Άσκηση 1 - ΕΠΛ 442

Όνομα : Έλια Νικολάου 1012334
Ημερομηνία παράδοσης: 15/10/20

Επεξήγηση – Σκοπός Προγράμματος

Σκοπός του προγράμματος είναι η υλοποίηση ενός Feed-Forward νευρωνικού δικτύου που μπορεί να μάθει να υπολογίζει την exclusive-OR (XOR) συνάρτηση. Ως μέθοδος μάθησης χρησιμοποιείται η Back Propagation.

Επεξήγηση Κλάσεων

1. Drive

Η κλάση αυτή λειτουργεί ως driver. Αποτελεί το σημείο εισόδου του προγράμματος. Εδώ διαβάζονται από τα δοθέντα αρχεία όλοι οι παράμετροι που χρειάζονται για την λειτουργία του προγράμματος.Η Drive κλάση είναι υπεύθυνη να δημιουργήσει το νευρωνικό μας δίκτυο (τύπου) Neurons_Network, και να καλέσει τις συναρτήσεις οι οποίες θα τρέξουν το training και το testing του νευρωνικού δικτύου καθώς επίσης είναι υπεύθυνη για να περάσει τα αποτελέσματα στα αρχεία υπολογίζοντας το error (training & testing) και το success rate (training & testing).
Η Drive παράγει δύο αρχεία

1. successrate.txt => περιλαμβάνει το ποσοστό ορθών αποτελεσμάτων κατά την εκμάθηση (training phase) και το ποσοστό ορθών αποτελεσμάτων κατά τον έλεγχο (testing phase).
2. errors.txt => περιλαμβάνει το λάθος εκμάθησης (training error) στο τέλος κάθε επανάληψης και το λάθος ελέγχου (testing error) στο τέλος κάθε επανάληψης.

2. Neurons_Network

Η κλάση αυτή αποτελεί την σημαντικότερη κλάση του προγράμματος μας. Είναι ουσιαστικά ένα πρότυπο ενός νευρωνικού δικτύου. Περιέχει ένα πίνακα από neurons (τύπου neuron), τα οποία χαρακτηρίζονται από το layer τους και τα βάρη τα οποία συνδέουν το κάθε νευρώνα με τους νευρώνες του επόμενου layer. Η κλάση λειτουργεί όπως και ένα νευρωνικό δίκτυο.
Αρχικά κάνει initialize τους νευρώνες του, δίνοντας τους random τιμές καθώς επίσης και το threshold του κάθε νευρώνα με τον ίδιο τρόπο.
Μετά όταν καλείται από το Drive, εκπαιδεύεται και ελέγχεται για συγκεκριμένα inputs και outputs

-> training με την χρήση του Forward_propagation & του Back_propagations (δες συναρτήσεις)
-> testing με την χρήση μόνο του Forward_propagation.

3. Layer

Αποτελεί ουσιαστικά βοηθητική κλάση. Δεν έχει κάποια πολύ βασική λειτουργία για το πρόγραμμα, αφού απλά περιέχει μεταβλητές που χρησιμοποιούνται από τις υπόλοιπες κλάσεις. Βασικά περιέχει πληροφορίες για τα επιμέρους layers, τι layers υπάρχουν και πόσους νευρώνες έχει το κάθε layer.

4. Neuron

Η κλάση αυτή αποτελεί πρότυπο ενός νευρώνα. Ένας νευρώνας χαρακτηρίζεται από το output, το bias, τα weights , το προηγούμενο bias και το layer στο οποίο ανήκει. Δεν έχει κάποια συνάρτηση μέσα. Στην δική μας περίπτωση χρησιμοποιείται ως αντικείμενο από την κλάση Neurons_Network.

