Εργασία Υπολογιστική Νοημοσύνη: Μέρος Α'

Ιωάννα Γώγου – ΑΜ1054388 – gogou@ceid.upatras.gr

Για την υλοποίηση της εργασίας χρησιμοποιήθηκαν οι εξής βιβλιοθήκες της Python: keras, numpy, sklearn και matplotlib. Οι κώδικες της άσκησης βρίσκονται στο παρακάτω link:
https://drive.google.com/open?id=1164N1aM7srv6gF2U4IIgnZfRg72XhEpm

Α1. Προεπεξεργασία και Προετοιμασία δεδομένων

Η δημιουργία και προεπεξεργασία του dataset υλοποιείται στο αρχείο κώδικα "create_dataset.py".

Αρχικά δημιουργούμε dataset με την δομή που χρειαζόμαστε. Κάθε γραμμή του dataset αντιστοιχεί σε έναν χρήστη (N γραμμές) και θα αποτελείται από N εισόδους και M εξόδους (M+N στήλες). Κάθε μία από τις N εισόδους αντιστοιχεί σε έναν χρήστη, καθώς εφαρμόζουμε one hot encoding για την αναπαράστασή τους. Κάθε μία από τις M εξόδους αντιστοιχεί σε βαθμολογία κάποιας ταινίας.

α) Κεντράρισμα:

Η μέθοδος του κεντραρίσματος είναι χρήσιμη στο πρόβλημά μας. Ας θεωρήσουμε τις βαθμολογίες στο διάστημα [1-5] και ότι το 3 αποτελεί την «βάση» των βαθμολογιών (μέση τιμή). Έστω ότι ένας χρήστης βαθμολογεί συχνά με 4, με εξαίρεση κάποιες ταινίες στις οποίες έχει βάλει 3 και 5. Για το νευρωνικό δίκτυο, οι ταινίες στις οποίες βάζει συχνά 4 δεν θα πρέπει να είναι τόσο σημαντικές για την διαδικασία μάθησης, διότι αποτελούν την προσωπική «βάση» αυτού του επιεικούς χρήστη. Επομένως ο κάθε χρήστης βαθμολογεί με την δική του βάση οι οποία μπορεί να είναι είτε χαμηλή είτε υψηλή. Το σημαντικό όμως σε αυτό, για το νευρωνικό, είναι οι διακυμάνσεις γύρω από αυτήν και όχι τόσο η ίδια η βάση. Για παράδειγμα ξέρουμε ότι ένας χρήστης βαθμολόγησε +2 μονάδες πάνω από την βάση του μια ταινία οπότε πρέπει να έχει ιδιαίτερη προτίμηση γι' αυτήν ή αντίθετα -1 οπότε δεν του άρεσε τόσο. Επομένως πρέπει να μετασχηματίσουμε όλες τις βαθμολογίες γύρω από μια κοινή βάση, το 0. Τον μετασχηματισμό τον υλοποιούμε αφαιρώντας από τις βαθμολογίες ενός χρήστη την μέση τιμή των βαθμολογιών αυτών. Οπότε, στο dataset το κεντράρισμα θα γίνει ανά γραμμή (χρήστη).Επομένως το νέο διάστημα στο οποίο κυμαίνονται οι τιμές δεν μπορούμε να το ορίσουμε ακριβώς διότι μεταβάλλεται ανά χρήστη (ανάλογα με την μέση βαθμολογία του). Στην πορεία, μόλις γίνει κανονικοποίηση, θα μας είναι ξεκάθαρο διότι είναι κοινό για όλους.

β) Ελλιπείς τιμές:

Αρχικά, θα θεωρήσουμε ότι συμπληρώνουμε τις ελλιπείς τιμές πριν κάνουμε το κεντράρισμα των τιμών. Θα εξετάσουμε όλες τις επιλογές μια μια. Πρώτον, εάν επιλέξουμε να γεμίσουμε τις ελλιπείς τιμές με μηδέν αυτό θα έχει πολύ μεγάλο αρνητικό αντίκτυπο στην ταινία αυτή, το οποίο θα λάβει υπόψιν του το νευρωνικό στην πρόβλεψή του. Όμως η μηδενική βαθμολογία μπορεί να απέχει πολύ από την πιθανή πραγματική τιμή που θα έβαζε ο χρήστης και το κόστος που θα έχει στην απόφαση του αλγορίθμου είναι πολύ μεγάλο για να ρισκάρουμε μια τέτοια επιλογή. Βέβαια αν οι μηδενικές τιμές συμπληρωθούν μετά το κεντράρισμα, η επιλογή ii ισοδυναμεί με την επιλογή iii. Δεύτερον, εάν επιλέξουμε μια τυχαία τιμή στο κατάλληλο εύρος, υπάρχει πιθανότητα να είναι πολύ κοντά στην πραγματική, αλλά η πιθανότητα αυτή είναι πολύ μικρή. Επίσης βάζοντας τυχαίες μεταβλητές καταστρέφουμε κάποιο πιθανώς κοινό πρότυπο στην βαθμολόγηση των χρηστών. Όμως η τρίτη επιλογή είναι πιο επιθυμητή καθώς επιλέγοντας τον μέσο όρο μπορεί μεν να απέχουμε μεγάλη απόσταση από την πραγματική τιμή αλλά η απόσταση

