ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

ΕΥΦΥΕΙΣ ΤΕΧΝΟΛΟΓΙΕΣ ΔΙΑΔΙΚΤΥΟΥ - WEBINTELLIGENCE

Χρήση μηχανικής μάθησης για την ανίχνευση της νόσου Alzheimer από ηλεκτροεγκεφαλικά σήματα

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΜΗΣΙΟΥ ΑΛΕΞΑΝΔΡΑ

Επιβλέπων : Δρ. Κωνσταντίνος Διαμαντάρας
Καθηγητής, ΔΙ.ΠΑ.Ε.

Θεσσαλονίκη, Οκτώβριος 2022

ΔΙΕΘΝΕΣ ΠΑΝΕΠΙΣΤΗΜΙΟ ΤΗΣ ΕΛΛΑΔΟΣ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ ΕΥΦΥΕΙΣ ΤΕΧΝΟΛΟΓΙΕΣ ΔΙΑΔΙΚΤΥΟΥ – WEB INTELLIGENCE

Χρήση μηχανικής μάθησης για την ανίχνευση της νόσου Alzheimer από ηλεκτροεγκεφαλικά σήματα

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΜΗΣΙΟΥ ΑΛΕΞΑΝΔΡΑ

Επιβλέπων : Δρ. Κωνσταντίνος Διαμαντάρας
Καθηγητής ΔΙ.ΠΑ.Ε.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή στις 1 Οκτωβρίου 2022.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Δρ. Κων. Διαμαντάρας	Όνομα Επώνυμο	Όνομα Επώνυμο
Καθηγητής ΔΙ.ΠΑ.Ε.	Choose an item. ΔΙ.ΠΑ.Ε.	Choose an item. ΔΙ.ΠΑ.Ε.

Θεσσαλονίκη, Οκτώβριος 2022

(Υπογραφή)

.....

Μήσιου Αλεξάνδρα

Μηχανικός Πληροφορικής Α.Τ.Ε.Ι. Θεσσαλονίκης

Περίληψη

Τα τελευταία χρόνια γίνονται ραγδαία άλματα στον τομέα Τεχνητής Νοημοσύνης. Έξυπνα σπίτια, πόλεις, εφαρμογές κάνουν την εμφάνιση τους χρησιμοποιώντας αλγορίθμους Μηχανικής Μάθησης. Πλέον όμως οι αλγόριθμοι της Μηχανικής Μάθησης προσπαθούν να μιμηθούν τον ανθρώπινο εγκέφαλο έτσι ώστε μια εφαρμογή να μπορέσει, αφού εκπαιδευτεί πάνω σε κάποια δεδομένα, να βρίσκει μόνη της τις λύσεις που επιδιώκει ο χρήστης της.

Η παρούσα εργασία έχει στόχο την κατηγοριοποίηση των ασθενών σε αυτούς που πάσχουν από Alzheimer's (AD), σε αυτούς που πάσχουν από ήπια μορφή άνοιας (MCI) και σε αυτούς που είναι υγιείς (Healthy). Σαν δεδομένα εισαγωγής δόθηκαν ηλεκτροεγκεφαλογραφήματα (EEG) τα οποία τοποθετήθηκαν με το διεθνές σύστημα τοποθέτησης 10-20. Γίνεται σύγκριση διαφόρων μεθόδων. Τα EEG διαχωρίστηκαν σε τμήματα. Για τη μέθοδο 2D Συνελικτικών Νευρωνικών Δικτύων (Convolutional Neural Networks - CNN) διαχωρίστηκαν ανά 5 δευτερόλεπτα, ενώ για τις υπόλοιπες μεθόδους ανά 2 δευτερόλεπτα.

Έγιναν πειράματα και ελέγχθηκε και ο ρόλος των διάφορων μεθόδων προεπεξεργασίας των EEG, όσο και ο ρόλος των παραμετροποιήσεων των μεθόδων. Τα δεδομένα διαχωρίστηκαν σε τμήματα (segments). Όσο αφορά την προεπεξεργασία δεδομένων με τη χρήση του φίλτρου band – pass filter, δεν φάνηκε να αποδίδει σε καμία από τις τρεις κατηγορίες. Αντίθετα η χρήση Fourier, Principal Component Analysis (PCA) και ScandardScaler φάνηκαν να βελτιώνουν σημαντικά την απόδοση των μοντέλων.

Αναφορικά με την επιλογή του μοντέλου η αρχική προσέγγιση ήταν μέσω των 2D-CNN η οποία δεν απέδωσε και ακολούθησε σύγκριση 8 ακόμη μεθόδων. Από αυτές τις μεθόδους ακολούθησε διεξοδικότερη ανάλυση και παραμετροποίηση των τριών που έδωσαν τα καλύτερα αποτελέσματα: Random Forest, K Nearest Neighbors (KNN) και Support Vector Machines (SVM). Έπειτα δοκιμάστηκαν και αναφέρονται οι παραμετροποιήσεις που έγιναν στις τρεις μεθόδους. Η εκτίμηση των αποτελεσμάτων έγινε μέσω της ακρίβειας (accuracy). Στα πειράματα που το test – set ήταν τυχαία segments έγινε χρήση της cross-validation η οποία δε φάνηκε να αποδίδει σε όλες τι μεθόδους. Έγιναν και δοκιμές ανά ασθενή, δηλαδή αφαιρέθηκαν όλα τα τμήματα των EEG ενός ασθενή από τα δεδομένα εκπαίδευσης (train set) και αποτέλεσαν τα δεδομένα που δοκιμάστηκαν (test set).

Λέξεις Κλειδιά: Μηχανική Μάθηση, Alzheimer's, Ταξινομητές, Σύγκριση Ταξινομητών.

Abstract

In recent years have been rapid developments in the field of Artificial Intelligence. Smart homes, cities, applications appear using Machine Learning algorithms. But now the Machine Learning algorithms try to imitate the human brain so that an application can, after being trained on some data, find the solutions that its user is looking for.

This research aims to categorize patients into those suffering from Alzheimer's (AD), those suffering from Mild Cognitive Impairment (MCI) and those who are healthy (Healthy). Electroencephalograms (EEGs) were provided as input data and were placed using the international 10-20 placement system. Various methods are compared. The EEGs were divided into segments. For the 2D Convolutional Neural Network method (2D CNN) they were separated every 5 seconds, while for the other methods every 2 seconds.

Experiments were carried out and the role of various EEG preprocessing methods was tested, as well as the role of the parameters tuning of the methods. The data was divided into segments. Regarding data processing band-pass filter did not appear to perform well in any of three categories. On the contrary, the use of Fourier, Principal Component Analysis (PCA) and StandardScaler appeared to significantly improve the performance of the models.

Regarding the selection of the model, the initial approach was through 2D-CNN which did not yield and followed by a comparison of 8 more methods. From these methods followed a more thorough analysis and parameters tuning of the three that gave the best results: Random Forest, KNeighbors and SVM. All parameters tuning that could be made in the three methods were then tested and reported. The assessment of the results was done through accuracy. In the experiments where the test-set was random segments, cross-validation was used, which did not seem to work for all methods. Tests were also done per patient, which mean all parts of one patient's EEG were removed from the training data (train set) and formed the tested data (test set).

Keywords: Machine Learning, Alzheimer’s Disease, Classifiers, Comparison of classifiers.

Θα ήθελα να ευχαριστήσω τον κύριο Κ. Διαμαντάρα για την πολύτιμη συμβολή του και καθοδήγηση στη διάρκεια της διπλωματικής μου εργασίας καθώς επίσης και τη διάνοιξη νέων οριζόντων κατά τη διάρκεια των μαθημάτων.

Πίνακας περιεχομένων

1 *Εισαγωγή*

1.1 Εφαρμογή της Μηχανικής Μάθησης στην Ιατρική

Η χρήση των υπηρεσιών που παρέχει η Μηχανική Μάθηση αυξάνεται όλο και περισσότερο. Βρίσκει εφαρμογή σε όλο και περισσότερους τομείς διαφόρων επιστημών ή και της καθημερινής μας ζωής. Μια από της επιστήμες που χρήζει βοήθεια από την επιστήμη των υπολογιστών είναι η Ιατρική. Προβλήματα αναγνώρισης ασθενειών, κυρίως αυτών που δεν γίνονται αντιληπτές εύκολα, ή η εξέλιξη κάποιων από αυτών καθώς η αντιμετώπιση τους είναι τομείς στους οποίους μπορεί να βρει εφαρμογή η Μηχανική Μάθηση. Η χρήση της Μηχανικής Μάθησης μπορεί να δώσει απαντήσεις σε θέματα που απασχολούν την Ιατρική ή μπορεί να βοηθήσει να έρθουν οι απαντήσεις σε πολύ πιο σύντομο χρονικό διάστημα από αυτό που θα χρειαζόταν εάν δεν γινόταν χρήση της Μηχανικής Μάθησης. Δεδομένου ότι ο χρόνος στην αντιμετώπιση μιας ασθένειας είναι πολύτιμος μπορεί κανείς να κατανοήσει τη μεγάλη συμβολή που θα έχει η εύρεση της λύσεων σε συντομότερο διάστημα ή με μεγαλύτερη ακρίβεια.

Διάφορα εργαλεία που έχουν στη φαρέτρα τους οι επιστήμονες αναλύουν και επεξεργάζονται καθημερινά τα δεδομένα που τους δίνονται έτσι ώστε να τα αξιοποιήσουν για να δώσουν γρήγορα και με ακρίβεια πληροφορίες που θα φανούν χρήσιμες για την βελτίωση της ποιότητας της ζωής των ανθρώπων. Η επιστήμη της πληροφορικής από τη μεριά της με τις κατάλληλες μεθόδους μηχανικής μάθησης, προσπαθεί να αποκωδικοποιήσει γρήγορα, με σαφήνεια και με μικρό ποσοστό λάθους τα δεδομένα αυτά, έτσι ώστε να παρέχει τη δυνατότητα να δοθούν όσο το γρηγορότερο λύσεις. Στη συγκεκριμένη εργασία έγινε προσπάθεια διάγνωσης της νόσου Alzheimer’s συγκρίνοντας διάφορες μεθόδους της μηχανικής μάθησης.

1.2 Αντικείμενο διπλωματικής

Σκοπός της παρούσας εργασίας είναι να ερευνηθεί κατά πόσο μπορεί να ανακαλυφθεί μέσω ηλεκτροεγκεφαλογραφημάτων η νόσος Alzheimer's. Πρόκειται για μια εκφυλιστική ασθένεια των εγκεφαλικών κυττάρων. Τα ηλεκτροεγκεφαλογραφήματα (EGG) δεν είναι ο μόνος τρόπος διάγνωσης. Γίνεται χρήση επίσης μαγνητικών ή αξονικών εξετάσεων.

Τα ηλεκτροεγκεφαλογραφήματα είναι μια εξέταση κατά την οποία τοποθετούνται σε συγκεκριμένα σημεία του εγκεφάλου αισθητήρες οι οποίοι καταμετρούν τη δραστηριότητα του εγκεφάλου η οποία συγχρόνως καταγράφεται. Συνήθως χρησιμοποιείται ο διεθνής τρόπος τοποθέτησής τους 10-20 (εικόνα 1). Αποτελεί μια εξέταση που είναι προσιτή ως προς το κόστος σε όλες τις ομάδες, αποτέλεσμα τις φθηνής αγοράς τέτοιων μηχανημάτων καθώς και του μικρού κόστους διατήρησης τους. Επίσης πέραν του κόστους κάποιοι ασθενείς κατά τη διάρκεια της μαγνητικής λόγω κλειστοφοβίας δε μπορούν να την ολοκληρώσουν, ακόμη και άτομα που δεν έχουν τέτοιες φοβίες δυσκολεύονται κάποιες φορές να την ολοκληρώσουν, κάτι που δεν συμβαίνει στην περίπτωση των ηλεκτροεγκεφαλογραφημάτων και καθιστά την εξέταση πιο εύκολη και ανώδυνη.

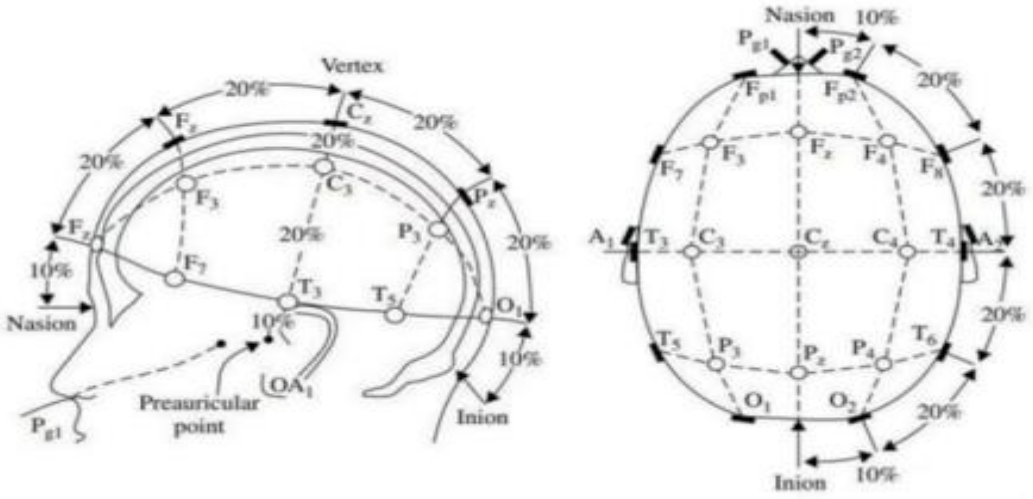

Figure 1. 10 -20 international standard lead [1]

1.2.1 Συνεισφορά - Προσέγγιση

Από την «Ελληνική Ένωση Νόσου Alzheimer's και συναφών διαταραχών» δόθηκαν δεδομένα 106 ασθενών. Από αυτά χρησιμοποιήθηκαν τα 54 με 18 από αυτά να αφορούν ασθενείς που είναι υγιείς, 18 ασθενείς με Ήπια Νοητική Διαταραχή (MCI – Mild Cognitive Impairment) και 18 με Alzheimer's (AD - Alzheimer's Disease). Αυτά τα δείγματα είναι δείγματα που κατά τη διάρκεια της εξέτασης οι ασθενείς είχαν κλειστά μάτια γιατί αυτές οι εξετάσεις έχουν λιγότερο θόρυβο οπότε καθιστούν το δείγμα πιο αξιόπιστο. Από το κάθε EGG πάρθηκαν 5 λεπτά, κάθε δευτερόλεπτο είναι 500 στιγμιότυπα άρα πάρθηκαν από κάθε ασθενή 150000 στιγμιότυπα τα

οποία για την εργασία ομαδοποιήθηκαν ανά 5 δευτερόλεπτα (2500 στιγμιότυπα) στην περίπτωση των δοκιμών με 2D Convolutional Neural Networks (CNN), και ανά 2 δευτερόλεπτα (1000 στιγμιότυπα) σε όλες τις υπόλοιπες μεθόδους.

Η αρχική προσέγγιση των μοντέλων μηχανικής μάθησης έγινε μέσω των CNN και συγκεκριμένα 2D.

