

ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ ΜΕΡΟΣ Α’ 2021-2022

ΣΠΕΝΤΖΑΡΗΣ ΠΑΝΑΓΙΩΤΗΣ 1071110

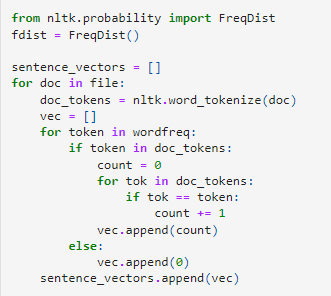
Υπολογιστική Νοημοσύνη

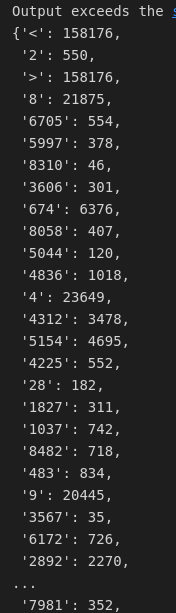
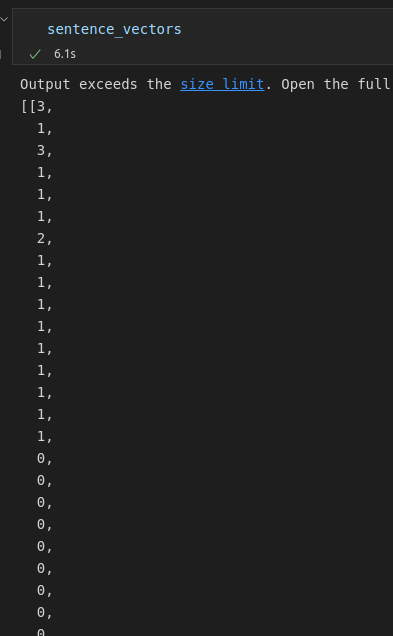
Κώδικας Link: <https://github.com/Pspetz/Text-Classification>

Α1. ΠΡΟΕΠΕΞΕΡΓΑΣΙΑ ΚΑΙ ΠΡΟΕΤΟΙΜΑΣΙΑ ΔΕΔΟΜΕΝΩΝ

**α) ΚΩΔΙΚΟΠΟΙΗΣΗ ΕΙΣΟΔΩΝ**

Ένα από τα σημαντικότερα βήματα στον τομέα της μηχανικής μάθησης είναι η σωστή προ επεξεργασία τον δεδομένων. Στην άσκηση μας ζητείται να εφαρμόσουμε την τεχνική BoW(back of words) όπου στην ουσία είναι μια αναπαράσταση κειμένου που περιγράφει την συχνότητα εμφάνισης των λέξεων μέσα σε ένα έγγραφο .Στην περίπτωση αυτή σαν πρώτο βήμα υπολογίζουμε την συχνότητα εμφάνισης κάθε λέξης από το αρχείο train-data,test-data και στην συνέχεια τα αναπαραστούμε σε ένα αραίο διάνυσμα.

  
 Wordfreq Dictionary: Sentence\_Vectors:

**β) ΠΡΟΣΑΡΜΟΓΗ ΔΕΔΟΜΕΝΩΝ ΣΕ ΔΙΑΦΟΡΕΤΙΚΗ ΚΛΙΜΑΚΑ**

Στην άσκηση μας ζητείται να εξετάσουμε τους τρόπου προσαρμογείς των τιμών σε ένα σύνολο δεδομένων και να εφαρμόσουμε τον καλύτερο για την περίπτωση μας.

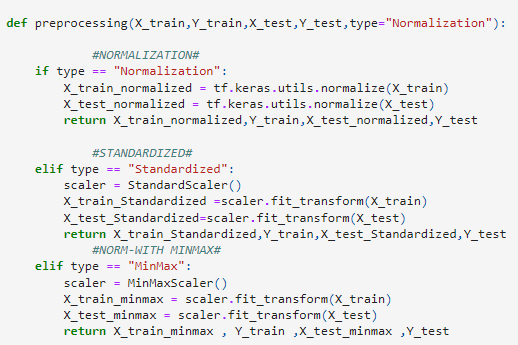
**Επεξήγηση**:

**Normalization**: Όταν εφαρμόσουμε την τεχνική του Normalization στα samples που διαθέτουμε, μετατοπίζουμε πρώτα την κλίμακα έτσι ώστε να ξεκινά από το 0 και στη συνέχεια τη συμπιέζουμε ώστε να τελειώσει στο 1. Αυτό γίνεται αφαιρώντας πρώτα την ελάχιστη τιμή και μετά διαιρώντας με τη νέα μέγιστη τιμή (η οποία είναι η παλιά μέγιστη τιμή μείον την παλιά ελάχιστη τιμή).Άρα καταλήγουμε όλα τα δείγματα μας να ανήκουν στην κλίμακα [0,1].

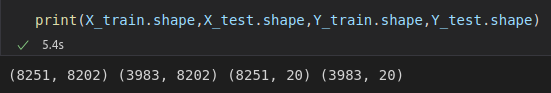
**Standardize**: Όταν εφαρμόσουμε την τεχνική του Standarization στα samples που διαθέτουμε, τότε κάθε μεταβλητή θα έχει μέσο όρο 0 και τυπική απόκλιση 1. Για να επιτευχθεί αυτό αρχικά αφαιρούμε τον μέσο όρο για να κεντράρουμε τη μεταβλητή και μετά διαιρούμε με την τυπική απόκλιση. Με αυτόν τον τρόπο η τυπική απόκλιση θα είναι 1.

**Centering**: Όταν εφαρμόσουμε την μέθοδο του Centering τότε αφαιρούμε τον μέσο όρο κάθε μεταβλητής ώστε ό μέσος όρος τελικά να μηδενιστεί. Επειδή όμως στο συγκεκριμένο project έχουμε εφαρμόσει την τεχνική του Bow και έχουμε την συχνότητα εμφάνισης κάθε λέξης δεν θα χρησιμοποιήσουμε centering γιατί δεν θα βοηθήσει τα δεδομένα μας.

* Στην περίπτωση της άσκησης μας πειραματίστηκα με την μέθοδο normalization kai standardize και κατέληξα πως η βέλτιστη προσαρμογή των δεδομένων μας είναι με την μέθοδο normalization μεταφέροντας τα δεδομένα μας στην κλίμακα [0,1], για αυτό τον λόγο χρησιμοποίησα την συνάρτηση **MinMaxScaler** από την βιβλιοθήκη scikit-learn.Στον κώδικα έχω δημιουργήσει μια συνάρτηση η οποία ανάλογα την μέθοδο προσαρμογής που του δώσουμε μας επιστρέφει σε κατάλληλη μορφή τα δεδομένα μας.Δεν χρησιμοποίησα την μέθοδο centering καθώς θα χαλάσει την μορφοποίηση των δεδομένων μας και δεν εξυπηρετεί τον σκόπο μας.Επίσης έχουμε πρόβλημα multilabel classification οπότε σίγουρα η μέθοδος centering δεν χρησιμοποιείται για αυτό τον στόχο.



* Επίσης τα δεδομένα μας τα έχω μορφοποιήσει κατάλληλα έχοντας το κατάλληλο shape για να μπορώ να τα περάσω αργότερα στο νευρωνικό δίκτυο.



**γ) ΔΙΑΣΤΑΥΡΟΥΜΕΝΗ ΕΠΙΚΥΡΩΣΗ(CROSS VALIDATION)**

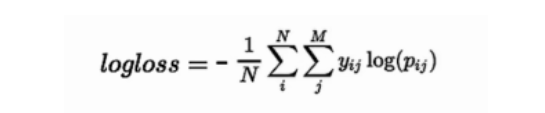
Η χρήση του cross validation δίνοντας του (5-fold) θα χωρίσει τα δεδομένα μας σε 5 υποσύνολα. Τα 4 εξ αυτών θα χρησιμοποιηθούν για την εκπαίδευση του νευρωνικού δικτύου μας ενώ το τελευταίο για το validation test.Το μοντέλο θα εκπαιδεύεται σε κάθε fold με διαφορετικά δεδομένα εκπαίδευσης και διαφορετικά δεδομένα επικύρωσης. Αυτό θα πραγματοποιείται 5 φορές όπου στο τέλος του κάθε fold θα υπολογίζεται η μέση απόδοση του μοντέλου μας .Αυτό θα βοηθήσει τόσο εμάς να υπολογίζουμε την απόδοση του με νέα testing δεδομένα όσο και το ίδιο στο να μπορεί να αποκτά την ικανότητα γενίκευσης.Έτσι κάθε φορά στο τέλος της εκπαίδευσης θα γνωρίζουμε μέσω της γραφικής παράστασης αν αποκλίνει η μπορεί να γενικεύση.

