NEURAL NETWORKS
3Ο ΠΑΡΑΔΟΤΈΟ (RBFNN)

- Καλλιμάνης Ιωάννης
- 10007
- Ηλεκτρολόγων μηχανικών και Μηχανικών Ηλεκτρονικών Υπολογιστών
- ikallima@ece.auth.gr

- 11-1-2024

Εισαγωγή

- Στο τρίτο μέρος της εργασίας, ζητήθηκε να εξετάσουμε την απόδοση τον RBF νευρωνικών δικτύων σε κάποια βάση δεδομένων. Όπως και στα προηγούμενα δύο μέρη, αποφάσισα να ασχοληθώ με την Cifar-10, ώστε να έχω ένα καλύτερο μέτρο σύγκρισης ως προς την απόδοση, καθώς έχουν ήδη εξεταστεί αλγόριθμοι για με την σειρά NearestCentroid, NearestNeighbor, MLP, CNN και διαφόρων ειδών SVM.
- Η χρήση των RBFNN δικτύων οδηγεί σε μια μείωση διάστασης των χαρακτηριστικών των δεδομένα στο στρώμα εισόδου, ώστε στην συνέχεια μέσω ενός πλήρως συνεκτικού MLP να γίνει πρόβλεψη των ετικετών των δεδομένων μας.

RBF – layer εισόδου

- Για την μετατροπή των δεδομένων μας, στα καινούργια δεδομένα διαφορετικής διάστασης, απαιτείται ο υπολογισμός κάποιων χαρακτηριστικών. Ως νέα δεδομένα μπορούμε να θεωρήσουμε την ομοιότητα που έχει το κάθε ένα από τα παραδείγματα μας ξεχωριστά σε σχέση με διάφορα κέντρα κλάσεων που έχουμε ορίσει. Αξίζει να σημειωθεί σε αυτό το σημείο ότι ως κλάσης δεν αναφερόμαστε μόνο στις πραγματικές κλάσεις των δεδομένων μας αλλά και σε καινούργιες που προκύπτουν από αλγορίθμους ομαδοποιήσεων όπως ο Kmeans.
- Ο όρος ομοιότητας που αναφέρθηκε παραπάνω μπορεί να έχει πολλές ερμηνείες. Για την ορίσουμε κάπως ώστε να περάσουμε στην υλοποίηση του μοντέλου μας ως ομοιότητα ορίσαμε μια πραγματική συνάρτηση της ευκλείδιας απόστασης κάποιου παραδείγματος ως προς τα κέντρα των εκάστοτε κλάσεων.
- Είναι, λοιπόν, φανερό ότι η διάσταση του νέου χώρο χαρακτηριστικών που προκύπτει είναι ίση με τον αριθμό των κέντρων που έχουμε ορίσει.

Συνάρτηση μετασχηματισμού

- Για να ορίσουμε την συνάρτηση μετασχηματισμού χρειαζόμαστε 2 διανυσματα

a) Το Κέντρο της κάθε κλάσης (c): Προκύπτει από την μέση τιμή του κάθε χαρακτηριστικού των παραδειγμάτων της κλάσης

b) Την ακτίνα της κάθε κλάσης (r): Είναι η απόσταση του παραδείγματος της κλάσης που απέχει περισσότερο από το κέντρο c.

- Οι συναρτήσεις που ελέχθηκαν είναι η Gaussian (1) και η Multiquadric (2)

$$e^{\frac{-(x-c_i)^2}{r^2_i}}$$, για κάθε κλάση i και κάθε δείγμα(εντός και εκτός κλάσης) x

(1)

$$\sqrt{\frac{r^2_i-(x-c_i)^2}{r_i}}$$ για κάθε κλάση i και κάθε δείγμα x

(2)

Περιγραφή Κώδικα

- Στην αρχή του κώδικα που δοκιμάστηκαν οι αποδόσεις με διάφορες τεχνικές ομαδοποίησης και με τις δύο συναρτήσεις μετασχηματισμού είτε με είτε χωρίς pca, χρησιμοποιώντας έναν μονοστρωματικό νευρωνικό δίκτυο μετά τον μετασχηματισμό με 40 εποχές εκπαίδευσης, optimizer Adam, CrossEntropy συνάρτηση κόστους και ρυθμό μάθησης 0,001 κατά κύριο λόγο.
- Στόχος μου ήταν η διερεύνηση της καλύτερης ομαδοποίησης και μετέπειτα η εφαρμογή σε πιο περίπλοκα νευρωνικά ώστε να αυξήσω την απόδοση μου.