Έπειτα ακολούθησε η μέθοδος Weight Average χρησιμοποιώντας τα παρακάτω 8 μοντέλα:

- Logistic Regression Classifier
- Decision Tree Classifier
- Random Forest
- Gaussian Naïve Bayes
- SVM (Support Vector Machines)
- K – πλησιέστερων γειτόνων (K-Nearest Neighbors KNN)
- AdaBoost
- QDA (Quadratic Discriminant Analysis)

Έγινε συνδυασμός χρήσης αυτών στην Weight Average και από τα αποτελέσματα επιλέχθηκαν και αναλύθηκαν περισσότερο οι 3 μέθοδοι με τα μεγαλύτερα ποσοστά επιτυχίας: Random Forest, SVM και KNN. Η αξιολόγηση των μεθόδων έγινε βάση του δείκτη ακρίβεια που αφορά το ποσοστό επιτυχίας ορθής τοποθέτησης των δειγμάτων που δόθηκαν κατά τη διαδικασία της δοκιμής του test set (του συνόλου δεδομένων που δόθηκε για να ελέγξουμε πόσο σωστά τοποθετήθηκαν δείγματα που είχαμε αποκρύψει από το αρχικό σύνολο δεδομένων). Εκτός από τον έλεγχο σε συνδυασμών των τριών κλάσεων που έχουμε, ελέγξαμε το σύνολο των δεδομένων και ανά δυο κλάσεις. Επιπροσθέτως κατόπιν του διαχωρισμού των EGG σε τμήματα (segments) έγινε έλεγχος και αποκρύπτοντας τυχαία δεδομένα (πειράματα ανά segment) αλλά και αποκρύπτοντας όλα τα δεδομένα ενός ασθενή (πειράματα ανά ασθενή).

1.3 Οργάνωση κειμένου

Το πρώτο κεφάλαιο περιέχει γενικές πληροφορίες για το αντικείμενο της διπλωματικής. Στο δεύτερο κεφάλαιο γίνεται αναφορά των μεθόδων για την προεπεξεργασία των EGG και των μοντέλων Μηχανικής Μάθησης. Στο 3ο κεφάλαιο γίνεται βιβλιογραφική ανασκόπηση των μοντέλων. Στο 4ο κεφάλαιο γίνεται η ανάλυση του προβλήματος. Στο 5ο γίνεται η ανάλυση των μοντέλων που χρησιμοποιήθηκαν και των παραμέτρων τους. Στο 6ο κεφάλαιο παρατίθενται τα αποτελέσματα των πειραμάτων καθώς και η αξιολόγηση αυτών. Στο 7ο κεφάλαιο αναφέρονται τεχνικές λεπτομέρειες των εργαλείων που χρησιμοποιήθηκαν και τέλος το 8ο γίνεται σύνοψη και καταγραφή ιδεών για μελλοντικές εργασίες.

2 *Σχετικές εργασίες*

Εξετάσεις EGG έχουν χρησιμοποιηθεί σε συνδυασμό με Μηχανική Μάθηση και για άλλες ασθένειες όπως ανίχνευση επιληπτικών κρίσεων [2]. Η εξέταση EGG ουσιαστικά είναι τοποθέτηση ηλεκτροδίων σε διάφορα σημεία του κρανίου (εικόνα 1) και κάθε ένα από αυτά τα ηλεκτρόδια μας δίνει ένα σήμα που λέγεται κανάλι. Μέσω των δεδομένων που ανιχνεύονται από τη δραστηριότητα του εγκεφάλου στα διάφορα κανάλια γίνεται προσπάθεια να κατηγοριοποιηθούν στις διάφορες ασθένειες που αφορούν κυρίως νευροεκφυλιστικές διαταραχές. Κύριο ρόλο ως προς την ορθότητα των αποτελεσμάτων έχει η προεπεξεργασία των δεδομένων. Η προσπάθεια δηλαδή να καθαριστεί ο θόρυβος και να ξεχωριστούν τα χαρακτηριστικά που θα μας βοηθήσουν να κατηγοριοποιήσουμε τα δεδομένα. Εξίσου σημαντική είναι όμως και η επιλογή του μοντέλου που θα χρησιμοποιηθεί. Στο παρόν κεφάλαιο αναφέρονται μέθοδοι προεπεξεργασίας δεδομένων καθώς και μέθοδοι μηχανικής μάθησης[2].

2.1 Προεπεξεργασία Δεδομένων

2.1.1 Band-pass filter

Πρόκειται για το πιο συχνά χρησιμοποιούμενο φίλτρο επεξεργασίας για τα EEG [3]. Ουσιαστικά φιλτράρει τα σήματα και εάν ένα σήμα ξεπεράσει το ανώτερο όριο αφαιρείται και το ίδιο συμβαίνει και αν περάσει κάτω από το κατώτερο όριο. Υπάρχει πάντα ο κίνδυνος απώλειας σημαντικών δεδομένων καθώς ιδίως στη συγκεκριμένη εργασία γίνεται χρήση ιατρικών δεδομένων.

2.1.2 Band-stop filter

Ένα άλλο φίλτρο επεξεργασίας των EEG είναι το Band-stop filter [3]. Είναι η εκ διαμέτρου αντίθετη τεχνική από την Band-pass filter. Εδώ δίνονται τα όρια και εάν είναι εντός των ορίων το σήμα αφαιρείται.

Ουσιαστικά τα φίλτρα χρησιμοποιούνται για την αφαίρεση του σήματος που υποθέτουμε ότι είναι θόρυβος και αλλοιώνει τα δεδομένα οπότε και προσπαθούμε να τα έχουμε όσο πιο καθαρά μπορούμε. Χρησιμοποιούνται συχνά στην επεξεργασία των EEG ή άλλων σημάτων στην απλή τους μορφή ή προσαρμοσμένα στις ανάγκες των εκάστοτε εφαρμογών [4]–[6] .

2.1.3 Independent Component Analysis - ICA

Χρησιμοποιείται για την εξαγωγή δεδομένων φιλτράροντας και κρατώντας μόνο τη χρήσιμη πληροφορία [7]. Δηλαδή Αναλύουμε τα σήματα σε στατιστικά ανεξάρτητες συνιστώσες. Αρκετές φορές τα σήματα, εικόνες, ο ήχος κτλ. έχουν πληροφορία που δεν είναι χρήσιμη ή είναι και επιβλαβής για την εξαγωγή σαφών αποτελεσμάτων. Η ICA χρησιμοποιείται για να καθαριστεί ένα κανάλι και να μείνει μόνο σαν δεδομένο η καθαρή πληροφορία.

2.2 Εξαγωγή Χαρακτηριστικών

2.2.1 Fourier Transform

Είναι μια τεχνική μετασχηματισμού σήματος [8]. Ο Jean Baptiste Fourier (1768-1830) απόδειξε ότι κάθε περιοδική y = f(x) συνάρτηση μπορεί να γραφεί ως ένα άπειρο άθροισμα ημιτονοειδών συναρτήσεων [9]. Χρησιμοποιείται αρκετά σε καθώς μειώνει κατά πολύ των αριθμό των αριθμητικών πράξεων που χρειάζονται. Στη ουσία απλοποιεί τα σήματα. Η πιο συνηθισμένη μορφή εισήχθη το 1965 και είναι η Fast Fourier Transform (FFT) [10]. Χρησιμοποιείται αρκετά στη Μηχανική Μάθηση με αποτέλεσμα τη βελτίωση των ποσοστών ταξινόμησης.

2.2.2 Principal Component Analysis - PCA

Η PCA (Principal Component Analysis) είναι μια τεχνική γραμμικής μείωσης διαστάσεων και επιλογής χαρακτηριστικών [11]. Αντλώντας το αρχικό σύνολο δεδομένων το συμπιέζει και βρίσκει ένα νέο σύνολο σημάτων που είναι στατιστικά ασυσχέτιστα. Μειώνει την πολυπλοκότητα των δεδομένων χρησιμοποιώντας τις κύριες συνιστώσες (principal component). Μειώνοντας την πολυπλοκότητα καθιστά τα δεδομένα πιο απλά και κατανοητά.

2.2.3 StandardScaler

Αφαιρεί τη μέση τιμή και κλιμακώνει τη διακύμανση. Η επεξεργασία γίνεται ξεχωριστά σε κάθε ένα χαρακτηριστικό και όχι μαζί στο σύνολο των δεδομένων. Η συγκεκριμένη μέθοδο δεν πετυχαίνει πάντα το σκοπό της καθώς η επιτυχίας της εξαρτάται από τον τύπο των δεδομένων. Για παράδειγμα αν κάποια χαρακτηριστικά δεν είναι τόσο κανικοποιημένα και ένα χαρακτηριστικό έχει μεγάλη διακύμανση τότε μπορεί να θεωρηθεί από τον αλγόριθμο ότι είναι πολύ σημαντικότερο από τα άλλα και η εκτίμηση να μην είναι η αναμενόμενη καθώς κάποια χαρακτηριστικά θα παραμεριστούν από το βασικό εκτιμητή.

2.3 Μοντέλα

Αν ανατρέξει κανείς στη βιβλιογραφία θα δει να χρησιμοποιούνται πολλά διαφορετικά μοντέλα. Παρατηρείται το φαινόμενο των λιγοστών δεδομένων που χρησιμοποιήθηκαν, για παράδειγμα υπάρχουν έρευνες με 12-23 ασθενείς συνολικά. Ακόμη και το δείγμα των 54 ασθενών που χρησιμοποιήθηκε στην παρούσα εργασία θεωρείται πολύ μικρό για να μπορέσουμε να μιλήσουμε για χρήση μοντέλων βαθιάς μάθησης. Επίσης συναντάται το φαινόμενο της ανομοιογένειας ως προς την κατανομή των δειγμάτων, υπερέχει συνήθως μια κλάση έναντι των υπολοίπων ή υστερεί μια έναντι των άλλων δυο. Επιπροσθέτως συνήθως ακόμη και σε αυτές που υπάρχουν δείγματα και από τις 3 κλάσεις γίνονται δοκιμές ανά δυο ζεύγη κλάσεων (Healthy με MCI, Healthy με AD και τέλος MCI με AD). Στην παρούσα εργασία παρότι στην αρχή επιχειρήθηκε η χρήση Συνελικτικών Νευρωνικών Δικτύων (Convolutional Neural Networks -CNN) τα αποτελέσματα δεν ήταν ικανοποιητικά. Γι' αυτό και σε δεύτερη προσπάθεια έγινε χρήση κλασσικών ταξινομητών και σύγκριση αυτών. Επιπροσθέτως στη βιβλιογραφία συναντάμε εργασίες που ερευνούν τα δεδομένα από τα ανοιχτά μάτια σε σχέση με αυτά των κλειστών [12], [13].

2.3.1 Κλασικοί Ταξινομητές

Στις περισσότερες περιπτώσεις στη βιβλιογραφία που αφορούν τη διάγνωση για το Alzheimer's θα συναντήσουμε τη χρήση των απλών ταξινομητών. Η ευκολία στη χρήση τους, οι λιγότεροι πόροι που χρειάζονται, η πιο γρήγορη εκτέλεση τους και η πιο εύκολη κατανόηση του τρόπου που λειτουργούν τους καθιστά πιο δημοφιλείς σε σχέση με βαθιά μοντέλα μάθησης. Support Vector Machines - SVM χρησιμοποίησαν οι Trambaiolli κα [14] (07/2011) χρησιμοποιώντας σαν δείγμα 16 ασθενείς AD και 19 υγιείς. Για την εκτίμηση των αποτελεσμάτων χρησιμοποίησαν τη μέθοδο cross spectrum σε ζευγάρια ανά δυο ηλεκτρόδια. Χώρισαν τα δεδομένα ανά ζώνες, τα 19 ηλεκτρόδια είχαν τοποθετηθεί με το διεθνές σύστημα τοποθέτησης 10-20. Η ακρίβεια που πέτυχε σε όλα τα πειράματα δεν ήταν κάτω από 80%. Οι Staudinger κα [15] (12/2011) είχαν αρκετά περισσότερα δείγματα στη διάθεση τους με τους ασθενείς AD να φτάνουν τους 79 και τους υγιείς τους 82. Χρησιμοποίησαν 16 ηλεκτρόδια, που τοποθετήθηκαν σύμφωνα με το διεθνές 10-20 σύστημα τοποθέτησης, χρησιμοποιήθηκαν στην έρευνα εξαίροντας από τα δεδομένα τα Fp1 και FP2 λόγω των συχνών οπτικών αντιδράσεων. Χρησιμοποίησαν cross validation για τη βελτίωση των αποτελεσμάτων. Χρησιμοποίησαν δυο μοντέλα για την εκτίμηση το Neural Network με επιτυχία μέχρι 66% και SVM με επιτυχία μέχρι 78% ακρίβεια.

Ο Pogorelec [16] (10/2012) δημιούργησε δικό του αλγόριθμο που τον ονόμασε genTrees, βασισμένο στα Decision Trees και είχε επιτυχία 86,05%. Είχε στη διάθεσή του ένα σετ από

208 άτομα. Από τα 16 ηλεκτρόδια χρησιμοποίησε μόνο δυο έτσι ώστε να βελτιώσει την απόδοση του αλγόριθμου του. Οι Kulkarni κα [17] (01/2017) στην εργασία τους είχαν δείγματα από 50 ασθενείς AD και 50 υγιείς. Χρησιμοποίησαν 16 ηλεκτρόδια τοποθετημένα σύμφωνα με το διεθνές σύστημα τοποθέτησης 10-20. Η αφαίρεση των τμημάτων που αφορούσαν πχ βλεφάρισμα ή μεγάλη διακύμανση έγινε χειρωνακτικά. Το ποσοστό πρόβλεψης έφτασε μέχρι 96% με τη χρήση της SVM. K-Nearest Neighbors (KNN) χρησιμοποίησαν στην έρευνά τους οι Al-nuaimi κα [18] (07/2017) χρησιμοποιώντας ένα ανομοιογενές δείγμα από 20 ασθενείς AD και 49 υγιών. Χρησιμοποιήθηκαν 21 ηλεκτρόδια σύμφωνα με το διεθνές σύστημα τοποθέτησης 10-20. Από αυτά τα 5 χρησιμοποιήθηκαν για να γίνει διαχωρισμός ανάμεσα σε AD και Healthy με ποσοστό ακρίβειας του Knn 84.61%.

Οι Durongbhan κα στην εργασία τους [19] (02/2019) χρησιμοποίησαν επίσης ένα ανομοιογενές δείγμα 20 ασθενών AD και 10 υγιών. Χρησιμοποιήθηκαν 19 ηλεκτρόδια σε ζεύγη, που είχαν τοποθετηθεί με ένα τροποποιημένο 10-10 σύστημα. Έκαναν χρήση του αλγορίθμου KNN με 10-fold cross-validation φτάνοντας το 99,23% σε ένα ζεύγος καναλιών. Οι Tavarest κα στην εργασία τους [20] (7/2019) χρησιμοποίησαν επτά μοντέλα απλών ταξινομητών: Logistic Regression, SVM, Random Forest, Extra Tree, SGD, AdaBoost και Gradient Boost με τις τρεις τελευταίες να εμφανίζουν το μεγαλύτερο ποσοστό επιτυχίας 97,14% όσο αφορά την ακρίβεια. Όταν αφαιρούνται χαρακτηριστικά η SVM φαίνεται να κερδίζει έδαφος. Χρησιμοποίησαν μόνο από δυο ασθενών Alzheimer's και υγιών τα οποία δεν ήταν ισόποσα 19 έναντι 16. Χρησιμοποίησαν 33 ηλεκτρόδια ακολουθώντας ένα 10-10 σύστημα τοποθέτησης. Διαχώρισαν τα δεδομένα ανά ζώνες (δέλτα, θήτα, άλφα, βήτα) και χρησιμοποίησαν φίλτρα low-pass. Χρησιμοποιήθηκαν 224 χαρακτηριστικά και στην πορεία αφαιρέθηκαν κάποια.

