbf{z}, \theta)P(\mathbf{z}|\theta)\right) - \ln P(\mathbf{X}|\theta_n) \quad (32)$$

Παρατηρούμε ότι η έκφραση περιλαμβάνει τον λογάριθμο ενός αθροίσματος. Επίσης μπορούμε να δείξουμε την ακόλουθη έκφραση:

$$\ln \sum_{i=1}^n \lambda_i x_i \geq \sum_{i=1}^n \lambda_i \ln(x_i) \quad (33)$$

για στάθερες $\lambda_i \geq 0$ με $\sum_{i=1}^n \lambda_i = 1$. Αυτό μπορεί να εφαρμοσθεί στη παραπάνω εξίσωση εφόσον προσδιορίσουμε ποιες είναι οι λ_i . Θα θέσουμε τις σταθερές στη μορφή $P(\mathbf{z}|\mathbf{X}, \theta_n)$. Εφόσον η $P(\mathbf{z}|\mathbf{X}, \theta_n)$ αποτελεί πιθανοτικό μέτρο, θα είναι θετική ή ίση του μηδενός και επίσης θα είναι $\sum_z P(\mathbf{z}|\mathbf{X}, \theta_n) = 1$ όπως απαιτείται.

Με βάση τα παραπάνω θα έχουμε:

$$\begin{aligned} L(\theta) - L(\theta_n) &= \ln\left(\sum_z P(\mathbf{X}|\mathbf{z}, \theta)P(\mathbf{z}|\theta)\right) - \ln P(\mathbf{X}|\theta_n) \\ &= \ln\left(\sum_z P(\mathbf{X}|\mathbf{z}, \theta)P(\mathbf{z}|\theta) \frac{P(\mathbf{z}|\mathbf{X}, \theta_n)}{P(\mathbf{z}|\mathbf{X}, \theta_n)}\right) - \ln P(\mathbf{X}|\theta_n) \\ &\geq \sum_z P(\mathbf{z}|\mathbf{X}, \theta_n) \ln\left(\frac{P(\mathbf{X}|\mathbf{z}, \theta)P(\mathbf{z}|\theta)}{P(\mathbf{z}|\mathbf{X}, \theta_n)}\right) - \ln P(\mathbf{X}|\theta_n) \\ &= \sum_z P(\mathbf{z}|\mathbf{X}, \theta_n) \ln\left(\frac{P(\mathbf{X}|\mathbf{z}, \theta)P(\mathbf{z}|\theta)}{P(\mathbf{z}|\mathbf{X}, \theta_n)P(\mathbf{X}|\theta_n)}\right) \end{aligned} \quad (34)$$

και θα ονομάσουμε το τελικό αποτέλεσμα $\Delta(\theta|\theta_n)$. Αξίζει να σημειωθεί ότι για την μετάβαση από την προτελευταία στη τελευταία σχέση της (34) χρησιμοποιήθηκε το γεγονός ότι $\sum_z P(\mathbf{z}|\mathbf{X}, \theta_n) = 1$ έτσι ώστε $\ln P(\mathbf{X}|\theta_n) = \sum_z P(\mathbf{z}|\mathbf{X}, \theta_n) \ln P(\mathbf{X}|\theta_n)$ το οποίο επιτρέπει στον όρο $\ln P(\mathbf{X}|\theta_n)$ να μπει στο άθροισμα. Συνεχίζουμε γράφοντας:

$$L(\theta) \geq L(\theta_n) + \Delta(\theta|\theta_n) \quad (35)$$

και για χάρην ευκολίας ορίζουμε:

$$l(\theta|\theta_n) = L(\theta_n) + \Delta(\theta|\theta_n) \quad (36)$$

Ετσι η 11 ορίζεται ως

$$L(\theta) \geq l(\theta|\theta_n) \quad (37)$$

Ετσι εχουμε μια συνάρτηση $l(\theta|\theta_n)$ η οποία είναι ανω φραγμενη απο την συνάρτηση πιθανότητας $L(\theta)$. Επιπλέον παρατηρούμε οτι μετα απο πράξεις καταλήγουμε στη σχέση:

$$l(\theta_n|\theta_n) = L(\theta_n) \quad (38)$$

και επομένως για $\theta = \theta_n$ οι δύο συναρτήσεις ταυτίζονται.

Στόχος μας είναι να διαλέξουμε τιμές για το θ τέτοιες ώστε να μεγιστοποιείται το $L(\theta)$. Εχουμε δείξει οτι η συνάρτηση $l(\theta|\theta_n)$ είναι ανω φραγμένη απο τη $L(\theta)$ και επίσης οτι οι τιμές των δυο αυτών συναρτήσεων είναι ίσες όταν $\theta = \theta_n$. Επομένως οποιοδήποτε θ αυξάνει την l , θα αυξάνει και την L . Προκειμένου να επιτύχουμε τη μεγαλύτερη δυνατή αύξηση της τιμης της L ο αλγόριθμος EM διαλέγει θ τέτοιο ώστε η l να μεγιστοποιείται. Αυτό αποτελεί τη νέα τιμη θ_{n+1}

Πιο τυπικά έχουμε:

$$\theta_{n+1} = \operatorname{argmax}_{\theta} (l(\theta|\theta_n)) = \operatorname{argmax}_{\theta} (L(\theta_n) + \sum_z P(\mathbf{z}|\mathbf{X}, \theta_n) \ln \left(\frac{P(\mathbf{X}|\mathbf{z}, \theta) P(\mathbf{z}|\theta)}{P(\mathbf{z}|\mathbf{X}, \theta_n) P(\mathbf{X}|\theta_n)} \right)) \quad (39)$$

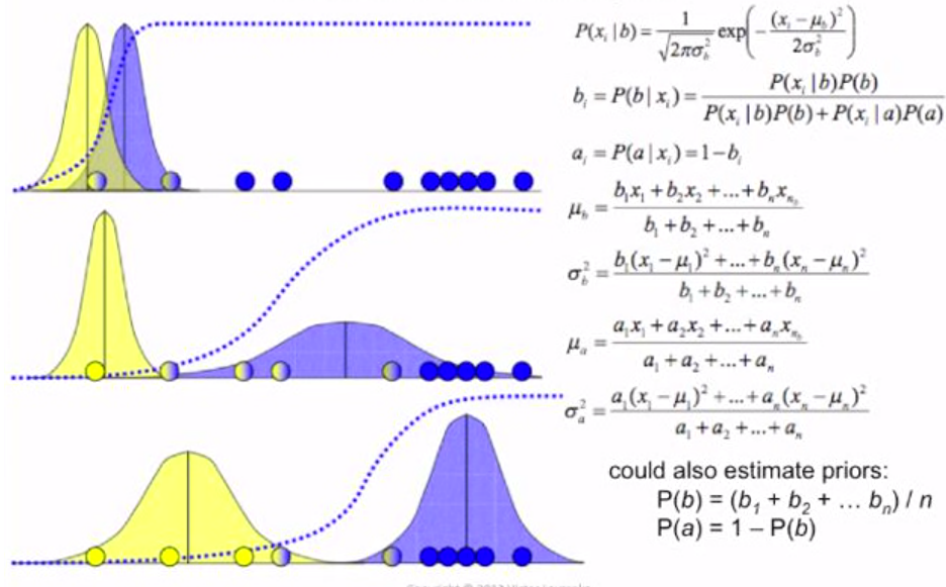
διαγραφουμε τους σταθερους ορους ως προς θ και:

$$\begin{aligned} &= \operatorname{argmax}_{\theta} \left(\sum_z P(\mathbf{z}|\mathbf{X}, \theta_n) \ln (P(\mathbf{X}|\mathbf{z}, \theta) P(\mathbf{z}|\theta)) \right) \\ &= \operatorname{argmax}_{\theta} \left(\sum_z P(\mathbf{z}|\mathbf{X}, \theta_n) \ln \left(\frac{P(\mathbf{X}, \mathbf{z}, \theta)}{P(\mathbf{z}, \theta)} \frac{P(\mathbf{z}, \theta)}{P(\theta)} \right) \right) \\ &= \operatorname{argmax}_{\theta} \left(\sum_z P(\mathbf{z}|\mathbf{X}, \theta_n) \ln P(\mathbf{X}, \mathbf{z}|\theta) \right) \\ &= \operatorname{argmax}_{\theta} (E_{\mathbf{z}|\mathbf{X}, \theta_n} [\ln P(\mathbf{X}, \mathbf{z}|\theta)]) \end{aligned} \quad (40)$$

Επομένως ο αλγόριθμος εμπεριέχει τα ακόλουθα βήματα:

1. βημα-E: Υπολογισμός της υπο συνθήκη προσδοκίας $E_{\mathbf{z}|\mathbf{X}, \theta_n} [\ln P(\mathbf{X}, \mathbf{z}|\theta_n)]$.
2. βημα-M: Μεγιστοποίηση αυτης της έκφρασης ως προς το θ

EM: 1-d example



2.10. Παράδειγμα εφαρμογής του αλγορίθμου στη μια διάσταση. Παρατηρούμε την προσαρμογή των παραμέτρων των κατανομών ανάλογα με τα δεδομένα και μετά την επαναπροσαρμογή των δεδομένων με βάση την πιθανότητα για την κάθε κατανομή

Σε αυτό το σημείο είναι λογικό να αναρωτηθούμε τι κερδίσαμε δεδομένου ότι απλά αλλάξαμε το ζήτημα της μεγιστοποίησης του $L(\theta)$ με αυτό της μεγιστοποίησης του $l(\theta|\theta_n)$. Η απάντηση κρύβεται στο γεγονός ότι η l λαμβάνει υπόψη της και τα ελλειπούμενα δεδομένα \mathbf{Z} . Στη περίπτωση όπου επιθυμούμε να υπολογίσουμε αυτές τις μεταβλητές, ο EM αλγόριθμος παρέχει ένα πλαίσιο για να γίνει αυτό. Επίσης, ενδέχεται να παρουσιαστούν τέτοιες κρυφές μεταβλητές που η μεγιστοποίηση της $L(\theta|\theta_n)$ να απλοποιείται λόγω της γνώσης αυτών.[12][13]

Συνολικά ο αλγοριθμος EM αποτελεί μια ιδιαίτερα χρήσιμη μέθοδο δεδομένου ότι είναι βέβαιο ότι θα συγκλίνει, καθώς και επειδή είναι αρκετά γρήγορος. Απο την άλλη, έχει το μειονέκτημα ότι βρίσκει τοπικά βέλτιστες λύσεις, καθώς και ότι είναι λόγω αυτού ιδιαίτερα ευαίσθητος στην αρχικοποίηση των παραμέτρων του.[14]

2.6. Support Vector Machines(SVM)

Οι μηχανές διανυσμάτων υποστήριξης (*SVM*) αποτελούν δυαδικές μηχανές οι οποίες κατασκευάζουν ένα υπερεπίπεδο ως επιφάνεια απόφασης (επιφάνεια δηλαδή που χωρίζει δύο κλάσεις προτύπων, εδώ θετικές και αρνητικές) με τρόπο ώστε το περιθώριο διαχωρισμού (μεταξύ των δύο κλάσεων) να μεγιστοποιείται. Αυτή η ιδέα επεκτείνεται έτσι ώστε να καλύπτει και την περίπτωση μη γραμμικά διαχωρήσιμων προτύπων.[1]

Το πιο απλό μοντέλο μηχανής διανυσμάτων υποστήριξης, το οποίο ήταν επίσης το πρώτο, είναι ο επονομαζόμενος ταξινομητής μέγιστου περιθωρίου (*maximal margin classifier*). Ο συγκεκριμένος ταξινομητής δουλεύει μόνο για δεδομένα τα οποία είναι γραμμικά διαχωρίσιμα, δηλαδή δεδομένα τα οποία είναι επαρκώς διαχωρισμένα το ένα από το άλλο έτσι ώστε να διασφαλίζεται ότι η επιφάνεια απόφασης είναι υπερεπίπεδο, στο χώρο των χαρακτηριστικών και επομένως δεν μπορεί να χρησιμοποιηθεί σε πολλές περιπτώσεις. Παρ' όλα αυτά, λόγω της απλότητας του είναι ο πιο εύκολος αλγόριθμος για να κατανοηθεί και ταυτόχρονα παρουσιάζει τις πιο σημαντικές ιδιότητες μιας τέτοιου τύπου μηχανής.[5]

Ορίζουμε τα δεδομένα εκπαίδευσης ως $[\mathbf{x}_i, y_i], i = 1, \dots, l, y_i \in \{-1, 1\}, \mathbf{x}_i \in \mathbf{R}^d$. Υποθέτουμε ότι έχουμε κάποιο υπερεπίπεδο το οποίο διαχωρίζει τα δεδομένα των θετικών από αυτά των αρνητικών δεδομένων. Τα σημεία \mathbf{x} που βρίσκονται στο υπερεπίπεδο ικανοποιούν τη σχέση $\mathbf{w}\mathbf{x} + b = 0$ όπου το \mathbf{w} ορίζει το υπερεπίπεδο και $b/\|\mathbf{w}\|$ αποτελεί την κάθετη απόσταση από το υπερεπίπεδο μέχρι το σημείο αναφοράς, με $\|\mathbf{w}\|$ να αποτελεί την ευκλείδεια νόρμα του \mathbf{w} . Εστω $d_+(d_-)$ η μικρότερη απόσταση από το διαχωριστικό επίπεδο στο κοντινότερο θετικό (αρνητικό) παράδειγμα. Ορίζουμε το 'διάκενο' του διαχωριστικού επιπέδου να είναι $d_+ + d_-$. Για την γραμμικά διαχωρίσιμη περίπτωση ο αλγόριθμος *SVM* απλά ψάχνει για το διαχωριστικό επίπεδο με το μεγαλύτερο διάκενο. Αυτό μπορεί να γραφεί μαθηματικά ως εξής: Εστω ότι όλα τα δεδομένα εκπαίδευσης ικανοποιούν τους ακόλουθους περιορισμούς:

$$\mathbf{x}_i \mathbf{w} + b \geq +1 \quad \text{for } y_i = +1 \quad (41)$$

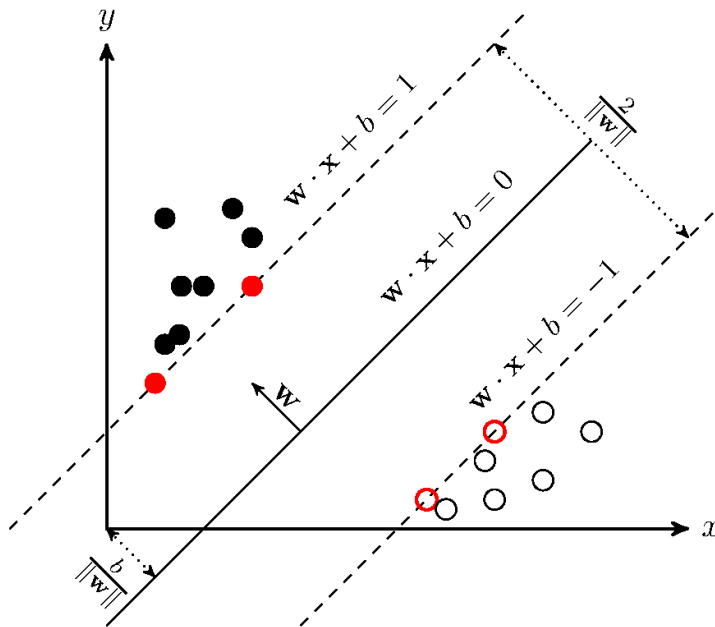
$$\mathbf{x}_i \mathbf{w} + b \leq -1 \quad \text{for } y_i = -1 \quad (42)$$

Αυτοί μπορούν να συνδυασθούν σε μια ανισότητα ως εξής:

$$y_i(\mathbf{x}_i \mathbf{w} + b) - 1 \geq 0 \quad \forall i \quad (43)$$

Τώρα, ως θεωρήσουμε τα σημεία για τα οποία ισχύει η ισότητα της ανισότητας (41) (το να ζητάμε να υπάρχει ένα τέτοιο σημείο ισοδυναμεί με να προσδιορίζουμε κατάλληλες τιμές για τα \mathbf{w} και b). Αυτά τα σημεία βρίσκονται

στο υπερεπίπεδο $H_1 : \mathbf{x}_i \mathbf{w} + b = +1$ με απόσταση απο το σημείο αναφοράς ίση με $|1 - b|/\|\mathbf{w}\|$. Αντίστοιχα , τα σημεία για τα οποία η (42) ισχύει ως ισότητα βρίσκονται στο υπερεπίπεδο $H_2 : \mathbf{x}_i \mathbf{w} + b = -1$ με απόσταση απο το σημείο αναφοράς ίση με $|-1 - b|/\|\mathbf{w}\|$. Επομένως $d_+ = d_- = 1/\|\mathbf{w}\|$ και το διάκενο είναι απλά ίσο με $2/\|\mathbf{w}\|$. Αξίζει να σημειωθεί οτι τα H_1 και H_2 είναι παράλληλα καθώς και οτι δεν υπάρχουν σημεία δεδομένων εκπαίδευσης μεταξύ τους. Επομένως μπορούμε να βρούμε το ζεύγος των υπερεπιπέδων τα οποία μεγιστοποιούν το διάκενο ,με το να ελαχιστοποιήσουμε τον όρο $\|\mathbf{w}\|^2$,δεδομένων των περιορισμών της σχέσης (43)



2.11.Βλέπουμε τη λύση μιας τυπικής δισδιάστατης περίπτωσης. Τα σημεία εκπαίδευσης για τα οποία ισχύουν οι συνθηκες ισότητας των περιορισμων ονομαζονται διανύσματα υποστήριξης. Παρατηρούμε οτι η αλλαγή των συντεταγμενων τους θα αλλάξει και τη λύση

Θα δούμε τώρα σε μια διαφορετική διατύπωση του προβλήματος (με χρήση *Lagrangians*) . Υπάρχουν δύο λόγοι για να γίνει αυτό. Αρχικά οι παραπάνω περιορισμοί θα αντικατασταθούν απο τους πολλαπλασιαστές *Lagrange* και ετσι θα είναι πολυ ευκολότερο να τους χειριστούμε. Επίσης, σε αυτή την αναδιατύπωση του προβλήματος , τα δεδομένα εκπαίδευσης θα εμφανιστούν μόνο με τη μορφή γινομένων μεταξύ διανυσμάτων. Αυτή αποτελεί μια κρίσιμη ιδιότητα η οποία μας επιτρέπει να γενικεύσουμε την διαδικασία για τη μη γραμ-

μική περίπτωση.

Επομένως ,εισάγουμε τους θετικούς πολλαπλασιαστές *Lagrange* $\alpha_i, i = 1, \dots, l$ ένα για κάθε περιορισμό της σχέσης περιορισμων (43). Ο κανόνας λέει οτι για περιορισμούς της μορφής $c_i \geq 0$,οι σχέσεις περιορισμών πολλαπλασιάζονται με θετικούς πολλαπλασιαστές και αφαιρούνται απο την αντικειμενική συνάρτηση,για να σχηματισθεί η *Lagrangian*. Για περιορισμούς τυπου ισότητας ,οι πολλαπλασιαστές δεν έχουν κάποιον περιορισμό. Απο τα παραπάνω προκύπτει:

$$L_P = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^l \alpha_i y_i (\mathbf{x}_i \mathbf{w} + b) + \sum_{i=1}^l \alpha_i \quad (44)$$

Τώρα επομένως πρέπει να ελαχιστοποιήσουμε την L_P ως προς τα \mathbf{w}, b και ταυτόχρονα να απαιτήσουμε οτι οι παράγωγοι της L_P ως προς όλα τα α_i να μηδενιστούν υπο τον περιορισμό $\alpha_i \geq 0$ (Εστω οτι αυτο το σετ περιορισμών ονομάζεται C_1). Αυτο αποτελεί ενα convex quadratic programming problem ,αφού η αντικειμενική συνάρτηση είναι απο μόνη της κυρτή και τα σημεια τα οποία ικανοποιούν τους περιορισμους επίσης σχηματίζουν ένα κυρτό συνολο (οποιοσδήποτε γραμμικός περιορισμός ορίζει ένα κυρτο σύνολο, και N γραμμικοί περιορισμοί ορίζουν μια τομή N κυρτών συνόλων ,η οποία αποτελεί επίσης κυρτο σύνολο). Αυτο σημαίνει οτι μπορούμε ισοδύναμα να λύσουμε το ακόλουθο δυαδικό πρόβλημα: Μεγιστοποίηση του L_P καθώς υπόκειται στους περιορισμούς για τους οποίους η κλίση του L_P ως προς το \mathbf{w} και το b μηδενίζεται και υποκειται επίσης στους περιορισμους όπου $\alpha_i \geq 0$ (εστω οτι αυτο το συνολο περιορισμών αποτελεί το C_2). Η συγκεκριμένη δυαδική διαμόρφωση του προβλήματος έχει την ιδιοτητα οτι το μέγιστο L_P δεδομένου του C_2 προκυπτει για τις ίδιες τιμές \mathbf{w}, b και α , με αυτές για τις οποίες έχουμε το ελάχιστο L_P υπο τις C_1 .

Απαιτώντας να μηδενιστεί η κλίση ως προς τα \mathbf{w} και b παίρνουμε:

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i \quad (45)$$

$$\sum_i \alpha_i y_i = 0 \quad (46)$$

Εφόσον αυτές αποτελούν τους περιορισμους ισότητας της δυαδικής φόρμας, μπορούμε να τις αντικαταστήσουμε στην εξίσωση (44) και να πάρουμε:

$$L_D = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \mathbf{x}_j \quad (47)$$

Παρατηρούμε ότι δώσαμε στη *Lagrangian* την ετικέτα D (πριν είχαμε P για *primal* τώρα D για *dual*) για να τονίσουμε ότι οι δυο διατυπώσεις είναι διαφορετικές : οι L_P, L_D προκύπτουν από την ίδια αντικειμενική συνάρτηση αλλά με διαφορετικούς περιορισμούς, και η λύση βρίσκεται είτε με την ελαχιστοποίηση της L_P είτε με την μεγιστοποίηση της L_D . Παρατηρούμε επίσης ότι αν διατυπώσουμε το πρόβλημα με $b = 0$ το οποίο σημαίνει ότι όλα τα υπερεπίπεδα θα περιέχουν το σημείο ανάφορας , ο περιορισμός (46) δεν θα εμφανισθεί. Αυτό αποτελεί ένα μικρό περιορισμό για χώρους υψηλών διαστάσεων δεδομένου ότι συνεπάγεται τη μείωση των βαθμών ελευθερίας κατά ένα.

Από τα παραπάνω προκύπτει ότι η εκπαίδευση της μηχανής διανυσμάτων υποστήριξης για την γραμμικώς διαχωρίσιμη περίπτωση προκύπτει από τη μεγιστοποίηση του όρου L_D ως προς τα α_i υπό τους περιορισμούς της (46) για θετικά α_i με λύση η οποία δίνεται από την (45). Βλέπουμε ότι υπάρχει ένας πολλαπλασιαστής *Lagrange* για κάθε σημείο εκπαίδευσης. Στη λύση ,αυτά τα σημεία για τα οποία $\alpha_i > 0$ ονομάζονται διανύσματα υποστήριξης και βρίσκονται σε κάποιο από τα υπερεπίπεδα H_1, H_2 . Όλα τα άλλα σημεία έχουν $\alpha_i = 0$ και βρίσκονται είτε στο H_1 είτε στο H_2 είτε σε αυτή τη μεριά των H_1, H_2 για την οποία ισχύουν οι αρχικοί ανισοτικοί περιορισμοί. Για τις μηχανές αυτές τα διανύσματα υποστήριξης αποτελούν κρίσιμα συστατικά στοιχεία του συνόλου εκπαίδευσης. Βρίσκονται πιο κοντά στο όριο της απόφασης: Αν όλα τα υπολοιπα σημεία του συνόλου εκπαίδευσης εξαφανιστούν και η φάση εκπαίδευσης επαναληφθεί από την αρχή , θα προκύψει το ίδιο υπερεπίπεδο.

Εδώ πρέπει να σημειωθεί ότι οι συνθήκες Karush-Kuhn-Tucker παίζουν έναν κεντρικό ρόλο τόσο στη θεωρία όσο και στην πρακτική της βελτιστοποίησης υπό περιορισμούς. Για το πρωτεύων πρόβλημα παραπάνω χρησιμοποιήθηκαν οι ακόλουθες KKT συνθήκες:

$$\frac{\partial L_P}{\partial w_v} = w_v - \sum_i \alpha_i y_i x_{iv} = 0 \quad , v = 1, \dots, d \quad (48)$$

$$\frac{\partial L_P}{\partial b} = - \sum_i \alpha_i y_i = 0 \quad (49)$$

$$y_i (\mathbf{x}_i \mathbf{w} + b) - 1 \geq 0 \quad i = 1 \dots l \quad (50)$$

$$\alpha_i \geq 0 \quad \forall i \quad (51)$$

$$\alpha_i (y_i (\mathbf{w} \mathbf{x}_i + b) - 1) = 0 \quad \forall i \quad (52)$$

Οι συνθήκες KKT ικανοποιούνται στη λύση οποιουδήποτε προβλήματος βελτιστοποίησης, (χυρτού ή μη) με οποιοδήποτε τύπο περιορισμών , δεδομένου

οτι η τομή του συνόλου των εφικτών κατευθύνσεων με το σύνολο των καθοδικών κατευθύνσεων συμπίπτει με την τομή του συνόλου των εφικτών κατευθύνσεων για γραμμικοποιημένους περιορισμούς με το σύνολο των καθοδικών κατευθύνσεων. Αυτή η μάλλον τεχνική υπόθεση της κανονικότητας ισχύει για όλες τις μηχανές διανυσμάτων υποστήριξης , δεδομένου ότι οι περιορισμοί είναι πάντα γραμμικοί. Επιπλέον, το πρόβλημα των SVM είναι κυρτό , και για κυρτά προβλήματα συνθήκες KKT είναι αναγκαίες και επαρκείς για τα w, b, α να είναι λύσεις. Έτσι, η επίλυση του προβλήματος SVM είναι ισοδύναμη με την εύρεση μιας λύσης για τις συνθήκες KKT . Το γεγονός αυτό οδηγεί σε διάφορες προσεγγίσεις για την εύρεση της λύσης.

Βλέπουμε οτι μέχρι στιγμής αυτό που έχουμε κάνει είναι να μετατρέψουμε το πρόβλημα σε ένα πρόβλημα βελτιστοποίησης όπου οι περιορισμοί είναι περισσότερο διαχειρίσιμοι απότι στην περίπτωση του αρχικού προβλήματος. Η ευρεση λύσης για πραγματικά προβλήματα τέτοιου είδους συνήθως απαιτεί αριθμητικές μεθόδους. Παρακάτω θα δούμε πως γενικεύεται η μέθοδος για μη γραμμικά διαχωρίσιμα πρότυπα.

Προκειμένου οι μηχανές διανυσμάτων υποστήριξης να μπορέσουν να χρησιμοποιηθουν σε δεδομένα τα οποία δεν είναι γραμμικά διαχωρίσιμα , εφαρμόζεται μια σχετικά παλιά μέθοδος η οποία όμως είναι ιδιαίτερα αποτελεσματική. Αρχικά παρατηρούμε οτι ο μόνος τρόπος με τον οποίο εμφανίζονται τα δεδομένα στο πρόβλημα εκπαίδευσης είναι μέσω του γινομένου $\mathbf{x}_i \mathbf{x}_j$. Ας υποθέσουμε τώρα οτι εκφράζαμε τα δεδομένα σε κάποιο άλλο (πιθανώς απειροδιάστατο) Ευκλείδειο χώρο H χρησιμοποιώντας μια εκφραση η οποία καλείται Φ ετσι ώστε:

$$\Phi : R^d \mapsto H \quad (53)$$

Τότε , οπως είναι λογικό ο αλγόριθμος εκπαίδευσης θα εξαρτάται απο το γινόμενο των συναρτήσεων στον H δηλαδή απο τις συναρτήσεις στη μορφή $\Phi(\mathbf{x}_i)\Phi(\mathbf{x}_j)$. Τωρα , θεωρώντας μια συνάρτηση πυρήνα K τέτοια ώστε $K(\mathbf{x}_i \mathbf{x}_j) = \Phi(\mathbf{x}_i)\Phi(\mathbf{x}_j)$ θα χρειαζόταν στον αλγόριθμο εκπαίδευσης να χρησιμοποιήσουμε μόνο την K , χωρίς να χρειάζεται καν να ξέρουμε μεμονωμένα ποια είναι η Φ . Για παράδειγμα στην περίπτωση όπου

$$K(\mathbf{x}_i \mathbf{x}_j) = e^{-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / 2\sigma^2} \quad (54)$$

το H είναι απειροδιάστατο , και επομένως δεν θα ήταν ιδιαίτερα εύκολο να δουλέψουμε με τις συναρτήσεις Φ . Παρόλα αυτα αν τα $\mathbf{x}_i \mathbf{x}_j$ αντικατασταθούν απο το $K(\mathbf{x}_i \mathbf{x}_j)$ παντού στον αλγόριθμο εκπαίδευσης , αυτός θα είναι σε θέση να παράξει ένα SVM το οποίο βρίσκεται σε έναν απειροδιάστατο χώρο και μάλιστα σε περίπου ίδιο χρόνο. Για αυτές τις περιπτώσεις ολα όσα αναφεραμε παραπάνω

ισχύουν, εφόσον ακόμα πραγματοποιούμε γραμμικό διαχωρισμό απλα σε άλλον χώρο.

Το ερώτημα εδώ είναι πως χρησιμοποιούμε μια τέτοια μηχανη. Αλλωστε το επίπεδο(που προσδιορίζεται απο το \mathbf{w}) θα βρίσκεται και αυτό στον χώρο H , αλλα στη φάση ελέγχου το SVM χρησιμοποιείται υπολογίζοντας το γινόμενο ενός προτύπου ελέγχου \mathbf{x} με το \mathbf{w} ή πιο συγκεκριμένα υπολογίζοντας το:

$$f(\mathbf{x}) = \sum_{i=1}^{N_S} \alpha_i y_i \Phi(\mathbf{s}_i) \Phi(\mathbf{x}) + b = \sum_{i=1}^{N_S} \alpha_i y_i K(\mathbf{s}_i, \mathbf{x}) + b \quad (55)$$

όπου \mathbf{s}_i αποτελούν τα διανύσματα υποστήριξης. Επομενως μπορούμε ξανα να αποφύγουμε τον μεμονωμένο υπολογισμό του $\Phi(\mathbf{x})$, και αντ αυτου να χρησιμοποιήσουμε τον αντίστοιχο πυρήνα.

Ας υποθέσουμε οτι ο χωρος στον οποίο βρίσκονται τα δεδομένα ονομάζεται L . Εδω αξίζει να σημειωθεί οτι μαζί με το γεγονός οτι το \mathbf{w} πλέον βρίσκεται στον H δεν υπάρχει γενικά κάποιο διάνυσμα του L το οποίο μέσω της Φ να εμφανίζεται στο \mathbf{w} . Εαν υπήρχε η $f(\mathbf{x})$ απο παραπάνω θα μπορούσε να υπολογισθεί σε ένα βημα και να αποφευχθεί το αθροισμα.

Γενικά καποιοι απο τους πρώτους πυρήνες που διερευνήθηκαν για το πρόβλημα της αναγνώρισης προτύπων είναι οι ακόλουθοι:

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{xy} + 1)^p \quad (56)$$

$$K(\mathbf{x}, \mathbf{y}) = e^{-\|\mathbf{x}-\mathbf{y}\|^2/2\sigma^2} \quad (57)$$

$$K(\mathbf{x}, \mathbf{y}) = \tanh(k\mathbf{xy} - \delta) \quad (58)$$

$$(59)$$

Τέλος πρέπει να σημειωθεί οτι γενικά τα SVM είναι δυαδικοί ταξινομητές αλλα μπορούν πολύ ευκολα να συνδυαστούν για να αξιοποιηθούν σε περίπτωση πολλών κλάσεων. Μια απλή αλλα αποτελεσματική περίπτωση συνδυασμού εκπαιδεύει N εναν-εναντίον-υπολοίπων ταξινομητές για την περίπτωση N κλάσεων και ετσι διαχωρίζει τις N κλάσεις.[6],[15]

2.7. Naive Bayesian Classifier

Στη περίπτωση του Μπαεσιανού ταξινομητή, ουσιαστικά υπολογίζουμε την $p(c_j|\mathbf{d})$ η οποία αποτελεί την πιθανότητα η παρατήρηση \mathbf{d} να ανήκει στην κλάση c_j . Σκοπός του είναι να ταξινομήσει νέα πρότυπα σε κλάσεις υπολογίζοντας για το καθένα την υψηλότερη πιθανότητα και ταξινομώντας το στην

αντίστοιχη κλάση. Προκειμένου να υπολογισθεί η συγκεκριμένη πιθανότητα , ο ταξινομητής κάνει χρήση του θεωρήματος του *Bayes*:

$$p(c_j|\mathbf{d}) = \frac{p(\mathbf{d}|c_j)p(c_j)}{p(\mathbf{d})} \quad (60)$$

όπου $p(\mathbf{d}|c_j)$ αποτελεί την πιθανότητα να παραχθεί το πρότυπο d με δεδομένο ότι ανήκει στην κλάση c_j ,ή αλλιώς ότι , το γεγονός ότι ανήκει στη c_j δίνει στο \mathbf{d} αυτή τη πιθανότητα να παραχθεί, και $p(c_j)$ δείχνει την πιθανότητα εμφάνισης της c_j (ουσιαστικά πόσο συχνή είναι η κλάση αυτή στα δεδομένα μας). Τέλος η πιθανότητα $p(\mathbf{d})$ υποδηλώνει την πιθανότητα να εμφανισθεί το \mathbf{d} και μπορεί να αγνοηθεί δεδομένου ότι είναι ίδιο για όλες τις κλάσεις.

Στην παραπάνω σχέση το ζήτημα είναι ο υπολογισμός του $p(\mathbf{d}|c_j)$. Χρησιμοποιώντας τον πιθανοτικό κανόνα αλυσίδας θα είναι:

$$\begin{aligned} p(\mathbf{d}|c_j) &= p(d_1, d_2, \dots, d_n|c_j) \\ &= p(d_1|c_j)p(d_2, \dots, d_n|c_j, d_1) \\ &= p(d_1|c_j)p(d_2|c_j, d_1)p(d_3 \dots d_n|c_j, d_1, d_2) \\ &= p(d_1|c_j)p(d_2|c_j, d_1) \dots p(d_n|c_j, d_1, d_2, \dots, d_{n-1}) \end{aligned}$$

θεωρώντας ότι το πρότυπο \mathbf{d} έχει n χαρακτηριστικά. Βλέπουμε επομένως ότι η πιθανότητα $p(\mathbf{d}|c_j)$ μπορεί να αναλυθεί ως το γινόμενο των αντιστοιχων πιθανοτήτων για το καθένα απο τα χαρακτηριστικά που απαρτίζουν το \mathbf{d} , δεδομένων όμως άλλων χαρακτηριστικών επιπλέον της κλάσης. Τώρα επομένως μένει να υπολογίσουμε τις πιθανότητες αυτές.

Ο λόγος για τον οποίο ο συγκεκριμένος ταξινομητής καλείται απλοικός (*naïve*) είναι ότι θεωρεί , προκειμένου να απλοποιήσει τον υπολογισμό του $p(d|c_j)$, ότι καθένα απο τα χαρακτηριστικά του \mathbf{d} είναι στατιστικά ανεξάρτητο απο τα υπόλοιπα χαρακτηριστικά του. Αυτό έχει σαν αποτέλεσμα οι παραπάνω δεσμευμένες μεταξύ των χαρακτηριστικών πιθανότητες να απλοποιούνται ως εξής:

$$\begin{aligned} p(d_i|c_j, d_{k1}) &= p(d_i|c_j) \\ p((d_i|c_j, d_{k1}, d_{k2}) &= p(d_i|c_j) \\ p(d_i|c_j, d_{k1}, d_{k2}, d_{k3}) &= p(d_i|c_j) \\ &\dots \\ p(d_i|c_j, d_{k1}, d_{k2}, \dots, d_{kn}) &= p(d_i|c_j) \end{aligned}$$

Επομένως μπορούμε να γράψουμε:

$$p(\mathbf{d}|c_j) = p(d_1|c_j)p(d_2|c_j)\dots p(d_n|c_j) \quad (61)$$

και έτσι η τελική σχέση παίρνει την μορφή:

$$p(c_j|\mathbf{d}) = \frac{p(c_j)p(d_1|c_j)p(d_2|c_j)\dots p(d_n|c_j)}{p(\mathbf{d})} \quad (62)$$

το οποίο μπορεί ευκολα να υπολογισθεί.

Ο ταξινομητής αυτός λόγω της παραπάνω υπόθεσης για την ανεξαρτησία των χαρακτηριστικών απλοποιεί πολύ την διαδικασία της μάθησης. Αν και αυτή η υπόθεση είναι σε γενικές γραμμές απλοϊκή, υπάρχουν πολλές περιπτώσεις, όπως για παράδειγμα η αναγνώριση κειμένου ή η ιατρική διάγνωση, όπου ο απλοϊκος ταξινομητής *Bayes* είναι αρκετα αποτελεσματικός και μπορεί να ανταγωνισθεί άλλους πιο περίπλοκους ταξινομητές.

Η επιτυχία του ταξινομητή αυτού σε περιπτώσεις όπου υπάρχει εξάρτηση μεταξύ των χαρακτηριστικών μπορεί να αιτιολογηθεί ως εξής: η βελτιστότητα οσον αφορά την απώλεια (το σφάλμα) σε περίπτωση λάθους κλάσης κατα την ταξινόμηση δεν σχετίζεται υποχρεωτικά άμεσα με την ποιότητα της εκπαίδευσης στην πιθανοτική κατανομή όπου κάνουμε την υπόθεση ανεξαρτησίας. Ανταυτού ένας βέλτιστος ταξινομητής μπορεί να αποκτηθεί εφόσον υπάρχει συμφωνία μεταξύ της πραγματικής και της υποθετικής(υπολογισμένης) κατανομής σχετικά με την πιθανότερη κλάση.

Παρολα αυτά, η παραπάνω εξήγηση είναι αρκετά γενική. Τελικά αυτό που επηρεάζει την απόδοση του *Naive Bayes* είναι τα χαρακτηριστικά των δεδομένων. Υπάρχουν μελέτες που έχουν δείξει ότι για συγκεκριμένες σχεδόν ντετερμινιστικές εξαρτήσεις μεταξύ των χαρακτηριστικών, ο ταξινομητής τα πάει καλά. (Αξιζει να αναφερθεί ότι εξαρτήσεις οι οποίες είναι σχεδόν ντετερμινιστικές αποτελούν συχνό φαινόμενο σε πολλά πρακτικά προβλήματα όπως για παράδειγμα σε συστήματα διαχείρισης και σε κώδικες διόρθωσης λαθών). Μάλιστα, έχει δειχθεί ότι ο *naive Bayes* πετυχαίνει τα καλύτερα αποτελέσματα όταν υπάρχει ανεξάρτησια μεταξύ των χαρακτηριστικών (όπως ήταν αναμενόμενο), καθώς και όταν υπάρχουν συναρτησιακά (και άρα ντετερμινιστικά) εξαρτόμενα χαρακτηριστικά, ενώ δεν είναι τόσο καλός για ενδιαμέσες περιπτώσεις μεταξύ αυτων των δύο άκρων.

Άλλο ένα απο τα ιδιαίτερα χαρακτηριστικά αυτού του ταξινομητή είναι ότι ένα καλό μέτρο για το πόσο αποτελεσματικός είναι, αποτελεί η απώλεια της πληροφορίας που εμπεριέχεται στα χαρακτηριστικά υπο την υποθεση μοντέλου ταξινομητή *Bayes*.

Πιο γενικά ,πρέπει να αναφερθεί ότι το σύστημα αυτό είναι ιδιαίτερα αποδοτικό απο άποψη χρόνου και μνήμης, δεδομένου ότι η μόνη σχετικά απαιτιτική διεργασία που έχει να επιτελέσει , είναι ο υπολογισμός των πιθανοτήτων για το κάθε χαρακτηριστικό ελέγχου.(υλοποιείται ευκολα με πινακα).

Ακόμα ένα πλεονέκτημα του ταξινομητή είναι ότι δεν επηρεάζεται εύκολα απο χαρακτηριστικά τα οποία δεν έχουν σχέση με τις κλάσεις που θέλουμε να ταξινομήσουμε (πχ το χαρακτηριστικό 'χρώμα ματιου ' όταν θέλουμε να ταξινομήσουμε ανθρώπους σε αρσενικά και θυλικά), λόγω του ότι -εφόσον έχουμε αρκετά δεδομένα τα οποία να είναι αντιπροσωπευτικά των κλάσεων που θέλουμε να διαχωρίσουμε - οι πιθανότητες ανα τις διαφορετικές κλάσεις για το μη σχετικό χαρακτηριστικό θα είναι αρκετά κοντά. [17],[18],[19]

2.8. Classifier Ensembles

2.8.1 Εισαγωγή

Κατα καιρούς, πολλοί ερευνητές [Breiman 1996a , Clemen 1989 , Perone 1993 , Wolpert 1992] έχουν ασχοληθεί με την τεχνική του συνδυασμού πολλών σχετικά πιο απλών ταξινομητών για την παραγωγή ενός μεμονωμένου ισχυρότερου ταξινομητή. Πιο συγκεκριμένα ,το σκεπτικό είναι ότι οι ταξινομητές εκπαιδεύονται ξεχωριστά και στη φάση του ελέγχου παρέχουν ο καθένας την απόφαση τους και μια μέθοδος συνδυασμού (πχ απλη ψηφοφορία),συνδυάζει τις αποφάσεις αυτές ,σε μια τελική απόφαση η οποία αντιπροσωπεύει έναν τελικό ταξινομητή και μια ενιαία απόφαση. Συνήθως ο προκύπτον ταξινομητής που παράγεται απο τέτοιες τεχνικές είναι πιο ακριβής από ότι οποιοσδήποτε από τους μεμονωμένους ταξινομητές που απαρτίζουν το σύνολο (*ensemble*).