Πορεία Κώδικα - Παραδείγματα

- Στην αρχή υλοποιήθηκε η ομαδοποίηση μέσω των ετικετών των δειγμάτων εκπαίδευσης. Δηλαδή προέκυπτε από την κάθε πραγματική κλάση το κέντρο και η ακτίνα της (συνολικά 10). Τα αποτελέσματα:

1). Χωρίς Pca

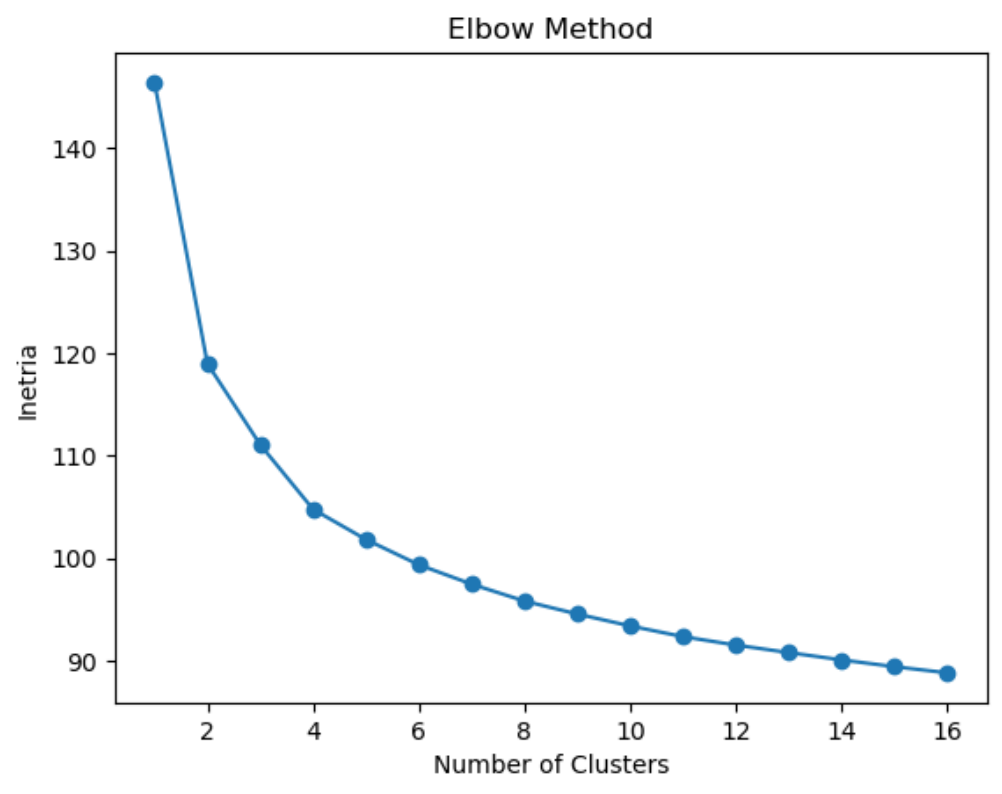
	Gaussian	Multiquadric
Transform Time	8.288 seconds	9.153 seconds
Training Accuracy	21.538 %	18.648 %
Testing Accuracy	21.72 %	19.47 %
Training time	34.376 seconds	36.497 seconds