Οι Safi κα στην έρευνά τους [21] (03/2021) έκαναν σύγκριση μεθόδων. Τα δείγματα που είχαν ήταν τριών κατηγοριών 20 AD, 35 Healthy και 31 mild AD. Χρησιμοποίησαν 20 ηλεκτρόδια τοποθετημένα με το διεθνές σύστημα τοποθέτησης 10-20. Οι μέθοδοι που επέλεξαν ήταν KNN, SVM και RLDA οι οποίες έφτασαν σε ποσοστό ακρίβειας μέχρι και 97,64%, 95,79% και 97,02% αντίστοιχα με την KNN να έρχεται πρώτη. Τον Μάρτιο του 2022 οι Puri κα στην εργασία τους [22] έκαναν σύγκριση επτά μοντέλων: KNN, SVM, Random Forest, MLDNN, Naïve Bayes και AdaBoost με την SVM να έχει το μεγαλύτερο ποσοστό ακρίβειας 97,80%. Χρησιμοποίησαν επίσης μόνο ασθενείς με Alzheimer's και υγιείς, 12 και 11 αντίστοιχα. Χρησιμοποιήθηκαν 16 ηλεκτρόδια τα οποία ήταν σύμφωνα με την 10-20 διεθνή τοποθέτηση. Από αυτά κρατήθηκαν τα δεδομένα μόνο από τα 6. Για το τεστ χρησιμοποιήθηκε 10-fold cross-validation.

2.3.2 Μοντέλα βαθιάς Μάθησης (Deep Learning Models)

Οι Morabito κα στην εργασία τους [23] (9/2016) χρησιμοποιούν δεδομένα από και τις τρεις κατηγορίες, είχαν στη διάθεση τους 63 ασθενείς AD, 23 Healthy και 56 MCI, πάρα ταύτα χρησιμοποίησαν 23 δείγματα από την κάθε κατηγορία. Έγιναν δοκιμές και με τις τρεις κατηγορίες επιτυγχάνοντας ακρίβεια 82% αλλά και ανά δυο κατηγορίες πετυχαίνοντας 85% για MCI-HC και AD-HC, και 78% για την κατηγορία MCI-AD. Χρησιμοποίησαν 19 ηλεκτρόδια τοποθετημένα σύμφωνα με το διεθνές 10-20 σύστημα τοποθέτησης. Εφάρμοσαν φίλτρο band-pass. Επέλεξαν για μέθοδο την multi-channels deep convolutional neural networks (MC-DCNN). Η εκτίμηση των αποτελεσμάτων έγινε με k-fold Cross - validation. Οι Ieracitano κα στην έρευνά τους [24] (01/2019) είχαν 3 κατηγορίες (AD, υγιής, MCI) με 63 άτομα για δείγμα από την κάθε κατηγορία. Χρησιμοποίησαν 19 ηλεκτρόδια τοποθετημένα σύμφωνα με το διεθνές 10-20 σύστημα τοποθέτησης. Έκαναν ταξινόμηση και για τις τρεις κλάσεις αλλά πήραν και ζευγάρια των 2 και τα σύγκριναν και με άλλες κλασσικές μεθόδους. Στην ταξινόμηση και των τριών κλάσεων οι μέθοδοι είχαν ακρίβεια: CNN 83.33%, MLP 59.40%, SVM 56.84% και LDA 55.70%. Στην ταξινόμηση MCI-Healthy είχαν: CNN 91,88%, MLP 76,07%, SVM 71,37% και LDA 69,66%. Στην ταξινόμηση AD-Healthy είχαν: CNN 92,05%, MLP 83,33%, SVM 82,69% και LDA 80,13%. Στην ταξινόμηση MCI-AD είχαν: CNN 84,62%, MLP 66,88%, SVM 60,47% και LDA 64,53%. Έγινε ταξινόμηση και ανά ασθενή με αποτέλεσμα 40,27% για την τριάδα AD, Healthy και MCI, 62,5% στη δυάδα MCI-Healthy, 58,33% στη δυάδα AD-Healthy και 45,83% στη δυάδα MCI-AD.

Οι Bi κα [25] (06/2019) χρησιμοποίησαν επίσης και τις τρεις κατηγορίες έχοντας δείγματα μόνο από 4 άτομα από την κάθε κατηγορία. Χρησιμοποίησαν 64 ηλεκτρόδια. Προτείνουν το μοντέλο DCssCDBM που είχε 95,04% ακρίβεια η οποία ήταν η μεγαλύτερη. Έκαναν σύγκριση με πολλά άλλα μοντέλα: SVM (rbf) 0,7833%, DBN-3 84,2%, GDBM-2 85,66%, FitNet-10 91,93%, PCANet-2 92,44%, Highway-10 90,68%, VGG-A 92,89%, VGG-B 90,6%, VGG-C 88,89%, VGG-D 87,35%, GoogleNet 91,18%, ResNet-10 88,2%, NIN 87,96% και CDBN 86%. Τέλος οι Zeng κα στην εργασία τους [26] (2020) χρησιμοποιούν σαν δείγμα 17 AD, 35 υγιή και 35 MCI άτομα. Χρησιμοποίησαν ένα προσαρμοσμένο Νευρωνικό Δίκτυο βασισμένο σε γραφήματα AST-GCN (spatial temporal graph convolutional networks). Τα ηλεκτρόδια ήταν 64 από τα οποία χρησιμοποιήθηκαν τα 62. Το ποσοστό ακρίβειας για την τριάδα AD, Healthy και MCI ήταν 91,07%, για τα ζεύγη MCI-Healthy και AD-Healthy ήταν 94%. Οι Morteza κα [27] (04/2021) χρησιμοποίησαν από 3 κατηγορίες δείγματα, 64 AD, 64 υγιή και 64 MCI. 19 ηλεκτρόδια τοποθετήθηκαν σύμφωνα με το διεθνές 10-20 σύστημα τοποθέτησης. Το ποσοστό ακρίβειας για το CNN ήταν 82,3%, βρέθηκε να υπερτερεί έναντι των KNN 71,4%, SVM 41,1% και LDA 43,8%.

2.4 Σύνοψη

Αναφορά	Δείγμα	Μοντέλο	Αποτελέσματα
[14] Trambaiolli κα (07/2011)	16 AD 19 Healthy	SVM	80%
[15] Staudinger κα (12/2011)	79 AD 82 Healthy	Neural Network SVM	66% 78%
[16] Pogorelec (10/2012)	208	genTrees	86%
[17] Kulkarni κα (01/2017)	50 AD 50 Healthy	SVM	96%
[18] Al-nuaimi κα (07/2017)	20 AD 49 Healthy	KNN	85%
[19] Durongbhan κα (02/2019)	20 AD 10 Healthy	KNN	99%
[20] Tavarest κα (7/2019)	19 AD 16 Healthy	Logistic Regression SVM Random Forest Extra Tree SGD AdaBoost	96% 91% 47% 50% 97% 97%

		Gradient Boost	97%
[21] Safi κα (03/2021)	20 AD 35 Healthy 31 mild AD	KNN SVM RLDA	98% 96% 96%
[22] Puri κα (03/2022)	12 AD 11 Healthy	KNN SVM Random Forest MLDNN Naïve Bayes AdaBoost	96% 98% 94% 94% 84% 94%
[23] Morabito κα (9/2016)	63 AD 23 Healthy 56 MCI	CNN	AD, Healthy, MCI: 82% MCI-Healthy:85% AD-Healthy:85% MCI-AD:78%
[24] Ieracitano κα (01/2019)	63 Ad 63 Healthy 63 MCI	CNN MLP SVM LDA CNN MLP SVM LDA CNN MLP SVM LDA	AD, Healthy, MCI: 83% 59% 57 56 MCI-Healthy: 92% 76% 71% 70% AD-Healthy: 93% 84% 83% 80% MCI-AD:

		CNN MLP SVM LDA	85% 67% 58% 65%
[25] **Bi κα** **(06/2019)**	4 HD 4 Healthy 4 MCI	SVM (rbf) DBN-3 GDBM-2 FitNet-10 PCANet-2 Highway-10 VGG-A VGG-B VGG-C VGG-D GoogleNet ResNet-10 NIN CDBN DCssCDBM	0.78 0.842 0.8566 0.9193 0.9244 0.9068 0.9289 0.906 0.8889 0.8735 0.9118 0.882 0.8796 0.86 0.9504
[26] **Zeng κα** **(2020)**	17 AD 35 Healthy 35 MCI	AST-GCN	AD, Healthy, MCI: 91% MCI-Healthy:94% AD-Healthy:94%
[27] **Morteza κα** **(04/2021)**	64 AD 64 Healthy 64 MCI	KNN SVM LDA CNN	71% 41% 44% 82%

3 *Θεωρητικό υπόβαθρο*

Μια πρώτη αρχική προσέγγιση υπήρξε η μέθοδος ταξινόμησης μέσω 2D Convolutional Neural Networks. Έπειτα ακολούθησε η τεχνική Weight Average. Ουσιαστικά πρόκειται για μια συνδυαστική χρήση πολλών μοντέλων. Οι ταξινομητές που χρησιμοποιήθηκαν είναι οι: Logistic Regression, Decision Tree, Random Forest, Naïve Bayes, Support Vector Machines (SVM), K Nearest Neighbors (KNN), AdaBoost και Quadratic Discriminant Analysis (QDA). Ακολουθεί περιγραφή των παραπάνω Classifier.

3.1 Μηχανική Μάθηση (Machine Learning)

Μηχανική Μάθηση είναι η συλλογή αλγορίθμων και μεθόδων με τις οποίες βελτιώνεται η αποδοτικότητα μιας μηχανής (πχ. υπολογιστή) στην εκτέλεση «ευφυών» εργασιών. Μπορεί να βοηθήσει στη λήψη ορθότερων αποφάσεων σε επιχειρήσεις, οργανισμούς.

3.1.1 Τύποι Αλγορίθμων Μηχανικής Μάθησης

- Εποπτευόμενη Μάθηση (Supervised Learning). Οι αλγόριθμοι εδώ μαθαίνουν από παραδείγματα δεδομένων ή σχετικές αποκρίσεις με ετικέτες, και αργότερα γίνεται πρόβλεψη με τη χρήση νέων παραδειγμάτων.
- Μη εποπτευόμενη μάθηση (Unsupervised Learning). Οι αλγόριθμοι εδώ μαθαίνουν από παραδείγματα χωρίς όμως αυτή τη φορά να υπάρχει κάποια σχετική απόκριση. Ο αλγόριθμος βρίσκει από μόνος του και κατηγοριοποιεί τα δεδομένα.
- Μάθηση με ενίσχυση (Reinforcement Learning). Οι αλγόριθμοι εκπαιδεύονται συνεχώς. Μονίμως γίνονται δοκιμές και σφάλματα. Αποκτά έτσι εμπειρία, μαθαίνει από τα λάθη του και προσπαθεί στο μέλλον να πάρει τις σωστές αποφάσεις

3.1.2 Βασικές κατηγορίες Εργασιών

- Αναγνώριση προτύπων (Pattern Recognition). Είναι η διαδικασία κατά την οποία μέσω αλγορίθμων επιτυγχάνεται η αναγνώριση των προτύπων. Σύμφωνα με τη γνώση που έχει αποκτήσει μπορεί και κατηγοριοποιεί τα δεδομένα που θα του δοθούν δίνοντας συγχρόνως και την πιθανότητα το συγκεκριμένο πρότυπο να ανήκει στη συγκεκριμένη κλάση. Καθώς τα δεδομένα που του δίνονται μπορεί να είναι σε διαφορετική μορφή από αυτήν που κατανοεί ένα υπολογιστικό σύστημα, π.χ. βίντεο, εικόνες, φωνή, κάνει

τη μετατροπή σε μορφή κατανοητή από τους υπολογιστές και έπειτα γίνεται η επεξεργασία.

- Εξαγωγή χαρακτηριστικών (Feature Extraction). Πολλές φορές τα δεδομένα που εισέρχονται για ανάλυση στο σύστημα είναι πολύ μεγάλα. Υπάρχουν φορές που μπορούμε να πάρουμε τα αποτελέσματα που θέλουμε χωρίς να χρειάζεται να πάρουμε όλη την πληροφορία. Για παράδειγμα εάν θέλουμε να επεξεργαστούμε εικόνα μπορούμε να τη συμπιέσουμε και να την επεξεργαστούμε χωρίς να χάσουμε ουσιαστική πληροφορία. Έτσι μικραίνουμε τον όγκο των δεδομένων, σε μικρά σύνολα δεδομένων μπορεί να μην είναι ορατά τα θετικά αποτελέσματα αλλά σε μεγάλα σύνολα δεδομένων ο χρόνος που χρειάζεται για να γίνει η επεξεργασία τους ελαττώνεται κατά πολύ.
- Πρόβλεψη (Prediction) και Παλινδρόμηση (Regression). Μία από τις πιο κοινές τεχνικές πρόβλεψης στη μηχανική μάθηση είναι η τεχνική της Παλινδρόμησης. Με τη χρήση αλγορίθμων της μηχανικής μάθησης. Μπορούμε για παράδειγμα κάποιες αλλαγές τιμών στην ατμόσφαιρα να προβλέψουμε ένα φαινόμενο που πιθανόν θα συμβεί.
- Ομαδοποίηση (Clustering). Η ομαδοποίηση είναι η διαδικασία κατά την οποία τα δεδομένα χωρίζονται σε ομάδες βάση της ομοιότητας τους ή κοινά σημεία που παρουσιάζουν. Για παράδειγμα μπορούμε να δώσουμε φωτογραφίες από γάτες και σκύλους και να βάλει τις εικόνες σε 2 κλάσεις (ομάδες). Η μια με τους σκύλους και η άλλη με τις γάτες. Όταν θα του δώσουμε μετά μια εικόνα με γάτα θα μπορέσει βλέποντας τα χαρακτηρίστηκα της να βρει σε ποια από τις δυο κλάσεις ανήκει.
- Γενίκευση (Generalization). Αφού εκπαιδεύσουμε ένα μοντέλο, αυτό μετά θα είναι σε θέση να γενικεύσει. Δηλαδή εφόσον έχουμε εκπαιδεύσει ένα μοντέλο πρέπει αυτό μετά να είναι σε θέση σε περίπτωση που του δώσουμε ένα νέο δεδομένο να μπορέσει να το βάλει στην κατάλληλη κλάση. Το δεδομένο που θα του δώσουμε δε θα έχει συμμετέχει στη διαδικασία της εκπαίδευσης. Για να έχει πετύχει σε επίπεδο γενίκευσης το μοντέλο μας θα πρέπει στην εκπαίδευση τα δεδομένα που θα του δοθούν να είναι ποικίλα. Δηλαδή αν για παράδειγμα είχαμε δυο κλάσεις μια με σκύλους και μια με λύκους, αν βάζαμε φωτογραφίες από σκύλους μόνο μιας συγκεκριμένης ράτσας τότε θα ήταν δύσκολο να γενικεύσουμε γιατί δίνοντάς μετά μια φωτογραφία από σκύλο άλλης ράτσας μπορεί να μην πετυχαίναμε τόσο καλά αποτελέσματα. Αν π.χ. στην κλάση με τους σκύλους δίναμε φωτογραφίες μόνο από τεριέ (εικόνα 2) και δεν δίναμε στην κλάση φωτογραφίες άλλης ράτσας σκυλιών τότε όταν θα πηγαίναμε μετά στη φάση της εκπαίδευσης να δώσουμε μια φωτογραφία με Σιβηρικό χάσκι (εικόνα 3), το πιθανότερο είναι να το κατατάξει στην κλάση με τους λύκους.

Figure 2. Τεριέ [5]

Figure 3. Σιβηριανό χάσκι [6]

3.2 Weight Average

Η μέθοδος Weight Average χρησιμοποιήθηκε ουσιαστικά για το συνδυασμό των ταξινομητών που περιγράφονται παρακάτω. Μπορεί να χρησιμοποιηθεί είτε για να ελέγξουμε τα αποτελέσματα κάποιον μεθόδων μέχρι να καταλήξουμε σε ποια ή για το ποιες θα χρησιμοποιήσουμε όπως έγινε και στη δική μας περίπτωση ή για να αποφέρει καλύτερα αποτελέσματα αφού μπορεί να συνδυάσει τα αποτελέσματα από διαφορετικές μεθόδους βελτιστοποιώντας τα αποτελέσματα τους[28].

