ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ & ΠΛΗΡΟΦΟΡΙΚΗΣ

ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ

ΜΕΡΟΣ Α'

ΓΙΑ ΤΟ ΜΑΘΗΜΑ

ΥΠΟΛΟΓΙΣΤΙΚΗ ΝΟΗΜΟΣΥΝΗ

ΣΥΝΕΡΓΑΤΙΚΟ ΦΙΛΤΡΑΡΙΣΜΑ ΜΕ ΧΡΗΣΗ ΝΕΥΡΩΝΙΚΩΝ ΔΙΚΤΥΩΝ ΓΙΑ ΣΥΣΤΑΣΕΙΣ ΤΑΙΝΙΩΝ

ΜΑΝΤΑΣ ΕΛΕΥΘΕΡΙΟΣ

Α.Μ. 1047128

Github link: https://github.com/LefterisMantas/Computational-Intelligence

ΠΑΤΡΑ 2020

Εισαγωγή:

Για την υλοποίηση των ερωτημάτων του project χρησιμοποιήθηκε η γλώσσα προγραμματισμού **Python** και τα εργαλεία **SciKit-Learn & Keras.**

Α1. Προεπεξεργασία και Προετοιμασία δεδομένων

α) Κεντράρισμα (*centering):*

Ο κώδικας για το ζητούμενο αυτό παρατίθεται ως **centering.py**

Το κεντράρισμα ουσιαστικά «σπάει» τη συμμετρία κατά τη μάθηση του δικτύου. Προκειμένου να μην είναι όλες οι τιμές θετικές και έτσι πάντα η κλίση να είναι θετική, προσαρμόζουμε τις τιμές γύρω από το 0. Στο ζήτημα αυτό ομαδοποιήθηκαν τα δεδομένα ως προς το χρήστη και ύστερα αφαιρέθηκε η μέση τιμή της βαθμολογίας του καθενός από τις αντίστοιχες βαθμολογίες. Παρατηρήθηκε πως το **νέο διάστημα** τιμών αξιολόγησης είναι το **[-4,4].**

Σημείωση: Η μέθοδος αυτή θα ήταν χρήσιμη εάν οι απόκλιση των τιμών ήταν πολύ μεγάλη, κάτι που δεν ισχύει για το πρόβλημά μας. Η μέθοδος αυτή υλοποιήθηκε αλλά κρίθηκε σκόπιμο να μην εφαρμοστεί στο υπόλοιπο της εργασίας.

β) Ελλιπείς τιμές (missing values):

Ο κώδικας για το ζητούμενο αυτό παρατίθεται ως **missing_values.py**

Για το ερώτημα αυτό στις θέσεις των μη υπαρχουσών εγγραφών (βαθμολογιών) προστέθηκε ο μέσος όρος, του διανύσματος αξιολογήσεων του κάθε συγκεκριμένου χρήστη, ως τιμή αξιολόγησης. Ύστερα όλες οι νέες και παλιές τιμές γράφονται στο νέο αρχείο **data.csv** που ύστερα χρησιμοποιείται για την εκπαίδευση του νευρωνικού δικτύου.

γ) Κανονικοποίηση (*rescaling):*

Στο ζήτημα αυτό χρησιμοποιήθηκε η συνάρτηση **fit_transform** του **MinMaxScaler** του **Sci-kit learn** ώστε να κανονικοποιηθούν τα δεδομένα στο διάστημα [0,1].

δ) Διασταυρούμενη Επικύρωση (*cross-validation):*

Για την υλοποίηση του ερωτήματος αυτού χρησιμοποιήθηκε η βιβλιοθήκη **Kfold του Sci-kit learn.** Χωρίσαμε τα δεδομένα σε 5 set train-test. Η διαδικασία μάθησης ακολουθείται 5 φορές, όπως ορίζεται για το 5-fold, παίρνοντας κάθε φορά κομμάτια του πλήρους συνόλου δεδομένων για training και testing. Συγκεκριμένα παίρνουμε 80% στο train και 20% στο test set.

Α2. Επιλογή Αρχιτεκτονικής

α) Οι δύο **μετρικές** που θα χρησιμοποιηθούν είναι

- **RMSE** (Ρίζα Μέσου Τετραγωνικού Σφάλματος):
  Υπολογίζει το τετράγωνο της διαφοράς μεταξύ της επιθυμητής και της προβλεπόμενης τιμής κάθε εξόδου και βρίσκει το μέσο όρο, ύστερα υπολογίζει τη τετραγωνική ρίζα αυτού. Ουσιαστικά υπολογίζει τη τετραγωνική ρίζα του μέσου σφάλματος.
  Μέσω του RMSE μπορούμε να εντοπίσουμε τα πολύ μεγάλα σφάλματα.
  Το RMSE αυξάνεται με βάση τη κατανομή συχνότητας του σφάλματος.
  Ο τύπος υπολογισμού του Μέσου Απόλυτου Σφάλματος είναι ο παρακάτω:

