Αναφορά 1ης Εργαστηριακής Άσκησης Νευρωνικά Δίκτυα

Κωνσταντίνος Καλλάς 03112057
Τζίνης Ευθύμιος 03112007

9 Δεκεμβρίου 2016

1 Εισαγωγή

Σε αυτή την εργαστηριακή άσκηση πειραματιστήκαμε με το περιβάλλον της *Matlab* για την υλοποίηση ενός Νευρωνικού Δικτύου και την εφαρμογή διαφορετικών παραμέτρων κατά την εκπαίδευσή του. Τα δεδομένα που χρησιμοποιήθηκαν για το *training* και το *testing* έχουν δοθεί από την ιστοσελίδα του μαθήματος και αποτελούνται από κωδικοποιημένα χαρακτηριστικά εικόνων. Η κεντρική δομή της αναφοράς αυτής είναι η εξής: Αρχικά παραθέτουμε κάποιες πολιτικές που ακολουθήσαμε κατά την υλοποίηση της άσκησης σύμφωνα με τα αντίστοιχα βήματα που αναφέρονται και στην ίδια την άσκηση. Εν συνεχεία, παραθέτουμε τις απαντήσεις στις αντίστοιχες ερωτήσεις της εκφώνησης της άσκησης. Προκειμένου να καταφέρουμε να έχουμε μία πιο εύλυπτη αναφορά, κατάλληλα διαγράμματα με αναλύσεις αυτών παραθέτονται κάτω από κάθε αντίστοιχη ερώτηση.

Σημαντικές Παραδοχές: Ως μετρική για την επίδοση του συστήματός μας έχουμε επιλέξει το *accuracy* διότι είδαμε μετά από κάποια αρχικά τρεξίματα ότι αυτή έχει το μικρότερο *variance* στις τιμές που μας δίνει. Επιπλέον, επειδή είναι τυχαίο και το χώρισμα των δεδομένων που θα εκπαιδεύσουμε το σύστημά μας αλλά και η αρχικοποίηση των βαρών στο Νευρωνικό Δίκτυο τα αποτελέσματα από κάθε παραμετροποίηση μπορεί να βγάζουν κάποιες τιμές που έχουν μεγάλη απόκλιση μεταξύ τους. Γι αυτό τον λόγο τρέξαμε κάθε παραμετροποίηση 5 φορές και πήραμε τον μέσο όρο αυτών στην μετρική μας.

Κώδικας: Ο κώδικάς μας βρίσκεται στον φάκελο *source* και είναι διαχωρισμένος σε κατάλληλα *scripts* που αντιστοιχούν με τα βήματα της άσκησης.

2 Βήματα

Ο σκοπός των βημάτων της άσκησης ήταν ο πειραματισμός με διαφορετικές παραμέτρους του νευρωνικού δικτύου και η εύρεση κάποιων συνδυασμών παραμέτρων με τους οποίους επιτυγχάνεται το καλύτερο αποτέλεσμα. Σε κάθε βήμα πειραματιστήκαμε με μία παράμετρο θεωρώντας δεδομένη την υπόλοιπη παραμετροποίηση. Τα αποτελέσματα από κάθε βήμα παρουσιάζονται στις ερωτήσεις.

Μετά από όλα τα βήματα καταλήξαμε στον καλύτερο συνδυασμό όσον αφορά την επιτυχία ταξινόμησης, τη δυνατότητα γενίκευσης και το χρόνο εκπαίδευσης. Οι παράμετροι που επιλέξαμε είναι οι παρακάτω.

- Συνάρτηση εκπαίδευσης: $traingdx$. Επιλέξαμε την $traingdx$ γιατί πετυχαίνει υψηλά ποσοστά επιτυχίας στην ταξινόμηση, εκπαιδεύεται γρήγορα και είναι πολύ ευέλικτη λόγω του $adaptivelearningrate$
- Νευρώνες 1ου - 2ου επιπέδου (20 - 15). Επιλέξαμε αυτό το συνδυασμό νευρώνων γιατί με αυτόν πετύχαμε τα καλύτερα ποσοστά ταξινόμησης.
- $EarlyStopping$: Βελτιώνει την ταχύτητα εκπαίδευσης αλλά και τη δυνατότητα γενίκευσης.
- Συνάρτηση ενεργοποίησης εξόδου: $purelin$. Η συνάρτηση ενεργοποίησης $purelin$ μας έδωσε τα καλύτερα αποτελέσματα σε συνδυασμό με την $traingdx$.

Σημειώνεται ότι συνολικά τα αποτελέσματα ταξινόμησης είχαν μεγάλη διακύμανση και εξαρτώνται πολύ από την τυχαία αρχικοποίηση του νευρωνικού. Για αυτό το λόγο στις μετρήσεις μας χρησιμοποιήσαμε το μέσο όρο των αποτελεσμάτων από πολλές εκτελέσεις.
Με τον συνδυασμό που περιγράψαμε παραπάνω καταφέραμε να πετύχουμε

$$Classification Accuracy = 93.46\%$$

.