3.3 Logistic Regression Classifier

Αυτό το είδος ταξινομητή χρησιμοποιεί Λογιστική Παλινδρόμηση, κάνει χρήση της θεωρίας των πιθανοτήτων. Είναι εποπτευόμενη μέθοδο μάθησης. Ουσιαστικά η μέθοδος αυτή γενικεύει τα γραμμικά μοντέλα, έτσι ώστε η εξαρτημένη μεταβλητή να ακολουθεί την εκθετική οικογένεια κατανομών. Συσχετίζει μια εξαρτώμενη μεταβλητή με μια η περισσότερες ανεξάρτητες. Αν πρόκειται για διακριτή μεταβλητή εκχωρούνται τα δεδομένα αλλιώς αν η μεταβλητής είναι συνεχής ακολουθείται η διαδικασία της παλινδρόμησης. Σύμφωνα με τα δεδομένα που έχει δεχθεί αποφασίζει ποια είναι η καλύτερη συνάρτηση που πρέπει να χρησιμοποιηθεί και δίνει σαν αποτέλεσμα ένα μοντέλο. Το μοντέλο αυτό χρησιμοποιείται αργότερα κατά τη διαδικασία της πρόβλεψης για την κατηγοριοποίηση των δεδομένων. Πρόκειται για μια στατιστική τεχνική μοντελοποίησης.

3.4 Decision Tree Classifier

Πρόκειται για εποπτευόμενη μέθοδο μάθησης. Ο συγκεκριμένος ταξινομητής προσπαθεί να βρει τα καλύτερα χαρακτηριστικά του κάθε δεδομένου για τη βέλτιστη ταξινόμηση. Σε κάθε κόμβο δημιουργούνται υποδέντρα και ο διαχωρισμός γίνεται σύμφωνα με τα χαρακτηριστικά - ιδιότητες που διακρίνουν τα δεδομένα. Ανακαλύπτουν ποια είναι τα καλύτερά χαρακτηριστικά έτσι ώστε να πάει σε καλύτερο διαχωρισμό υποδέντρου. Αυτό συνεχίζεται μέχρι να φτάσουμε σε κόμβο φύλλο, δηλαδή να μην υπάρχει περίπτωση να υπάρχει κάποιο

χαρακτηριστικό για τη δημιουργία νέου υποδέντρου ή να έχει ξεπεραστεί το βάθος ανάλυσης που θέλουμε. Το βάθος ανάλυσης μπορούμε να το παραμετροποιήσουμε εμείς όπως επιθυμούμε κατά το κτίσιμο του μοντέλου μας. Έπειτα όταν έρχεται καινούργιο δεδομένα για ταξινόμηση ακολουθεί μια διαδρομή στο δέντρο σύμφωνα με τα χαρακτηριστικά που το διακρίνει και ταξινομείται στην κατάλληλη κλάση [29], [30].

3.5 Random Forest

Βασίζεται στους ταξινομητές που είναι τύπου δέντρα και SVM. Ουσιαστικά χρησιμοποιεί τη μέθοδο «Διαίρει και Βασίλευε». Αντλεί τυχαία χαρακτηριστικά από δείγματα που λαμβάνει κατά τη διαδικασία της εκπαίδευσης και τα χρησιμοποιεί για τη δημιουργία υποδέντρου τύπου Δέντρου Απόφασης. Το γεγονός ότι επιλέγονται τυχαία τα χαρακτηριστικά δίνει τη δυνατότητα ενός μεγάλου εύρους ποικιλόμορφων υποδέντρων. Ένα θετικό είναι ότι δεν χρειάζονται να προϋπάρχουν ετικέτες για το διαχωρισμό των κλάσεων. Εφόσον ότι διαχωρισμός γίνεται έχει να κάνει με την τυχαία επιλογή χαρακτηριστικών αναλαμβάνει η μέθοδος και το διαχωρισμό. Στην πορεία μπορεί να αποδειχτεί ότι κάποια από αυτά τα δέντρα απόφασης έδωσαν λανθασμένα αποτελέσματα [31]–[33].

3.6 Gaussian Naive Bayes

Πρόκειται για εποπτευόμενη μέθοδο μάθησης. Είναι ένας ταξινομητής που κάνει χρήση πιθανοτήτων. Λαμβάνει κάθε χαρακτηριστικό ως μια ξεχωριστή μεταβλητή που λειτουργεί ως ανεξάρτητη. Προτέρημα της αποτελεί οτι χρειάζεται λίγα από τα κύρια χαρακτηριστικά για να μπορεί να ταξινομήσει. Όταν τα δεδομένα μας έχουν συνεχόμενες τιμές μπορούμε λαμβάνοντας υπόψιν τη μέση και τυπική απόκλιση να τα κατηγοριοποιήσουμε σε κλάσεις [34].

3.7 Support Vector Machines - SVM

Πρόκειται για εποπτευόμενη μέθοδο μάθησης. Ο διαχωρισμός των δεδομένων σε κλάσεις γίνεται με βάση τα χαρακτηριστικά τους. Ορίζεται ένα όριο με βάση το οποίο θα γίνει ο διαχωρισμός ανάμεσα στις κλάσεις. Αν οπτικοποιήσει κανείς αυτό το όριο τότε θα δούμε μια ευθεία γραμμή ανάμεσα στα δεδομένα. Όσο πιο μακριά στη γραμμή είναι ένα δείγμα τόσο αυξάνεται η πιθανότητα να έχει ταξινομηθεί σωστά. Παράλληλα με αυτή τη γραμμή υπάρχουν και 2 άλλες. Αυτές αποτελούν τα "περιθώρια'. Τα δείγματα που είναι μέσα στα περιθώρια είναι αυτά που μπορεί και να έχουν ταξινομηθεί λάθος καθώς δεν είναι πολύ διακριτή η διαφοροποίηση των χαρακτηριστικών τους. Αν αφαιρέσει κανείς τα δείγματα που βρίσκονται στα περιθώρια τότε τα υπόλοιπα δεδομένα είναι πολύ πιο ξεκάθαρο σε ποια κλάση ανήκουν. Με αυτόν τον τρόπο δίνεται ένα περιθώριο καλύτερου διαχωρισμού ανάμεσα στις κλάσεις με

αποτέλεσμα να επιτυγχάνονται μεγαλύτερα ποσοστά επιτυχίας ως προς την κατηγοριοποίηση - ταξινόμηση ενός δείγματος [35].

3.8 K Nearest Neighbors - KNN

Πρόκειται για εποπτευόμενη ταξινόμηση. Αποτελεί ένα από τα πιο απλά είδη ταξινόμησης. Η ταξινόμηση αυτή βασίζεται στην ομοιότητα των δειγμάτων μεταξύ τους. Αυτός είναι και ο λόγος που την καθιστά μια από τις καλύτερες μεθόδους όσο αφορά την ταξινόμηση αντικειμένων. Κατά τη διάρκεια της εκπαίδευσης ο αλγόριθμός μαθαίνει ποια είναι τα όμοια χαρακτηριστικά που έχουν τα δεδομένα που ανήκουν στην ίδια κλάση. Εξετάζει τους γείτονες ενός δείγματος και το ταξινομεί στην κλάση που πλειοψηφεί. Δεν χρησιμοποιούνται πιο περίπλοκοι μηχανισμοί ταξινόμησης, απλά και μόνο με την ομοιότητα των δειγμάτων γίνεται η κατηγοριοποίηση των εισερχομένων δεδομένων [36].

3.9 AdaBoost

Πρόκειται για μια μέθοδο βελτίωσης αδυνάτων αλγορίθμων μηχανικής μάθησης. Ανήκει στην κατηγορία των εποπτευόμενων μεθόδων μάθησης. Στην εργασία τους [37] οι G. Wang, Xu, H. Wang και Zou παρουσιάζουν μια παραλλαγή στη μεθόδου η οποία συμβάλει στην ακόμη μεγαλύτερη βελτιστοποίηση των αποτελεσμάτων σε συνδυασμό με την SVM. Την αξιολόγηση των αποτελεσμάτων τη βασίσουμε σε κάθε μέθοδο σύμφωνα με κάποιο κριτήριο πάνω στα αποτελέσματα και κατά τη διάρκεια της εκμάθησης και κατά τη διάρκεια της εκπαίδευσης. Εδώ η αξιολόγηση των αποτελεσμάτων γίνεται με κριτήριο το μέσο τετραγωνικό σφάλμα (MSE - mean squared error) και στο μέσο απόλυτο σφάλμα (MAE - mean absolute). Σύμφωνα με τις τιμές που παρουσιάζονται σε αυτά τα δυο κριτήρια γίνεται προσπάθεια βελτίωσης των αποτελεσμάτων.

3.10 Quadratic Discriminant Analysis - QDA

Είναι αρκετά παρόμοια με την ανάλυση γραμμικής διάκρισης (LDA), εκτός από το ότι υπολογίζουμε ξεχωριστά το μέσο όρο και τη συνδιακύμανση. Πρόκειται για αλγόριθμό χωρίς περίπλοκους υπολογισμούς, χωρίς να χρήζει τη χρήση πολλών παραμέτρων και μπορεί να χρησιμοποιηθεί για ταξινόμηση πολλών κλάσεων. Ανήκει στην κατηγορία των εποπτευόμενων μεθόδων μάθησης. Ένα πρόσθετο πλεονέκτημα θα ήταν να υπάρχει ήδη γνώση για το εάν κάποιες τάξης εμφανίζουν διακριτές συνδιακυμάνσεις. Επειδή όμως παρέχει μεγαλύτερη ευελιξία στον πίνακα συνδιακύμανσης απ' ότι η LDA, εφόσον χρησιμοποιεί ξεχωριστό πίνακα για κάθε τάξη, ναι μεν ταιριάζει καλύτερα τα δεδομένα αλλά έχει περισσότερες παραμέτρους. Θα πρέπει κάποιος να τη χρησιμοποιεί με προσοχή όταν τα χαρακτηριστικά είναι πάρα πολλά

και έχει επιπλέον το μειονέκτημα οτι δε μπορεί να κάνει μείωση διαστάσεων. Και επιπροσθέτως θα υπάρχει πρόβλημα αν υπάρχουν πολλές τάξεις χωρίς να υπάρχουν πολλά δείγματα [38]–[40].

3.11Συνελικτικά Νευρωνικά Δίκτυα - Convolutional Neural Networks 2D

Είναι σύνηθες να χρησιμοποιούνται στα πειράματα που άπτονται ιατρικών θεμάτων [41] τα Convolutional Networks καθώς έχουν καλή απόδοση όταν σαν εισαγωγή δίνονται εικόνες. Είτε πρόκειται να δοθεί σαν είσοδος ολόκληρη η εικόνα πχ ακτινογραφία, μαγνητική κτλ. είτε πρόκειται για τμήματα που αποκτούνται από την τμηματοποίηση μιας εξέτασης όπως για παράδειγμα τα EEG που χρησιμοποιήθηκαν και στην παρούσα εργασία. Τα πιο συχνά χρησιμοποιούμενα είναι τα 2D που χρησιμοποιούνται κυρίως όταν έχουμε να κάνουμε με εικόνες. Ένα παράδειγμα του τρόπου που λειτουργούν τα CNN αποτυπώνεται στην εικόνα 4.

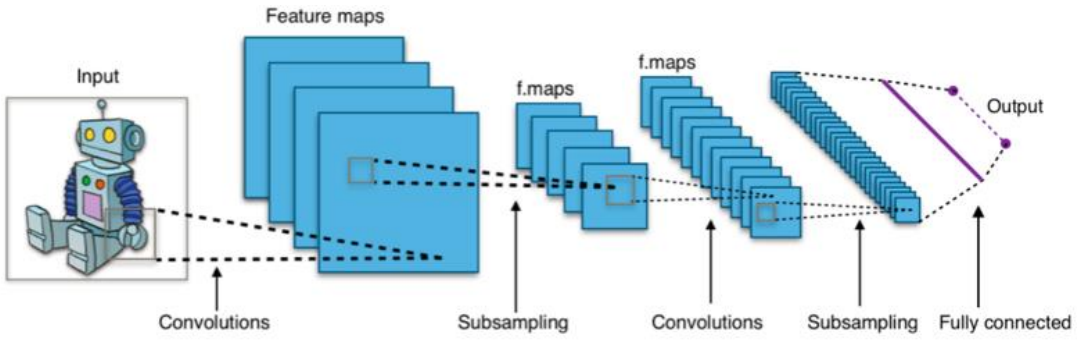

Figure 4. Τυπικό CNN [42]

Ο τρόπος με τον οποίο μια εικόνα επεξεργάζεται μέσα σε ένα στρώμα απεικονίζεται στην εικόνα 5[1].

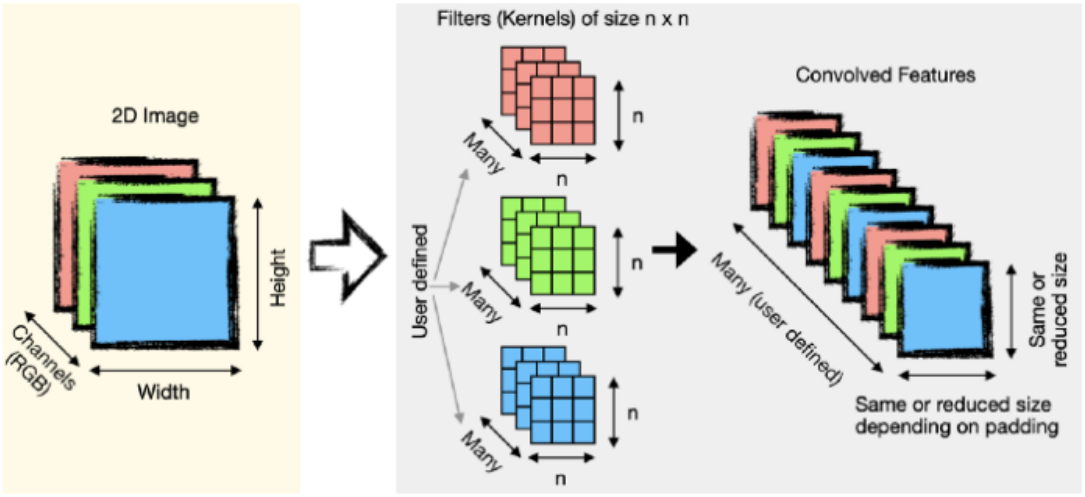

Figure 5. Πέρασμα της εικόνας μέσα στα στρώματα των συνελικτικών δικτύων [43]

Η παραπάνω εικόνα αναφέρεται σε έγχρωμη εικόνα. Γι' αυτό και βλέπουμε τα τρία βασικά χρώματα, αν ήταν ασπρόμαυρη τότε θα είχαμε ένα μόνο κανάλι. Αναλόγως με τις παραμετροποιήσεις που θα δώσουμε γίνεται και η κατάλληλη επεξεργασία στις εικόνες. Για παράδειγμα στις εικόνες 5 και 6 έχουμε μια εικόνα μεγέθους 9x9 η οποία μετατρέπεται σε 3x3. Το 3x3 έχει να κάνει με την παράμετρο filer, ενώ η τελική απεικόνιση με το τι αριθμούς θα κρατήσει με το αν θα επιλέξουμε max (εικόνα 6) ή average (εικόνα 7) pooling. Στην παρούσα εργασία δοκιμάστηκε και το max και το average pooling με το δεύτερο να φαίνεται να δίνει καλύτερα αποτελέσματα. Οι υπόλοιποι παράμετροι εξηγούνται στο κεφάλαιο 5 στην παράγραφο 10. Η παραμετροποίηση έχει να κάνει πάντα και με τον τύπο των δεδομένων εισόδου που δέχεται η μέθοδος. Μπορεί μια τεχνική που ανεβάζει το ποσοστό επιτυχίας σε κάποια δεδομένα σε άλλα δεδομένα να το κατεβάζει.