Α2. ΕΠΙΛΟΓΗ ΑΡΧΙΤΕΚΤΟΝΙΚΗΣ

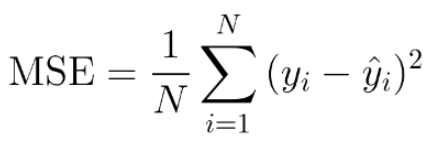
**α) ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΑΞΙΟΛΟΓΗΣΗ ΤΩΝ ΜΟΝΤΕΛΩΝ**

Το συγκεκριμένο πρόβλημα που καλούμαστε να αντιμετωπίσουμε είναι Multiclass Multilabel , αποτελείται από 20 κλάσεις(0-19 ετικέτες) άρα καταλαβαίνουμε ότι πλέον δεν έχουμε πρόβλημα πρόβλεψης μίας κλάσης εξ αυτών αλλά πολλές ταυτόχρονα αφού πχ(σε κάθε παράγραφο του κειμένου μπορεί να υπάρχουν παραπάνω από μία ετικέτες που το αντιπροσωπεύουν).

* **Loss function**: Οι συναρτήσεις κόστους ποσοτικοποιούν πόσο αποδοτικά έχει εκπαιδευτεί το νευρωνικό δίκτυο, χρησιμοποιώντας τα δεδομένα εκπαίδευσης. Οι συναρτήσεις κόστους εκφράζουν μία μέτρηση με βάση το σφάλμα που παρατηρείται στις προβλέψεις του δικτύου. Ο μέσος όρος των σφαλμάτων σε ολόκληρο το σύνολο δεδομένων εκφράζει πόσο κοντά είναι το μοντέλο σε ένα ιδανικό μοντέλο που δεν κάνει ποτέ λάθος. Η αναζήτηση αυτής της ιδανικής κατάστασης ισοδυναμεί με την εύρεση των παραμέτρων που θα ελαχιστοποιήσουν τη συνάρτηση κόστους. Με βάση αυτή τη λογική, οι συναρτήσεις κόστους χρησιμοποιούνται έτσι ώστε η αποδοτική εκπαίδευση του νευρωνικού δικτύου να ανάγεται σε ένα πρόβλημα βελτιστοποίησης. Υπάρχουν πολλές συναρτήσεις κόστους που χρησιμοποιούνται στα προβλήματα βελτιστοποίησης. Για προβλήματα παλινδρόμησης (regression) η πιο κοινή συνάρτηση κόστους είναι αυτή του μέσου τετραγωνικού σφάλματος (mean squared error – mse), ενώ σε προβλήματα κατηγοριοποίησης (classification) συνήθως χρησιμοποιείται η συνάρτηση εντροπίας (cross entropy).Άρα καταλαβαίνουμε από τώρα ότι σίγουρα η MSE δεν θα είναι αποδοτική σε σύγκριση με την Cross entropy.
* **Cross Entropy:** Θα χρησιμοποιήσουμε **binary\_crossentropy** καθώς έχουμε ένα πρόβλημα Multilabel.Επειδή έχουμε ταξινόμηση πολλαπλών ετικετών ,θέλουμε τον συνδυασμό πολλαπλών ανεξάρτητων δυαδικών ταξινομήσεων. Στην περίπτωση αυτή διαθέτουμε 20 κλάσεις με 20 δυαδικές ταξινομήσεις. Έτσι σε κάθε μια ταξινόμηση γίνεται εκπαίδευση ανεξάρτητα, άρα μπορούμε να παράγουμε πολλές ετικέτες για κάθε δείγμα.

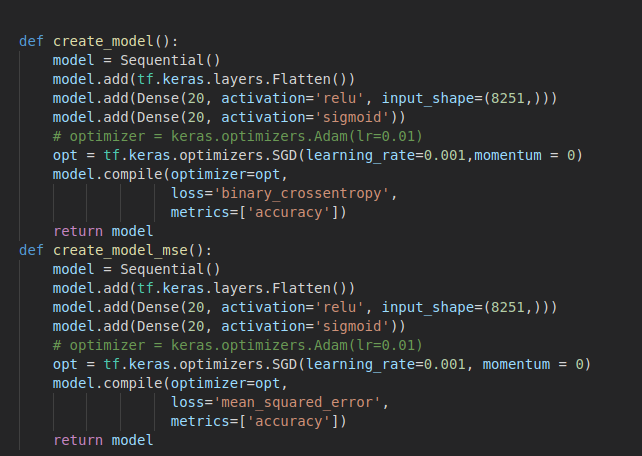


* **MSE:** Η MSE χρησιμοποιείται για την πρόβλεψη αριθμητικών τιμών και υπολογίζεται ως ο μέσος όρος των τετραγωνικών διαφορών μεταξύ των προβλεπόμενων και των πραγματικών τιμών και παίρνει πάντα θετική τιμή ενώ στην αντίθετη όψη το cross entropy παίρνει και αρνητικές τιμές .Στην συγκεκριμένη περίπτωση εάν ένα μοντέλο δυαδικής ταξινόμησης εκπαιδεύεται με τη συνάρτηση κόστους MSE, δεν είναι εγγυημένο ότι θα ελαχιστοποιείται η συνάρτηση κόστους του. Επίσης, η χρήση του MSE ως συνάρτησης κόστους προϋποθέτει την κατανομή Gauss, η οποία δεν ισχύει για τη δυαδική ταξινόμηση. Άρα καταλαβαίνουμε δεν είναι βέλτιστη για προβλήματα binary classification.



**β) ΝΕΥΡΩΝΕΣ ΕΞΟΔΟΥ**

Δεδομένου ότι βρισκόμαστε αντιμέτωποι με πρόβλημα Multiclass Multilabel classification θα πρέπει να διαχειριστούμε τις εξόδους διαφορετικά σε σχέση με ένα πιο απλό πρόβλημα που αντιπροσωπεύεται από μία ετικέτα ανά δείγμα. Στην περίπτωση μας θα χρησιμοποιήσουμε **20 νευρώνες εξόδου ,**καθώς στο κείμενο που διαθέτουμε παρατηρούμε ότι πάνω από μία ετικέτες αντιπροσωπεύουν κάθε δείγμα. Πιο συγκεκριμένα έχουμε 20 διαφορετικές κλάσεις για το πρόβλημα μας και οι αντιπροσωπευτικές ετικέτες είναι από (0-19).



Στο πρόβλημα μας μετά την προ επεξεργασία των δεδομένων καταλήξαμε σε μεγέθη , όπου τα δεδομένα προπόνησης έχουν μέγεθος 8251.Άρα **έχουμε input\_shape = 8251**

**γ) ΣΥΝΑΡΤΗΣΗ ΕΝΕΡΓΟΠΟΙΗΣΗΣ ΚΡΥΦΩΝ ΚΟΜΒΩΝ**

**ReLU:**Στην άσκηση μας ως συνάρτηση ενεργοποίησης **επιλέξαμε την ReLU**.Η ReLu για συντομία είναι μία τμηματικά linear function που εξάγει απευθείας την είσοδο εάν είναι θετική αλλιώς βγάζει μηδέν. Πλέον έχει γίνει η προεπιλεγμένη συνάρτηση ενεργοποίησης για πολλούς τύπους νευρωνικών δικτύων, επειδή το μοντέλο που την χρησιμοποιεί, είναι πιο εύκολο να εκπαιδευτεί και συχνά επιτυγχάνει καλύτερη απόδοση. Το αρνητικό που έχει η Relu είναι ότι όταν ο νευρώνας γίνει αρνητικός παίρνει τιμή 0 και μετά μπορεί να μην ανακάμψει ποτέ. Άρα αν χρησιμοποιήσουμε ReLu θα καταλήξουμε με πολλούς περιττούς ή νεκρούς κόμβους σε ένα νευρωνικό δίκτυο (αυτοί που έχουν αρνητική έξοδο) που δεν συμβάλλουν στο αποτέλεσμα γιατί δεν εκπαιδεύονται. H ReLU θεωρείται η καλύτερη συνάρτηση ενεργοποίησης επειδή έχει αποδειχθεί ότι λειτουργεί σε πολλές διαφορετικές καταστάσεις. Επειδή η κλίση της είναι είτε μηδενική είτε σταθερή, η συνάρτηση δεν παρουσιάζει συχνά το πρόβλημα της υπερβολικής αύξησης της κλίσης.