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

- **MAE** (Μέσο Απόλυτο Σφάλμα): Μετράει το μέσο σφάλμα σε ένα σύνολο από προβλέψεις. Υπολογίζει την απόλυτη διαφορά μεταξύ της επιθυμητής και της προβλεπόμενης τιμής κάθε εξόδου και βρίσκει το μέσο όρο.
  Ο τύπος υπολογισμού του Μέσου Απόλυτου Σφάλματος είναι ο παρακάτω:

$$\mathrm{MAE} = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j|$$

β) Ο αριθμός των νευρώνων **εισόδου** στο ΤΝΔ θα είναι ακριβώς όσοι και οι χρήστες, δηλαδή **943**. Θα χρησιμοποιηθεί τεχνική **one-hot encoding**. Συγκεκριμένα, όταν επιλέγουμε να βρούμε τις αξιολογήσεις για ένα συγκεκριμένο χρήστη μόνο η αντίστοιχη είσοδος θα ενεργοποιείται και όλες οι υπόλοιπες θα συνεισφέρουν με 0 άρα δε θα λαμβάνονται υπόψιν. Η one-hot encoding μέθοδος χρησιμοποιείται γιατί μας βοηθάει η αραιή δομή του μητρώου (διαγώνιο με 1 στην διαγώνιο και 0 στις άλλες θέσεις).

γ) Στην **έξοδο** του ΤΝΔ θα χρειαστούμε **1682** νευρώνες, ακριβώς όσες είναι οι ταινίες του συνόλου δεδομένου. Αυτό συμβαίνει διότι θέλουμε ως έξοδο να παίρνουμε τις αξιολογήσεις όλων των ταινιών για τον ζητούμενο χρήστη.

δ) Η κατάλληλη συνάρτηση ενεργοποίησης για το κρυφό επίπεδο νευρώνων είναι η **σιγμοειδής συνάρτηση (sigmoid)**. Δε μπορούμε να χρησιμοποιήσουμε τη Relu, η οποία είναι η πλέον ευρέως χρησιμοποιούμενη συνάρτηση, στο κρυφό επίπεδο διότι οι τιμές μας έχουν κανονικοποιηθεί στο διάστημα [0,1]. Επομένως η κατάλληλη συνάρτηση ενεργοποίησης είναι η σιγμοειδής, που σαν έξοδο έχει 0 ή 1 ανάλογα με την είσοδο. Έτσι, μία από αυτές τις 2 τιμές περνάει στο επίπεδο εξόδου του δικτύου.

ε) Η καταλληλότερη συνάρτηση ενεργοποίησης για το επίπεδο εξόδου είναι η **γραμμική συνάρτηση (linear).** Δε μπορούμε να χρησιμοποιήσουμε τη Relu στο επίπεδο εξόδου διότι οι τιμές μας έχουν κανονικοποιηθεί στο διάστημα [0,1]. Η έξοδος έτσι αναγνωρίζει τις τιμές τοποθετώντας αυτές στο κατάλληλο διάστημα.

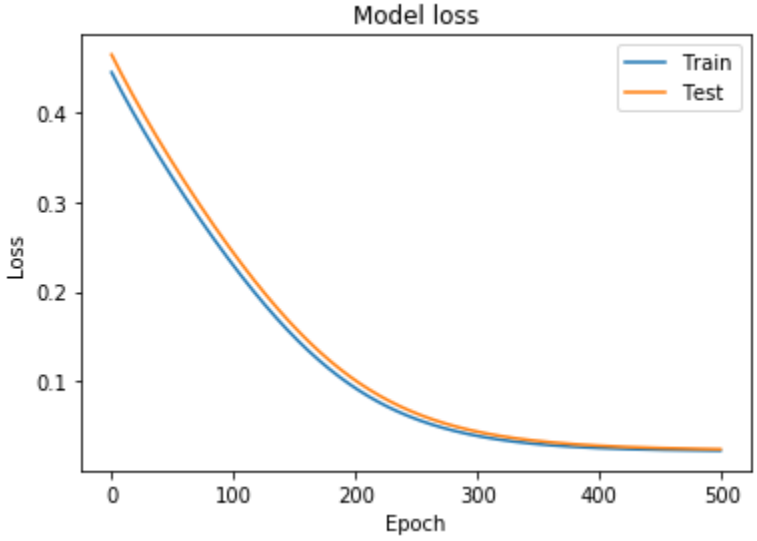
Ως απόδειξη του παραπάνω συνδυασμού συναρτήσεων ενεργοποίησης καθ' όλη την διάρκεια των πειραμάτων ήταν η καλύτερη απόδοση του δικτύου.

στ)

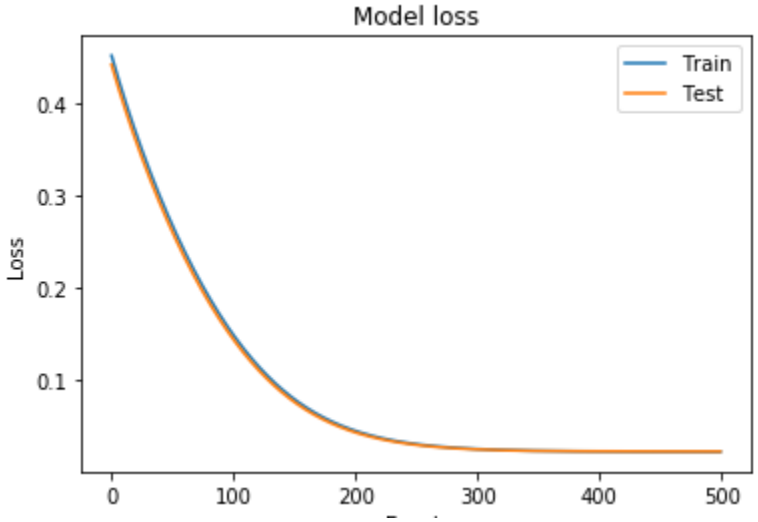
Αριθμός νευρώνων στο κρυφό επίπεδο	RMSE	MAE
H=10	**0.15**	**0.14**
H=20	**0.15**	**0.13**
H=30	**0.15**	**0.13**