Τοσο η θεωρητική ,όσο και η εμπειρική έρευνα έχει δείξει ότι ένας καλος ταξινομητής συνόλου είναι αυτος στον οποίο οι επιμέρους ταξινομητες είναι εύστοχοι και πραγματοποιούν σφάλματα σε διαφορετικά τμήματα του συνόλου εκπαίδευσης. Γενικότερα η φιλοσοφία πίσω απο αυτό είναι ότι ο ένας βασικός ταξινομητής αντισταθμίζει τα λάθη του άλλου. Η απλή εκπαίδευση όμως του συστήματος (πχ να βάλουμε απλά το σύνολο δεδομένων όπως είναι σε όλους τους απλού ταξινομητές) δεν μπορεί να λύσει το πρόβλημα. Ένας ταξινομητής συνόλου προκειμένου να πετυχαίνει καλά αποτελέσματα θα πρέπει να έχει επιμέρους ταξινομητές οι οποίοι έχουν πολυμορφία με την έννοια ότι είναι ικανοι σε διαφορετικά σημεία των δεδομένων εκπαίδευσης και προφανώς για να επιτευχθεί κατι τέτοιο θα πρέπει να έχουν μεταξύ τους σχετικά διαφορετική εκπαίδευση.[20],[21]

Οι τεχνικές για την δημιουργία ενός ταξινομητή συνόλου μπορούν να χωρισθούν σε έξι γενικές ομάδες:

1. Δημιουργία με αξιοποίηση των παραμέτρων εκπαίδευσης: Εδώ η πολυμορφία μπορεί να επιτευχθεί με την αλλαγή των παραμέτρων του ταξινομητή για κάθε ξεχωριστό βασικό ταξινομητή. Για παράδειγμα αν έχουμε νευρωνικά δίκτυα θα μπορούσαμε να αλλάζουμε την αρχικοποίηση των βαρών και το ρυθμό μάθησης ανα νευρωνικό ετσι ώστε να αλλάζουμε τις τιμές των βαρών στο κάθε δίκτυο. Τετοιοι ταξινομητες συνόλου γενικά επιτυγχάνουν καλύτερη γενίκευση.
2. Δημιουργία με αξιοποίηση συνάρτησης σφάλματος: Σε αυτη την περίπτωση, εαν δύο βασικοί ταξινομητές έχουν για κάποια δεδομένα τα ίδια ακριβώς σφάλματα, επιβάλλεται ένα σφάλμα στο σύστημα και ολοι οι ταξινομητές εκπαιδεύονται ταυτόχρονα και διαδραστικά μέσω της συσχέτισης (ή για την ακρίβεια του σφάλματος συσχέτισης) μεταξύ των συναρτήσεων σφαλματος τους.
3. Δημιουργία με αξιοποίηση του χώρου χαρακτηριστικών: Σε αυτην την ομάδα, διαφορετικά υποσύνολα του χώρου των χαρακτηριστικών χρησιμοποιούνται για την εκπαίδευση των βασικών ταξινομητών. Αυτη η ομάδα ταξινομητών συνόλου έχει σχετικά χαμηλότερη απόδοση απο τις άλλες ομάδες.
4. Δημιουργία με αξιοποίηση των ετικετων(κλάσεων) εξόδου: Εδώ οι ταξινομητές συνόλου κατασκευάζονται με την αξιοποίηση της εξόδου. Κάθε ταξινομητής παράγεται αλλάζοντας την ετικέτα της κλάσης για ένα τμήμα των δεδομένων εκπαίδευσης, τα οποία επιλέγονται τυχαία απο το αρχικό σύνολο εκπαίδευσης.
5. Δημιουργία με ομαδοποίηση: Ταξινομητές συνόλου μπορούν να παραχθούν με την κατάτμηση των δεδομένων εκπαίδευσης σε μη επικαλυπτόμενες συστάδες και για κάθε τέτοια με την εκπαίδευση ενός βασικού ταξινομητή. Τέτοιοι ταξινομητές ονομάζονται ταξινομητές ομαδοποιημένου συνόλου (*clustered ensemble*). Τα πρότυπα τα οποία έχουν τη τάση να είναι κοντα ως προς Ευκλείδεια απόσταση φυσικά προσδιορίζονται κατα την διαδικασία. Θεωρείται οτι ένα πρότυπο μπορεί να ανήκει μόνο σε μιά ομάδα και επομένως απαιτείται μια διαδικασία επιλογής απο τα δεδομένα για την κλάση στην οποία ανήκουν. Αυτές οι μέθοδοι στοχεύουν στον περιορισμό της πολυπλοκότητας μάθησης των δεδομένων ειδικά σε μεγάλα σύνολα δεδομένων. Αξίζει να σημειωθεί οτι δεν παρέχεται κανένας μηχανισμός για την εύρεση του βέλτιστου αριθμού συστάδων (*clusters*)

Κάποιοι ερευνητές παρέχουν μηχανισμούς για την πιθανοτική κατάτμηση(soft partitioning) των δεδομένων , πράγμα το οποίο μπορεί να οδηγήσει σε καλύτερη απόδοση του ταξινομητή. Σε αυτές τις περιπτώσεις οι ταξινομητές συνόλου παράγονται 1.με την κατάτμηση των δεδομένων σε συστάδες σε διαφορετικά στρώματα και κατόπιν 2. με την εκπαίδευση των βασικών ταξινομητών σε συστάδες διαφορετικών στρωμάτων.

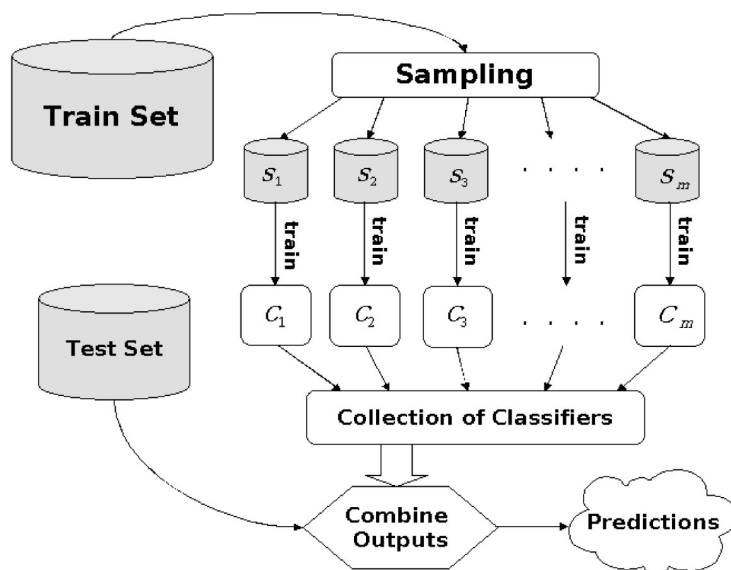
6. Δημιουργία με αξιοποίηση των προτυπων εκπαίδευσης: Τις περισσότερες φορές οι ταξινομητές συνόλου παράγονται απο την αξιοποίηση των δεδομένων εκπαίδευσης, με τους βασικούς ταξινομητές να εκπαιδεύονται σε διαφορετικά δεδομένα. Η διαφορά ανα μέθοδο βρίσκεται στην παραγωγή των διαφορετικών υποσυνόλων εκπαίδευσης.

Δυο δημοφιλείς μέθοδοι για την δημιουργία σχετικά καλών συνόλων (πράγμα που σημαίνει οτι εμπίπτουν στην εκτη κατηγορία),αποτελούν οι μέθοδοι *Bagging* [Breiman 1996a] και *Boosting* [Freund Schapire 1996 , Schapire 1990]. Οι μέθοδοι αυτοί βασίζονται γενικά στην επανατοποθέτηση των δεδομένων με στόχο την απόκτηση διαφορετικών συνόλων εκπαίδευσης για τον κάθε πιο αδύναμο ταξινομητή που ανήκει στο σύνολο.

Μελέτες έχουν δείξει οτι οι δυο αυτές τεχνικές δουλεύουν ιδιαίτερα καλά με δέντρα απόφασης σαν αδύναμους ταξινομητές. Παρολα αυτά , δεν έχει υπάρξει μεγάλη εφαρμογή τέτοιων μεθόδων για νευρωνικά δίκτυα.

Πριν συνεχίσουμε ,είναι ενδιαφέρον οτι γενικά τα συστήματα που βασίζονται σε σύνολα ταξινομητών είναι ιδιαίτερα χρήσιμα σε περιπτώσεις όπου έχουμε πολλά δεδομένα και -μη αναμενόμενα- σε σύνολα όπου δεν υπάρχουν αρκετά δεδομένα. Όταν ο αριθμός των δεδομένων είναι πολύ μεγάλος και η εκπαίδευση ενός μεμονωμένου ταξινομητή γίνεται δύσκολη, τα δεδομένα μπορούν οπως προαναφέρθηκε να χωριστούν στρατηγικά σε μικρότερα υποσύνολα και να χρησιμοποιήσουμε ξεχωριστούς ταξινομητές για το κάθε διαφορετικό υποσύνολο, με το τελικό αποτέλεσμα να προκύπτει ως συνδυασμός. Απο την άλλη μεριά , όταν έχουμε πολύ λιγα δεδομένα , μπορούμε να χρησιμοποιήσουμε πιο αποτελεσματικά την τεχνική του *bootstrapping* για να εκπαιδευσουμε διαφορετικούς ταξινομητές με διαφορετικά σύνολα εκπαίδευσης τα οποία έχουν προκύψει απο την τυχαία επιλογή προτυπων απο το αρχικό σύνολο εκπαίδευσης. Αυτό θα έχει σαν αποτέλεσμα το σύστημα που προκύπτει να έχει μειωμένη μεταβλητότητα σε σχέση με το αρχικό σύστημα.

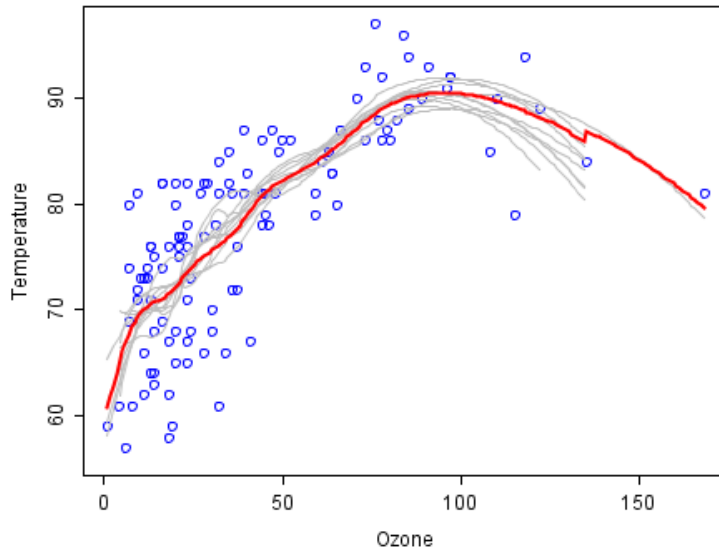
Παρακάτω θα μελετήσουμε πιο διεξοδικά τις τεχνικές αυτές και θα δώσουμε έμφαση σε κάποιους απο τους πιο δημοφιλείς αλγορίθμους για την κάθε τεχνική. [21] [20],[23]



2.12. Η διαδικασία για την παραγωγή ενός ταξινομητή συνόλου. Παρατηρούμε ότι ενδέχεται τα τόσο τα σύνολα εκπαίδευσης για τον κάθε ταξινομητή, όσο και οι ίδιοι οι επιμέρους ταξινομητές να διαφέρουν

2.8.2 Bagging

Η τεχνική *Bagging* αποτελεί μια μέθοδο για την δημιουργία ταξινομητών συνόλου η οποία αξιοποιεί την τεχνική του *bootstrap* και δημιουργεί βασικούς ταξινομητές οι οποίοι εκπαιδεύονται ο καθένας σε μια διαφορετική ανακατανομή του συνόλου εκπαίδευσης. Πιο συγκεκριμένα, το σύνολο εκπαίδευσης του κάθε βασικού ταξινομητή παράγεται από την τυχαία επιλογή, με εναλλαγή πράγμα που σημαίνει ότι κάποιο ή κάποια από τα παραδείγματα μπορεί στο νέο δείγμα να επιλεγούν πάνω από μία φορά- N παραδειγμάτων από το αρχικό σύνολο εκπαίδευσης, με N να αποτελεί το μέγεθος του αρχικού συνόλου εκπαίδευσης. Από τα παραπάνω είναι κατανοητό ότι πολλά από τα αρχικά παραδείγματα ενδέχεται να επαναλαμβάνονται σε ορισμένα σύνολα εκπαίδευσης των βασικών ταξινομητών, ενώ αλλα ενδέχεται να μην εμφανίζονται καθόλου σε κάποια σύνολα. Όπως αναμένεται ο κάθε ξεχωριστός ταξινομητής του συνόλου παράγεται από ένα διαφορετικό από τα σύνολα εκπαίδευσης που έχουν παραχθεί.



2.13.Εφαρμογή της τεχνικής για δεδομένα όζοντος.Οι γκρι γραμμές υποδηλώνουν τους επιμέρους ταξινομητές και η κόκκινη το τελικό αποτέλεσμα.

Απο τα παραπάνω καταλαβαίνουμε ότι στην περίπτωση του *Bagging* ο καθένας από τους παραγόμενους βασικούς ταξινομητές είναι πολύ πιθανό να έχει μεγαλύτερο σφάλμα απότι ένας μεμονωμένος ταξινομητής ο οποίος έχει εκπαιδευθεί με όλα τα δεδομένα απο το αρχικό σύνολο. Παρολα αυτά όταν αυτοι συνδυαστούν σε ένα αποτέλεσμα μπορούν συνήθως να παράξουν ένα αποτέλεσμα το οποίο έχει χαμηλότερο σφαλμα απο αυτό του απλού μεμονωμένου ταξινομητή(τις πιο πολλές φορές η πολυμορφία αυτών των επιμέρους ταξινομητών αντισταθμίζει το σχετικά υψηλό σφάλμα του καθενός τους).

Ο *Breiman* (δημιουργος της τεχνικής) έχει δείξει ότι το *Bagging* αποτελεί μια αποτελεσματική τεχνική για ασταθείς αλγόριθμους μάθησης όπου μικρές διαφορές στο σύνολο εκπαίδευσης μπορούν να προξενήσουν μεγάλες αλλαγές στις προβλέψεις των δεδομένων ελέγχου όπως για παράδειγμα νευρωνικά δίκτυα ή δέντρα απόφασης.[20]

Παρακάτω περιγράφουμε τον αλγόριθμο *random forest* ο οποίος αποτελεί έναν δημοφιλή μηχανισμό που βασίζεται στην τεχνική αυτή.

2.8.2.1 *Random forests*

Ο γενικός αλγόριθμος εκπαίδευσης για το *random forest* εφαρμόζει την

τεχνική του *bagging* σε δέντρα απόφασης (ή παλινδρόμησης αν το πρόβλημα είναι τέτοιου είδους). Δεδομένου ενός συνόλου εκπαίδευσης με $X = x_1, x_2, \dots, x_n$ με αποκρίσεις $U = y_1, y_2, \dots, y_n$, εφαρμόζουμε την τεχνική επαναλαμβανόμενα (έστω B φορές) και εκπαιδεύουμε τα δέντρα το καθένα με κάποιο διαφορετικό από τα B παραγόμενα σύνολα εκπαίδευσης. Επομένως:

Για $b = 1 \dots B$:

1. Εφαρμόζουμε *Bagging* και παράγουμε ένα νέο σύνολο εκπαίδευσης με n στοιχεία, έστω X_b, Y_b .
2. Εκπαιδεύουμε ένα δέντρο απόφασης f_b στα X_b, Y_b

Αφού εκπαιδεύσουμε όλο το σύστημα, η προβλέψη για κάποιο δεδομένο ελέγχου (έστω x') γίνεται πέρνοντας τον μέσο όρο από τους διαφορετικούς ταξινομητές (εαν αναφερόμαστε σε παλινδρόμηση) ως εξής:

$$f_{av} = \frac{1}{B} \sum_{b=1}^B f_b(x') \quad (63)$$

ενω αν αναφερόμαστε σε ταξινόμηση η τελική απόφαση για την ταξινόμηση του στοιχείου προκύπτει από πλειοψηφική ψηφοφορία.

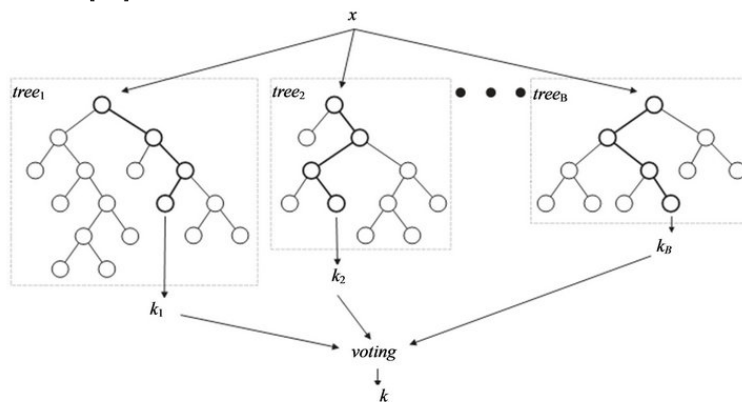
Η διαδικασία του *Bootstrap* οδηγεί σε καλύτερη απόδοση λόγω της μειωμένης μεταβλητότητας που παρέχει χωρίς να αυξήσει την σχετική προτίμηση. Αυτό σημαίνει ότι ενώ οι προβλέψεις από ένα μεμονωμένο δέντρο είναι ιδιαίτερα ευαίσθητες από πιθανό θορυβό που υπάρχει στα δεδομένα εκπαίδευσης του, ο μέσος όρος από πολλά δέντρα δεν είναι αντίστοιχα ευαίσθητος, εφόσον τα επιμέρους δέντρα είναι ασυσχετίστα. Εφόσον η απλή εκπαίδευση πολλών δέντρων με ένα κοινό σύνολο εκπαίδευσης θα έδινε δέντρα τα οποία θα ήταν πολυσυσχετισμένα μεταξύ τους (ή ακόμη και ίδια ίσως αν ο αλγόριθμος εκπαίδευσης είναι ντετερμινιστικός), η τεχνική του *bagging* αποτελεί έναν τρόπο για να το αποφύγουμε αυτό και να φτιαξουμε ένα ασυσχετίστο σύνολο μέσω διαφορετικών συνόλων εκπαίδευσης.

Ο αριθμός των δειγμάτων (αρα και των δέντρων) B είναι ελεύθερος. Συνηθισμένες τιμές είναι από μερικές εκατοντάδες μέχρι λίγες χιλιάδες δέντρα, ανάλογα με το μέγεθος και τη φύση του συνόλου εκπαίδευσης. Ένα καλό B μπορεί να βρεθεί χρησιμοποιώντας cross validation ή παρακολουθώντας την απόδοση δεδομένων για ταξινομητές στα σύνολα εκπαίδευσης των οποίων δεν ανήκουν αυτά τα δεδομένα. (out of bag error)

Η παραπάνω διαδικασία περιγράφει τον απλό αλγόριθμο για την κατασκευή ενός ισχυρού ταξινομητή με δέντρα απόφασης σαν βασικούς ταξινομητές χρησιμοποιώντας την τεχνική του *bagging*. Η διαφορά αυτού με τον αλγόριθμο του

random forest είναι μόνο στο γεγονός ότι στην περίπτωση του *random forest* χρησιμοποιείται ένας σχετικά τροποποιημένος αλγόριθμος εκπαίδευσης δέντρου απόφασης (παλινδρόμησης) ο οποίος επιλέγει σε κάθε κόμβο διαχωρισμού στο δέντρο κατά την μάθηση ένα τυχαίο υποσύνολο των χαρακτηριστικών με τα οποία μόνο θα συνεχίσει την δημιουργία του. Ο λόγος για τον οποίο γίνεται αυτό είναι ότι είναι η συσχέτιση που εξακολουθεί να υπάρχει στη συνηθισμένη τεχνική *bagging* με χρήση δεντρών: Αν κάποιο ή κάποια χαρακτηριστικά είναι ιδιαίτερα κυρίαρχα για την απόκριση, είναι πολύ πιθανό ότι θα επιλεγθούν από παρα πολλά δέντρα του B με αποτέλεσμα υπάρξει εντονη συσχέτιση μεταξύ τους. Με τον τρόπο αυτό εφόσον επιλέγονται τυχαία κάποια χαρακτηριστικά αυτό δεν γίνεται.

Ο αλγόριθμος αυτός έχει ένα σύνολο από καλά χαρακτηριστικά. Κατ'αρχάς, έχει παρα πολύ καλή ευστοχία (*accuracy*), ενώ τρέχει ιδιαίτερα γρήγορα ακόμη και για μεγάλα σύνολα δεδομένων. Επίσης, μέσω της τεχνικής που περιγράφηκε παραπάνω, δίνει μια καλή εικόνα για το ποια από τα υπάρχοντα δεδομένα είναι σημαντικά για την ταξινόμηση και αποτελεί μια καλή πειραματική μέθοδο για την διερεύνηση των σχέσεων μεταξύ των μεταβλητών. Τέλος, αν και μεμονωμένα τα δέντρα απόφασης παρουσιάζουν ευαισθησία στην υπερεκπαίδευση, ο συγκεκριμένος αλγόριθμος δεν υπερεκπεδεύεται. Όσα δέντρα απόφασης και να κατασκευαστούν ο αλγόριθμος θα εξακολουθεί να δουλεύει σωστά.[22]



2.14. Γενική αρχιτεκτονική του *random forest*

2.8.3 Boosting

Το *Boosting* περιλαμβάνει ένα σύνολο μεθόδων. Στόχος είναι ουσιαστικά

να δημιουργηθεί μια σειρά απο βασικούς ταξινομητές. Το σύνολο εκπαίδευσης που χρησιμοποιείται για κάθε μέλος της σειράς επιλέγεται με βάση την απόδοση του προηγούμενου ταξινομητή της σειράς. Πιο συγκεκριμένα ,στο *Boosting* παραδείγματα τα οποία έχουν προβλεφθεί λανθασμένα απο προηγούμενους ταξινομητές στη σειρά, επιλέγονται πιο συχνά απότι παραδείγματα τα οποία έχουν προβλεφθεί σωστά. Αυτό έχει σαν αποτέλεσμα ,η συγκεκριμένη τεχνική να προσπαθεί να παράξει νέους ταξινομητές οι οποίοι είναι καλύτεροι στο να προβλέπουν παραδείγματα για τα οποία η παρούσα απόδοση του μέχρι τώρα συνόλου είναι χαμηλή.[20]

Άλλες πηγές εξειδικεύουν την τεχνική σε συγκεκριμένο αλγόριθμο ο οποίος αξίζει να αναφερθεί για λόγους πληρότητας. Με βάση αυτό, κάθε επανάληψη της μεθόδου δημιουργεί τρεις αδύναμους ταξινομητές: ο πρώτος ταξινομητής C_1 εκπαιδεύεται με κάποιο τυχαία παραγόμενο υποσύνολο των δεδομένων εκπαίδευσης. Το υποσύνολο των δεδομένων εκπαίδευσης για τον δεύτερο ταξινομητή C_2 προκύπτει με βάση το πλέον κατατοπιστικό υποσύνολο (most informative subset) δεδομένου του C_1 . Πιο συγκεκριμένα ,ο C_2 εκπαιδεύεται σε δεδομένα τα οποία είναι μόνο κατα το ήμισυ σωστά ταξινομημένα απο τον C_1 και με τα άλλα μισα να μην ταξινομούνται σωστά. Ο τρίτος ταξινομητής C_3 που παράγεται με βάση αυτόν τον αλγόριθμο εκπαιδεύεται απο τα πρότυπα στα οποία διαφωνούν οι C_1 και C_2 . Οι τρεις ταξινομητές συνδυάζονται μεσω τριπλής πλειοψηφικής ψηφοφορίας. Παρακάτω παρουσιάζεται ο συγκεκριμένος αλγόριθμος κατα βήματα:

Algorithm: Boosting

Input:

- Training data S of size N with correct labels $\omega_i, \Omega = \{\omega_1, \omega_2\}$;
- Weak learning algorithm **WeakLearn**.

Training

1. Select $N_1 < N$ patterns without replacement from S to create data subset S_1 .
2. Call **WeakLearn** and train with S_1 to create classifier C_1 .
3. Create dataset S_2 as the most informative dataset, given C_1 , such that half of S_2 is correctly classified by C_1 , and the other half is misclassified.:
 - a. Flip a fair coin. If Head, select samples from S , and present them to C_1 until the first instance is misclassified. Add this instance to S_2 .
 - b. If Tail, select samples from S , and present them to C_1 until the first one is correctly classified. Add this instance to S_2 .
 - c. Continue flipping coins until no more patterns can be added to S_2 .
4. Train the second classifier C_2 with S_2 .
5. Create S_3 by selecting those instances for which C_1 and C_2 disagree. Train the third classifier C_3 with S_3 .

Test – Given a test instance x

1. Classify x by C_1 and C_2 . If they agree on the class, this class is the final classification.
2. If they disagree, choose the class predicted by C_3 as the final classification.

Polikar, © 2008

Μελέτες έχουν δείξει οτι το σφάλμα του συγκεκριμένου αλγορίθμου έχει

άνω φράγμα. Πιο συγκεκριμένα έχειδειχθεί ότι αν κάποιος αλγόριθμος A χρησιμοποιείται για την δημιουργία τριών ταξινομητών C_1, C_2, C_3 έχει κάποιο σφάλμα ϵ για σύνολο εκπαίδευσης S , τότε το σφάλμα του τελικού ταξινομητή του συνόλου θα είναι ανω φραγμένο από την

$$f(\epsilon) = 3\epsilon^2 - 2\epsilon^3$$

Βλέπουμε εδώ ότι $f(\epsilon) \leq \epsilon$ αν $\epsilon < 1/2$. Από αυτό προκύπτει ότι αν ο A μπορεί να τα πάει λίγο καλύτερα από το να μαντεύει τυχαία, τότε το σύνολο που εκπαιδεύεται με αυτήν την τεχνική για τις τρεις κατανομές του S θα έχει πάντα καλύτερη απόδοση από τον A . Επίσης, το σφάλμα του συνόλου εδώ θα αποτελεί ένα όριο σφάλματος εκπαίδευσης και επομένως είναι αναμενόμενο να παράγεται ένας ισχυρότερος ταξινομητής από τους τρεις πιο αδύναμους βασικούς. Μεσω αναδρομικής εφαρμογής αυτής της τεχνικής μπορούν να δημιουργηθούν πολύ ισχυροί ταξινομητές. Αξίζει να αναφερθεί ότι ένας περιορισμός της μεθόδου είναι ότι εφαρμόζεται μόνο για δεδομένα που αφορούν την ταξινόμηση μεταξύ δύο κλάσεων[22].

Παρακάτω θα περιγράψουμε συνοπτικά τον αλγόριθμο *adaboost* ο οποίος αποτελεί έναν από τους πλέον χρησιμοποιούμενους και αντιπροσωπευτικούς αλγορίθμους της συγκεκριμένης τεχνικής.

2.8.3.1 Adaboost

Σκοπός του συγκεκριμένου αλγορίθμου είναι η σύσταση μιας πολύ δυνατής ομάδας από έιδικούς' απλούς ταξινομητές. Όπως αναμένεται, για κάθε πρότυπο ελέγχου x_i κάθε ειδήμων από την ομάδα έστω k_j που έχει επιλεγεί εκφράζει μια άποψη έστω $k_j(x_i) \in \{-1, 1\}$ και η τελική απόφαση που προκύπτει από την ομάδα των ειδικών K θα είναι:

$$\text{sgn}(C(x_i)) = \text{sgn}\left(\sum_{j=1}^M \alpha_j k_j(x_i)\right) \quad (64)$$

όπου k_j δηλώνει τον κάθε ειδήμωνα, N ο συνολικός αριθμός των ειδικών και sgn είναι η συνάρτηση σήματος. Οι σταθερές $\alpha_j, j = 1 \dots M$ είναι τα βάρη τα οποία θα τοποθετήσουμε στη γνώμη του κάθε ειδικού της ομάδας. Ο κάθε ειδικός βεβαια θα έχει γνώμη τυπου 'ναι' ή 'όχι' (-1 ή 1), επομένως ο παραπάνω σχηματισμός είναι ένας γραμμικός συνδυασμός από ταξινομητές που συνοδεύεται από μία μη γραμμική συνάρτηση απόφασης (έδώ αναλύουμε την περίπτωση ταξινόμησης). Για απλότητα θα θεωρήσουμε ότι μας δίνεται ένα ευρή σύνολο από ταξινομητές που απαρτίζεται έστω από L ταξινομητές (προφανώς με $L \geq M$).

Ο αλγόριθμος πρέπει επομένως να εντοπίσει τον κατάλληλο κάθε φορά ταξινομητή, να τον βάλει στην ομάδα και να του αναθέσει το αντίστοιχα κατάλληλο βάρος. Ο εντοπισμός πραγματοποιείται μέσω του έλεγχου των ταξινομητών του συνόλου χρησιμοποιώντας ένα σύνολο εκπαίδευσης T το οποίο περιλαμβάνει N πολυδιάστατα πρότυπα x_i . Για κάθε σημείο x_i έχουμε μία ετικέτα(κλάση) $y_i = +1$ ή $y_i = -1$. Εξετάζουμε και βαθμονομούμε κάθε ταξινομητή στο σύνολο βάζοντας ένα κόστος e^β κάθε φορά που ο ταξινομητής αποτυγχάνει να ταξινομήσει σωστά ένα πρότυπο εκπαίδευσης και ένα κόστος $e^{-\beta}$ κάθε φορά που ο ταξινομητής επιλέγει την σωστή ετικέτα y_i για κάποιο πρότυπο x_i . Ζητάμε $\beta > 0$ έτσι ώστε οι αποτυχίες να έχουν μεγαλύτερο κόστος από τις επιτυχίες. Μπορεί να φαίνεται παράξενο το ότι βαθμολογούμε τις επιτυχίες με κόστος μεγαλύτερο του 0, αλλά όσο το κόστος της επιτυχίας είναι μικρότερο του αντίστοιχου κόστους της αποτυχίας $e^\beta > e^{-\beta}$ δεν υπάρχει πρόβλημα. Αυτού του είδους η συνάρτηση σφάλματος είναι διαφορετική από την κλασική τετραγωνισμένη Ευκλείδεια απόσταση και καλείται συνάρτηση εκθετικής απώλειας (exponential loss function). Ο αλγόριθμος χρησιμοποιεί αυτή την συνάρτηση σαν κριτήριο σφάλματος.

Όταν ελέγχουμε τους L ταξινομητές φτιάχνουμε έναν πίνακα S στον οποίο καταγράφουμε τις ευστοχίες με 1 και τις αστοχίες με 0 για κάθε ταξινομητή k_j , για κάθε πρότυπο εκπαίδευσης x_i .

Το κύριο σκεπτικό του αλγορίθμου είναι να εξάγουμε έναν ταξινομητή από το σύνολο για κάθε μία από τις M επανλήψεις (M ο αριθμός των ταξινομητών). Τα στοιχεία του συνόλου εκπαίδευσης βαθμονομούνται ανάλογα με την τωρινή τους αναγκαιότητα σε κάθε επανλήψη. Στην αρχή σε όλα τα στοιχεία ανατίθεται το ίδιο βάρος (πχ 1 ή $1/N$ αν θέλουμε να είναι κανονικοποιημένα με άθροισμα για όλα τα βάρη ίσο με 1). Καθώς η επιλογή των ταξινομητών προχωρά, στα πιο δύσκολα παραδείγματα, δηλαδή αυτά για τα οποία η μέχρι τώρα ομάδα δεν έχει καλή απόδοση, ανατίθενται όλο και πιο μεγάλα βάρη. Έτσι, η διαδικασία επιλογής επικεντρώνεται στην διαλογή νέων ταξινομητών από το σύνολο οι οποίοι θα μπορούν να βοηθήσουν με αυτά τα -ακόμα- λανθασμένα ταξινομημένα παραδείγματα. Δεν θα είχε νόημα να φέρουμε στην ομάδα έναν ταξινομητή ο οποίος θα είχε πάντα ή σχεδόν πάντα τα ίδια αποτελέσματα με έναν άλλο ταξινομητή της ομάδας. Αν θέλαμε να έχουμε έναν ταξινομητή πάνω από μια φορά θα μπορούσαμε απλά να πολλαπλασιάσουμε το βάρος του. Η καλύτερη ομάδα είναι αυτή η οποία μπορεί να παρέχει ποικιλία γνώμων. Οι ταξινομητές που διαλέγονται πρέπει να αλληλοσυμπληρώνονται με έναν βέλτιστο τρόπο.

Σε κάθε επανλήψη πρέπει να βαθμονομήσουμε όλους τους ταξινομητές έτσι ώστε να επιλέξουμε τον τωρινό καλύτερο από το σύνολο. Στην m επανλήψη έχουμε ήδη συμπεριλάβει $m - 1$ ταξινομητές στην ομάδα και θέλουμε να φέρουμε

τον επόμενο. Ο παρόν γραμμικός συνδυασμός των ταξινομητών είναι:

$$C_{(m-1)}(x_i) = \alpha_1 k_1(x_i) + \alpha_2 k_2(x_i) + \dots + \alpha_{m-1} k_{m-1}(x_i) \quad (65)$$

και θέλουμε να το επεκτείνουμε σε

$$C_m(x_i) = C_{(m-1)}(x_i) + \alpha_m k_m(x_i) \quad (66)$$

Στην πρώτη επανάληψη ($m = 1$) το $C_{(m-1)}$ είναι 0. Ορίζουμε το συνολικό κόστος ή ολικό σφάλμα το εκτεταμένου ταξινομητή ως

$$E = \sum_{i=1}^N e^{-y_i(C_{(m-1)}(x_i) + \alpha_m k_m(x_i))} \quad (67)$$

όπου τα α_m και k_m ακόμα αναμένεται να προσδιορισθούν βέλτιστα. Εφόσον στόχος μας είναι να διαλέξουμε τον k_m ξαναγράφουμε την εκφραση ως:

$$E = \sum_{i=1}^N w_i^{(m)} e^{-y_i \alpha_m k_m(x_i)} \quad (68)$$

όπου

$$w_i^{(m)} = e^{-y_i C_{(m-1)}(x_i)} \quad (69)$$

για $i = 1, 2, \dots, N$. Στην πρώτη επανάληψη είναι $w_i^1 = 1$ για $i = 1, 2, \dots, N$. Σε επόμενες επαναλήψεις, το διάνυσμα $w^{(m)}$ υποδηλώνει το βάρος που ανατίθεται σε κάθε πρότυπο στο σύνολο εκπαίδευσης στην επανάληψη m . Εδώ μπορούμε να διαχωρίσουμε την (67) σε δύο αθροίσματα έτσι ώστε:

$$E = \sum_{y_i = k_m(x_i)} w_i^{(m)} e^{-\alpha_m} + \sum_{y_i \neq k_m(x_i)} w_i^{(m)} e^{\alpha_m} \quad (70)$$

Αυτό σημαίνει ότι το συνολικό κόστος είναι το κόστος όλων των επιτυχιών συν το κόστος όλων των αστοχιών. Γράφοντας τον πρώτο όρο ως $W_c e^{-\alpha_m}$ και τον δεύτερο ως $W_e e^{\alpha_m}$ έχουμε :

$$E = W_c e^{-\alpha_m} + W_e e^{\alpha_m} \quad (71)$$

Για την επιλογή του k_m η ακριβής τιμή του $\alpha_m > 0$ δεν έχει σχέση αφού για δεδομένο α_m η ελαχιστοποίηση του E είναι ισοδύναμη με την ελαχιστοποίηση του $e^{\alpha_m} E$ και αφού από την προηγούμενη σχέση θα είναι :

$$e^{\alpha_m} E = W_c + W_e e^{2\alpha_m} \quad (72)$$

Εφόσον $e^{2\alpha_m} > 0$ μπορούμε να ξαναγράψουμε την παραπάνω σχέση ως:

$$e^{\alpha_m} E = (W_c + W_e) + W_e(e^{2\alpha_m} - 1) \quad (73)$$

Τώρα τα $W_c + W_e$ είναι το συνολικό άθροισμά W των βαρών όλων των δεδομένων το οποίο για την παρούσα επανάληψη είναι σταθερό. Η δεξιά μεριά της σχέσης ελαχιστοποιείται όταν διαλέξουμε στην επανάληψη m τον ταξινομητή με το χαμηλότερο συνολικό κόστος W_e (και αυτή θα είναι και η μικρότερη δυνατή τιμή του αντίστοιχου βάρους). Ενστικτωδώς αυτό είναι λογικό αφού ο επόμενος ταξινομητής που επιλέγεται θα πρέπει να είναι αυτός με το μικρότερο σφάλμα για το παρόν σύνολο βαρών.

Εχοντας διαλέξει το m -οστο μέλος της ομάδας αυτό που μένει είναι να προσδιορίσουμε το α_m . Απο την παραπάνω σχέση θα είναι:

$$\frac{dE}{d\alpha_m} = -W_c e^{-\alpha_m} + W_e e^{\alpha_m} \quad (74)$$

Εξισώνοντας με το 0 θα έχουμε:

$$-W_c + W_e e^{2\alpha_m} = 0 \quad (75)$$

και επομένως το βέλτιστο α_m θα είναι:

$$\alpha_m = \frac{1}{2} \ln \frac{W_c}{W_e} \quad (76)$$

Δεδομένου ότι έχουμε ορίσει ως W το συνολικό βάρος η έκφραση μπορεί να γραφεί:

$$\alpha_m = \frac{1}{2} \ln \frac{1 - e_m}{e_m} \quad (77)$$

όπου $e_m = W_e/W$ είναι το ποσοστό των λανθασμένα ταξινομημένων προτύπων.

Απο τα παραπάνω προκύπτει ότι δεδομένου ενός συνόλου εκπαίδευσης T το οποίο περιλαμβάνει N σημεία (πρότυπα) x_i και τις αντίστοιχες ετικέτες $y_i = (+1, -1)$ θέτουμε $w_i^{(1)}$ σε όλα τα σημεία x_i . Θέλουμε να δημιουργήσουμε μια ομάδα απο M ταξινομητες επιλέγοντας απο ένα σύνολο L ταξινομητών. Εκτελούμε M επαναλήψεις. Σε κάθε επανάληψη ονομάζουμε W το άθροισμά των βαρών όλων των σημείων με W_e το άθροισμα των βαρών των δεδομένων στα οποία ο ταξινομητής κάνει λάθος και έχει αναθέσει λάθος ετικέτα(κλάση). Με βάση τα παραπάνω ο αλγόριθμος θα έχει την ακόλουθη μορφή κατα βήματα:
Για $m = 1..M$

1. Επιλέγουμε απο το σύνολο τον ταξινομητή k_m ο οποίος ελαχιστοποιεί το :

$$W_e = \sum_{y_i \neq k_m(x_i)} w_i^{(m)} \quad (78)$$

2. Θέτουμε το βάρος α_m του ταξινομητή ως:

$$\alpha_m = \frac{1}{2} \ln \frac{1 - e_m}{e_m} \quad (79)$$

όπου $e_m = W_e/W$

3. Ανανεώνουμε τα βάρη για την επόμενη επανάληψη. Αν το $k_m(x_i)$ είναι αστοχία θέτουμε:

$$w_i^{(m+1)} = w_i^{(m)} e^{\alpha_m} = w_i^{(m)} \sqrt{\frac{1 - e_m}{e_m}} \quad (80)$$

αλλιώς θέτουμε:

$$w_i^{(m+1)} = w_i^{(m)} e^{-\alpha_m} = w_i^{(m)} \sqrt{\frac{e_m}{1 - e_m}} \quad (81)$$

Εδώ πρέπει να γίνουν κάποια σχόλια. Κατάρχας, αναφορικά με τα βάρη, είναι δυνατό να επανασχεδιάσουμε το βήμα ανανέωσης των βαρών έτσι ώστε μόνο οι αστοχίες να οδηγούν σε αναδιαμόρφωση του βαρους.

Επίσης, παρατηρούμε ότι το διάνυσμα $w^{(m)}$ σχεδιάζεται επαναληπτικά. Μπορεί να επαναυπολογίζεται εντελώς σε κάθε βήμα, αλλά είναι πιο αποδοτικό αυτός ο υπολογισμός να γίνεται προσθετικά.