αυτή θα είναι πάντα μικρότερη από την χειρότερη περίπτωση της επιλογής ii, δηλαδή την περίπτωση να είναι η πρόβλεψη στο ένα άκρο του διαστήματος και η πραγματική τιμή στο άλλο άκρο.

γ) Κανονικοποίηση:

Αφού κάνουμε κεντράρισμα στο μηδέν, επιλέξαμε να κάνουμε κανονικοποίηση στο διάστημα [-1,1] διαιρώντας με την τυπική απόκλιση των βαθμολογιών κάθε χρήστη.

Το κεντράρισμα και η κανονικοποίηση υλοποιούνται στον κώδικα με τον StandardScaler του sklearn.

δ) Διασταυρούμενη Επικύρωση:

Η διασταυρούμενη επικύρωση υλοποιήθηκε χρησιμοποιώντας την συνάρτηση kfold.split() της βιβλιοθήκης sklearn η οποία επιστρέφει δύο πίνακες, έναν για τα indices των δεδομένων που θα χρησιμοποιηθούν στην εκπαίδευση, και έναν για τα indices στον έλεγχο. Κάθε γραμμή των πινάκων αντιστοιχεί σε κάθε επανάληψη της διασταυρούμενης επικύρωσης. Ο βρόχος της διασταυρούμενης επικύρωσης περιέχει όλη την διαδικασία εκπαίδευσης και ελέγχου του νευρωνικού δικτύου, και φαίνεται στον κώδικα του αρχείου “neural_network.py”.

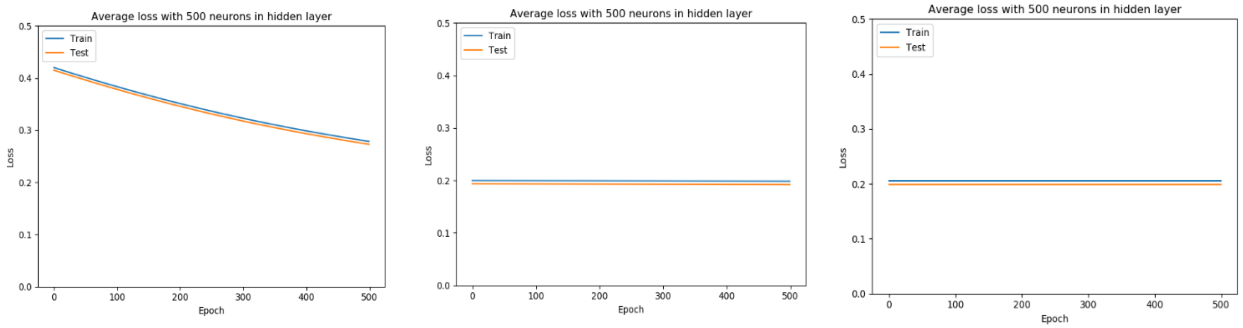
Α2. Επιλογή αρχιτεκτονικής

α) Οι δύο μετρικές αυτές υπολογίζουν το μέσο σφάλμα (τετραγωνικό και απόλυτο αντίστοιχα) που προκύπτει συγκρίνοντας όλα τα διανύσματα πραγματικών βαθμολογιών με αυτά των προβλεπόμενων για κάθε χρήστη, στο τέλος της τελευταίας εποχής εκπαίδευσης. Χρησιμοποιούμε αυτές τις μετρικές καθώς το πρόβλημά μας είναι τύπου γραμμικής παλινδρόμησης οπότε προσπαθούμε να προσεγγίσουμε την τιμή της βαθμολογίας κι ας μην είναι η ίδια. Επομένως κάποια μετρική που υπολογίζει τις λάθος ταξινομήσεις σε κατηγορίες, όπως η ακρίβεια, δεν θα λειτουργούσε στην περίπτωσή μας. Μάλιστα η ακρίβεια θα μπορούσε να παίρνει συνεχώς τιμή μηδέν, εξαιτίας του ότι στην έξοδο δεν έχουμε τις ακριβείς τιμές αλλά προσέγγιση τους. Στην μετρική RMSE οι διαφορές των τιμών υψώνονται στο τετράγωνο. Αυτό έχει ως αποτέλεσμα σε μεγάλες διαφορές το RMSE να μεγεθύνεται ακόμη πιο γρήγορα σε σχέση με το MAE, και επομένως τα μεγάλα σφάλματα έχουν μεγαλύτερο βάρος στην μετρική αυτή. Τέτοια μεγάλα σφάλματα μπορούν να προκύψουν και σε περίπτωση που το dataset μας έχει outliers, δηλαδή τιμές που δεν συμβαδίζουν καθόλου με το πρότυπο. Για παράδειγμα εάν κάποιος βαθμολογήσει χαμηλά μια ταινία που τα άτομα με κοινά ενδιαφέροντα με αυτών βαθμολόγησαν πολύ ψηλά θεωρείται outlier. Επομένως περιμένουμε το RMSE να είναι μεγαλύτερο από το MAE.