Παρακάτω παρουσιάζονται οι γραφικές παραστάσεις για τη σύγκλιση του δικτύου:

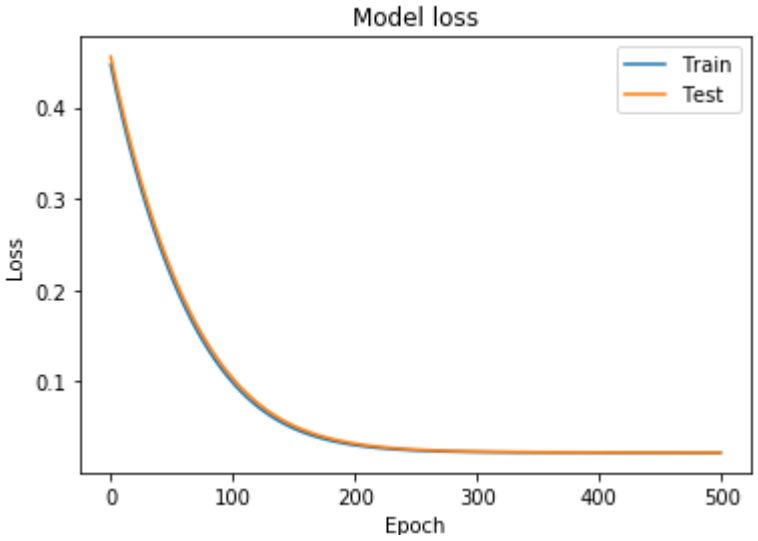
1) H=10 νευρώνες στο κρυφό επίπεδο



2) H=20 νευρώνες στο κρυφό επίπεδο

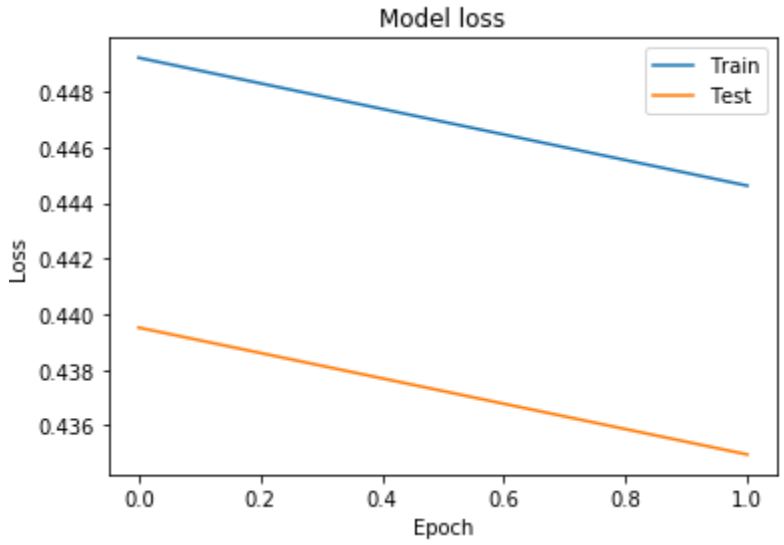


3) H=30 νευρώνες στο κρυφό επίπεδο



Συμπέρασμα: Παρατηρούμε πως πιο σύντομα και ομαλά συγκλίνει το δίκτυο όταν έχει **20 κόμβους στο κρυφό επίπεδο**. Για αυτό το λόγο επιλέγουμε αυτό τον αριθμό για τη συνέχεια των πειραμάτων μας. Επίσης, παρατηρούμε πως τα σφάλματα είναι σχεδόν ίδια σε κάθε περίπτωση, που ενδεχομένως να φταίει το dataset και η έλλειψη του σε τιμές, κάτι που επηρεάζει την εκπαίδευση του δικτύου.

ζ). Η τεχνική του πρόωρου σταματήματος (Early Stopping) εφαρμόζεται όταν υπάρχει κάποιο σημείο στο οποίο η γραφική της εκπαίδευσης πηγαίνει πάνω από της επικύρωσης. Για να απαντηθεί το ερώτημα αυτό κρίθηκε σκόπιμο να εξεταστεί η γραφική παράσταση μεταξύ train και test loss.



Παρατηρούμε πως αυτό συμβαίνει στην εποχή 2, όμως είναι πάρα πολύ νωρίς για να σταματήσουμε την εκπαίδευση του δικτύου.