3 Ερωτήσεις

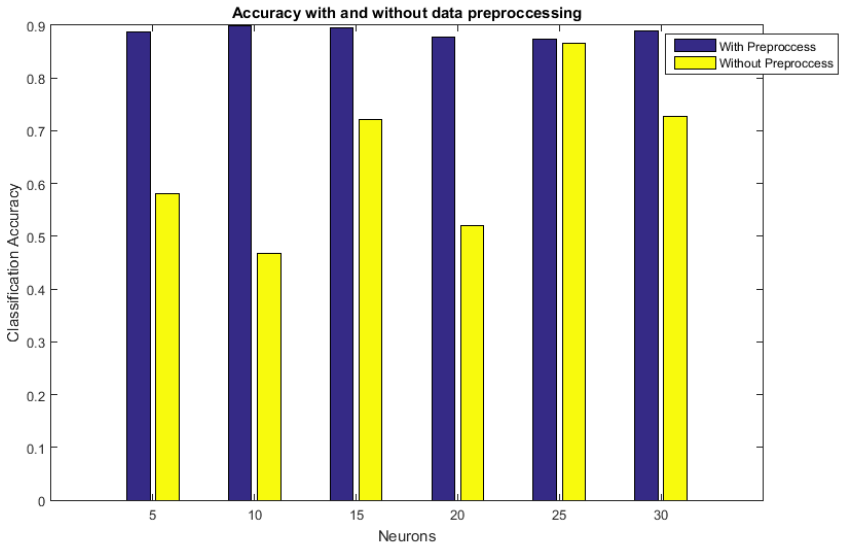
Ερώτηση 1

Για την προ-επεξεργαία των δεδομένων ακολουθήσαμε τα παρακάτω βήματα:

- *Equal Data For Training over All Classes* : Προκειμένου να μπορέσουμε να εκπαιδεύσουμε το δίκτυό μας με *δίκαιο* τρόπο για όλες τις 5 κλάσεις μας, βρήκαμε την κλάση που έχει τα λιγότερα δείγματα και κρατήσαμε τον ίδιο αριθμό δειγμάτων και για τις άλλες κλάσεις.
- *Remove Constant Rows* : Για όλα τα δείγματα ανεξάρτητα από σε ποιά κατηγορία ανήκουν βλέπουμε ότι για τα αρχικά 535 χαρακτηριστικά από κάθε δείγμα, διώχνουμε όλα αυτά τα χαρακτηριστικά που έχουν την ίδια τιμή για όλα τα δείγματά μας. Αυτά τα χαρακτηριστικά έχουν ουσιαστικά μηδενική συνδιακύμανση στον χώρο όλων των χαρακτηριστικών και άρα δεν συνεισφέρουν σε τίποτα στο να βοηθήσουν το δίκτυό μας στο $classification$ εν αντιθέσι προσθέτουν υπολογιστική πολυπλοκότητα στο μοντέλο μας και πολλές φορές οδηγούν σε χαμηλότερη επίδοση.
- *Principal Component Analysis* : Αυτή η μέθοδος έχει επικρατήσει στην βιβλιογραφία γύρω από τα Νευρωνικά Δίκτυα εξαιτίας του ότι είναι εφικτό να μειώσουμε σημαντικά τις διαστάσεις του προβλήματός μας όταν προβάλλουμε την πληροφορία των $N - space$ διανυσμάτων με χαρακτηριστικά σε

κάποιους άλλους ορθογώνιες συνιστώσες, γύρω από τις οποίες τα διανύσματά μας έχουν πολύ μεγαλύτερη συνδιακύμανση. Παίρνοντας λιγότερες συνιστώσες μειώνουμε την πολυπλοκότητα του προβλήματός μας και ταυτόχρονα φτιάχνουμε δείγματα με πιο αντιπροσωπευτικά χαρακτηριστικά.

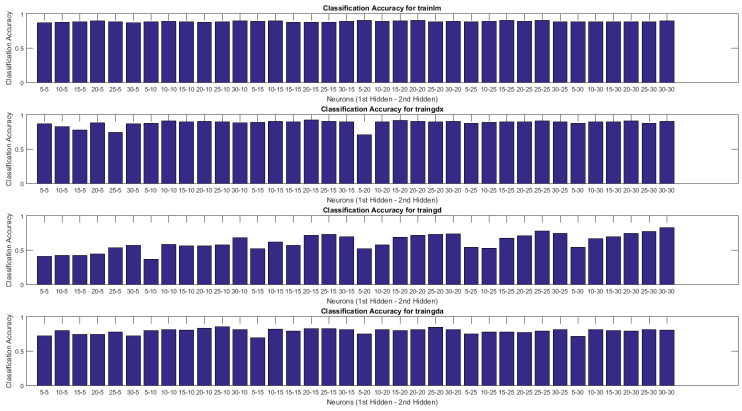
Προφανώς και θα ήταν εφικτό να μην κάνουμε όλη αυτήν την προεπεξεργασία για να εκπαιδεύσουμε το σύστημά μας όμως αυτό δεν θα έδινε τα επιθυμητά αποτελέσματα και επίσης θα ήταν πολύ πιο ακριβό από υπολογιστικης άποψης. Παρακάτω στην εικόνα 1 παραθέτουμε έναν διάγραμμα που αναδεικνύει αυτή την διαίσθησή μας που εξάγεται ως αποτέλεσμα της επανάληψης του βήματος 4 για μη προ επεξεργασμένα δεδομένα.