Τέλος, ένας ταξινομητής ο οποίος έχει απόδοση τυχαία (δηλαδή $e_m = 1/2$) πέρνει μηδενικό βάρος. Ένας τέλειος ταξινομητής ($e_m = 0$) θα έπαιρνε άπειρο βάρος εφόσον θα ήταν το μόνο μέλος της ομάδας που θα χρειαζόμασταν. Ένας τέλειος 'ψευτής' ($e_m = 1$) θα έπαιρνε αρνητικό άπειρο βάρος. Απλά θα αντι-στρεφόταν η απόφαση του και θα ήταν η μόνη επιλογή που θα χρειαζόμασταν.

3. Πρακτικό Μέρος

3.1. Συναρτήσεις που χρησιμοποιήθηκαν και διαδικασίες που προηγήθηκαν της εφαρμογής των αλγορίθμων

Στο παρόν κεφάλαιο καταγράφουμε συνοπτικά τις διάφορες συναρτήσεις που χρησιμοποιήθηκαν, καθώς και την προεπεξεργασία η οποία προηγήθηκε πριν από την εφαρμογή των διαφόρων αλγορίθμων η οποία όμως είναι κοινή για τις διάφορες ξεχωριστές εφαρμογές.

Οι συναρτήσεις του Matlab οι οποίες χρησιμοποιήθηκαν είναι οι εξής:

1. `clearvars` : Απομακρύνει όλες τις μεταβλητές από το workspace . Με την σημαία `-except var1 var2 ...` απομακρύνει όλες τις μεταβλητές εκτός από τις αναφερόμενες.
2. `display` : Τυπώνει στο command line του matlab .
3. `find(X)` : Βρίσκει τους δείκτες των μη μηδενικών στοιχείων του X.
4. `load` : Φορτώνει δεδομένα από το δοθέν M-file στο χώρο εργασίας.
5. `mapstd(X)` : Επεξεργάζεται τα δεδομένα πραγματοποιώντας χαρτογράφηση της μέσης τιμής και της τυπικής απόκλισης στο 0 και στο 1 αντίστοιχα (ανα γραμμή).
6. `min /max (X)`: Για διανύσματα επιστρέφει το μεγαλύτερο / μικρότερο στοιχείο του X. Για πίνακες επιστρέφει ένα διάνυσμα γραμμή το οποίο περιέχει σε κάθε στοιχείο του για κάθε στήλη του πίνακα X το μέγιστο/ελάχιστο στοιχείο.
7. `newff(P,T,S)` : Δημιουργεί δίκτυο εμπρόσθιας τροφοδότησης με οπισθοδιάδοση σφάλματος με P,T,S συμβολίζονται οι πίνακες με τα πρότυπα εκπαίδευσης, οι αντίστοιχες ετικέτες τους και οι διαστάσεις του δικτύου αντίστοιχα.
8. `nnz(X)` : Μετρά τον αριθμό των μη μηδενικών στοιχείων.

9. `processpca` : Επεξεργάζεται τα δεδομένα των γραμμών εφαρμόζοντας `pca` (`principal component analysis`) έτσι ώστε οι γραμμές να αποσχετισθούν και να διαταχθούν με βάση την συνεισφορά τους στην συνολική μεταβλητότητα. Επιπροσθέτως οι γραμμές των οποίων η συμβολή είναι πολύ χαμηλή μπορεί να αφαιρεθούν.
10. `randperm(N)` : Επιστρέφει ένα διάνυσμα το οποίο περιέχει μια τυχαία μετάθεση των ακερίων 1:N.
11. `removeconstantrows(X)` : Αφαιρεί από τον πίνακα που περνά σαν όρισμα τις γραμμές με σταθερές τιμές.
12. `round(X)` : Στρογγυλοποίηση προς τον κοντινότερο ακέραιο ως προς τον X.
13. `selforgmap(D)` : δημιουργεί ένα πλέγμα χάρτη αυτοοργάνωσης με βάση τις δοθείσες διαστάσεις με την δοθείσα παραμετροποίηση για τον τύπο του δικτύου.
14. `sim(NET,T)` : Εκτελεί προσομοίωση στο δίκτυο NET των προτύπων ελέγχου T.
15. `size(X)` : Επιστρέφει ένα διάνυσμα με τις διαστάσεις του πίνακα X. Αν έχουμε ως όρισμα `size(X,i)` , επιστρέφει το μέγεθος της διάστασης που πρόσδιορίζεται.
16. `sort(X)` : Ταξινόμηση του X. Αν X πίνακας , τότε ταξινομεί την κάθε στήλη σε σύζουσα σειρά.
17. `sqrt(X)` : Επιστρέφει την τετραγωνική ρίζα των στοιχείων του X.
18. `sum(X)` : Επιστρέφει άθροισμα. Σε περίπτωση διανύσματος επιστρέφει το άθροισμα των στοιχείων του. Σε περίπτωση πίνακα επιστρέφει διάνυσμα γραμμής το οποίο περιέχει τα αθροίσματα όλων των στηλών του πίνακα X.
19. `textread` : Διάβασμα από αρχείο με την μορφή δοθείσας φόρμας.
20. `tic/toc` : Αρχίζει και τελειώνει χρονική καταμέτρηση η οποία εκτυπώνεται.
21. `train(NET,X,T)` : Εκπαιδεύει το νευρωνικό δίκτυο NET με τον πίνακα των προτύπων X και πίνακα κλάσεων T.

22. `xlsread(FILE,SHEET)` : Διαβάζει τα δεδομένα απο το δοθέν αρχείο excel (FILE)για το δοθέν φύλλο (SHEET).
23. `zero(M,N)` : Δημιουργεί έναν πίνακα MXN με ολα τα στοιχεία του ίσα με 0.

[25]

Όσον αφορά τα βήματα προεπεξεργασίας που χρησιμοποιήθηκαν , αρχικά απο τα δεδομένα πρότυπα διαγράφηκαν χαρακτηριστικά τα οποία είναι σταθερά (χρησιμοποιώντας την `removeconstantrows`). Στην συνέχεια κανονικοποιούμε ως προς την μέση τιμή και την διακύμανση των δεδομένων (`mapstd`) και τέλος χρησιμοποιούμε τν ορθογωνικό μετασχηματισμό *pca* ετσι ώστε να αποσυσχετίσουμε τα δεδομένα εισόδου.

Σχετικά με την διαδικασία ολοκλήρωσης της εκπαίδευσης , για λόγους απλότητας , εφόσον δεν μελετάμε στην παρούσα εργασία τρόπους βελτίωσης της, επιλέχθηκε η πιο απλή δυνατη περίπτωση όπου η εκπαιδευτική διαδικασία των δικτύων σταματά με έλεγχο κάθε φορά της ευστοχίας του σύνολου εκπαίδευσης, χωρίς να υπάρχει συνολο επικύρωσης. Η ευστοχία ολοκλήρωσης , κάθε φορά προσδιορίζεται με *cross-validation* ανάλογα με τις ανάγκες του δεδομένου σύνολου και ποικίλει αρκετά εφόσον σε κάποια σύνολα η ευστοχια δεν μπορεί να υπερβεί κάποια δεδομένη τιμή πιθανώς λόγω μεγάλης επικάλυψης των δεδομένων, ενώ σε άλλα μπορεί να φτάνει σε μεγάλες ποσοστιαίες τιμες χωρίς αυτο να είναι καλό , λόγω *overtrain*. Κάθε φορά θέλουμε να προσδιορίζουμε κάποια τιμή που να είναι αρκετά αντιπροσωπευτική των δυνατοτήτων του δικτύου στο δεδομένο σύνολο.

Στην συνέχεια παρουσιάζονται οι αλγόριθμοι που σχεδιάστηκαν. Η δομή που ακολουθείται είναι αυτή η οποία αναφέρεται στην εισαγωγή.

3.2. Μελέτη συμπεριφοράς ομάδας εξειδικευμένων Νευρωνικών Δικτύων

3.2.1 Γενικά.

Σε αυτή την περίπτωση θα μελετηθεί η συμπεριφορά μιας ομάδας νευρωνικών δικτύων η οποία έχει την ακόλουθη ιδιότητα. Αντι τα νευρωνικά να εκπαιδεύονται είτε στο σύνολο είτε σε ένα υποσύνολο των δεδομένων εκπαίδευσης με βάση κάποιο χωρικό κριτήριο, εκπαιδεύονται σε διαφορετικές κατηγορίες των δεδομένων εκπαίδευσης.

Πιο συγκεκριμένα, αρχικά χωρίζουμε τα δεδομένα εκπαίδευσης σε ορισμένες υπερ-κατηγορίες ανάλογα με την κατηγορία στην οποία έχει ταξινομηθεί το συγκεκριμένο πρότυπο. Έτσι, εάν υποθέσουμε ότι έχουμε n πιθανές τελικές κατηγορίες όπου μπορεί να ταξινομηθεί κάποιο πρότυπο εκπαίδευσης, διαλέγεται ένας αριθμός $k < n$ ο οποίος σηματοδοτεί το πλήθος των υπερ-κατηγοριών. Για παράδειγμα, έστω ότι $n = 10$ και επιλέγεται $k = 3$. Αυτό σημαίνει ότι χωρίζουμε τις 10 αρχικές κατηγορίες σε τρεις ομάδες (υπερ-κατηγορίες) πχ την ομάδα 1 που αποτελείται από τις τελικές κατηγορίες 1-3 την ομάδα 2 που αποτελείται από τις κατηγορίες 4-6 και την ομάδα 3 που αποτελείται από τις 7-10.

Αυτές αποτελούν τις ομάδες (υπερκατηγορίες) οι οποίες θα καθορίσουν την εκπαίδευση του κάθε νευρωνικού μας. Έτσι άλλο νευρωνικό θα εκπαιδευτεί στο να ταξινομεί πρότυπα που ανήκουν στην ομάδα 1 και άλλο στο να ταξινομεί πρότυπα που ανήκουν στη ομάδα 2. Συνολικά λοιπόν για κάθε υπερ-κατηγορία (ας ονομάσουμε για διευκόλυνση από εδώ και έπειτα το σύνολο των υπερκατηγοριών $C1$ και το σύνολο των τελικών κατηγοριών $C0$) θα χρησιμοποιήσουμε ένα νευρωνικό δίκτυο (ή θα μπορούσαμε κάποιον άλλο απλό ταξινομητή). Αυτό θα αναλάβει αφού εκπαιδευτεί, για το κάθε πρότυπο του στοιχείου του $C1$ που αντιπροσωπεύει να το μεταφέρει σωστά στο αντίστοιχο στοιχείο του $C0$. (Στο παραπάνω παράδειγμα, αν έχουμε ένα πρότυπο ελέγχου το οποίο ανήκει στην κατηγορία 4, θα πρέπει να ταξινομηθεί από τον ταξινομητή της ομάδας 2 και αυτός θα έχει εκπαιδευτεί να ταξινομεί σε τρεις κατηγορίες οι οποίες αντιπροσωπεύουν τις 4,5 και 6)

Από τα παραπάνω γεννιέται η ακόλουθη απορία: πως θα ξέρει το σύστημα σε ποιο μέλος του $C1$ να αποδώσει το κάθε πρότυπο ελέγχου. Η απάντηση σε αυτό το ερώτημα είναι εύκολη. Θα χρησιμοποιήσουμε ακόμα έναν αλγόριθμο ταξινόμησης του οποίου το καθήκον θα είναι να ταξινομήσει το κάθε πρότυπο

στην ανίστοιχη κατηγορία του $C1$. Έτσι θα έχουμε ένα σύνολο από $k + 1$ ταξινομητές. Το κάθε πρότυπο αρχικά θα ταξινομείται σε κάποια από τις κατηγορίες του $C1$ και κατόπιν ο αντίστοιχος ταξινομητής -εκπρόσωπος που θα έχει επιλεγεί θα φροντίζει για την περαιτέρω ταξινόμηση του προτύπου στην αντίστοιχη κατηγορία του $C0$.

Εδώ αξίζει να παρατηρήσουμε ότι ήδη ο πρώτος ταξινομητής ανάλογα με την ομάδα που θα κατηγοριοποιήσει το κάθε πρότυπο περιορίζει τις επιλογές για την κατηγορία του $C0$ που ανήκει το πρότυπο στις κατηγορίες που ανήκουν στην ομάδα που έχει επιλεγεί. Φυσικά το συγκεκριμένο σύστημα αφορά σύνολα δεδομένων τα οποία αποτελούνται από σχετικά μεγάλο αριθμό κλάσεων. Είναι κατανοητό ότι αν έχουμε για παράδειγμα ένα σύνολο δεδομένων με δύο μόνο κλάσεις, το συγκεκριμένο σύστημα δεν έχει νόημα εφόσον το k το οποίο θα πρέπει να επιλεγεί πρέπει να είναι μικρότερο από το n προκειμένου ο αλγόριθμος να έχει νόημα. Αφού όμως το $n = 2$ αναγκαστικά το σύνολο $C1$ θα έχει μια μόνο κατηγορία, και δεν θα χρειάζεται επιπλέον ταξινομητής για να αποφανθεί στο που (ποιο στοιχείο του $C1$) ανήκει κάποιο πρότυπο. Έτσι το όλο σύστημα θα χρειαστεί μόνο k ταξινομητές. Όμως $k = 1$. Άρα αναγόμεστε στην περίπτωση απλού ταξινομητή.

Αντίθετα παρατηρούμε ότι αν το σύνολο έχει από τρεις κατηγορίες και πάνω ο αλγόριθμος έχει (κάποιο τουλάχιστον μόνο για τρεις) νόημα. Πράγματι εάν $n = 3$ και δεδομένου ότι ζητάμε $k < n$ εάν θέσουμε $k = 1$ θα αναχθούμε και πάλι στην περίπτωση ενός μόνο απλού ταξινομητή. Αν όμως θέσουμε $k = 2$, η μία ομάδα τότε (το ένα στοιχείο του $C1$) θα περιλαμβάνει 1 και η άλλη ομάδα (το άλλο στοιχείο του $C1$) 2 κλάσεις. Προφανώς για την ομάδα που περιέχει μία μόνο κλάση δεν θα χρειαστεί επιπλέον ταξινομητής, αφού κάθε πρότυπο το οποίο ανήκει σε αυτήν την ομάδα, θα αντιστοιχηθεί απευθείας στην μοναδική κλάση που αυτή αντιπροσωπεύει. Για την ομάδα όμως που περιλαμβάνει δύο κλάσεις, θα χρειαστούμε επιπλέον ταξινομητή για να αποφανθεί σε ποια κλάση από τις δύο ανήκει κάποιο πρότυπο που κατατάχθηκε στην ομάδα του. Επιπλέον θα χρειαστούμε ακόμα έναν ταξινομητή για το παραπάνω επίπεδο (που θα αποφασίζει δηλαδή το στοιχείο του $C1$). Έτσι, μπορεί να ορισθεί το συγκεκριμένο σύστημα και αν και κάπως λειψα και για τρεις κατηγορίες και σε αυτή τη περίπτωση θα έχει 2 ταξινομητές.

Μια άλλη παρατήρηση η οποία αξίζει να αναφερθεί σχετίζεται με την μείωση των κλάσεων εξόδου για το σύστημα. Πράγματι, παρατηρούμε ότι μια ιδιότητα του συγκεκριμένου συστήματος είναι ότι βάζει περισσότερους ταξινομητές οι οποίοι όμως έχουν πιο απλό έργο από τον ένα που έχουμε στην περίπτωση ενός ταξινομητή. Πιο συγκεκριμένα, σε αυτό το σύστημα έχουμε ένα σύνολο k ταξινομητών το οποίο οι οποίοι έχουν ίδιο αριθμό χαρακτηριστικών εισόδου, αλλά μικρότερο αριθμό κλάσεων εξόδου, και προφανώς μικρότερο αριθμό

προτύπων εκπαίδευσης καθώς και έναν με ίδιο αριθμό προτύπων εκπαίδευσης ,αλλά με μικρότερο αριθμό κλάσεων(για τον μετα ταξινομητή ,αφου $k < n$),σε σχέση με την περίπτωση ενός απλού ταξινομητή.

Είναι σημαντικό οτι πρέπει να προσεχθεί ο τρόπος με τον οποίο χωρίζονται οι κατηγορίες δεδομένου οτι αν δεν είμαστε προσεκτικοί μπορεί να δώσουμε πολλά πρότυπα σε κάποιο ταξινομητή και πολύ λιγα σε κάποιον άλλο κατα την διαδικασία της εκπαίδευσης ανάλογα με τις επιμέρους κατηγορίες τις οποίες καλείται ο καθέννας να ταξινομήσει. Γενικά η φιλοσοφία του συστήματος είναι να μετατρέψει ένα δύσκολο έργο σε πόλλα εύκολα -διαίρεε και βασιλευε- επομένως θα ήταν ανούσιο για παράδειγμα να πιέσουμε κάποιον απο τους βασικούς ταξινομητές του συστήματος με πολλές κλάσεις, και να αφήσουμε με λιγότερο φόρτο τους υπολοιπους. Γενικά όσο καλύτερα ισοκατανεμηθεί η εργασιά τόσο καλύτερα αναμένεται να είναι και τα αποτελέσματα του συγκεκριμένου συστήματος.

Κατι ακόμα που πρέπει να σημειωθεί είναι οτι το ποσοστό των σωστά ταξινομημένων προτύπων για το συγκεκριμένο σύστημα εξαρτάται απο όλους τους ταξινομητες του. Ειδικότερα, αρχικά ο ταξινομητής ο οποίος ταξινομεί στο σύνολο $C1$ καθορίζει σε πολύ μεγάλο βαθμό την συνολική ευστοχία του συστήματος αφού αυτός θα ταξινομήσει σε κάποια ομάδα όλα τα πρότυπα ελέγχου. Αν ταξινομήσει λάθος κάποιο πρότυπο , αυτό αυτομάτως ταξινομείται λάθος συνολικά.

Κατα συνέπεια ο ταξινομητής αυτός έχει μάλλον τον σημαντικότερο ρόλο. Απο την άλλη μεριά , οι υπόλοιποι ταξινομητες ,οι οποίοι γενικά αναμένεται να τα πηγαίνουν γενικά καλά απο αποψη ευστοχίας και να έχουν μικρούς χρόνους εκπαίδευσης, παίζουν και αυτοί σημαντικό ρόλο με τον τρόπο τους. Αμα κάποιος απο αυτούς δεν εκπαιδευτει σωστα , τα χαμηλά ποσοστά του θα επηρεάσουν το συνολικό ποσοστό επιτυχίας (σωστα ταξινομημένων προτύπων) του συστήματος.

Με βάση τα παραπάνω μια σχέση η οποία θα μπορούσε να δώσει με αρκετή ακρίβεια την συνολική ευστοχία του συστήματος είναι η ακόλουθη:

$$acc = acc_{C1} \sum_{i=1}^k w_i acc_i \quad (82)$$

όπου acc είναι η συνολική ευστοχία του συστήματος, acc_{C1} είναι η ευστοχία του ταξινομητή που ταξινομεί για το $C1$ και acc_i αποτελούν τις ευστοχίες του καθενός απο τους υπόλοιπους ταξινομητές για τις k ομάδες. Το βάρος w_i αποτελεί το πόσο σημαντικός είναι ο συγκεκριμένος αριθμός για το σύνολο και θα μπορούσε να προσδιορισθεί ως το πλήθος των προτύπων ελέγχου που ανήκουν σε αυτη την κατηγορία προς το συνολικό πλήθος προτύπων ελέγχου.

Η σχέση αυτή πηγάζει απο το γεγονός οτι ο αρχικός ταξινομητής θα προσδιορίσει την συνολικη ευστοχία για όλους τους άλλους. Έτσι για παράδειγμα εαν η ευστοχία του αρχικού (του $C1$) ταξινομητή είναι 80 τοις εκατο ,μπορούμε να πούμε οτι περίπου τα 80 απο τα 100 πρότυπα περνανε σωστά για να ταξινομηθούν απο οποιοδήποτε ταξινομητή του επόμενου επιπέδου. Στο επίπεδο αυτό, όπως είναι λογικό , το βάρος του κάθε ταξινομητή καθορίζεται απο το πλήθος των στοιχειων που ανήκουν στην ομάδα που αντιπροσωπεύει.Γενικά η ευστοχία του ανώτερου επιπέδου 'μεταδίδεται' στο κατώτερο. Μια βελτίωση αυτου φυσικά θα μπορούσε να είναι η αντικατάσταση της συνολικής ευστοχίας απο τα επι μέρους ποσοστά ανα κατηγορία, αλλα σε κάθε περίπτωση αυτή αποτελεί μια πολύ απλή σχέση η οποία δίνει με σχετικά καλή αρίβεια μια γρήγορη προσέγγιση του αποτελέσματος.

Συνολικά λοιπον απ όλα τα παραπάνω γινεται κατανοητό οτι αυτό το σύστημα βασίζεται στην μέθοδο διαίρει και βασίλευε και αν και δεν είναι ιδιαίτερα χρήσιμο για σύνολα δεδομένων με λίγες κλάσεις εξόδου, μπορεί να βοηθήσει ιδιαίτερα αν έχουμε πολλές κλάσεις στην έξοδο. Παραταυτα, αν και έχει τις προουθέσεις για μεγάλα σύνολα να αποτελεί ένα καλύτερο τοσο απο άποψη χρόνου όσο και απο άποψη απόδοσης σύστημα απότι ένας μεμονωμένος ταξινομητής, πρέπει να τονισθεί οτι χρειάζεται ιδιαίτερη προσοχή δεδομένου οτι το σύστημα είναι ιδιαίτερα ευαίσθητο αφού η απόδοση του εξαρτάται απο τις διαφορετικες αποδόσεις πολλών επιμέρους ταξινομητών.

3.2.2 Υλοποίηση

Στην πρώτη περίπτωση υλοποιούμε το σύστημα για την ταξινόμηση των διαφόρων συνόλων δεδομένων χρησιμοποιώντας έναν ταξινομητή $k - nn$ σαν ταξινομητή στο σύνολο $C1$ και κατόπιν χρησιμοποιούμε για τις επιμέρους ταξινομήσεις στις τελικές κατηγορίες ταξινομητές νευρωνικών δικτύων. Η εκπαίδευση των επιμέρους ταξινομητών αυτών γίνεται με τον συνήθες τρόπο χρησιμοποιώντας τις γνωστές συναρτήσεις που δίνονται απο το matlab .

Η διαδικασία περιλαμβάνει ως συνήθως αρχικά την ανάγνωση όλων των δεδομένων και κατόπιν την δημιουργία του πίνακα εκπαίδευσης , καθώς και του αντίστοιχου πίνακα ετικετών -κλάσεων , οπως και την κατασκευή των ανίστοιχων πινάκων ελέγχου. Η κατασκευή των πινάκων αυτών εαν δεν δίνονται έτοιμοι , γίνεται με τον διαχωρισμό των αρχικών δοθέντων πινάκων οι οποίοι απλά περιλαμβάνουν τα δεδομένα και τις αντίστοιχες κλάσεις στους παραπάνω υποπίνακες εκπαίδευσης -ελέγχου. Πρίν όμως πραγματοποιηθεί ο διαχωρισμός αυτός πάντα πραγματοποιούμε μια τυχαία μετάθεση στο σύνολο των δεδομένων ετσι ώστε

να μην είναι σταθερό σε κάθε εκτέλεση ποια δεδομένα είναι εκπαίδευσης και ποια είναι ελέγχου. (Φυσικά εάν δίνονται ήδη απο πριν έτοιμοι οι πίνακες εκπαίδευσης και ελέγχου αυτο το βήμα πραγματοποιείται στους επιμέρους πίνακες και όχι στους αρχικούς πραγμα που όμως σημαίνει οτι τα δεδομένα εκπαίδευσης και ελέγχου κάθε φορά είναι ίδια απλά έχουν διαφορετική σειρά). Στη συνέχεια πραγματοποιείται η προεπεργασία των δεδομένων κατα τα γνωστά.

Στον συγκεκριμένο αλγόριθμο πραγματοποιούμε επιπλέον το βήμα του διαχωρισμού των δεδομένων εκπαίδευσης (και των αντίστοιχων ετικετων) ανάλογα με την κατηγορία $C1$ στην οποία ανήκει το κάθε δεδομένο εκπαίδευσης με βάση τον διαχωρισμό που εμείς αυθαίρετα έχουμε επιλέξει με βάση τις κλάσεις των δεδομένων εκπαίδευσης όπως αναφέρεται παραπάνω. Αυτός ο διαχωρισμός θα αποτελέσει τον πυρήνα της εκπαίδευσης καθώς κάθενας απο τους χαμηλότερου επιπέδου ταξινομητές θα εκπαιδευθεί στο αντίστοιχο σύνολο. Έτσι ο καθένας απο τους ταξινομητές αυτούς θα μάθει να ταξινομεί σε κάποιο υποσύνολο του συνόλου των πιθανων κατηγοριών των δεδομένων και μεταξύ αυτων των υποσυνόλων δεν θα υπάρχει κάποια τομη. Άρα ο καθένας ταξινομητής εξειδικεύεται σε κάποιο υποσύνολο των δεδομένων με κριτήριο την κλάση στην οποία τα δεδομένα ανήκουν.

Εφόσον λοιπόν ξέρουμε τις κλάσεις των δεδομένων εκπαίδευσης μπορούμε να τα διαχωρίσουμε για να εκπαιδεύσουμε τον καθένα απο τους ταξινομητές αυτούς (του χαμηλότερου επιπέδου).

Προκειμένου να κάνουμε πιο αποδοτική την εκπαίδευση των δικτύων μας πραγματοποιούμε ακόμα ένα βήμα προεπεργασίας για όλα τα νευρωνικά του χαμηλότερου επιπέδου. Δεδομένου οτι το κάθε νευρωνικό ουσιαστικά πλέον διαχωρίζει απο λιγότερες κλάσεις, είναι προφανές οτι δεν χρειάζεται να υπάρχουν οι επιπλέον σταθερές γραμμές στο κάθε επιμέρους σύνολο κλάσεων (δηλαδή στα `TrainDataTargets1-n`) που υποδηλώνουν κλάσεις στις οποίες ο προς εκπαίδευση ταξινομητής δεν πρόκειται να ταξινομήσει κάποιο πρότυπο λόγω του παραπάνω διαχωρισμού. Επομένως προκειμένου να μειώσουμε το υπολογιστικό κόστος χρησιμοποιούμε την `removeconstantrows` και κάθε φορά πριν απο την εκπαίδευση του κάθε επιμέρους ταξινομητή διαγράφουμε τις αντίστοιχες σταθερές γραμμές στο προς εκπαίδευση υποσύνολο ετικετών.

Στη συνέχεια πραγματοποιούμε την διαδικασία εκπαίδευσης. Κατα τα συνηθισμένα χρησιμοποιουμε τις γνωστές συναρτήσεις και την παραμετροποίηση των νευρωνικών που έχουμε επιλέξει, με βασικό κριτήριο την ευστοχία του νευρωνικού στα δεδομένα εκπαίδευσης (όσο πιο απλό κριτήριο γίνεται).

Η ευστοχία τερματισμού επιλέγεται 'χειροκίνητα' ανάλογα με το σύνολο εκπαίδευσης και συνήθως προσδιορίζεται μετα απο αρκετές επαναλήψεις, σε συνάρτηση με το πόσο καλά αποδίδει ο ταξινομητής (ή εδώ το σύστημα) στα δεδομένα ελέγχου καθώς και με το πόσο χρόνο κάνει το σύστημα για να φτάσει

στο δεδομένο αποτέλεσμα.

Κατι ακόμα που αξίζει να αναφερθεί σχετικά με την μοντελοποίηση της διαδικασίας εκπαίδευσης , είναι οτι επιλέχθηκαν ως επιμέρους εξειδικευμένοι ταξινομητές σχετικά μικρα δίκτυα (με μέγεθος [5 4]) διότι τα επιμέρους δίκτυα έχουν σχετικά μικρότερο φόρτο ως προς το σύνολο των δεδομένων τόσο απο αποψη ποσότητας δεδομένων εκπαίδευσης προς επεξεργασία , όσο και απο αποψη πλήθους κατηγοριών εξόδου. Η εκπαίδευση επομένως μπορεί να γίνει εξίσου αποδοτικά (αλλα πιο γρηγορα και με λιγότερο κόστος σε μνήμη) απο μικρότερα δίκτυα.

Αφού ολοκληρωθεί η διαδικασία της εκπαίδευσης των εξειδικευμένων ταξινομητών πραγματοποιούμε την εκπαίδευση του ανώτερου επιπέδου ταξινομητή(για την ταξινόμηση στις υπερομάδες του $C1$). Εδώ έχουμε δύο περιπτώσεις: εξετάζουμε την περίπτωση εκπαίδευσης και εφαρμογής νευρωνικού δικτύου ως τέτοιο ταξινομητή , καθώς και την περίπτωση εφαρμογής ταξινομητή knn για το συγκεκριμένο πρόβλημα.

3.2.2.1 Περίπτωση Νευρωνικού Δικτύου σαν Ταξινομητή υπερομάδων

Στην πρώτη περίπτωση όπου έχουμε νευρωνικό δίκτυο δεν θα έχουμε κάποια ουσιαστική αλλαγή στην διαδικασία της εκπαίδευσης σε σχέση με τα παραπάνω, απλα όπως είναι λογικό θα πρέπει να του δώσουμε για την εκπαίδευση του σαν πιθανές ετικέτες - κλάσεις μια αντίστοιχία των κλάσεων των δεδομένων εκπαίδευσης στις αντίστοιχες υπερκλάσεις τους με βάση τα όσα αναφέρθηκαν παραπάνω. Ετσι στο πρόγραμμα σχηματίζουμε κάθε φορα έναν πίνακα αρχικοποιημένο με μηδενικά (διαστάσεων: πλήθος υπερομάδων X μέγεθος δεδομένων εκπαίδευσης) όπου το αντίστοιχο 1 σε κάθε στήλη αντιστοιχεί στην αντίστοιχη υπερομάδα στην οποία ανήκει το συγκεκριμένο δεδομένο εκπαίδευσης. Εκπαιδεύουμε λοιπόν τον ταξινομητή αυτόν με αυτό το σύνολο σαν κλάσεις και έτσι αυτός θα καθορίζει ποιος (εκπαιδευένος πλέον ' εξειδικευμένος ταξινομητής) θα αποφανθεί για την κλάση κάποιου δεδομένου ελέγχου χρησιμοποιώντας αυτές τις υπερομάδες που αντιστοιχούν η κάθεμια σε ένα σύνολο κλάσεων.

Κατόπιν εφαρμόζουμε το σύνολο των δεδομένων ελέγχου στο σύστημα έτσι ώστε να μετρήσουμε την απόδοσή του. Για να γίνει αυτό ,αρχικά χρησιμοποιούμε την συνάρτηση sim στα δεδομένα ελέγχου στον ταξινομητή των υπερομάδων έτσι ώστε αυτός να αποφανθεί σε ποια υπερομάδα ανήκει το κάθε δεδομένο ελέγχου. Αφού πάρουμε τον πίνακα που προκύπτει (ο οποίος θα έχει διαστάσεις: πλήθος υπερομάδων X μέγεθος δεδομένων ελέγχου), ξέρουμε με

βάση το γεγονός ότι στην εκπαίδευση του δώσαμε πίνακα μηδενικών με τα 1 να συμβολίζουν την απόφαση της υπερομάδας για κάποιο δεδομένο εκπαίδευσης ότι η μέγιστη πεποίθηση του νευρωνικού για κάποια υπερομάδα θα είναι η μέγιστη τιμή που υπάρχει ανα στήλη. (Φυσικά η κάθε στήλη όπως πάντα αντιπροσωπεύει ένα πρότυπο). Με αυτό σαν δεδομένο , μεταφέρουμε τα αποτελέσματα σε έναν δεύτερο πίνακα ίδιου μεγέθους. Σε αυτόν , με βάση τα παραπάνω , η μέγιστη τιμή της κάθε στήλης μεταφράζεται σε 1 και οι υπόλοιπες σε 0 προκειμένου να έχουμε μια κατα κάποιο τρόπο απόλυτη τελική απόφαση για την κατηγορία του κάθε δεδομένου ελέγχου.

Στη συνέχεια για διευκόλυνση μεταφέρουμε τα δεδομένα σε ένα διάγραμμα ακεραίων όπου το κάθε 'κελί' συμβολίζει την αντίστοιχη στήλη και η κάθε τιμή συμβολίζει την θέση του δεδομένου 1 του προηγούμενου πίνακα.

Πλέον είμαστε έτοιμοι να εκτελέσουμε τις προσομοιώσεις για τα δεδομένα ελέγχου στους χαμηλότερου επιπέδου εξειδικευμένους ταξινομητές. Θα μπορούσαμε να εκτελέσουμε ξεχωριστά προσομοίωση για το κάθε πρότυπο ανάλογα με την υπερομάδα στην οποία ανήκει με το αντίστοιχο εξειδικευμένο νευρωνικό, αλλά αυτό δεν είναι αποδοτικό χρονικά πιθανώς λόγω των πολλαπλών ξεχωριστών κλίσεων της συνάρτησης *sim* (και των υπολοίπων κλίσεων που αυτή πραγματοποιεί. Αντ αυτού πραγματοποιούμε την κάθε προσομοίωση όλου του συνόλου ελέγχου για όλους τους εξειδικευμένους ταξινομητές και μετά διαλέγουμε με βάση το παραπάνω διάγραμμα , το αντίστοιχο αποτέλεσμα που θεωρείται κατάλληλο.

Εδώ είναι ιδιαίτερα σημαντικό να τονίσουμε ότι προκειμένου να έχουμε το τελικό αποτέλεσμα της απόφασης για το ποια κλάση έχει επιλέγει χρειάζεται ακόμα ένα βήμα. Δεδομένου ότι το αποτέλεσμα του κάθε ταξινομητή θα είναι ένα υποσύνολο του συνόλου των κλάσεων , το αποτέλεσμα πρέπει να επανέλθει κατα κάποιον τρόπο στην αρχική μορφή του συνόλου των κλάσεων. Αν και ξέρουμε ότι έχει επιλεγεί κάποιος δεδομένος ταξινομητής , θα πρέπει να δείξουμε στο τελικό αποτέλεσμα ποια είναι η τελική κλάση για κάποιο πρότυπο ελέγχου. Αυτό το πετυχαίνουμε φτιάχνοντας τα αποτελέσματα των προσομοιώσεων έτσι ώστε να έχουν το σύνολο των κλάσεων και για όποια κλάση είμαστε σίγουροι ότι δεν θα επιλέγει απο τον συγκεκριμένο ταξινομητή , της βάζουμε 0. Έτσι καταλήγουμε με έναν αριθμό πινάκων (οσοι οι εξειδικευμένοι ταξινομητές) με διαστάσεις ίδιες με αυτές των κλάσεων στόχων των δεδομένων ελέγχου. Έτσι με το παραπάνω βήμα της επιλογής με βάση τα αποτελέσματα απο το διάγραμμα, κατασκευάζουμε έναν πίνακα ο οποίος είναι έτοιμος για σύγκριση με τον πίνακα των αποτελεσμάτων.

Τέλος χρησιμοποιούμε την συνάρτηση για τον υπολογισμό των μεγεθών απόδοσης (accuracy , precision, recall) και έχουμε τα αποτελέσματα της απόδοσης του συστήματος.

3.2.2.2 Περίπτωση *knn* σαν Ταξινομητή υπερομάδων

Η συγκεκριμένη υλοποίηση δεν έχει μεγάλες διαφορές από την παραπάνω, απλά χρησιμοποιούμε ταξινομητή *knn* αντι νευρωνικού δικτύου για ταξινομητή υπερομάδων. Το μόνο που αξίζει εδώ να αναφερθεί είναι ότι λόγω της φύσης του συγκεκριμένου ταξινομητή δεν υπάρχει για τις υπερομάδες φάση εκπαίδευσης. Για κάθε πρότυπο ελέγχου ταξινομούμε με βάση τις υπερομάδες που ανήκουν τα *k* γειτονικά πρότυπα εκπαίδευσης. Στην υλοποίηση που έχουμε πραγματοποιήσει πάμε πλειοψηφικά με ένα αποτέλεσμα.

3.2.3 Αποτελέσματα

Ο συγκεκριμένος αλγόριθμος εφαρμόστηκε σε 4 από τα σύνολα εκπαίδευσης που έχουμε, δεδομένου ότι για να έχει εφαρμογή του κάποιο νόημα, όπως αναφέρθηκε παραπάνω θα πρέπει να έχουμε γενικά πάνω από τρεις κατηγορίες στην έξοδο.

Πιο συγκεκριμένα, εφαρμόστηκε για το σύνολο αναγνώρισης γραμμάτων από διάφορα ειδικά χαρακτηριστικά του κάθε γράμματος, για το σύνολο ταξινόμησης ασθενών σε ομάδες καρκίνου, για το σύνολο αναγνώρισης εννιαίων σχηματισμών σε εικόνα, καθώς και για το σύνολο *abalone*.

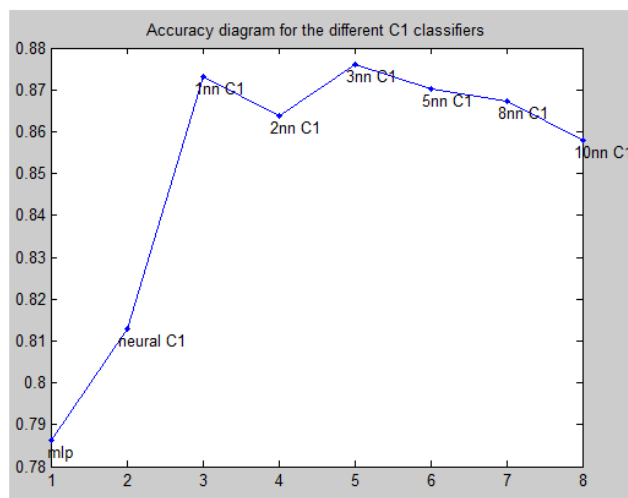
Παρακάτω παραθέτουμε για κάθε ένα από τα σύνολα όπου εφαρμόστηκε ο αλγόριθμος τους πίνακες των αποτελεσμάτων της ευστοχίας και της ανάκλησης (*recall*) ανά κατηγορία για την ταξινόμηση των δεδομένων. Για σύγκριση και μεγαλύτερη κατανόηση εφαρμόζουμε και ταξινόμηση με απλό *mlp* καθώς και επίσης διαφορετικές εκδοχές του αλγόριθμου που αναπτύχθηκε παραπάνω. Στη συνέχεια παρουσιάζουμε τις αντίστοιχες γραφικές παραστάσεις που προκύπτουν (για κάθε σύνολο) και επίσης πίνακες με τους χρόνους εκτέλεσης που χρειάστηκαν για την κάθε περίπτωση.

| Type/Neighbors | Neur | 1nn | 2nn | 3nn | 5nn | 8nn | 10nn | mlp |
|----------------|---------|------|-------|-------|-------|-------|-------|---------|
| Letters | 100-200 | 109 | 130 | 150 | 200 | 270 | 330 | 100-160 |
| CTG | 20-30 | 9-12 | 10-12 | 11-13 | 11-13 | 14-18 | 16-20 | 40-50 |
| Picture | 4-6 | 2-3 | 2-4 | 3-4 | 3-4 | 3-5 | 3-5 | 4-6 |
| Abalone | - | - | 23 | 24 | 30 | 35 | 40 | 18 |

Οι πίνακες των τιμών βρίσκονται στο επισυναπτόμενο αρχείο. Δεν πραγματοποιήθηκαν γραφικές παραστάσεις για την ακρίβεια του συστήματος λόγω πιθανών απροσδιοριστιών (0/0) σε ορισμένες εφαρμογές κάποιων συνόλων, παρόλα αυτά έχουμε πίνακα precision

Τέλος είναι σημαντικό να σημειωθεί ότι κάθε ένα από τα αποτελέσματα έχει προέλθει από τον μέσο όρο δέκα εκτελέσεων, προκειμένου να εξαιρεθεί κατά το δυνατό η μεταβλητότητα των αποτελεσμάτων που προέρχεται από την τυχαία μετάθεση που πάντα εφαρμόζουμε πριν την εφαρμογή του κάθε αλγορίθμου.

3.2.3.1 Letters



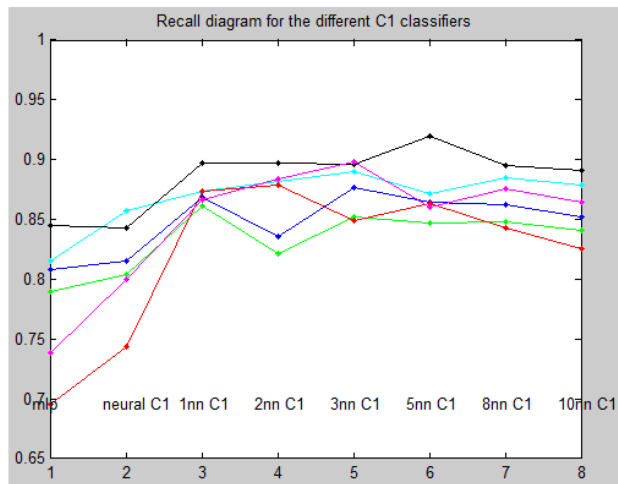
3.2.1 Η ευστοχία για το σύνολο αναγνώρισης γραμμάτων. Αποτυπώνονται στις δηλωμένες με ετικέτες θέσεις οι αντίστοιχοι αλγόριθμοι που χρησιμοποιήθηκαν. Στην πρώτη περίπτωση έχουμε απλό *multilayer perceptron*, ενώ παρακάτω εφαρμόζουμε τον αλγόριθμο που αναφέρουμε στην ενότητα για νευρωνικό και k nn (για διάφορα k) ταξινομητή υπερσυνόλων

Παρατηρούμε από το παραπάνω διάγραμμα ότι όσον αφορά την συνολική

ευστοχία , για το συγκεκριμένο σύνολο, ο αλγόριθμος έχει καλύτερη απόδοση για όλες τις διαφορετικές περιπτώσεις που πήραμε σε σχέση με το απλό *mlp* (χρησιμοποιήθηκε απλο *mlp* μεγέθους [15 12])

Επίσης , απο την παρατηρούμενη γραφική παράσταση είναι αρκετά σαφές οτι για το συγκεκριμένο σύνολο ,οτι η περίπτωση ταξινομητή υπερομάδων *knn* υπερτερεί σε σχέση με την περίπτωση νευρωνικού δικτύου. Αυτό ενδέχεται να οφείλεται στο γεγονός οτι στο συγκεκριμένο σύνολο εκπαίδευσης υπάρχει μεγάλη τοπικότητα μεταξύ των δεδομένων και άρα ο προσδιορισμός ενός προτύπου μπορεί ευκολα να γίνει απο κάποιο πολύ κοντινό του.