Λεπτομέρειες λειτουργίας (instructions)

Το πρόγραμμα μου δεν χρειάζεται κάποια συγκεκριμένη διαχείριση για να λειτουργήσει.
Κάνω όμως την παραδοχή ότι το file parameters.txt είναι σταθερό. (Το έχω ρητά ως μεταβλητή μέσα στο πρόγραμμα)

Παραδείγματα Λειτουργίας (simulations)

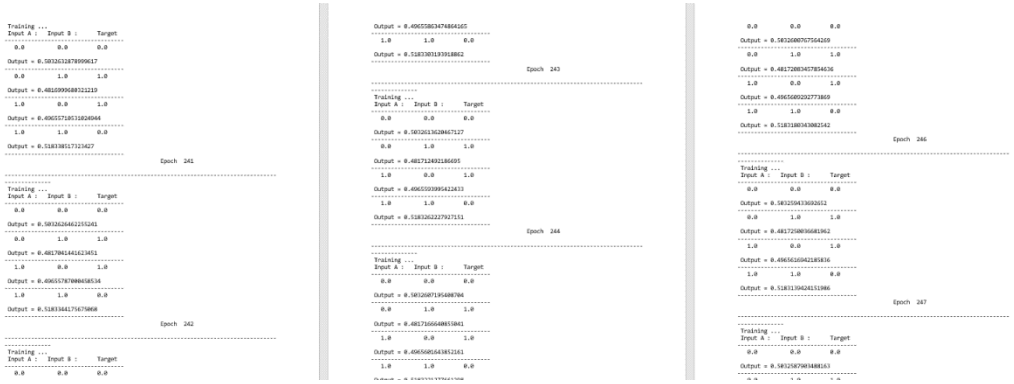
Δημιούργησα πολλά παραδείγματα αλλάζοντας το learning_rate, το momentum και τα iterations. Τα διάφορα αποτελέσματα μου έδιναν στοιχεία και συμπεράσματα. Ωστόσο ο αριθμός των layer είναι σταθερός | 2 (input_neurons) , 2 (Hidden1_neurons) , 0 (Hidden2_neurons) , 1 (output_neurons) |

Παραθέτω κάποια από τα παραδείγματα από την αρχή ως ένα από τα τελικά «βέλτιστα» μου, με πολύ καλή εκπαίδευση του δικτύου. (Δεν εμφανίζονται ολόκληρα, επειδή τα iterations είναι πολλά)

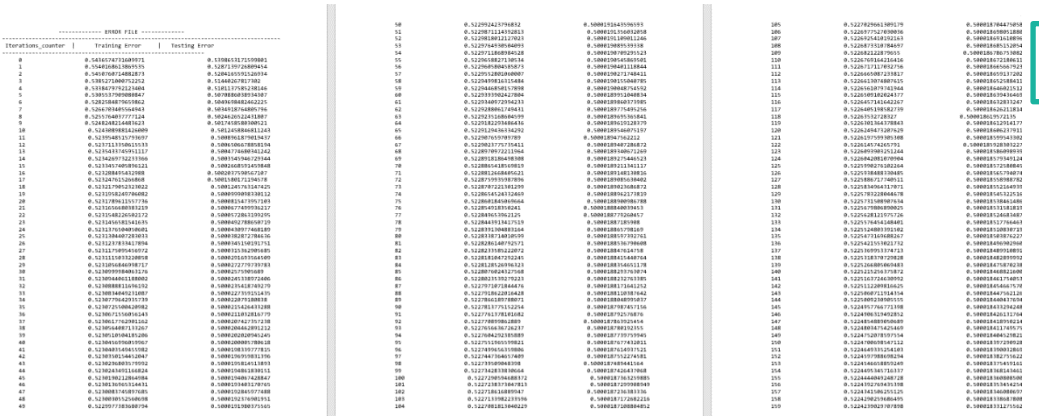
1.

