

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

**ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ Η/Υ ΚΑΙ ΠΛΗΡΟΦΟΡΙΚΗΣ**

|  |
| --- |
|  |
|  |

**ΥΠΟΛΟΓΙΣΤΙΚΗ ΝΟΗΜΟΣΥΝΗ**

Ακαδημαϊκό Έτος 2021-2022

**Εργαστηριακή Άσκηση**

**Μέρος Α’**

**Γεώργιος Κοντογιάννης**

**1070908 – Δ’ έτος**

**Code Repo Link:**

https://github.com/gkontogiannhs/Text-Recognition-NLP

**α)** Η κωδικοποίηση των λέξεων είναι γωνστή και ανήκει στο εύρος [0, 8519]. Για το λόγο αυτό, ο CountVectorizer παραμετροποιήθηκε με αυτό το ήδη υπάρχον λεξικό. Καλώντας την μέθοδο transform, μετατράπηκαν τα δεδομένα εισόδου σε BoW και κατ’ επέκταση δημιουργήθηκε το Document Term Matrix.

**β)** Η κλιμάκωση(scaling) των δεδομένων, είναι ένα από τα πιο σημαντικά βήματα προεπεξεργασίας δεδομένων στη μηχανική εκμάθηση. Οι αλγόριθμοι που υπολογίζουν την απόσταση μεταξύ των χαρακτηριστικών, ωθούνται προς τις αριθμητικά μεγαλύτερες τιμές (outliers) εάν τα δεδομένα δεν είναι κλιμακωμένα.

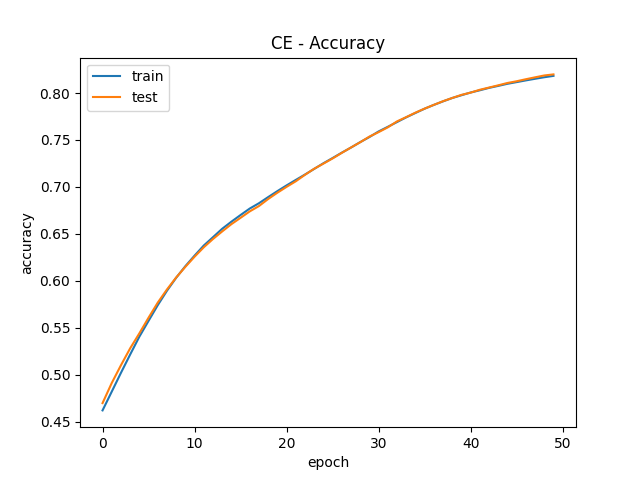
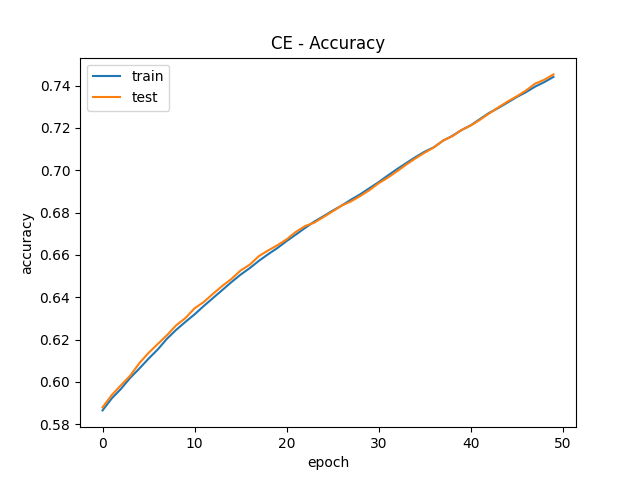
1. To **Κεντράρισμα** (centering) αφαιρεί μία σταθερή τιμή απο κάθε μεταβλητή εισόδου. Πρακτικά, επαναπροσδιορίσει το σημείο 0 για τον προγνωστικό παράγοντα, ώστε να είναι οποιαδήποτε τιμή αφαιρέθηκε. Μετατοπίζει την κλίμακα, αλλά διατηρεί τις μονάδες.

2. Η **κανονικοποίηση** (scaling) χρησιμοποιείται για τη μετατροπή των χαρακτηριστικών σε παρόμοια κλίμακα. Το νέο σημείο υπολογίζεται απο το τύπο: . Αυτό κλιμακώνει το εύρος σε [0, 1] ή μερικές φορές [-1, 1]. *Η κανονικοποίηση είναι χρήσιμη όταν δεν υπάρχουν ακραίες τιμές, καθώς δεν μπορεί να τις αντιμετωπίσει*.

3. Η **Τυποποίηση** (standardization) είναι ο μετασχηματισμός χαρακτηριστικών με αφαίρεση τον μέσο όρο και διαίρεση με τυπική απόκλιση. . Η τυποποίηση μπορεί να είναι χρήσιμη σε περιπτώσεις όπου τα δεδομένα ακολουθούν μια κατανομή Gauss. Ωστόσο, αυτό δεν είναι απαραίτητο να ισχύει. *Η τυποποίηση δεν επηρεάζεται από ακραίες τιμές επειδή δεν υπάρχει προκαθορισμένο εύρος μετασχηματισμένων χαρακτηριστικών.*