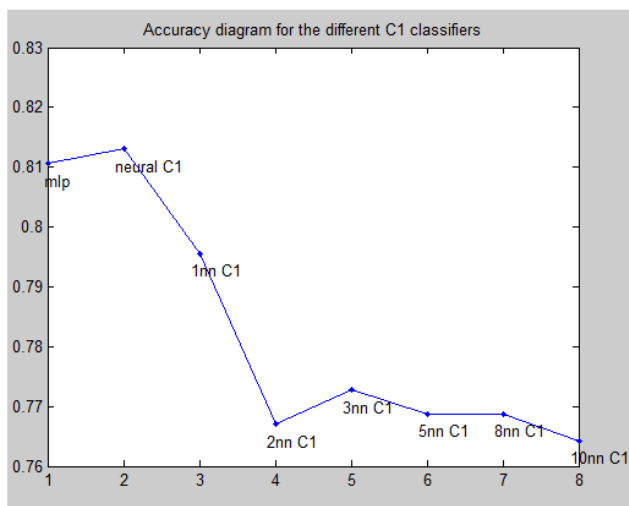
Μάλιστα, φαίνεται εύκολα οτι αναφορικά με την αύξηση της γειτονίας στον *knn* έχουμε σε γενικές γραμμές κάθε φορά κάποια πολύ μικρή μείωση της απόδοσης. Αυτό ενδεχομένως ανάγεται στον παραπάνω ισχυρισμό περι τοπικότητας και με την αύξηση της γειτονίας, βάζοντας περισσότερους πιθανούς γείτονες, μειώνουμε την ισχύη που έχει ο κοντινότερος , ο οποίος είναι και ο πιο σημαντικός. Φυσικά όλα τα παραπάνω αφορούν την υπερομάδα στην οποία θα 'στείλουμε' το πρότυπο μας για να το ταξινομήσει ο αντίστοιχος εξειδικευμένος Ταξινομητής.



3.2.2 Η ανάκληση (*recall*) του αλγορίθμου για τις διάφορες κατηγορίες του συνόλου αναγνώρισης γραμμάτων.

Εδώ αποτυπώνεται η ανάκληση για την κάθε ξεχωριστή κατηγορία απο τις 6 που έχουμε ορίσει για το συγκεκριμένο σύνολο (πιο συγκεκριμένα χρώμα μπλε για τα γράμματα A-E ,πράσινο για τα γράμματα F-J ,κυανό για τα K-O ,κοκκίνο για τα P-S , μύρο για τα T-W και φούξια για τα X-Z)

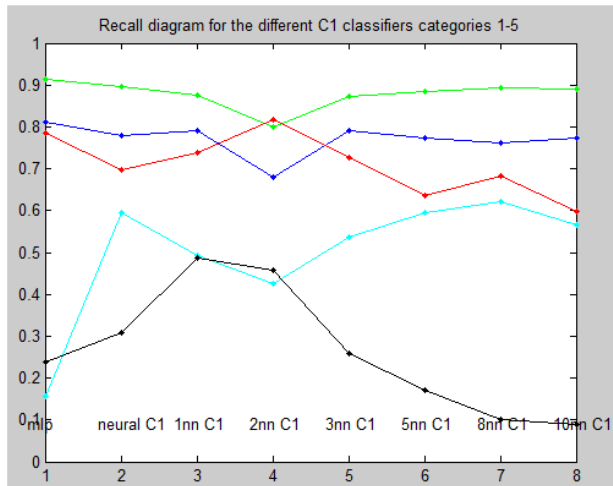
Εχει αρκετό ενδιαφέρον το γεγονός οτι υπάρχει μια γενική συνέπεια στις τιμές των περισσότερων κατηγοριών, (με την κόκκινη για παράδειγμα να είναι σταθερα μια απο τις πιο μη αποδοτικές κατηγορίες και την μαύρη αντίστοιχα να είναι πάντα απο τις πιο αποδοτικές) ενώ επίσης είναι ιδιαίτερα αξιοσημείωτο το οτι στην περίπτωση αλγόριθμου που εφαρμόζουμε σε σχέση με την περίπτωση του απλού *mlp* έχουμε γενικά σύγκλιση (ειδικά στην περίπτωση του *knn*) μεταξύ των αποδόσεων των μεμονωμένων κατηγοριών (με αυτή μαλιστα ,ισως συμπτωματικά να κορυφώνεται μια απο τις βέλτιστες συνολικές αποδόσεις στην περίπτωση του *1nn*). Οι δεδομένες συμπεριφορές είναι σχετικά αναμενόμενες δεδομένου του οτι έχουμε πλέον εξειδικευμένους ταξινομητές και εφόσον κάθε ταξινομητής είναι υπεύθυνος για λιγότερες κατηγορίες το αποτέλεσμα της εκπαίδευσης γενικά αναμένεται πιο σταθερό ανα τις κατηγορίες.

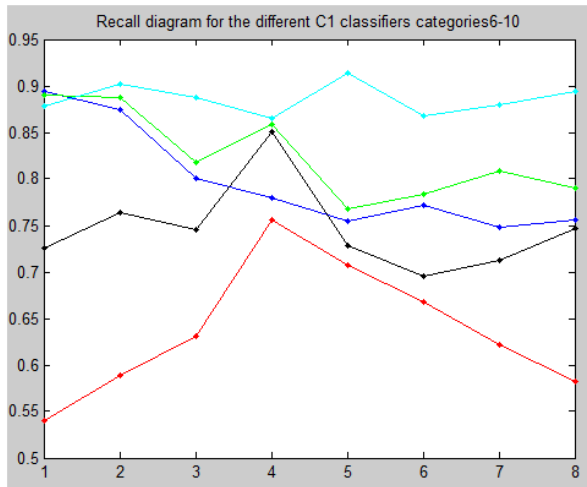


3.2.3 Η ευστοχία για το σύνολο CTG.

3.2.3.2 CTG

Παρατηρούμε ότι σε αυτήν την περίπτωση το σύστημα έχει αρκετά διαφορετική συμπεριφορά. Αρχικά, είναι σημαντικό το γεγονός ότι η περίπτωση που έχουμε ταξινομητή υπερκατηγοριών νευρωνικό δίκτυο είναι σχετικά καλύτερη από την περίπτωση όπου πραγματοποιούμε *knn*. Για την ακρίβεια το απλό *mlp* και η περίπτωση ταξινομητή νευρωνικού ως υπερομάδα έχουν τις καλύτερες αποδόσεις αναφορικά με την ευστοχία του συστήματος. Ο *knn* δεν τα πάει καλά και καθώς μάλιστα αυξάνεται η γειτονία του όσο πάει αποδίδει και γενικά χειρότερα όπως φαίνεται από το διάγραμμα. Από ένα σημείο και μετά πάντως παρατηρούμε κάποια σταθερότητα ανεξάρτητα (ανάμεσα στο 76-78 τοις εκατό) για τις πύο μεγάλες γειτονίες. Γενικά η σχετικά χαμηλή απόδοση σε όλες τις περιπτώσεις του αλγορίθμου ενδέχεται να οφείλεται στο γεγονός ότι οι επιμέρους ταξινομητές μας δεν είναι αρκετά εξειδικευμένοι.





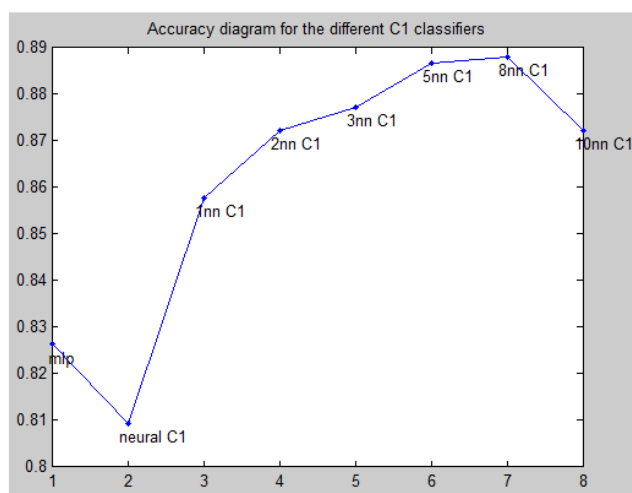
3.2.4 Η ανάκληση(recall) για το σύνολο CTG για όλες τις κατηγορίες. Δεδομένου ότι είχαμε πολλές σχετικά κατηγορίες στο συγκεκριμένο σύνολο εκπαίδευσης, χρησιμοποιήθηκαν 2 εικόνες .

Βλέπουμε απο τα διαγράμματα κατ'άρχάς οτι υπάρχουν κάποιες κατηγορίες οι οποίες δεν έχουν καθόλου ικανοποιητικές τιμές σε σχέση με άλλες κατηγορίες. Παρατηρούμε ειδικά οτι στο πρώτο διαγραμμα η γαλάζια και η μαύρη κατηγορία έχουν πολύ χαμηλή απόδοση για κάποιους ταξινομητές. Επίσης και στα δύο διαγράμματα , υπάρχουν κάποιες κατηγορίες οι οποίες αιτιολογουν την χαμηλή απόδοση των μεταταξινομητών *knn* σε σχέση με τους υπόλοιπους (για παράδειγμα στο πρώτο διάγραμμα η μαύρη και η κόκκινη κατηγορία και στο δεύτερο διάγραμμα , η πράσινη και ίσως η μαύρη). Φυσικά στη συνολική απόδοση οι παραπάνω κατηγορίες δεν αναμένεται να έχουν το ίδιο βάρος και έτσι είναι σχετικά δύσκολο απο τα δύο διαγράμματα να βγάλουμε κάποια ασφαλή συμπεράσματα σχετικά με την αιτία της σχετικά χαμηλής απόδοσης στην περίπτωση του *knn*.

Σχετικά με την υψηλή απόδοση των νευρωνικών , αντίστοιχα , οι κατηγορίες που κατέστησαν σχετικά αδύναμες τις περιπτώσεις του *knn* ισχυροποιούν τα

νευρωνικά. Μια ακόμα παρατήρηση σχετίζεται με την χαμηλή απόδοση του $2nn$ σε σχέση με τους πιο κοντινούς του nn . Αυτό (που κάλλιστα θα μπορούσε να ωφείλεται σε τύχη για τις συγκεκριμένες εκτελέσεις που δώσαν τα αποτελέσματα) προσδιορίζεται παρατηρώντας τη σχετικά χαμηλή απόδοση για το συγκεκριμένο αλγόριθμο κάποιων κατηγοριών του πρώτου διαγράμματος. Από την άλλη όμως παρατηρούμε ότι στο δεύτερο διάγραμμα για τον συγκεκριμένο ταξινομητή έχουμε κάποιες σχετικά υψηλές τιμές για κάποιες κατηγορίες. Υποπευόμαστε λοιπόν ότι οι αντίστοιχες πρώτες κατηγορίες (απο το πρώτο διάγραμμα) έχουν μεγαλύτερο βάρος σε σχέση με αυτές του δευτερου ετσι ώστε να προσδιορίσουν αυτη τη απόδοση.

3.2.3.3 Picture

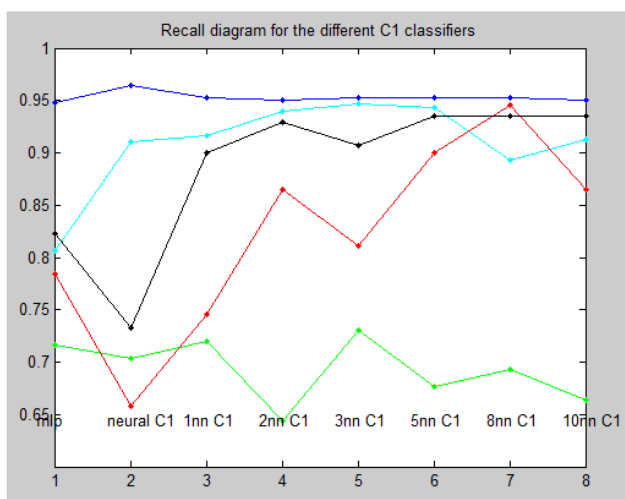


3.2.5 Η ευστοχία για το σύνολο Picture.

Σε αυτό το σύνολο παρατηρούμε ότι η χρήση του αλγορίθμου είχε πολύ καλά

αποτελέσματα σε σχέση με το προηγούμενο, σε σύγκριση με την περίπτωση ενός απλού νευρωνικού δικτύου. Βεβαία αξίζει να αναφερθεί ότι το συγκεκριμένο σύνολο είναι αρκετά μικρό απο άποψη πλήθους προτύπων πράγμα που γενικά προκάλεσε μια αστάθεια στη συνέπεια των αποτελεσμάτων και για τον λόγο αυτό οι προσομοιώσεις εκτελέστηκαν για κάθε είδος πάνω απο 10 φορές (20-30). Σε κάθε περίπτωση πάντως το απλό νευρωνικό δίκτυο υστερούσε ως προς την περίπτωση των *knn* και ήταν ίσο ή υπερτερούσε οριακά ως προς την περίπτωση νευρωνικού για ταξινομητή του *C1*

Απο το διάγραμμα , είναι εμφανές ότι γενικά με την αύξηση της γειτονίας στον *knn* αυξάνεται και η απόδοση του συστήματος μέχρι ένα σημείο. (στους 8-10 γείτονες). Επίσης ,είναι αρκετά εμφανές η σχετική υστέρηση της περίπτωσης του νευρωνικού ως *C1* σε σχέση με το *knn*.



3.2.6 Η ανακλήση για το σύνολο *Picture*.

Στην περίπτωση του διαγράμματος της ανάκλησης(*recall*) για τις διάφορες κατηγορίες βλέπουμε ότι έχουμε την αναμενόμενη συμπεριφορά με βάση την απόδοση που παρατηρήθηκε παραπάνω. Βλέπουμε αισθητή πτώση σε διάφορες κατηγορίες στην περίπτωση του νευρωνικού για *C1* ,ενώ βλέπουμε σχετική

σταθερότητα ή άνοδο για διάφορες κατηγορίες στην περίπτωση του knn (για παράδειγμα η μπλέ κατηγορία μένει σχετικά στάσιμη , η κόκκινη ανεβαίνει , όπως και η μάρνη και η γαλάζια σε γενικές γραμμές. Η πράσινη κατηγορία είναι γενικά ασταθής ,αλλά έχει μικρή διακύμανση για να προκύψει απο αυτή κάποια ουσιώδης διαφοροποίηση).

3.2.4 Επεκτάσεις

Το επόμενο λογικό βήμα επέκτασης του συγκεκριμένου αλγορίθμου έρχεται όπως είναι αναμενόμενο, με τον προσδιορισμό ανωτέρων επιπέδων υπερκατηγοριών οι οποίες περιέχουν υπερκατηγορίες χαμηλότερου επιπέδου προς τον σχηματισμό ενός 'δέντρου' νευρωνικών δικτύων όπου το κάθε επίπεδο θα ερχεται ένα βήμα πιο κοντά στην τελική ταξινόμηση των προτύπων απο ένα σύνολο εξειδικευμενων ταξινομητών.

Για παράδειγμα ας υποθέσουμε οτι έχουμε ένα σύνολο εκπαίδευσης με 15 πιθανές κατηγορίες εξόδου. Τότε εάν θέλουμε να χωρίσουμε τις κατηγορίες σε εξειδικευμένους ταξινομητες (στο $C0$) ο καθένας των οποίων ταξινομεί 2 κατηγορίες, όπως είναι λογικό θα χρειαστούμε 7 τέτοιους ταξινομητές (εκ των οποίων ο ένας θα ταξινομεί μεταξύ τριων κατηγοριών). Έτσι το επίπεδο $C1$ θα έχει 7 κατηγορίες και με βάση τα παραπάνω ο ταξινομητής του $C1$ θα πρέπει να ταξινομήσει το κάθε πρότυπο σε κάποια απο αυτές τις υπερκατηγορίες. Ομως με την αύξηση των κατηγοριών , αυξάνεται και η πιθανότητα να 'μοιάζουν' κάποιες κατηγορίες μεταξύ τους και επιπροσθέτως μεγαλώνει ο υπολογιστικός φόρτος για τον ένα ταξινομητή που έχει αναλάβει το έργο και επομένως είναι πολύ πιθανό οτι θα έχουμε σχετικά χαμηλή απόδοση λόγω του ταξινομητή του $C1$. Για να αποφύγουμε αυτήν την τοπική δυσκολία, θα μπορούσαμε να δεχτούμε περισσότερους του ενός ταξινομητές για το επίπεδο $C1$ Εστω στο παράδειγμα οτι και εδώ επιδιώκουμε τον ελάχιστο δυνατό φόρτο για τον κάθε ταξινομητή και έτσι διαλέγουμε για τον κάθε ταξινομητή να ταξινομεί και εδώ μεταξύ δύο ή το πολύ τριών κατηγοριών. Σε αυτή την περίπτωση θα έχουμε στο $C1$ τρεις ταξινομητές ο καθένας απο τους οποίους θα κατατάσει τα πρότυπα σε κάποιον ταξινομητή του $C0$. Τα σύνολα που ταξινομεί ο κάθε ταξινομητής απο το $C1$ θα είναι και πάλι ξένα μεταξύ τους όπως στην περίπτωση του $C0$. Προκειμένου να αποφασίσουμε σε ποιόν απο τους ταξινομητές του $C1$ θα πάει κάποιο πιθανό πρότυπο ελέγχου, θα χρησιμοποιήσουμε άλλο ένα επίπεδο ,στο παράδειγμα το $C2$ στο οποίο θα έχουμε να διαλέξουμε μεταξύ τριών κατηγοριών και θα εκπαιδεύσουμε έναν ταξινομητή για να ταξινομεί στο επίπεδο αυτο.

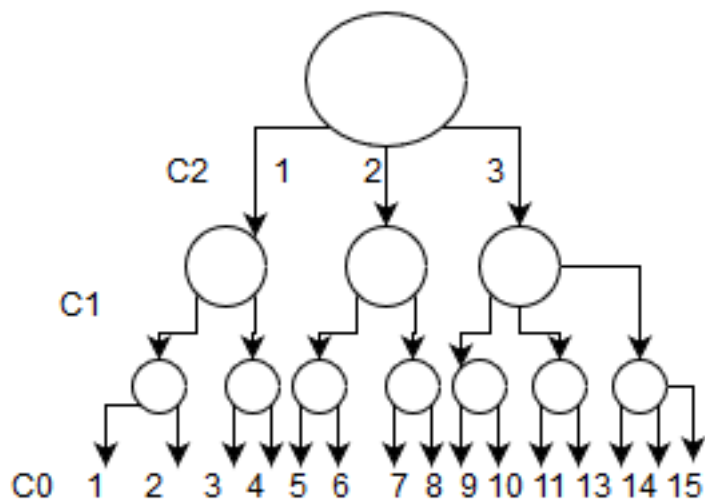
Έτσι, ένα πρότυπο ελέγχου ,οσον αφορά το παράδειγμα, αρχικά θα πάει

στον ταξινομητή του $C2$ ο οποίος θα καθορίσει ποιος ταξινομητής του $C1$ θα το ταξινομήσει, ο οποίος ταξινομητής του $C1$ θα αποφασίσει για το ποιός τελικός ταξινομητής του $C0$ θα αποφανθεί για το ποιά είναι η κατηγορία στην οποία ανήκει.

Φυσικά και εδώ δεν είναι υποχρεωτικό να έχουμε νευρωνικά δίκτυα. Η μέθοδος αυτή θα μπορούσε να εφαρμοσθεί και με τον knn για διαφορετικά σύνολα για την κάθε εφαρμογή του αλγορίθμου. Πιο συγκεκριμένα, αρχικά θα εφαρμοζόταν για όλα τα δεδομένα εκπαίδευσης, τα οποία θα χωρίζονταν σε κατηγορίες τύπου $C2$. Για οποια κατηγορία θα είναι η επικρατέστερη για το κάθε δεδομένο ελέγχου, θα εφαρμόσουμε knn στα δεδομένα εκπαίδευσης που ανήκουν μόνο σε αυτή την κατηγορία, αυτή τη φορά με ετικέτες τύπου $C1$. Τέλος στην επικρατούσα κατηγορία θα επαναλάβουμε την διαδικασία για ετικέτες τύπου $C0$.

Το σύστημα αυτό φυσικά θα μπορούσε να επεκταθεί και για πάνω από 3 επίπεδα. Γενικά θα μπορούσαμε να έχουμε επίπεδο Cn για κάποιο δεδομένο n . Είναι επίσης φανερό δεδομένου ότι έχουμε δενδρική δομή στο σύστημα ότι εάν έχουμε σταθερό πλήθος κατηγοριών εξόδου ανά ταξινομητή για κάθε επίπεδο που ανεβαίνουμε στο δέντρο θα έχουμε εκθετική μείωση του πλήθους των κατηγοριών ανά επίπεδο. (αν έχουμε 27 κατηγορίες στο σύνολο εκπαίδευσης και δεχτούμε σαν τρεις κατηγορίες ανά ταξινομητή θα είναι $27-9-3$ αφού $27 = 3^3$)

Αν και αρκετά εμφανες, είναι ιδιαίτερα σημαντικό να τονισθεί ότι η τελική απόφαση για την κατηγορία κάποιου προτύπου προκύπτει συνολικά κατά την διαδικασία 'κατάβασης' στο δέντρο. Για κάθε επίπεδο που κατεβαίνουμε αποκλείεται ως πιθανή τελική κατηγορία εξόδου ένα σύνολο κατηγοριών. Στο παραπάνω παράδειγμα με τις 15 τελικές κατηγορίες, αν επιλεγθεί στο σε κάποιο πρότυπο, στο $C2$ η δεύτερη κατηγορία, θα αποκλεισθούν απευθείας (εάν τα σύνολα επιλογής είναι ξένα μεταξύ τους) όλες οι τελικές κατηγορίες με εξαίρεση τις 5-8. Βλέπουμε λοιπόν ότι η διαδικασία είναι σταδιακή και επομένως θα μπορούσε να μην πραγματοποιηθεί και ολοκληρωτικά, αλλά ως κάποιος περιορισμός των κατηγοριών.



3.2.7 Το αναφερόμενο παράδειγμα και η δομή του δέντρου.

Σε γενικές γραμμές το σύστημα μοιάζει με δέντρο απόφασης στο οποίο όμως ο κάθε κόμβος αποτελεί έναν ξεχωριστό ταξινομητή. Παρακάτω εξετάζουμε το πώς θα μπορούσε ο αλγόριθμος να γίνει πιο αποδοτικός ανάλογα με την ομοιότητα και την συσχέτιση μεταξύ κάποιων κατηγοριών.

3.2.4.1 Επιλογή των κατηγοριών ανα ταξινομητή

Όπως είναι λογικό, στην περίπτωση όπου εκπαιδεύουμε τον ταξινομητή μας, όσο πιο 'ξεχωριστές' είναι δύο κατηγορίες τόσο πιο εύκολος θα είναι και ο μεταξύ του διαχωρισμός. Για παράδειγμα αν έχουμε μια εικόνα και πρέπει να αποφανθούμε για το που ανηκει το κάθε εικονοστοιχείο (έστω θάλασσα, δάσος, σπίτι, ουρανός) και χρησιμοποιούμε το σύστημα μας, ίσως δεν θα ήταν καλή ιδέα να βάλουμε μαζί στον ίδιο εξειδικευμένο να ταξινομήσει ουρανό και θάλασσα γιατί αυτές οι δύο κατηγορίες ενδέχεται να μοιάζουν στα πρότυπα εκπαίδευσης.

Σε περιπτώσεις όπως τα νευρωνικά δίκτυα όπου ο ταξινομητής μαθαίνει απο παραδείγματα θα μπορούσαμε πριν απο την εκπαίδευση να βρούμε με κάποιο τρόπο κάποια σχέση μεταξύ των κατηγοριών και να φτιάξουμε κάποια μετάθεση έτσι ώστε ο κάθε εξειδικευμένος ταξινομητής να μαθαίνει τις όσο γίνεται πιο ανόμοιες μεταξύ τους κλάσεις.

Το πιο απλό που θα μπορούσαμε να κάνουμε προς αυτή την κατεύθυνση είναι το ακόλουθο: Εστω ότι μας δίνεται σύνολο εκπαίδευσης $T, [\mathbf{x}(i), y(i)]_{i=1}^N$.

Εδώ , μετα την προεπεξεργασία των δεδομένων θα μπορούσαμε να βρούμε τον μέσο όρο για όλα τα πρότυπα ανα κατηγορία, ετσι ώστε εαν έχουμε K πιθανές κατηγορίες (οι οποίες εδω αποτυπώνονται μέσω του y) να είναι:

$$\bar{\mathbf{x}}_{category_k} = \frac{\sum_{i:x(i)=y(i)=k} \mathbf{x}(i)}{N_k} \quad (83)$$

με το N_k να συμβολίζει το πλήθος των προτύπων που ανήκουν στη δεδομένη κατηγορία. Κατόπιν θα ορίσουμε με βάση κάποιο μέγεθος (όπως πχ η ευκλείδεια απόσταση) την διαφορά μεταξύ όλων των πιθανών κλάσσων. Ετσι για δύο κλάσσες k_1, k_2 μας ενδιαφέρει η τιμή:

$$d_{k_1 k_2} = \|\bar{\mathbf{x}}_{k_1} - \bar{\mathbf{x}}_{k_2}\|_2 \quad (84)$$

η οποία κατα κάποιον τρόπο συμβολίζει αυτην απόσταση μεταξύ δύο κατηγοριών.

Σε επόμενο επίπεδο θα κατασκευάσουμε έναν πίνακα $k \times k$ ο οποίος θα περιέχει αυτές τις αποστάσεις για όλες τις κατηγορίες και θα προσπαθήσουμε να βρούμε για την κάθε κατηγορία , την αντίστοιχη κατηγορία απο την οποία αυτή να απέχει την μέγιστη απόσταση. Αν αυτή την κατηγορία την έχει ήδη επιλέξει μια άλλη κατηγορία ως μέγιστη της , τότε επιλέγουμε την επόμενη μέγιστη κλπ. Πραγματοποιούμε αυτή την διαδικασία για όλες τις γραμμές του πίνακα μας (και άρα για όλες τις κατηγορίες) και κατόπιν βάζουμε τα ζευγάρια που έχουν διαμορφωθεί στους ίδιου ταξινομητές .Αν θέλουμε ο κάθε εξειδικευμένος ταξινομητής μας να έχει πάνω απο 2 κατηγορίες μπορούμε στον πίνακα μας να διαλέγουμε με αντίστοιχο τρόπο τις n μεγαλύτερες αποστάσεις.

Ασφαλώς , ο τρόπος αυτός έχει πολλά μειονεκτήματα. Κατ αρχας δεν του έχουμε δώσει την ικανότητα να διακρίνει ποια κατηγορία έχει μεγαλύτερη ανάγκη μια άλλη κατηγορία. Για παράδειγμα ενδέχεται σε μια κατηγορία να έχουμε πολλές κατηγορίες οι οποίες είναι πολύ κοντά της (με βάση τους αντιπροσώπους που έχουμε ορίσει) ,αλλα μία να είναι μακρια και να μην μπορεί αυτη να επιλεγεί επειδή έχει ήδη επιλεγεί απο άλλη κατηγορία. Για να αντιμετωπίσουμε το συγκεκριμένο ζήτημα θα μπορούσαμε να προσδιορήσουμε κάποιου ήδους προτεραιότητα επιλογής στις πιο ευαίσθητες κατηγορίες(μέσω πχ κάποιου είδους ταξινόμησης).

Επιπροσθετα, δεν λαμβάνει υπόψιν την πιθανή διακύμανση των δεδομένων. Ο μέσος όρος δεν είναι ισχυρό στατιστικό μέτρο και επομενωσ ο κάθε αντιπροσωπος των κατηγοριών (που προέρχεται απο τον μέσο όρο) ενδέχεται στην πραγματικότητα να μην είναι αντιπροσωπευτικός της κατηγορίας.

Επίσης στην παραπάνω προσέγγιση δεν λαμβάνονται υπόψιν οι πιθανές εξαρτήσεις - συσχετίσεις μεταξύ των κατηγοριών. Προκειμένου να μπορέσουμε να

έχουμε μια καλή ταξινόμηση ,θα ήταν ιδιαίτερα επιθυμητό να μπορέσουμε να αποσυσχετίσουμε τις διάφορες κατηγορίες μεταξύ τους. Αυτό θα μπορούσαμε να το επιτύχουμε εφαρμόζοντας ίσως bagging κάθε φορά σε κάποιο υποσύνολο (που προσδιορίζεται απο την κατηγορία στην οποία ανήκουν τα δεδομένα) του συνόλου εκπαίδευσης (πχ bagging στα δεδομένα εκπαίδευσης που ανήκουν στις κατηγορίες 3-5 ενός συνόλου 15 κατηγοριών) και κάθε φορά ελέγχοντας την ικανότητα διαχωρισμού των κατηγοριών χρησιμοποιώντας έναν εξειδικευμένο ταξινομητή. Απο εκεί θα πάρουμε μια ομάδα με τις καλύτερες αποδόσεις (ομάδα έτσι ώστε να καλύπτουμε πιθανές απώλειες δεδομένων που έχουν προκύψει απο την διαδικασία του bootstrap) και θα επαναλάβουμε για όλες τις κατηγορίες του συνόλου εκπαίδευσης. Το αποτέλεσμα αναμένεται να είναι αρκούντως αποσυσχετισμένο όμως τοπικά, επμένως λογικά η διαδικασία θα πρέπει να επαναληφθεί και στα παραπάνω επίπεδα (C_1, \dots, C_n) του δέντρου για να έχουμε ένα πιο συνολικό αποτέλεσμα.

3.3. Μελέτη συμπεριφοράς συστήματος Νευρωνικών Δικτύων που βασίζεται στο boosting

3.3.1 Γενικά.

Σε αυτό το τμήμα μελετάμε ένα σύστημα το οποίο βασίζεται στη λογική του *boosting*, χωρίς να αποτελεί ακριβώς τη συγκεκριμένη τεχνική. Πιο συγκεκριμένα το σύστημα αυτής της ενότητας βασίζεται στη λογική της σύστασης μίας ομάδας ταξινομητών, η εκπαίδευση των οποίων είναι σχεδιασμένη έτσι ώστε ο καθένας τους να ισχυροποιεί την ομάδα και να ισχυροποιείται σε πρότυπα στα οποία, τα προηγούμενα μέλη της ομάδας είναι σχετικά ανίσχυρα.

Η τεχνική που χρησιμοποιούμε αποτελεί μια από τις απλούστερες τεχνικές αυτού του τύπου που θα μπορούσε να χρησιμοποιηθεί. Αρχικά εκπαιδεύουμε έναν ταξινομητή (εδώ νευρωνικό δίκτυο) σε όλα τα πρότυπα εκπαίδευσης του συνόλου εκπαίδευσης. Από τον συγκεκριμένο ταξινομητή δεν ζητάμε ιδιαίτερα καλή απόδοση στο σύνολο εκπαίδευσης (ή το αντίστοιχο σύνολο επικύρωσης) δεδομένου ότι ο αλγόριθμος θα χρησιμοποιήσει και άλλα δίκτυα και επομένως δεν χρειάζεται να επιβαρύνουμε κάποιο περισσότερο από όσο χρειάζεται.

Κατόπιν χρησιμοποιώντας ως κριτήριο ποια από τα δεδομένα εκπαίδευσης έχει ταξινομήσει σωστά ο συγκεκριμένος ταξινομητής, χωρίζουμε το σύνολο εκπαίδευσης σε δύο υποσύνολα του: Το σύνολο των προτύπων το οποίο αυτός ταξινομήσει σωστά και αυτό το οποίο απέτυχε να ταξινομήσει με επιτυχία. Αυτό θα αποτελέσει την είσοδο του επόμενου ταξινομητή. Το επόμενο δίκτυο επομένως θα εκπαιδευθεί στα πρότυπα στα οποία απέτυχε ο προηγούμενος ταξινομητής.

Η διαδικασία θα επαναληφθεί μέχρις ότου να υπάρξει κάποια συνθήκη τερματισμού της, όπως για παράδειγμα η απαίτηση για ορθή ταξινόμηση από το σύστημα συνολικά τουλάχιστον ενός ποσοστού του συνόλου εκπαίδευσης. Εάν δεν υπάρξει, θεωρητικά θα μπορούσε να σταματήσει όταν ταξινομηθούν σωστά όλα τα πρότυπα εκπαίδευσης ορίζοντας φυσικά την απαραίτητη συνθήκη.

Στη συνέχεια, κατά τη φάση του ελέγχου, προσδιορίζουμε την έξοδο με δύο τρόπους. Αρχικά σαν ένα σταθμισμένο άθροισμα της ομάδας των νευρωνικών, με μεγαλύτερο βάρος στους ταξινομητές οι οποίοι είναι υπεύθυνοι (έχουν ταξινομήσει επιτυχώς) για περισσότερα πρότυπα και με μικρότερο βάρος σε αυτούς οι οποίοι είναι υπεύθυνοι για λιγότερα. Επίσης, με τον δεύτερο τρόπο εφαρμόζουμε αλγόριθμο *knn* προκειμένου να προσδιορίσουμε το ή τα δίκτυα στα οποία ανήκει με βάση την διάταξη του στον χώρο των προτύπων το εν λόγω

πρότυπο ελέγχου. Τη συγκεκριμένη διαδικασία την πραγματοποιούμε με δύο τεχνικές. Είτε βρίσκουμε τους κοντινότερους γείτονες και ψηφίζουμε για το πού ανήκει, δηλαδή ποιο από τα νευρωνικά της ομάδας θα είναι υπευθyno για να το ταξινομήσει, το δεδομένο πρότυπο ελέγχου ανάλογα με την πλειοψηφία των γειτόνων, είτε εφαρμόζουμε την τεχνική αναλογικά, δίνοντας βάρος στο δίκτυο στο οποίο ανήκει κάποιο πρότυπο αναλογα (ή για την ακρίβεια αντίστροφως ανάλογα από την απόσταση) με το πόσο κοντά βρίσκεται το πρότυπο αυτό στο υπο εξέταση πρότυπο ελέγχου, λαμβάνοντας υπόψιν τα k κοντινότερα πρότυπα και κατόπιν κανονικοποιώντας και πέρνοντας το άθροισμα των βαρών αυτών στην προσομοίωση από το κάθε δίκτυο

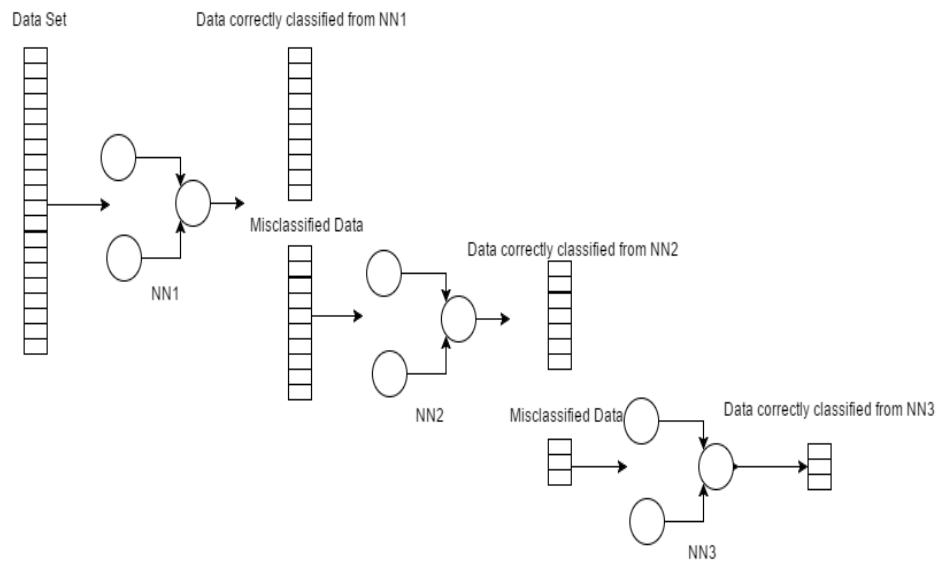
Το σύστημα αυτό αν και είναι αρκετά απλό σαν σύλληψη, έχει το πλεονέκτημα ότι αν και έχει το θετικό του *boosting*, δηλαδή σχηματίζουμε μια σχετικά ισχυρή ομάδα ταξινομητών στην οποία ο ένας καλύπτει την αδυναμία των προηγούμενων, είναι αρκετά γρήγορο στην διαδικασία της εκπαίδευσης δεδομένου ότι κάθε φορά οι ταξινομητές εκπαιδεύονται με λιγότερα πρότυπα. Εδώ φυσικά ελοχεύουν διαφοροί κίνδυνοι, τους οποίους θα εξετάσουμε παρακάτω.

Αρχικά πρέπει να τονισθεί ότι από άποψη μεμονωμένου νευρωνικού, είναι αρκετά επίφοβο το γεγονός ότι στα πιο προχωρημένα επίπεδα του συστήματος τα πρότυπα με τα οποία θα εκπαιδευθεί κάποιο νευρωνικό θα είναι πάρα πολύ λίγα (για την ακρίβεια ένα σχετικά μικρό ποσοστό του συνόλου εκπαίδευσης). Αυτό μπορεί να επιδράσει ιδιαίτερα αρνητικά στο συγκεκριμένο δίκτυο και κατ'έπextαση στο σύστημα γιατί το δίκτυο εφόσον θα μάθει μόνο τα δεδομένα αυτά είναι αρκετα πιθανό ότι θα φτάσει σε μια κατάσταση αντίστοιχη με *overfitting*.

Προκειμένου να αποφύγουμε αυτή τη δυσκολία θα μπορούσαμε να απαιτήσουμε σε σχετικά μεγάλο ποσοστό επί του συνόλου των προτύπων εκπαίδευσης τον τερματισμό της διαδικασίας έτσι ώστε ακόμα και το τελευταίο νευρωνικό να έχει ένα αρκετά μεγάλο ποσοστό για να μπορέσει αν είναι δυνατό να γενικεύσει σωστά. Φυσικά παρατηρούμε ότι η συγκεκριμένη διαδικασία είναι ιδιαίτερα ευαίσθητη σε σφάλμα και κατα κάποιο τρόπο απλοική δεδομένου ότι ανάλογα με κάθε διαφορετικό σύνολο θα χρειασθεί επαναπροσδιορισμός του ποσοστού τερματισμού.

Άλλο ένα ζήτημα του συστήματος αφορά τα μεγάλα σύνολα, όπου όπως είναι αναμενόμενο από τον τρόπο σχεδιασμού του αναμένεται να είναι (και όντως είναι όπως προκύπτει από το σύνολο των 16000 προτύπων) αρκετα αργό αφού λαμβάνει αρχικά από ένα μόνο νευρωνικό δίκτυο το σύνολο των προτύπων εκπαίδευσης -αν και είναι γεγονός ότι δεδομένου ότι έχουμε μειωμένες απαιτήσεις απόδοσης από το αρχικό δίκτυο (για παράδειγμα όταν ταξινομήσει σωστά το 60 τοις εκατό των προτύπων) η διαδικασία προχωράει αρκετα πιο γρηγορα από την απλή περίπτωση του ενός νευρωνικού γιατί τα επόμενα δίκτυα έχουν ένα ποσοστό μόνο της εκπαίδευσης του αρχικού-. Ειδικά μάλιστα στις

περιπτώσεις όπου αποφασίζουμε για τα πρότυπα ελέγχου με τον αλγόριθμο knn έχουμε αρκετά μεγάλη καθυστέρηση στο σύστημα λόγω της σχετικά μεγάλης υπολογιστικής πολυπλοκότητας του αλγορίθμου αυτού. (για όλα τα πρότυπα ελέγχου πρέπει να βρούμε τις αποστάσεις τους από τα πρότυπα εκπαίδευσης και κατόπιν τους k γείτονες άρα χοντρικά αναμένεται $O(size(TrainingDataSet) * size(TestDataSet))$)



3.3.1 Στην εικόνα απεικονίζεται η διαδικασία που ακολουθείται στον συγκεκριμένο αλγόριθμο. Αρχικά έχουμε ένα σύνολο και κατόπιν φιλτράρεται από το νευρωνικό NN1. Στη συνέχεια, τα πρότυπα εκπαίδευσης τα οποία δεν μπόρεσε να ταξινομήσει ορθά το NN1 πάνε στο NN2 και η διαδικασία επαναλαμβάνεται στο NN3 τό οποίο ταξινομεί ορθά τα πρότυπα του και επομένως η διαδικασία ολοκληρώνεται.