Learning Rate	Momentum	Iterations
0.3	0.2	200

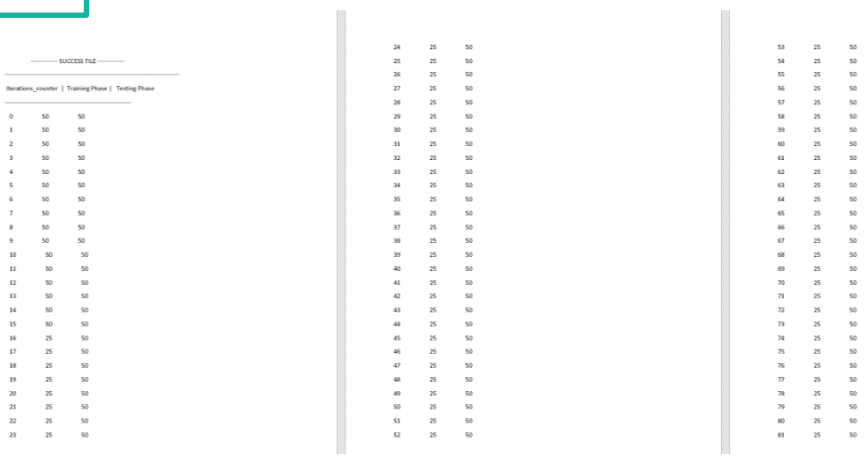
Console



Errors.txt

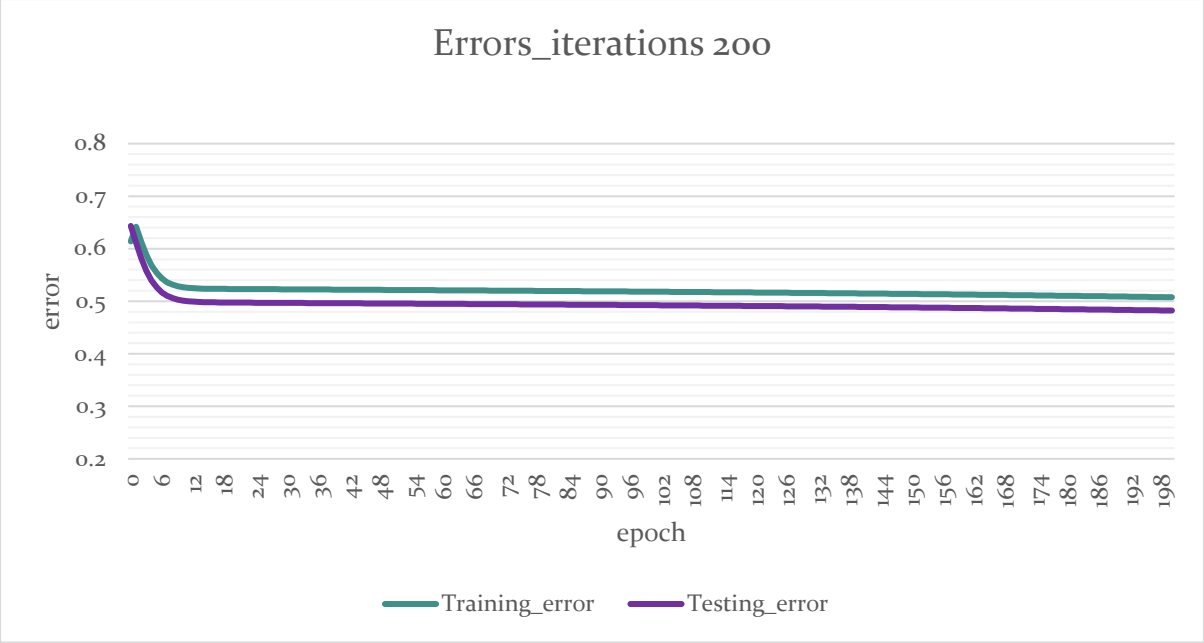


SuccessRate.txt



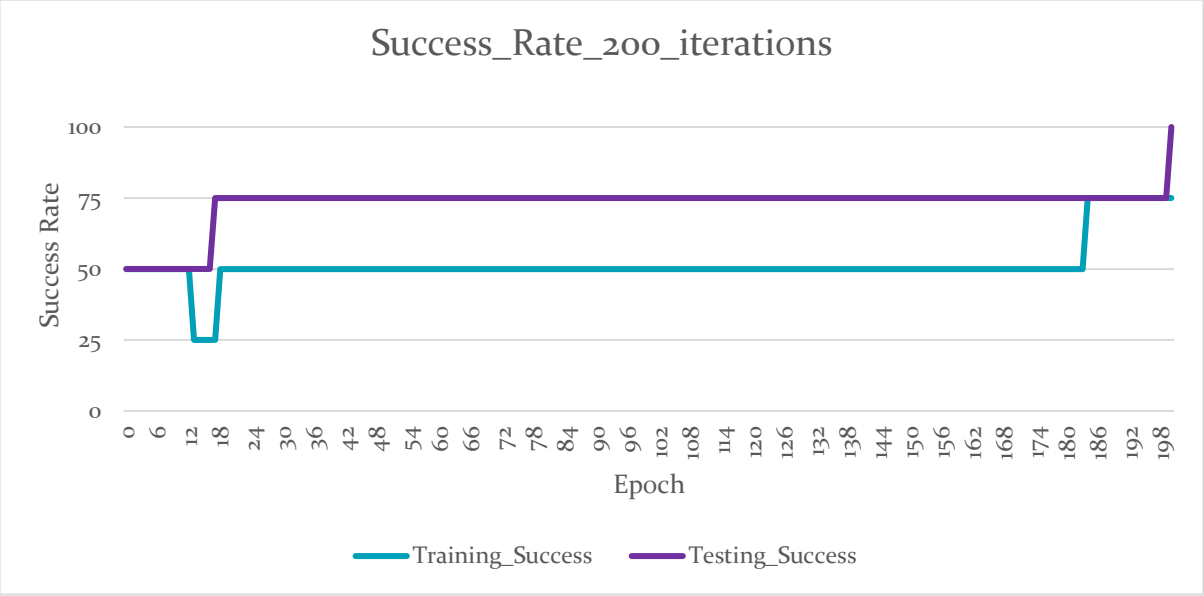
Παρατηρήσεις :

Παρατηρώ ότι παρόλο που το πρόγραμμα δίνει λίγο καλύτερα αποτελέσματα όσο προχωράνε τα iterations – οι εποχές μου, το δίκτυο ωστόσο δεν εκπαιδεύεται άριστα. Πιθανή Λύση είναι να αυξήσω τον αριθμό των iterations μου και να αλλάξω τις υπόλοιπες μεταβλητές



Παρατηρήσεις :