β) Το νευρωνικό δίκτυο θα έχει πλήθος εισόδων ίσο με το πλήθος των χρηστών (N). Αυτό συμβαίνει καθώς ένας χρήστης αναπαρίσταται με N bit μέσω one hot encoding. Είναι προτιμότερη μέθοδος από την απλή δυαδική κωδικοποίηση γιατί το διάνυσμα αναπαράστασης κάθε χρήστη ισαπέχει από οποιοδήποτε άλλο διάνυσμα αναπαράστασης χρήστη. Αυτό είναι απαραίτητο ώστε οι προβλέψεις του νευρωνικού δικτύου για κάποιον χρήστη να μην επηρεάζονται από την απόσταση που έχει η αναπαράσταση του με την αναπαράσταση άλλων χρηστών, διότι δεν είναι επιθυμητό κριτήριο ομοιότητας μεταξύ χρηστών.

γ) Το νευρωνικό δίκτυο θα έχει πλήθος εξόδων ίσο με το πλήθος των ταινιών (M). Αυτό συμβαίνει διότι το νευρωνικό δίκτυο πρέπει στην έξοδο του να βγάζει τις βαθμολογίες του χρήστη για κάθε ταινία.

δ) Δοκιμάζοντας τρείς διαφορετικές συναρτήσεις ενεργοποίησης (sigmoid,hyperbolic tangent,rectified linear unit) παρατηρήσαμε το εξής. Οι συναρτήσεις ενεργοποίησης οι οποίες έπαιρναν τιμή μηδέν στο μηδέν (tanh,relu) είχαν ως αποτέλεσμα το σφάλμα να μην μειώνεται και να παραμένει σχεδόν σταθερό (βλέπε παρακάτω διάγραμμα), δηλαδή το νευρωνικό δίκτυο δεν εκπαιδευόταν**. Επομένως επιλέξαμε την συνάρτηση sigmoid.**



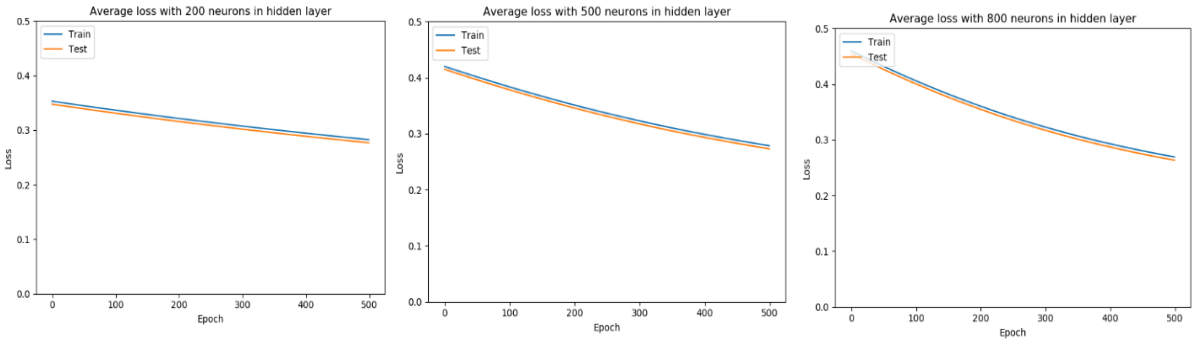
Εικόνα 1: Τα διαγράμματα του σφάλματος ανά εποχή με sigmoid, relu, tanh αντίστοιχα και 500 νευρώνες στο κρυφό επίπεδο.

