**Υλοποίηση MLP με Gradient Descent**

**Εισαγωγή**

Αυτή η αναφορά περιγράφει την υλοποίηση ενός νευρωνικού δικτύου MLP στη γλώσσα προγραμματισμού C. Το δίκτυο εκπαιδεύεται χρησιμοποιώντας τον αλγόριθμο βελτιστοποίησης gradient descent και αξιολογείται για τις δυνατότητές του να γενικεύει χρησιμοποιώντας ένα σύνολο δεδομένων ελέγχου. Ο κώδικας έχει οργανωθεί σε διάφορες συναρτήσεις και δομές δεδομένων για να ενισχύσει την κατανόηση.

**Δομές Δεδομένων**

* **Network\_t**: Αναπαριστά τη δομή του νευρωνικού δικτύου, περιλαμβάνοντας τα επίπεδα και τους νευρώνες.
* **Input\_t**: Αποθηκεύει τα χαρακτηριστικά εισόδου (x1, x2) μαζί με τις αντίστοιχες ετικέτες κατηγορίας.
* **Neuron\_t**: Αναπαριστά έναν μεμονωμένο νευρώνα, συμπεριλαμβανομένων των βαρών, των παραγώγων σφάλματος και των τιμών εξόδου.

**Αρχικοποίηση (Συνάρτηση** init**)**

* Φορτώνει τα σύνολα εκπαίδευσης και ελέγχου από αρχεία CSV (train\_dataset.csv και test\_dataset.csv).
* Αρχικοποιεί το νευρωνικό δίκτυο με τυχαία βάρη για όλους τους νευρώνες.
* Ορίζει παραμέτρους όπως ο αριθμός των νευρώνων ανά επίπεδο, οι συναρτήσεις ενεργοποίησης και άλλες ρυθμίσεις.

**Κωδικοποίηση Κατηγοριών Εισόδου (Συνάρτηση** encode\_input**)**

* Κατηγοριοποιεί τα σημεία δεδομένων εισόδου σε τέσσερις κατηγορίες (C1, C2, C3 και C4) βάσει συγκεκριμένων συνθηκών.
* Χρησιμοποιεί μέθοδο κωδικοποίησης 1-από-p για να αντιστοιχίσει ετικέτες κατηγορίας σε τα δεδομένα εισόδου.

**Συνάρτηση** forward\_pass

* Εκτελεί την προώθηση δεδομένων μέσα από το νευρωνικό δίκτυο.
* Υπολογίζει την έξοδο κάθε νευρώνα σε κάθε επίπεδο, συνδυάζοντας το βαρυτικό άθροισμα των εισόδων με μια συνάρτηση ενεργοποίησης.

**Συνάρτηση** reverse\_pass

* Υλοποιεί την αντίστροφη διάβαση ή backpropagation για τον υπολογισμό των σημάτων σφάλματος (deltas) για κάθε νευρώνα.
* Χρησιμοποιεί τον κανόνα της αλυσίδας για να μεταδώσει τα σφάλματα προς τα πίσω μέσω των επιπέδων.

**Εκπαίδευση με Gradient Descent (Συνάρτηση** train\_using\_gradient\_descent**)**

* Εκπαιδεύει το νευρωνικό δίκτυο χρησιμοποιώντας τη μέθοδο gradient descent.
* Επαναλαμβάνει τις εποχές και τα mini-batches για να ενημερώσει τα βάρη του δικτύου.
* Υπολογίζει και μερικές παραγώγους για τις ενημερώσεις των βαρών.
* Καταγράφει τα σφάλματα εκπαίδευσης, αποθηκεύοντας τα σε ένα αρχείο CSV για ανάλυση.
* Υπολογίζει το σφάλμα εκπαίδευσης στο τέλος κάθε εποχής.
* Το σφάλμα ορίζεται ως ο μέσος τετραγωνικός όρος μεταξύ των προβλεπόμενων και των πραγματικών ετικετών κατηγορίας.
* Ενημερώνει τα βάρη για κάθε νευρώνα σε κάθε επίπεδο χρησιμοποιώντας τις συσσωρευμένες μερικές παραγώγους και τον ρυθμό μάθησης (n).