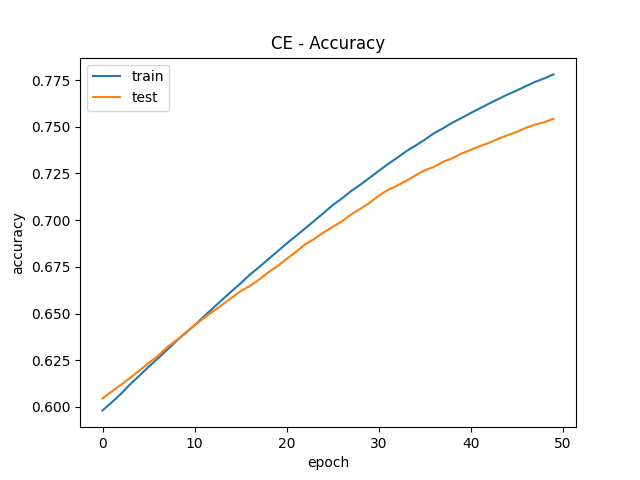
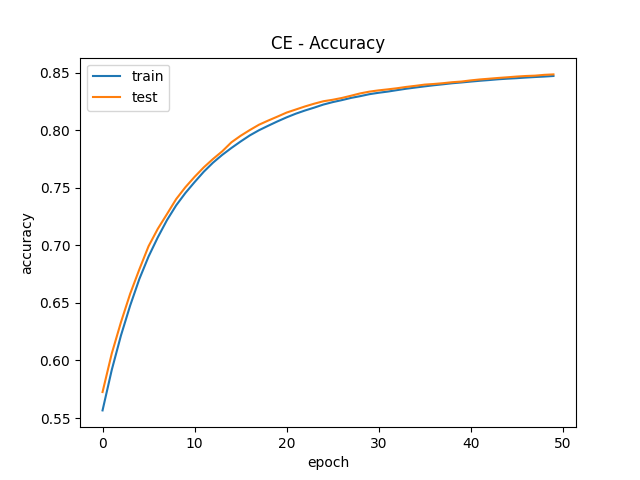
***\*\*\* Πρακτικά, ο συνδυασμός του κεντραρίσματος και κανονικοποίησης έχει ώς αποτέλεσμα την τυποποίηση*.**

* Tα παρακάτω πειράματα έγιναν για να δείξουν τη συμπεριφορά του αλγορίθμου ανά είδος αρχικής προεπεξεργασίας των δεδομένων εισόδου Τυπικά, τα πειράματα έγιναν με 20 νευρώνες στο κρυφό επίπεδο, 50 epoch και binary CE ως loss function.



No Scaling -

Centering -



Normalization -

Standardization -

***Πόρισμα 1.1***:

Απο το παραπάνω πείραμα εύκολα βλέπουμε οτι η **κανονικοποίηση** αποφέρει καλύτερα. Αυτό δε θα έπρεπε να μας παραξενεύει καθώς τα δεδομένα εισόδου είναι όλα θετικά, και στη περίπτωση των outliers, όπου η διαφορά με τη πληθώρα των μηδενικών γίνεται μεγάλη (τάξη του ), αυτή εξαλείφεται.

***Πόρισμα 1.2:***

Η **τυποποίηση** και το **κεντράρισμα** εισάγουν αρντητικές τιμές εισόδων. Αυτό δεν είναι επιθυμητό για τον αλγόριθμο διότι οι αρνητικές τιμές συχνότητας επαναλήψεων δεν έχουν πρακτικό νόημα. Ωστόσο, στο σημείο αυτό να αναφέρουμε, οτι η δεύτερη καλύτερη επιλογή είναι να μην εφαρμόσουμε κάποιο μετασχηματισμό στα δεδομένα. Ούτε αυτό θα έπρεπε να μας παραξενέψει καθώς τα έγγραφα έχουν υποστεί αφαίρεση των stopwords και stemming. Μεγάλες συχνότητες των λέξεων προσδίνουν μεγάλη βαρύτητα, το οποίο είναι λογικό.

**α)** Κάθε μια απο τις μετρικές αυτές, είναι μια μέθοδος αξιολόγησης του πόσο καλά μοντελοποιεί ο αλγόριθμός το σύνολο δεδομένων. Εάν οι προβλέψεις είναι εντελώς «εκτός», η συνάρτηση κόστους θα παράγει μεγαλύτερο αριθμό. Αν είναι αρκετά καλά, θα βγάζει μικρότερο αριθμό.Αυτός ο αριθμός είναι πολύ σημαντικός γιατί μας βοηθάει να εντοπίσουμε τοπικά ελάχιστα. Στο δικό μας πρόβλημα, αρχικά να πούμε οτι οι απόλυτες διαφορές ανάμεσα στην πραγματική έξοδο με την επιθυμητή είναι πάντα ανάμεσα στο [0, 1]. Άρα, στη περίπτωση του **Cross-Entropy**, μικρές πιθανότητες πρόβλεψης τιμωρούνται περισσότερο. Στο **MSE**, τα μεγαλύτερα σφάλματα είναι αυτά που «τιμωρούνται» περισσότερο λόγω του τετραγώνου. Τέλος, το **Acc** ελέγχει πόσες απο τις ετικέτες είναι ίδιες και διαιρεί με το πλήθος των πραγματικών και ψευδών ετικετών. Άρα, ο αλγόριθμος δε συμπεριλαμβάνει τις ψευδώς ‘1’ εντοπισμένες κλάσεις.