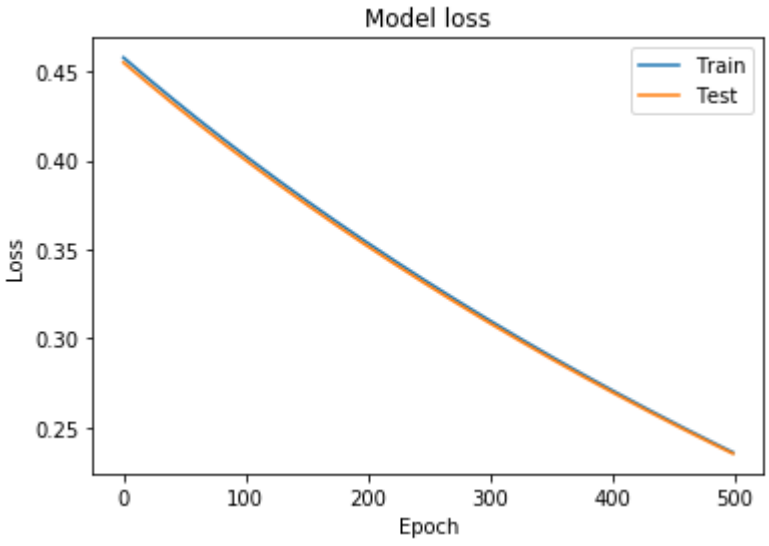
Συμπεραίνουμε, λοιπόν, πως δεν είναι απαραίτητο να χρησιμοποιηθεί η τεχνική πρόωρου σταματήματος σαν κριτήριο τερματισμού καθώς δεν απαιτείται με βάση τον αριθμό των **εποχών** και των batches που επιλέχθηκαν.

Α3. Μεταβολές στο ρυθμό εκπαίδευσης και σταθεράς ορμής

η	m	RMSE	MAE
0.001	**0.2**	**0.48**	**0.45**
0.001	**0.6**	**0.32**	**0.30**
0.05	**0.6**	**0.15**	**0.13**
0.1	**0.6**	**0.15**	**0.13**

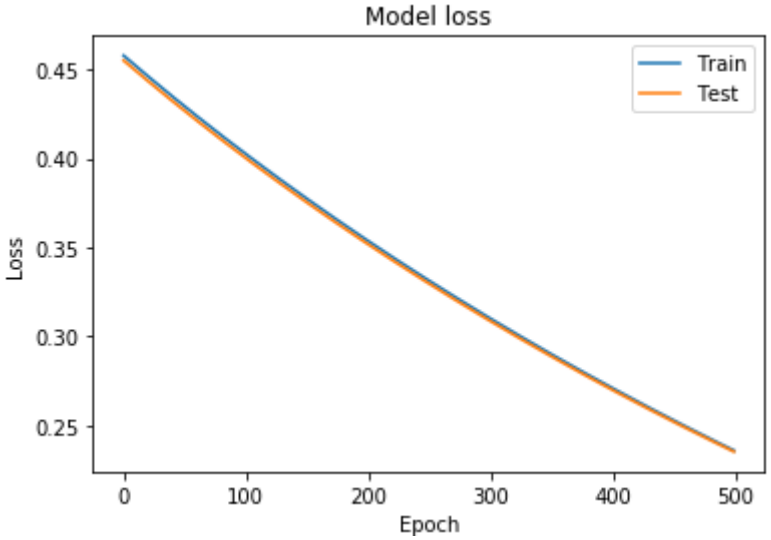
1) **η=0.001 m=0.2 .**

Με αυτές τις τιμές στις υπερπαραμέτρους ρυθμού εκπαίδευσης παρατηρούμε πως το μοντέλο δεν έχει καλή απόδοση. Το σφάλμα αυξάνεται πάρα πολύ, περίπου κατά 300%. Ο ρυθμός μάθησης εδώ είναι πολύ μικρός και έτσι το δίκτυο δυσκολεύεται να συγκλίνει.



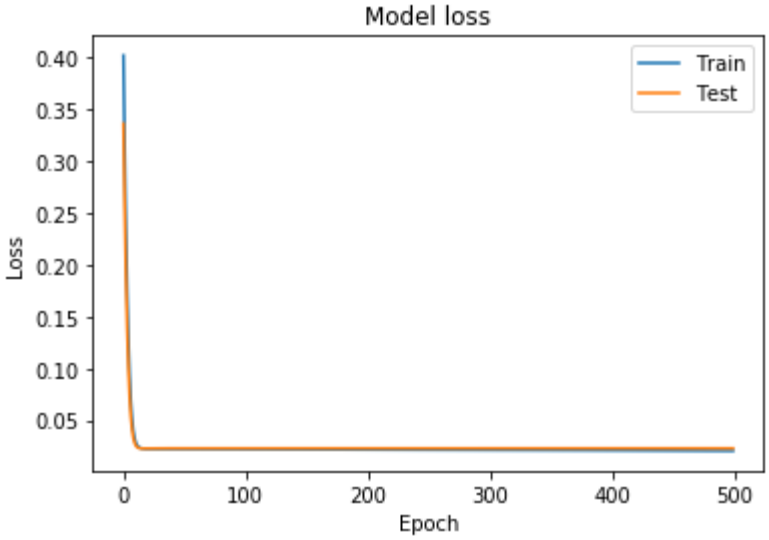
2) η=0.001 m=0.6 .