Σχήμα 1: Ταξινόμηση με και χωρίς προεπεξεργασία δεδομένων

Ερώτηση 2

Στην εικόνα 2 παρατηρούμε ότι το καλύτερο $configuration$ το πήραμε για 2 Επίπεδα Νευρώνων με το 1ο επίπεδο να έχει 20 Νευρώνες και το 2ο επίπεδο να έχει 15 Νευρώνες. Επιπλέον, οι συναρτήσεις εκπαίδευσης $traingdx$, $trainlm$ είχαν πάρα πολύ αντίστοιχα αποτελέσματα. Επιλέγουμε την πρώτη καθώς είχε λίγο καλύτερα αποτελέσματα ενώ επίσης συνδυάζει όχι μόνο το $adaptive - learning$ μειώνοντας το $learningrate$ με την πάροδο του χρόνου αλλά έχει και την έννοια του $momentum$ προκειμένου να μπορεί να συγκλίνει καλύτερα σε περιοχές που τα βήματα είναι πολύ μεγάλα εξαιτίας του επιλεγμένου $learningrate$. Προφανώς από τα αποτελέσματά μας βλέπουμε ότι δεν παίζει ρόλο μόνο ο αριθμός των νευρώνων αλλά και η τοπολογία του δικτύου μας που είναι προτιμότερο (στις περισσότερες περιπτώσεις) να έχει περίπου ίδιο αριθμό νευρώνων μεταξύ των επιπέδων. Φυσικά τόσο η συνάρτηση εκπαίδευσης που δείχνει να είναι καλύτερη όσο και η τοπολογία του δικτύου μας εξαρτάται κυρίως από την φύση των χαρακτηριστικών μας και γι αυτό δεν θα ήταν σωστό να εξάγουμε κάποιο γενικό συμπέρασμα μόνο από την εφαρμογή σε ένα απλό $image - classification\ task$.



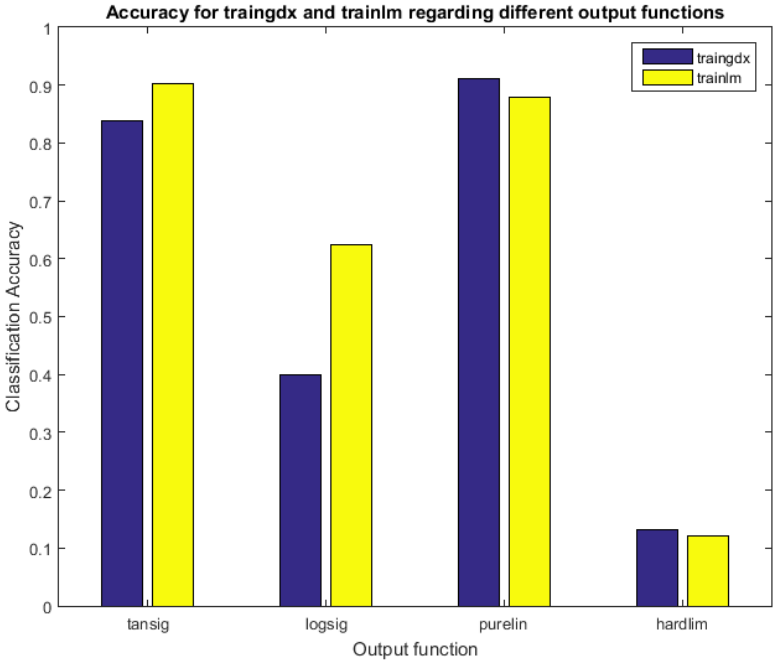
Σχήμα 2: Ταξινόμηση με διάφορες μεθόδους μάθησης και διαφορετικές διατάξεις νευρώνων πρώτου και δευτέρου επιπέδου

Ερώτηση 3

Επιλέξαμε την συνάρτηση ενεργοποίησης $purelin$ η οποία είδαμε ότι καταφέρνει να έχει τα καλύτερα αποτελέσματα σε συνδυασμό με τον τρόπο εκπαίδευσης $traingdx$. Παρατηρούμε ότι και άλλες συναρτήσεις ενεργοποίησης που χρησιμοποιούνται συνήθως έχουν διαφορετική συμπεριφορά. Όπως για παράδειγμα η $tansig$ η οποία αποδεικνύει ότι επιφέρει τα καλύτερα αποτελέσματά της σε συνδυασμό με την επιλογή της $trainlm$ για την ανανέωση των βαρών. Λογικά ένας αλγόριθμος ανανέωσης βαρών που βασίζει την ανανέωση των βαρών στην απλή παραγώγηση τότε θα ήταν προτιμότερο οι συναρτήσεις ενεργοποίησης να έχουν σταθερή παράγωγο και η απόφασή τους να επηρεάζεται μόνο από κάποιε σταθερές. Σε αντίθεση η συνάρτηση ενεργοποίησης της $tansig$ έχει διαφορετική τιμή στην παράγωγο για κάθε έξοδο που την κάνει καταλληλότερη σε σχέση με τον αλγόριθμο ανανέωσης $Levenberg - Marquardt$ ο οποίος συγκλίνει πολύ πιο γρήγορα εξαιτίας της δυνατότητας επιλογής κατάλληλης σταθεράς μ. Όταν αυτή η σταθερά γίνει πολύ μεγάλη τότε η μέθοδος ταυτίζεται με το $Gradient - Decent$ ενώ όταν είναι κοντά στο 0 τότε ο πίνακας της ανανέωσης βαρών γίνεται ένας $Hessian$ πίνακας και η μέθοδος γίνεται ίδια με την $Newton$ μέθοδο. Προκειμένου να αλλάξουμε μέσω της $newff$ την επιλεγμένη συνάρτηση εξόδου αρκεί να αλλάξουμε την κατάλληλη παράμετρο. Θα μπορούσαμε και χειροκίνητα αλλά θα έπρεπε να πάρουμε την έξοδο του νευρωνικού και να την ενώσουμε με κάποια άλλη συνάρτηση. Παραθέτουμε στην εικόνα 3 το διάγραμμα που δείχνει για κάθε συνάρτηση ενεργοποιησης και για τις δύο διαφορετικές μεθόδους $training$.