2). Με Pca

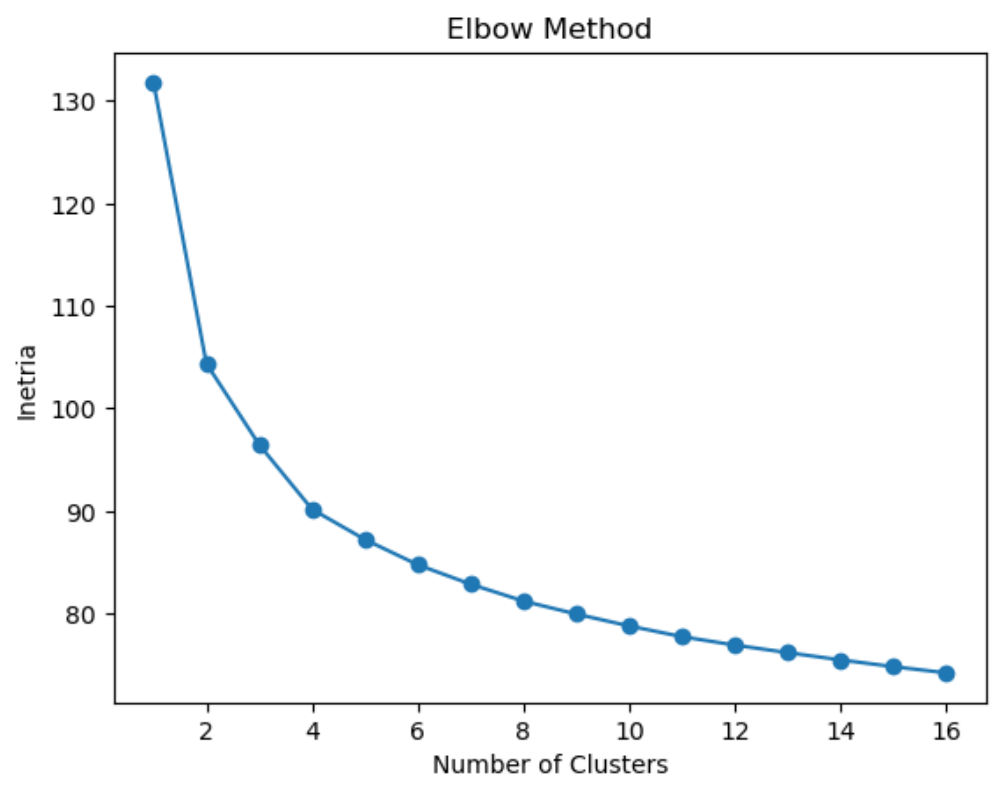
	Gaussian	Multiquadric
Transform Time	5.868 seconds	6.566 seconds
Training Accuracy	20.262 %	19.278 %
Testing Accuracy	20.26 %	19.2 %
Training time	35.046 seconds	35.134 seconds

- Στην πορεία δοκίμασα να εφαρμόσω τον κανόνα του αγκώνα για την ομαδοποίηση μέσω Kmeans, για να δω σε πόσες κλάσεις χωρίζονται τα δεδομένα μου ιδανικά μέσω μόνο τις θέσεις τους στον χώρο. Όπως παρατηρείται και από την εφαρμογή και με και χωρίς PCA ο αγκώνας εμφανίζεται για Number of Clusters ~ 6

1). Χωρίς PCA



2). Με PCA



- Όποτε αν χωρίσουμε τα δεδομένα μας με αριθμό ομάδων ίσο με 6, θα προκύψουν νέα κέντρα με νέες ακτίνες. Τα αποτελέσματα από την χρήση αυτού του μοντέλου είναι (πλέον στον χρόνο του transform μόνο για την περίπτωση του Gaussian έχει προστεθεί και ο χρόνος αρχικοποίησης του μοντέλου). Οι τιμές στις αποδόσεις που προέκυψαν δεν παρουσιάζουν σημαντική διαφορά με πριν:

- 1). Χωρίς PCA

	Gaussian	Multiquadric
Transform Time	88.911seconds	4.664 seconds
Training Accuracy	21.284 %	18.574 %
Testing Accuracy	21.71 %	18.574 %
Training time	35.398 seconds	34.284 seconds

2). Με PCA

	Gaussian	Multiquadric
Transform Time	5.876 seconds	3.885 seconds
Training Accuracy	21.152 %	18.434 %
Testing Accuracy	21.152 %	18.45 %
Training time	34.737 seconds	34.680 seconds

- Στην συνέχεια έγινε μια προσπάθεια ομαδοποίησης σε 20 ομάδες μέσω της Kmeans με εφαρμογή στο συνολικό dataset, ώστε να ανεβάσουμε τα δεδομένα σε υψηλότερη διάσταση από 10 που αποτελεί τον αριθμό των νευρώνων εξόδου. Σε αυτό το σημείο αρχίζει να γίνεται εμφανές ότι αν και λαμβάνουμε σταδιακά καλύτερες αποδόσεις ο χρόνος αρχικοποίησης και μετασχηματισμού των δεδομένων μας αρχίζει να ξεφεύγει:

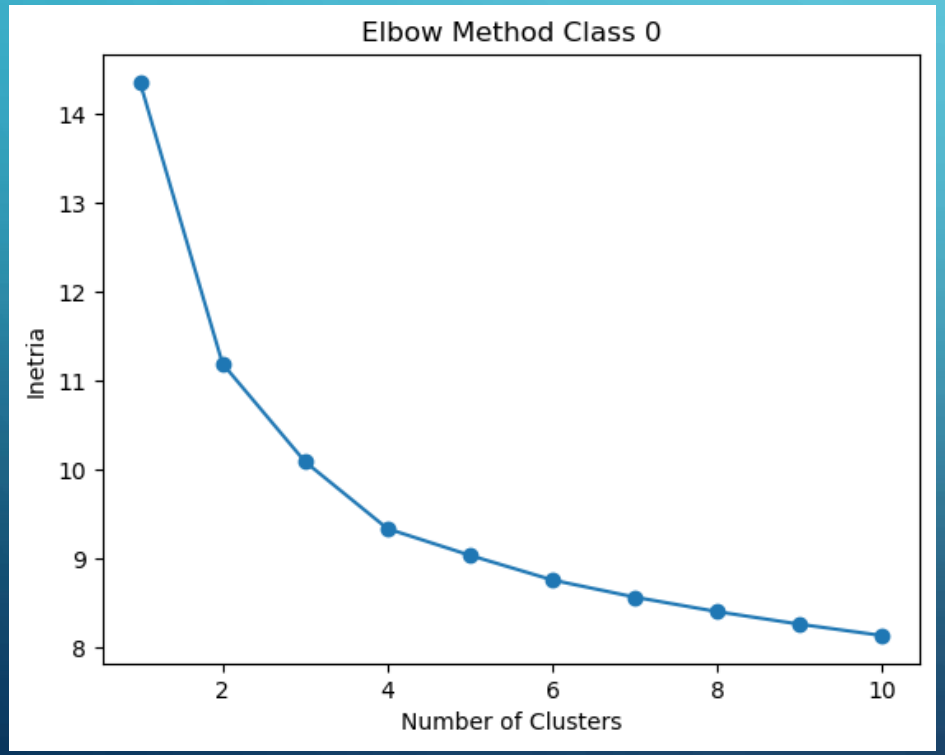
- 1). Χωρίς PCA

	Gaussian	Multiquadric
Transform Time	322.437 seconds	15.868 seconds
Training Accuracy	26.636 %	23.51 %
Testing Accuracy	27.43 %	23.96 %
Training time	34.497 seconds	35.518 seconds

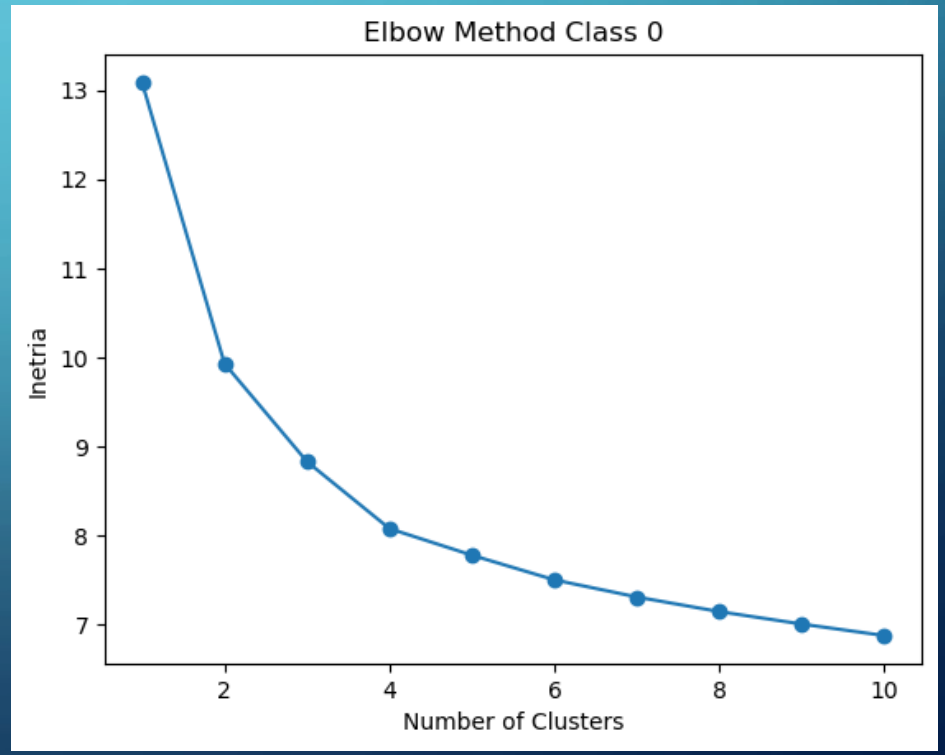
2). Με PCA

	Gaussian	Multiquadric
Transform Time	20.372 seconds	13.222 seconds
Training Accuracy	26.674 %	23.458 %
Testing Accuracy	27.26 %	24.03 %
Training time	35.201 seconds	34.721 seconds