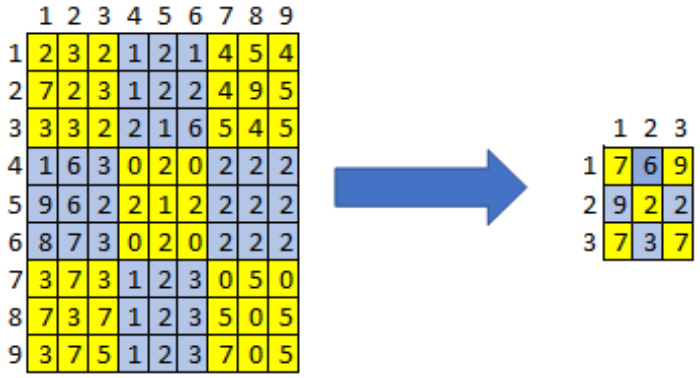

Figure 6. Μετατροπή εικόνας 9x9 σε 3x3 με max pooling

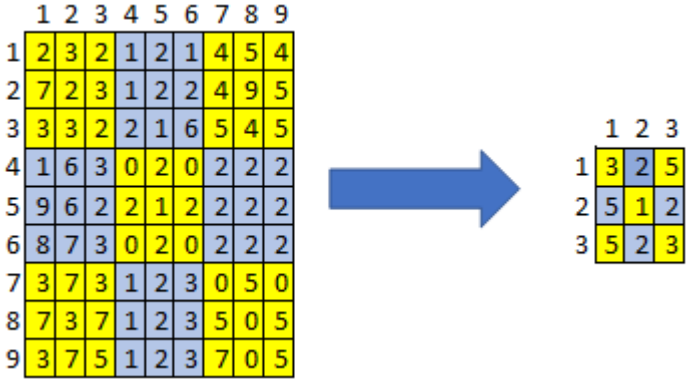

Figure 7. Μετατροπή εικόνας 9x9 σε 3x3 με average pooling

4 Προεπεξεργασία των Δεδομένων

Εδώ θα ακολουθήσει η ανάλυση του προβλήματος που διαπραγματεύεται η διπλωματική, δηλαδή τα δεδομένα που χρησιμοποιήθηκαν και η επεξεργασία που υπέστησαν πριν από την είσοδο αυτών σε κάποιο μοντέλο, καθώς το υλικό που χρησιμοποιήθηκε.

4.1 Υλικό και Λογισμικό

Για την επεξεργασία των δεδομένων καθώς και για τα πειράματα χρησιμοποιήθηκε το διαδικτυακό εργαλείο Colab. Πρόκειται για ένα online εργαλείο της Google που δίνει τη δυνατότητα στους χρήστες να μπορούν να γράφουν και να εκτελούν κώδικα σε python στον περιηγητή τους. Δε χρειάζεται οποιαδήποτε εγκατάσταση και κάνει εφικτό το διαμοιρασμό του κώδικα. Στην παρούσα εργασία επιλέχθηκε Colab Pro που κάνει χρήση της Python 3.6.9, το απλό Colab δεν ήταν σε θέση να εκτελέσει τα πειράματα από μόνο του, καθώς ο όγκος των δεδομένων ειδικά στην περίπτωση των Convolutional Networks ήταν πολύ μεγάλος και δεν επαρκούσαν οι πόροι για να εκτελεστούν τα πειράματα. Επίσης σε αρχικό στάδιο έγινε χρήση του Jupiter Notepad για την εξαγωγή των EGG σε πίνακες αλλά και εκεί αρκετές φορές δεν επαρκούσε η μνήμη του υπολογιστή για να εκτελεστούν οι εργασίες που έπρεπε.

Ο επεξεργαστής του υπολογιστή που έγιναν όλες οι εργασίες της διπλωματικής είναι Intel(R) Core(TM) i5-4310U CPU @ 2.00GHz 2.60 GHz. Επειδή η μνήμη του υπολογιστή δεν επαρκούσε αγοράστηκε χώρος στο google drive και τα δεδομένα αντλούνταν και αποθηκεύονταν σε αυτό το χώρο.

4.2 Δεδομένα

Τα δεδομένα μας τα παρείχε η «Ελληνική Ένωση Νόσου Αλτσχάιμερ και Συναφών Διαταραχών». Μας παραδόθηκαν αρχεία σε μορφή EDF (European Data Format). Χρησιμοποιείται στην ιατρική για χρονοσειρές όπως τα EEG.

4.2.1 Μορφή Δεδομένων

Μας παρείχαν δεδομένα από 54 ασθενείς. Τα δείγματα ήταν 3 κατηγοριών: υγιείς (Healthy), στο στάδιο της Ήπιας Νοητικής Διαταραχής (Mild Cognitive Impairment - MCI) και ασθενείς Αλτσχάιμερ (Alzheimer's). Τα δείγματα ήταν κατανεμημένα ισόποσα σε 18 ασθενείς από την κάθε κατηγορία. Σε κάθε αρχείο συμπεριλαμβανόταν πληροφορίες με τη μορφή artifact για το εάν τα μάτια ήταν ανοιχτά, κλειστά κτλ. Κάθε αρχείο ήταν μια εξέταση η οποία διαρκούσε περίπου 15 λεπτά. Από αυτά κρατήθηκαν μόνο 5 λεπτά κατά τη διάρκεια των οποίων ο ασθενής είχε τα μάτια κλειστά. Κάθε δευτερόλεπτο αντιστοιχούσε σε 500 στιγμιότυπα. Για τα πειράματα διαχωρίσαμε τα δεδομένα σε τμήματα των 2 δευτερολέπτων (1000 στιγμιότυπα) στις περιπτώσεις των πειραμάτων που αφορούσαν απλές μεθόδους ενώ για την περάτωση των πειραμάτων μέσω Convolution Networks έγιναν τμήματα των 1000, 1500, 2500 και 5000 στιγμιοτύπων, στα τελικά πειράματα την καλύτερη απόδοση έδωσαν αυτά των 5 δευτερολέπτων (2500 στιγμιοτύπων). Στις περιπτώσεις των 1000 στιγμιοτύπων είχαμε 150 τμήματα από κάθε ασθενή δεδομένου του ότι τα 5 λεπτά είναι 300 δευτερόλεπτα x 500 στιγμιότυπα, 150000 στιγμιότυπα σύνολο από τον κάθε ασθενή. Στις περιπτώσεις των 2500 στιγμιοτύπων είχαμε 60 τμήματα από τον κάθε ασθενή.

Στα αρχεία που δόθηκαν φαίνονται 21 ηλεκτρόδια, από αυτά έγινε χρήση των μόνο των 19 (Fp1, Fp2, F7, F3, Fz, F4, F8, T3, C3, Cz , C4, T4, T5, P3, Pz, P4, T6, O1 και O2) στα 500Hz..

Για την προεπεξεργασία των δεδομένων έγινε χρήση δυο βιβλιοθηκών: NumPy [44], [45] και MNE [46]. Η NumPy (Numeric Pyrhon), αποτελεί λογισμικό ανοιχτού κώδικα και χρησιμοποιείται για την περάτωση μαθηματικών συναρτήσεων. Μία από αυτές τις συναρτήσεις που έγινε χρήση στην παρούσα εργασία είναι η Fourier. Στην Python χρησιμοποιείται κυρίως για το χειρισμό πινάκων πολλών διαστάσεων. Η MNE αποτελεί επίσης ένα πακέτο ανοιχτού κώδικα [47]–[52]. Χρησιμοποιείται για την επεξεργασία των EEG παρέχοντας μεθόδους προεπεξεργασίας, όπως τα φίλτρα low-band pass και high pass που χρησιμοποιήθηκαν στην παρούσα εργασία, στατιστικές αναλύσεις και άλλες πληροφορίες για περιοχές του εγκεφάλου. Συνεργάζεται με άλλες βιβλιοθήκες για διάφορες άλλες εργασίες, όπως υπολογισμούς (π.χ. NumPy), οπτικοποίηση (π.χ. matplotlib) και νευροαπεικονίσεις (Nibabel).

4.2.2 Outliers

Εξ αιτίας της φύσης της εξέτασης ο θόρυβος σε τέτοιου είδους εξετάσεις είναι κάτι αναπόφευκτο. Σε πολλά σημεία του EEG το σήμα ξέφευγε αρκετά, είτε για ένα κανάλι είτε για πολλά κανάλια. Αυτό μπορεί να συμβαίνει είτε γιατί υπήρξε ένας ήχος την ώρα την εξέτασης που τράβηξε την προσοχή του ασθενή, είτε κάποια κίνηση, είτε οτιδήποτε άλλο που του προκάλεσε κάποια ενόχληση και σαν αποτέλεσμα είχε την αλλαγή των δεδομένων που καταγράφηκαν κατά την εξέταση (εικόνα 8). Λόγω του ότι πρόκειται για ιατρικά δεδομένα και

η παραμικρή παρέμβασή μας θα μπορούσε να αλλοιώσει το αποτέλεσμα των εξετάσεων θα πρέπει να είναι κανείς πολύ προσεκτικός και να εξετάζει ενδελεχώς τις κινήσεις που κάνει για να «καθαρίσει» το σήμα με οποιοδήποτε τρόπο αυτός επιθυμεί. Παρατηρήθηκε το φαινόμενο ότι ειδικά τα κανάλια Fp1 και Fp2 (εικόνα 9) αρκετές φορές σε κάποια σημεία τα σήματα είχαν ξεφύγει αρκετά από τα όρια τόσο που δεν φαινόταν στο EEG. Αυτό μπορεί να συμβαίνει γιατί τα Fp1 και Fp2 είναι κοντά στην περιοχή των ματιών και ένα βλεφάρισμα και μόνο των ματιών είναι αρκετό για να βγάλει τις μετρήσεις εκτός ορίων.

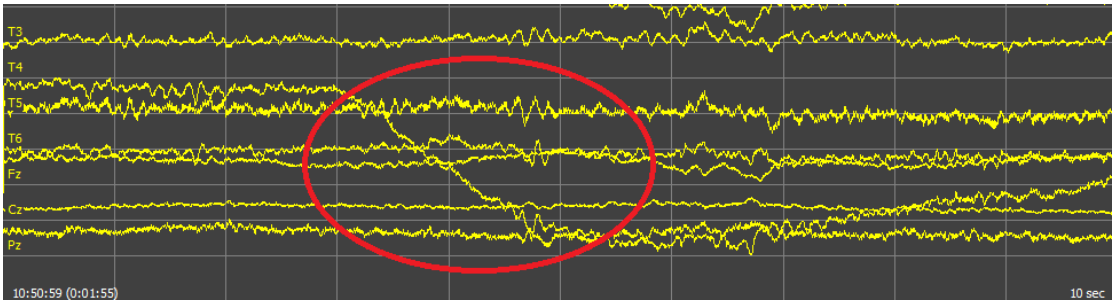

Figure 8. Στο T4 υπάρχει μεγάλη αλλαγή ξαφνικά η οποία επανέρχεται σε δευτερόλεπτα.

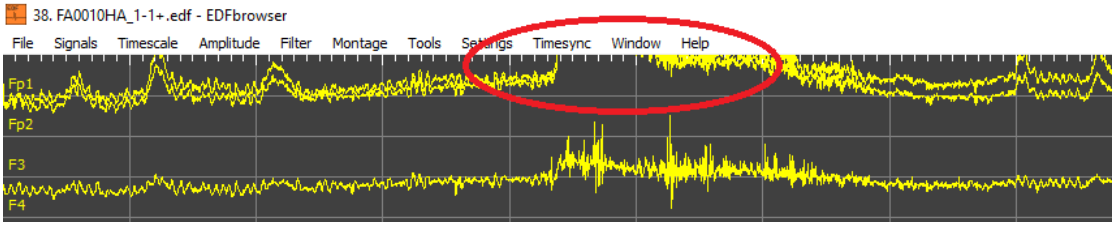

Figure 9. Στο F1&F2 υπάρχει διακοπή σήματος.

Ένας τρόπος για να καθαριστεί το σήμα είναι να υπολογιστεί μέση τιμή για κάθε σήμα ξεχωριστά και παίρνοντας την απόσταση του κάθε στιγμιότυπου από τη μέση απόσταση να αποκλείονται αυτά που έχουν απόσταση πάνω και κάτω από ένα όριο. Αυτό όμως μειώνει σημαντικά την ποσότητα των δεδομένων που θα παραμείνουν τελικά μετά τον καθαρισμό του σήματος. Επίσης επειδή στην παρούσα εργασία πρόκειται για ιατρικά δεδομένα το να ξεφύγει ένα σήμα από το όριο μπορεί να είναι και το χαρακτηριστικό που θα το κατατάξει σε άλλη κατηγορία.

Από την άλλη πάλι στις περιπτώσεις που τα EEG μετατράπηκαν σε γραφικές παραστάσεις και για να υπάρχει ομοιομορφία πάρθηκε η ανώτερη και κατώτερη τιμή σε κάθε σήμα (η κάθε γραφική παράσταση αφορούσε ένα σήμα) και δόθηκαν ως τα όρια του άξονα x, αποτέλεσμα αυτού ήταν τα EGG με φυσιολογικές τιμές να είναι σχεδόν σαν ευθεία γραμμή, μη δίνοντας τη δυνατότητα να ξεχωρίζουν ουσιαστικά το ένα από το άλλο. Από την άλλη όταν δεν δινόταν τιμές ορίων στον άξονα x το αποτέλεσμα ήταν γραφικές παραστάσεις που έμοιαζαν αλλά στην ουσία ήταν πολύ διαφορετικές οι τιμές τους (εικόνες 10-11). Τα καλύτερα αποτελέσματα δόθηκαν στην δεύτερη περίπτωση που δεν καθορίστηκαν τιμές για τους άξονες x και y.

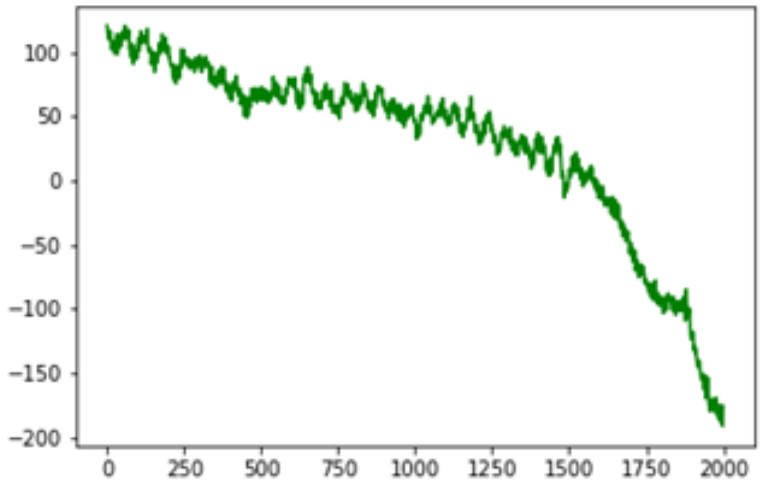

Figure 10. Γραφική παράσταση 4 δευτερολέπτων με ανώτατο άκρο περίπου 100 και κατώτατο περίπου -200.

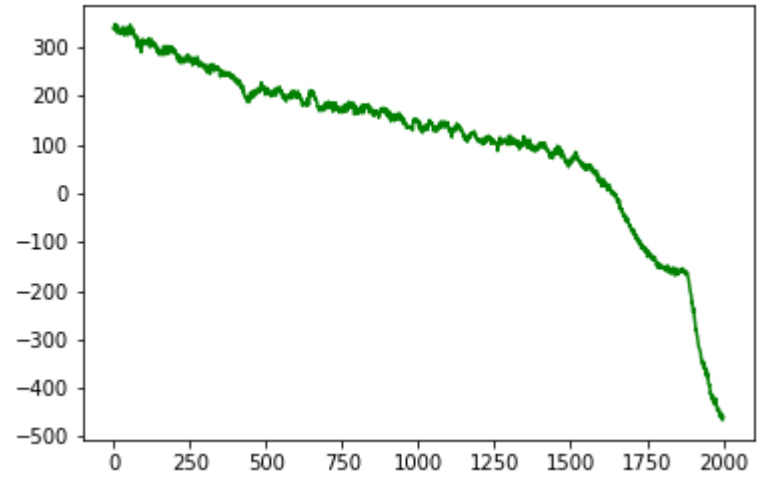

Figure 11. Γραφική παράσταση 4 δευτερολέπτων με ανώτατο άκρο περίπου 300 και κατώτατο περίπου -500.