**LeakyReLU**:Στην άσκηση μας δοκίμασα να χρησιμοποιήσω μια παραλλαγή της ReLU, την LeakyReLU η οποία δεν μηδενίζει τις αρνητικές τιμές αλλά τους δίνει μια πολύ μικρή θετική τιμή .Επίσης είναι πιο γρήγορη στην φάση της εκπαίδευσης σε σύγκριση με την ReLU.Όμως με την δοκιμή της που έκανα πάνω στο νευρωνικό παρουσίασε χειρότερα αποτελέσματα σε σύγκριση με την κανονική ReLU.Οπότε θα χρησιμοποιήσουμε την ReLU.

**Linear:** Με χρήση Linear Function, υπάρχει μια γραμμική σχέση μεταξύ εισόδου και εξόδου, και η συνάρτηση που θέλουμε να μοντελοποιήσουμε είναι γενικά μη γραμμική και επομένως δεν μπορούμε να τη μοντελοποιήσουμε. Άρα προτιμούμε συναρτήσεις μη γραμμικές όπως η ReLU.

**δ) ΣΥΝΑΡΤΗΣΗ ΕΝΕΡΓΟΠΟΙΗΣΗΣ ΕΞΟΔΟΥ**

Στο συγκεκριμένο πρόβλημα **χρησιμοποίησα sigmoid**.H sigmoid function μας επιτρέπει να έχουμε μεγάλες πιθανότητες για όλες τις κλάσεις(0-19), μερικές από αυτές ή καμία από αυτές,δηλαδή είναι καλύτερη για χρήση σε προβλήματα που πάνω από μία ετικέτες αντιπροσωπεύουν το δείγμα μας.Ενώ αντίθετα η softmax χρησιμοποιείται για ταξινόμηση πολλών κλάσεων όπου υπάρχει μόνο μία αντιπροσωπευτική ετικέτα κάθε φορά.Η softmax επιβάλει ότι το άθροισμα των πιθανοτήτων των κλάσεων εξόδου μας είναι ίσο με μία από αυτές, επομένως για να αυξηθεί η πιθανότητα μιας συγκεκριμένης κλάσης, το μοντέλο πρέπει αντίστοιχα να μειώσει την πιθανότητα τουλάχιστον μιας από τις άλλες κλάσεις πράγμα που θα μειώσει την ακρίβεια που θα πετυχαίνει το νευρωνικό μας.

**ε) ΠΕΙΡΑΜΑΤΙΣΜΟΣ ΜΕ ΕΝΑΛΛΑΓΗ ΤΙΜΩΝ ΣΤΟ ΚΡΥΦΟ ΕΠΙΠΕΔΟ**

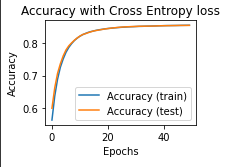
**Epochs**:Αρχικά θα χρησιμοποιήσουμε 30 εποχές στο μοντέλο μας.

**Optimizer:**Για optimizer θα χρησιμοποιήσουμε τον SGD.Παρόλο που πειραματίστηκα με την χρήση adam για optimizer και τα αποτελέσματα ήταν πολύ καλύτερα σε σύγκριση με τον sgd, ο adam δεν υποστηρίζει την χρήση momentum που ζητείται και στα επόμενα ερωτήματα. Άρα κατέληξα στην χρήση sgd με learning rate=0.001.

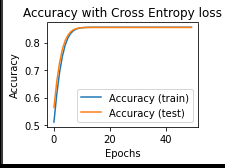
**Metrics:**Για μετρική στο πρόβλημα μας,στο μοντέλο που χρησιμοποιώ cross entropy στην **αρχή πειραματίστικα με την χρήση metrics=[‘binarry\_accurary’]** καθώς χρησιμοποιώ loss binary\_crossentropy και τα αποτελέσματα ήταν πολύ υψήλα και κατέληγα σε σύγκλιση με accuracy 0.8550.Όμως παρατήρησα ότι το loss δεν μειωνόταν παρόλο το υψηλό acc που πετύχαινε το μοντέλο πράγμα που είναι αναμενόμενο γιατί με την χρήση της metrics=[‘binary\_accuracy’] το μοντέλο μας δέχεται όλες τις τιμές άνω του 0.5 σαν έξοδο 1 ενώ όλες τις άλλες σαν 0.Οπότε αν έχουμε ταξινόμηση πολλών ετικετών όπως στην περίπτωση μας 20 labels και τα πιο πολλά είναι 0 στην έξοδο είναι αναμενόμενο να έχουμε υψηλό accuracy γιατί θα είναι κατά κάποιο τρόπο πλασματικό.Αυτό συμβαίνει γιατί το μοντέλο μας δεν θα κάνει μόνο πρόβλεψη ποιες ετικέτες αντιπροσωπεύουν το κείμενο αλλά θα δέχεται σωστά και όλα τα μηδέν.Οπότε υπάρχει περίπτωση πχ(άν έχουμε 3 ετικέτες σε 1 και 17 ετικέτες σε 0, αν τις κάνει λάθος και τις 3 αυτές ετικέτες που είναι στο 1 , θα έχουμε 14/20 σωστές προβλέψεις δηλαδή θα θεωρήσει σωστά όλα τα μηδέν που βρήκε,άρα το μοντέλο μας δεν εντοπίζει τις σωστές ετικέτες και παρόλα αυτά το accuracy θα είναι αρκετά υψηλό).**Για αυτό το λόγο κατέληξα σαν μετρική να χρησιμοποιήσω το default accuracy που παρέχει το keras δηλαδή metrics=[‘accuracy’].**

Παρακάτω παραθέτω screenshot με την χρήση της μετρικής binary\_accuracy:

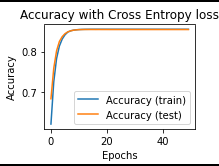
**20 ΝΕΥΡΩΝΕΣ ΣΤΟ ΚΡΥΦΟ ΕΠΙΠΕΔΟ**

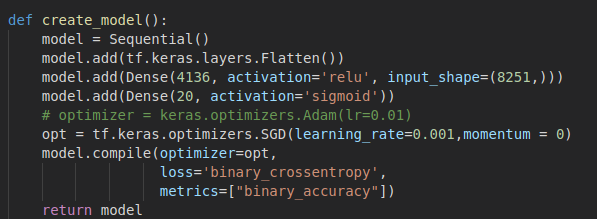
****

**4136 ΝΕΥΡΩΝΕΣ ΣΤΟ ΚΡΥΦΟ ΕΠΙΠΕΔΟ**

****

**8271 ΝΕΥΡΩΝΕΣ ΣΤΟ ΚΡΥΦΟ ΕΠΙΠΕΔΟ**

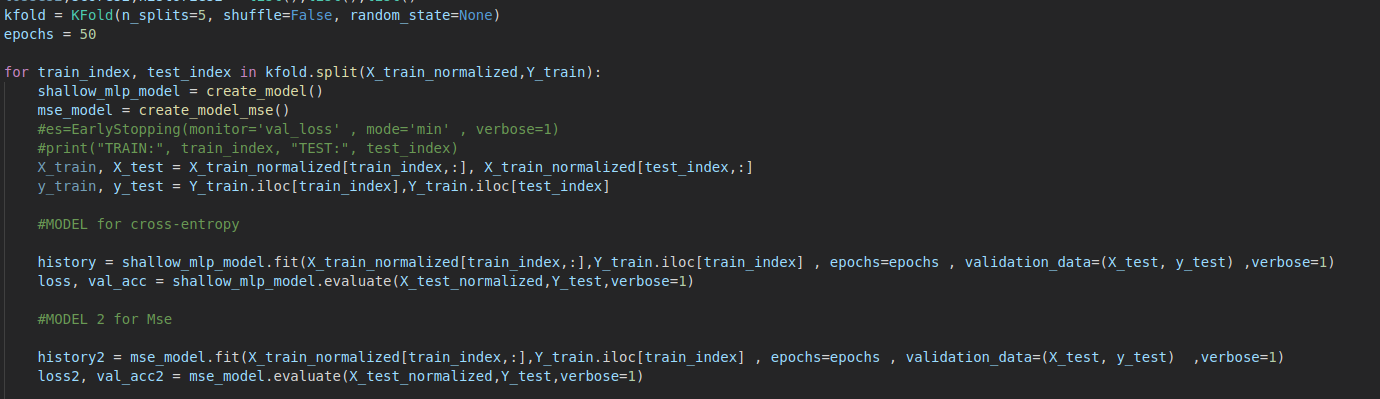
****

****

**Αλλά όπως προαναφέρθηκα κατέληξα να μην χρησιμοποιήσω metrics=[‘binary\_classification’] αλλά την απλή μετρική accuracy.**

ΠΑΡΑΚΑΤΩ ΓΙΝΕΤΑΙ ΕΠΕΞΗΓΗΣΗ ΤΟΥ ΚΩΔΙΚΑ ΠΟΥ ΕΧΩ ΔΗΜΙΟΥΡΓΗΣΕΙ ΓΙΑ ΤΗΝ ΧΡΗΣΗ KFOLDS.