Στη περίπτωση αυτή «βοηθάμε» το δίκτυο αυξάνοντας την υπερπαράμετρο σταθεράς ορμής m (momentum). Παρατηρούμε πως το σφάλμα μειώνεται αισθητά αλλά όχι αρκετά ώστε να κρατήσουμε αυτές τις υπερπαραμέτρους για το δίκτυό μας.

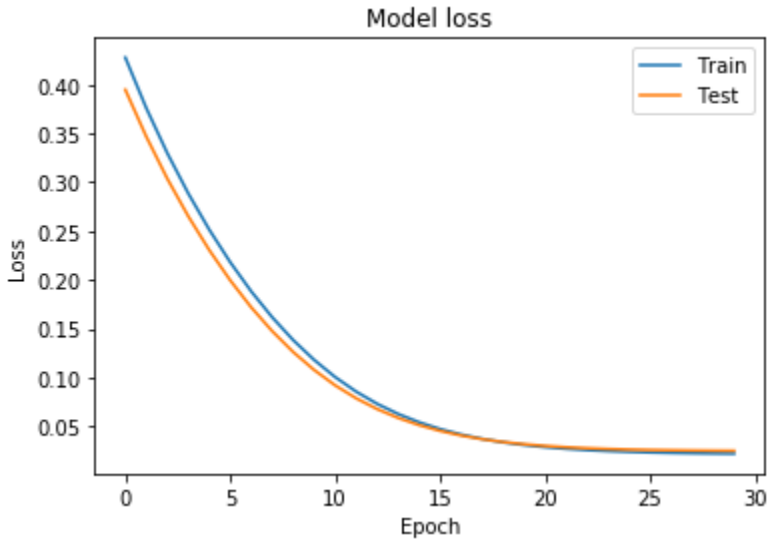


3) η=0.05 m=0.6 .

Με αυτές τις τιμές στις υπερπαραμέτρους ρυθμού εκπαίδευσης παρατηρούμε πως το δίκτυο έχει αρκετά χαμηλό σφάλμα. Ο ρυθμός εκπαίδευσης αυξήθηκε οπότε η σύγκλιση γίνεται πολύ πιο γρήγορα σε αυτή τη περίπτωση.

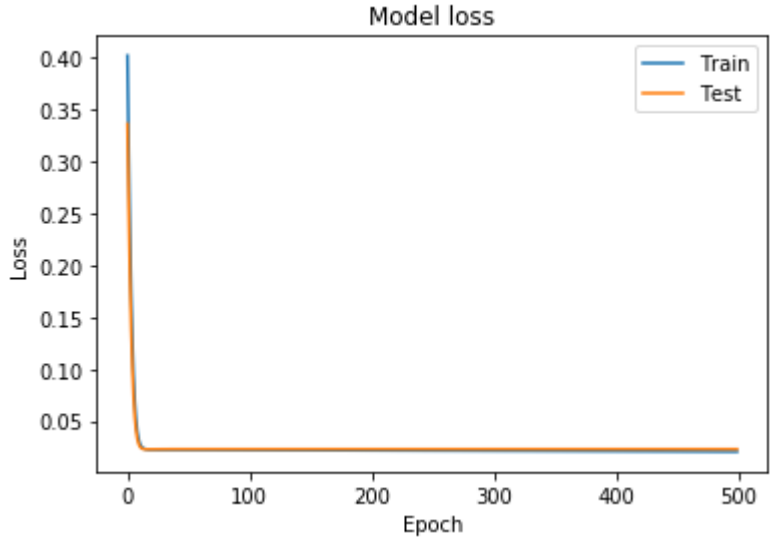


Μπορούμε να ελαττώσουμε τις εποχές από 500 σε 30 όπως φαίνεται παρακάτω και επομένως το δίκτυο να είναι πολύ ταχύτερο. Η σύγκλιση γίνεται στις 15 εποχές.

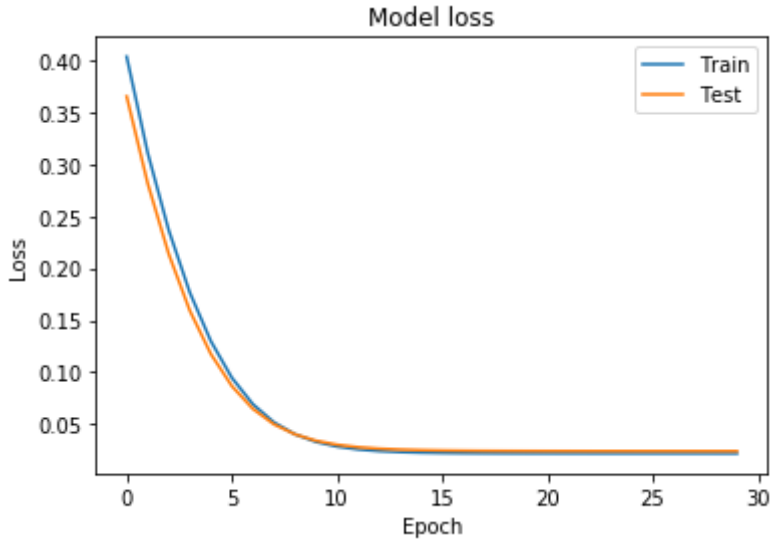


4) η=0.1 m=0.6 .