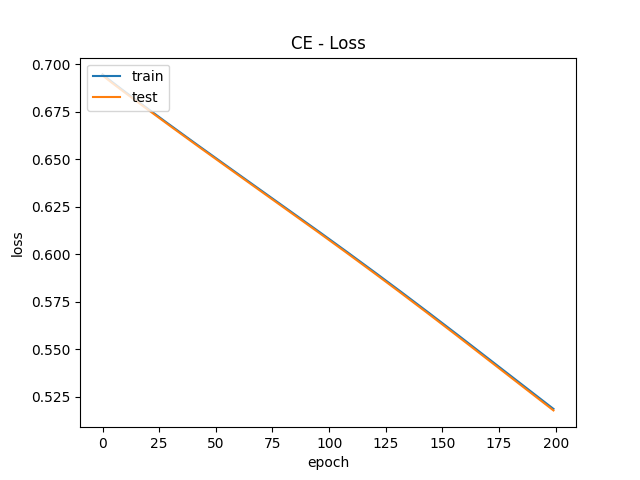
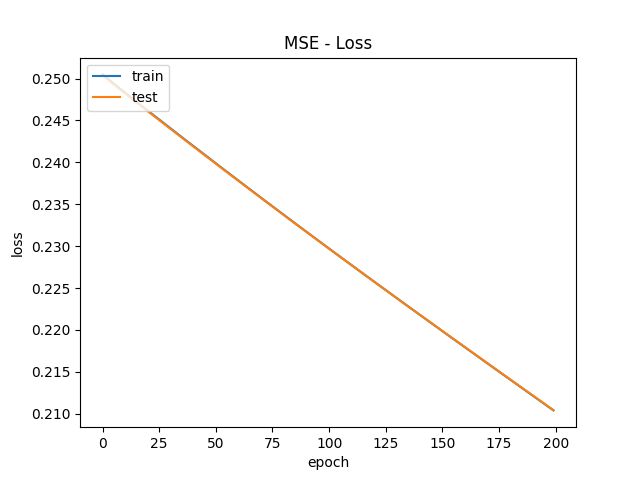
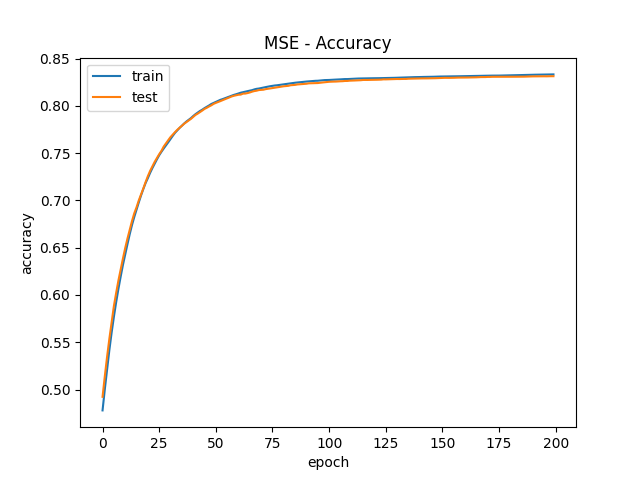
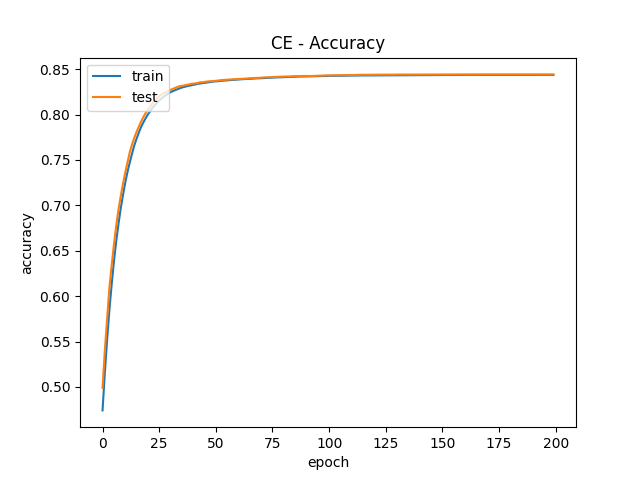
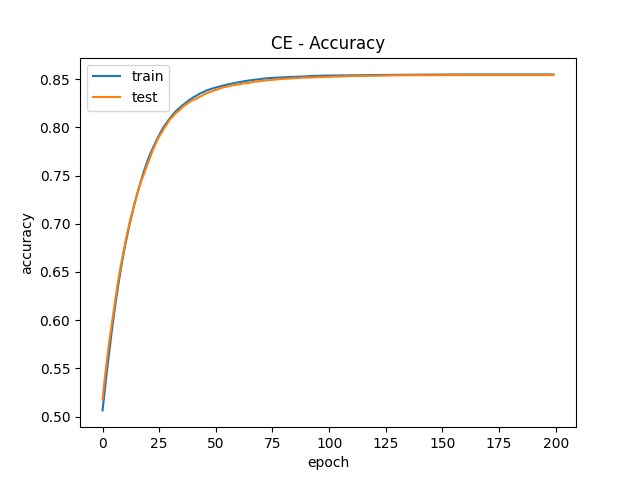
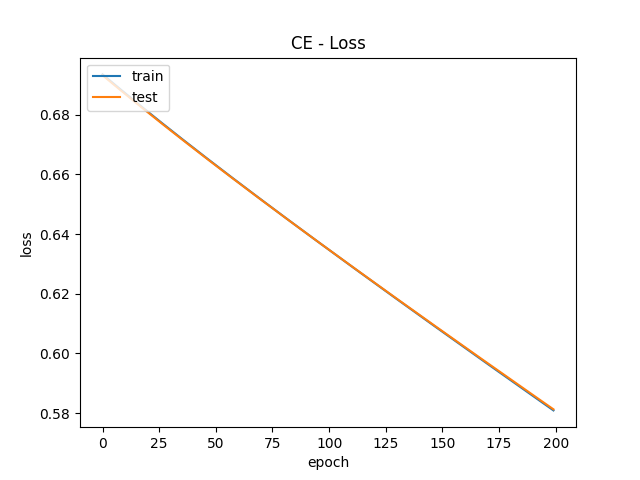
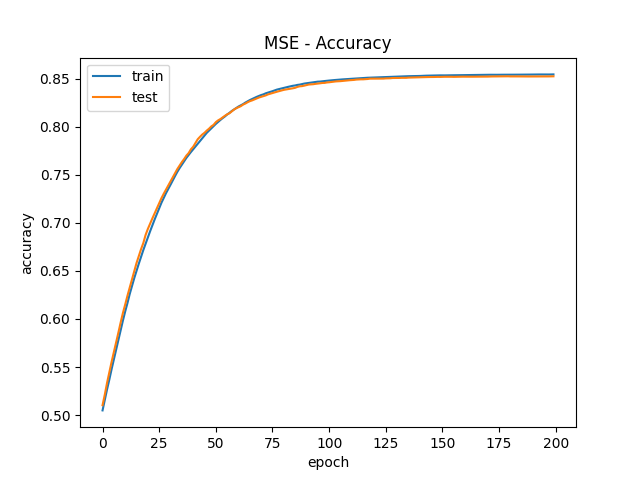
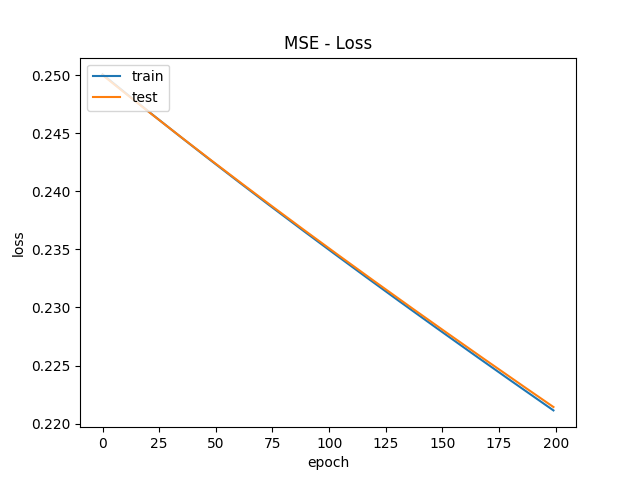
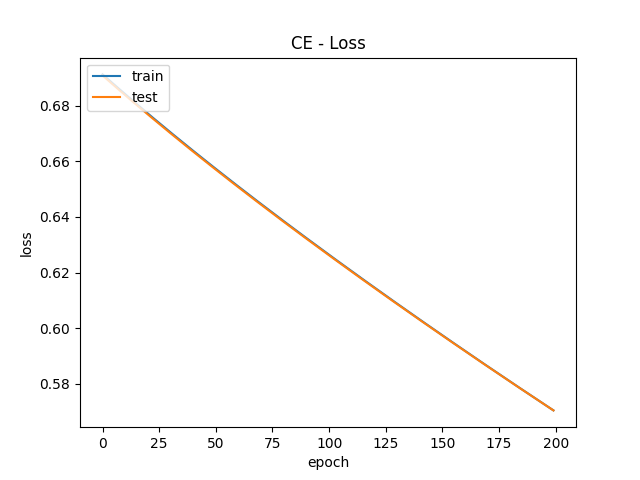
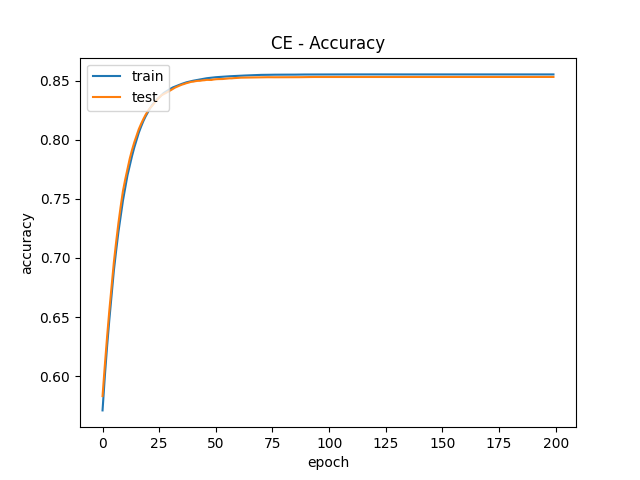
**β)** Η ταξινόμηση πολλαπλών ετικετών μπορεί να υποστηριχθεί απευθείας από νευρωνικά δίκτυα, προσδιορίζοντας τον αριθμό των ετικετών-στόχων που υπάρχουν στο πρόβλημα ως τον αριθμό των κόμβων στο επίπεδο εξόδου. Στη περίπτωση μας, το πρόβλημα έχει είκοσι (20) πιθανές ετικέτες εξόδου (κλάσεις), άρα για το επίπεδο εξόδου του νευρωνικού δικτύου απαιτούνται ***20 κόμβοι/νευρώνες εξόδου***.