ε) Αρχικά, επειδή για το πρόβλημα που προσπαθούμε να λύσουμε θέλουμε να χρησιμοποιήσουμε back propagation, δεν μπορούμε να χρησιμοποιήσουμε γραμμική συνάρτηση ενεργοποίησης καθώς η παράγωγός της είναι σταθερή με αποτέλεσμα ο αλγόριθμος gradient descent να μην λειτουργεί. Επίσης, τα δεδομένα μας κανονικοποιήθηκαν στο διάστημα [-1,1] οπότε πρέπει να επιλέξουμε συνάρτηση ενεργοποίησης που έχει στην έξοδο τιμές στο διάστημα αυτό, αλλά η σιγμοειδής έχει τιμές στο [0,1]. **Επομένως χρησιμοποιούμε την hyperbolic tangent συνάρτηση ενεργοποίησης που αντιστοιχεί στο διάστημά μας.**

στ) Δοκιμάζουμε τους παρακάτω αριθμούς νευρώνων με 500 εποχές εκπαίδευσης. Ο αριθμός αυτός επιλέξαμε να είναι K<M,N ώστε να προσομοιωθεί η μείωση της διαστατικότητας που συμβαίνει στην τεχνική collaborative filtering με embeddings. Τα βάρη στους K αυτούς νευρώνες θα είναι ο τιμές των embeddings. Η βελτίωση των βαρών στην εκπαίδευση του νευρωνικού γίνεται χρησιμοποιώντας το MAE (loss). Παράλληλα στο τέλος κάθε εποχής υπολογίζονται ως μετρικές (δηλαδή δεν επηρεάζουν την εκπαίδευση) τα RMSE και MAE (metrics). Τα RMSE και MAE υπολογίζονται από των μέσο όρο όλων των επαναλήψεων του cross validation.

Αριθμός νευρώνων στο κρυφό επίπεδο	RMSE	MAE
200	**1.012**	**0.282**
500	1.014	0.279
800	1.015	0.267

Παρατηρούμε μια ελάχιστη μείωση του σφάλματος MAE με την αύξηση των νευρώνων και ελάχιστη αύξηση του RMSE. Παρακάτω βλέπουμε τα διαγράμματα σύγκλισης του σφάλματος σε κάθε επανάληψη του cross validation.



Εικόνα 2: Σύγκλιση σφάλματος ανά περίπτωση πλήθους νευρώνων.

Από τα διαγράμματα σύγκλισης συμπεραίνουμε ότι όσο αυξάνουμε τους νευρώνες το διάγραμμα σύγκλισης μετατρέπεται από γραμμικό σε καμπύλη. Αυξάνοντας τους νευρώνες μειώθηκε ελάχιστα το σφάλμα στην τελική εποχή. Όμως με λιγότερους νευρώνες, στις αρχικές εποχές έχουμε σημαντικά χαμηλότερο σφάλμα. Κάνοντας περαιτέρω πειράματα με ακόμα λιγότερους (50) και περισσότερους νευρώνες (1300) παρατηρήσαμε ότι ισχύει το παρακάτω. Μειώνοντας τους νευρώνες το διάγραμμα τείνει σε σταθερότητα διότι το δίκτυο αργεί να μάθει με τόσο λίγους νευρώνες, όμως ξεκινάει με χαμηλό σφάλμα. Αυξάνοντας τους νευρώνες το νευρωνικό μαθαίνει πιο γρήγορα όμως ξεκινάει με υψηλό σφάλμα. **Επομένως επειδή με λιγότερους (200) νευρώνες παρατηρούμε χαμηλότερο σφάλμα και μειώνεται σημαντικά το υπολογιστικό κόστος, επιλέγουμε τους λιγότερους με την ελπίδα ότι αυξάνοντας τον ρυθμό εκπαίδευσης το δίκτυο θα μάθει γρηγορότερα.**

ζ) Μπορεί να εφαρμοστεί η τεχνική του πρόωρου σταματήματος καθώς αν αφήσουμε το νευρωνικό να εκπαιδευθεί για υπερβολικά πολλές εποχές μπορεί να καταλήξουμε σε υπερεκπαίδευση του στα δείγματα που του δόθηκαν, με αποτέλεσμα να έχει κακή απόδοση σε νέες εισόδους (δεν γενικεύει). Χρησιμοποιούμε το μοντέλο early stopping που προσφέρει το keras (EarlyStopping()). Ως κριτήρια τερματισμού χρησιμοποιούμε τα παρακάτω. Αρχικά, η εκπαίδευση τερματίζει αν η μείωση του σφάλματος (στην περίπτωσή μας χρησιμοποιούμε το mean absolute error) μεταξύ των εποχών είναι κάτω από κάποια σταθερά (min_delta=0.001). Επιλέξαμε αυτό διότι αν δεν παρατηρούμε σημαντική βελτίωση σημαίνει ότι το νευρωνικό δεν μπορεί να εκπαιδευτεί άλλο και επομένως πρέπει να σταματήσουμε. Δεύτερο κριτήριο τερματισμού είναι μόλις το σφάλμα αυξηθεί σε σχέση με την προηγούμενη εποχή (mode='min'). Αν συμβεί αυτό, τότε κρατιούνται τα βάρη της καλύτερης εποχής (restore_best_weights=True). Αυτή η μέθοδος καταλήγει μεν σε τοπικά ελάχιστα αλλά μας δίνει την δυνατότητα να προλάβουμε την περίπτωση που το σφάλμα αυξάνεται επ' άπειρον. Οπότε, για να λύσουμε το πρόβλημα των τοπικών ελαχίστων δίνουμε ένα περιθώριο 5 εποχών στον αλγόριθμο (patience=5) μόλις εντοπίσει αύξηση στο σφάλματος, σε περίπτωση που μέσα σε αυτές τις 5 εποχές καταφέρει να ξαναμειώσει το σφάλμα.