Με αυτές τις τιμές στις υπερπαραμέτρους ρυθμού εκπαίδευσης παρατηρούμε πως το σφάλμα πάλι είναι χαμηλό. Ο ρυθμός μάθησης είναι ακόμα μεγαλύτερος και έτσι η σύγκλιση γίνεται πιο γρήγορα.



Μπορούμε να ελαττώσουμε τις εποχές από 500 σε 30 όπως φαίνεται παρακάτω και επομένως το δίκτυο να είναι πολύ ταχύτερο. Η σύγκλιση γίνεται στις 7 εποχές πού είναι συγκριτικά ο πιο αποδοτικός συνδυασμός.



Σταθερά ορμής (momentum): Η σταθερά ορμής βοηθά στην επιτάχυνση των διανυσμάτων κλίσης προς τις σωστές κατευθύνσεις, οδηγώντας έτσι σε ταχύτερη σύγκλιση. Η σύγκλιση για να εξασφαλιστεί για τη σταθερά ορμής πρέπει να ισχύει: $0<m<1$. Ώστε οι παλαιότερες μεταβολές των βαρών να βαρύνουν λιγότερο την ανανέωση των βαρών. Μέσω της σταθεράς ορμής επιτυγχάνεται η αποφυγή εγκλωβισμού σε τοπικά ελάχιστα.

Α4. Ομαλοποίηση

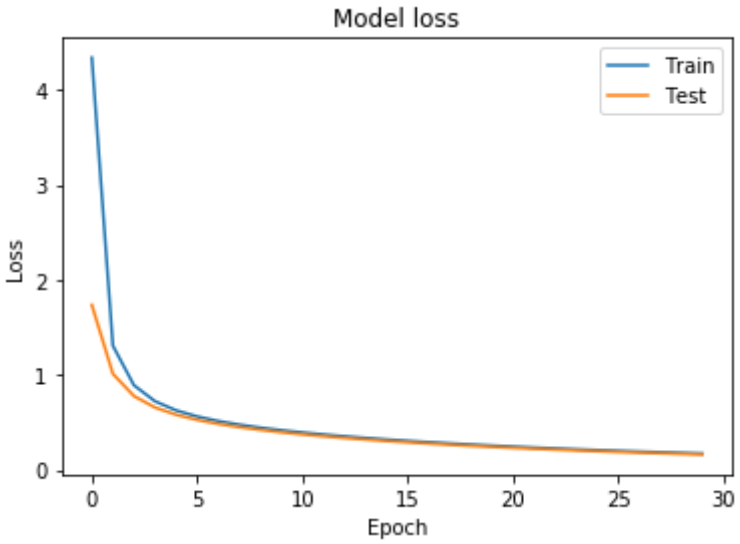
Συντελεστής Φθοράς	RMSE	MAE
0.1	**0.37**	**0.34**
0.5	**0.36**	**0.33**
0.9	**0.41**	**0.39**

Προσπαθούμε να αποφύγουμε την υπερ-εκπαίδευση του μοντέλου πράγμα που σημαίνει πως το σφάλμα αυξάνεται και αυτό είναι λογικό και θεμιτό. Αφού γενικεύουμε, αντί ο ταξινομητής μας να μαθαίνει πολύ καλά τα συγκεκριμένα δεδομένα και άρα να υπερ-προσαρμοστεί σε αυτά με μικρό σφάλμα, πλέον είναι έτοιμος να δεχτεί άλλα δεδομένα και να μπορέσει πάλι να λειτουργήσει. Το σφάλμα αυξάνεται καθώς δεν έχουμε πολύ ακρίβεια στην εύρεση του στόχου.

Παρακάτω φαίνονται οι γραφικές παραστάσεις για τις διάφορες τιμές:

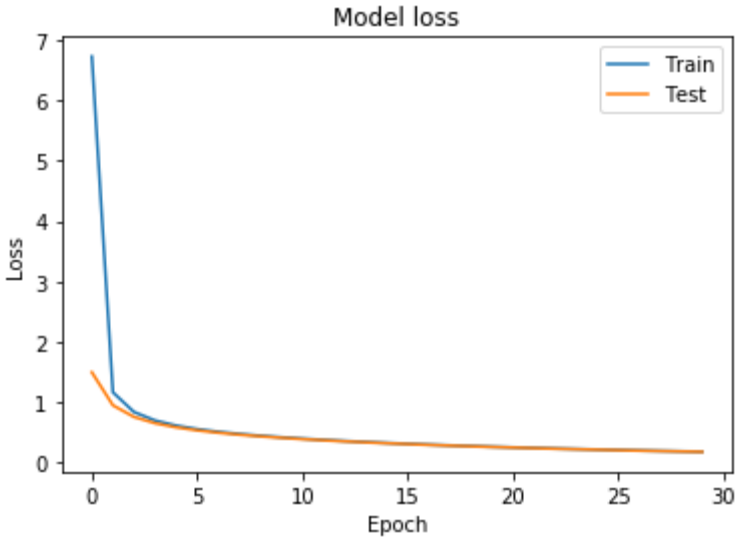
1) **r=0.1**

Ο συντελεστής εδώ είναι σχετικά μικρός και έτσι η σύγκλιση είναι σχετικά αργή.