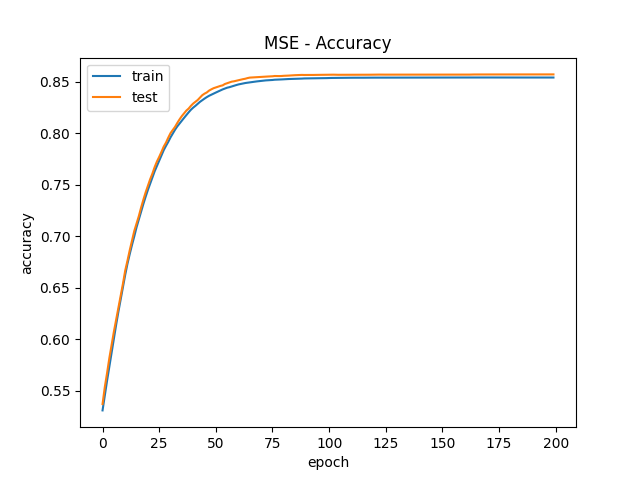
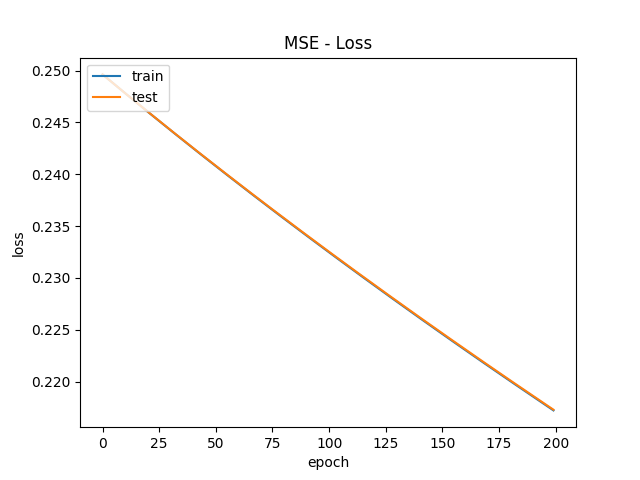
**γ)** Κάθε κόμβος στο κρυφό επίπεδο θα χρησιμοποιεί την **ReLU** συνάρτηση ενεργοποίησης. Τα χαρακτρηριστικά της μας φαίνοται ιδιαίτερα χρήσιμα στο συγκεκριμένο πρόβλημα καθώς οι τιμές των διανύσματων εισόδου αποτελούνται απο διακριτές στο εύρος [0, k], με k , δίνοντας έτσι στην εκάστοτε τιμή το βάρος που της αναλογεί, καθώς λέξη με μεγάλη συχνότητα έχει νόημα να εισαχθεί στο δίκτυο με μεγαλύτερο βάρος.

**δ)** Κάθε κόμβος στο επίπεδο εξόδου πρέπει να χρησιμοποιεί την **σιγμοειδή** συνάρτηση ενεργοποίησης. Ο βασικός λόγος είναι ότι οι πιθανότητες που παράγονται από τη σιγμοειδή είναι ανεξάρτητες και δεν περιορίζονται στο να αθροιστούν στο ‘1’.