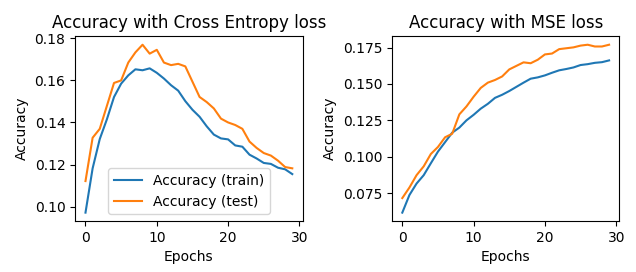
**ΕΠΕΞΗΓΗΣΗ**:

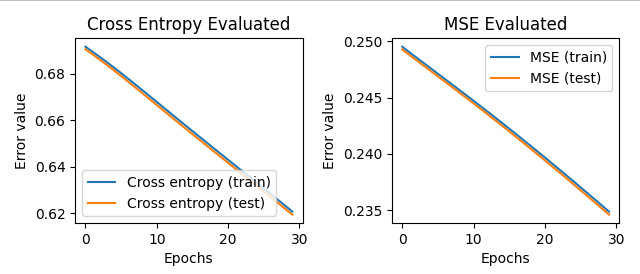


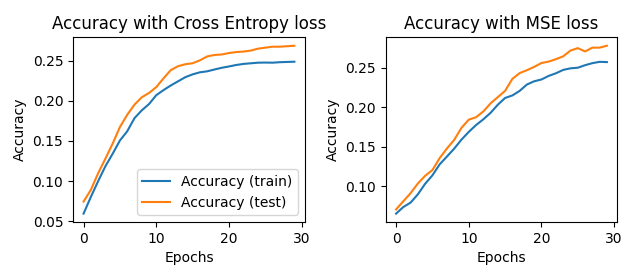
Κάθε φορά που εκτελείται ο κώδικας εκπαιδεύει 2 νευρωνικά δίκτυα το MSE και το CE ώστε να μειώσουμε τον συνολικό χρόνο εκτέλεσης του προγράμματος .Σε κάθε fold χρησιμοποιούμε ως validation\_data τα test data που προκύπτουν από τον διαχωρισμό σε 5 fold ώστε να μπορούμε να δούμε την δυνατότητα αν γενικεύει το μοντέλο μας με τα δεδομένα που δεν έχει εκπαιδευτεί. Στο evaluate χρησιμοποίησα τα έτοιμα test data από το αρχείο Delicious-MIL ώστε να ελέγξουμε τα αποτελέσματα για τις συναρτήσεις κόστους και την ακρίβεια στο test που δεν έχει λάβει μέρος στην εκπαίδευση του μοντέλου μας.

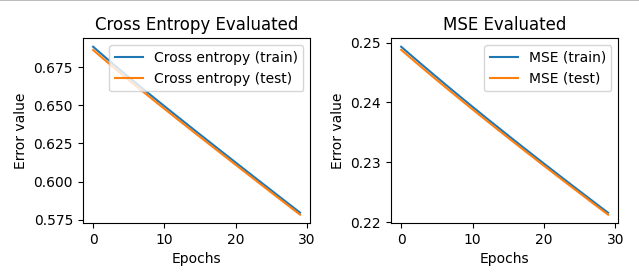
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Αριθμός νευρώνων στο κρυφό επίπεδο** | **CE loss** | **MSE loss** | **Acc CE** | **Acc MSE** |
| **H1=O =20** | 0.6286055326 | 0.2318877071 | 0.2004519194 | 0.0925935231 |
| **H1=(O+I)/2 = 8271/2=4136** | 0.5839186429 | 0.2218084663 | 0.2519206583 | 0.2377102702 |
| **H1=I+O = 8271** | 0.5725417375 | 0.2184330910 | 0.2564398765 | 0.2339944750 |

**20 ΝΕΥΡΩΝΕΣ ΣΤΟ ΚΡΥΦΟ ΕΠΙΠΕΔΟ**

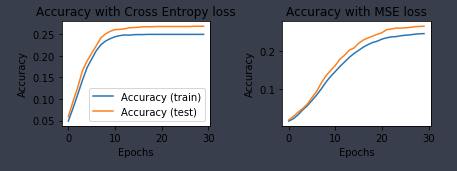


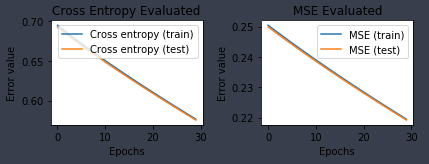


**4136 ΝΕΥΡΩΝΕΣ ΣΤΟ ΚΡΥΦΟ ΕΠΙΠΕΔΟ** 



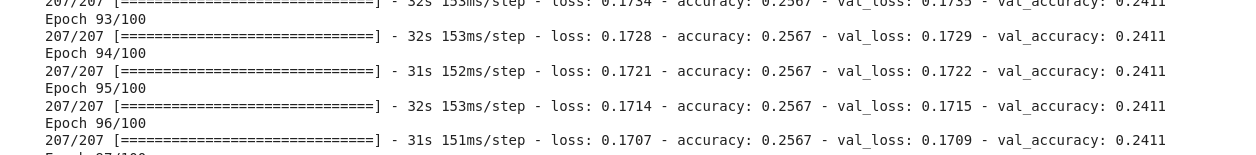
**8271 ΝΕΥΡΩΝΕΣ ΣΤΟ ΚΡΥΦΟ ΕΠΙΠΕΔΟ**





ΣΥΜΠΕΡΑΣΜΑ:

* Παρατηρούμε ότι το μοντέλο μας παρουσιάζει υπερεκπαίδευση με χρήση 20 νευρώνων στο κρυφό επίπεδο αυτό το καταλαβαίνουμε από την πτώση στην απόδοση του πάνω στην γραφική παράσταση ,άρα σίγουρα δεν θα κρατήσουμε 20 νευρώνες στο κρυφό επίπεδο.
* Με χρήση 4136 νευρώνες βλέπουμε ότι για 30 εποχές η ακρίβεια που πετυχαίνει το μοντέλο μας είναι σαφώς καλύτερη από την χρήση 20 νευρώνων καθώς επίσης παρατηρούμε ότι το μοντέλο οδεύει προς την σύγκλιση με την CE ενώ αντίθετα με την MSE θα συγκλίνει σίγουρα σε παραπάνω εποχές.Έπειτα από πείραμα που έγινε με ίδιο αριθμό νευρώνων αλλά 100 εποχές , η CE χρειαστήκε 35 εποχές για να συγκλίνει ενώ με χρήση MSE χρειαζέται 92 εποχές πράγμα υπερβολικά χρονοβόρο.



* Με χρήση 8271 νευρώνες παρατηρούμε ότι πετυχαίνει σχεδόν όμοια ακρίβεια με το νευρωνικό που έχει 4136 νευρώνες(ελαφρώς μικρότερη).Επίσης με 8271 νευρώνες το νευρωνικό επιβαρύνεται αρκετά και ξοδεύει πολύ περισσότερο χρόνο για να εκτελεστεί σε σύγκριση με τους 4136 νευρώνες πράγμα που δεν είναι αποδοτικό να χρησιμοποιήσουμε. Επίσης βλέπουμε ότι το μοντέλο μας με αυτό τον αριθμό νευρώνων συγκλίνει με CE ενώ με MSE τείνει να συγκλίνει αλλά χρειάζεται πάλι παραπάνω εποχές για να γίνει αυτό.

Άρα θα χρησιμοποιήσουμε στο κρυφό επίπεδο **4136 νευρώνες**.

* Παρατηρούμε ότι το accuracy που πετυχαίνει και στις 3 περιπτώσεις το Cross entropy είναι πολύ καλύτερο από το MSE πράγμα αναμενόμενο γιατί το MSE δεν συμπεριφέρεται το ίδιο καλά για binary classification,αλλά το loss του MSE φαίνεται να είναι μικρότερο με χρήση MSE αφού αργεί και παραπάνω να συγκλίνει. Άρα στο πρόβλημα η καταλληλότερη συνάρτηση κόστους είναι η Cross Entropy και αυτό το καταλαβαίνουμε και από τις γραφικές παραστάσεις .Παρόλα αυτά και η MSE είναι καλή ειδικά με δεδομένα τα οποία είναι κανονικοποιημένα και σίγουρα θα φτάσει σε σύγκλιση όπως προαναφέρθηκα παραπάνω, αλλά θα χρειαστεί πολύ περισσότερες εποχές.