Α3. Μεταβολές στο ρυθμό εκπαίδευσης και σταθεράς ορμής

Αρχικά, όπως φαίνεται και στον τύπο, ο ρόλος της ορμής στην μεταβολή των βαρών αφορά το περιθώριο που δίνεται στα βάρη να κρατήσουν τον ρυθμό αύξησης που είχαν πριν παρά της μείωση του ρυθμού μεταβολής του σφάλματος,

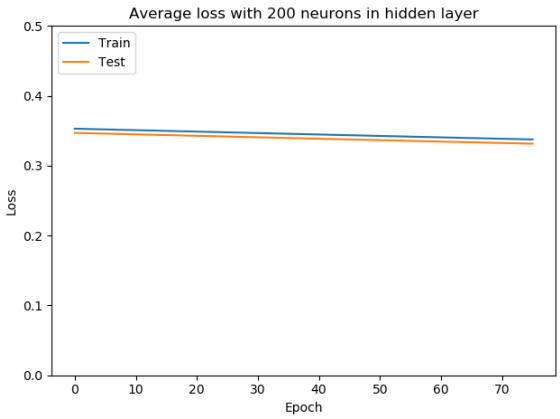
$$\Delta w_{ij} = (\eta * \frac{\partial E}{\partial w_{ij}}) + (\gamma * \Delta w_{ij}^{t-1})$$

momentum factor

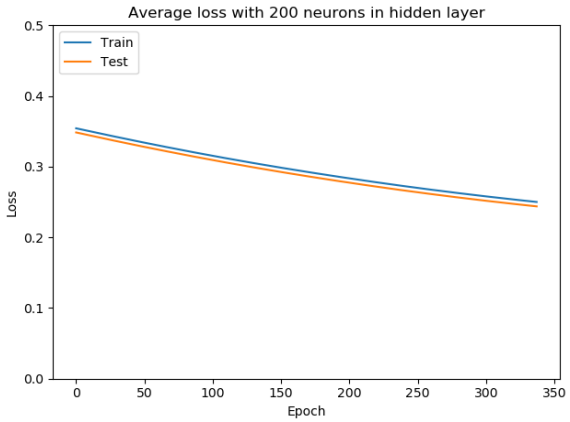
weight increment, previous iteration

δηλαδή επιτρέπει στο νευρωνικό να ξεπερνάει τοπικά ελάχιστα. Επομένως αφού ο στόχος του είναι η συγκράτηση της προηγούμενης μεταβολής των βαρών (ή ενός μέρους της) δεν έχει νόημα να την επαυξάνει και επομένως ο παράγοντας δεν είναι πολλαπλασιαστικός(>1) αλλά μεριστικός(<1).

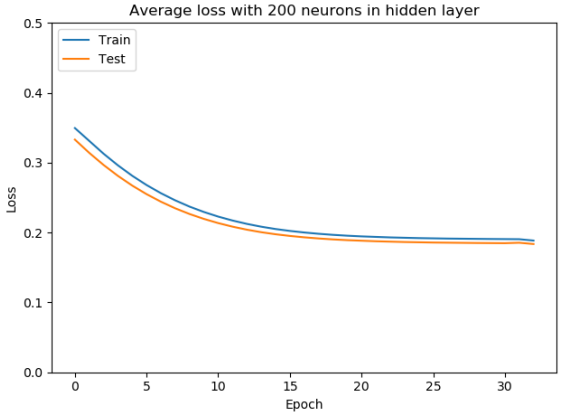
η	m	RMSE	MAE
0.001	0.2	1.021	0.335
0.001	0.6	1.007	0.247
0.05	0.6	0.999	0.189
0.1	**0.6**	**0.999**	**0.189**