**ε)**

|  |  |  |  |
| --- | --- | --- | --- |
| **#Νευρώνων στο κρυφό επίπεδο** | **CE Loss** | **MSE** | **Acc (CE/MSE)** |
| H1 = O = 20 | **0.5180** | **0.2104** | **0.8540 /** **0.8314** |
| H1=(I+O)/2=4270 | **0.5813** | **0.2214** | **0.8542 / 0.8523** |
| H1 = (I+O) = 8540 | **0.5704** | **0.2173** | **0.8531 / 0.8572** |

* ******20 νευρώνες κρυφού επιπέδου**
* **4270 νευρώνες κρυφού επιπέδου**
* **8540 νευρώνες κρυφού επιπέδου**

****

***Πόρισμα 2.1:***

Απο το παραπάνω πείραμα, διαπιστώνεται ότι με είκοσι (20) νευρώνες κρυφου επιπέδου ο αλγόριθμος αποδίδει καλύτερα, καθώς το loss είναι μικρότερο. Πολύ κοντά στη ποσότητα αυτή βρίσκονται οι δύο άλλες τιμές. Όσον αφορά το accuracy, και στις τρείς περιπτώσεις του H1 οι τιμές είναι πάρα πολύ κοντά.

***Πόρισμα 2.2:***

Όσον αφορά τη συνάρτηση κόστους, παρατηρείται ότι η CE είναι πιο «αυστηρή» απο την MSE. Αυτο συμβαίνει γιατί οι έξοδοι που θέλουμε να εκτιμήσουμε είναι τιμές στο εύρος [0, 1], και ο λογάριθμος είναι πολύ πιο ευαίσθητος σε αυτές απο το τετράγωνο. Ωστόσο, η CE συνάρτηση κόστους φαίνεται να βοηθά τη σύγκλιση της ακρίβειας του αλγορίθμου πιο γρήγορα απο την MSE και στις τρείς (3) περιπτώσεις του H1.

***Πόρισμα 2.3:***

Παρατηρούμε ότι η σύγκλιση της ακρίβειας του αλγορίθμου στις περιπτώσεις H1=20 και H1=8540 έχει ολοκληρωθεί μόλις απο την πεντηκοστή εποχή εκπαίδευσης (CE Loss function) και οι υπόλοιπες 150 συμβάλλουν μόνο στην ελαχιστοποίηση του loss. Παρόμοια είναι η συμπεριφορά και για H1=4270 με τη διαφορά οτι αργέι περισσότερο η σύγκλιση (κατα 25% πιο αργά). Αξίζει να παρατηρηθεί επίσης, οτι η σύγκλιση εκμάθησης του αλγορίθμου με CE είναι κατά 50 εποχές πιο γρήγορη απο την MSE.

***Πόρισμα 2.4:***

**Ωστόσο στο σημείο αυτό να πούμε οτι και στις τρείς περιπτώσεις ο αλγόριθμος υπερεκπεδεύεται. Αυτό το καταλαβαίνουμε απο τις γραφικές παραστάσεις του Loss, που φαίνεται να συνεχίζει να μειώνεται με την εμπειρία.**

|  |  |  |  |
| --- | --- | --- | --- |
| **#Νευρώνων στο κρυφό επίπεδο** | **CE Loss** | **MSE** | **Acc (CE/MSE)** |
| H2 = 15 | **0.6270** | **0.2327** | **0.8420 / 0.7922** |
| H2 = 20 | **0.6015** | **0.2245** | **0.8193 / 0.8248** |
| H2 = 40 | **0.6054** | **0.2278** | **0.8527 / 0.8504** |