Τέλος είναι αρκετά εμφανές ότι σε κάθε περίπτωση η ομάδα που σχηματίζουμε δεν είναι ισοκατανομημένη. Όπως είναι λογικό, οι πρώτοι ταξινομητές του συστήματος είναι πιο σημαντικοί για την υπο σύσταση ομάδα από τους τελευταίους δεδομένου ότι έχουν υπό την ευθύνη τους περισσότερα πρότυπα. Είναι προφανές ότι με τις προαναφερθείσες μεθόδους απόφασης για το νευρωνικό στο οποίο θα ταξινομηθεί το κάθε πρότυπο ελέγχου αυτό αποτυπώνεται.

Με την περίπτωση του απλού βεβαρημένου και κανονικοποιημένου αθροίς-

ματος είναι δεδομένο αφού ο κάθε συντελεστής για το κάθε δίκτυο θα προκύπτει από το ποσοστό των προτύπων που έχει ταξινομησει σωστά (και άρα έχει υπεύθυνη του). Έτσι άμεσα τα δίκτυα που έχουν λίγα πρότυπα πέρνουν και μικρό βάρος στο σύνολο ενώ αυτά που έχουν πολλά , πέρνουν μεγάλο.

Με την περίπτωση του knn κάθε πρότυπο ελέγχου θα προσδιορίζεται ουσιαστικά από τους γείτονές του στον χώρο των προτυπων εκπαίδευσης. Δεδομένου όμως ότι , όπως προαναφέρθηκε τα πιο αρχικά δίκτυα του συστήματος θα είναι υπεύθυνα για περισσότερα πρότυπα εκπαίδευσης, είναι πιθανότερο για ένα πρότυπο ελέγχου προσδιορισθεί τελικά (ή να έχει μεγάλο συντελεστή) από ένα δίκτυο που έχει πολλά πρότυπα -και άρα κάποιο από τα πρώτα-αφού είναι πιθανότερο να προκύψει γείτονας από κάποιο τέτοιο δίκτυο σε σχέση με το να προκύψει γείτονας που ανήκει σε κάποιο δίκτυο με λιγότερα πρότυπα. Αυτή η μεγαλύτερη πιθανότητα προσδιορίζει επομένως υψηλότερους συντελεστες στον τελικό προσδιορισμό για τα δίκτυα με σχετικά υψηλό αριθμό προτύπων εκπαίδευσης.

Παρακάτω θα εξετάσουμε την υλοποίηση του αλγορίθμου στο *MATLAB* και στη συνέχεια θα παρουσιάσουμε τα αποτελέσματα από τις διάφορες εκτελέσεις του αλγορίθμου.

3.3.2 Υλοποίηση.

Τα αρχικά βήματα που πραγματοποιήθηκαν για την υλοποίηση του συγκεκριμένου αλγορίθμου είναι παρόμοια με αυτά των υπολοίπων . Αρχικά κάνουμε την ανάγνωση των δεδομένων μετά εκτελούμε τυχαία μετάθεση εάν χρειάζεται και δημιουργούμε τους πίνακες εκπαίδευσης και ελέγχου και τέλος πραγματοποιούμε την προεπεξεργασία των δεδομένων (με στόχο την μείωση της διάστασης της εισόδου). Για να επιτύχουμε τις συγκεκριμένες διαδικασίες χρησιμοποιούμε τις γνωστές συναρτήσεις που χρησιμοποιούνται και στους προηγούμενους αλγορίθμους.

Στον ίδιο τον αλγόριθμο, σε πρώτο επίπεδο αρχικοποιούμε το διάνυσμα των συντελεστών(cv) , τον δείκτη(q) και τον αριθμό που δηλώνει το ποσοστό των σωστά ταξινομημένων προτύπων από το σύστημα συνολικά στο οποίο θέλουμε να σταματήσει η διαδικασία(ou).

Στη συνέχεια αρχίζουμε την επαναληπτική διαδικασία κατά την οποία μέχρι να ικανοποιηθεί η συνθήκη του παραπάνω ποσοστού, δημιουργούμε νέα δίκτυα καθένα από τα οποία εκπαίδευεται για να μάθει κάποιο ποσοστό του εκάστοτε συνόλου εκπαίδευσης. Και εδώ για συνθήκη τερματισμού της εκπαίδευσης έχουμε την πολύ απλή συνθήκη απόδοσης (ευστοχίας). Όταν ταξινομηθεί σωστά ένα ποσοστό των δεδομένων εκπαίδευσης η εκπαίδευση του συγκεκριμέ-

νου δικτύου ολοκληρώνεται. Και εδώ όπως και στον προηγούμενο αλγόριθμο έχουμε αυτό το πολύ απλό κριτήριο προκειμένου να εστιάσουμε στη διαδικασία που ακολουθείται από το σύστημα συνολικά.

Επειτα, χωρίζουμε το σύνολο εκπαίδευσης στα δύο υποσύνολα ορθής και λανθασμένης ταξινόμησης από το συγκεκριμένο ταξινομητή (της συγκεκριμένης επανάληψης) χρησιμοποιώντας την συνάρτηση *newroundTrainData* στην οποία βρίσκουμε στην πιο πρόσφατη προσομοίωση στον έλεγχο κριτηρίου, ποια από τα πρότυπα ελέγχου έχει ταξινομήσει σωστά και ποια λάθος ο ταξινομητής χρησιμοποιώντας το σύνολο των ορθών ετικετών.

Μετά από αυτό προσδιορίζουμε τους συντελεστές του συγκεκριμένου ταξινομητή με την ακόλουθη λογική: ο νέος συντελεστής προσδιορίζεται ως το πλήθος των ορθά ταξινομημένων προς το πλήθος του παρόντος συνόλου εκπαίδευσης επί το ποσοστό του συνόλου αυτού στο αρχικό σύνολο εκπαίδευσης. Ο προσδιορισμός γίνεται χρησιμοποιώντας μια αναδρομική σχέση.

Αφού προσδιορίσουμε τους συντελεστές, ελευθερώνουμε τους πίνακες του τωρινού συνόλου εκπαίδευσης, αποθηκεύουμε το ορθά ταξινομημένο σύνολο σε μία δομή *cell* (όπου δείχνει για τον κάθε ταξινομητή τα ορθά ταξινομημένα από αυτόν πρότυπα εκπαίδευσης) και μεταφέρουμε τα λάθος ταξινομημένα πρότυπα στο σύνολο εκπαίδευσης της επόμενης επανάληψης. Τέλος ενημερώνουμε τον δείκτη του παρόντος ταξινομητή (q) και αναθέτουμε στο νούμερο του ποσοστού για την ολοκλήρωση της εκπαίδευσης τον πιο πρόσφατο συντελεστή. Μετα δημιουργούμε άλλο ένα νευρωνικό δίκτυο προκειμένου να ταξινομήσουμε, τι απέμεινε από την διαδικασία. Αξίζει να σημειωθεί ότι από το συγκεκριμένο δίκτυο μπορεί να απαιτηθεί σχετικά μεγαλύτερη απόδοση, επειδή είναι το τελευταίο δίκτυο. Επίσης, ο συντελεστής του συγκεκριμένου δικτύου προσδιορίζεται από την κανονικοποίηση στο 1 ($1 -$ αθροισμα των υπολοίπων συντελεστών)

Αφότου τελειώσουμε με την διαδικασία εκπαίδευσης χρησιμοποιούμε τους δύο τρόπους που αναφέραμε παραπάνω για να ταξινομήσουμε τελικά το σύνολο ελέγχου. (χρησιμοποιούμε σχόλια για να εφαρμόσουμε τον ένα ή τον άλλο τρόπο).

Πρώτα χρησιμοποιούμε την μέθοδο του *knn*. Εδώ πρέπει να σημειωθεί ότι στο μεγάλο σύνολο -το *letters*- έγινε προσπάθεια να επιταχύνουμε κατά το δυνατό την διαδικασία δεδομένου ότι καθυστέρουσε αρκετά. Για τον λόγο αυτό αλλάξαμε λίγο την μέθοδο, αντί για εύρεση γειτόνων και κατόπιν προσδιορισμός κατηγορίας, ενώσαμε τις δύο διαδικασίες (προσδιορίζουμε την κατηγορία κατά την εύρεση του γείτονα) και για να αποφύγουμε την προσπέλαση δομής, πριν ξεκινήσουμε τον *knn* μεταφέραμε την δομή σε πίνακα τριών διαστάσεων (τον *TrBig*) αν και αυτό δεν είμαστε βέβαιοι ότι είχε επίδραση στην συνολική απόδοση. Τέλος, πριν από τον *knn* προσδιορίσαμε οποιοδήποτε πλήθος θα μας χρειαζόταν σε κάποια επανάληψη έτσι ώστε αυτός ο υπολογισμός να μην

γίνεται κατά τον αλγόριθμο και επιβαρύνει. Πραγματι μετά από την εφαρμογή των παραπάνω ο χρόνος εκτέλεσης μειώθηκε αρκετά, αλλά παραμένει αρκετά μεγάλος.

Μετά από την εφαρμογή του *knn* και δεδομένου ότι έχουμε βρεί τους γείτονες και τους αντίστοιχους ταξινομητές όπου αυτοί ανήκουν για όλα τα δεδομένα ελέγχου, εφαρμόζουμε με δύο τρόπους τα αποτελέσματα μας όπως προαναφέραμε. Με τον πρώτο τρόπο, για όλα τα δεδομένα προσδιορίζουμε πλειοψηφικά τον ένα ταξινομητή ο οποίος θα είναι υπεύθυνος για την ταξινόμηση ενός δεδομένου προτύπου ελέγχου (χρησιμοποιούμε την συνάρτηση *nnz* για να μετρήσουμε πόσοι από τους γείτονες ανήκουν στο κάθε ταξινομητή και στην συνέχεια βρισκουμε τον ταξινομητή με τις μεγιστες ψήφους για το κάθε πρότυπο ελέγχου). Με τον δεύτερο τρόπο για τον κάθε γείτονα ορίζουμε έναν συντελεστή ο οποίος προσδιορίζεται ως το αντίστροφο της απόστασης του συγκεκριμένου γείτονα από το πρότυπο ελέγχου προς το άθροισμα των αντίστροφων όλων των γειτόνων (για την κανονικοποίηση) και για κάθε πρότυπο ελέγχου πέρνουμε το βεβαρημενο άθροισμα με αυτούς τους συντελεστές των αντίστοιχων ταξινομητών (δηλαδή των προσομοιώσεων των ταξινομητών για το κάθε πρότυπο ελέγχου) όπου ανήκει ο κάθε γείτονας.

Τέλος στην πιο απλή περίπτωση όπου χρησιμοποιούμε απλά το άθροισμα όλων των ταξινομητών με βάρη τους κανονικοποιημένους συντελεστες που προσδιορήσθηκαν παραπάνω δεν κάνουμε κάτι περισσότερο από το να πάρουμε για όλα τα δεδομένα το άθροισμα αυτό (άρα κοινοι συντελεστές για όλα τα δεδομένα).

3.3.3 Αποτελέσματα.

Και σε αυτήν την περίπτωση, για την καλύτερη παρουσίαση των αποτελεσμάτων χρησιμοποιήθηκαν γραφήματα. Όπως και πριν, για την παρουσίαση της συνολικής απόδοσης στο σύνολο των δεδομένων επιλέχθηκε σαν μέτρο η ευστοχία, ενώ για την παρουσίαση της απόδοσης ανά κατηγορία η ανάκληση (*recall*). Επιπλέον μετρούνται για κάθε εκτέλεση η ακρίβεια και ο χρόνος, αλλά θα παρουσιαστούν σε μορφή πίνακα. Γενικότερα για τις εκτελέσεις ισχύει ο,τι ίσχυε στην προηγούμενη περίπτωση. Το μόνο που πρέπει να σημειωθεί είναι ότι η περίπτωση εκτέλεσης απλού *mlp* πολλών σταδίων δεν περιλαμβάνεται σε αυτό το σύνολο δεδομένου ότι δεν αναμένεται να αλλάξει κάτι (και τα χαρακτηριστικά αναμένονται περίπου τα ίδια). Μόνο στη περίπτωση του μικρότερου συνόλου το συμπεριλαμβάνουμε γιατί το σύνολο έχει σχετική αστάθεια στα αποτελέσματα του, για λόγους συγκρισης.

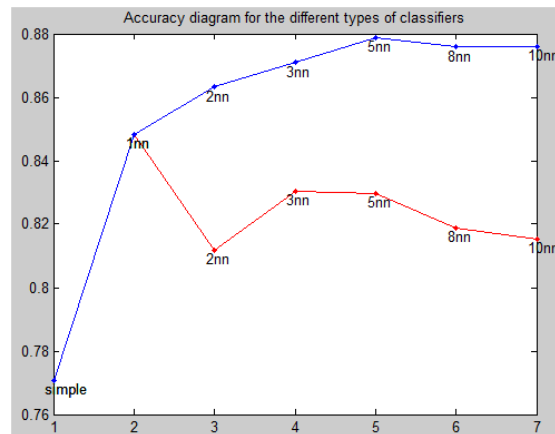
Πριν την παρουσίαση για το κάθε σύνολο παρουσιάζονται σε πίνακα οι

χρόνοι εκτελέσεως του συγκεκριμένου αλγορίθμου με βάση την τυπική για αυτόν παραμετροποίηση εκπαίδευσης που έχει ακολουθηθεί:

| Type/neighbors: | simple factors | 1nn | 2nn | 3nn | 5nn | 8nn | 10nn |
|-----------------|----------------|-------|-------|-------|-------|-------|-------|
| Letters | 150-250 | 350 | 400 | 400 | 420 | 470 | 490 |
| CTG | 28-40 | 30-40 | 35-45 | 35-50 | 40-50 | 40-50 | 50-55 |
| Picture | 3-6 | 4-6 | 5-6 | 5-7 | 6-8 | 6-9 | 6-10 |
| Abalone | - | 39 | - | 40 | 40 | 42 | 49 |
| Contraceptive | - | - | - | 14 | 15 | 16 | 18 |

3.3.3.1 Letters

Παρακάτω παρουσιάζεται για το σύνολο *Letters* η ευστοχία για τις διάφορες εκτελέσεις που πραγματοποιήθηκαν (αναφέρεται πιο αναλυτικά η κάθε μια κάτω από την γραφική παράσταση):

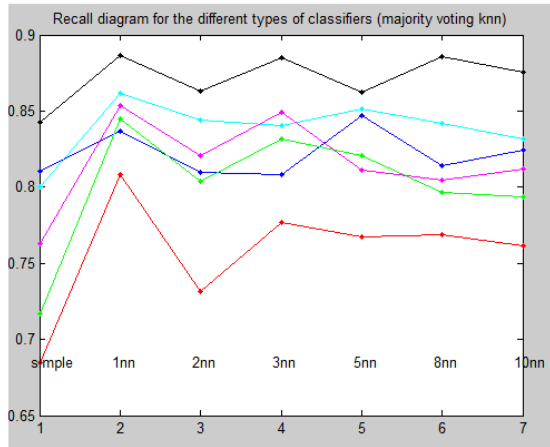


3.3.2 Η ευστοχία του συστήματος για το *letters*. Ως *simple* έχουμε ονομάσει την περίπτωση όπου έχουμε απλούς συντελεστές με βάση το ποσοστό του συνόλου που ταξινομεί ο κάθε ταξινομητής-ίδιοι για όλα τα πρότυπα ελέγχου.

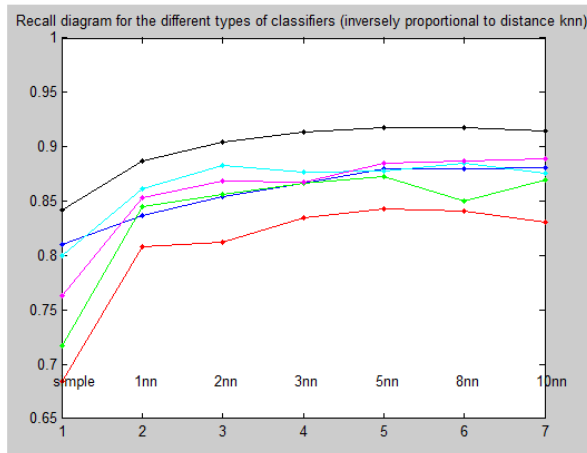
Με knn , $k = 1, 2, 3, 5, 8, 10$ ονομάζουμε τον αντίστοιχο knn που εφαρμόζουμε για να προσδιορίσουμε που ανήκει το κάθε πρότυπο ελέγχου ξεχωριστά. Με κόκκινο συμβολίζει η πλειοψηφική και με μπλέ (στα αποτελέσματα για knn) η αναλογική εφαρμογή του αλγορίθμου

Απο το παραπάνω διάγραμμα είναι αρκετά ξεκάθαρο ότι για το συγκεκριμένο σύνολο η αναλογική εφαρμογή του αλγορίθμου knn κατά την φάση διαμοιρασμού των προτύπων ελέγχου στα διάφορα νευρωνικά δίκτυα που έχουν εκπαιδευτεί υπερτερεί από άποψη ευστοχίας σε σχέση με την πλειοψηφική εφαρμογή του. Γενικά αυτό είναι σχετικά αναμενόμενο λόγω της πιο κατανεμημένης φύσης της αναλογικής εφαρμογής, άλλωστε ας μην ξεχνάμε ότι στην πλειοψηφική εφαρμογή το κάθε πρότυπο θα ταξινομηθεί από ένα μόνο δίκτυο. Η απόδοση φαίνεται να κορυφώνεται για την αναλογική εφαρμογή μετά τους τρεις γείτονες, ενώ στην πλειοψηφική περίπτωση ακόμη και το ενδεχόμενο ενός γείτονα υπερτερεί σε σχέση με τις υπόλοιπες περιπτώσεις.

Κατι ακόμα το οποίο είναι αρκετά εμφανές για το συγκεκριμένο σύνολο είναι ότι γενικότερα ο knn και η ξεχωριστή ανάθεση στα διάφορα δίκτυα υπερτερεί αρκετά σε σχέση με την πιο απλή περίπτωση όπου έχουμε σταθερούς συντελεστές για το κάθε δίκτυο ανάλογα με την σημασία του στο σύνολο εκπαίδευσης. Αυτό εξηγείται από το γεγονός ότι στη περίπτωση του knn πέρνουμε ξεχωριστές περιπτώσεις για το κάθε πρότυπο ελέγχου ξεχωριστά και επομένως η επίλυση είναι πολύ πιο σχετική με τα πρότυπα. Αυτό όμως όπως φαίνεται άλλωστε και από τους πίνακες συνεπάγεται πολύ μεγαλύτερο χρόνο εκτέλεσης. Έχουμε επομένως ανταλλαγή απόδοσης -χρόνου. Κατά τα άλλα ο αλγόριθμος φαίνεται να έχει τις χειρότερες προσπάθειες του για μικρές (2) ή σχετικά μεγάλες (8,10) γειτονίες στον πλειοψηφικό knn .



3.3.3 Το μέγεθος recall για τις διάφορες κατηγορίες του συνόλου letters για την περίπτωση πλειοψηφικής εφαρμογής του knn, καθώς και για την περίπτωση χρήσης απλών συντελεστών.



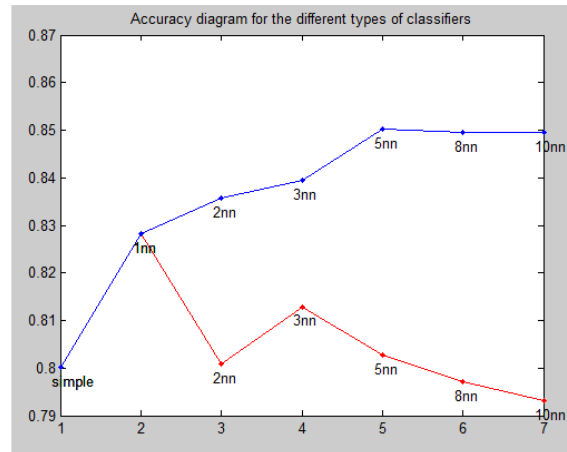
3.3.4 Το μέγεθος recall για τις διάφορες κατηγορίες του συνόλου letters για

την περίπτωση αναλογικήςεφαρμογής του knn , καθώς και για την περίπτωση χρήσης απλών συντελεστών για λόγους πληρότητας.

Γενικά στα παραπάνω γραφήματα βλέπουμε ότι η απόδοση (με βάση το κριτήριο της ανάκλησης) για την κάθε κατηγορία αντιστοιχίζεται με αρκετή ακρίβεια στα αποτελέσματα της συνολικής απόδοσης για το συγκεκριμένο σύνολο εκπαίδευσης. Πράγματι , αν παρατηρήσουμε τόσο για την πλειοψηφική όσο και για την αναλογική περίπτωση την απόδοση ανα την κατηγορία , είναι πολύ αναμενόμενο να προκύψουν τα δεδομένα αποτελέσματα που φαίνονται παραπάνω για την συνολική απόδοση. Βλέπουμε ότι η κάθε κατηγορία είναι σχετικά σταθερή σε σχέση με την απόδοση της ως προς τις άλλες κατηγορίες και η άνοδος ή πτώση των ποσοστων επιτυχίας είναι ανα τις διαφορετικές εκτελέσεις αντίστοιχη για την κάθε μια. (Παρατηρείται για παράδειγμα πως στην πλειοψηφική περίπτωση απο την εκτέλεση για απλούς συντελεστές έχουμε μια σχετικά ομοιόμορφη για όλες τις κατηγορίες απότομη αύξηση για την περίπτωση $1nn$)

Φυσικά η κάθε κατηγορία -ειδικά για την πλειοψηφική περιπτωση- έχει τις διακυμάνσεις τις πχ $3nn$ με $5nn$, βλέπουμε όμως ότι οι διακυμάνσεις αυτές έχουν πολύ μικρό εύρος για να επηρεάσουν την συνολική αναμενόμενη συμπεριφορά που παρουσιάζεται στο διάγραμμα ευστοχίας.

3.3.3.2 CTG

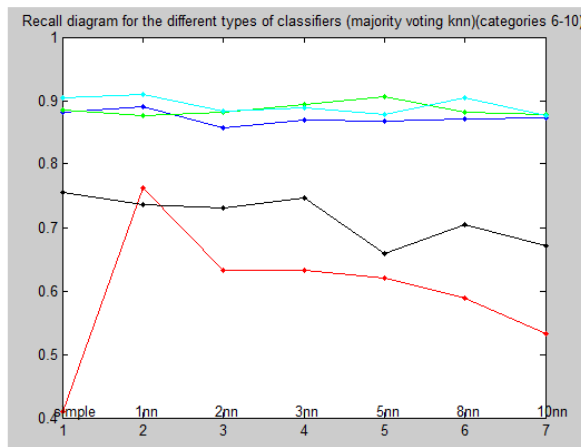
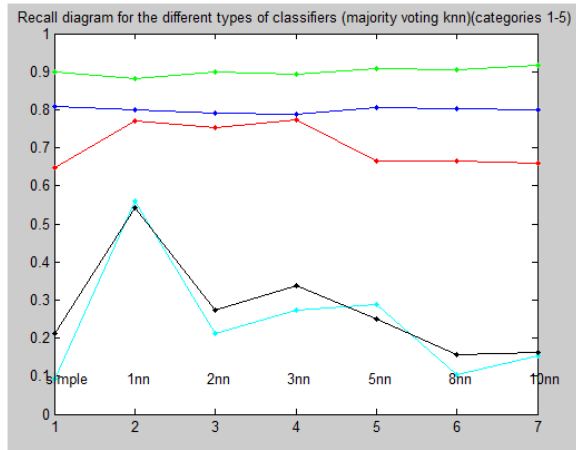


3.3.5 Ομοίως με παραπάνω, ευστοχία για το σύνολο CTG

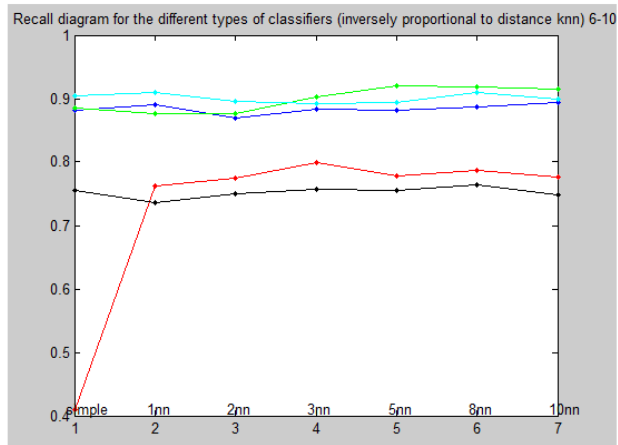
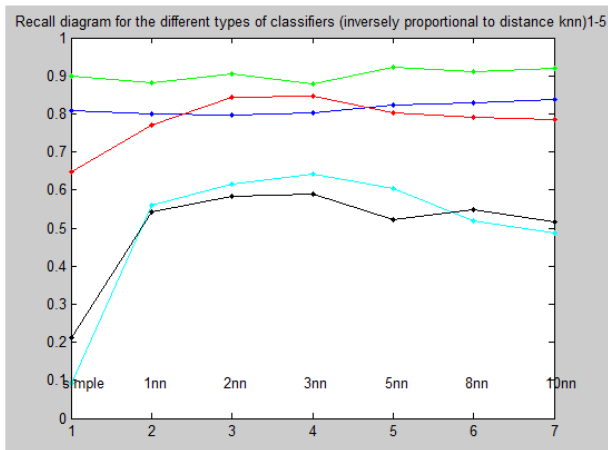
Για το συγκεκριμένο σύνολο παρατηρούμε αρχικά μια χαρακτηριστικά έντονη πτώση της απόδοσης της πλειοψηφικής περίπτωσης (κόκκινη γραφική) με την αύξηση της γειτονίας του αλγοριθμού knn , ενώ αντίθετα στην αναλογική εφαρμογή έχουμε μέχρι ένα σημείο (5nn) αύξηση της απόδοσης. Και πάλι αυτό μπορεί να αιτιολογείται με σχετικά ακατάλληλες επιλογές δικτύων για την ταξινόμηση των διαφόρων προτύπων στην πλειοψηφική περίπτωση, πράγμα το οποίο φυσικά δεν ισχύει για την αναλογική δεδομένου ότι κατα πάσα πιθανότητα πάνω από ένα δίκτυο θα συμμετάσχει στην ταξινόμηση κάποιου προτύπου. Μάλιστα, η πολύ μεγάλη πτώση που παρατηρείται στις πιο μεγάλες γειτονικές περιοχές (μικρότερο ακόμα και από την περίπτωση απλών συντελεστών) είναι πιθανό ότι είναι απόρροια του ότι σχετικά μακρινά πρότυπα αποφασίζουν για το δίκτυο που ανήκει κάποιο πρότυπο με το ίδιο βάρος ψήφου με πιο κοντινά πρότυπα εκπαίδευσης και άρα είναι πιθανότερο να προκύψει κάποιο σχετικά πιο άσχετο

δίκτυο.

Η παραπάνω συμπεριφορά έχει σαν αποτέλεσμα με την αύξηση της γειτονίας να αυξάνεται κατα πολύ και το χάσμα των δύο περιπτώσεων εκτέλεσης



3.3.6 Ομοίως με παραπάνω, recall πλειοψηφικού knn για το σύνολο CTG

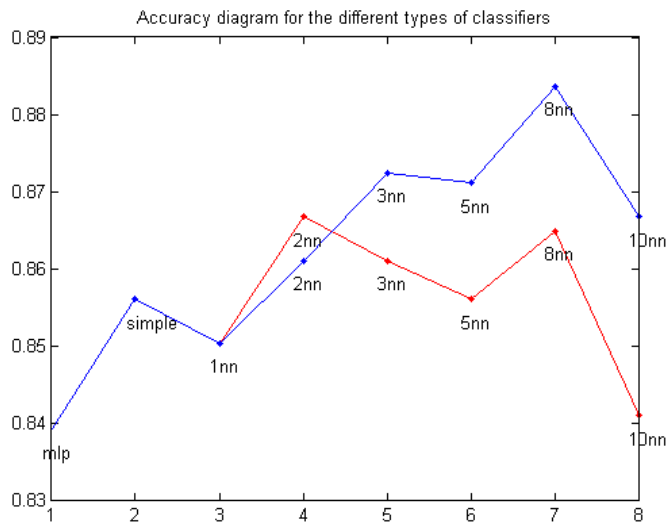


3.3.7 Ομοίως με παραπάνω, recall αναλογικό knn για το σύνολο CTG

Γενικά απο τις παραπάνω γραφικές της απόδοσης ανα κατηγορία παρατηρούμε οτι και πάλι η περίπτωση των απλών συντελεστών υστερεί αφού σε κάποιες κατηγορίες έχει πολύ χαμηλότερη απόδοση σε σχέση με τον knn και επομένως αναμένεται η χαμηλότερη συνολική απόδοση.

Στην αναλογική περίπτωση , για τις διάφορες υποπεριπτώσεις του knn βλέπουμε οτι έχουμε γενικά μια πιο σταθερή συμπεριφορά ανα τις κατηγορίες , καθώς σε άλλες αυξάνεται και σε άλλες μειώνεται η απόδοση , αλλα , γενικά έχουμε μια σχετικά μικρή διακύμανση.

3.3.3.3 Picture

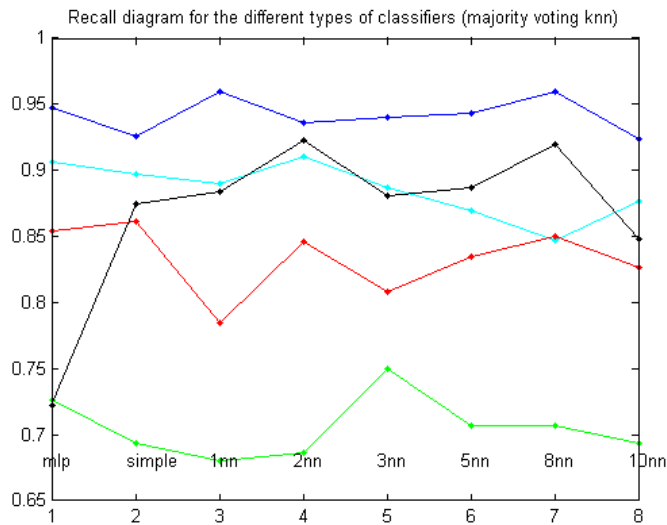


3.3.8 Ομοίως με παραπάνω,ευστοχία για το σύνολο *Picture*. Συμπεριλαμβάνεται και η περίπτωση του απλού *mlp* για λόγους πληρότητας

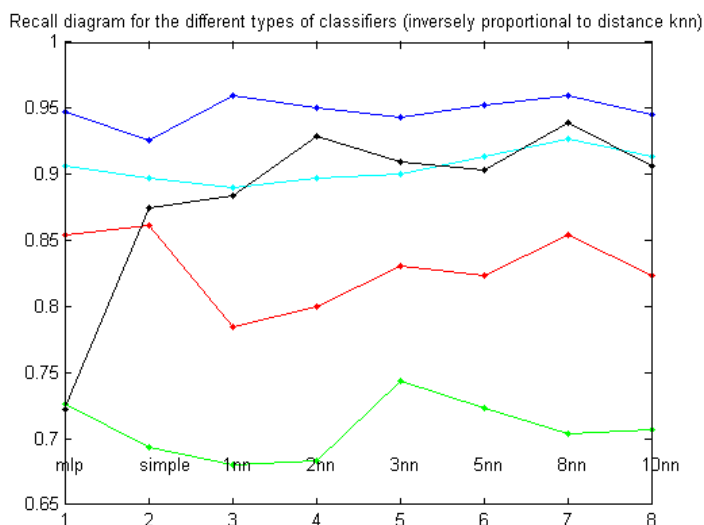
Το σύνολο αυτό γενικά χαρακτηρίζεται απο αστάθεια στις μετρήσεις σε σχέση με τα υπόλοιπα σύνολα (γενικά παρατηρούμε μεγαλύτερο εύρος για

κάποιο πιθανό αποτέλεσμα ίσως λόγω του ότι είναι αρκετά μικρό). Ανεξάρτητα από αυτό παρατηρούμε ότι και σε αυτήν την περίπτωση, αναφορικά με την ευστοχία του συστήματος έχουμε καλύτερη συμπεριφορά στην περίπτωση της εφαρμογής του knn σε σχέση με την περίπτωση των προ-προσδιορισμένων συντελεστών, τουλάχιστον για τιμές γειτονίας μεγαλύτερες του 1- εξαίρεση αποτελεί μόνο η πλειοψηφική $10nn$.

Παρατηρούμε επίσης ότι για ακόμη μία φορά η πλειοψηφική μέθοδος μειονεκτεί της αναλογικής με μοναδική εξαίρεση την περίπτωση του $2nn$. Επίσης βλέπουμε ότι σχεδόν όλες οι περιπτώσεις εφαρμογής του συστήματος υπερτερούν από άποψη απόδοσης σε σχέση με το απλό mlp ακόμα και σε αυτό το αρκετά μικρό σύνολο.



3.3.9 Ομοίως με παραπάνω, για recall πλειοψηφικού



3.3.10 Ομοίως με παραπάνω, για recall αναλογικού

Εδώ αρχικά αξίζει να παρατηρηθεί ότι υπάρχει αρκετή ποικιλία στην συμπεριφορά των διαφόρων κατηγοριών για τις διαφορετικές εκτελέσεις. Πιο συγκεκριμένα παρατηρούμε ότι ορισμένες εκτελέσεις (ανεξάρτητα από την περίπτωση που είμαστε - πλειοψηφική, αναλογική) έχουν μεγάλη άνοδο για την περίπτωση του συστήματος σε σχέση με το απλό νευρωνικό δίκτυο, ενώ άλλες έχουν κάποια πτώση. Από εκεί και έπειτα, για τις διάφορες υποπεριπτώσεις του συστήματος φαίνεται ότι η συμπεριφορά είναι γενικά πιο σταθερή, με εξαίρεση την αλλαγή για κάποιες κατηγορίες κατά την μετάβαση από την χρήση συντελεστών στην χρήση του *knn*.

Κάτι ιδιαίτερα χαρακτηριστικό είναι ότι τα γραφήματα των 2 υποπεριπτώσεων του *knn* μοιάζουν αρκετά μεταξύ τους. Συγκεκριμένα οι διάφορες κατηγορίες έχουν συμπεριφορά η οποία σε γενικές γραμμές είναι παρόμοια. Εξαίρεση αποτελεί η σταδιακή πτώση της γαλάζιας κατηγορίας στην πλειοψηφική περίπτωση για τις πιο μεγάλες γειτονίες, η οποία είναι αντιθέτως σταθερή για την αναλογική περίπτωση σε αυτές τις εκτελέσεις. Αυτό πιθανώς εξηγείται, όπως αναφέρθηκε και παραπάνω λόγω του κακού συνδυασμού μεγάλης γειτονιάς- μεμονωμένης απόφασης.

3.3.4 Επεκτάσεις / Τροποποιήσεις.

3.3.4.1 Συνυπολογισμός εμπιστοσύνης στον προσδιορισμό αποστασης του knn

Μια αρκετά ενδιαφερόμενη τροποποίηση που θα μπορούσαμε να κάνουμε είναι να προσπαθήσουμε με κάποιο τρόπο για κάθε πρότυπο ελέγχου να βάλουμε κάποιο παράγοντα για το κατά πόσο εμπιστευόμαστε ότι οι γείτονες του είναι αρκετά αξιόπιστοι προκειμένου να τους αφήσουμε να επηρεάσουν κατά τον παράγοντα απόστασης τους στην τελική ταξινόμηση του προτύπου αυτού.

Για να κάνουμε κάτι τέτοιο όπως είναι λογικό πρέπει να προσδιορίσουμε κάποιον σχετικά ισχυρό παράγοντα αξιοπιστίας για το κάθε πρότυπο εκπαίδευσης. Η αξιοπιστία αυτή θα πρέπει να σχετίζεται με τον ταξινομητή ο οποίος είναι υπεύθυνος για τον υπο εξέταση γείτονα και με το κατά πόσο μπορεί όντως με επιτυχία να ταξινομήσει τον γείτονα αυτο του προτύπου ελέγχου.

Για να γίνει πιο κατανοητό αυτό που επιχειρούμε να κάνουμε, μπορούμε να σκεφτούμε το εξής: Εστω ότι έχουμε κατά την φάση της ταξινόμησης ένα πρότυπο ελέγχου και όλοι οι γείτονές του από τα πρότυπα εκπαίδευσης προσπαθούν να το φέρουν στην ομάδα τους, δηλαδή στον ταξινομητή τους, με επιχειρήμα τον αριθμό τους, ή την απόσταση τους στον χώρο εισόδου. Το πρότυπο ελέγχου όμως πριν αποφανθεί για το ποιόν-ή ποιούς- ταξινομητή θα διαλέξει από τις προτάσεις των γειτόνων του από την εκπαίδευση προσπαθεί για τον κάθε γείτονα να προσδιορίσει κατά πόσο τα αποτελέσματα τους από τον ταξινομητή που είναι αντίστοιχα υπεύθυνος, είναι ορθά (γιατί έχουμε τις ετικέτες εξόδου από το κάθε πρότυπο εκπαίδευσης). Ξέρουμε βέβαια ότι ο αντίστοιχος ταξινομητής ταξινομηθεί σωστά το πρότυπο εκπαίδευσης του, αλλά όχι κατά πόσο.

Εδώ λοιπόν θα χρησιμοποιήσουμε παλινδρόμηση. Θα βρούμε για το καθέ ένα από τους γείτονες πόσο απέχει το αποτέλεσμα του από το σωστό αποτέλεσμα της ταξινόμησης και κατόπιν θα κανονικοποιήσουμε τα αποτελέσματα για το κάθε γειτονικό πρότυπο. Εστω για κάποιο πρότυπο ελέγχου \mathbf{x}_i ότι έχουμε μια γειτονία N με k γείτονες έτσι ώστε $N = \{\mathbf{x}_{n1}, \mathbf{x}_{n2}, \dots, \mathbf{x}_{nk}\}$, με αντίστοιχο σύνολο ετικετών(κλάσσεων εξόδου) $N_{out} = \{\mathbf{y}_{n1}, \mathbf{y}_{n2}, \dots, \mathbf{y}_{nk}\}$. Προκειμένου να υπολογίσουμε την αξιοπιστία του κάθε γείτονα θα έχουμε:

$$tr_i = \|\mathbf{y}_{ni} - \mathbf{y}_{nouti}\|_2, \quad i = 1 \dots k \quad (85)$$

όπου το $\|\cdot\|_2$ συμβολίζει την ευκλείδεια νόρμα και το \mathbf{y}_{nouti} την αντίστοιχη πραγματική εξόδο του νευρωνικού δικτύου που εκπροσωπεί τον γείτονα i (προ-

φανώς με είσοδο τον συγκεκριμένο γείτονα). Στη συνέχεια κανονικοποιούμε:

$$a_{2i} = \frac{tr_i}{\sum_{i=1}^k tr_i}, \quad i = 1 \dots k \quad (86)$$

και θεωρώντας a_{1i} το κλασσικό κριτήριο απόφασης που έχουμε απο πριν (κανονικοποιημένο αντίστροφο απόστασης) θα μπορούσαμε να ορίσουμε τον τελικό συντελεστή a ως:

$$a_i = c_1 a_{1i} + c_2 a_{2i}, \quad i = 1 \dots k \quad (87)$$

απαιτώντας

$$c_1 + c_2 = 1 \quad (88)$$

ετσι ώστε να διατηρείται η κανονικοποίηση. Πράγματι παρατηρούμε οτι:

$$\sum_{i=1}^k a_i = 1 \quad (89)$$

Αυτή η διαδικασία θα επαναληφθεί για όλα τα πρότυπα ελέγχου και καθένας απο τους γείτονες θα έχει τον αντίστοιχο συντελεστή. Αξίζει να σημειωθεί οτι εάν έχουμε το πλειοψηφικό σύστημα απόφασης η επιρροή της παραπάνω αξιοπιστίας θα πρέπει να επιδράση στην ψηφοφορία για την επιλογή του πιο κατάλληλου νευρωνικού δικτύου και δεν θα επηρεάσει κατανεμημένα την τελική απόφαση.

Επίσης , το πιο λογικό είναι , το βάρος της αξιοπιστίας c_2 να είναι σχετικά μικρότερο απο το βάρος του πόσο κοντά είναι το πρότυπο εκπαίδευσης c_1 . Θα πρέπει κάποιος πολύ κοντινός γείτονας (με μεγάλο a_1) να έχει πολύ χαμηλή αξιοπιστία (σχετικά χαμηλο a_2 και άρα να έχει οριακά ταξινομηθεί σωστά απο το δίκτυο που προέκυψε υπεύθυνο για αυτόν) για να αποκτήσει πιο μεγάλη σημασία κάποιος πιο μακρινος γείτονας (ο οποίος έχει πιο υψηλη αξιοπιστία).