Ερώτηση 4

Παρατηρούμε ότι για τον απλό αλγόριθμο $traingd$ που αλλάζει τα βάρη με βάση το $gradient\ descend$ είναι τις περισσότερες φορές πολύ αργός για να συγκλίνει αν δεν επιλεγεί η κατάλληλη πολλαπλασιαστική σταθερά (ή όχι σταθερά αν είναι η μέθοδος $adaptive\ learning$ του $learning - rate$. Για αυτό τον λόγο χρειαζόμαστε περισσότερες εποχές προκειμένου να συγκλίνει. Επιπλέον, όταν επιλέγουμε να



Σχήμα 3: Ταξινόμηση με διάφορες μεθόδους μάθησης και διαφορετικές διατάξεις νευρώνων πρώτου και δευτέρου επιπέδου

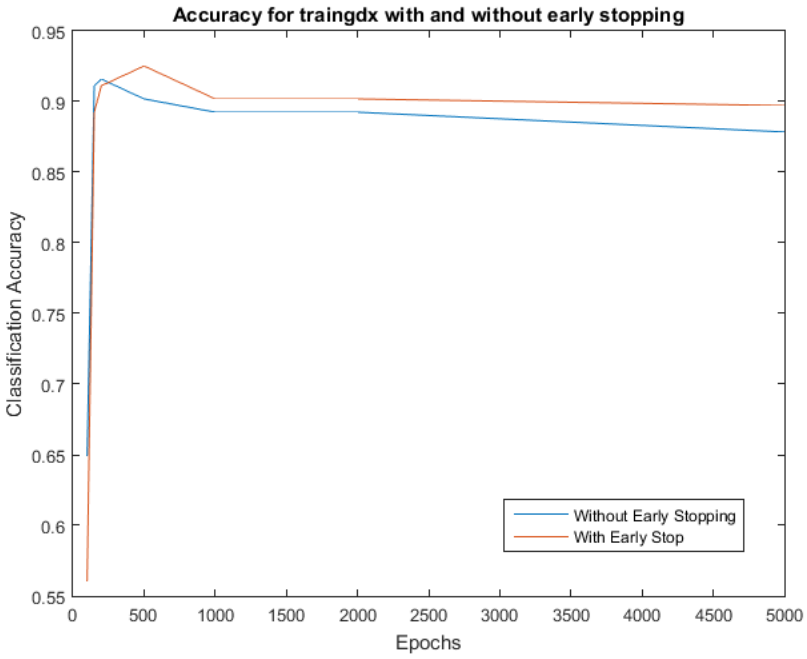
ανανεώσουμε τα βάρη μας με βάση την *traingdm* προσθέτουμε και έναν όρο που εξαρτάται από την σταθερά του *momentum* προκειμένου να μην μπορούμε να ξεφύγουμε υπερβολικά πολύ από την προηγούμενη τιμή του βάρους. Βλέπουμε ότι αν επιλέξουμε έναν επαρκή αριθμό από εποχές και την καλύτερη αρχιτεκτονική ο 2ος αλγόριθμος έχει σαφέστατα καλύτερα αποτελέσματα 90.84% *accuracy* αντί 79.63% της πρώτης μεθόδου.

Ερώτηση 5

Με την μέθοδο του *Early Stopping* καταφέρνουμε να αποφύγουμε το φαινόμενο του *overfitting* του μοντέλου μας στα δεδομένα. Καταρχάς χωρίζουμε τα δεδομένα μας σε *train*, *validation*, *test* σύνολα. Μετά από κάθε επανάληψη που κάνουμε εκπαίδευση στο μοντέλο μας με τα δεδομένα εκπαίδευσης τεστάρουμε την επιτυχία του μοντέλου μας στα δεδομένα του *validation*. Στο τέλος υπολογίζουμε το *Minimum Squared Error* και με βάση αυτό βρίσκουμε το *performance* του συστήματός μας. Όταν το *performance* αρχίσει να μειώνεται και πάλι μετά από κάποιο αριθμό επαναλήψεων τότε καταλαβαίνουμε ότι έχουμε το φαινόμενο της υπερεκπαίδευσης και θα πρέπεί να σταματήσουμε να εκπαιδεύουμε το μοντέλο μας. Μπορούμε να έχουμε και κάποιο *lookahead* προκειμένου να μπορούμε να περάσουμε κάποιοα τοπικά μέγιστα και να βρούμε το ολικό μέγιστο. Στο διάγραμμα 4 παρακάτω φαίνεται ακριβώς αυτό, δηλαδή ότι το μοντέλο μας βελτιώνει το *accuracy* του μέχρι κάποιο αριθμό επαναλήψεων και μετά αυτή η μετρική παρουσιάζει χειρότερη συμπεριφορά. Προκειμένου να εξάγουμε αυτό το διάγραμμα

χρειάστηκε να αλλάξουμε το επίπεδο νευρώνων σε 1 με 20 νευρώνες και να επιλέξουμε την καλύτερη μέθοδο ανανέωσης βαρών $traingdx$. Τα 2 επίπεδα είδαμε ότι δεν παρουσιάζουν μια τέτοια χαρακτηριστική καμπύλη υπερεκπαίδευσης λόγω της ικανότητάς τους σε εκφραστικότητα. Επιπλέον στο τελευταίο κομμάτι της άσκησης που υλοποιήσαμε μόνοι μας έναν αλγόριθμο $backpropagation$ φαίνεται ακόμη πιο ευκρινές το προαναφερθέν αποτέλεσμα.