Παρατηρούμε ότι το Training Error είναι υψηλότερο σε σχέση με το Testing Error πράγμα που σημαίνει ότι το συνολικό σφάλμα είναι πιο μεγάλο(εάν το σκεφτούμε ως απόκλιση, τότε η απόκλιση είναι μεγαλύτερη από το επιθυμητό μας αποτέλεσμα). Παρατηρούμε ωστόσο ότι καθώς αυξάνονται οι εποχές το δίκτυο εκπαιδεύεται λίγο καλύτερα αλλά όχι, δεν εκπαιδεύεται ολόσωστα αφού βλέπουμε ότι παραμένει σταθερό χωρίς μείωση του σφάλματος. Αλλάζοντας τις παραμέτρους μπορούμε να επιτύχουμε μεγάλη ελαχιστοποίηση του σφάλματος.

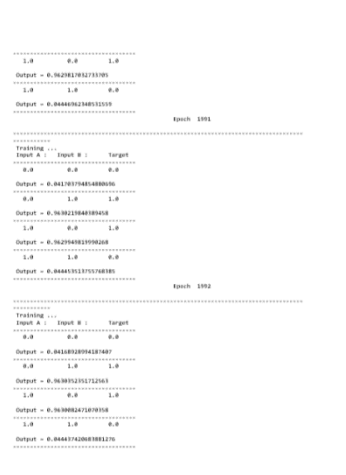


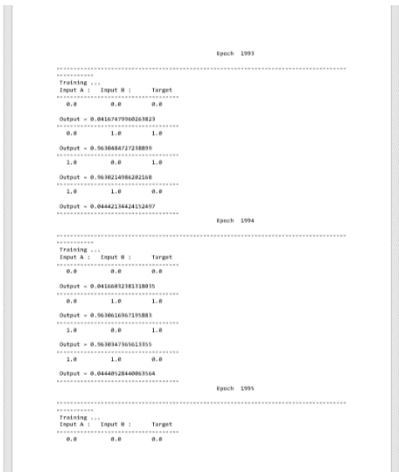
Παρατηρήσεις :