4.3 Φιλτράρισμα

4.3.1 Band Pass Filters

Όπως εξηγήθηκε και στην παράγραφο 2.1.1 το συγκεκριμένο φίλτρο δέχεται τιμές και αν το σήμα ξεπεράσει το ανώτερο ή κατώτερο όριο τότε αφαιρείται γιατί θεωρείται ότι πρόκειται για θόρυβο. Στη συγκεκριμένη εργασία όπως φαίνεται και στον πίνακα αποτελεσμάτων στο κεφάλαιο 6 όταν γινόταν χρήση φίλτρων το αποτέλεσμα των μετρήσεων ήταν χαμηλότερο σε σχέση με αυτό που είχαμε αν τα αφαιρούσαμε. Δεν υπήρχε λόγος να χρησιμοποιηθεί Bass-stop Filter γιατί έτσι θα χανόταν ή πληροφορία που έπρεπε να υπάρχει για τη διεκπεραίωση της εργασίας.

4.4 Εξαγωγή Χαρακτηριστικών

4.4.1 Fourier Transform

Στη παρούσα εργασία εφαρμόστηκε FFT σε κάθε ένα κανάλι. Το κάθε κανάλι του κάθε ασθενή κόπηκε σε 150 μέρη αποτελούμενο από 1000 σημεία που αντιστοιχούν σε 2 δευτερόλεπτα εξέταση. Στους πίνακες που δεν εφαρμόστηκε φάνηκε τα τελικά αποτελέσματα να είναι πιο χαμηλά.

4.4.2 PCA

Είναι η μέθοδος που βοήθησε να έχουμε κατά πολύ καλύτερη απόδοση σε κάθε μοντέλο των πειραμάτων. Το ποσοστό επιτυχίας του κάθε μοντέλου με τη χρήση της PCA αυξήθηκε ακόμη και αν δεν υπήρξε κάποια άλλη μέθοδο από τις υπόλοιπες που έχουν αναφερθεί. Έπαιξε καθοριστικό ρόλο στην καλυτέρευση αυτών. Η κύρια παράμετρος είναι η n_components η οποία παίρνει έναν ακέραιο αριθμό και αφορά το πόσα στοιχεία επιθυμούμε να έχουμε μετά την εφαρμογή της PCA. Για παράδειγμα αν έχουμε ένα δισδιάστατο πίνακα με 190 στήλες και στην παράμετρο n_components βάλουμε τον αριθμό 19 τότε μετά την εφαρμογή της PCA ο νέος πίνακας που θα δημιουργηθεί θα έχει 19 στήλες. Αν δεν δώσουμε κάποιον αριθμό τότε θα παραμείνουν όλα τα στοιχεία. Πάνω σε αυτή την παράμετρο πειραματιστήκαμε με την καλύτερη απόδοση, τις περισσότερες φορές, να έχει το n_components=19, δοκιμάστηκαν επίσης και το 190 και 1900. Το 19 φάνηκε να αυξάνει τα αποτελέσματα. Πάρα ταύτα ακόμη και το n_components=1900 έδωσε κατά πολύ καλύτερα αποτελέσματα από τη μη χρήση της PCA.

4.4.3 StandardScaler

Στη μέθοδο που χρησιμοποιήθηκε και ήταν σαφές ότι θα πρέπει να γίνει χρήση της ήταν η SVM. Στην περίπτωση ανά segment αύξησε το τελικό ποσοστό ενώ στις περιπτώσεις ανά ασθενή αν δεν γινόταν χρήση της είχαμε μηδενικά αποτελέσματα. Έγινε εφαρμογή της μεθόδου στην KNN χωρίς να υπάρχουν σαφή αποτελέσματα για το εάν η εφαρμογή της βοηθάει ή όχι την απόδοση της μεθόδου. Στην περίπτωση ανά segment έδωσε μεγαλύτερο ποσοστό αλλά στις περιπτώσεις των ασθενών σε κάποιους φάνηκε να αυξάνει το ποσοστό και σε κάποιους να το μειώνει με το τελικό αποτέλεσμα ανά ασθενή να δίνει καλύτερο αποτέλεσμα η μη χρήσης της. Το ίδιο αποτέλεσμα είχε και η εφαρμογή της στην Random Forest.

4.4.4 Προσέγγιση

Στην παρούσα εργασία η χρήση ο συνδυασμός των PCA και Fourier φάνηκε να δίνει τα καλύτερα αποτελέσματα. Τα δεδομένα που κρατήθηκαν και χρησιμοποιήθηκαν στα τελικά πειράματα για τη σύγκριση μεθόδων που πέτυχαν και τα καλύτερα αποτελέσματα ήταν αυτά των 2 δευτερολέπτων για τους κλασικούς ταξινομητές και 5 δευτερολέπτων για το 2D-CNN. Ουσιαστικά η απόδοση της κάθε μιας αλλά και του συνδυασμού αυτών απεικονίζεται στους πίνακες με τα αποτελέσματα στο 6ο κεφάλαιο. Επιπροσθέτως η KNN χωρίς τη χρήση της StandardScaler φάνηκε να δίνει ελαφρός καλύτερα αποτελέσματα σε σχέση με τις δυο άλλες μέθοδος. Οι παραμετροποιήσεις ήταν αυτές που σε κάθε μέθοδο αύξησαν πολύ το ποσοστό επιτυχίας τους.

5 *Μοντέλα*

Όπως εξηγήθηκε και στα προηγούμενα κεφάλαια έγινε χρήση διάφορων μοντέλων. Χρησιμοποιήθηκαν συνολικά 54 δείγματα (18 υγιής, 18 MCI και 18 AD). Χρησιμοποιήθηκε το διαδικτυακό εργαλείο της Google το Colab Pro που κάνει χρήση της python 3.9 και τα πακέτα TensorFlow[53], sklearn[54], mne[46], nympy[44], [45] και pandas[55].

5.1 Εισαγωγή

Παρακάτω δίνεται η δομή και οι παράμετροι που έχει η κάθε μέθοδος. Οι πρώτες 8 αποτελούν function από τη βιβλιοθήκη της python scikit-learn [54]. Η CNN εν αντιθέσει ανήκει στη βιβλιοθήκη Tensorflow/Keras [53].

5.2 Logistic Regression Classifier

Αυτό το είδος ταξινομητή χρησιμοποιεί Λογιστική Παλινδρόμηση. Πρόκειται για εποπτευόμενη μέθοδο μάθησης.

5.2.1 Δομή

LogisticRegression(penalty='l2', dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver='lbfgs', max_iter=100, multi_class='auto', verbose=0, warm_start=False, n_jobs=None, l1_ratio=None)

5.2.2 Παράμετροι

penalty

Καθορίζεται ο κανόνας του penalty. Προκαθορισμένη τιμή 'l2'. Οι υπόλοιπες τιμές που μπορεί να πάρει είναι 'l1', 'none' δεν θα υπάρχει penalty, 'elasticnet' συγχρόνως l1 και l2.

dual

Παίρνει λογική τιμή, η προκαθορισμένη είναι false. Η dual εφαρμόζεται όταν το penalty έχει τιμή l2.

tol

Παίρνει πραγματική τιμή. Αφορά το κριτήρια παύσης και η προκαθορισμένη τιμή είναι 1e-4.

C

Παίρνει πραγματική τιμή. Αφορά την αντιστροφή της δύναμης κανονικοποίησης. Η προκαθορισμένη τιμή είναι 1.0.

fit_intercept

Παίρνει λογική τιμή. Η προκαθορισμένη είναι true. Αφορά για το εάν μπορεί μια μεταβλητή να προστεθεί στη συνάρτηση απόφασης.

intercept_scaling

Παίρνει πραγματική τιμή. Η προκαθορισμένη τιμή είναι 1.0. Χρησιμεύει όταν η παράμετρος solver έχει τιμή 'liblinear' και η παράμετρος self.fit_intercept είναι True.

class_weight

Η προκαθορισμένη τιμή είναι none και σημαίνει ότι όλες οι κατηγορίες έχουν το ίδιο βάρος.

random_state

Παίρνει ακέραιο αριθμό. Η προκαθορισμένη τιμή είναι none. Χρησιμοποιείται για την τυχαία αναπαραγωγή των δεδομένων. Η παράμετρος solver πρέπει να έχει τιμή 'sag', 'saga' ή 'liblinear'.

solver

Επιλέγουμε τον αλγόριθμο που θα χρησιμοποιηθεί για τη βελτιστοποίηση. Καθορισμένη τιμή 'lbfgs'. Άλλες τιμές που μπορεί να πάρει είναι 'newton-cg', 'liblinear', 'sag', 'sag'.

5.3 Decision Tree Classifier

Πρόκειται για εποπτευόμενη μέθοδο μάθησης. Σύμφωνα με τα χαρακτηριστικά των δεδομένων που εισάγονται βάση απλών κανόνων προσπαθεί τα ταξινομήσει τα δεδομένα.

5.3.1 Δομή

DecisionTreeClassifier(criterion='gini', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_decrease=0.0, class_weight=None, ccp_alpha=0.0)

5.3.2 Παράμετροι

Criterion

Εδώ επιλέγεται με τι κριτήρια θα γίνει ο διαχωρισμός. Οι πιθανές τιμές είναι 'gini', 'entropy' ή 'log_loss'. Η προκαθορισμένη τιμή είναι gini.

Splitter

Επιλογή για το αν ο διαχωρισμός θα γίνει τυχαία ή η καλύτερη. Οι τιμές που μπορεί να πάρει είναι ‘best’ και ‘random’ με προκαθορισμένη την best.

max_depth

Καθορίζει το μέγιστο βάθος του δέντρο σύμφωνα με τον ακέραιο που θα δώσουμε. Η προκαθορισμένη είναι None που σημαίνει οτι το δέντρο επεκτείνετε μέχρι τα φύλλα να είναι λιγότερα από το min_samples_split.

min_samples_split

Ο ελάχιστος αριθμός δειγμάτων που απαιτείται για το διαχωρισμό ενός κόμβου. Μπορεί να είναι ακέραιος ή δεκαδικός αριθμός, η προκαθορισμένη τιμή είναι 2.

min_samples_leaf

Ο ελάχιστος αριθμός δειγμάτων που θα πρέπει να έχει ένας κόμβος ο οποίος είναι φύλλο. Η προκαθορισμένη τιμή είναι 1.

min_weight_fraction_leaf

Παίρνει πραγματικό αριθμό. Η προκαθορισμένη τιμή είναι 0.0. Τα δείγματα έχουν το ίδιο βάρος όταν δεν δοθεί κάποια τιμή.

max_features

Παίρνει ακέραιο, δεκαδικό αριθμό ή 'auto', 'sqrt', 'log2'. Προκαθορισμένη τιμή none. Ο αριθμός των χαρακτηριστικών που θα χρησιμοποιηθούν κατά τη διάρκεια του διαχωρισμού.

5.4 Random Forest

Πρόκειται για εποπτευόμενη μέθοδο μάθησης η οποία χρησιμοποιεί έναν αριθμό δέντρων απόφασης.

5.4.1 Δομή

RandomForestClassifier(n_estimators=100, *, criterion='gini', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='sqrt', max_leaf_nodes=None, min_impurity_decrease=0.0, bootstrap=True, oob_score=False, n_jobs=None, random_state=None, verbose=0, warm_start=False, class_weight=None, ccp_alpha=0.0, max_samples=None)

5.4.2 Παράμετροι

Έχει αρκετές παραμέτρους που είναι ίδιες με αυτές της μεθόδου ταξινόμησης των Δέντρων Απόφασης όπως η παράμετρος Criterion, max_depth, min_samples_split, min_samples_leaf, min_samples_leaf και max_features με την ίδια ακριβώς λειτουργία όπως αυτές περιγράφονται παραπάνω, με τις ίδιες δυνατές τιμές.

n_estimators

Παίρνει ακέραια τιμή η οποία αφορά τον αριθμό των δέντρων. Η προκαθορισμένη τιμή είναι 100. Δοκιμάστηκαν οι τιμές 50, 100, 150, 170 και 200 με την 200 να δίνει το καλύτερο αποτέλεσμα.

max_samples

Παίρνει ακέραιο ή δεκαδικό αριθμό. Η προκαθορισμένη τιμή είναι none. Εδώ θα δοθεί ο αριθμός των δειγμάτων που θα χρησιμοποιηθεί για την εκπαίδευση. Χρησιμοποιείται μόνο όταν στην παράμετρο Bootstrap δοθεί η τιμή true.

max_leaf_nodes

Παίρνει ακέραια τιμή. Η προκαθορισμένη είναι none η οποία αν δοθεί σημαίνει απεριόριστος αριθμός κόμβων οι οποίοι είναι φύλλα.

min_impurity_decrease

Παίρνει πραγματικό αριθμό, Η προκαθορισμένη τιμή είναι 0.0. Καθορίζει η τιμή αν θα διαχωριστεί ένας κόμβος ή όχι.

Bootstrap

Παίρνει λογική τιμή. Η προκαθορισμένη είναι true. Αν είναι false τότε όλο το σύνολο δεδομένων χρησιμοποιείται για τη δημιουργία του δέντρου. Έγιναν δοκιμές και με τις δυο τιμές με την false να δίνει τα καλύτερα αποτελέσματα.

oob_score

Παίρνει λογική τιμή. Η προκαθορισμένη είναι false. Καθορίζεται εάν στη διαδικασία της βαθμολόγησης της γενίκευσης της μεθόδου θα χρησιμοποιηθούν άλλα δείγματα. Η false φάνηκε να δίνει καλύτερο αποτέλεσμα.

n_jobs

Παίρνει ακέραιο αριθμό ή none. Η προκαθορισμένη τιμή είναι none. Καθορίζει εάν θα τρέχουν παράλληλα εργασίες.

random_state

Καθορίζει και των αριθμό των χαρακτηριστικών που θα χρησιμοποιηθούν για το διαχωρισμό των δέντρων αλλά και στη περίπτωση που χρησιμοποιηθεί όλο το σύνολο των δεδομένων για

τη δημιουργία του δέντρου (η παράμετρος Bootstrap είναι true) την τυχαιότητα των δειγμάτων. Δοκιμάστηκαν οι τιμές 1,2 και 20 με την τελευταία να δίνει το καλύτερο αποτέλεσμα.

Verbose

Παίρνει ακέραια τιμή με προκαθορισμένη τιμή να είναι το 0. Καθορίζει τον τρόπο με τον οποίο θα δούμε τα αποτελέσματα.

warm_start

Παίρνει λογική τιμή με προκαθορισμένη τιμή το false. Αν είναι true προσθέτει εκτιμητές αλλά κάνει ένα άλλο δάσος. Η true ήταν αυτή που ανέβασε το ποσοστό των αποτελεσμάτων.

class_weight

Βάρη τα οποία σχετίζονται με τις κλάσεις. Αν δεν δοθεί τότε όλες οι κλάσεις έχουν το ίδιο βάρος. Η προκαθορισμένη τιμή είναι none. Τιμές που μπορεί να πάρει είναι: 'balanced', 'balanced_subsample'. Δοκιμάστηκαν και οι δυο με την balanced_subsample να δίνει ελαφρός καλύτερο αποτέλεσμα.

ccp_alpha

Παίρνει θετική πραγματική τιμή. Προκαθορισμένη τιμή 0.0. Χρησιμοποιείται για την επιλογή υποδέντρου. Επιλέγεται αυτό που έχει τη μεγαλύτερη πολυπλοκότητα που είναι χαμηλότερη από την τιμή της παραμέτρου.