Η μέθοδος του $Early\ Stopping$ φαίνεται και στο διάγραμμα ότι επιτυγχάνει πάντα λιγότερο $overfitting$ (σε πολλές επαναλήψεις έχουμε μικρότερη πτώση στο $accuracy$). Τέλος, με την μέθοδο του $Early\ Stopping$ φτάνουμε μέχρι τις 150 επαναλήψεις το πολύ κάθε φορά άρα γλυτώνουμε και υπολογιστικό κόστος λόγω της μεθόδου αυτής!



Σχήμα 4: Ταξινόμηση με και χωρίς $early - stopping$

Ερώτηση 6

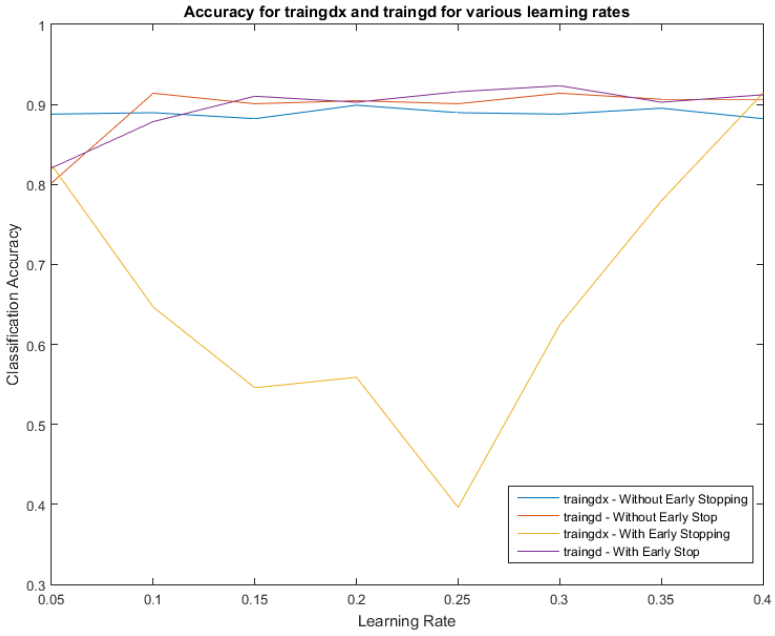
Τα συμπεράσματά μας είναι εμφανώς ότι όταν αυξάνουμε τον αριθμό εποχών τότε βλέπουμε ότι μετά από κάποιο σημείο έχουμε το φαινόμενο της υπερμάθησης. Σε αντίθεση όταν έχουμε πάρα πολύ λίγες εποχές έχουμε το αντίθετο φαινόμενο του $underfitting$. Στο φαινόμενο της υποεκπαίδευσης το μοντέλο μας δεν έχει εκπαιδευτεί κατάλληλα ούτως ώστε να μπορέσει να κάνει την κατάλληλη κατηγοριοποίηση στις εικόνες του $test$. Το προηγούμενο διάγραμμα 4 αυτό το αποτέλεσμα φαίνεται πολύ ξεκάθαρα.

Ερώτηση 7

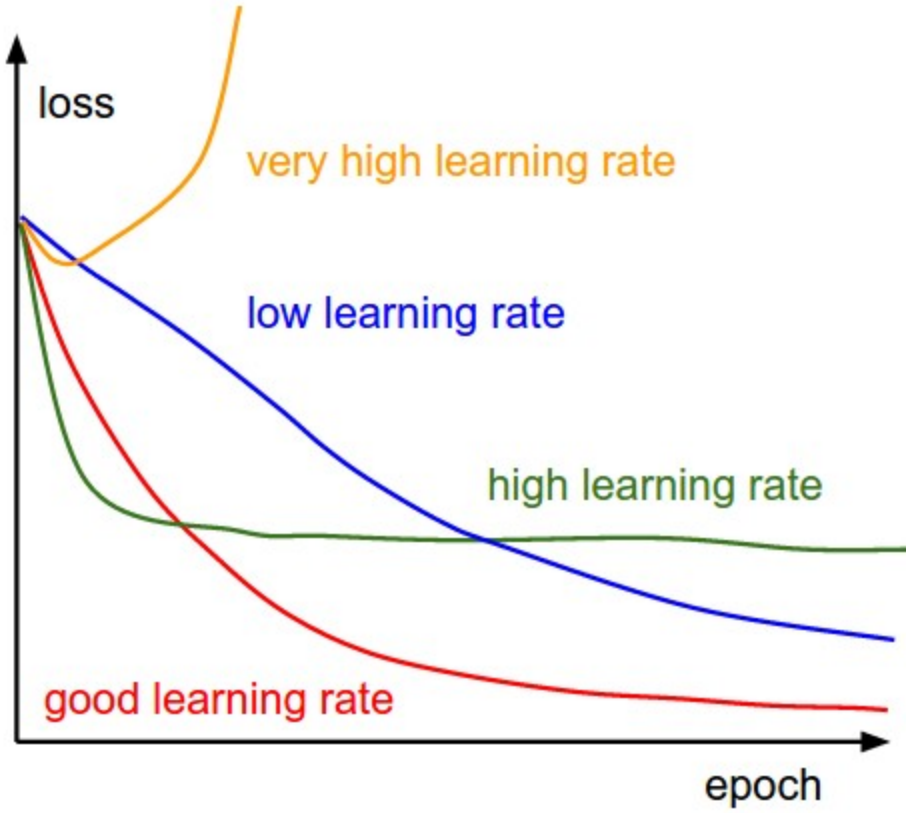
Όταν η παράμετρος του $learning-rate$ είναι πιό μεγάλη γνωρίζουμε ότι η ανανέωση των βαρών μπορεί εν γένει να είναι μεγαλύτερη ενώ όταν αυτό είναι πιο μικρό τότε δεν αφήνουμε να κάνει μεγάλα βήματα στην μέθοδο της απότομης καθόδου με αποτέλεσμα να μπορεί να παγιδευτεί σε κάποιο τοπικό ελάχιστο. Στην εικόνα 6 φαίνεται θεωρητικά η αναμενόμενη επίδραση του $learningrate$ πάνω στην μετρική μας. Βλέπουμε λοιπόν ότι η εύρεση μιας καλής τιμής αυτής της παραμέτρου εξαρτάται κυριως από τα δεδομένα μας.