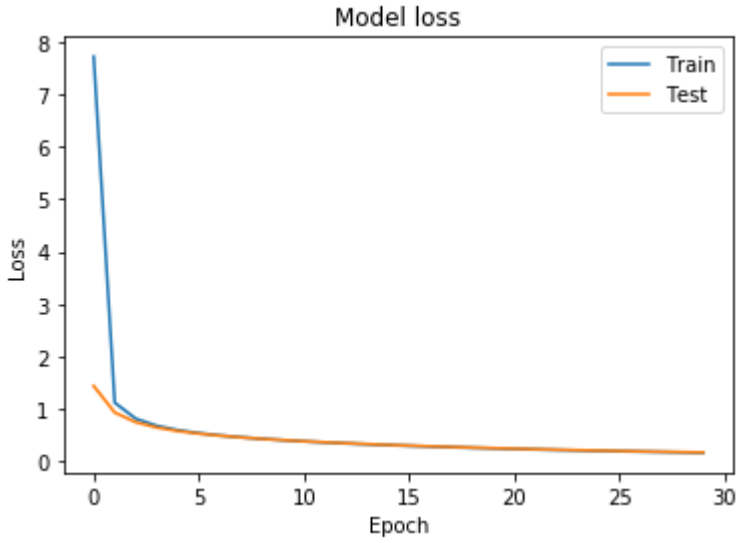
2) **r=0.5**

Ο συντελεστής εδώ είναι αρκετά καλός. Ούτε πολύ μικρός ούτε πολύ μεγάλος και έτσι αποφεύγεται η υπερ-εκπαίδευσης καθώς και η υπο-εκπαίδευση.



3) **r=0.9**

Ο συντελεστής εδώ είναι σχετικά μεγάλος και έτσι το δίκτυο κινδυνεύει από υπο-εκπαίδευση.



<u>L1 ομαλοποίηση</u>: Η L1 ομαλοποίηση ή **Lasso** προσθέτει την απόλυτη τιμή μεγέθους του συντελεστή β ως ποινή στη συνάρτηση κόστους. Ουσιαστικά, συρρικνώνει τον συντελεστή του λιγότερο σημαντικού χαρακτηριστικού στο μηδέν, αφαιρώντας εντελώς κάποια χαρακτηριστικά. Έτσι, λειτουργεί καλά για την επιλογή χαρακτηριστικών σε περίπτωση που έχουμε τεράστιο αριθμό διανυσμάτων. Απομακρυνόμαστε από τα βάρη που θα μας προκαλούσαν προβλήματα, όπως υπερ-εκπαίδευση. Η L1 «σπρώχνει» τα χαρακτηριστικά προς το 0 αναλόγως με το αν είναι εξ' αρχής αρνητικά ή θετικά.

Παρακάτω φαίνεται ο τύπος υπολογισμού με ομαλοποίηση L1:

$$Loss = Error(y, \hat{y}) + \lambda \sum_{i=1}^{N} |w_i|$$

Παρακάτω φαίνεται ο τύπος υπολογισμού με ομαλοποίηση L2 (Ridge Regression):

$$Loss = Error(y, \hat{y}) + \lambda \sum_{i=1}^{N} w_i^2$$

Οι πολύ μεγάλες τιμές στο συντελεστή λ της εξίσωσης οδηγεί το δίκτυο σε υπο-εκπαίδευση.

Η ουσιαστική διαφορά μεταξύ L1 και L2 ομαλοποιήσεων είναι η παρακάτω. Η **L1** αφαιρεί (ποσοτικά) τα χαρακτηριστικά εκείνα που δεν συνεισφέρουν πολύ στην εκπαίδευση του δικτύου και έτσι το δίκτυο θα υπερ-εκπαιδευόταν λόγω αυτών. Η εκτίμηση, έτσι, του στόχου γίνεται από κάποια και όχι όλα τα χαρακτηριστικά. Η **L2**, αντιθέτως, αφαιρεί (ποιοτικά) τα χαρακτηριστικά αυτά και ο υπολογισμός του στόχου γίνεται με όλα τα χαρακτηριστικά μαζί.