Τέλος ,είναι σημαντικό το οτι η αναφερθείσα μέθοδος δεν έχει δοκιμασθεί και αν και αναμένεται να υπάρχουν περιπτώσεις όπου θα είναι ωφέλιμη, δεν είμαστε σε θέση να ξέρουμε πως θα συμπεριφερθεί στην υλοποίηση.

3.3.4.2 Πρόσθεση προτύπων σε μικρα σύνολα εκπαίδευσης

Οπως αναφέραμε παραπάνω, σε προχωρημένα στάδια της διαδικασίας που ακολουθεί ,ο αλγόριθμος αυτής της ενότητας οδηγεί σε πολύ μικρά συνολα

εκπαίδευσης τα οποία όπως είναι λογικό υστερούν σε ποιότητα εκπαίδευσης για τα δίκτυα με τα οποία σχετίζονται. Για να αποφύγουμε αυτό το πρόβλημα και δεδομένου ότι θέλουμε κατά το δυνατό να διατηρήσουμε το πλεονέκτημα της ταχύτητας που παρέχει αυτή μέθοδος μπορούμε να ορίσουμε κάποιο σταθερό μέγεθος από το οποίο κάποιο σύνολο δεν θα πρέπει να γίνει μικρότερο.

Το μέγεθος αυτό θα θέλαμε να είναι είναι αρκετό ώστε να παρέχει ικανοποιητική εκπαίδευση, αλλά να μην καθυστερεί ιδιαίτερα την διαδικασία. Κατι τέτοιο είναι δύσκολο να προσδιορισθεί, αλλά είναι αρκετά αναμενόμενο να προσδιορισθεί ως προς ποσοστό του αρχικού συνόλου. Το πιο απλό αλλά και χρονοβόρο που θα μπορούσε να γίνει είναι να εξετάσουμε τις αποδόσεις του συνόλου σε συνδυασμό με την ταχύτητα του για διάφορα μεγέθη και να αποφανθούμε για το κατάλληλο στο συγκεκριμένο σύνολο.

Σε αυτή την προσέγγιση το επόμενο ζήτημα είναι πως θα 'γεμίσουμε' τις νέες θέσεις του συνόλου (για το κάθε νέο σύνολο που θα προκύπτει). Αυτό θα μπορούσαμε να το πραγματοποιήσουμε με διάφορους τρόπους.

Αρχικά, με βάση τον πιο κλασικό, θα μπορούσαμε να βάλουμε στις νέες θέσεις επαναλήψεις προτύπων του ίδιου του συνόλου έτσι ώστε να δημιουργήσουμε το νέο μεγαλύτερο σύνολο. Οι επαναλήψεις αυτές θα μπορούσαν να μούν με τυχαίο τρόπο, ή με βάση κάποιας λογικής, όπως για παράδειγμα να μούν σε περισσότερες επαναλήψεις τα πρότυπα τα οποία στο προηγούμενο σύνολο απέτυχαν χειρότερα. Το ποιό απέτυχαν χειρότερα θα μπορούσαμε να το βρούμε χρησιμοποιώντας την παραπάνω σχέση (85) - απόστασης της πραγματικής κλάσης από το αποτέλεσμα που βγάζει το προηγούμενο νευρωνικό για κάθε πρότυπο εκπαίδευσης του. Η σχέση αυτή αν και δεν είναι απόλυτα ακριβής, όμως αποτελεί έναν γενικό δείκτη. Απόσα πρότυπα έχουν αποτύχει, τα οποία αποτελούν και το καινούριο σύνολο, θα βρούμε αυτά με την χειρότερη απόδοση και θα τους δώσουμε μεγαλύτερο βάρος για το νέο σύνολο, με αποτέλεσμα να έχουν μεγαλύτερη πιθανότητα να μούν σε περισσότερες θέσεις.

Μια άλλη παρόμοια προσέγγιση είναι να συμπληρώσουμε με κάποια από τα επιτυχημένα του προηγούμενου γύρου και πάλι είτε τυχαία είτε με κάποιο αντίστοιχο κριτήριο για να βρούμε τα λιγότερο επιτυχημένα. Φυσικά, είναι εμφανές ότι με αυτήν την προσέγγιση θα είναι πιο δύσκολο και αρα λιγότερο πιθανό σε κάθε νέο 'γύρο' να ταξινομήσουμε σωστά τα αποτυχημένα πρότυπα (δεδομένου ότι τώρα θα εμπλέκονται και επιτυχημένα πρότυπα) και επομένως λογικά θα απαιτηθούν περισσότεροι γύροι για να φτάσουμε στο επιθυμητό ποσοστό ορθά ταξινομημένων προτύπων από το σύστημα.

3.4. Μελέτη συστήματος ξεχωριστής εκπαίδευσης επι μέρους συστάδων

3.4.1 Γενικά.

Σε αυτό το τμήμα εξετάζουμε τη περίπτωση συστήματος όπου χρησιμοποιούμε μια ομάδα ταξινομητών οι οποίοι εκπαιδεύονται σε διαφορετικά πρότυπα του συνόλου , με βάση κάποια ομαδοποίηση η οποία έχει προηγηθεί. Πιο συγκεκριμένα ,εκπαιδεύουμε κάθε ταξινομητή με πρότυπα εκπαίδευσης τα οποία ανήκουν σε ξεχωριστούς *clusters* οι οποίοι έχουν προκύψει από προηγούμενη φάση ομαδοποίησης. Για κάθε διαφορετικό *cluster* αντιστοιχούμε και κάποιον διαφορετικό ταξινομητή και τον εκπαιδεύουμε με τα αντίστοιχα πρότυπα εκπαίδευσης που ανήκουν σε αυτόν τον.

Αρχικά είναι σημαντικό να παρατηρήσουμε ότι αυτό το σύστημα , δεδομένου ότι ουσιαστικά τμηματοποιεί το σύνολο εκπαίδευσης, απευθύνεται κυρίως σε πιο μεγάλα σύνολα , στα οποία έχει νοήμα να τα τμηματοποιήσουμε για λόγους συντομευσης της διαδικασίας της εκπαίδευσης ή ακόμα και βελτίωσης της απόδοσης (σε ένα μεγάλο σύνολο είναι πιθανότερο να υπάρχουν περιοχές οι οποίες λόγω για παράδειγμα υψηλής συσχέτισης να δυσχεραίνουν συνολικά την εκπαιδευτική διαδικασία. Τέτοιες περιοχές είναι καλύτερο για το σύστημα να αντιμετωπίζονται μεμονωμένα). Σε μικρότερα σύνολα αντίθετα ίσως είναι καλύτερο να προσεγγίζουμε πιο συνολικά την ταξινόμηση , χρησιμοποιώντας μεμονωμένους ταξινομητές ή ομαδικές μεθόδους οι οποίες απευθύνονται καθολικά στο σύνολο.

Πολύ σημαντικό βήμα του αλγορίθμου είναι η διαδικασία σχηματισμού των συστάδων (*clusters*). Αν οι συστάδες δεν έχουν χωρισθεί ισοκατανεμημένα, ενδέχεται για κάποιους ταξινομητές να έχουμε πολύ λιγα πρότυπα ή πάρα πολλά για άλλους. Πρέπει επομένως να χρησιμοποιηθεί κάποιος σχετικά αξιόπιστος αλγόριθμος ομαδοποίησης για αυτήν την φάση. Ακόμα όμως και αν η ομαδοποίηση είναι σχετικά καλή , ενδλεχεται να υάρξουν συστάδες των οποίων η εκπαίδευση να αποτελεί χρονοβόρα διαδικασία εξαιτίας διαφόρων λόγων.

Αφότου παρέλθει και το βήμα της εκπαίδευσης και σε αυτό στο σύστημα όπως και σε προηγούμενο , έχουμε επιλογή για το πώς θα παρθεί η απόφαση για τα πρότυπα ελέγχου. Η πιο απλή λύση είναι κάθε πρότυπο ελέγχου να ταξινομείται ανάλογα με την μέση απόφαση όλων των ταξινομητών για όλες τις συστάδες. Κάτι τέτοιο όμως φάνηκε μετά απο κάποιες δοκιμαστικές εκτελέσεις σε κάποια σύνολα ότι δεν οδηγεί σε υψηλή απόδοση , πράγμα το οποίο είναι

λογικό, αφού διαφορετικές συστάδες μπορεί να λάβουν αρκετα ετεροκλητες αποφάσεις για κάποιο πρότυπο ελέγχου. Αλλωστε είναι κάπως παράλογο να αποφασίζει για κάποιο πρότυπο ελέγχου που ανήκει σε μια περιοχή A κάποιος ταξινομητής που ανήκει σε μια περιοχή B η οποία δεν έχει καμία σχέση με την A, γιατί αυτός θα έχει εκπαιδευτεί με πρότυπα τα οποία έχουν εντελώς διαφορετικές τιμές χαρακτηριστικών απο αυτές της A και διαφορετικό τρόπο ταξινόμησης.

Αντάυτου, για το συγκεκριμένο σύστημα το οποίο ουσιαστικά χαρακτηρίζεται απο χωρικές διαφοροποιήσεις είναι σημαντικό να αποφασίσουμε χωρικά για το ποιός ταξινομητής θα είναι αρμόδιος για την ταξινόμηση κάποιου προτύπου. Για μια ακόμη φορά μια σχετικά άμεση και απλή λύση θα ήταν ο *knn*, όμως το μειονέκτημα είναι ότι έδω αν εφαρμόσουμε τη συγκεκριμένη μέθοδο στα πρότυπα εκπαίδευσης, ζητώντας γείτονες για τα πρότυπα ελέγχου κατα κάποιο τρόπο στερούμε το πλεονέκτημα της αναγωγής του συνόλου εκπαίδευσης σε συστάδες -οι οποίες έχουν κάποια κέντρα αναφοράς- απο άποψη χρόνου.

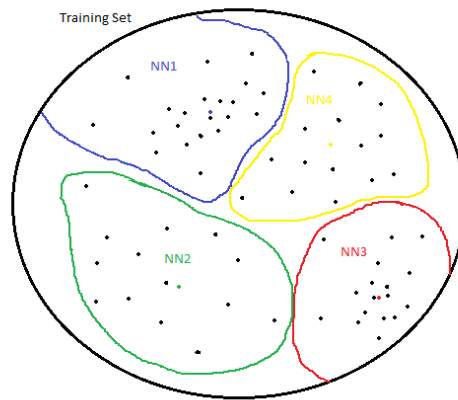
Φυσικά κάτι τέτοιο θα ήταν ιδιαίτερα ασύμφορο χρονικά. Ετσι, θα μπορούσαμε αντί να ψάχνουμε στον *knn* για γείτονες στο σύνολο εκπαίδευσης, να κάνουμε αυτήν την αναζήτηση στο σύνολο των συστάδων (εφόσον έχουν προσδιορισθεί τα προαναφερθέντα κέντρα). Και αυτή η σκέψη όμως, μετα απο πειραματική επαλήθευση φαίνεται ότι δεν είναι ιδιαίτερα καλή απο άποψη απόδοσης, ίσως λόγω του ότι εμπεριέχονται και κάποιοι γείτονες οι οποίοι έχουν σχετικά μεγάλη απόσταση απο το πρότυπο.

Μετα απο κάποιους πειραματισμούς, εφαρμόσαμε δύο σχετικά απλές μεθόδους. Στην πιο απλή περίπτωση την ταξινόμηση του κάθε προτύπου την αποφασίζει η κοντινότερη σε αυτόν συστάδα. Αυτό είναι σχετικά αναμενόμενο να τα πάει καλύτερα απο τις προηγούμενες μεθόδους και πάλι λόγω της ισχυρότερης χωρικής σχέσης συστάδας -προτύπου.

Ακόμη, εφαρμόζουμε μια προσέγγιση η οποία φαίνεται ότι τα πάει αρκετα καλά. Ουσιαστικά παραλλάσσουμε κάπως τον *knn* και βαθμονομούμε για τον κάθε *cluster* το πόσο κόντα του είναι κάποιο πρότυπο ελέγχου. Αν είναι πολυ κοντά (για παράδειγμα σε απόσταση μικρότερη απο τη μέση ακτίνα της συστάδας-δηλαδή της απόστασης που προκύπτει απο την μέση τιμή των αποστασεων των προτύπων της συστάδας-) τότε βαθμολογούμε την συγκεκριμένη συστάδα με μεγάλη βαθμολογία. Αν είναι 'μέσα στον ευρύτερο *cluster*' δηλαδή σε απόσταση μικρότερη της μέγιστης απόστασης απο το κέντρο του αυτή βαθμολογείται αλλα με μικρή βαθμολογία. Αν είναι έξω απο την συστάδα τότε αυτή δεν βαθμολογείται καθόλου. Κατόπιν κανονικοποιούμε και εξομοιώνουμε με τους συντελεστές που προκύπτουν για την κάθε συστάδα.

Η μέθοδος αυτή φαίνεται να έχει αρκετά καλή συμπεριφορά τουλάχιστον για τα σύνολα τα οποία χρησιμοποιήσαμε. Αυτό μπορεί να οφείλεται στο ότι δεν

χρησιμοποιήσαμε την απόσταση σαν ένα απόλυτο μέτρο 'σημασίας' της κάθε συστάδας, αλλά αξιοποιήσαμε το κατα πόσο κάποιο πρότυπο ελέγχου είναι ή όχι μέλος στην συστάδα. Το κατα πόσο είναι ή όχι μέλος προσδιορίζει και τον συντελεστή για την κάθε συστάδα.



3.4.1 Παρουσιάζεται σχηματικά ένα παράδειγμα εφαρμογής της παραπάνω διαδικασίας όπου χρησιμοποιούνται 4 επιμέρους δίκτυα. Το σύνολο εκπαίδευσης χωρίζεται σε συστάδες και κάθε συστάδα αναλαμβάνει να εκπαιδεύσει κάποιο διαφορετικό δίκτυο. Με μαύρες τελείες αποτυπώνονται τα πρότυπα εκπαίδευσης του συνόλου εκπαίδευσης και με τελείες στο χρώμα του κάθε cluster τα αντίστοιχα κέντρα

3.4.2 Υλοποίηση.

Κατα την υλοποίηση σε αυτόν τον αλγόριθμο, αφού πραγματοποιούμε τα συνηθισμένα βήματα, χωρίσαμε την διαδικασία σε δύο τμήματα (χρησιμοποιώντας 2 συναρτήσεις) .

Στο πρώτο τμήμα -συνάρτηση *clusteredpartition*- πραγματοποιούμε την διαμόρφωση των *clusters* με χρήση του αλγορίθμου *som*. Εκπαιδεύουμε τον χάρτη που θα αποτελείται από τόσες συστάδες όσες ορίσαμε πριν πάμε στην συνάρτηση (στο κύριο πρόγραμμα το n το οποίο περνά σαν παράμετρος) και χρησιμοποιούμε την πιο ελεύθερη μορφή πλέγματος (αλυσίδα) ούτως ώστε να έχουμε αρκετή ελευθερία και ελαστικότητα για τον προσδιορισμό των συστάδων. Ανάλογα με το σύνολο τον αριθμό των εποχών τον προσδιορίζουμε είτε σε 300

στα πιο μικρά είτε σε 100 στα μεγαλύτερα.

Μετα την εκπαίδευση του χάφτη , βάζουμε σε έναν πίνακα τα αντίστοιχα προσδιορισθέντα κέντρα και πραγματοποιούμε το εξής: Επειδή μετα απο αρκετές πειραματικές εκτελέσεις παρατηρήθηκε οτι όταν κάθε δεδομένο εκπαίδευσης ανήκει σε μόνο μία συστάδα ανάλογα με το πιο κέντρο απέχει πιο κοντά απο αυτό , συνήθως τα δίκτυα δεν εκπαιδεύονται ικανοποιητικά και το σύστημα οδηγείται σε χαμηλότερη απόδοση , επιλέγουμε οτι το εκάστοτε πρότυπο εκπαίδευσης θα ανήκει στα k κοντινότερα σε αυτό δίκτυα (κέντρα). Ορίζουμε ετσι τον συντελεστή k ο οποίος αντιπροσωπεύει τον αριθμό των *clusters* που ανήκει το κάθε πρότυπο εκπαίδευσης και η επιλογή των συστάδων αυτών γίνεται με βάση την απόσταση τους απο το κάθε πρότυπο χωριστά . Εμείς εφαρμόσαμε την πιο απλή δυνατή περίπτωση όπου το k είναι σταθερό για όλα τα πρότυπα. Αυτό οδηγεί το σύστημα να εκπαιδευθεί συνολικά με k -φορές το πλήθος των προτύπων εκπαίδευσης.

Αφού λοιπον βρούμε για όλα τα πρότυπα εκπαίδευσης τους k κοντινότερους *clusters* , φτιάχνουμε την δομή *cluster* όπου βάζουμε για τον κάθε *cluster* τα πρότυπα που ανήκουν σε αυτόν και το αντίστοιχο κέντρο του.

Τέλος, βρίσκουμε για όλες τις συστάδες την απόσταση απο το μακρινότερο πρότυπο καθώς και την μέση τιμή της απόστασης απο τα πρότυπα τους ετσι ώστε να προσδιορίσουμε δύο ακτίνες για την κάθε συστάδα, μια μικρής και μια μεγάλης σημασίας για το που ανήκουν τα πρότυπα ελέγχου. Θα μας χρειαστεί για πιο μετά.

Στο δεύτερο βήμα -συνάρτηση *nn_grid_create*- ουσιαστικά βάζουμε στο παραπάνω πλέγμα νευρωνικά δίκτυα -ένα για την κάθε συστάδα- και το καθένα το εκπαιδεύουμε με τα αντίστοιχα πρότυπα της συστάδας στην οποία ανήκει, που έχουν προσδιορισθεί απο το προηγούμενο βήμα. Αξίζει να σημειωθεί οτι όταν έχουμε ορίσει υπερβολικά πολλές συστάδες για σχετικά μικρά σύνολα και αν κάθε πρότυπο ανήκει σε μία μόνο συστάδα, υπάρχει πολύ μεγάλη πιθανότητα να υπάρχουν δίκτυα-συστάδες- που δεν έχουν πρότυπα. Αυτήν την περίπτωση πρέπει να την εξετάζουμε πριν προχωρήσουμε στην φάση της εκπαίδευσης. Αν δεν υπάρχουν πρότυπα , το συγκεκριμένο δίκτυο δεν μπορεί να εκπαιδευθεί. Η συγκεκριμένη συνάρτηση θα έχει για έξοδο μια δομή με το σύνολο των νευρωνικών δικτύων μετα απο την φάση εκπαίδευσης τους.

Μετα απο αυτή τη φάση δημιουργίας - εκπαίδευσης των νευρωνικών δικτύων στα πρότυπα των αντίστοιχων συστάδων, αρχίζουμε την φάση του ελέγχου. Αρχικά πραγματοποιούμε το σύνολο των προσομοιώσεων για όλα τα δίκτυα μας και βάζουμε όλα τα αποτελέσματα σε έναν μεγάλο πίνακα τριών διαστάσεων. Στη συνέχεια ,βρίσκουμε για όλα τα πρότυπα ελέγχου το κέντρο μικρότερης απόστασης και ορίζουμε την τελική απόφαση ως την ταξινόμηση του κάθε προτύπου ελέγχου απο τον κοντινότερο σε αυτό *cluster*-δηλαδη το αντίστοιχο δίκτυο-.

Αυτή είναι η μία περίπτωση προσδιορισμού των προτύπων ελέγχου. Για την άλλη περίπτωση, κάνουμε αυτό που αναφέραμε στο πρώτο κομμάτι της παρούσας ενότητας και βρίσκουμε για το κάθε πρότυπο ελέγχου τις συστάδες στις οποίες αυτό ανήκει και πόσο 'δυνατά' ανήκει σε αυτές. Εφόσον έχουμε αποπρίν προσδιορίσει κάποιες 'ακτίνες' (μικρής σχετικά και μεγαλύτερης) σημασίας για την κάθε συστάδα, βλέπουμε αν κάποιο πρότυπο ελέγχου έχει απόσταση από το αντίστοιχο κέντρο μικρότερη από κάποια από αυτές τις ακτίνες και ανάλογα του δίνουμε για κάποιους πόντους που ποσοτικοποιούν κατά πόσο αυτό ανήκει στον συγκεκριμένο *cluster*. Κατόπιν, με αυτήν την ποσοτικοποίηση σαν συντελεστή για τον κάθε *cluster*, πέρνουμε το κανονικοποιημένο άθροισμα και αυτό είναι το αποτέλεσμά μας για το κάθε πρότυπο ελέγχου.

3.4.3 Αποτελέσματα.

Και εδώ η διαδικασία που ακολουθείται στην παρουσίαση είναι παρόμοια. Σε πρώτο επίπεδο παρουσιάζουμε τους χρόνους των σύνολων από τις διάφορες εκτελέσεις που πραγματοποιήθηκαν για κάθε K:

K=1:

| Number of clusters | N=5 | N=10 | N=20 | N=40 | N=80 |
|--------------------|---------|---------|-------|-------|---------|
| Letters | 130-160 | 120-130 | 80-90 | 90 | 160-190 |
| CTG | 10-20 | 20-30 | 25-30 | 35-40 | - |
| Picture | 7 | 7-10 | 12-15 | 20 | - |
| Abalone | 16 | 18 | 28 | - | - |
| Contraceptive | 12 | 14 | 20 | - | - |

K=3:

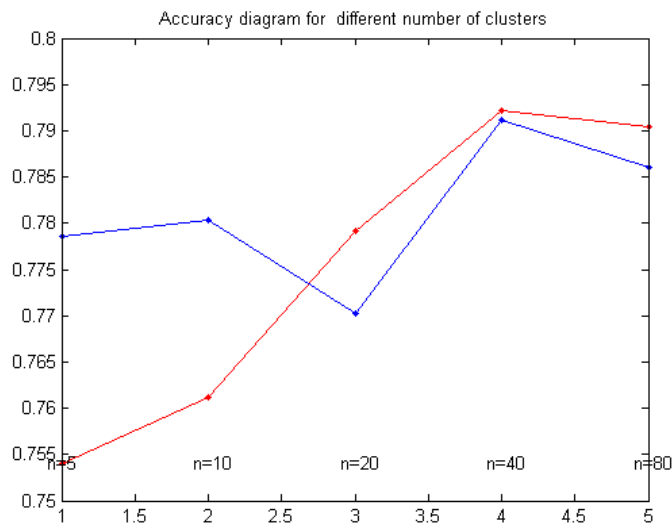
| Number of clusters | N=5 | N=10 | N=20 | N=40 | N=80 |
|--------------------|-------|---------|-------|-------|---------|
| Letters | 660 | 370-460 | 300 | 250 | 220-240 |
| CTG | 50-60 | 40-60 | 40-50 | 60-70 | - |
| Picture | 9-14 | 12-15 | 17-20 | 23-28 | - |
| Abalone | 80 | 62 | 56 | - | - |
| Contraceptive | 24 | 21 | 35 | - | - |

K=5:

| Number of clusters | N=5 | N=10 | N=20 | N=40 | N=80 |
|--------------------|---------|--------|-------|-------|------|
| Letters | - | - | - | - | - |
| CTG | 100-150 | 80-130 | 80-90 | 80-90 | - |
| Picture | 12-16 | 24-30 | 32-34 | 42-44 | - |
| Abalone | - | - | 130 | 100 | - |
| Contraceptive | - | 40 | 50 | - | - |

3.4.3.1 Letters

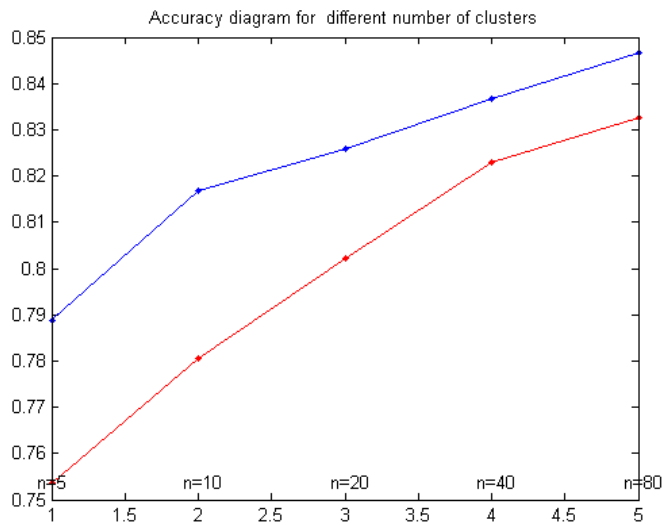
Σε αυτήν την περίπτωση δεδομένου ότι για κάθε σύνολο ουσιαστικά μελετάμε τρεις διαφορετικές υποπερίπτώσεις -με εξαίρεση το πρώτο για το οποίο η περίπτωση K=5 είναι τελείως ασύμφορη από άποψη χρόνου (σε σύγκριση με άλλες μεθόδους που έχουμε δει) - στην αποτύπωση των αποτελεσμάτων και στην παρατήρησή τους, για την κάθε υποπερίπτωση δεν αναφερόμαστε στις επιμέρους κατηγορίες. Επίσης είναι αξιοσημείωτο ότι όταν $n = 5$ και K=5 έχουμε περίπτωση απλού νευρωνικού για μεμονωμένη επιλογή (ή για την ακριβεία επιλογή μεταξύ 5 απλών που έχουν εκπαιδευθεί με όλα τα δεδομένα) K=1:



3.4.2 Η ευστοχία του συνόλου (μέση τιμή 10 εκτελέσεων) letters στην

εφαρμογή του αλγορίθμου μας για 5,10,20,40 και 80 clusters. Με κόκκινο συμβολίζουμε την απλή εφαρμογή με επιλογή ενός μεμονωμένου cluster απο κάθε πρότυπο ελέγχου , ενώ με μπλέ την πολλαπλή επιλογή με τον τρόπο που αναφέραμε παραπάνω.Εδώ το κάθε πρότυπο εκπαίδευσης αντιστοιχεί σε έναν μόνο cluster

K=3:



3.4.3 Παρόμοια με πριν , με το κάθε πρότυπο εκπαίδευσης να εκπαιδεύει τους 3 κοντινότερους clusters

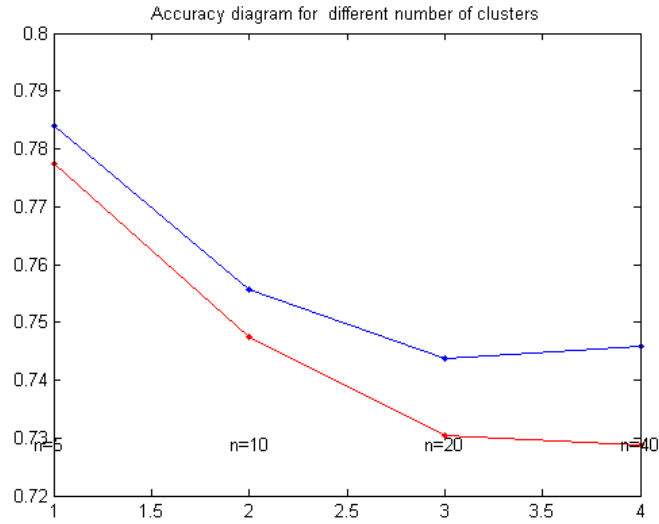
Σχετικά με το συγκεκριμένο σύνολο παρατηρούμε ότι η εφαρμογή του αλγορίθμου μας , απο άποψη ευστοχίας έχει για την περίπτωση K=1 σχετικά μέτρια αποτελέσματα, τα οποία κατα πάσα πιθανότητα ωφείλονται στην ελλιπή εκπαίδευση που παρέχεται στα δίκτυα των clusters, αφού κάθε πρότυπο αντιστοιχεί σε ένα μόνο δίκτυο. Ειδικά στις περιπτώσεις με τα λιγότερα δίκτυα (n=5,10) με ταξινόμηση των προτύπων ελέγχου σε μία συστάδα την φορά , είναι σαφές ότι ο αλγόριθμος δεν ταξινομεί ικανοποιητικά αφού όπως φαίνεται υστερεί ακόμα και σε σχέση με την περίπτωση όπου έχουμε μόνο ένα mlp (εκεί ευστοχία περίπου 78 τοις εκατό) .

Η κατάσταση βελτιώνεται για μεγαλύτερα μεγέθη συστάδων χωρίς όμως να παρατηρείται κάποια εντυπωσιακή βελτίωση. Αναφορικά με την περίπτωση όπου επιλέγουμε πάνω από έναν *cluster* βλέπουμε σχετικά καλύτερη συμπεριφορά για τις περιπτώσεις με τις λίγες συστάδες, αλλά σχετική σταθερότητα με την αύξηση των συστάδων (συνολικά έχουμε διακύμανση από 0.77- 0.7925 σε σχέση με 0.755- 0.795 που έχουμε στην άλλη περίπτωση). Τέλος αξίζει να αναφερθεί ότι η χαμηλή απόδοση για τα μικρότερα μεγέθη συστάδας ίσως είναι απόρροια της πιθανώς μικρότερης σχέσης που έχει συνολικά η συστάδα με το δεδομένο κάθε φορά πρότυπο ελέγχου, εφόσον έχουμε ένα αρκετά μεγάλο σύνολο εκπαίδευσης και προφανώς μη επαρκή αριθμό συστάδων.

Στην περίπτωση όπου $K=3$ τα πράγματα είναι αισθητά καλύτερα. Και πάλι για μικρό αριθμό συστάδων έχουμε μικρή απόδοση στο συστημά μας τόσο για απλή όσο και για πολλαπλή επιλογή συστάδων, καθώς όμως αυξάνουμε το πλήθος βλέπουμε πως έχουμε σχετικά καλή αύξηση της απόδοσης, η οποία πλησιάζει το 0.84 και 0.85 για τις δύο περιπτώσεις επιλογής που μελετάμε. Μάλιστα, η αύξηση είναι σχετικά γραμμική, πράγμα που σημαίνει ότι όσο περισσότερες συστάδες έχουμε τόσο το καλύτερο. Αυτό λογικά συμβαίνει διότι κατά κάποιον τρόπο 'προμηθεύουμε την κάθε συστάδα με επαρκή υλικό' από άποψη δεδομένων εκπαίδευσης.

3.4.3.2 CTG

K=1:

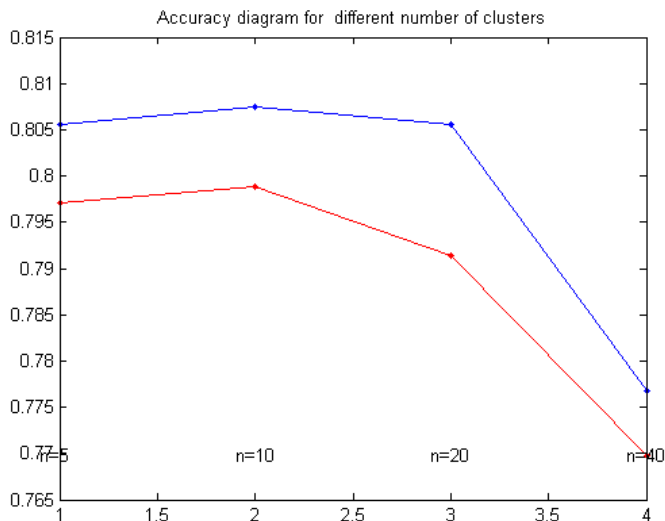


3.4.4 Παρόμοια με πριν , για το σύνολο ctg με $K=1$

Στην περίπτωση αυτή για το συγκεκριμένο σύνολο παρατηρούμε ότι σε αντίθεση με το προηγούμενο σύνολο , τουλάχιστον για $K=1$ έχουμε για τον μικρότερο αριθμό συστάδων την υψηλότερη απόδοση. Αυτό δεν μας εκπλήσσει δεδομένου ότι αυτήν τη φορά έχουμε ένα αρκετά μικρότερο σύνολο και άρα λογικά θα χρειάζεται κάποιον σχετικά μικρότερο αριθμό συστάδων. Για περισσότερες συστάδες μαλιστα, δεδομένου ότι τα πρότυπα πλέον δεν επαρκούν για την εκπαίδευση του κάθε δικτύου ξεχωριστά είναι αναμενόμενη αυτή η συμπεριφορά.

Επίσης συνολικά βλέπουμε με αρκετή σαφήνεια ότι και πάλι για αυτό το K το σύστημα δεν συμπεριφέρεται ιδιαίτερα ικανοποιητικά ειδικά στην μεμονωμένη επιλογή. Απο αυτό σε συνδυασμό με το παραπάνω προκύπτει ότι αν θέλουμε από το σύστημα μας σχετικά καλή απόδοση ,το $K=1$ δεν είναι η καλύτερη επιλογή.

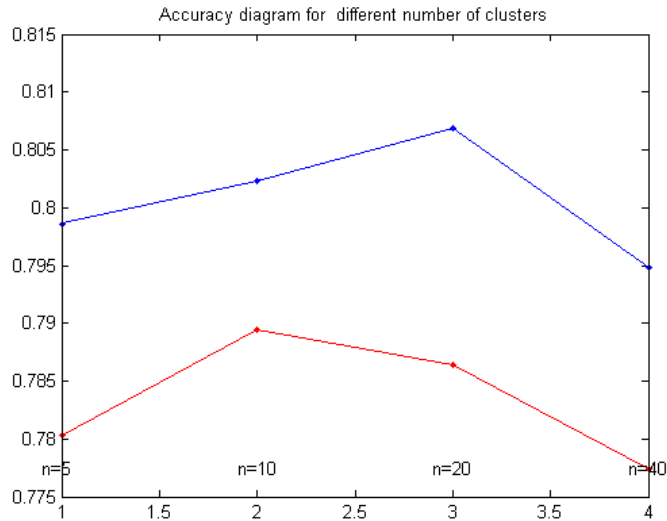
$K=3$:



3.4.5 Παρόμοια με πριν , για το σύνολο ctg με $K=3$

Και για αυτό το σύνολο παρατηρούμε ότι για $K=3$ έχουμε γενικά καλύτερη συμπεριφορά από ότι για $K=1$, πράγμα που είναι αναμενόμενο για τους λόγους που αναφέραμε στην προηγούμενη περίπτωση. Για άλλη μια φορά παρατηρούμε για το συγκεκριμένο σύνολο μία κάπως καλύτερη συμπεριφορά για τους μικρότερους αριθμούς συστάδων, ενώ συνολικά ότι είναι προτιμότερη η επιλογή πολλών συστάδων κατά την φάση του ελεγχου αφού οδηγεί σταθερά σε κάπως καλύτερη απόδοση.

$K=5$:

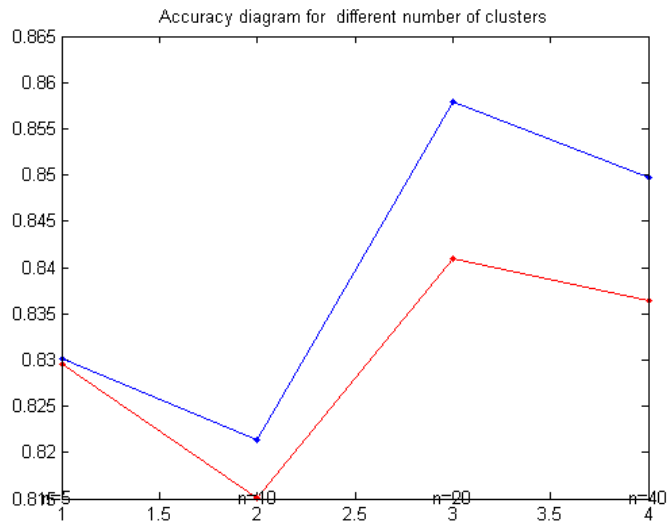


3.4.6 Παρόμοια με πριν , για το σύνολο ctg με $K=5$

Εδώ το πρώτο στοιχείο που παρατηρούμε -όπως άλλωστε αναμενόταν - είναι ότι πλέον για $n = 5$ όπου έχουμε τον μικρότερο αριθμό συστάδων δεν έχουμε την καλύτερη απόδοση , πράγμα που εξηγείται ,αφου πλέον στις περιπτώσεις όπου έχουμε περισσότερους *clusters* έχουμε και αρκετά δεδομένα για να τους εκπαιδύσουμε επαρκώς. Πέραν αυτού είναι σημαντικό το ότι απο άποψη απόδοσης αυτή η περίπτωση ($K=5$) δεν υπερτερεί σχεδόν καθόλου σε σχέση με την προηγούμενη, αλλά έχει μικρότερη διακύμανση για τα διάφορα μεγέθη που εξετάζουμε.

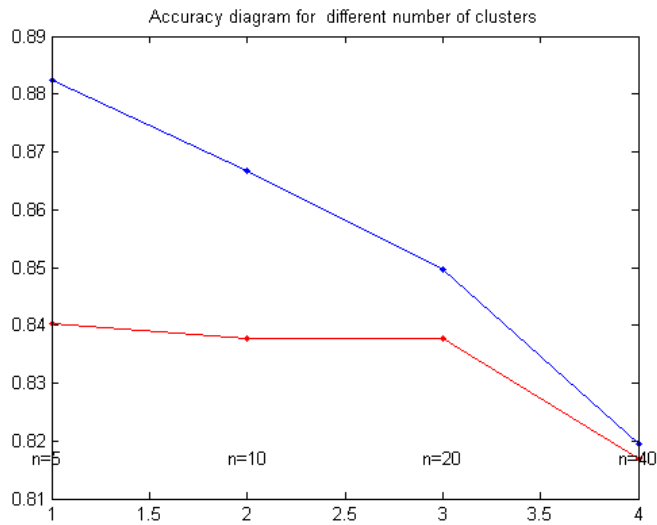
3.4.3.3 Pictures

K=1:



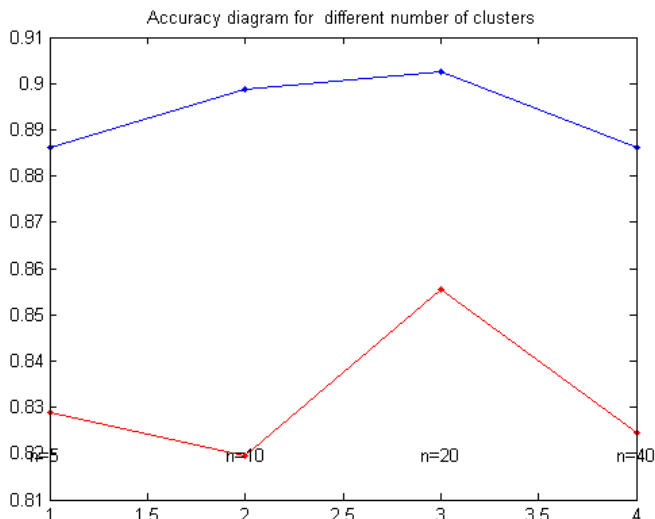
3.4.7 Παρόμοια με πριν , για το σύνολο picture με K=1

K=3:



3.4.8 Παρόμοια με πριν , για το σύνολο picture με K=3

K=5:



3.4.9 Παρόμοια με πριν , για το σύνολο picture με K=5

Για K=1: Εδώ βλέπουμε μια αρκετά μη αναμενόμενη συμπεριφορά αφού, εφόσον το συγκεκριμένο σύνολο είναι μικρό , αναμενόταν να έχει σχετικά καλύτερη συμπεριφορά για μικρά n ιδίως στην συγκεκριμένη περίπτωση (K=1). Αντίθετα , παρατηρούμε οτι η καλύτερη δυνατή ευστοχία σημειώνεται για $n = 20$, ενώ κατι ακόμα που είναι αξιοπρόσεκτο, έγγυται στην σταθερή υπεροχή της περίπτωσης επιλογής πολλών συστάδων σε σχέση με την μεμονωμένη και για αυτό το σύνολο, το οποίο επιβεβαιώνεται και στις υπόλοιπες γραφικές. Συνολικά πάντως ήδη για K=1 βλέπουμε μια ικανοποιητική σχετικά συμπεριφορά απο τον αλγόριθμο αυτό τουλάχιστον απο άποψη απόδοσης.

Για K=3: Παρατηρούμε οτι έχουμε δύο διαφορετικές συμπεριφορές για την μεμονωμένη και την πολλαπλή επιλογή. Στην περίπτωση της μεμονωμένης επιλογής ο αλγόριθμος είναι ιδιαίτερα σταθερός ανεξάρτητα του n μέχρι το 20. Κατόπιν παρατηρούμε μια μικρή πτώση. Απο την άλλη μεριά στην περίπτωση της πολλαπλής επιλογής έχουμε συνεχή πτώση της ευστοχίας πράγμα που πιθανώς να εξηγείται απο την λήψη απόφασης απο λιγότερο σχετικούς με το αντίστοιχο

πρότυπο ελέγχου *clusters*. Σε κάθε περίπτωση η συνολική συμπεριφορά του αλγορίθμου είναι ικανοποιητική για το συγκεκριμένο K .

Για $K=5$: Εδώ βρισκόμαστε προ εκπλήξεως. Δεδομένου ότι ο συγκεκριμένος αλγόριθμος σε κάθε άλλη περίπτωση δεν είχε πουθενά την καλύτερη απόδοση για κάποιο σύνολο, είναι αρκετά εντυπωσιακό ότι για αυτό το K , στην περίπτωση πολλαπλής επιλογής ο αλγόριθμος τα πάει εκπληκτικά στο συγκεκριμένο σύνολο (0.89-0.90 για τα περισσότερα n για νευρωνικά που εκπαιδεύονται με όριο το 0.80 στο σύνολο εκπαίδευσης). Φυσικά, όλα τα δίκτυα (τουλάχιστον μέχρι το $n = 20$) εκπαιδεύονται σίγουρα ικανοποιητικά, αλλά όταν συγκρίνουμε την μεμονωμένη επιλογή -η οποία είναι σαν μια καλύτερη περίπτωση της γραφικής για $K=1$ - βλέπουμε μεγάλη διαφορά. Φαίνεται πως οι επιλογές για αυτήν την περίπτωση γίνονται με τρόπο που βελτιστοποιεί την ποιότητα της τελικής επιλογής, πράγμα που ίσως ωφείλεται στην χωρική τοποθέτηση των προτύπων.