Στο παρακάτω διάγραμμα 5 συγκρίνουμε τα αποτελέσματα για τις 2 διαφορετικές μεθόδους $traingd-traingdx$ που είναι η απλή μέθοδος και η μέθοδος που συμπεριλαμβάνει τόσο τον όρο με το $momentum$ αλλά και με το $adaptive-learning$ με την παράμετρο του $learning-rate$ να είναι μία φθίνουσα συνάρτηση του αριθμού των επαναλήψεων. Σε αντίθεση στην απλή μέθοδο $traingd$ αυτός ο αριθμός είναι πάντα σταθερός και ίσος με αυτός που του έχουμε αναθέσει.

Απο το διάγραμμα φαίνεται ότι το $earlystop$ με την μέθοδο του $adaptivelearning$ $traingdx$ είναι καλύτερο σε σχέση με το απλό $traingd$ αφού μπορεί να προσαρμόζει κατάλληλα τον πολλαπλασιαστικό παράγοντα του $learning-rate$ και να τον μειώνει με την πάροδο κάποιων εποχών. Με αυτόν τον τρόπο, μετά από κάποιες επαναλήψεις η παράμετρος αυτή μειώνεται και δεν επιτρέπει στον αλγόριθμο να μεταβαίνει σε άλλα τοπικά ελάχιστα που χρειάζονται μεγαλύτερο βήμα.



Σχήμα 5: $traingdandtraingdxaccuracybasedonlearningrate$



Σχήμα 6: $Learning - Rate\ Effect$

Ερώτηση 8

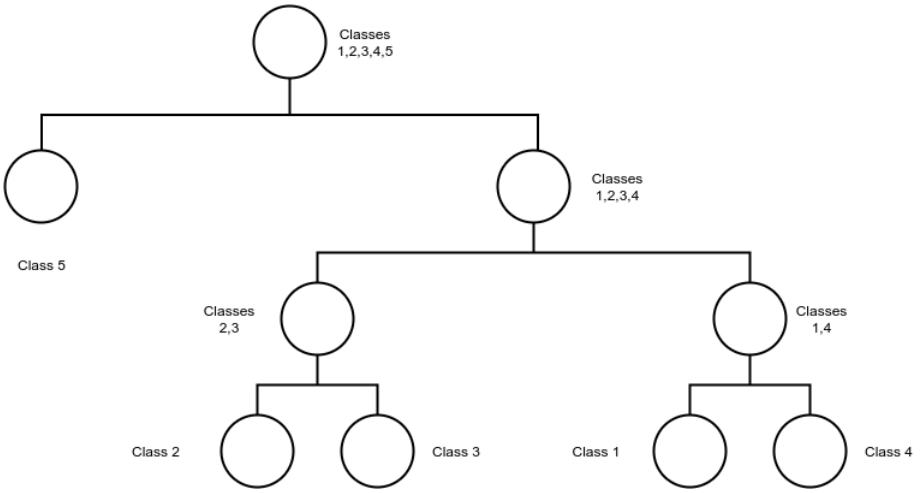
Παρατηρούμε προφανώς από τον $Confusion - Matrix$ παρακάτω

$ConfusionMatrix$					
	Cl 1	Cl 2	Cl 3	Cl 4	Cl 5
$Class$ 1	40	1	0	1	0
$Class$ 2	0	27	3	0	0
$Class$ 3	0	2	28	0	0
$Class$ 4	0	0	0	26	0
$Class$ 5	0	0	1	0	30

ότι κάποιες κατηγορίες έχουν δείγματα που μοιάζουν περισσότερο στα χαρακτηριστικά που έχουμε επιλέξει και για αυτό τον λόγο ο ταξινομητής μας τα κατατάσει λάθος και τα μπερδεύει. Όπως για παράδειγμα με την 2η και την 3η κατηγορία. Εν αντιθέσι, η τελευταία κατηγορία διακρίνουμε ότι ο ταξινομητής μας δεν την μπερδεύει με τα χαρακτηριστικά κάποιας άλλης. Φυσικά αυτό δεν σημαίνει ότι δείγματα άλλων κατηγοριών δεν θα μπορούσαν να είναι όμοια με αυτήν και τότε το $Precision$ να μην είναι 100% αλλά το $Recall$ να είναι.

Ένας πολύ καλός τρόπος για να διορθώσουμε το παραπάνω πρόβλημα θα ήταν

να λειτουργήσουμε με περισσότερους ταξινομητές που λειτουργούν συνεργατικά. Δηλαδή, αυτοί οι ταξινομητές να λειτουργούν κάθε φορά με μία δενδρική δομή και να χωρίζουν σε δύο υποσύνολα το σύνολο δεδομένων κάθε φορά. Με αυτόν τον τρόπο μπορούμε να κάνουμε μια ταξινόμηση σαν και αυτή που φαίνεται στην εικόνα 7. Έτσι εκμεταλευόμαστε τις κλάσεις που είναι πιο κοντά ή πιο τυχαίες και αφήνουμε την απόφαση για πιο μετά. Ενώ η εύκολη απόφαση που είναι αν ανήκει στην τελευταία κλάση ή σε κάποια άλλη την λαμβάνουμε πρώτη. Με αυτόν τον τρόπο τα ποσοστά μας θα αυξηθούν σημαντικά αφού κάθε φορά η ταξινόμηση θα γίνεται σε 2 υποσύνολα με αποτέλεσμα να μπορούμε να μειώσουμε το *disambiguity* στο ελάχστο δυνατό.