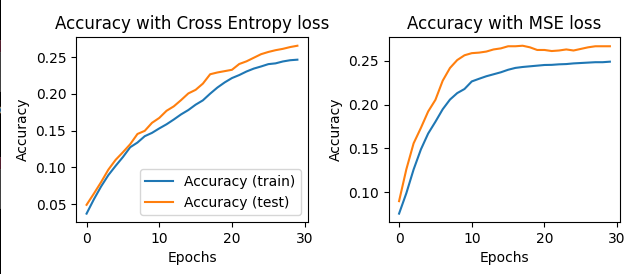
Άρα θα χρησιμοποιήσουμε σαν συνάρτηση κόστους την **Cross Entropy**

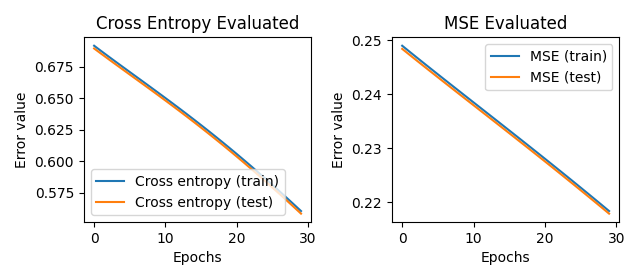
**στ) ΠΡΟΣΘΗΚΗ ΕΝΟΣ ΕΠΙΠΛΕΟΝ ΕΠΙΠΕΔΟΥ**

Στην φάση που βρισκόμαστε έχουμε κρατήσει το μοντέλο μας με ένα κρυφό επίπεδο με 4136 νευρώνες. Θα προσθέσουμε ακόμα ένα και θα πειραματιστούμε με λιγότερους νευρώνες, με ίδιους και με περισσότερους .Έπειτα από τα πειράματα που έγιναν για τον αριθμό των νευρώνων που παρουσιάζονται στον παρακάτω πίνακα παρατηρούμε τα εξής αποτελέσματα.

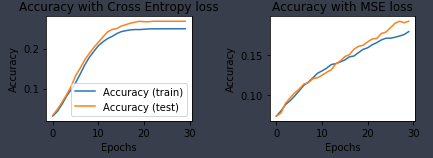
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Αριθμός νευρώνων στο κρυφό επίπεδο** | **CE loss** | **MSE loss** | **Acc CE** | **Acc MSE** |
| **H2=4136** | 0.5537485599 | 0.2170102864 | 0.2565403014 | 0.2412252038 |
| **H2=4236** | 0.5537143349 | 0.2176792085 | 0.2583479881 | 0.2314335912 |
| **H2= 2068** | 0.5534034967 | 0.2187744438 | 0.2581973433 | 0.1821240276 |

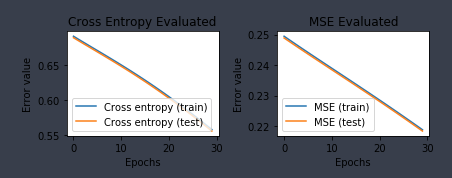
**4136 ΝΕΥΡΩΝΕΣ ΣΤΟ ΔΕΥΤΕΡΟ ΚΡΥΦΟ ΕΠΙΠΕΔΟ**



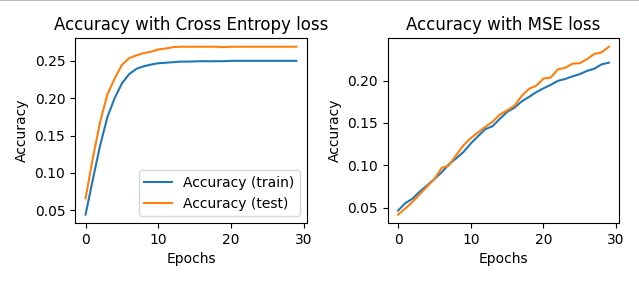


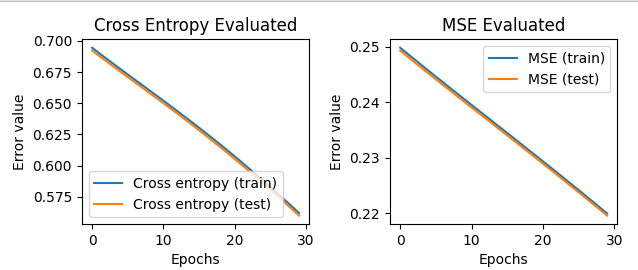
**2068 ΝΕΥΡΩΝΕΣ ΣΤΟ ΔΕΥΤΕΡΟ ΚΡΥΦΟ ΕΠΙΠΕΔΟ**





**4236 ΝΕΥΡΩΝΕΣ ΣΤΟ ΔΕΥΤΕΡΟ ΚΡΥΦΟ ΕΠΙΠΕΔΟ**





ΣΥΜΠΕΡΑΣΜΑ:

* Πλέον έχουμε προσθέσει ένα παραπάνω κρυφό επίπεδο πράγμα που αυξάνει την πολυπλοκότητα του μοντέλου μας και γίνεται ακόμα πιο χρονοβόρα λόγω της αύξησης των πράξεις που εκτελεί για να εκτελέσει το πρόβλημα μας. Όπως βλέπουμε και στις 3 περιπτώσεις παρόλο που θεωρητικά όταν μπαίνουμε σε πιο deep learning μοντέλα περιμένουμε μεγαλύτερη ακρίβεια στην συγκεκριμένη περίπτωση δεν έχουμε κάποιο όφελος.
* Με χρήση 2068 νευρώνες δηλαδή λιγότεροι νευρώνες στο δεύτερο κρυφό επίπεδο παρατηρούμε ότι με το CE συγκλίνει πολύ πιο γρήγορα ενώ στο MSE χρειάζονται πάλι παραπάνω εποχές για να φτάσει σε σύγκλιση .Επίσης παρατηρούμε και ότι η ακρίβεια του μοντέλου μας δεν είναι καλύτερη σε σύγκριση με την χρήση μόνο ενός επιπέδου αν συν υπολογίσουμε το παραπάνω χρόνο εκτέλεσης που χρειάζεται.
* Με χρήση ίδιου αριθμού νευρώνων(4136) και στα 2 επίπεδα παρατηρούμε ότι αυτή την φορά το MSE φτάνει πιο γρήγορα σε σύγκλιση σε αντίθεση με το CE.Παρόλα αυτά και σε αυτή την περίπτωση δεν έχουμε κάποια βελτίωση στην ακρίβεια που να δικαιολογεί τον χρόνο αναμονής που έχουμε
* Με χρήση παραπάνω νευρώνων(4236) παρατηρούμε ότι και πάλι δεν έχουμε κάποιο όφελος με την χρήση παραπάνω ενός παραπάνω επιπέδου νευρώνων.

Άρα καταλήγουμε να παραμείνουμε **με την χρήση ενός μόνο κρυφού επιπέδου** με 4136 νευρώνες.

**ζ) ΚΡΗΤΗΡΙΟ ΤΕΡΜΑΤΙΣΜΟΥ**

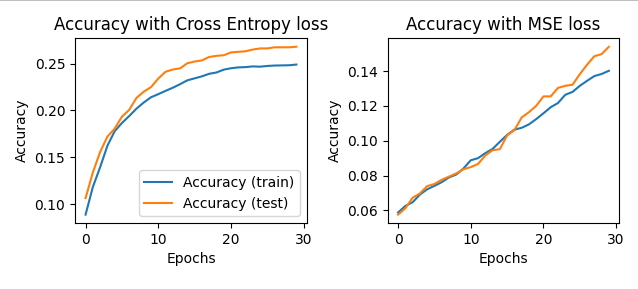
Το **EarlyStopping** χρησιμοποιείται για να αποτρέψει το overfitting του μοντέλου μας.Υπάρχει μεγάλη πιθανότητα αν αυξήσουμε και βάλουμε πολλές εποχές στο μοντέλο μας τότε το validation error θα αυξηθεί και ένας τρόπος αποφυγής της υπερκπαίδευσης που θα προκύψει είναι με χρήση του EarlyStopping.Επίσης στις γραφικές παραστάσεις που έχουμε στα προηγούμενα ερωτήματα παρατηρούμε ότι υπάρχει ένα πολύ μικρό ‘’τρεμόπαιγμα’’ της καμπύλης πράγμα που υποδηλώνει ότι αν αυξήσουμε τις εποχές είναι πολύ πιθανό να εμφανιστεί υπερεκπαίδευση όπως και στην περίπτωση με την χρήση 20 νευρώνων στο ερώτημα ε).Βέβαια ο καλύτερος τρόπος για να βεβαιωθούμε αν όντως το χρειαζόμαστε είναι μέσα από πείραμα που θα κάνουμε παρακάτω. Θα προπονήσουμε το μοντέλο μας με την χρήση του EarlyStopping και θα δούμε τα αποτελέσματα που παίρνουμε σε σύγκριση με το μοντέλο που έχουμε χωρίς χρήση πρόωρου τερματισμού.