- Σαν τελευταία προσπάθεια να εξάγω κάποια συμπεράσματα για το πώς θα ομαδοποιήσω τα δεδομένα που, πως και ποιες κλάσεις να λάβω καθώς και τι μετασχηματισμό / μετασχηματισμούς να χρησιμοποιήσω αποφάσισα να εφαρμόσω εκ νέου τον κανόνα του αγκώνα αλλά για κάθε κλάση ξεχωριστά. Τα διαγράμματα που προέκυψαν για κάθε κλάση είναι περίπου τα ίδια, αλλά τα συμπεράσματα που εξάγουμε κοινά. Ο αγκώνας και με και χωρίς PCA λαμβάνεται για Number of Clusters = 4
- 1). Χωρίς PCA



2). Με PCA



- Από την παραπάνω υλοποίηση προκύπτουν συνολικά 4 κέντρα για κάθε κλάση δηλαδή συνολικά 40 κέντρα με της αντίστοιχες ακτίνες. Τα αποτελέσματα:
- 1). Χωρίς PCA

	Gaussian	Multiquadric
Transform Time	96.477 seconds	26.972 seconds
Training Accuracy	26.844 %	23.828 %
Testing Accuracy	27.93 %	24.67 %
Training time	34.161 seconds	35.482 seconds

2). Με PCA

	Gaussian	Multiquadric
Transform Time	17.610 seconds	14.906 seconds
Training Accuracy	26.824 %	26.952 %
Testing Accuracy	27.82 %	27.98 %
Training time	34.433 seconds	35.943 seconds

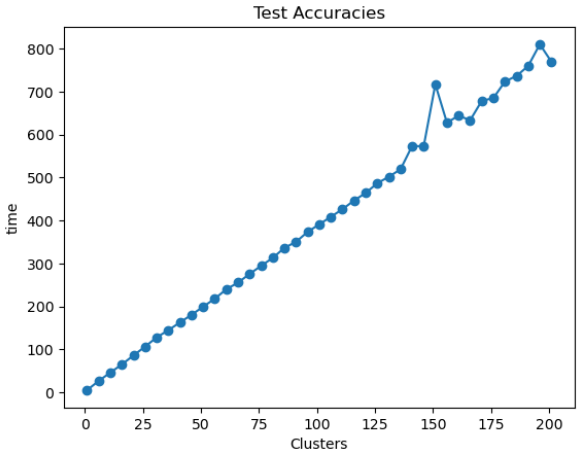
Αρχικά Συμπεράσματα

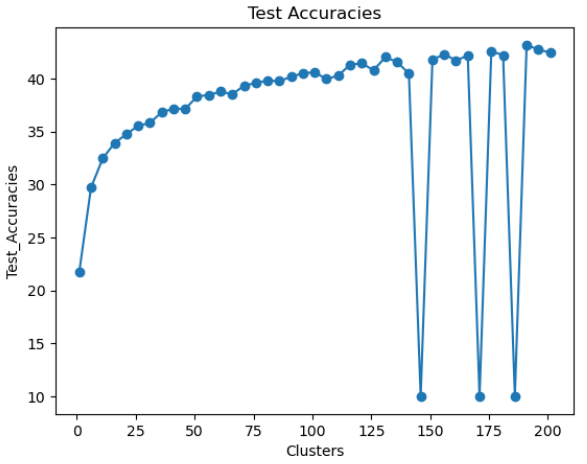
- Τα αποτελέσματα που προέκυψαν από την παραπάνω ανάλυση οδήγησαν στην εξαγωγή σημαντικών συμπερασμάτων. Η εφαρμογή ή όχι του PCA δεν εμφανίζει διαφορές στην απόδοση του μοντέλου, αλλά παρουσιάζει σημαντικές διαφορές στον χρόνο αρχικοποίησης και μετασχηματισμού των δεδομένων μέσω του RBF δικτύου, όποτε και προτιμάται. Επιπλέον ο μετασχηματισμός μέσω της gaussian συνάρτησης παρουσιάζει καλύτερες αποδόσεις σε σχέση με αυτόν της multiquadric, για αυτό επίσης προτιμάται.