3.4.4 Επεκτάσεις/Τροποποιήσεις.

3.4.4.1 Boosting σε κάθε μεμονωμένο *cluster*

Μια σχετικά απλή επέκταση η οποία θα μπορούσε να πραγματοποιηθεί είναι στον κάθε μεμονωμένο *cluster* να εφαρμόσουμε κάποια τεχνική *boosting*. Δηλαδή για κάθε έναν από τους *cluster* μας να χρησιμοποιήσουμε πάνω από έναν ταξινομητές σε μια συστάδα η οποίοι να αλληλοσυμπληρώνονται για ένα ισχυρότερο αποτέλεσμα. Σε αυτήν την περίπτωση δεν αναμένεται να αλλάξει κάτι στη φάση του ελέγχου, καθώς τα κέντρα των *clusters* δεν θα αλλάζουν, αλλά αναμένεται να αυξηθεί η ευστοχία για την κάθε συστάδα και επομένως η συνολική ευστοχία του συστήματος. Θα μπορούσε να χρησιμοποιηθεί κάποια διαδικασία *boosting* όπως η *adaboost* για να πετύχουμε το ζητούμενο αποτέλεσμα.

Πρέπει να σημειωθεί ότι για την συγκεκριμένη ενότητα αρχικά σκοπός ήταν η ανάπτυξη ενός αλγορίθμου ο οποίος εφαρμόζε *boosting* καταναμημένα στο σύστημα μετά από κάθε εκπαίδευση του συνόλου των *clusters*. Κατά κάποιον τρόπο ήταν ένας συνδυασμός της διαδικασίας που περιγράψαμε -και εκτελέσαμε - στη παρούσα ενότητα με της διαδικασίας που εκτελέσαμε στην προηγούμενη.

Πιο συγκεκριμένα, σκοπός ήταν μετά από την φάση εκπαίδευσης (για όλους τους *clusters*), να φιλτράρουμε συνολικά τα πρότυπα που δεν εκπαιδεύθηκαν σωστά και στην συνέχεια να πραγματοποιούμε με αυτά καιπάλι την διαδικασία χωρισμού σε *clusters* και εκπαίδευσης νέων δικτύων τα οποία θα εκπαιδεύονται πάνω σε αυτά τα πρότυπα. Η διαδικασία θα ολοκληρωνόταν όταν θα είχε

εκπαιδευθεί σωστά κάποιο ζητούμενο ποσοστό του συνόλου εκπαίδευσης

Το συγκεκριμένο σύστημα δοκιμάστηκε σε κάποια σύνολα και τα αποτελέσματα δεν ήταν ικανοποιητικά. Το αντίστοιχο απλούστερο σύστημα που περιγράφηκε στην ενότητα εμφάνιζε πολύ καλύτερα αποτελέσματα από αυτήν την πιο σύνθετη περίπτωση. Λογικά αυτό συνέβη γιατί δεν υπήρξε κάποια αλλαγή στη φάση του ελέγχου μετά την εφαρμογή αυτής της διαδικασίας. Εφόσον σε κάθε επανάληψη αλλάζαμε τον χώρο της εκπαίδευσης (ή καλύτερα ,τον περιορίζαμε) , η τελική απόφαση για το που θα ταξινομηθεί κάποιο πρότυπο ελέγχου είναι κάπως μη συμβατό να λαμβάνεται μεμονωμένα με κριτήριο την απόσταση στο σύνολο του χώρου εκπαίδευσης. Αυτό φυσικά είναι μια υπόθεση για το τι εφταιξε στην αποτυχία αυτής της προσπάθειας.

Απο την άλλη αυτο που περιγράψαμε σε αυτήν την παράγραφο (4.4.1) αφορά τον κάθε αρχικό *cluster* μεμονωμένα και εν γένει αντιμετωπίζει τον κάθεναν σαν κάποιο ξεχωριστό σύνολο εκπαίδευσης και άρα απο τη φύση του κάτι τέτοιο δεν θα έχει το παραπάνω πρόβλημα αφού ο χώρος εκπαίδευσης καθόλη την διαδικασία παραμένει σταθερός.

Παντως , ανεξάρτητα απο το ποια απο τις 2 τεχνικές εφαρμόζουμε κάτι που χρειάζεται προσοχή είναι η επάρκεια των προτύπων. Εφόσον σπάμε το σύνολο σε πολλά υποσύνολα και μετα εφαρμόζουμε μια διαδικασία που χρειάζεται και πάλι πολλά δίκτυα πρέπει να έχουμε έναν πολύ μεγάλο αριθμό προτύπων. Αν δεν ισχύει αυτό , μια ιδέα είναι να εφαρμόσουμε την τεχνική των προτύπων εκπαίδευσης που ανήκουν σε k συστάδες το καθένα που περιγράφηκε στην ενότητα 2.2. Φυσικά αυτό θα αυξήσει την χρονική διάρκεια εκτέλεσης.

3.4.4.2 Αξιοποίηση των κατηγοριών των προτύπων εκπαίδευσης κατα την φάση της δημιουργίας των *clusters*

Το σκεπτικό εδώ είναι οτι κατα τον σχηματισμό των συστάδων θέλουμε να αξιοποιήσουμε το οτι ξέρουμε την κατηγορία στην οποία ανήκει το κάθε πρότυπο εκπαίδευσης και αυτό θελουμε να αποτυπωθεί στον σχηματισμό αυτόν.

Γενικά θα είχε ενδιαφέρον η κατηγορία που ανήκει το κάθε πρότυπο εκπαίδευσης επηρεάσει την ομαδοποίηση με κάποιο τρόπο, αφού αυτό θα οδηγήσει σε κάποια διαφορετική διαμόρφωση των συστάδων η οποία πιθανώς να μπορεί να ευνοήσει την εκπαιδευτική διαδικασία των ταξινομητών μας.

Ενας σχετικά απλός τρόπος για να επηρεάσουμε την ομαδοποίηση ανάλογα με την κατηγορία του κάθε προτύπου εκπαίδευσης είναι ο ακόλουθος. Αρχικά ως υποθέσουμε οτι

$$\mathbf{x} = [x_1 x_2 \dots x_n] \quad (90)$$

είναι κάποιο πρότυπο απο το σύνολο εκπαίδευσης. Θυμίζουμε οτι στην απλή περίπτωση χαρτη αυτοοργάνωσης ,για το πρότυπο αυτό, το μοντέλο $\mathbf{m}_i(t) = [m_1(t)m_2(t)...m_n(t)]$ επηρεάζεται ως εξής για την επανάληψη t :

$$\mathbf{m}_i(t+1) = \mathbf{m}_i(t) + a(t)h_{ci}(t)[\mathbf{x} - \mathbf{m}_i] \quad (91)$$

όπου $a(t)$ ο παράγοντας μάθησης (εδώ εξαρτώμενος απο τον χρόνο) και h_{ci} η συνάρτηση γειτονίας για το μοντέλο i . Αυτό που μπορούμε να κάνουμε είναι να δώσουμε στο κάθε μοντέλο m_i μια κατηγορία-αναφορά, έστω $ct(m_i)$ η οποία θα αποτελεί κάποια απο τις πιθανές κατηγορίες των προτύπων εκπαίδευσης και αυτό ουσιαστικά θα έλκεται' απο τα πρότυπα τα οποία ανήκουν σε αυτην την κατηγορία και θα άπωθείται' απο πρότυπα τα οποία ανήκουν σε άλλες κατηγορίες. Η βασική σχέση λοιπόν αναμένεται να επεκτείνεται κάπως έτσι με βάση αυτην την τροποποίηση:

$$\mathbf{m}_i(t+1) = \mathbf{m}_i(t) + a_1(t)h_{ci}(t)[\mathbf{x} - \mathbf{m}_i] + sgt(t)a_2(t)h_{ci}(t)[\mathbf{x} - \mathbf{m}_i] \quad (92)$$

με

$$sgt(t) = \begin{cases} +1 & \text{if } ct(\mathbf{x}) = ct(\mathbf{m}_i) \\ -1 & \text{if } ct(\mathbf{x}) \neq ct(\mathbf{m}_i) \end{cases}$$

,ζητωντας επίσης

$$a_1(t) + a_2(t) = a(t) \quad \forall t \quad (93)$$

Η παραπάνω σχέση επειδέχεται δύο ερμηνείες. Αρχικά μπορούμε να πούμε οτι εάν η κατηγορία κάποιου προτύπου εκπαίδευσης δεν είναι ίδια με την κατηγορία αναφοράς του m_i και αν $a_2 < a_1$, κατα κάποιον τρόπο περιορίζουμε την 'μάθησή' του μοντέλου απο το συγκεκριμένο πρότυπο, ενώ αν είναι ίδια την ενισχύουμε.

Απο την άλλη αυτη η διαδικασία μπορεί να ερμηνευθεί σαν μια ακόμη επίδραση στην διαδικασία μάθησης συνολικά η οποία ωθεί τα μοντέλα πιο κοντά στα πρότυπα της κατηγορίας τους.

Φυσικά , όπως σε κάθε άλλη περίπτωση , έτσι και με αυτην την τεχνική υπάρχουν διάφορα ζητήματα. Κατάρχάς πρέπει να αποφασίσουμε πως θα κατανήσουμε τις κατηγορίες στα μοντέλα. Η πιο απλή λύση είναι να μετρήσουμε το ποσοστό που καταλαμβάνει η κάθε κατηγορία στο σύνολο των προτύπων εκπαίδευσης και να το αποτυπώσουμε στην ανάθεση κατηγοριών στα μοντέλα (πχ αν απο 1000 πρότυπα εκπαίδευσης, μια κατηγορία έχει 300 και θέλουμε 10 συστάδες, αυτη θα την κάνουμε κατηγορία αναφοράς για 3 μοντέλα κατα την ομαδοποίηση).

Ενα άλλο σημαντικό ζήτημα σχετίζεται με το γεγονός ότι η αρχική θέση των μοντέλων ενδέχεται να είναι τυχαία πράγμα που σημαίνει ότι ενδέχεται κάποιο μοντέλο να ξεκινήσει σε κάποια κακή για αυτό θέση όπου σε όλη του την γειτονία υπάρχουν μόνο πρότυπα διαφορετικών κατηγοριών. Αυτό με τη σειρά του θα οδηγήσει σε μείωση του ρυθμού προσαρμογής για το συγκεκριμένο μοντέλο χωρίς ουσιαστικά να του παρέχει κάποια κατεύθυνση προς πρότυπα της κατηγορίας αναφοράς του, εκτός και αν τυχαία , με την αλλαγή της θέσης του τύχει να φτάσει κοντά σε κάποιο /κάποια τέτοια.

3.4.4.3 Χρήση Γκαουσιανών στη θέση των ακτίνων κατά τον προσδιορισμό των προσδιορισμό των προτύπων ελέγχου

Αυτή η τροποποίηση σχετίζεται με την αλλαγή των ακτίνων τις οποίες προσδιορήσαμε κατά την υλοποίηση για να βρούμε κατά πόσο ανήκει κάποιο πρότυπο ελέγχου σε κάποιον *cluster* με μια συνέχη και παραγωγίσιμη συνάρτηση ,όπως η γκαουσιανή.

Με αυτόν τον τρόπο , ουσιαστικά αλλάζουμε μια ασυνεχή συνάρτηση περιπτώσεων η οποία είναι χαρακτηριστικά απότομη , με μια πιο ομαλή συνάρτηση. Αυτό θα μπορούσε να εξασφαλίσει το ότι πρότυπα τα οποία είναι αρκετά κοντά , θα έχουν πιο λογικά κοντινές τιμές από την προηγούμενη περίπτωση , όπου ενδέχεται 2 πολύ κοντινα πρότυπα να είχαν πολύ διαφορετικές τιμές σημασίας για κάποιον *cluster*.

Φυσικά δεδομένου ότι οι γκαουσιανές μηδενίζονται πρακτικά στο άπειρο, αυτό θα σημαίνει ότι όλα τα πρότυπα ελέγχου θα έχουν συμμετοχή (έστω και πολύ μικρή) σε όλα τα δίκτυα (συστάδες) , πράγμα το οποίο ίσως δεν είναι επιθυμητό λόγω του ότι κάποιο πρότυπο ελέγχου μπορεί να είναι πολύ μακριά από τα πρότυπα με τα οποία εκπαιδεύθηκε άποιο δίκτυο, πράγμα που συνεπάγεται ότι το συγκεκριμένο δίκτυο κατά πάσα πιθανότητα δεν μπορεί να το ταξινομήσει σωστά. Για να αντιμετωπίσουμε τέτοια συμβάντα θα μπορούσαμε να 'κόψουμε' τις γκαουσιανές μας σε κάποιο σχετικά 'στρατηγικό σημείο'

3.5. Μελέτη συστήματος προσαρμογής των clusters προτύπων εκπαίδευσης στα πρότυπα ελέγχου

3.5.1 Γενικά.

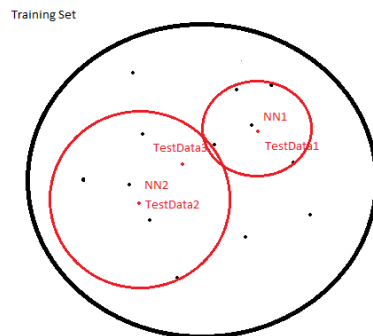
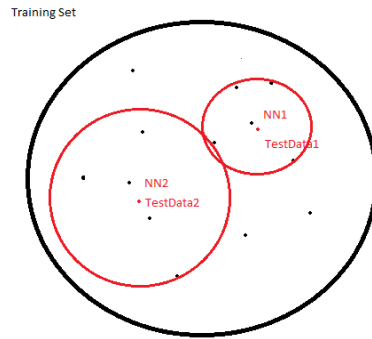
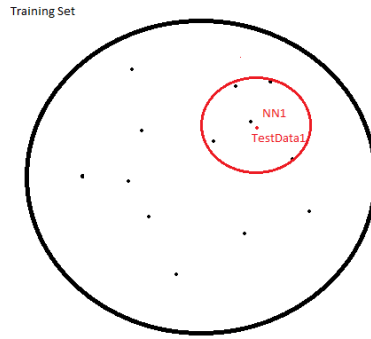
Σε αυτό το τμήμα θα μελετήσουμε την συμπεριφορά ενός συστήματος το οποίο σε γενικές γραμμές μοιάζει με το προηγούμενο, αλλά εφαρμόζει την διαδικασία σχηματισμού των *clusters* (συστάδων) με αρκετά διαφορετικό τρόπο.

Πιο συγκεκριμένα το σύστημα αν και υπάγεται στην επιβλεπόμενη μάθηση, δεν θα έχει φάση εκπαίδευσης, παρα μόνο φάση ελέγχου. Το σκεπτικό είναι ότι θέλουμε η εκπαίδευση του συστήματος να προσαρμόζεται στα δεδομένα ελέγχου κάθε φορά. Αυτό φυσικά συνεπάγεται ότι για διαφορετικές εκτελέσεις όπου παρεμβάλλουμε τυχαίες μεταθέσεις μεταξύ των δεδομένων εκπαίδευσης και ελέγχου εάν βέβαια δίνεται αυτή η δυνατότητα απο το εκάστοτε σύνολο δεδομένων αναμένεται να έχουμε και διαφορετικό πλήθος δικτύων ανάλογα με τις ανάγκες του συνόλου ελέγχου κάθε φορά.

Η διαδικασία που ακολουθείται είναι η εξής: Για κάθε νέο πρότυπο ελέγχου που 'έρχεται' στο σύστημα εξετάζουμε αν -ή κατα πόσον - ανήκει σε κάποια συστάδα.

Αν δεν ανήκει σε κάποιον *cluster*, τότε δημιουργούμε ένα δίκτυο με κέντρο το συγκεκριμένο πρότυπο ελέγχου και το εκπαιδεύουμε με τα *rad* κοντινότερα πρότυπα εκπαίδευσης έτσι ώστε να σχηματισθεί μια συστάδα. Ο χώρος (R^n όπου n η διάσταση του χώρου χαρακτηριστικών) που θα λάβει ο κάθε *cluster* καθορίζεται απο την απόσταση απο το πρότυπο εκπαίδευσης που απέχει πιο πολύ.

Αν ανήκει σε κάποιον *cluster* -ή αν ξεπερνά ενα όριο επικάλυψης σε διάφορους *clusters*-, τότε θεωρούμε ότι έχουμε αρκετή πληροφορία για να αποφασίσουμε πώς θα ταξινομηθεί και οι διάφοροι *clusters* στους οποίους ανήκει, αποφασίζουν για την ταξινόμηση του. Η επικάλυψη καθορίζεται γενικά με το αν ανήκει ή όχι στην συστάδα το δεδομένο πρότυπο ελέγχου και αυτό με την σειρά του προσδιορίζεται απο την απόσταση απο το μακρινότερο πρότυπο εκπαίδευσης του συγκεκριμένου *cluster*. Παρακάτω ακολουθεί ένα σχηματικό παράδειγμα το οποίο θα κάνει πιο σαφή την διαδικασία.



3.5.1 Παρουσιάζεται σχηματικά ένα παράδειγμα εφαρμογής της παραπάνω διαδικασίας. Τα πρότυπα ελέγχου παρουσιάζονται με κόκκινο και τα πρότυπα εκπαίδευσης με μαύρο. Παρατηρούμε ότι στην περίπτωση του 3ου προτύπου ελέγχου (τρίτη εικόνα) δεν σχηματίζεται νέο δίκτυο (και άρα συστάδα) δε-

δομένου ότι έχουμε επικάλυψη και επομένως το NN2 θα αποφανθεί για την ταξινόμηση του *TestData3* . Επίσης έχουμε ορίσει μέγεθος *cluster* ίσο με 5 και άρα 5 πρότυπα εκπαίδευσης για κάθε οριζόμενη συστάδα.

Φυσικά αυτό που ορίζουμε ως ικανοποιητική επικάλυψη είναι σχετικό. Για παράδειγμα , μπορούμε στον αλγόριθμό αυτό να ορίσουμε ως ικανοποιητική επικάλυψη για ένα πρότυπο ελέγχου τους 5 *clusters*. Αυτό σημαίνει ότι αν κάποιο πρότυπο επικαλύπτεται από λιγότερες συστάδες (πχ 4) θα δημιουργήσει νέο δίκτυο και νέα συστάδα. Αυτό με την σειρά του αναμένεται να προσδώσει καλύτερα αποτελέσματα στο σύστημα , με μεγαλύτερη όμως καθυστέρηση δεδομένου ότι είναι πιθανότερο να σχηματισθούν περισσότερες συστάδες σε μια τέτοια περίπτωση.

Γενικά αυτή η μέθοδος αποτελεί μια συνολικά καλή τεχνική , αφού παρακολουθεί την διαφοροποίηση στα πρότυπα ελέγχου και εκμεταλλεύεται την διαφορετική τοποθέτηση τους στον χώρο των χαρακτηριστικών για να φτιάξει μια χωρικά δυνατή ομάδα σχεδιασμένη ειδικά για το συγκεκριμένο κάθε φορά σύνολο ελέγχου. Το κύριο μειονέκτημα έγκυται στο ότι για μικρότερα σύνολα η συγκεκριμένη τεχνική είναι αργή τουλάχιστον σε σχέση με άλλες μεθόδους. Αυτό ωφείλεται στο γεγονός ότι γενικά έχει την τάση να χρησιμοποιεί αρκετά δίκτυα ιδίως όταν έχουμε ετερογενή πρότυπα ελέγχου.

Κάτι που ακόμη αξίζει να σημειωθεί είναι ότι σε αυτήν την τεχνική φυσικά μπορούμε να εφαρμόσουμε και πάλι διάφορα κριτήρια για τον προσδιορισμό της βαρύτητας των συντελεστών που θα έχει το κάθε δίκτυο από αυτά στα οποία ανήκει το δεδομένο πρότυπο ελέγχου, όπως για παράδειγμα το αντιστρόφως ανάλογο της απόστασης κριτήριο. Βεβαίως αν το πρότυπο έχουμε θεωρήσει ότι δεν ανήκει κάπου και δημιουργεί δικό του δίκτυο , το λογικό είναι ότι αυτό θα έχει όλο το βάρος της απόφασης για το πρότυπο αυτό.

3.5.2 Υλοποίηση

Εδώ, αφού πραγματοποιήσουμε τα βήματα ανάγνωσης και προεπεξεργασίας των δεδομένων μας, αρχικοποιούμε την ακτίνα (η οποία θα μετρείται σε πλήθος προτύπων και όχι σε καθαρή απόσταση) που θα έχουν οι συστάδες μας, καθώς και την δομή μέσω της οποίας προσδιορίζουμε τις συστάδες που θα ανήκει το κάθενα από τα πρότυπα ελέγχου. Η δομή αυτή *member1* θα περιλαμβάνει έναν πίνακα σημαιών (*member1(i).oncluster(iq)*) η οποία θα δείχνει κατα πόσο το Test Data είναι μέλος της συστάδας *iq* , καθώς και έναν πίνακα ο οποίος θα

έχει την απόσταση του κάθε προτύπου απο το κέντρο της/των συστάδων στις οποίες ανήκει.

Αφου λοιπόν αρχικοποιήσουμε τα παραπάνω,για διευκόλυνση θα ορίσουμε και αρχικοποιήσουμε επιπλέον έναν πίνακα (checksum) ο οποίος θα προσδιορίζει το πλήθος των συστάδων στο οποίο ανήκει το δεδομένο κάθε φορά πρότυπο

Κατόπιν ελέγχουμε αν στο σύστημα μας υπάρχουν συστάδες (δίκτυα) και αν υπάρχουν , για κάθε δίκτυο του συστήματος , για το συγκεκριμένο πρότυπο ελέγχου στο οποίο είμαστε ,αφού αρχικοποιήσουμε την δομή του για το νέο πρότυπο, εξετάζουμε αν ανήκει στον *cluster* που κάθε φορά ελέγχεται συγκρίνοντας την απόσταση του προτύπου απο το κέντρο σε σχέση με την απόσταση του πιο μακρινού προτύπου εκπαίδευσης απο αυτο. Αν ανήκει, κάνουμε την αντιστοιχη σημαία 1 και θέτουμε την αντίστοιχη απόσταση. Στη συνέχεια προσδιορίζουμε στην θέση του πίνακα *checksum* για το συγκεκριμένο πρότυπο το πλήθος των συστάδων στις οποίες ανήκει.

Στην περίπτωση όπου το συγκεκριμένο πρότυπο δεν ανήκει σε καμία συστάδα (έχουμε λάβει την απλούστερη περίπτωση όπου μία επικάλυψη θεωρείται οτι επαρκεί για να μην δημιουργηθει καινούργια συστάδα), ή είναι το πρώτο (δεν υπάρχει κανένα δίκτυο), ενημερώνουμε τον μετρητή δικτύων και ορίζουμε νεες θέσεις στους πίνακες του κελιού της δομής *member* για το δεδομένο πρότυπο, οι οποίες αφορούν το νέο δίκτυο που αναμένεται να δημιουργηθεί. Ενημερώνουμε και το *checksum* αφού πλέον , με το νέο δίκτυο , το πρότυπο ελέγχου αυτο θα ανήκει στον νεοσχηματισθέν *cluster*.

Επειτα, βρίσκουμε για όλα τα πρότυπα εκπαίδευσης τις αποστάσεις τους απο το συγκεκριμένο πρότυπο ελέγχου, η θέση του οποίου θα αποτελεί και το κέντρο του νεου *cluster* και ταξινομούμε τον πίνακα. Τα *radius* πιο κοντινά πρότυπα εκπαίδευσης τα βάζουμε στους πίνακες για τα δεδομένα μιας νέας δομής -της *clusters*- ,ως δεδομένα εκπαίδευσης του συγκεκριμένου *cluster*. Σαν μέγιστη απόσταση και άρα ακτίνα της συστάδας(*cluster*) ορίζουμε την απόσταση που έχει το πρότυπο εκπαίδευσης με τιμή *radius* στον ταξινομημένο πίνακα μας με τις αποστάσεις (αρα το πιο μακρινό).

Μετα και απο τα παραπάνω εκπαιδεύουμε το δίκτυο -εκπρόσωπο της νέας συστάδας με τον κλασσικό απλό μας τρόπο και στη συνέχεια ορίζουμε στην δομή το κέντρο του δεδομένου *cluster* ως τη θέση του συγκεκριμένου προτύπου.

Η διαδικασία αυτή θα επαναληφθει για όλα τα πρότυπα ελέγχου, ετσι ώστε να φτιαχτεί ένα σύνολο απο συστάδες το οποιο τελικά θα περιέχει όλα τα πρότυπα.

Αφου γίνει αυτό , ορίζουμε έναν μεγάλο πίνακα προσομοιώσεων και -όπως και τις προηγούμενες φορές και για λόγους εξοικονόμησης χρόνου- προσομοιώ-

νομε όλα τα δίκτυα σε αυτόν.

Τέλος για όλα τα πρότυπα ελέγχου αρχικοποιούμε τους πίνακες των τελικών αποτελεσμάτων και για όλα τα δίκτυα προσδιορίζουμε για το καθένα τους αντίστοιχους συντελεστές(αντίστροφα αναλογικά της απόστασης και ίσο κανονικοποιημένο ποσοστό για τον κάθε *cluster*).

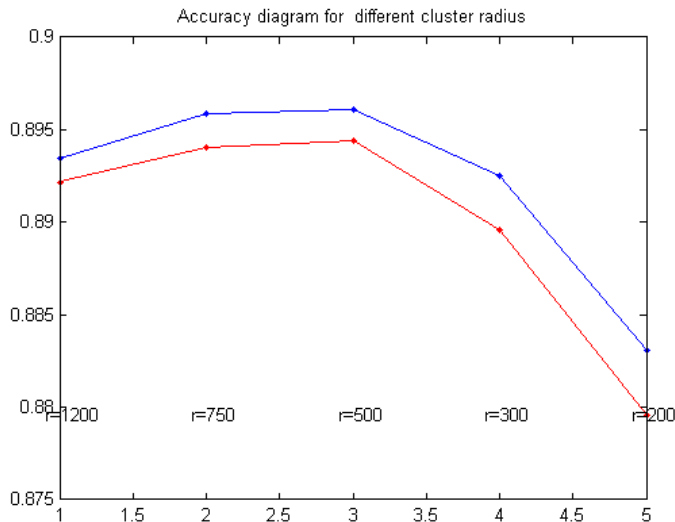
3.5.3 Αποτελέσματα.

Ως συνήθως , παρουσιάζουμε πρώτα τους χρόνους των αντίστοιχων εκτελέσεων για την δεδομένη παραμετροποίηση που χρησιμοποιήθηκε:

| Number of examples | Rad=1200 | Rad=750 | Rad=500 | Rad=300 | Rad=200 |
|--------------------|----------|---------|---------|---------|---------|
| Letters | 500-600 | 320-390 | 270-300 | 250-280 | 240-260 |
| CTG | 100-150 | 80-120 | 65-100 | 70-90 | 65-90 |
| Picture | - | - | 11-17 | 15-19 | 15-20 |
| Abalone | - | 70 | 45 | 52 | 70 |
| Contraceptive | - | 37 | 37 | - | - |

3.5.3.1 Letters

Αρχικά παρουσιάζουμε την ευστοχία του συστήματος για το συγκεκριμένο σύνολο κατα τα γνωστά , για τις διαφορες τιμές ακτίνας συστάδας που δοκιμάσαμε. Κατόπιν εμφανίζουμε το μέγεθος *recall* για τις αντίστοιχες τιμές ακτίνας ανα κατηγορία , ως συνήθως έτσι ώστε να έχουμε μια μετρική για την απόδοση του δεδομένου συστήματος για το συγκεκριμένο σύνολο σε κάθε κατηγορία.

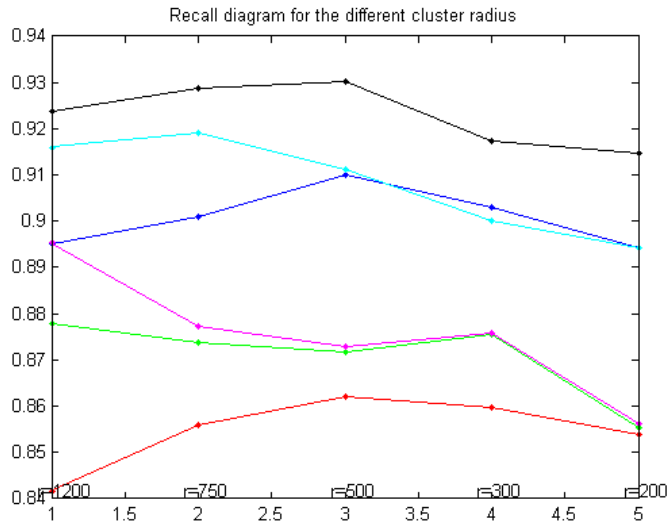


3.5.2 Η ευστοχία του συνόλου (μέση τιμή 10 εκτελέσεων) letters στην εφαρμογή του αλγορίθμου για τιμές ακτίνας (δηλαδή αριθμό προτύπων που θα εκπαιδεύσουν τον δεδομένο cluster). Με κόκκινο συμβολίζουμε την απλή εφαρμογή με επιλογή ενός μεμονωμένου cluster από κάθε πρότυπο ελέγχου, ενώ με μπλέ αναλογική εφαρμογή για πρότυπα ελέγχου που δεν ορίζουν συστάδα.

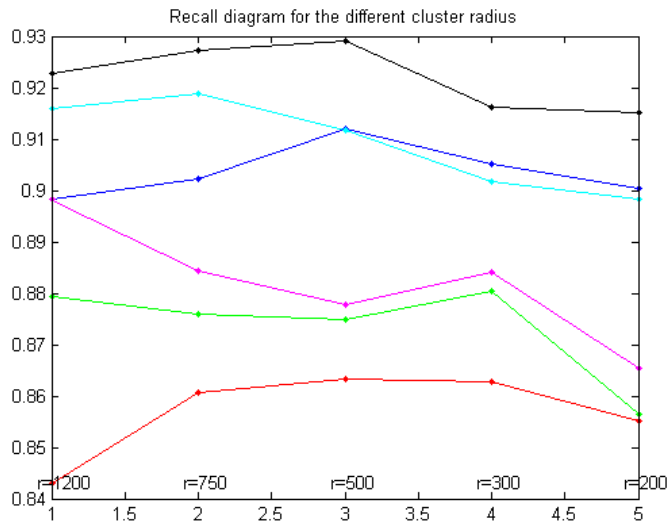
Βλέπουμε εδώ ότι για το συγκεκριμένο σύνολο η ευστοχία του συστήματος κορυφώνεται στην περίπτωση προσδιορισμού αναλογικών συντελεστών, για ακτίνα συστάδας η οποία ισούται με περίπου 500 πρότυπα εκπαίδευσης. Γενικά είναι αρκετά εμφανές το ότι υπάρχει μια σχεδόν σταθερή διαφορά μεταξύ των δύο περιπτώσεων προσδιορισμού συντελεστών που μελετάμε η οποία μάλιστα είναι σχετικά ανεξάρτητη από την ακτίνα.

Πιο συγκεκριμένα βλέπουμε ότι η κόκκινη γραφική παράσταση είναι σχετικά ίδια με την μπλέ με μικρότερη απόδοση. Επίσης, φαίνεται ότι οι αρχικές ακτίνες είναι σχετικά μεγάλες για το σύστημα και οι τελευταίες κάπως μικρές δεδομένου ότι οι υψηλότερες αποδόσεις είναι σχετικά στην μέση. Για πολύ μικρές ακτίνες μάλιστα ($r = 300, 200$) βλέπουμε ότι η απόδοση είναι η χαμηλότερη που παρέχει το σύστημα, ενώ στις πιο μεγάλες η διαφορές είναι σχετικά μικρότερες.

Συνολικά πάντως η ευστοχία του είναι ομολογουμένως εντυπωσιακά υψηλή σε σχέση με τις άλλες μεθόδους που έχουμε εξετάσει (η υψηλότερη που έχουμε δει για το συγκεκριμένο σύνολο.) πράγμα το οποίο πιστοποιεί σε συνδυασμό με τον ικανοποιητικό χρόνο εκτέλεσης τον οποίο παρέχει ότι ο συγκεκριμένος αλγόριθμος είναι πολύ καλός για σχετικά μεγαλύτερα σύνολα.



3.5.3 H recall του συστήματος για όλες τις κατηγορίες, για τις διαφορετικές ακτίνες που δοκιμάσαμε στην περίπτωση επιλογής μεμονωμένης συστάδας



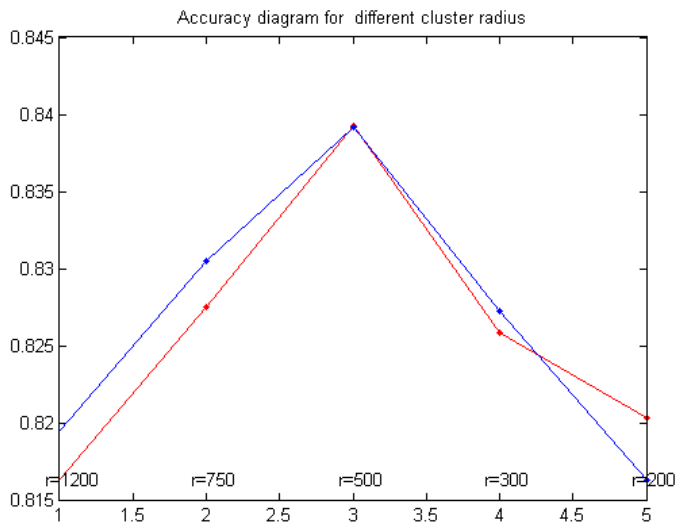
3.5.4 H recall του συστήματος για όλες τις κατηγορίες, για τις διαφορε-

τικές ακτίνες που δοκιμάσαμε στην επιλογή με βάρος αντίστροφως ανάλογο της απόστασης σε όσες συστάδες ανήκει το δεδομένο πρότυπο ελέγχου.

Γενικά όπως προκύπτει οι γραφικές για τις δύο περιπτώσεις μοιάζουν πάρα πολύ για τις περισσότερες κατηγορίες . Και στις δύο περιπτώσεις το ευρος είναι περίπου 9 τοις εκατό (84-93 τοις εκατό) και η συμπεριφορά της κάθε κατηγορίας ως προς την ακτίνα είναι σχεδόν ίδιες. Για τον λόγο αυτο λογικά έχουν και τελική αντίστοιχη συμπεριφορά οι ευστοχίες τους.

Καθώς η ακτίνα μειώνεται βλέπουμε ότι ανα κατηγορία δεν υπάρχει κάποια φαινομενικά σταθερή συμπεριφορά αλλά παρόλα αυτά , για τις μικρές τιμές ακτίνας είναι αρκετα εμφανές μετα απο κάποια πρατήρηση ότι οι περισσότερες κατηγορίες έχουν σχετικά χαμηλότερη απόδοση απότι στις αρχικές και στις μέσες τιμές ακτίνας, ενώ στην μέση , για 500 έχουμε για τις περισσότερες κατηγορίες μια πιο καλή απόδοση. Τα παραπάνω και πάλι αιτιολογούν την διακύμανση της γραφικής της ευστοχίας αυτην την φορά με παράμετρο την ακτίνα των συστάδων

3.5.3.2 CTG

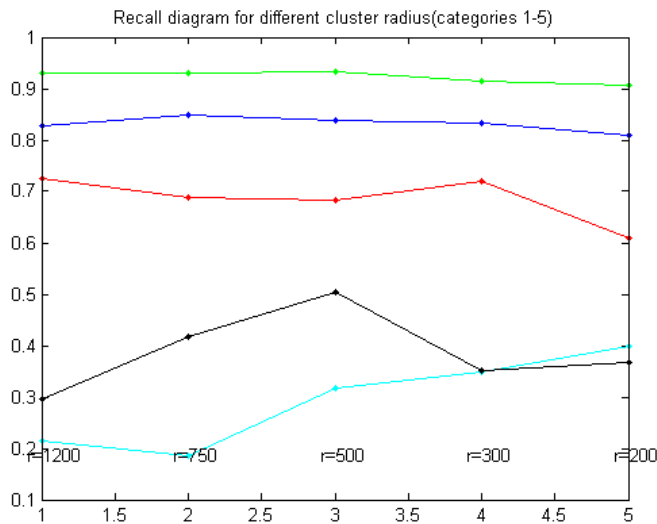


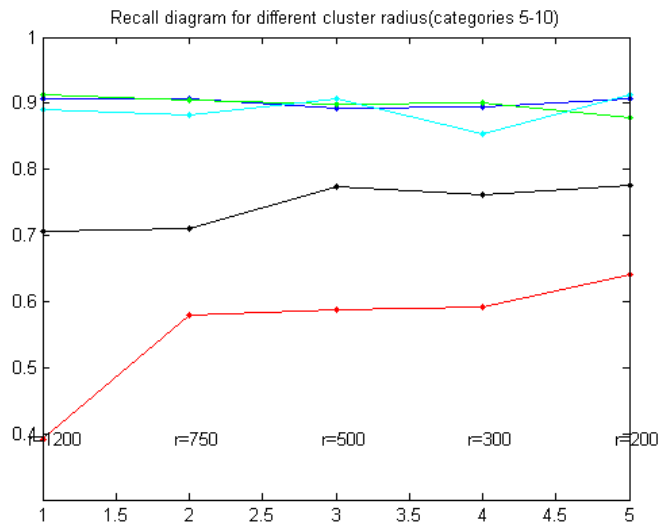
3.5.5 Η ευστοχία του συνόλου (μέση τιμή 10 εκτελέσεων) CTG αντίστοιχα με πριν.

Εδώ τα πράγματα είναι αρκετά ξεκάθαρα, καθότι βλέπουμε και πάλι ότι στην πλειοψηφία των περιπτώσεων η περίπτωση πολλαπλής αναλογικής επιλογής υπερτερεί της μεμονωμένης επιλογής *cluster*, ενώ και πάλι παρατηρούμε ότι έχουμε κορύφωση για ακτίνα συστάδας ίση με 500 πρότυπα εκπαίδευσης.

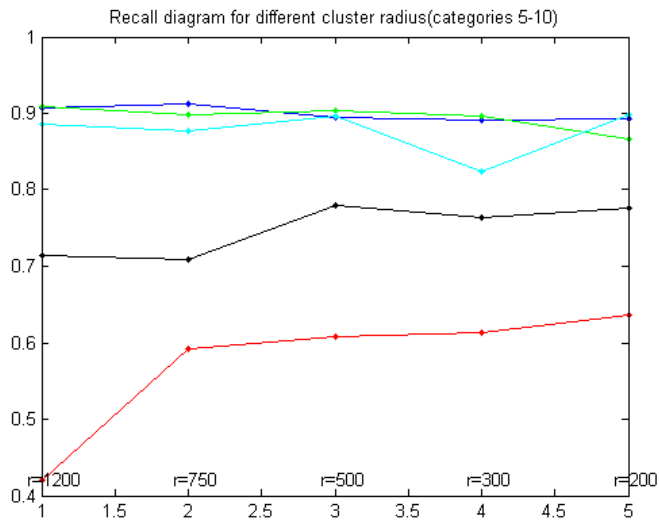
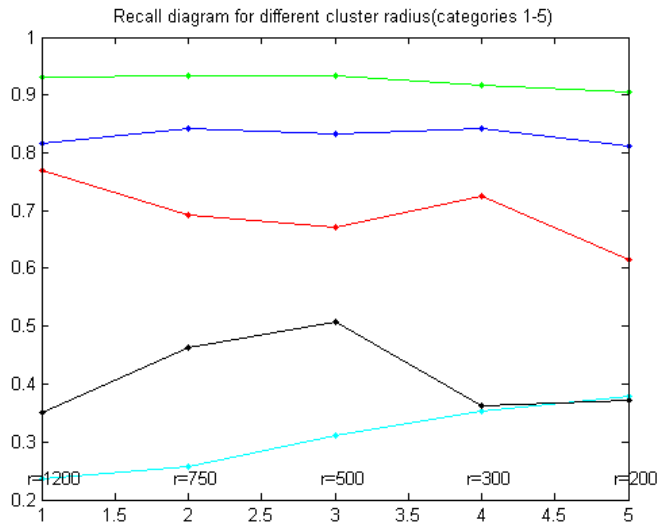
Έχει ενδιαφέρον το ότι για τις διάφορες ακτίνες που δοκιμάσαμε βλέπουμε μια συμμετρικότητα γύρω από την κορύφωση στην ακτίνα 500. Επίσης, παρατηρούμε πως το συνολικό εύρος για τις διαφορες τιμές είναι σχετικά μικρό. Ειδικότερα, βλέπουμε πως είναι της τάξης του μόλις 2.5 τοις εκατό (απο 81.6 για 200 και 1200 μέχρι 84 για 500).