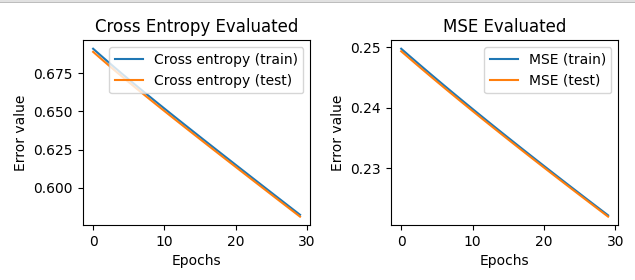
Όπως βλέπουμε στην παρακάτω εικόνα χρησιμοποιώ patience=5,άρα αν το μοντέλο μας δει ότι δεν υπάρχει κάποια βελτίωση στην ακρίβεια για 5 συνεχόμενες εποχές τότε θα σταματήσει την εκπαίδευση του.



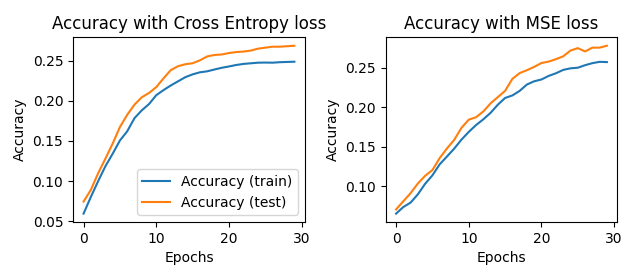
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Αριθμός νευρώνων** | **CE loss** | **MSE loss** | **CE acc** | **MSE acc** |
| **4136 & EarlyStopping** | 0.5842065509 | 0.2267713844 | 0.2561887979 | 0.1739392429 |
| **4136 without EarlyStopping** | 0.5839186429 | 0.2218084663 | 0.2519206583 | 0.2377102702 |

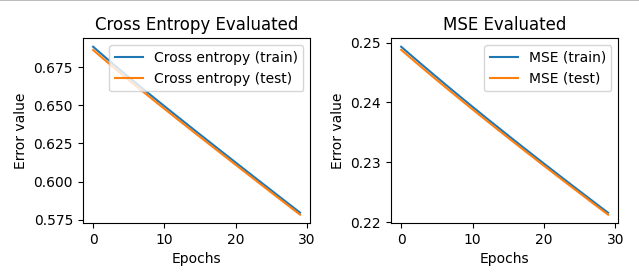
**4136 ΜΕ EARLYSTOPPING**





**4136 ΧΩΡΙΣ EARLYSTOPPING**

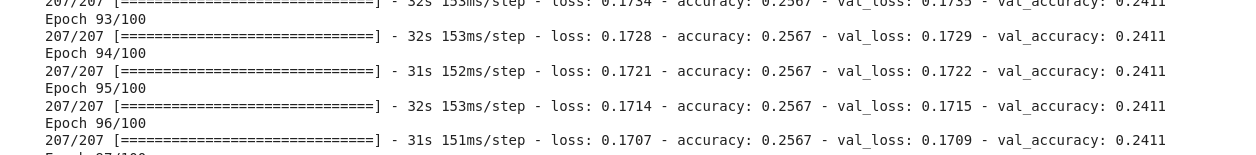




Συμπέρασμα:

* Επειδή χρησιμοποιούμε λίγες εποχές βλέπουμε ελάχιστες διαφορές.Εάν αυξήσουμε τις εποχές πάνω από 100 είναι πιθανό να υπάρχουν υπερεκπαίδευση σε μεγάλο βαθμό άρα το EarlyStopping θα μας βοηθήσει σε αυτή την περίπτωση.Το χρησιμοποιούμε καθώς βλέπουμε ότι τα αποτελέσματα που παράγει είναι ίδιας τάξης σε περίπτωση που δεν χρησιμοποιήσουμε το Earlystopping.

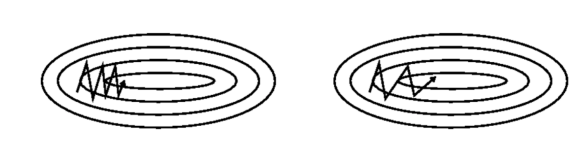
Επίσης δοκίμασα να τρέξω το μοντέλο σε 100 εποχές χωρίς την χρήση EarlyStopping κι μετά από 40 εποχές δεν υπήρχε κάποια βελτίωση ως προς το score που πετυχαίνουμε,άρα σίγουρα θα βοηθήσει η χρήση του Earlystopping.



Α3. ΜΕΤΑΒΟΛΕΣ ΣΤΟΝ ΡΥΘΜΟ ΕΚΠΑΙΔΕΥΣΗΣ ΚΑΙ ΣΤΑΘΕΡΑΣ ΟΡΜΗΣ

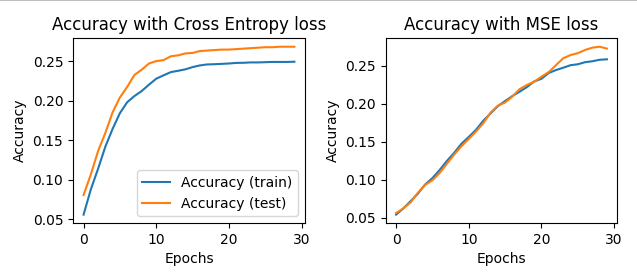
**Learning Rate:**Είναι ιδιαίτερα χρήσιμη καθώς αλλάζοντας την τιμή τους στην ουσία αλλάζουμε τον τρόπο που υπολογίζονται τα βάρη των νευρώνων και η διόρθωση των σφαλμάτων.Βάζοντας έναν μικρό αριθμό στο learning rate είναι σίγουρο ότι θα έχουμε πολύ μικρότερη πιθανότητα να συγκλίνει το μοντέλο μας άρα θα χρειαστούμε πολλές εποχές εκπαίδευσης για να γίνει αυτό.Αν χρησιμοποιήσουμε μεγάλο αριθμό εκπαίδευσης τότε θα έχουμε μια απότομη και πιο σύντομη σύγκλιση με πολύ λιγότερες εποχές αφού θα βρεθούμε γύρω από τις βέλτιστες τιμές που θα έχουν τα βάρη του μοντέλου μας.

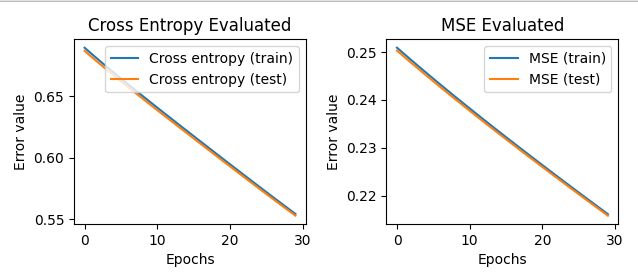
**Momentum:** Ο αλγόριθμος στοχαστικής μείωσης κλήσης (SGD) έχει πρόβλημα πλοήγησης στις «χαράδρες», δηλαδή σε περιοχές όπου η επιφάνεια καμπυλώνεται πολύ πιο απότομα σε μία διάσταση από ότι σε μία άλλη, οι οποίες είναι κοινές γύρω από το τοπικό βέλτιστο.Σε αυτές τις περιπτώσεις, ο ‘SGD’ ταλαντώνεται στις πλαγιές της χαράδρας ενώ παράλληλα κάνει αργή πρόοδο στην προσπάθεια προσέγγισης του τοπικού βέλτιστου.Στο παρακάτω σχήμα στα αριστερά φαίνεται η πρόοδος του SGD χωρίς όρμη ενώ δεξιά με όρμη. Η ορμή αυξάνεται για τις διαστάσεις των οποίων οι κλίσεις δείχνουν προς τις ίδιες κατευθύνσεις και μειώνουν τις ενημερώσεις για τις διαστάσεις των οποίων οι κλίσεις αλλάζουν κατευθύνσεις. Ως αποτέλεσμα, επιτυγχάνεται ταχύτερη σύγκλιση και μειωμένη ταλάντωση.