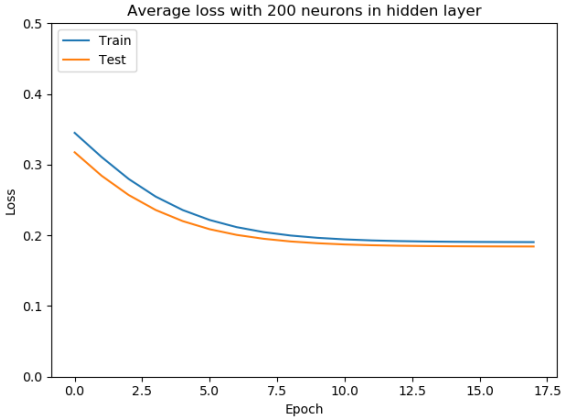
Εικόνα 3: η=0.001 m=0.2



Εικόνα 4: η=0.001 m=0.6



Εικόνα 5: η=0.05 m=0.6



Εικόνα 6: η=0.1 m=0.6

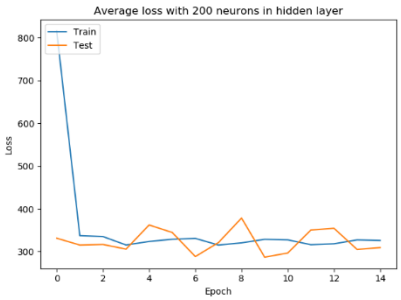
Παρατηρούμε βλέποντας την Εικόνα 3 ότι οι παράμετροι δεν είναι κατάλληλοι και ότι το νευρωνικό μαθαίνει τόσο αργά που αναγκάζεται να σταματήσει λόγω early stopping. Η μεγαλύτερη μείωση στο σφάλμα παρατηρείται αλλάζοντας την σταθερά ορμής από 0.2 σε 0.6, κάτι αναμενόμενο αφού η ορμή επηρεάζει την ταχύτητα σύγκλισης στο σφάλμα. Την μεγαλύτερη αλλαγή στο πλήθος των εποχών την παρατηρούμε στην αλλαγή του ρυθμού εκπαίδευσης από 0.001 σε 0.05 λόγω της μεγάλης αύξησης στον ρυθμό ο οποίος επηρεάζει τον χρόνο εκπαίδευσης (εποχές). Οπότε συνολικά, αυξάνοντας τον ρυθμό εκπαίδευσης και την σταθερά ορμής το νευρωνικό δίκτυο μαθαίνει πιο γρήγορα και το σφάλμα μειώνεται**. Οι παράμετροι η=0.1 m=0.6 είναι η καλύτερη επιλογή διότι έχουμε μειώσει και το σφάλμα αλλά και το πλήθος των εποχών που χρειάζονται για να φτάσουμε εκεί.**

Α4. Ομαλοποίηση

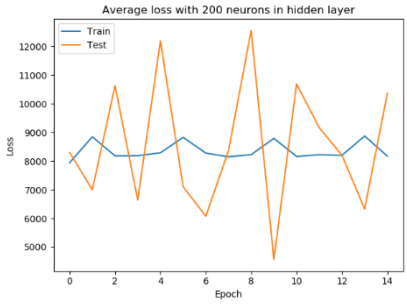
Η L1 ομαλοποίηση έχει το πλεονέκτημα ενάντια της L2 στην τάση της να «σπρώχνει» τα βάρη προς το μηδέν με αποτέλεσμα χαρακτηριστικά που δεν επηρεάζουν την εκπαίδευση να απενεργοποιούνται. Αυτό βοηθάει στην περίπτωσή μας καθώς δεν γνωρίζουμε από πριν ποια χαρακτηριστικά είναι σημαντικά διότι είναι «λανθάνοντα» δηλαδή διαμορφώνονται μέσω τον βαρών στο κρυφό επίπεδο. Επίσης είναι πολλά σε αριθμό και υπάρχει περιθώριο μείωσης τους. Έτσι το μοντέλο απλοποιείται και ελαττώνεται η πιθανότητα υπερεκπαίδευσης.

decay (r)	RMSE	MAE
0.1	1.001	0.218
0.5	1.012	0.315
0.9	1.019	0.332

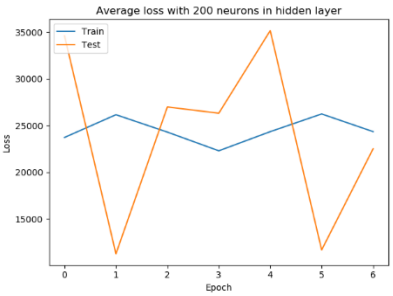
Δυστυχώς βλέπουμε το σφάλμα να έχει αυξηθεί. Παρατηρούμε τα διαγράμματα για να διερευνήσουμε γιατί συμβαίνει αυτό.