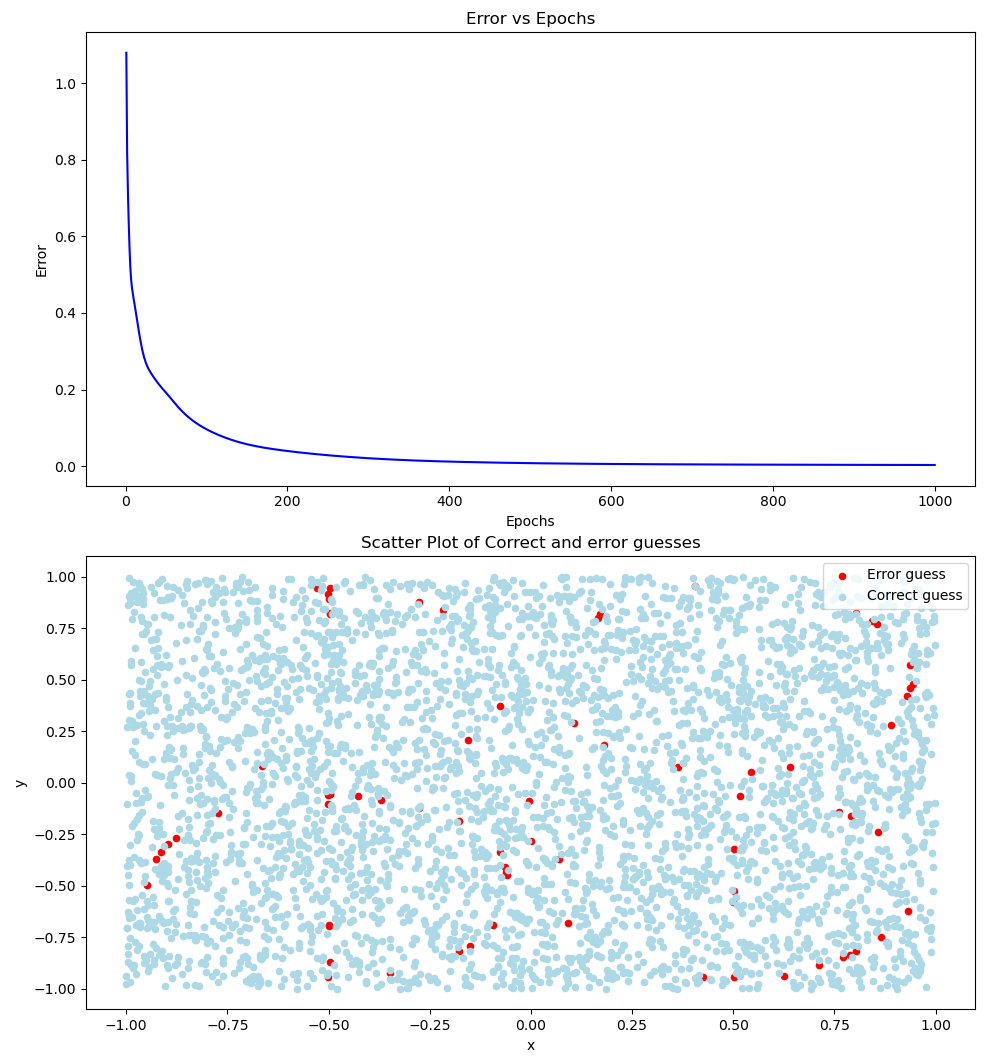
**Αξιολόγηση Γενίκευσης (Συνάρτηση** calculate\_generalization\_capability**)**

* Αξιολογεί τις δυνατότητες γενίκευσης του εκπαιδευμένου δικτύου χρησιμοποιώντας το σύνολο δεδομένων ελέγχου.
* Καταγράφει σωστά και εσφαλμένα ταξινομημένα σημεία δεδομένων.
* Υπολογίζει το ποσοστό σφάλματος και την ακρίβεια των προβλέψεων του δικτύου.