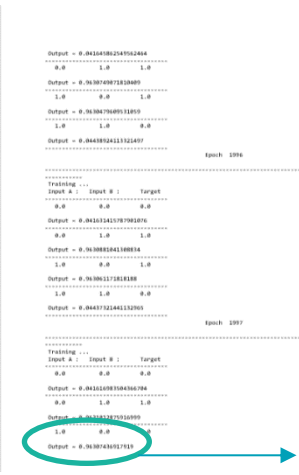
Παρατηρούμε ότι το Training_Success_Rate είναι χαμηλότερο σε σχέση με το Testing_Success_Rate. Ωστόσο το δίκτυο πετυγχαίνει 100% success στα τελευταία iterations, εάν αυξήσουμε το learning rate & το momentum μπορούμε να δούμε ότι το δίκτυο εκπαιδεύεται καλύτερα ως αποτέλεσμα να έχουμε πιο πολύ αποτελεσματικότητα και για το training και το testing.

2. Αλλαγή του αριθμού των iterations , του learning_rate και του momentum

Learning Rate	Momentum	Iterations
0.5	0.5	200

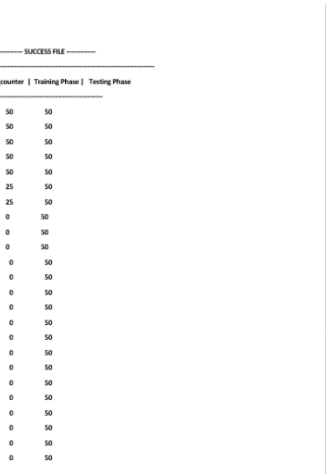


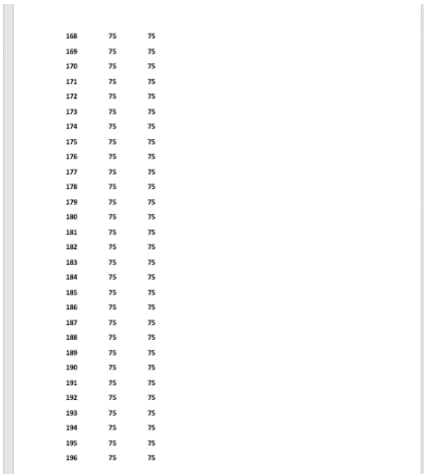


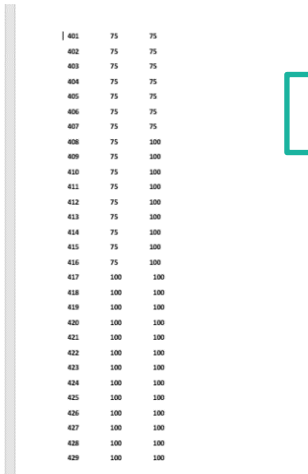


Console

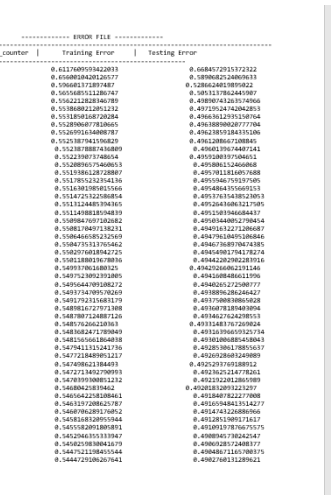
Καταλήγουμε να έχουμε τιμές πολύ κοντά στα επιθυμητά μας αποτελέσματα