Συνέχεια Πορείας Κώδικα

- Αφού συμπέρανα αρχικά ότι με τις 40 κλάσεις του τελευταίου μοντέλου που παρουσιάστηκε έχω καλές αποδόσεις δοκιμάστηκε η εφαρμογή του σε νευρωνικό με 2 ή και 3 στρωμάτων, που αύξησαν την απόδοση του δικτύου μου κατά ~15%.
- Παρόλα αυτά θέλησα να εμβαθύνω στην ταχτική ομαδοποίησης της κάθε κλάσης ξεχωριστά μέσω του Kmeans και μέσω ενός μονοστρωματικού δικτύου να ελέγξω σε πόσες ομάδες να χωρίσω τα δεδομένα μου.

- Το πρώτο διάγραμμα παρουσιάζει τον χρόνο αρχικοποίησης και μετασχηματισμού των δεδομένων, ενώ το δεύτερο την ακρίβεια που πετυχαίνει το μοντέλο, και τα δύο σε σχέση με τον αριθμό των ομάδων της κάθε κλάσης.
- Παρατηρούμε ότι ο χρόνος αυξάνεται γραμμικά, ενώ οι αποδόσεις φαίνεται να συγκλίνουν μετά τις 75-100 ομάδες.





Επιλογή Υπερπαραμέτρων

- Στην πορεία, αποφάσισα να δοκιμάσω για τρεις-τέσσερις τιμές του αριθμού των ομάδων διάφορες τιμές για τους νευρώνες των κρυφών επιπέδων σε μοντέλο με 3 κρυφά στρώματα νευρώνων ώστε να επιλέξω τις καλύτερες από αυτές. Δοκιμάστηκαν επίσης τιμές για τους λόγους dropout της συνάρτησης nn.Dropout που συμβολίζουν τον λόγο drop out για κάθε κρυφό επίπεδο. Έτσι οι τιμές που έλαβα ως ιδανικές ήταν
- Number_of_clusters=150hidden_size1=512,hidden_size2=1024,dropout1=0.2, dropout2=0.
- Με τα αποτελέσματα που προκύπτουν να είναι τα εξής:

	Gaussian
Transform Time	586.5 seconds
Training Accuracy	59.168 %
Testing Accuracy	48.32 %

Επιλογή optimizer και συνάρτησης ενεργοποίησης

- Στην συνέχεια υλοποιήθηκε ένας παρόμοιος έλεγχος για την επιλογή του βέλτιστου optimizer και της ιδανικής συνάρτησης ενεργοποίησης για τα δεδομένα της Cifar-10.
- Διαπιστώθηκε εκ νέου ότι ιδανικές ήταν οι επιλογές που είχα κάνει εξαρχής για optimizer Adam και συνάρτηση ενεργοποίησης ReLU
- Αξιοσημείωτες είναι και οι συναρτήσεις Swish, GELU, SiLU που δίνουν αποτελέσματα κοντά σε αυτά της ReLU.

Τετραπλό MLP

- Στην πορεία έγινε και υλοποίηση τετρα-στρωματικού νευρωνικού δικτύου και επιλογή των βέλτιστων παραμέτρων κρυφών νευρώνων και λόγων dropout, αφού προηγήθηκε και πέρασμα από autoencoder σε περίπτωση που τα δεδομένα είχαν κάποιον θόρυβο(πράγμα που όπως φανερώνεται και από τα loss που εκτυπώθηκαν ήταν κάπως αχρείαστο).
- Οι τιμές που προέκυψαν είναι:
- Input_size=1500, hidden_size1=1024, hidden_size2 = 512, hidden_size3 = 64, dropout1=0. , dropout2=0.1 , dropout3=0.3.
- Τα αποτελέσματα που προέκυψαν είναι:
- Training Accuracy of the network: 53.37 %
- Testing accuracy of the network: 49.16 %

Ενισχυμένο Μοντέλο

- Στην συνέχεια το τελευταίο μοντέλο χρησιμοποιήθηκε για την δημιουργία 30 παρόμοιων μοντέλων που θα προέβλεπε το καθένα κάποια ετικέτα. Αυτή που επιλέγεται τελικά είναι αυτή με τις περισσότερες ψήφους από αυτά τα μοντέλα.
- Τα αποτελέσματα που προέκυψαν είναι 51.21%