**Αποτελέσματα:**

| H1 | H2 | H3 | Learning rate | Batch size | Activation function | Accuracy |
| --- | --- | --- | --- | --- | --- | --- |
| 10 | 10 | 10 | 0.1 | 40 | Logistic | 96,75% |
| 10 | 10 | 10 | 0.1 | 40 | Tanh | 18.7% |
| 10 | 10 | 10 | 0.1 | 40 | Relu | 30.97% |
| 10 | 30 | 10 | 0.1 | 40 | Logistic | 88.48% |
| 10 | 30 | 10 | 0.1 | 40 | Tanh | 30.97% |
| 30 | 30 | 30 | 0.1 | 40 | Logistic | 95.8% |
| 30 | 30 | 30 | 0.1 | 40 | Tanh | 31.42% |
| 30 | 30 | 30 | 0.1 | 40 | Relu | 30.97% |
| 30 | 30 | 30 | 0,001 | 40 | Logistic | 97.52% |
| 30 | 30 | 30 | 0,0001 | 40 | Logistic | 97.25% |
| 30 | 30 | 30 | 0,01 | 400 | Logistic | 96.60% |
| 50 | 50 | 50 | 0,001 | 40 | Logistic | 97.67% |
| 10 | 50 | 50 | 0,001 | 40 | Logistic | 95.95% |
| 30 | 50 | 50 | 0,001 | 40 | Logistic | 97.28% |
| 30 | 50 | 30 | 0,001 | 40 | Logistic | 97.55% |
| 30 | 70 | 30 | 0,001 | 40 | Logistic | 97.35% |
| 30 | 30 | 30 | 0,001 | 40 | Logistic | 97.12% |
| 30 | 30 | 30 | 0,0001 | 40 | Logistic | 97.87% |
| 30 | 50 | 50 | 0,0001 | 40 | Logistic | 98,30% |

Παρατηρούμε ότι το δίκτυο είναι πιο ακριβές χρησιμοποιώντας H1: 30, H2: 50, H3: 50 με learning rate 0,0001 και batch size: 40.

Οπτικοποίηση των αποτελεσμάτων της καλύτερης περίπτωσης:

**S**

**Αρχεία και Περιγραφές**

**1.** mlp.c

* Περιγραφή: Υλοποίηση του πολυεπίπεδου νευρωνικού δικτύου MLP.

**2.** mlp.h

* Περιγραφή: Περιέχει κώδικα για τη δημιουργία των συνόλων εκπαίδευσης και δοκιμής.

**3.** Makefile

* Περιγραφή: Makefile για τη μεταγλώττιση του προγράμματος.

**4.** generate\_dataset.py

* Περιγραφή: Σενάριο Python για τη δημιουργία των συνόλων δοκιμής και εκπαίδευσης.

**5.** visualize.py

* Περιγραφή: Κώδικας για την οπτικοποίηση των αποτελεσμάτων.

**Μεταγλώττιση και Εκτέλεση**

Για να μεταγλωτίσετε και να εκτελέσετε το πρόγραμμα, ακολουθήστε αυτά τα βήματα:

* Ανοίξτε ένα τερματικό.
* Πλοηγηθείτε στον φάκελο του έργου που περιέχει τα αναφερόμενα αρχεία.
* Εκτελέστε τις παρακάτω εντολές:  
  make && ./mlp

**Πρόσθετες Σημειώσεις**

* Προσαρμόστε τις παραμέτρους και τις ρυθμίσεις εντός των αρχείων κώδικα (mlp.c, generate\_dataset.py, κλπ.) όπως απαιτείται για τα file paths.
* Χρησιμοποιήστε τον κώδικα visualize.py για να οπτικοποιήσετε τα αποτελέσματα που προκύπτουν από την εκτέλεση του πολυεπίπεδου νευρωνικού δικτύου.