Εικόνα 7: r=0.1



Εικόνα 8: r=0.5



Εικόνα 9: r=0.9

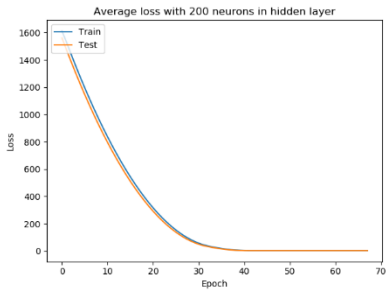
Αρχικά στα διαγράμματα παρατηρούμε αύξηση του σφάλματος σε κάποιες εποχές. Αξίζει να υπενθυμιστεί ότι εφαρμόσαμε μεν early stopping για τις εποχές που παρατηρείται αύξηση του σφάλματος, αλλά ορίσαμε περιθώριο 5 εποχές σε περίπτωση που η εκπαίδευση καταφέρει να ξαναμειώσει το σφάλμα στο διάστημα αυτό, και γι' αυτό η εκπαίδευση δεν σταματάει κατευθείαν. Επίσης το διάγραμμα του σφάλματος, το οποίο ορίσαμε να είναι το mean absolute error, έχει διαφορετικές τιμές σε τάξη από το mean absolute error που υπολογίζεται στο τέλος της εκπαίδευσης (βλέπε πίνακα) διότι στο διάγραμμα παρουσιάζονται οι τιμές του σφάλματος αφού εφαρμοστούν σε αυτές τα πέναλτι της ομαλοποίησης, που στην συνέχεια θα χρησιμοποιηθούν για την βελτιστοποίηση των βαρών. Αυτό φαίνεται πιο ξεκάθαρα από τον παρακάτω τύπο του σφάλματος εκπαίδευσης με L1 ομαλοποίηση.

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

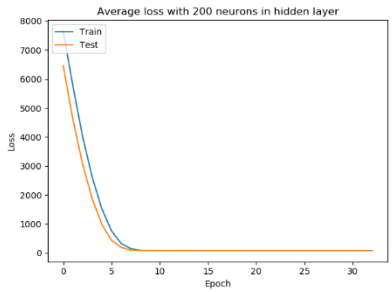
Ας διερευνήσουμε τώρα το πρόβλημα. Κοιτώντας τα παραπάνω διαγράμματα παρατηρούμε μια «περιοδικότητα» στο σφάλμα μετά από κάποια εποχή. Αυτό συμβαίνει διότι δυστυχώς η επιλογή της παραμέτρου ρυθμού εκπαίδευσης του προηγούμενου ερωτήματος αποδείχθηκε πολύ μεγάλη για τις συγκεκριμένες τιμές φθοράς με αποτέλεσμα ο αλγόριθμος gradient descent να κάνει πολύ μεγάλα

βήματα γύρω από το τοπικό ελάχιστο χωρίς να το φτάνει. Επίσης το σφάλμα ελέγχου ξεπερνάει το σφάλμα εκπαίδευσης σε κάποιες εποχές, κάτι που δηλώνει υπερεκπαίδευση. Επομένως μια λύση είναι είτε να μειώσουμε τον ρυθμό εκπαίδευσης είτε να μειώσουμε τον συντελεστή φθοράς καθώς όπως φαίνεται από το διάγραμμα η διακυμάνσεις μειώνονται για μικρότερη τιμή του. **Εμείς θα διορθώσουμε την επιλογή του προηγούμενου ερωτήματος με καινούργιο συντελεστή εκπαίδευσης lr=0.001. Τα αποτελέσματα με την νέα διόρθωση φαίνονται παρακάτω.**

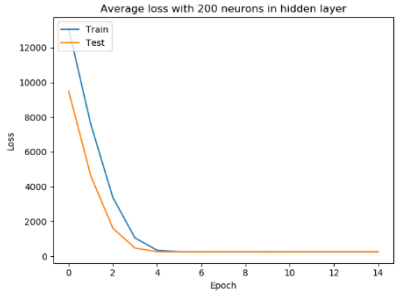
decay (r)	RMSE	MAE
0.1	**0.999**	**0.185**
0.5	0.999	0.186
0.9	1.000	0.188