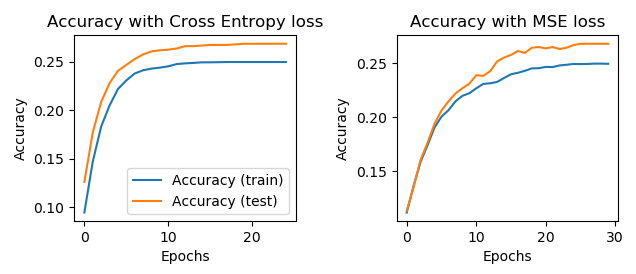
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **η** | **m** | **CE loss** | **MSE loss** | **CE acc** | **MSE acc** |
| **0.001** | **0.2** | 0.57635886669 | 0.21569296121 | 0.250414264 | 0.243384385 |
| **0.001** | **0.6** | 0.50416482090 | 0.18546475172 | 0.258247554 | 0.254230481 |
| **0.05** | **0.6** | 0.38715788722 | 0.11843857020 | 0.258347988 | 0.258347988 |
| **0.1** | **0.6** | 0.39282930493 | 0.11735013723 | 0.245593777 | 0.258347988 |

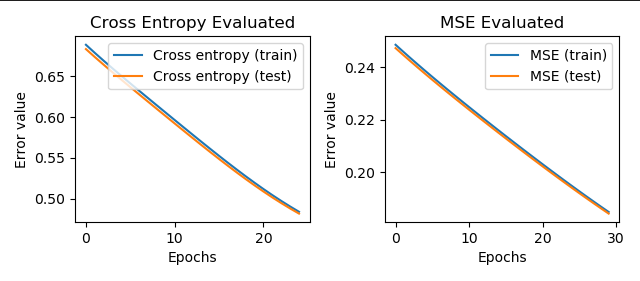
**Learning Rate=0.001 & Momentum=0.2**



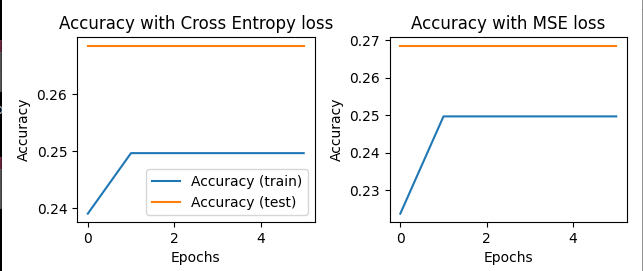


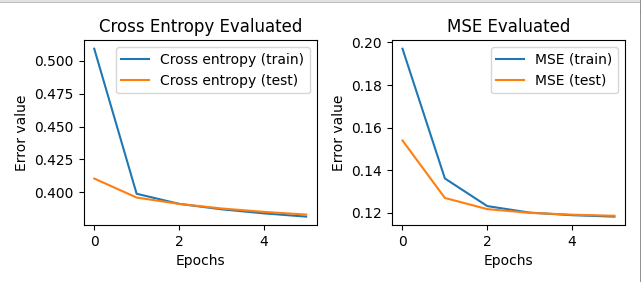
**Learning Rate=0.001 & Momentum=0.6**



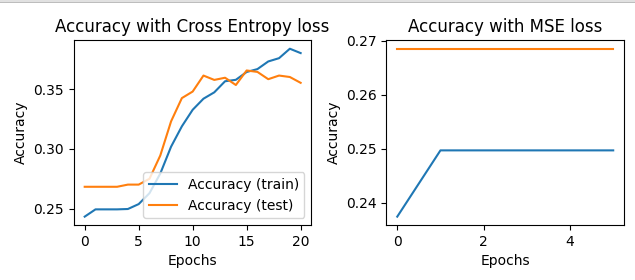


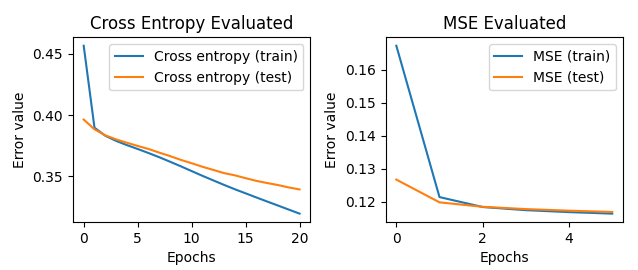
**Learning Rate=0.05 & Momentum=0.6**





**Learning Rate=0.1 & Momentum=0.6**





Παρατηρούμε ότι καθώς αυξάνουμε το learning rate το μοντέλο μας μαθαίνει πολύ πιο γρήγορα και πετυχαίνει μεγαλύτερη ακρίβεια και στις 2 περιπτώσεις του CE και του MSE.Επίσης καθώς αυξάνουμε το momentum παρατηρούμε ότι η σύγκλιση είναι πολύ πιο γρήγορη και σε λιγότερες εποχές. Οι αλλαγές που γίνονται στα βάρη του δικτύου όσο αυξάνουμε το learning rate είναι μεγάλες οπότε η εκπαίδευση που γίνεται μπορεί να ολοκληρωθεί και σε λιγότερες εποχές, για αυτό η χρήση του EarlyStopping σε αυτή την περίπτωση βοήθησε ώστε να αποφύγουμε την υπερκπαίδευση του μοντέλου μας. Θα χρησιμοποιήσω για το ερώτημα A4 το μοντέλο με learning rate=0.1 και momentum=0.6 καθώς βλέπουμε ότι το accuracy που πετυχαίνει σε σχέση με τα προηγούμενα είναι σαφώς καλύτερο κυρίως στο Cross Entropy.

Α4. ΟΜΑΛΟΠΟΙΗΣΗ

Ομαλοποίηση: Βασίζεται στην υπόθεση ότι ένα μοντέλο με μικρά βάρη είναι κατά κάποιο τρόπο απλούστερο από ένα δίκτυο με μεγάλα βάρη. Οι ποινές χρησιμοποιούνται για να κρατηθούν τα βάρη μικρά ή ανύπαρκτα (μηδέν). Μια εναλλακτική ονομασία στη βιβλιογραφία για τις ποινές βάρους είναι η «αποσύνθεση του βάρους» (weight decay), καθώς αναγκάζει τα βάρη να φθίνουν προς το μηδέν. Οι ομαλοποιήσεις που χρησιμοποιούνται κυρίως για τον περιορισμό των βαρών είναι η L1 και η L2

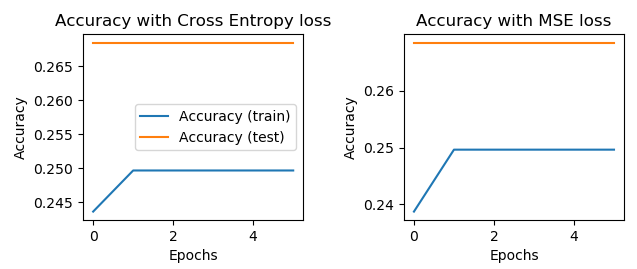
**Ομαλοποίηση L2:**

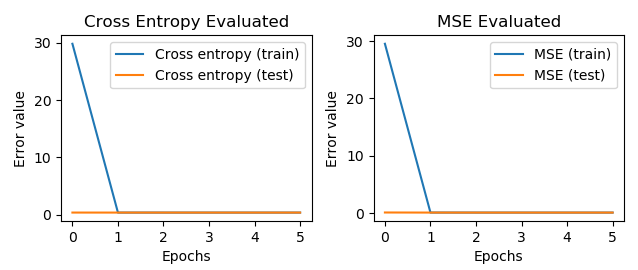
• Για κάθε βάρος w προστίθεται ο όρος 1 2 𝜆𝑤 2 στη συνάρτηση κόστους. Έτσι η συνάρτηση κόστους λαμβάνει τη μορφή: 𝐽(𝑥, 𝑦)𝑛𝑒𝑤 = 𝐽(𝑥, 𝑦)𝑜𝑙𝑑 + 0.5𝜆𝑤 2

• Τείνει να οδηγεί όλα τα βάρη σε μικρότερες τιμές.