5.5 Gaussian Naive Bayes

Πρόκειται για εποπτευόμενη μέθοδο μάθησης.

5.5.1 Δομή

GaussianNB(*, priors=None, var_smoothing=1e-09)

5.5.2 Παράμετροι

Priors

Παίρνει ένα πίνακα ο οποίος έχει μέσα προηγούμενες πιθανότητες από τις κλάσεις.

var_smoothing

Παίρνει πραγματική τιμή, η προκαθορισμένη είναι 1e-9. Χρησιμοποιείται για τον υπολογισμό της σταθερότητας.

5.6 Support Vector Machines

Πρόκειται για εποπτευόμενη μέθοδο μάθησης.

5.6.1 Δομή

SVC(*, C=1.0, kernel='rbf', degree=3, gamma='scale', coef0=0.0, shrinking=True, probability =False, tol=0.001, cache_size=200, class_weight=None, verbose=False, max_iter=- 1, decisio n_function_shape='ovr', break_ties=False, random_state=None)

5.6.2 Παράμετροι

C

Παίρνει πραγματική τιμή, η προκαθορισμένη τιμή είναι 1.0. Αφορά την κανονικοποίηση. Πρέπει να είναι θετικός αριθμός. Στην παρούσα εργασία η συγκεκριμένη παράμετρος βοήθησε στο να ανεβεί το ποσοστό επιτυχίας της μεθόδου. Η τιμή που φάνηκε να δίνει το καλύτερο αποτέλεσμα ήταν η 1.5. Άλλες τιμές που δοκιμάστηκαν ήταν η 1.0, 1.2, 1.3, 1. 6, 1.7, 1.8, 2 και 3.

Kernel

Καθορίζει τον τύπο kernel που θα χρησιμοποιηθεί στο πρόγραμμα. Οι προκαθορισμένη τιμή είναι ‘rbf’. Επίσης μπορεί να δοθεί και ένας τετραγωνικός πίνακας τύπου (n_samples, n_samples). Άλλες τιμές είναι το 'sigmoid' και 'poly'. Στην εργασία η τιμή ‘rdf’ έδωσε τα καλύτερα αποτελέσματα.

Degree

Παίρνει ακέραιο αριθμό, η προκαθορισμένη τιμή είναι 3. Αυτό αφορά μόνο την περίπτωση που η συνάρτηση είναι poly.

Gamma

Οι τιμές που μπορεί να πάρει είναι 'scale', 'auto' ή μια πραγματική τιμή, η προκαθορισμένη τιμή είναι 'scale'. Είναι ένας συντελεστής που μπορούμε να βάλουμε σε περίπτωση που η παράμετρος kernel είναι 'rbf', 'poly' ή 'sigmoid'. Η τιμή που έδωσε τα καλύτερα αποτελέσματα ήταν το 1.4e-1. Δοκιμάστηκαν επίσης οι τιμές 1e-1, 1.2e-1, 1.5e-1, 1.6e-1 και 1.7e -1.

coef0

Παίρνει δεκαδικό αριθμό. Η προκαθορισμένη τιμή είναι 0.0. Χρησιμοποιείται στη συνάρτηση kernel όταν αυτήν είναι 'poly' ή 'sigmoid'.

Shrinking

Παίρνει λογική τιμή. Η προκαθορισμένη τιμή είναι true. Αφορά το αν θα γίνει συρρίκνωση ή όχι. Δοκιμάστηκαν και οι δυο λογικές τιμές και το αποτέλεσμα δεν επηρεάστηκε καθόλου.

Probability

Παίρνει λογική τιμή. Η προκαθορισμένη τιμή είναι false. Αφορά το εάν θα γίνει εκτίμηση πιθανοτήτων ή όχι. Δοκιμάστηκαν και οι δυο λογικές τιμές και το αποτέλεσμα δεν επηρεάστηκε καθόλου.

Tol

Παίρνει πραγματική τιμή. Η προκαθορισμένη τιμή είναι 1e-3. Είναι το κριτήριο ανοχής για την παύση. Δοκιμάστηκαν και οι τιμές 1.3e-3, 1.5e-3 και 1.8e-3 οι οποίες δεν επηρέασαν καθόλου το τελικό αποτέλεσμα.

cache_size

Παίρνει πραγματική τιμή. Η προκαθορισμένη τιμή είναι 200. Αφορά το μέγεθος της μνήμης cache για τον Kernel.

class_weight

Η προκαθορισμένη τιμή είναι none κατά την οποία όλες οι κλάσεις έχουν τα ίδια βάρη. Εφόσον έχει δοθεί τιμή στην παράμετρο C για μια κλάση x ισχύει class_weight[x]*C.

Verbose

Παίρνει λογική τιμή. Η προκαθορισμένη είναι false. Καθορίζει την έξοδο των αποτελεσμάτων.

max_iter

Παίρνει ακέραιο αριθμό. Η προκαθορισμένη τιμή είναι -1 το οποίο σημαίνει ότι δεν υπάρχει όριο. Καθορίζει το όριο των επαναλήψεων.

decision_function_shape

Παίρνει τις τιμές 'ovo' ή 'ovr' με προκαθορισμένη τιμή 'ovr'. Καθορίζει εάν όπως στους υπόλοιπους classifier θα επιστραφεί μι συνάρτηση απόφασης one-vs-rest ή η αρχική συνάρτηση απόφασης one-vs-one.

break_ties

Παίρνει λογική τιμή. Η προκαθορισμένη τιμή είναι false όπου και επιστρέφει σε περίπτωση ισοβαθμίας την πρώτη κατηγορία.

random_state

Παίρνει ακέραιο αριθμό. Η προκαθορισμένη τιμή είναι none. Αφορά τη δημιουργία τυχαίων αριθμών για το ανακάτεμα των δεδομένων στη πρόβλεψη πιθανοτήτων. Δοκιμάστηκαν και άλλες τιμές αλλά το τελικό αποτέλεσμα δεν επηρεάστηκε καθόλου.

5.7 K Nearest Neighbors

Πρόκειται για εποπτευόμενη μέθοδο ταξινόμησης.

5.7.1 Δομή

KNeighborsClassifier(n_neighbors=5, weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=None)

5.7.2 Παράμετροι

n_neighbors

Αναφέρεται για τον αριθμό των γειτόνων που θα χρησιμοποιήσει. Η προκαθορισμένη τιμή είναι 5. Στην παρούσα εργασία η τιμή 3 φάνηκε να έχει τα καλύτερα αποτελέσματα.

Weights

Τα βάρη χρησιμοποιούνται για την πρόβλεψη. Η προκαθορισμένη τιμή είναι 'uniform'. Οι τιμές που μπορεί να πάρει η συγκεκριμένη παράμετρος είναι:

- 'uniform'. Σε αυτήν την περίπτωση όλοι οι γείτονες έχουν το ίδιο βάρος.
- 'distance'. Εδώ οι πιο κοντινοί γείτονες έχουν μεγαλύτερη βαρύτητα απ' ότι οι πιο μακρινοί.
- [callable]. Σε αυτήν την περίπτωση ο χρήστης δημιουργεί ένα πίνακα με αποστάσεις και επιστρέφεται ένας πίνακας με το ίδιο μέγεθος που έχει τα αντίστοιχα βάρη για κάθε απόσταση.

Στην παρούσα εργασία η τιμή distance έδωσε τα καλύτερα αποτελέσματα. Δοκιμάστηκε και η τιμή uniform αλλά το αποτέλεσμα ήταν ελαφρός χειρότερα.

Algorithm

Ο αλγόριθμός που χρησιμοποιείται για την πρόβλεψη των γειτόνων. Οι τιμές αυτής της παραμέτρου μπορεί να είναι: 'ball_tree', 'kd_tree', 'brute' και 'auto' κατά την οποία θα προσπαθήσει να αποφασίσει από μόνο του σύμφωνα με τα δεδομένα ποια είναι η κατάλληλη μέθοδος. Η προκαθορισμένη τιμή είναι 'auto'. Δοκιμάστηκαν και οι 4 τιμές και δε παρατηρήθηκε καμία αλλαγή στα αποτελέσματα.

leaf_size

Παίρνει έναν ακέραιο αριθμό. Η προκαθορισμένη τιμή είναι 30. Εξαρτάται από το είδος του προβλήματος ποια θα είναι η καλύτερη τιμή. Χρησιμοποιείται σε περίπτωση που ο αλγόριθμος είναι BallTree ή ο KDTree. Μπορεί να επηρεάσει την ταχύτητα καθώς και τη μνήμη που θα χρειαστεί για την κατασκευή του δέντρου. Δόθηκαν επίσης οι τιμές 10, 20 και 50 με το τελικό αποτέλεσμα να μένει ίδιο.

p

Παίρνει έναν ακέραιο αριθμό. Η προκαθορισμένη τιμή είναι 2. Καθορίζει αν θα χρησιμοποιήσει ευκλείδεια απόσταση (p=2) ή manhattan (p=1). Η τιμή 1 έδωσε το καλύτερο αποτέλεσμα.

Metric.

Εδώ επιλέγουμε τη μέθοδο για τη μέτρηση της απόστασης που θέλουμε για το δέντρο. Η προκαθορισμένη είναι 'minkowski'. Αν το p είναι 2 έχουμε ευκλείδεια. Οι άλλες επιλογές που έχουμε είναι 'manhattan', 'chebyshev', 'minkowski', 'wminkowski', 'seuclidean' και 'mahalanobis'. Οι τιμές minkowski και manhattan ήταν αυτές που έδωσαν το καλύτερο αποτέλεσμα.

metric_params

Μπορούμε να βάλουμε πρόσθετα ορίσματα που αφορούν τη συνάρτηση για τη μέθοδο μέτρησης που θα χρησιμοποιηθεί. Η προκαθορισμένη τιμή είναι None.

n_jobs

Πόσες εργασίες για αναζήτηση γειτόνων γίνονται ταυτόχρονα. Παίρνει ένα αριθμό. Για παράδειγμα το 1, που είναι και η default τιμή, σημαίνει ότι δεν γίνεται κάτι παράλληλα ενώ αν είναι -1 χρησιμοποιεί όλες τις GPU που είναι διαθέσιμες.

5.8 AdaBoost

Πρόκειται για εποπτευόμενη μέθοδο μάθησης.

5.8.1 Δομή

AdaBoostRegressor(base_estimator=None, *, n_estimators=50, learning_rate=1.0, loss='linear', random_state=None)Parameters

5.8.2 Παράμετροι

base_estimator

Παίρνει σαν τιμή ένα αντικείμενο. Η προκαθορισμένη τιμή είναι none. Αφορά το βασικό εκτιμητή. Αν δεν δοθεί κάτι τότε είναι ο DecisionTreeRegressor με αρχικό μέγιστο βάθος 3.

n_estimators

Παίρνει ακέραια τιμή. Η προκαθορισμένη είναι 50. Παίρνει θετικές τιμές. Αφορά το μέγιστο αριθμό εκτιμητών που σταματάει η διαδικασία.

learning_rate

Παίρνει πραγματικό αριθμό. Η προκαθορισμένη τιμή είναι 1.0. Αφορά το ρυθμό εκμάθησης.

Loss

Οι τιμές που παίρνει είναι 'linear', 'quare', 'exponential' με προκαθορισμένη την 'linear'. Αφορά τη συνάρτηση που χρησιμοποιείται για τον επαναπροσδιορισμό των βαρών κάθε φορά.

random_state

Παίρνει ακέραια τιμή. Η προκαθορισμένη τιμή είναι none. Χρησιμοποιείται μόνο όταν ο βασικός εκτιμητής εκθέτει μια τυχαία κατάσταση.

5.9 Quadratic Discriminant Analysis

Πρόκειται για εποπτευόμενη μέθοδο μάθησης.

5.9.1 Δομή

QuadraticDiscriminantAnalysis(*, priors=None, reg_param=0.0, store_covariance=False, tol=0.0001)

5.9.2 Παράμετροι

Priors

Δέχεται ένα πίνακα. Η προκαθορισμένη τιμή είναι none όπου και οι αναλογίες των κλάσεων συμπεραίνονται από την εκπαίδευση των δεδομένων.

reg_param

Δέχεται έναν ακέραιο. Η προκαθορισμένη τιμή είναι 0.0. Ρυθμίζει τις εκτιμήσεις συνδιακύμανσης ανά κλάση.

store_covariance

Δέχεται λογική τιμή. Η προκαθορισμένη είναι false. Αν είναι true οι πίνακες συνδιακύμανσης αποθηκεύονται στην παράμετρο store_covariance.

Tol

Παίρνει πραγματική τιμή. Η προκαθορισμένη είναι 1.0e-4. Είναι το όριο που τίθεται για την εκτίμηση της κατάταξης δημιουργώντας ένα πίνακα δειγμάτων.

5.10 Convolutional Neural Networks 2D

5.10.1 Δομή

model = models.Sequential()
model.add(layers.Conv2D(*layer 1*,(*kernel*), strides=(*stride*),
input_shape=(*ImageSize,Outpouts*)))

```
model.add(layers.MaxPooling2D(pool_size=(3, 3), strides=1, padding='same'))
ή
model.add(layers.AveragePooling2D(pool_size=(3, 3), strides=1, padding='same'))
model.add(Dropout(dropout))
model.add(BatchNormalization())
model.add(Dense(layer 2, activation='relu'))
model.add(layers.Flatten())
model.add(layers.Dense(num_classes, activation='softmax'))


model.compile( optimizer=tf.keras.optimizers.Adam(learning_rate=LearningRate), loss="LossType", metrics= ["accuracy"])

#Δοκιμάζουμε το μοντέλο μας, εκτελούμε για να δούμε πόσο καλά πήγε
Modelhistory = model.fit(X_train,
                         y_train,
                         validation_data=(X_valid, y_valid),
                         batch_size=BatchSize,
                         steps_per_epoch = StepsPerEpoch,
                         epochs = Epochs,
                         verbose = 1)
```

5.10.2 Παράμετροι

layer 1, layer 2, num_classes

Δέχεται σαν παράμετρο έναν ακέραιο αριθμό. Ουσιαστικά θέτουμε τα layer κάθε φορά. Στο τελευταίο (num_classes) θα βάλουμε τον αριθμό των κλάσεων που έχουμε. Αρχίζουμε από μικρότερο αριθμό για να συλλέξουμε όσο το δυνατόν περισσότερες πληροφορίες και στην πορεία αυξάνουμε για να έχουμε πιο καθολικές – αντιπροσωπευτικές πληροφορίες.

Kernel

Πρόκειται για ένα φίλτρο στο οποίο μπαίνουν τα δεδομένα. Αφορά το πόση πληροφορία θέλουμε να κρατήσουμε, μεγαλύτερή αν θέλουμε μεγαλύτερο αριθμό pixel ή μικρότερη αν θέλουμε πιο συμπυκνωμένη πληροφορία. Είναι της μορφής Y x Y όπου το Y συνήθως είναι περιττός αριθμός.

Stride

Αναφέρεται στο διασκελισμό που θα κάνει κατά τη διαδικασία του pooling. Είναι ο αριθμός των pixel που θέλουμε να παραλείψουμε οριζόντια ή κάθετα. Χρησιμοποιείται για να μειώσουμε αρκετά τα δεδομένα εισόδου.

ImageSize

Εδώ δίνονται οι διαστάσεις τις εικόνας πχ. Αν η εικόνα είναι 32x32 βάζουμε 32,32 ή 432,288 όπως στην εργασία αυτή αν οι διαστάσεις της εικόνας είναι 432x288.

Outpouts

Εδώ δίνουμε των αριθμό των εξόδων που θέλουμε να έχει. Στην παρούσα εργασία δόθηκε ο αριθμός 19 καθώς 19 ήταν τα κανάλια που χρησιμοποιήσαμε για τον κάθε ασθενή.