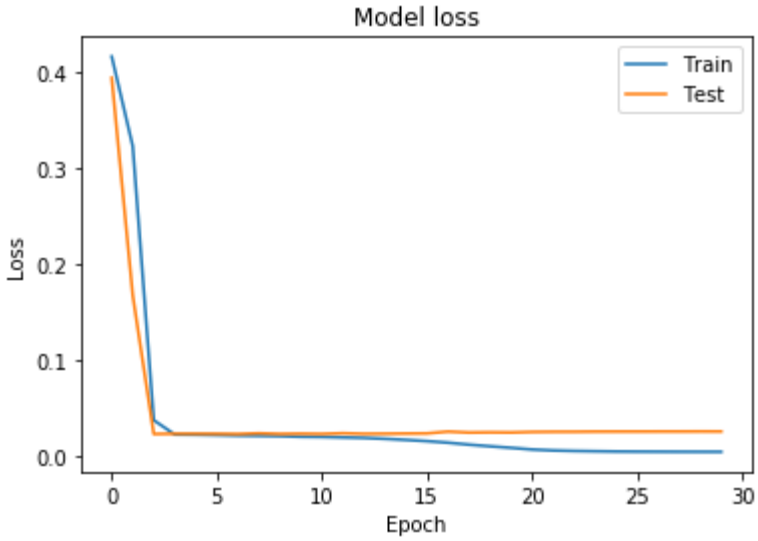
Ο λόγος που στο συγκεκριμένο πρόβλημα επιλέγεται η L1 είναι πως η έλλειψη πλήρους συνόλου δεδομένων μας ανάγκασε να τροφοδοτήσουμε το δίκτυο με πολλές ίδιες τιμές που, εφόσον έχουν την ίδια συνεισφορά, μπορούν να υπερ-εκπαιδεύσουν το δίκτυο.

Α5. Βαθύ Νευρωνικό Δίκτυο

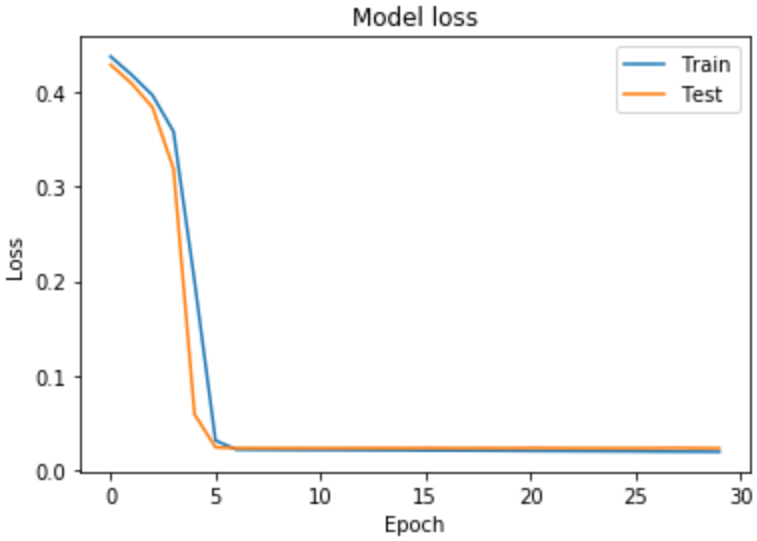
Η επιλογή των νευρώνων και των επιπέδων για το βαθύ νευρωνικό δίκτυο είναι μια εμπειρική διαδικασία, επομένως οποιοσδήποτε άλλος υλοποιούσε το ίδιο δίκτυο θα μπορούσε να στοιχίσει διαφορετικά τους νευρώνες των κρυφών επιπέδων καθώς και να επιλέξει διαφορετικούς αριθμούς κόμβων σε αυτά. Εδώ, μετά από πειραματισμό χρησιμοποιήθηκαν 3 κρυφά επίπεδα για το δίκτυο.

Οι συναρτήσεις ενεργοποίησης μετά από εκτενή πειραματισμό κρίθηκε ορθό να αλλάξουν στο κρυφό επίπεδο σε συνάρτηση **relu**, ενώ για το επίπεδο εξόδου διατηρήθηκε η **γραμμική (linear).**

Γενικώς, **ο αριθμός των νευρώνων** καθώς πηγαίνουμε πιο «βαθιά» στο νευρωνικό δίκτυο συνήθως **αυξάνεται**. Πειραματικά, όταν οι νευρώνες μειώνονται σε κάθε επίπεδο το δίκτυο λειτουργεί πολύ χειρότερα όπως φαίνεται παρακάτω. Φαίνεται να γίνεται σύγκλιση όμως μετά τις 15 εποχές η γραφική του σφάλματος του test ανεβαίνει πάνω από αυτή του train κάτι που δεν είναι θεμιτό.



Αντιθέτως στη περίπτωση που ο αριθμός των νευρώνων ήταν αυξανόμενος το δίκτυο σύγκλινε καλύτερα και με μικρότερο (κατά λίγο) σφάλμα. Η γραφική παράσταση φαίνεται παρακάτω.



Άλλος ένας λόγος για την αύξηση του αριθμού των νευρώνων σταδιακά στα κρυφά επίπεδα είναι πως έτσι μειώνεται η διαστατικότητα.

Καταλήγουμε λοιπόν στον εξής πίνακα τιμών σφαλμάτων **RMSE** και **MAE** για τους διάφορους αριθμούς νευρώνων στα κρυφά επίπεδα.

Αριθμός νευρώνων στο κρυφό επίπεδο	**RMSE**	**MAE**
H_1=5, H_2=10, H_3=20	**0.15**	**0.13**
H_1=10, H_2=20, H_3=30	**0.15**	**0.13**
H_1=30, H_2=20, H_3=10	**0.16**	**0.13**

Η τελική επιλογή των νευρώνων είναι **5,10 και 20** αντίστοιχα στα 3 κρυφά επίπεδα του δικτύου.

Ο κώδικας για το ερώτημα αυτό παρατίθεται στο αρχείο **DNN.py**