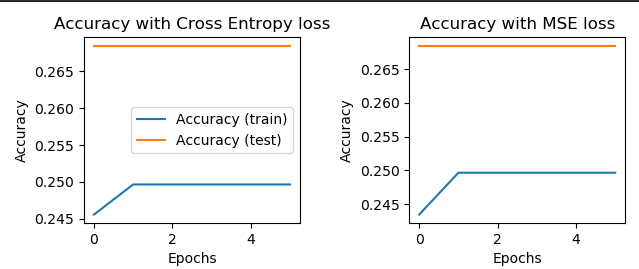
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Συντελεστής φθοράς** | **CE loss** | **MSE loss** | **CE acc** | **MSE acc** |
| **0.1** | 0.3867090344 | 0.1167973235 | 0.2583479881 | 0.258347988 |
| **0.5** | 0.3866612732 | 0.1167955264 | 0.2583479881 | 0.258347988 |
| **0.9** | 0.3867571175 | 0.1168013975 | 0.2583479881 | 0.258347988 |

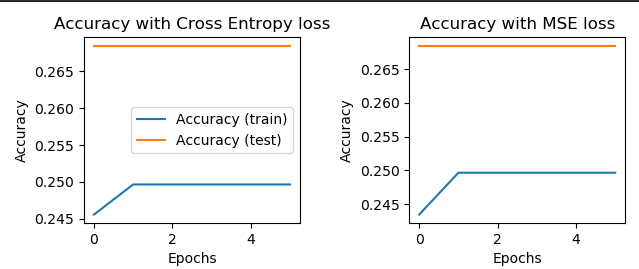
**R=0.1**



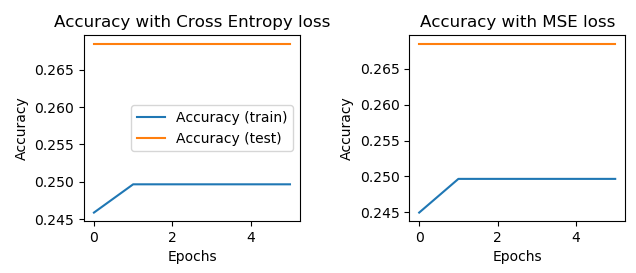


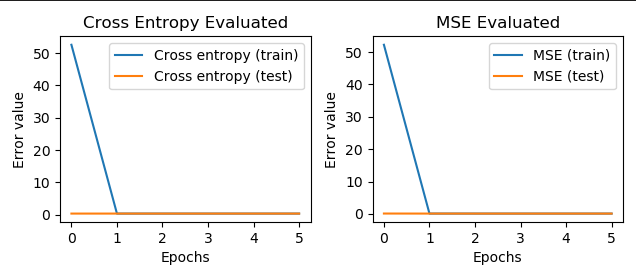
**R=0.5**





**R=0.9**





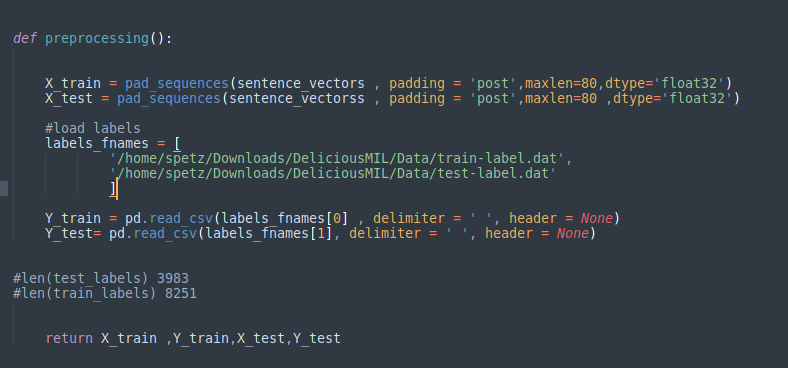
Συμπέρασμα: Παρατηρούμε ότι παρά την χρήση L2 ομαλοποίησης με 3 διαφορετικούς παραμέτρους r και στις 3 περιπτώσεις οι γραφικές είναι όμοιες με ελάχιστες αλλαγές στο CE loss.Επειδή το μοντέλο μας ήδη γενίκευε με την χρήση της L2 ομαλοποίηση το μοντέλο προσπαθεί να ενισχύσει την γενίκευση του, πράγμα που χρήση της L2 δεν έφερε κάποιο θετικό αποτέλεσμα. Αντιθέτως παρατηρούμε ότι και στις 3 περιπτώσεις η συνάρτηση κόστους CE αντιγράφει την MSE ενώ παρόλο που χωρίς την χρήση L2 το μοντέλο μας πετύχαινε πολύ καλύτερη ακρίβεια με την συνάρτηση κόστους CE έναντι της MSE.Επίσης βλέπουμε ότι σε κάθε περίπτωση η σύγκλιση γίνεται απευθείας. Γενικά αναφέρεται ότι το L2 είναι καλύτερο να παίρνει τιμές μεταξύ του 0.01 , 0.02…,0.1.Άρα ήταν αναμενόμενο ότι η χρήση του δεν θα βοηθούσε στην περίπτωση μας το μοντέλο που έχουμε.

Α5. ΕΝΣΩΜΑΤΩΣΕΙΣ ΛΕΞΕΩΝ

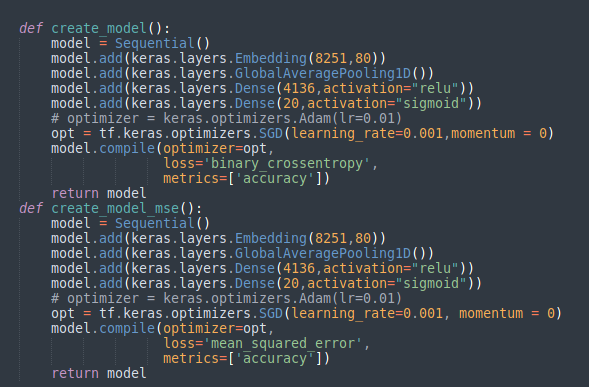
**α) Embedding**

Το επίπεδο ενσωμάτωσης μας δίνει τη δυνατότητα να μετατρέψουμε κάθε λέξη σε ένα διάνυσμα σταθερού μήκους και καθορισμένου μεγέθους. Το διάνυσμα που προκύπτει είναι ένα πυκνό με πραγματικές τιμές αντί για 0 ​​και 1. Το σταθερό μήκος των διανυσμάτων των λέξεων μας βοηθά να αναπαραστήσουμε τις λέξεις με καλύτερο τρόπο μαζί με μειωμένες διαστάσεις.Με αυτόν τον τρόπο η ενσωμάτωση του επιπέδου λειτουργεί σαν πίνακας αναζήτησης. Οι λέξεις είναι τα κλειδιά σε αυτόν τον πίνακα, ενώ τα πυκνά διανύσματα λέξεων είναι οι τιμές.

Για την υλοποίηση έχουμε κρατήση ακριβώς τον ίδιο τρόπο υπολογίζοντας σε ένα αραιό μητρώο την συχνότητα εμφάνισης κάθε λέξεις και στην συνέχεια εφαρμόζω padding με την χρήση της έτοιμης συνάρτησης που μας παρέχει το keras(pad\_sequences).

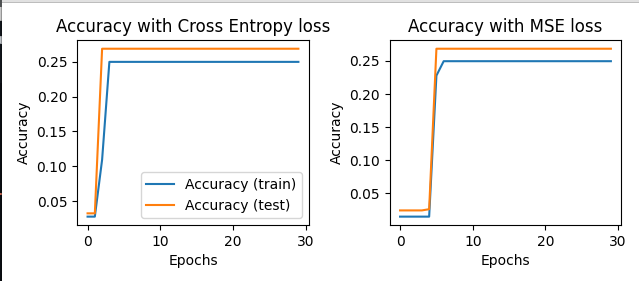


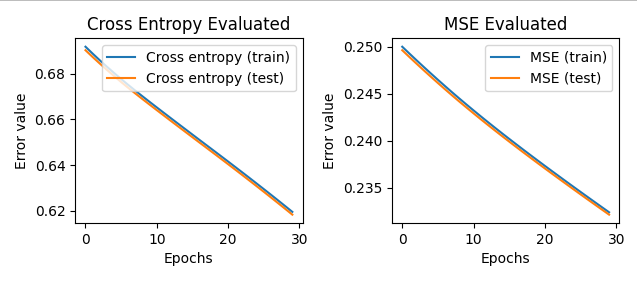
β)Θα χρησιμοποιήσουμε το μοντέλο με ένα κρυφό επίπεδο και 4136 νευρώνες και όπως παρατηρούμε στην παρακάτω εικόνα αυτή την φορά αντί για flatten() , έγινε χρήση της GlobalAveragePooling.



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Νευρώνες** | **CE loss** | **MSE loss** | **CE acc** | **MSE acc** |
| **4136** | 0.620357859 | 0.231812995 | 0.258347798 | 0.258347988 |

ΓΡΑΦΙΚΗ





Συμπέρασμα:

* Παρατηρούμε ότι με την χρήση μεθόδου embedding οι γραφικές παραστάσεις πλέον και στις 2 περιπτώσεις συγκλίνουν ταχύτατα.Όμως δεν υπάρχει κάποια αλλαγή στην ακρίβεια που επιτυγχάνουν , αλλά το MSE συγκλίνει στον ίδιο ακριβώς χρόνο με το CE ενώ στην αντίθετη περίπτωση δηλαδή στο ερώτημα Α2 η σύγκλιση του MSE γινόταν μετά από αρκετές εποχές.