* **Το πείραμα αυτό θα συνεχιστεί με H1 = 20**

**στ)**

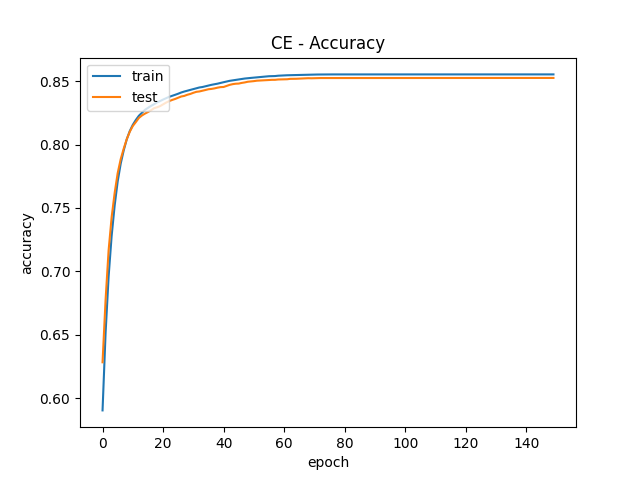
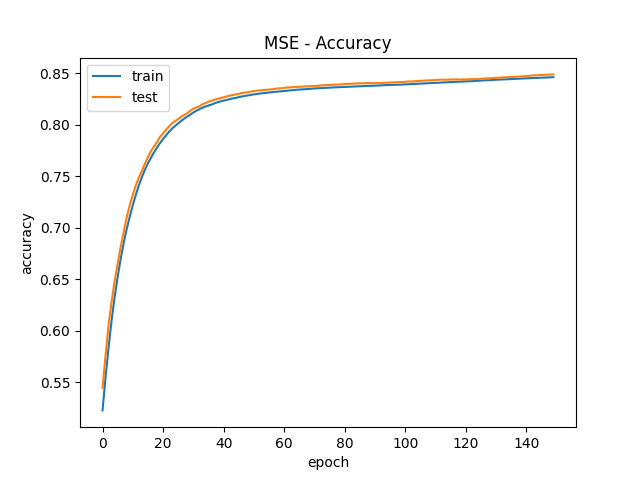
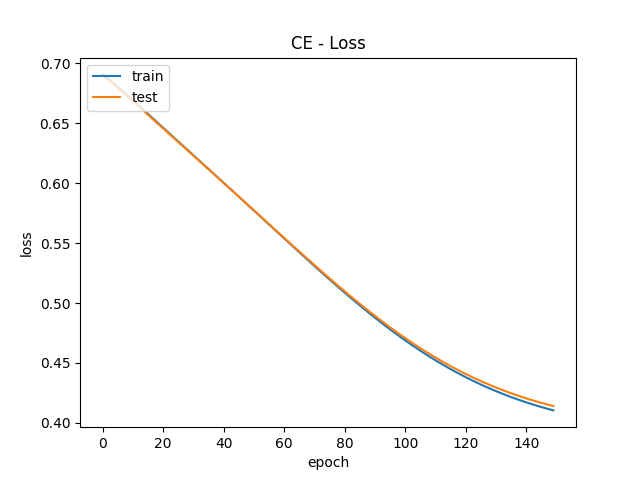
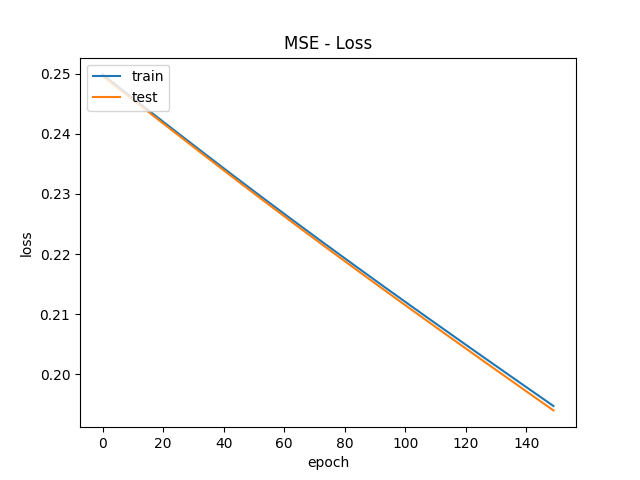
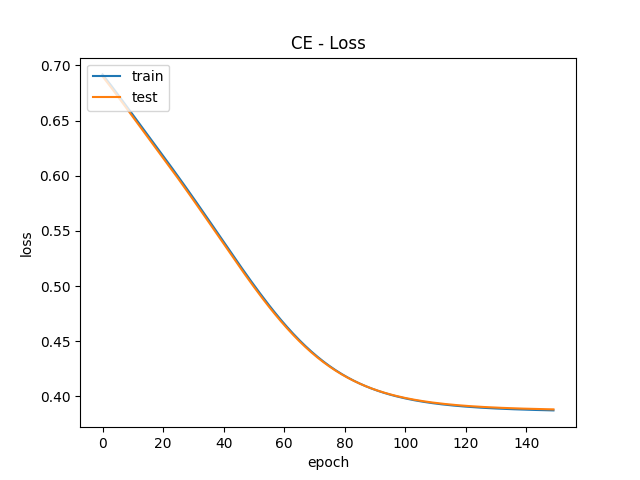
***Πόρισμα 2.4:***

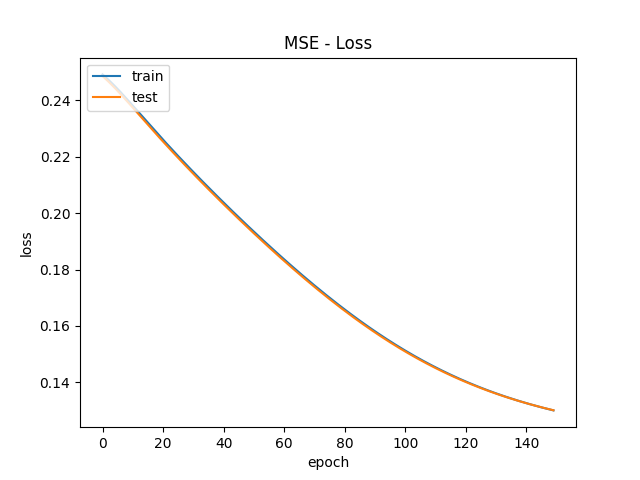
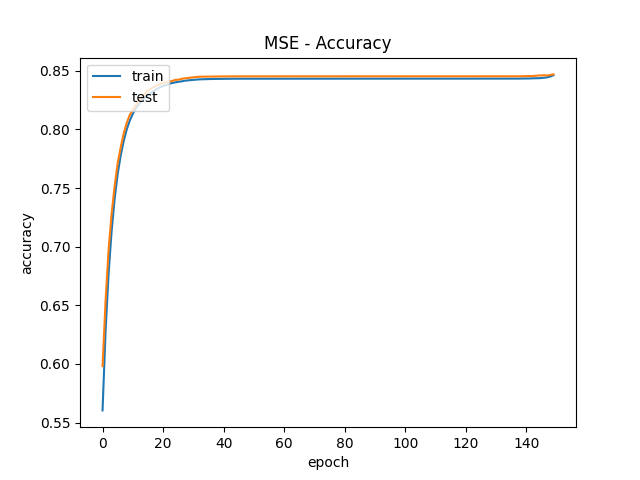
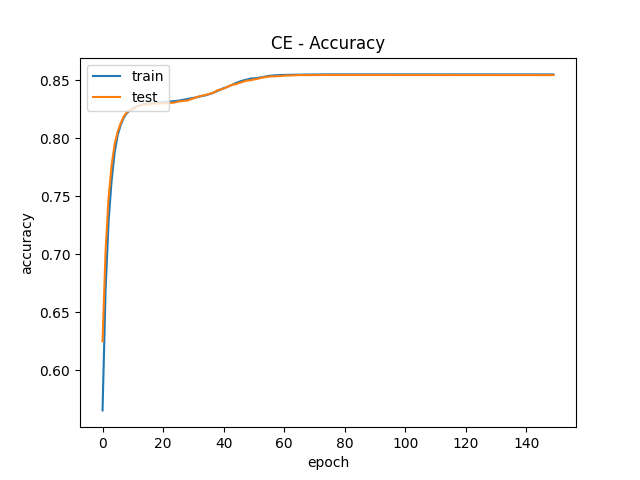
Απο το παραπάνω πινακάκι προκύπτει οτι η CE συμπεριφέρεται καλύτερα (πάλι). Όσον αφορά τον αριθμό νευρώνων του 2ου κρυφού επιπέδου φαίνεται για H1=H2 να αποδίδει καλύτερα. Ωστόσο, το 2ο κρυφό επίπεδο δε προσδίδει ουσιαστικό όφελος στην εκμάθηση του αλγορίθμου και για αυτό στα παρακάτω πειράματα θα γίνουν με αρχιτεκτονική 8520-20-20.