Εικόνα 10: η=0.001 r=0.1



Εικόνα 11: η=0.001 r=0.5



Εικόνα 12: η=0.001 r=0.9

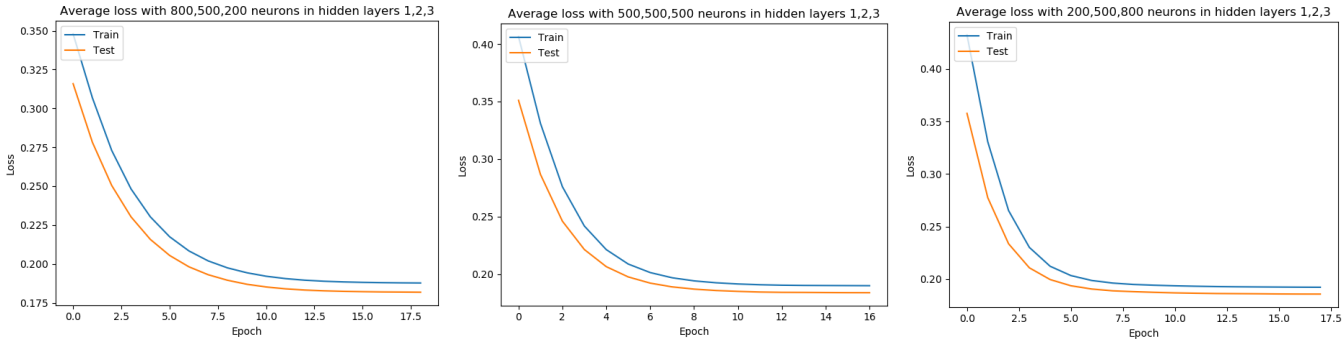
Παρατηρούμε τώρα το σφάλμα να μειώνεται φυσιολογικά στα διαγράμματα και να είναι αρκετά μικρότερο σε σχέση με την προηγούμενη επιλογή μας. **Ως τιμή του συντελεστή φθοράς θα επιλέξουμε την 0.1 διότι δίνει το μικρότερο σφάλμα.** Βέβαια, συγκρίνοντας με τα αποτελέσματα του ερωτήματος Α3, επιλέγοντας τις παραμέτρους η=0.1 και m=0.6 χωρίς ομαλοποίηση είχαμε σφάλμα πολύ κοντά στο τωρινό (0.189 σε σχέση με 0.185). Επομένως σίγουρα η ομαλοποίηση βοήθησε στην περίπτωση που ο ρυθμός εκπαίδευσης ήταν χαμηλός, αλλά αυξάνοντας τον ρυθμό έχουμε παρόμοια αποτελέσματα, και αυτό δείχνει ότι το μοντέλο μας γενικεύει αρκετά καλά και χωρίς ομαλοποίηση.

Α5. Βαθύ Νευρωνικό Δίκτυο

Θα χρησιμοποιήσουμε 3 κρυφά επίπεδα στο συγκεκριμένο ερώτημα για να πειραματιστούμε με το πλήθος των νευρώνων. Θα θέσουμε ως παραμέτρους: η=0.1 m=0.6 χωρίς ομαλοποίηση, δεδομένων των συμπερασμάτων μας στο ερώτημα Α4. Ο κώδικας βρίσκεται στο αρχείο “deep_neural_network.py”.

Μια αρχική ιδέα θα ήταν να ακολουθήσουμε την λογική των embeddings όπως κάναμε στο ερώτημα Α2. Δηλαδή οι νευρώνες του κρυφού επιπέδου να είναι μικρότερη σε πλήθος από τους νευρώνες των επιπέδων πριν και μετά το επίπεδο αυτό (K<M,N). Επομένως αναδρομικά το επίπεδο 1 θα έχει K1<M,N νευρώνες, το K2<M,K1, το K3<M,K2, και επομένως K3<K2<K1<M,N. Επίσης, εξασφαλίζει χαμηλότερο σφάλμα να ξεκινούμε από ένα πολύπλοκο μοντέλο και να το απλοποιήσουμε σταδιακά (μειούμενο πλήθος νευρώνων) παρά να ξεκινήσουμε από ένα απλό μοντέλο, χάνοντας πολύ πληροφορία, και να προσπαθήσουμε να «παράγουμε» πολυπλοκότητα από αυτό (αυξανόμενο πλήθος νευρώνων). Θα δοκιμάσουμε πειραματικά διάφορες επιλογές για να εξετάσουμε την ιδέα μας.

Συνδυασμός πλήθους νευρώνων στα κρυφά επίπεδα	RMSE	MAE
K1=800,K2=500,K3=200	**0.999**	**0.186**
K1=500,K2=500,K3=500	1.000	0.189
K1=200,K2=500,K3=800	1.000	0.192



Εικόνα 13: Πείραμα με διάφορους συνδυασμούς νευρώνων

Παρατηρούμε ότι το ελάχιστο σφάλμα εμφανίζεται στην επιλογή μειούμενου πλήθους νευρώνων, ενώ στην επιλογή αυξανόμενου πλήθους εμφανίζεται το μεγαλύτερο σφάλμα. Έτσι επιβεβαιώνουμε την υπόθεση μας.