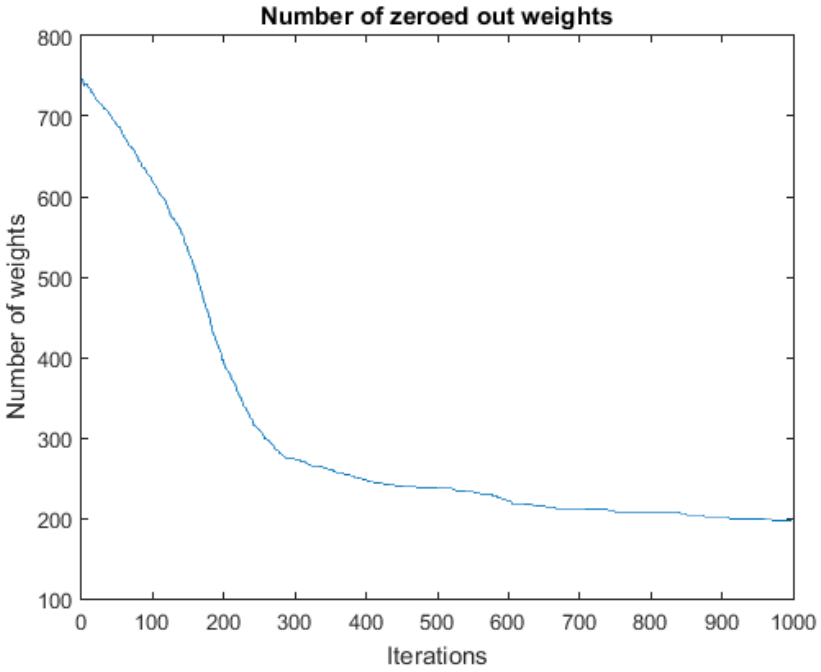


Σχήμα 7: $Binary - Split\ Classification$

4 Η μέθοδος της αποσύνθεσης βαρών

Σε αυτό το κομμάτι της άσκησης υλοποιήσαμε τον αλγόριθμο που περιγράφεται στην άσκηση παραμετροποιώντας τις παραμέτρους του $Neural\ Net$ όυτως ώστε $\lambda = 0.01$, $(learning - rate)$ $d = 1$, $N_{epochs} = 200$. Επειδή η μέθοδος του *Gradient Descend* από μόνη της είναι πάρα πολύ αργή αυξήσαμε το $learning - rate$ για να συγκλίνει ο αλγόριθμός μας πιο γρήγορα (με μικρότερο αριθμό επαναλήψεων). Βλέπουμε ότι με την βελτίωση του κλαδέματος που εισάγουμε μπορούμε να πετύχουμε τα ίδια ποσοστά *accuracy* με αυτά που έχει ένα νευρωνικό 30 νευρώνων ενός επιπέδου με μόνο το 25% των συνδέσεων που απαιτούσε το πλήρως συνδεδεμένο δίκτυο. Με αυτόν τον τρόπο εκτός του ότι βελτιώνουμε σημαντικά τις απαιτήσεις του συστήματός μας σε χώρο και σε υπολογιστικό χρόνο μπορούμε να δούμε το πόσο *expressive* μπορεί να γίνει ένα νευρωνικό δίκτυο με μόνο 1 επίπεδο και 30 νευρώνες.

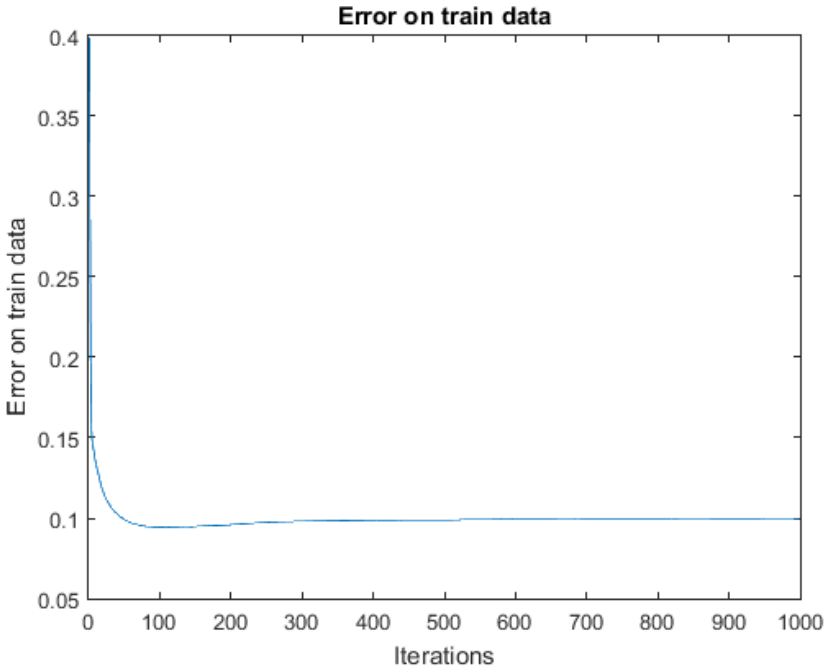
Προκειμένου να δείξουμε το πώς μεταβάλλεται ο αριθμός των συνδέσεων-βαρών με την πάροδο κάποιου αριθμού επαναλήψεων, παραθέτουμε το διαγραμμα 8. Για το σφάλμα εκπαίδευσης μπορούμε να το δούμε στο αντίστοιχο διάγραμμα 9 , το οποίο το υπολογίσαμε με το μέσο τετραγωνικό σφάλμα μέσω του $performance$