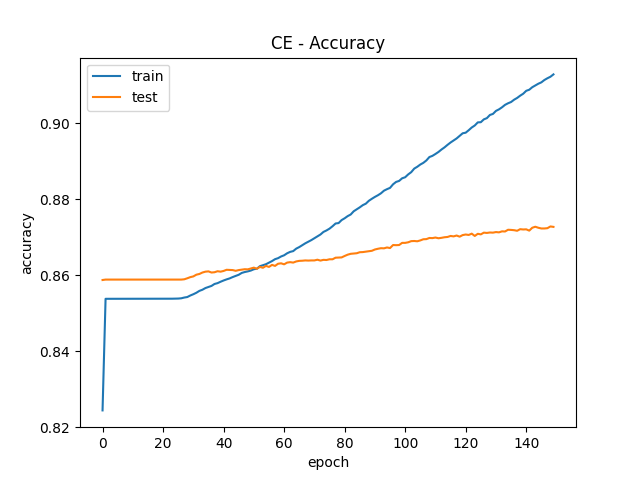
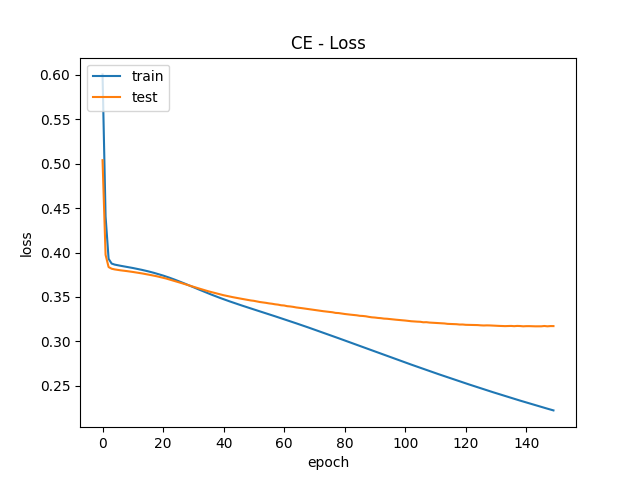
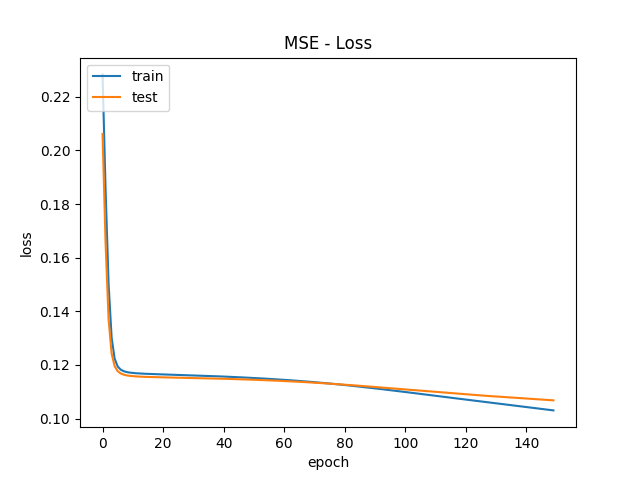
## ζ) ζ) Απο τις γραφικές σύκλισης, παρατηρούμε οτι ένα καλό κριτήριο τερματισμού που θα θέσουμε σε κάθε fold είναι το πρώιμο σταμάτημα οταν το loss του validation set σταματάει να μειώνεται.

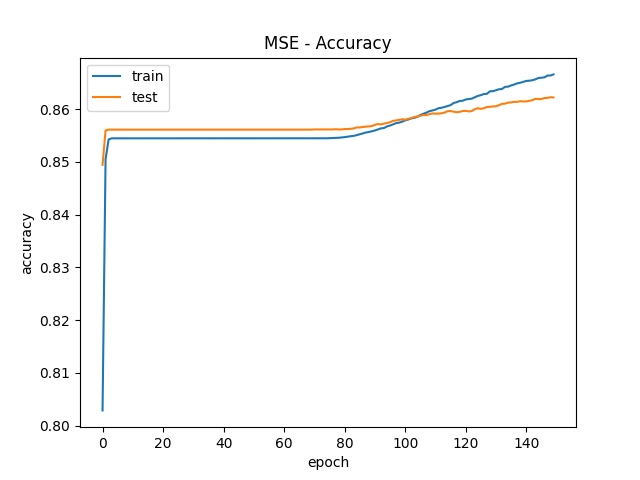
Η πρώιμη διακοπή θα μπορούσε να χρησιμοποιηθεί με διασταυρούμενη επικύρωση k-fold, αν και δεν συνιστάται. Η διαδικασία διασταυρούμενης επικύρωσης k-fold έχει σχεδιαστεί για να εκτιμήσει το σφάλμα γενίκευσης ενός μοντέλου επανατοποθετώντας και αξιολογώντας το επανειλημμένα σε διαφορετικά υποσύνολα ενός συνόλου δεδομένων. Η πρώιμη διακοπή έχει σχεδιαστεί για να παρακολουθεί το σφάλμα γενίκευσης ενός μοντέλου και να διακόπτει την εκπαίδευση όταν το σφάλμα γενίκευσης αρχίζει να υποβαθμίζεται. Είναι σε αντίθεση επειδή η διασταυρούμενη επικύρωση προϋποθέτει ότι δεν γνωρίζετε το σφάλμα γενίκευσης και η πρόωρη διακοπή προσπαθεί να σας δώσει το καλύτερο μοντέλο που βασίζεται στη γνώση του σφάλματος γενίκευσης.