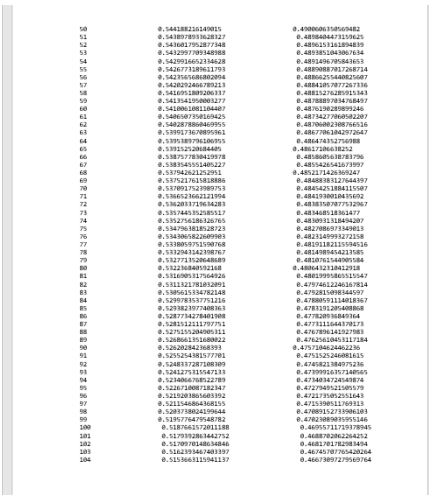


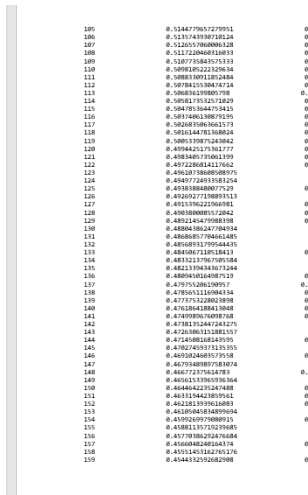




SuccessRate.txt





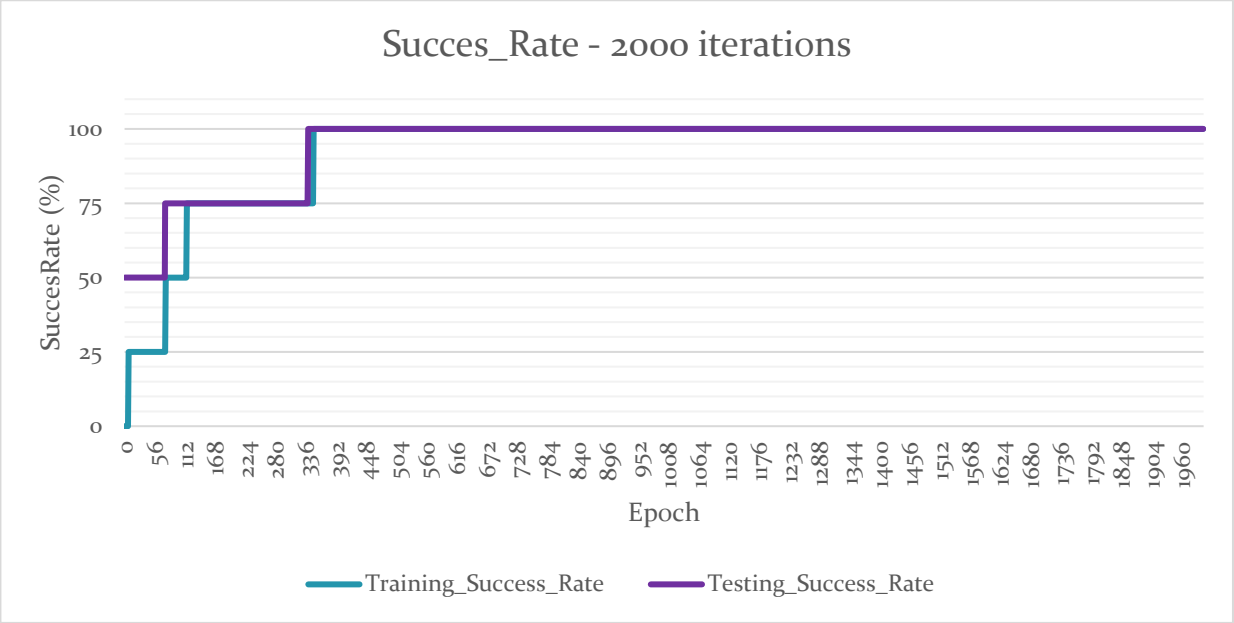


Errors.txt

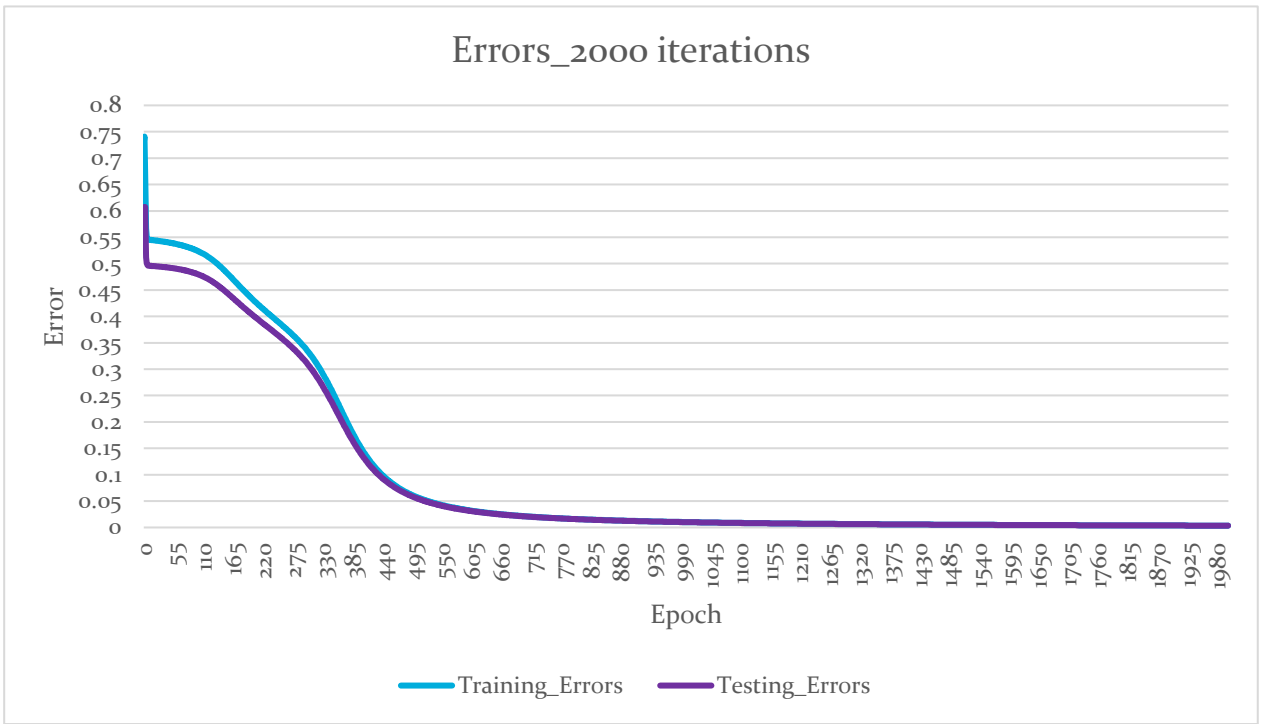
Παρατηρήσεις :

Εάν δούμε το output από το screenshot της κονσόλας παρατηρούμε ότι είναι πολύ κοντά στα επιθυμητά μας αποτελέσματα (κύκλος) . Επιπλέον, παρατηρούμε ότι τα success rate φτάνουν στο μέγιστο δυνατό 100% και τα errors ελαχιστοποιούνται σε μεγάλο βαθμό. Άρα κάνουμε εδώ την παραδοχή ότι το learning_rate & το momentum κάνουν μεγάλη αλλαγή καθώς αυξάνονται.

Γραφικές Παραστάσεις
= για τα αποτελέσματα της γραφικές πήρα τις τιμές της τελευταίας μου εκτέλεσης, όπου δεδομένα είναι το πιο πάνω παράδειγμα



Παρατηρούμε ότι το Training_Success_Rate είναι χαμηλότερο σε σχέση με το Testing_Success_Rate. Παρατηρούμε ωστόσο ότι καθώς αυξάνονται οι εποχές το δίκτυο εκπαιδεύεται πολύ καλά σε σημείο που έχει τα ίδια αποτελέσματα με το Testing_Success_Rate στο μέγιστο 100% - βρίσκει δηλαδή όλα τα επιθυμητά outputs.



Παρατηρούμε ότι το Training Error είναι υψηλότερο σε σχέση με το Testing Error. Παρατηρούμε ωστόσο ότι καθώς αυξάνονται οι εποχές το δίκτυο εκπαιδεύεται πολύ καλά σε σημείο που έχει τα ίδια αποτελέσματα με το Testing_Error όπου το error ελαχιστοποιείται.