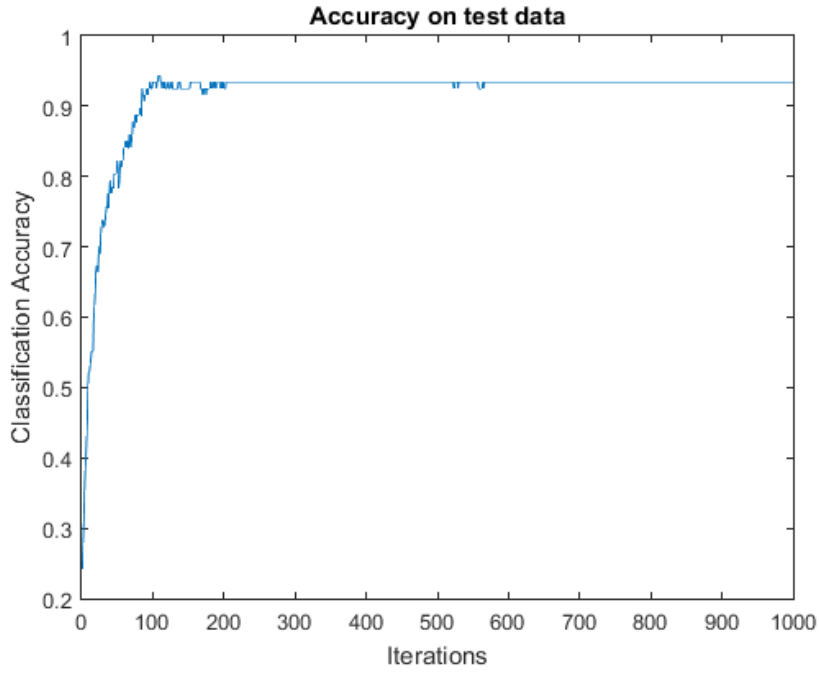
Σχήμα 8: Μηδενισμένα Βάρη

του δικτύου. Είναι αρκετά σημαντικό να δούμε το πώς εξελίσσεται το σφάλμα στο $test - set$ καθώς αυξάνεται ο αριθμός των επαναλήψεων 10.

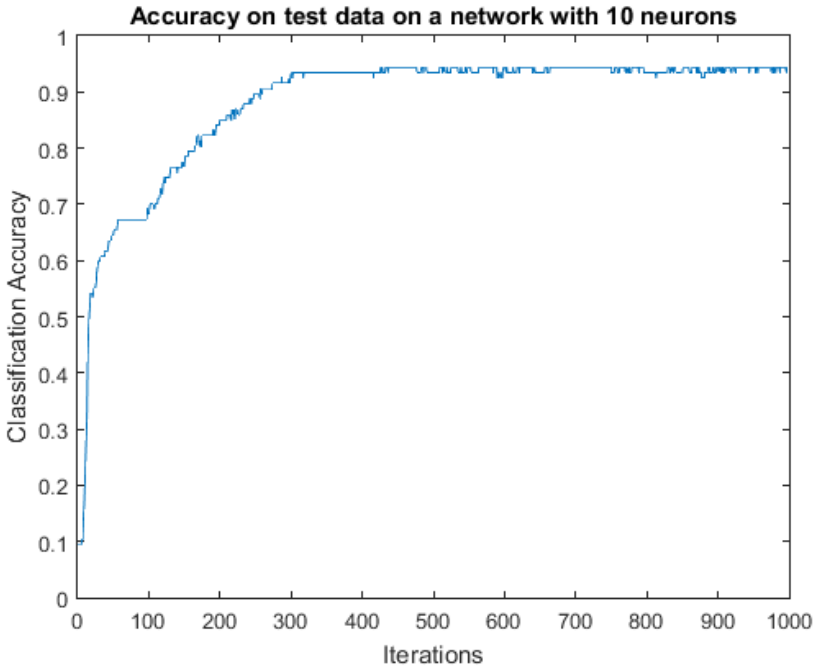
Με βάση το παραπάνω διάγραμμα στο 8 θα προσπαθήσουμε να πετύχουμε το ίδιο αποτέλεσμα με το νευρωνικό 30 νευρώνων με ένα νευρωνικό δίκτυο 10 νευρώνων (33% των νευρώνων). Παρακάτω φαίνεται το διάγραμμα επιτυχίας ταξινόμησης για δίκτυο 10 νευρώνων 11. Από το παρακάτω διάγραμμα επιβεβαιώνεται η διαίσθηση μας ότι μπορούμε να πετύχουμε το ίδιο αποτέλεσμα με λιγότερες συνδέσεις.



Σχήμα 9: Επίδοση του νευρωνικού στα δεδομένα εκπαίδευσης



Σχήμα 10: Επιτυχία Ταξινόμησης νευρωνικού στα δεδομένα ελέγχου



Σχήμα 11: Επιτυχία Ταξινόμησης νευρωνικού στα δεδομένα ελέγχου (10 νευρώνες)