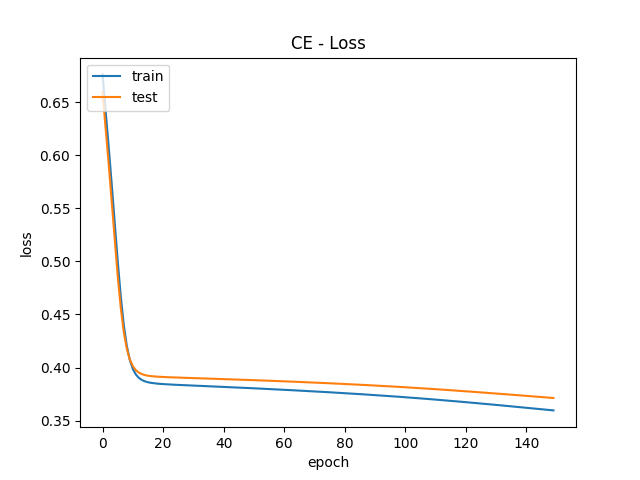
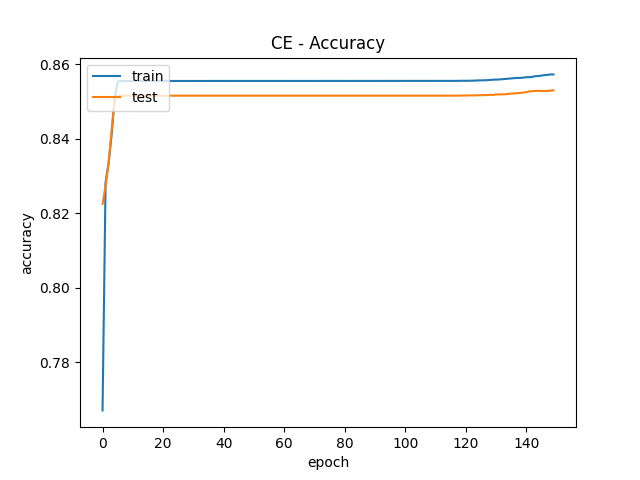
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **η** | **m** | **CE Loss** | **MSE** | **Acc (CE/MSE)** |
| 0.001 | 0.2 | **0.4809** | **0.1940** | **0.8543 / 0.8490** |
| 0.001 | 0.6 | **0.3892** | **0.1302** | **0.8534 / 0.8469** |
| 0.05 | 0.6 | **0.3172** | **0.1068** | **0.8727 / 0.8622** |
| 0.01 | 0.6 | **0.3713** | **0.1165** | **0.8531 / 0.8549** |

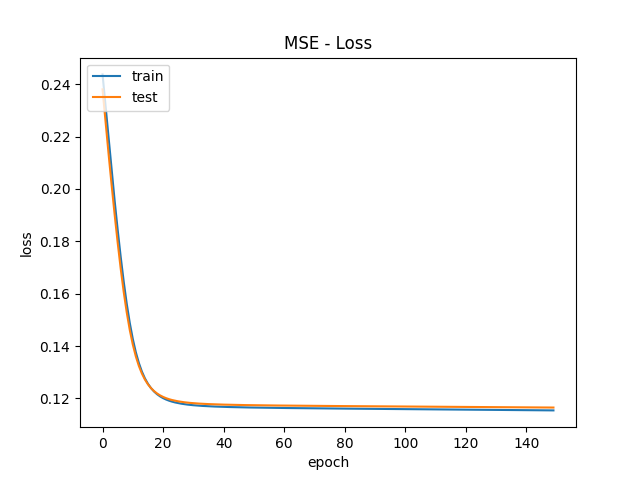
* (0.001,0.2)
* (0.001, 0.6)

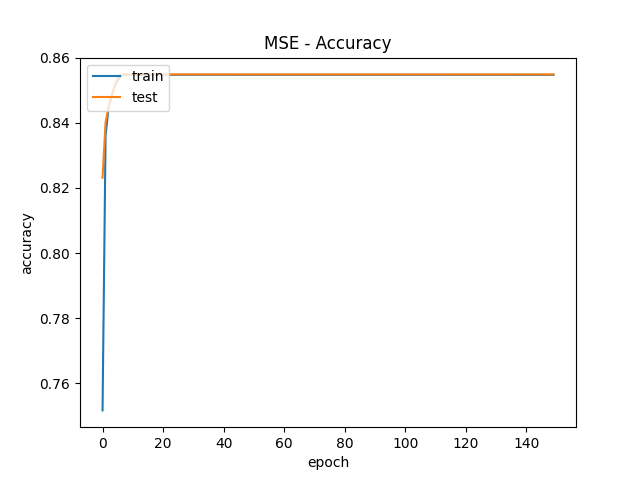
  
  
  
  
  


* (0.05, 0.6)



* (0.01, 0.6)





* Εάν θέσουμε την υπερπαράμετρο ορμής ‘1’ ή έστω πολύ κοντά στο 1 (π.χ. 0,99999) όταν χρησιμοποιήσουμε σαν optimizer SGD, τότε ο αλγόριθμος πιθανότατα θα αυξήσει πολύ την ταχύτητα, ελπίζοντας ότι θα κινηθεί περίπου προς το καθολικό ελάχιστο, αλλά η ορμή του θα το μεταφέρει ακριβώς πέρα ​​από το ελάχιστο. Μετά θα επιβραδύνει και θα επανέλθει, θα επιταχύνει ξανά, θα ξεπεράσει ξανά και ούτω καθεξής. Μπορεί να ταλαντωθεί με αυτόν τον τρόπο πολλές φορές πριν συγκλίνει, επομένως συνολικά θα χρειαστεί πολύ περισσότερος χρόνος για να συγκλίνει από ό,τι με μια μικρότερη τιμή ορμής.
* Από τα παραπάνω πειράματα διαπιστώνεται ότι ο αγλόριθμος ευνοείται κατά πολύ, καθώς βλέπουμε, συγκριτικά και με τα πειράματα του Α2, οτι για 150 εποχές, ο αλγόριθμος συγκλίνει, ενώ πριν οχι. Για (0.001, 0.2) δε παρατηρούμε ιδιαίτερη βελτίωση. Για m=0.6 η διαφορά βελτίωσης είναι εμφανής για όλες τις επιλογές του η, με καλύτερη απόδοση να να επιτυγχάνεται για το συνδυασμό (0.01, 0.6) καθώς η καμπύλες του loss έχουν σταθεροποιηθεί με κοινές τιμές. Στη περίπτωση (0.05, 0.6), η καθοδική πορεία μόνο του training set υποδυκνύει πώς πάλι ο αλγόριθμος υπερεκπεδεύεται.