Πάντως και πάλι είμαστε ικανοποιημένοι από την συνολική απόδοση του αλγορίθμου δεδομένου ότι το η απόδοση του είναι ικανοποιητική σε όλο το εύρος των δυνατών περιπτώσεων που δοκιμάσαμε, τουλάχιστον σε σύγκριση με τις άλλες περιπτώσεις και την περίπτωση απλού *mlp*.





3.5.6 Η recall του συστήματος για όλες τις κατηγορίες, για τις διαφορετικές ακτίνες που δοκιμάσαμε στην περίπτωση επιλογής μεμονωμένης συστάδας. Έχουμε 10 κατηγορίες και τις αποτυπώνουμε με δύο γραφικές για λόγους απλότητας και κατανόησης



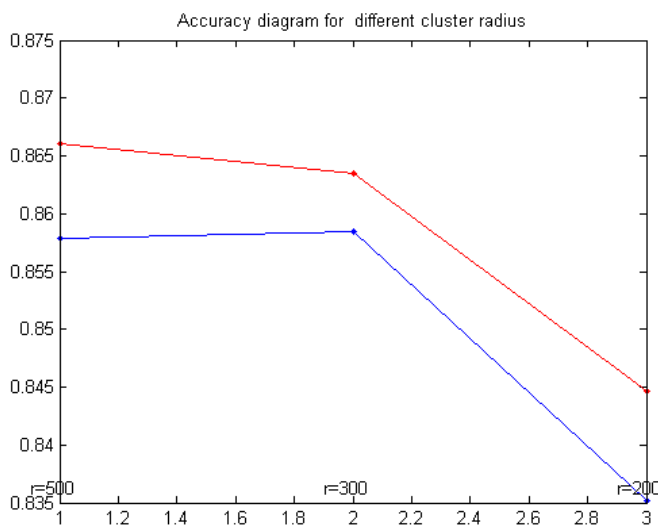
3.5.7 Η recall του συστήματος για όλες τις κατηγορίες , για τις διαφορετικές ακτίνες που δοκιμάσαμε στην επιλογή με βάρος αντίστροφως ανάλογο της απόστασης σε όσες συστάδες ανήκει το δεδομένο πρότυπο ελέγχου. Έχουμε 10 κατηγορίες και τις αποτυπώνουμε με δύο γραφικές για λόγους απλότητας και κατανόησης

Βλέπουμε εδώ ότι γενικά οι κατηγορίες με την μείωση της ακτίνας είναι αρκετά σταθερές και δεν επηρεάζονται σε μεγάλο βαθμό από αυτήν, ειδικά στην περίπτωση της μεμονωμένης ακτίνας. Αυτό προφανώς αιτιολογεί και την σχετικά μικρή διακύμανση του συνόλου τιμών της ευστοχίας που σχολιάσαμε παραπάνω. Σχετικά μεγάλη διακύμανση για την περίπτωση μεμονωμένης επιλογής (από 20 τοις εκατό και πάνω) έχουμε μόνο για 2 από τις δέκα κατηγορίες.

Γενικά οι κατηγορίες στην ακτίνα 500 δεν έχουν ελάχιστα, ενώ κάποιες έχουν και μέγιστα πράγμα το οποίο όπως είναι λογικό αιτιολογεί την μέγιστη τιμή ευστοχίας για ακτίνα ίση με 500.

Όσον αφορά την αναλογική βαθμολόγηση βαρών (δεύτερη ομάδα γραφικών) παρατηρούμε ότι έχουμε σχετικά παρόμοια συμπεριφορά, όμως μερικές κατηγορίες έχουν την τάση να μειώνονται πιο έντονα για μεγαλύτερες ακτίνες.

3.5.3.3 Picture



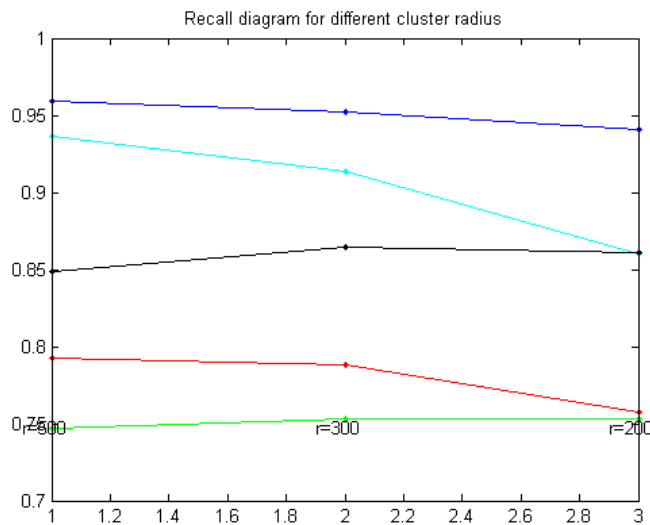
3.5.8 Η ευστοχία του συνόλου (μέση τιμή 10 εκτελέσεων) picture αντίστοιχα με πριν. Δεν συμπεριλαμβάνονται οι περιπτώσεις 1200 και 750 γιατί δεν έχουμε αρκετά μεγάλο σύνολο για αυτές

Για αυτό το σύνολο βλέπουμε πως και πάλι έχουμε σχετικά καλύτερη συμπεριφορά για την περίπτωση ακτίνας 500 προτύπων εκπαίδευσης σε σχέση με τις μικρότερες. Αντίθετα όμως με όλες τις άλλες περιπτώσεις (ακόμα και για δι-

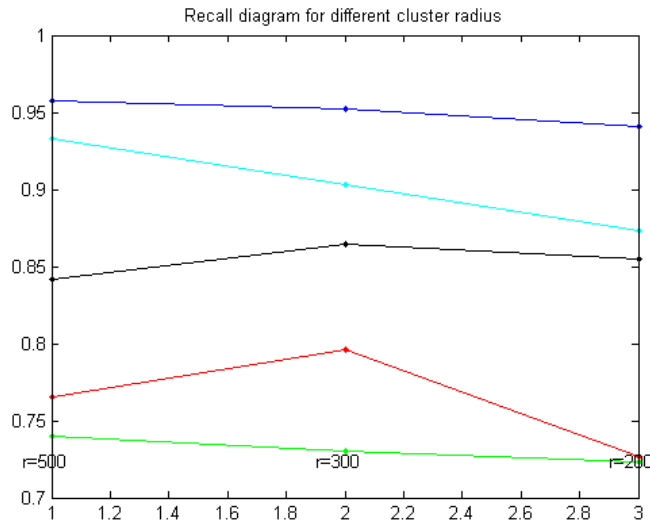
αφορετικά συστήματα) είναι η πρώτη φορά οπου το σχεπτικό της απλής επιλογής υπερτερεί τοσο ξεκάθαρα σε σχέση με αυτο του συνδυασμού επιλογών. Γενικά παντως θα πρέπει να αναφέρουμε για άλλη μια φορά πως το συγκεκριμένο σύνολο είναι αρκετά ασταθές στις αποκρίσει του και επομένως αυτη η συμπεριφορά ενδέχεται να μην προέκυπτε για παράδειγμα για μια διαφορετική μεση τιμή δεκα διαφορετικών εκτελέσεων.

Ανεξάρτητα απο αυτά παντως βλέπουμε πως και πάλι η συμπεριφορά των 2 γραφικών είναι σχετικά παρόμοια, καθώς αν εξαιρέσουμε την σταθερή υπεροχή της μεμονωμένης περίπτωσης κατα 1-2 μονάδες ,είναι αρκετά ίδιες.

Τέλος να σημειώσουμε οτι γενικά είμαστε και πάλι σχετικά ικανοποιημένοι απο το σύστημα τουλάχιστον απο αποψη απόδοσης , δεδομένου οτι αν και δεν είναι οι καλύτερες αποδόσεις που έχουμε δει απο σύστημα , είναι αρκετα κοντά σε αυτές άλλων συστημάτων που έχουμε εξετάσει.



3.5.9 Η recall του συστήματος για όλες τις κατηγορίες , για τις διαφορετικές ακτίνες που δοκιμάσαμε στην περίπτωση επιλογής μεμονωμένης συστάδας



3.5.10 Η recall του συστήματος για όλες τις κατηγορίες, για τις διαφορετικές ακτίνες που δοκιμάσαμε στην επιλογή με βάρος αντίστροφως ανάλογο της απόστασης σε όσες συστάδες ανήκει το δεδομένο πρότυπο ελέγχου.

Εδώ δεν έχουμε να παρατηρήσουμε κάτι καινούργιο πέρα από την πάρα πολύ μεγάλη ομοιότητα των κατηγοριών για τις 2 περιπτώσεις επιλογής, καθώς και την σχετικά μικρή διακύμανση του συνόλου τιμών για την κάθε κατηγορία με την αύξηση της ακτίνας η οποία συνήθως αποτελεί μια ελαφριά πτώση.

3.6. Σχολιασμός αποτελεσμάτων για τα πιο δύσκολα -για τους συγκεκριμένους ταξινομητές- σύνολα

Εκτός από τα τρία παραπάνω σύνολα , εφαρμόσαμε τους αλγόριθμους μας, καθώς και απλή περίπτωση νευρωνικού δικτύου σε ακόμα 2 σύνολα που βρήκαμε στο *UCI* (το σύνολο *abalone* και το σύνολο *contraceptive*). Και στις δύο περιπτώσεις , η περίπτωση μεμονωμένου νευρωνικού δικτύου , όπως φαίνεται δυσκολεύεται να ταξινομήσει σωστά τα πρότυπα. Όπως προκύπτει τελικά , το ίδιο ισχύει και για τους αλγόριθμους που φτιάξαμε εμείς.

Αναφορικά με το σύνολο *abalone* , ξέρουμε από τις πληροφορίες ότι πρόκειται για ένα έντονα συσχετισμένο σύνολο το οποίο περιέχει πολλές επικαλύψεις και στο οποίο για νευρωνικά δίκτυα , υπο ίδιες συνθήκες με αυτές που πραγματοποιούμε εμείς (ίδια πρότυπα για εκπαίδευση , ίδια κατηγοριοποίηση σε κλάσεις), το μέσο ποσοστό του δικτύου είναι 64 τοις εκατό. Αυτό το επαληθεύουμε και εμείς. Γενικά στο συγκεκριμένο σύνολο κρίνεται ότι πρέπει για να επιτευχθεί καλύτερο αποτέλεσμα να εφαρμοσθεί κάποιος επιπλέον χωρικός μετασχηματισμός (γεωμετρικής ίσως φύσης) έτσι ώστε να αποσυσχετισθούν παραπάνω τα δεδομένα αφού ίσως ο *raw* δεν κάνει για την συγκεκριμένη περίπτωση.(ή θα μπορούσε να συνδυασθεί με κάποιον άλλο μετασχηματισμό προεπεξεργασίας)

Για τα συστήματα που αναπτύξαμε , παρουσιάζουμε τα αποτελέσματα στο αρχείο *abalone results* . Συνολικά , όπως άλλωστε αναμενόταν , κάποιες περιπτώσεις τα πήγαν σχετικά καλύτερα από το απλό δίκτυο και κάποιες σχετικά χειρότερα. Πάντως η συντριπτική πλειοψηφία των διαφορετικών εκτελέσεων είχε ποσοστά περίπου 60-64 τοις εκατό. Επομένως γενικά δεν είδαμε μεγάλη διαφορά από την περίπτωση του απλού *mlp*.

Συγκριτικά με τους δικούς μας αλγόριθμους τα καλύτερα δυνατά αποτελέσματα τα είδαμε για τον αλγόριθμο του τμήματος 4 (*clusterisation* και κατόπιν εκπαίδευση συγκεκριμένου αριθμού *clusters*) αν και πρέπει να αναφερθεί ότι χρονικά ο αλγόριθμος ήταν αρκετά ικανοποιητικός μόνο για $K=1$. Μετά από αυτό , για μεγαλύτερα K καθυστέρουσε σε σχέση με το απλό δίκτυο. Από την άλλη , τα χειρότερα αποτελέσματα τα είχαν οι αλγόριθμοι των τμημάτων 2 και 3 όπου είχαμε απόδοση περίπου 60 τοις εκατό. Ο αλγόριθμος του τμήματος 5 ήταν περίπου στο ενδιάμεσο.

Από την άλλη μεριά στο σύνολο *contraceptive* παρατηρούμε ότι ο αλγόριθμος μπορεί να εκπαιδευθεί μέχρι ικανοποιητικές αποδόσεις για το σύνολο εκπαίδευσης του. Παρόλα αυτά όταν μεταφερόμαστε στο σύνολο ελέγχου, η απόδοση σε κάθε περίπτωση είναι απογοητευτική. Ειδικά μάλιστα εάν αφήναμε το σύστημα στην περίπτωση του απλού δικτύου να υπερεκπαιδευθεί, τότε τα

αποτελέσματα για το σύνολο ελέγχου ήταν τα χειρότερα δυνατά (περίπου 49 τοις εκατό κατα μέσο όρο για απλό mlp). Μια πιθανή σκέψη είναι ότι ίσως χρειάζονται περισσότερα πρότυπα για την εκπαίδευση του συστήματος και λιγότερα για τον έλεγχο και αυτό γιατί ίσως δεν έχουμε επαρκείς αποτυπώσεις για όλες τις δυνατές περιπτώσεις που καλύπτει όλο το σύνολο, πράγμα που σημαίνει ότι ίσως πολλές φορές στα δεδομένα ελέγχου, αρκετά πρότυπα είναι εκτός του χώρου που καλύπτουν τα δεδομένα εκπαίδευσης.

Μια ακόμα παρατήρηση είναι ότι για την περίπτωση του απλού mlp είχαμε και την μεγαλύτερη διακύμανση σε σχέση με τους δικούς μας αλγορίθμους για την καλύτερη περίπτωση εκπαίδευσης που μπορέσαμε να επιτύχουμε για αυτό (σύνολο εκπαίδευσης ποσοστό περίπου 65 τοις εκατό για να τερματισθεί η εκπαίδευση, το οποίο αντιστοιχούσε σε μέσο όρο περίπου 52 τοις εκατό για απλό mlp , το οποίο όμως σε κάποιες εκτελέσεις εφτανε τα 56.5 τοις εκατό, ενώ σε κάποιες άλλες το 47 τοις εκατό). Φυσικά για αυτό το σύνολο δεδομένου ότι δεν δίνονταν έτοιμα ποια πρότυπα είναι εκπαίδευσης και ποια είναι ελέγχου, εφαρμόζαμε εμείς τυχαίες μεταθέσεις, πράγμα που ενδέχεται εν μέρει να αιτιολογεί την μεγάλη μεταβλητότητα.

Σχετικά με τους δικούς μας αλγορίθμους, αρκετά καλά τα πήγε ο αλγόριθμος του τμήματος 5, ο οποίος στην πιο ισχυρή εκδοχή του, (χρήση πολλαπλών επικαλύψεων μέχρι να ικανοποιηθεί δεδομένο όριο) πέτυχε αρκετά υψηλότερα αποτελέσματα κατα μέσο όρο (περίπου 56 τοις εκατό) χωρίς ωστόσο να προσφέρει κάτι το πραγματικά εντυπωσιακό σε σχέση με την απλή περίπτωση. Ακόμα και αυτό παντως το αποτέλεσμα είχε μεγάλη χρονική καθυστέρηση σε σχέση με την απλή περίπτωση πράγμα που προφανώς είναι ανεπιθύμητο.

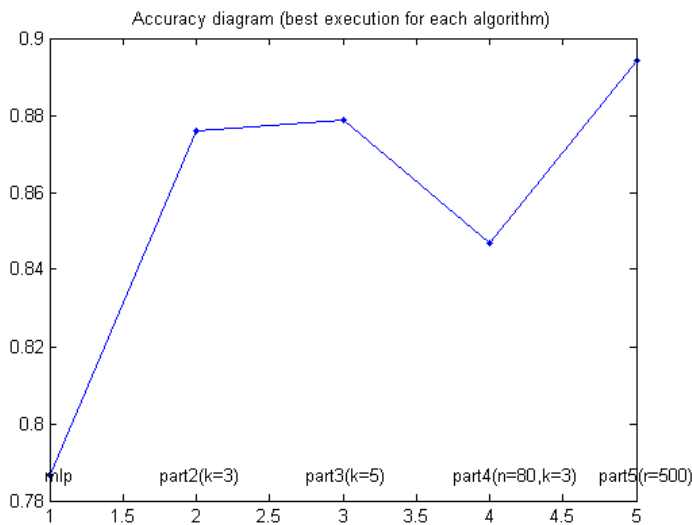
Πέραν αυτού, σχετικά καλά τα πηγαίνει και η πιο απλή περίπτωση του 5 (απλή επικάλυψη) για συγκεκριμένη ακτίνα και με σχετικά καλύτερο χρόνο, αλλά για χαμηλότερη απόδοση.

Εκτος από τους παραπάνω, οι υπόλοιποι αλγόριθμοι που δοκιμάσαμε δεν φαίνεται να τα πηγαίνουν ιδιαίτερα ικανοποιητικά (αποδόσεις παρόμοιες με την απλή περίπτωση ή χαμηλότερες), με εξαίρεση ίσως κάποιων αποτελεσμάτων του αλγόριθμου 3 για κάποια k στον knn .

Γενικά πρέπει να αναφερθεί ότι για τα σύνολα αυτά δεν εφαρμόστηκαν όλες οι υποπεριπτώσεις αλγορίθμων που εξετάστηκαν στα προηγούμενα παρα μόνο αυτές που είχαν τα στατιστικά καλύτερα αποτελέσματα από άποψη απόδοσης.

3.7. Συγκρίσεις/Τελικά αποτελέσματα

Παρακάτω παραθέτουμε γραφικές που αποτυπώνουν την καλύτερη απόπειρα για τον καθένα απο τους αλγορίθμους που χρησιμοποιήθηκαν για το κάθε σύνολο. Αναγράφουμε για το κάθε αποτέλεσμα την χαρακτηριστική του παραμετροποίηση.



3.7.1 Συγκριση της ευστοχίας για την καλύτερη προσπάθεια που καταγράφηκε απ' τον κάθε αλγόριθμο για το σύνολο letters

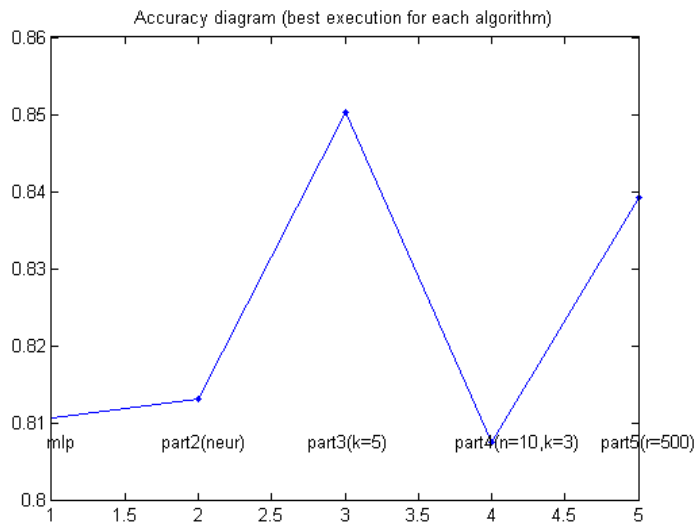
Παρατηρούμε αρχικά οτι για το συγκεκριμένο σύνολο (το οποίο είναι αρκετά μεγάλο) όλες οι μέθοδοι οι οποίες εφαρμόσαμε καθαρά απο άποψη συνολικής ευστοχίας υπερτερούν σε σχέση με την περίπτωση μεμονωμένου απλού νευρωνικού δικτύου. Αυτό είναι σχετικά αναμενόμενο δεδομένου οτι σε όλες τις περιπτώσεις μεθόδων τις οποίες εξετάζουμε χρησιμοποιούμε ομάδα νευρωνικών δικτύων και έτσι δεδομένου του μεγέθους του συγκεκριμένου συνόλου, είναι λογικό οτι μια σχετικά αποτελεσματική αξιοποίηση αυτης της ομάδας θα φέρει σχετικά καλύτερα αποτελέσματα και πιο γρήγορα.

Αναφορικά με την απόδοση για τις διάφορες μεθόδους που χρησιμοποιήσαμε, παρατηρούμε οτι για το συγκεκριμένο σύνολο έχουμε την καλύτερη για την μεθοδο του τμήματος 5, πράγμα λογικό δεδομένου οτι αυτη προορίζεται για μεγάλα σύνολα λόγω της ικανότητάς της να τμηματοποιεί/μοιράζει σε πολλά επι

μέρους δίκτυα το σύνολο εκπαίδευσης σχετικά αποδοτικά και να χρησιμοποιεί πολλά δίκτυα κατά την εκπαιδευτική διαδικασία της.

Είναι αρκετά εμφανές επίσης ότι η βέλτιστη περίπτωση για τη μέθοδο του τμήματος 4 υπολείπεται κάπως σε σχέση με τις υπόλοιπες. Από την άλλη μεριά, οι τεχνικές των τμημάτων 2 και 3 τα πάνε αρκετά καλά με περίπου ίδια βέλτιστη απόδοση περίπου στο 88 τοις εκατό. Συνολικά πάντως όλες οι απόδοσεις είναι ικανοποιητικές ειδικά σε σχέση με την περίπτωση απλού δικτύου.

Σε σχέση με τους χρόνους για το συγκεκριμένο σύνολο παρατηρούμε (οι χρόνοι είναι για την μέθοδο του 2 130 για το 3 420 για το 4 220-240 και για το 5 270 sec), ότι με διαφορά η πιο γρήγορη μέθοδος τουλάχιστον για το σύνολο είναι η τεχνική του 2, πράγμα το οποίο πιστοποιεί ότι για μεγάλα σύνολα όπως το συγκεκριμένο η μέθοδος 'διαίρει και βασίλευε' στην οποία βασίζεται η τεχνική αυτή επιταχύνει τις διαδικασίες αρκετά. Επίσης ικανοποιητικούς χρόνους έχουν και οι τεχνικές του 4 και του 5, εφόσον κατανέμουν τον υπολογισμό σε μικρότερα δίκτυα. Αντίθετα, η περίπτωση του 3 δεν τα πηγαίνει καθόλου καλά από άποψη χρόνου λόγω του knn που πραγματοποιεί μετά από την εφαρμογή της διαδικασίας του *boosting*, καθώς και λόγω της καθυστέρησης στο πρώτο δίκτυο το οποίο έχει ουσιαστικά να εκπαιδευτεί με το σύνολο των δεδομένων εκπαίδευσης.



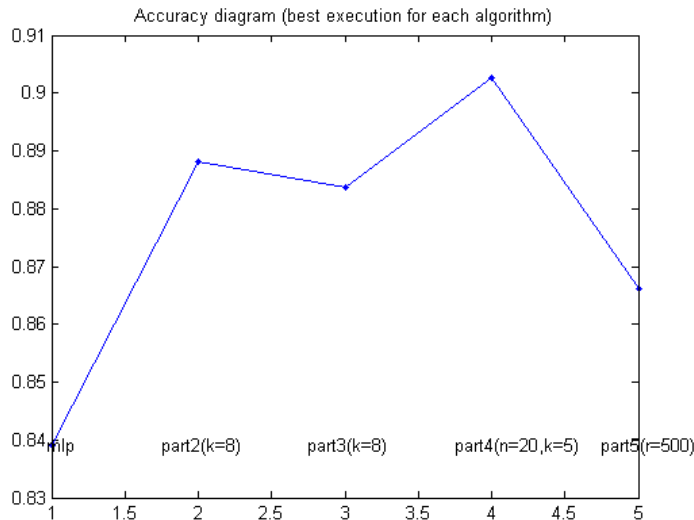
3.7.2 Συγκριση της ευστοχίας για την καλύτερη προσπάθεια που καταγράφηκε απ' τον κάθε αλγόριθμο για το σύνολο CTG

Σχετικά με το σύνολο *CTG* βλέπουμε ότι οι 2 μέθοδοι μας (οι 3 και 5) τα πάνε καλά -καλύτερα από την βέλτιστη περίπτωση για *mhp*-, ενώ οι άλλες δύο έχουν περίπου αντίστοιχη απόδοση με αυτή του απλού δικτύου. Αυτό και για τις δύο περιπτώσεις του 2 και του 4 μπορεί ίσως να αιτιολογηθεί από τον μεγάλο αριθμό κλάσεων που έχουμε στο συγκεκριμένο σύνολο.

Όσον αφορά το 2 λόγω της ιεραρχίας που χρησιμοποιήσαμε στο σύστημα μας, η οποία είχε μόνο 2 επίπεδα (έφτασε μέχρι *C1* για τρεις υπερκλάσεις, κάθε μία από τις οποίες εμπεριείχε επίσης 3 κλάσεις). Δεδομένου του μεγάλου αριθμού κατηγοριών, θα ήταν καλύτερα αν είχαμε μεγαλύτερη ιεραρχία και αυτό ίσως αποτυπώνεται στο αποτέλεσμα.

Σε σχέση με το 4, δεδομένου ότι έχουμε προπροσδιορισμένο αριθμό δικτύων και μεγάλο αριθμό κατηγοριών οι οποίες μάλιστα δεν έχουν ισοκαταναμημένο αριθμό πρότυπων είναι πολύ πιθανό τα επι μέρους δίκτυα του συστήματος να μην εκπαιδεύονται ικανοποιητικά άφου κάποια ενδέχεται να μην έχουν πρότυπα για ορισμένες κατηγορίες.

Από άποψη χρόνου, παρατηρούμε ότι (είναι για το 2: 20-30sec για το 3:40-50sec για το 4: 40-60sec και για το 5:60-90sec) και πάλι η περίπτωση του 2 υπερτερεί χρονικά σε σχέση με τις άλλες μεθόδους. Αυτήν τη φορά η μέθοδος του 3 τα πάει καλύτερα σε σχέση με τα άλλα σύνολα και αυτό μπορεί να αιτιολογηθεί από το ότι ο *knn* αυτή τη φορά δεν καθυστερεί ιδιαίτερα την διαδικασία εκπαίδευσης-ελέγχου, εφόσον, αν και έχει μεγάλη σχετικά πολυπλοκότητα, (και άρα σε μεγαλύτερα σύνολα όντως καθυστερεί αρκετά), το σύνολο δεν είναι τόσο μεγάλο. Όσον αφορά τις μεθόδους του 4 και του 5, παρατηρούμε ότι καθυστερούν, αν και σε ανεκτά πλαίσια σε σχέση με τις προαναφερθείσες μεθόδους. Αυτό μπορεί να αιτιολογηθεί από το ότι οι μέθοδοι αυτές χρησιμοποιούν πολλά δίκτυα και αν και αυτό μπορεί να επισπεύσει την διαδικασία εκπαίδευσης για μεγάλα δίκτυα, είναι προφανές ότι σε μικρότερα μπορεί να καθυστερεί κάπως. Δεδομένου όμως ότι έχουμε μικρότερη πολυπλοκότητα για αυτές τις μεθόδους από ότι για τη μέθοδο του 3 και εφόσον οι χρόνοι έχουν πιο μεγάλη σημασία για τα μεγαλύτερα σύνολα όπου καταγράφονται και οι πιο ουσιώδεις καθυστερήσεις, αυτό δεν μας πειράζει ιδιαίτερα.



3.7.3 Συγκριση της ευστοχίας για την καλύτερη προσπάθεια που καταγράφηκε απ' τον κάθε αλγόριθμο για το σύνολο picture

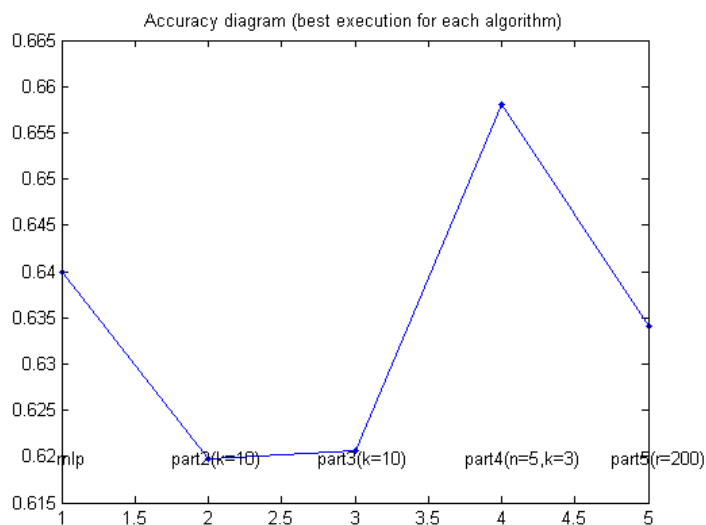
Για το σύνολο αυτο, το οποίο είναι και το μικρότερο και το πιο ασταθές, παρατηρούμε ότι από άποψη απόδοσης και πάλι οι τεχνικές που εφαρμόσαμε επιτυγχάνουν πολύ ικανοποιητικά αποτελέσματα. Από τις τεχνικές μας αυτή η οποία έχει την χειρότερη απόδοση, αν και ακόμα ικανοποιητική είναι αυτή του τμήματος 5. Αυτό είναι εντελώς αναμενόμενο δεδομένου ότι η μέθοδος αυτή έχει την τάση να χρησιμοποιεί τον μεγαλύτερο αριθμό δικτύων και δεδομένου ότι έχουμε πολύ μικρό σύνολο εκπαίδευσης, είναι αναμενόμενο να μην τα πηγαίνει τόσο καλά γιατί τα πρότυπα δεν επαρκούν για όλα τα δίκτυα.

Η απόδοση η οποία παρατηρείται για το τμήμα 4 είναι όντως εντυπωσιακή, δεδομένου του μεγάλου πλήθους δικτύων που χρησιμοποιείται. Αν όμως αναλογισθούμε ότι το K είναι 5 και άρα, κάθε πρότυπο εκπαίδευσης αποτελεί μέλος 5 δικτύων και επομένως αν και έχουμε την εκπαίδευση κάθε δικτύου μία φορά, στην πραγματικότητα χρησιμοποιούμε 5 φορές το σύνολο εκπαίδευσης, μπορούμε να καταλάβουμε ότι η απόδοση αυτή αιτιολογείται, αφού έτσι τα δίκτυα μας έχουν αρκετό υλικό για να εκπαιδευθούν καλύτερα.

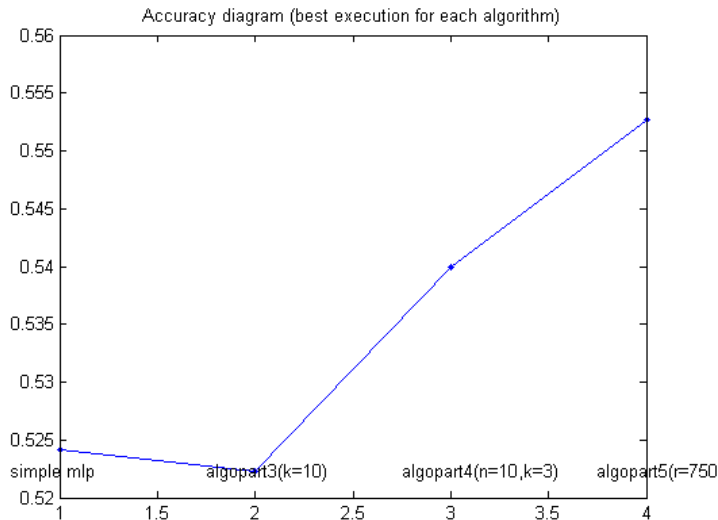
Η μέθοδος του 3 για άλλη μια φορά τα πηγαίνει ικανοποιητικά και παρατηρούμε ότι έχει η βέλτιστη περίπτωση που καταγράφηκε είναι για μεγάλη σχετικά γειτονία στον knn .

Στην περίπτωση του 2, η απόδοση είναι και πάλι ιδιαίτερα ικανοποιητική

και εφόσον ο αριθμός των κλάσεων του συγκεκριμένου συνόλου είναι 6, η διεπίπεδη δομή επαρκεί για να καλύψει τις ανάγκες του συνόλου.



3.7.4 Συγκριση της ευστοχίας για την καλύτερη προσπάθεια που καταγράφηκε απ' τον κάθε αλγόριθμο για το σύνολο *abalone*



3.7.5 Συγκριση της ευστοχίας για την καλύτερη προσπάθεια που καταγράφηκε απ' τον κάθε αλγόριθμο για το σύνολο *contraceptive*

Για τα παραπάνω σύνολα παρατηρούμε πως συνολικά, για όλες τις περιπτώσεις -συμπεριλαμβανομένης της περίπτωσης απλού *mlp*- έχουμε σχετικά χαμηλή ευστοχία. Ειδικά στο σύνολο *abalone* παρατηρούμε ότι έχουμε 3 από τις 4 περιπτώσεις μας να υπολείπονται ως προς την απόδοση σε σχέση με το απλό *mlp* και ότι συνολικά οι αποδόσεις δεν ξεπερνάνε το 65 τοις εκατό. Γενικά προκειται για ένα έντονα συσχετισμένο σύνολο στο οποίο υπάρχουν πολλές επικαλύψεις (αναφέρεται στην παρουσίαση του συνόλου). Έτσι, η εκπαιδευτική διαδικασία δεν μπορεί να ξεπεράσει κάποιο όριο τουλάχιστον για τα νευρωνικά δίκτυα. Επομένως αυτό είναι ένα συνολικό όριο (το 65 με 66 τοις εκατό) το οποίο δεν μπορεί να ξεπεραστεί από το νευρωνικό δίκτυο σαν ταξινομητή συνολικά, εκτός αν εφαρμοσθεί κάποιος κατάλληλος μετασχηματισμός πριν την εκπαίδευση στον χώρο των προτύπων.

Σχετικά με την περίπτωση του *contraceptive* τα πράγματα είναι κάπως πιο περίπλοκα. Εδώ, η εκπαιδευτική διαδικασία πραγματοποιείται ικανοποιητικά και τα δίκτυα μπορούν να φτάσουν ικανοποιητικές αποδόσεις στο σύνολο εκπαίδευσης. Παρόλα αυτά στην φάση του ελέγχου, αν και όπως παρατηρούμε οι μέθοδοι μας τα πάνε γενικά καλύτερα από την χρήση απλού δικτύου, η απόδοση σε όλες ανεξαιρέτως τις περιπτώσεις είναι πολύ χαμηλή (περίπου 50-60 τοις εκατό). Μια πιθανή αιτία ίσως σχετίζεται με υψηλή επικάλυψη σε ορισ-

μένα χαρακτηριστικά. Με αυτό , ύστερα απο κάποια εκπαίδευση τα δίκτυα θα μπορούσαν να υπερεκπαιδευθούν και να μάθουν τις κλασσικες μεμονωμένων προτύπων εκπαίδευσης (και έτσι να αυξηθεί η ευστοχία κατα την εκπαίδευση), αλλα θα υπάρχει 'συγχυση' κατα την φάση του ελέγχου.

4. Επίλογος/Προτεινόμενες επεκτάσεις

Συνολικά στην παρούσα εργασία μελετήσαμε διαφορετικές μεθόδους αποτελεσματικότερης αξιοποίησης νευρωνικών δικτύων στα πλαίσια μίας ομάδας απο τέτοιους ταξινομητές.

Απο τα αποτελέσματα της προηγούμενης ενότητας βγάζουμε το συμπέρασμα ότι ανάλογα με το σύνολο εκπαίδευσης και τις διαφορετικές απαιτήσεις/ χαρακτηριστικά που αυτό έχει ,ενδέχεται να αλλάζει το ποιος αλγόριθμος είναι καταλληλότερος για την ταξινόμηση του. Πάντως απο τις διάφορες περιπτώσεις τις οποίες εξετάσαμε , παρατηρήσαμε ότι την πιο αξιόπιστη συνολικά συμπεριφορά την έχει η μέθοδος της ενότητας 5 του πρακτικού μέρους , αν και υστερούσε χρονικά στις περιπτώσεις μικρών συνόλων. Αντίθετα , η περίπτωση του απλού ανεξάρτητου *mlp* αν και προφανώς είναι η απλούστερη , στα περισσότερα συνολα που εξετάστηκαν , υστερούσε απο άποψη απόδοσης σε σύγκριση με τις υπόλοιπες μεθόδους.

Συμπερασματικά λοιπον μπορούμε με αρκετή βεβαιότητα να πούμε πως ειδικά για πιο μεγάλα και πιο σύνθετα σύνολα (για παράδειγμα με περισσότερες κατηγορίες ή σχετικά ανισοκατανομημένα στον χώρο των χαρακτηριστικών) η χρήση μιας ομάδας δικτύων συνιστάται τοσο απο πλευράς απόδοσης όσο και απο πλευράς χρόνου, εφόσον όμως μπορούμε να την σχηματίσουμε ετσι ωστε να είναι αποτελεσματική για την επίλυση του συγκεκριμένου συνόλου-αν πχ έχουμε ένα σύνολο με σχετικά λίγα πρότυπα , αλλά στο οποίο υπάρχουν πάρα πολλές διαφορετικές κλάσεις η μέθοδος του τμήματος 2 ίσως θα πρέπει να προτιμηθεί απο κάποια άλλη μέθοδο -.

Φυσικά , υπάρχουν και περιπτώσεις στις οποίες ο ταξινομητής του νευρωνικού δικτύου δεν συμπεριφέρεται αναμενόμενα , είτε λόγω υψηλής συσχέτισης των δεδομένων , είτε εξαιτίας άλλων αιτιών. Σε τέτοιες περιπτώσεις , οι μέθοδοι που εξετάσαμε ,δεδομένου ότι στον πυρήνα τους όλες χρησιμοποιούν τον συγκεκριμένο ταξινομητή δεν αποδίδουν και αυτές όπως θα περιμέναμε. Σε τέτοιες περιπτώσεις η καλύτερη ιδέα είναι η χρήση ενός άλλου διαφορετικού ταξινομητή για την κατηγοριοποίηση, ή ακόμα θα μπορούσε να δοκιμασθεί κάποιος χωρικός μετασχηματισμός για τον επαναπροσδιορισμό των δεδομένων και ατόπιν η χρήση κάποιας απο τις μεθόδους.

Τέλος, αναφορικά με τις πιθανές επεκτάσεις οι οποίες μπορούν να πραγματοποιηθούν , όπως προκύπτει και απο το πρακτικό τμήμα της εργασίας ,είναι αρκετά ποικιλόμορφες και ετερόκλητες. Δεδομένου ότι απο την φύση του

το θέμα της εργασίας είναι αρκετά γενικό ,και καλύπτει μεγάλο εύρος , είναι κατανοητό οτι οι επιλογές για αλλαγές και περαιτέρω πειραματισμό είναι πραγματικά πολλές.

Βιβλιογραφία/Αναφορές:

- 1.Simon Haykin. Νευρωνικά Δίκτυα και Μηχανική Μάθηση
- 2.url: <http://blog.peltarion.com/2006/07/10/classifier-showdown/>
- 3.Stanford University coursera machine learning description
4. Ron Kohavi; Foster Provost (1998). "Glossary of terms". Machine Learning 30: 271–274.
5. An Introduction to Support Vector Machines and Other Kernel-based Learning By Nello Cristianini, John Shawe-Taylor
6. A Tutorial on Support Vector Machines for Pattern Recognition CHRISTOPHER J.C. BURGES burges@lucent.com Bell Laboratories, Lucent Technologies
- 7.University of Cambridge Engineering Part IIB Module 4F10: Statistical Pattern Processing Handout 6: Multi-Layer Perceptrons
- 8.A Detailed Introduction to K-Nearest Neighbor (KNN) Algorithm
- 9.Machine learning: An artificial intelligence approach
- 10.Scholarpedia/kohonen network(Dr. Teuvo Kohonen, Academy of Finland Dr. Timo Honkela, Helsinki University of Technology)
- 11.SOMs for machine learning
- 12.Mixture models (1-5) Victor Lavrenko online lectures university of Endinburg
- 13.The Expectation Maximization Algorithm A short tutorial 2004
- 14.EM-algorithm for motif discovery - University of California
- 15.Support Vector Machine (SVM)<https://www.sec.in.tum.de/assets/lehre/ws0910/ml/slideslecture8.pdf>
- 16.Caltech online courses Radial Basis Function
- 17.URL: <http://www.cs.ucr.edu/~eamonn/CE/Bayesian20Classification20withInsect-examples.pdf>
- 18.An Empirical Study of the naive Bayes Classifier Irina Rish -IBM
- 19.Naive Bayes Victor Lavrenko online lectures university of Endinburg
- 20.Popular Ensemble Methods: An Empirical Study David Opitz Department of Computer Science University of Montana Missoula, MT 59812 USA opitz@cs.umt.edu Richard Maclin maclin@d.umn.edu Computer Science Department University of Minnesota Duluth, MN 55812 USA rmaclin@d.umn.edu
- 21.International Journal of Computer Trends and Technology (IJCTT) Ensemble Classifiers and Their Applications: A Review Akhlaqur Rahman and Sumaira Tasnim

22. <https://www.stat.berkeley.edu/breiman/RandomForests/cc-home.html>
23. Scholarpedia/ Ensemble Learning/ Boosting (Dr. Robert Polikar, Rowan University)
24. AdaBoost and the Super Bowl of Classifiers A Tutorial Introduction to Adaptive Boosting Ra/ul Rojas Computer Science Department Freie Universität at Berlin Christmas 2009
25. matlab manual pages

Επίσης , χρησιμοποιήθηκαν τα προγράμματα:

Matlab
draw.io
latex