Padding

Χρησιμοποιείται για να συμπληρώσει γραμμές ή στήλες με μηδενικά έτσι ώστε να κρατηθεί ένα σταθερό μέγεθος για να μην χαθεί πληροφορία που βρίσκεται στις άκρες. Παίρνει την τιμή 'same' ή 'valid'.

Dropout

Παίρνει ένα δεκαδικό αριθμό. Ο αριθμός που ταιριάζει μπορεί να βρεθεί μετά από πειράματα. Στην προκειμένη εργασία το 0.5 έδωσε το καλύτερο αποτέλεσμα. Με αυτό το μηχανισμό μερικοί νευρώνες απορρίπτονται έτσι ώστε να μην υπάρξει overfitting.

activation

έχει τις επιλογές 'relu', 'softmax', 'linear' και 'sigmoid'. Βάζουμε συνήθως τη relu η οποία βοηθά να ακολουθήσει ένα μη γραμμικό τρόπο διαχωρισμού των κλάσεων. Φαίνεται να είναι πιο αποτελεσματική από τη σιγμοειδή. Στο τελευταίο activation επιλέγουμε 'softmax'.

pooling

Το συγκεκριμένο στρώμα χρησιμοποιείται για τη μείωση χαρακτηριστικών. Εφαρμόζεται μετά από ένα στρώμα συνέλιξης. Έχουμε τις επιλογές MaxPooling και AveragePooling (εικόνες 6 και 7 στην παράγραφο 3.11). Καλύτερα αποτελέσματα είχε η χρήση του AveragePooling. BatchNormalization()

Τοποθετείται μετά τη συνάρτηση ενεργοποίησης, και χρησιμοποιείται για την κανονικοποίηση (μέσο κεντράρισμα και κλιμάκωση διακύμανσης) της εισόδου για να επόμενα στρώματα. Σε περίπτωση που η συνάρτηση είναι σιγμοειδής όμως πρέπει να τοποθετηθεί πριν την ενεργοποίηση της συνάρτησης.

layers.Flatten()

Κάνει την είσοδο μια μονοδιάστατη λίστα.

Optimizers

Μπορεί να πάρει Adamdelta, Adamgrad, Adam, RMSprop, SGD, Adamax και Nadam. Στην συγκεκριμένη εργασία ο Nadam φάνηκε να έχει την καλύτερη απόδοση πάρα ταύτα όμως έχει μεγάλο loss.

learning_rate

Παίρνει μια δεκαδική τιμή ανάμεσα στο 0 και το 1. Αφορά το ρυθμό εκμάθησης που επιθυμούμε να έχει το μοντέλο μας, δηλαδή το πόσο γρήγορα συγκλίνουν τα βάρη. Βοηθά στην προσαρμογή των βαρών στη συνάρτηση σφάλματος που έχουμε επιλέξει. Αν ο αριθμός είναι

μικρός τότε χρειάζεται πολύ χρόνο, αντιστρόφως όμως αν είναι μεγάλος μπορεί η μάθηση μπορεί να είναι ασταθής. Το ποια τιμή θα είναι η καλύτερη βρίσκεται μετά από δοκιμές.

loss

Δέχεται τις τιμές 'MSE' (για regression, είναι η μέση τετραγωνική απώλεια για κάθε παράδειγμα) και 'categorical_crossentropy' ή 'binary_crossentropy' για classification. Στην παρούσα εργασία επιλέχθηκε το 'categorical_crossentropy'. Πρέπει για την αξιολόγηση των αποτελεσμάτων να λαμβάνεται υπόψιν και ο δείκτης loss. Είναι ένας δείκτης που δείχνει πόσο κακή μπορεί να είναι η πρόβλεψη ενός μοντέλου σε ένα παράδειγμα. Αν δεν υπάρχουν καθόλου λάθη το loss είναι μηδέν. Όσο μεγαλύτερο είναι τόσο χειρότερο είναι το μοντέλο πρόβλεψης.

metrics

Μπορεί να πάρει τιμές που αφορούν τον τρόπο με τον οποίο θα υπολογίσουμε την απόδοση του μοντέλου. Μπορεί να πάρει τις τιμές Accuracy, Sensitivity, Specificity κα. Στην παρούσα εργασία η απόδοση μετρήθηκε σύμφωνα με την ακρίβεια (Accuracy).

batch_size

Δέχεται ακέραιο αριθμό και καθορίζει τον αριθμό των δειγμάτων που θα χρησιμοποιηθούν για μια ενημέρωση των παραμέτρων του μοντέλου. Ιδανικά θα έπρεπε να χρησιμοποιηθούν όλα τα δείγματα. Το μέγεθός τους θα καθορίσει το χρόνο που απαιτείται για την εκπαίδευση καθώς και την ακρίβεια του μοντέλου.

steps_per_epoch

Παίρνει ακέραια τιμή και πρέπει να είναι σε συνδυασμό με το batch_size έτσι ώστε να παρθούν όσο το δυνατόν περισσότερα δείγματα για εκμάθηση.

epochs

καθορίζει τον αριθμό των φορών που θα λειτουργήσει ο αλγόριθμος πάνω στα δεδομένα εκμάθησης.

verbose

Παίρνει τιμές 0, 1 ή 2. Ουσιαστικά αναφέρει κανείς τον τρόπο με τον οποίο θέλει να βλέπει την διαδικασία εκμάθησης. Αν βάλει 0 δεν βλέπει τίποτα, αν βάλει 1 βλέπει μια κινούμενη γραμμή όπως [=======] και αν βάλει 2 βλέπει απλά σε ποια εποχή είναι πχ αν έχει 20 εποχές και είναι στη δεύτερη βλέπει Epoch 2/20. Στις περιπτώσεις 1 και 2 μπορεί να διακρίνει κανείς και ακρίβεια και loss για κάθε εποχή.

6 Πειράματα και Αξιολόγηση

Εδώ θα παρουσιάσουμε τα αποτελέσματα των πειραμάτων που έχουν γίνει για κάθε μοντέλο και τις παραμετροποιήσεις που έγιναν καθώς και για τις μεθόδους προεπεξεργασίας των δεδομένων.

6.1 Παράμετροι αξιολόγησης

Για την αξιολόγηση ως προς τα δεδομένα χρησιμοποιήθηκαν δυο μέθοδοι. Στην πρώτη τα δεδομένα χωρίστηκαν σε ποσοστό 80% για εκπαίδευση και 20% για δοκιμή. Στη δεύτερη έγινε διαχωρισμός ανά ασθενή, έμενε εκτός της εκπαίδευσης ένας ασθενής και το ποσοστό που εμφανίζεται αφορά το ποσοστό επιτυχίας ανακάλυψης της κλάσης του συγκεκριμένου ασθενή. Δοκιμάστηκαν και 54 ασθενείς. Επίσης δοκιμάστηκαν και οι τρεις κλάσεις αλλά και ανά δυο ζεύγη κλάσεων.

6.2 Σύστημα αξιολόγησης

Για την αξιολόγηση αυτό που λάβαμε υπόψιν ήταν η ακρίβεια (accuracy). Στην περίπτωση που χρησιμοποιήθηκαν όλα τα δεδομένα και αποκρύφτηκαν κάποια segment έγινε και cross validation. Επιπροσθέτως στις περιπτώσεις ανά ασθενή λήφθηκε υπόψιν και πόσους ασθενείς κατέταξε σωστά.

6.3 Οργάνωση πειραμάτων

Στην περίπτωση των 2D Convolutional Neural Networks τα δεδομένα ήταν κοινά για όλα τα πειράματα, το σύνολο των δεδομένων ήταν ένα πίνακας με 3240 γραμμές. Εμφανίζονται σε πίνακα οι αλλαγές στις παραμέτρους που έγιναν στο κάθε πείραμα και το αποτέλεσμα που έδωσε η αλλαγή αυτή.

Τα πειράματα όσο αφορά τις υπόλοιπες κλασσικές μεθόδους θα χωριστούν σε τρεις κατηγορίες. Σε αυτά που δεν χρησιμοποιήθηκε καμία προεπεξεργασία στα δεδομένα, σε αυτά που έγιναν διάφορες προεπεξεργασίες πριν τα δεδομένα εισαχθούν στη μέθοδο αλλά χωρίς

καμία παραμετροποίηση στη μέθοδο και τέλος στις διάφορες παραλλαγές όσο αφορά τις διάφορες παραμέτρους των μεθόδων. Για τα πειράματα ανά ασθενή καθώς και σε αυτά που έγιναν ανά ζεύγη κλάσεων οι μέθοδοι εκτελέστηκαν με τις παραμέτρους που έδωσαν το καλύτερο αποτέλεσμα στις εκτελέσεις ανά segment με τη χρήση τριών κλάσεων.

Ο διαχωρισμός των δεδομένων όσο αφορά το διαχωρισμό σε δεδομένα για εκπαίδευση και τεστ ήταν 80% για εκπαίδευση και 20% δοκιμή στις περιπτώσεις ανά segment. Όταν χρησιμοποιήθηκε η παράμετρος stratify στο διαχωρισμό σε κάποιες μεθόδους ανέβηκε το ποσοστό επιτυχίας ενώ σε άλλες δεν υπήρξε διαφορά από τη χρήση της. Στο διαχωρισμό ανά ασθενή ουσιαστικά χρησιμοποιήθηκαν 150 segment που έχει ο ένας ασθενής για δοκιμή και τα υπόλοιπα για εκπαίδευση. Στις περιπτώσεις αυτών των μεθόδων που χρησιμοποιήθηκαν και οι τρεις κλάσεις δημιουργήθηκε ένας πίνακας με 8100 γραμμές, οπότε τα δεδομένα για εκπαίδευση ήταν οι 7950 γραμμές, ενώ στην περίπτωση των πειραμάτων ανά ζεύγη το σύνολο των δεδομένων ήταν 5400 γραμμές με τις 5250 να χρησιμοποιούνται για εκπαίδευση.

6.4 Αποτελέσματα

Στην παρούσα ενότητα δίνεται η παρουσίαση των αποτελεσμάτων με μορφή πινάκων.

6.4.1 2D Convolutional Neural Networks

6.4.1.1 Παραμετροποίηση στο χτίσιμο του μοντέλου

Οι δοκιμές έγιναν με optimizer = SGD, learning rate=0.0001, batch size=5, steps per epoch=499 και epochs=20.

	channels	kernels	stride	dropout	Pooling	Dence relu	Total params	loss	accuracy	correct labels	incorrect labels
1	32	3x3	3x3	0,4	Max	64	2661891	3,210	0,574	185	139
2		5x5	3x3				2625795	2,820	0,617	201	123
3			5x5				958659	1,853	0,593	190	134
4		7x7					973251	1,816	0,571	183	141
5			7x7				512259	1,913	0,512	165	159
6		9x9					520003	1,625	0,559	179	145
7			9x9				346435	1,591	0,500	161	163
8		5x5	3x3		Average		2625795	1,037	0,642	207	117
9	64					128	5252683	1,174	0,670	216	108
10					Max		5252683	4,839	0,611	198	126
11		9x9	7x7		Average		1044099	0,880	0,620	199	125
12		5x5	3x3	0,3			5252683	1,261	0,642	208	216

13	64			0,5		128	5252683	1,072	0,688	223	101
14				0,6			5252683	1,281	0,642	209	115
15	128			0,5		256	10527747	1,994	0,614	199	125

6.4.1.2 Παραμετροποίηση στο compile

Οι δοκιμές έγιναν με την παράμετρο channels να είναι στην τιμή 64, το dropout=0.5, το Pooling να είναι Average, το Dence Relu 128, το batch size=5, steps per epoch=499 και epochs=20.

	kernels	stride	Total params	optimizer	Learning rate	loss	accuracy	correct labels	incorrect labels
16	5x5	3x3	5252683	SGD	0,001	1,133	0,667	217	107
17			5252683		0,00001	1,093	0,657	214	110
18	5x5	3x3	5252683	Adam	0,0001	1,251	0,648	208	116
19	7x7	5x5	1950595			2,185	0,694	226	98
20		3x3	5193859			9,567	0,556	180	144
21	9x9	7x7	1044099			1,660	0,657	213	111
22		5x5	1934979			2,419	0,648	210	114
23	5x5	3x3	5252683	RMSprop		**2,260**	**0,704**	**228**	**96**
24	7x7	5x5	1950595			2,486	0,651	211	113
25		3x3	5193859			22,797	0,386	125	199
26	9x9	7x7	1044099			2,358	0,586	188	136
27		5x5	1934979			3,265	0,574	186	138
28	5x5	3x3	5252683	Adamax		3,007	0,688	223	101
29	7x7	5x5	1950595			1,214	0,664	214	110
30		3x3	5193859			2,065	0,660	215	109
31	9x9	7x7	1044099			0,959	0,679	225	99
32		5x5	1934979			1,607	0,694	226	98
33	5x5	3x3	5252683	Nadam		1,597	0,707	229	95
34	7x7	5x5	1950595			2,896	0,633	205	119
35		3x3	5193859			7,584	0,586	190	134
36	9x9	7x7	1044099			1,352	0,667	217	107
37		5x5	1934979			2,024	0,701	227	97

6.4.1.3 Παραμετροποίηση στο model.fit

Οι δοκιμές έγιναν με channels να είναι στην τιμή 64, το dropout=0.5, το Pooling να είναι Average και το Dence Relu 128.

	kernels	stride	Total params	optimizer	Learning rate	Batch size	Steps per epoch	epochs	loss	accuracy	correct labels	incorrect labels
38	5x5	3x3	5252683	SGD	0,0001	10	259	20	1,093	0,639	208	116
39			5252683			15	172		1,039	0,651	210	114
40			5252683			20	129		1,037	0,645	207	117
41			5252683			5	499	25	**1,050**	**0,673**	**215**	**109**
42			5252683					30	1,156	0,664	212	112
43			5252683	RMSprop		10	259	20	7,004	0,660	214	110
44			5252683			15	172		10,813	0,664	215	109
45			5252683			20	129		32,252	0,426	138	186
46			5252683			5	499	25	6,811	0,676	219	105
47			5252683					30	6,806	0,673	218	106
48			5252683	Nadam		10	259	20	6,178	0,645	208	116
49			5252683			15	172		8,149	0,602	195	129
50			5252683			20	129		12,149	0,552	179	145
51			5252683			5	499	25	6,786	0,608	197	127
52			5252683					30	5,344	0,679	219	105
53	7x7	5x5	1950595	Adam		10	259	20	2,907	0,611	199	125
54			1950595			15	172		3,173	0,685	223	101
55			1950595			20	129		3,086	0,710	230	94
56			1950595			5	499	25	2,344	0,679	220	104
57			1950595					30	3,596	0,651	211	113
58	9x9	5x5	1934979	Adamax		10	259	20	2,973	0,685	222	102
59			1934979			15	172		3,024	0,691	225	99
60			1934979			20	129		3,113	0,694	226	98
61			1934979			5	499	25	3,696	0,698	226	98
62			1934979					30	3,967	0,694	224	100

6.4.2 Σύγκριση Κλασσικών ταξινομητών

6.4.2.1 Χωρίς Προεργασία

Δεν χρησιμοποιήθηκε κάποιο φίλτρο, ούτε Fourier ή PCA. Επίσης στις μεθόδους δεν έγινε κάποια παραμετροποίηση αλλά χρησιμοποιήθηκαν στη γενική τους μορφή.

6.4.2.1.1 Weighted Avg Accuracy

Classifier	Score	Classifier	Score	Classifier	Score	Weighted Avg Accuracy
SVM	0,888	DecisionTree	0,682	KNN	0,911	93,235
LogisticRegression	0,444	DecisionTree	0,683	GaussianNB	0,533	66,963
SVM	**0,888**	**Random Forest**	**0,913**	**KNN**	**0,911**	**94,173**
AdaBoost	0,591	QDA	0,344	KNN	0,911	92,148

6.4.2.1.2 Random Forest

	filter	fourier	pca	Accuracy	Predicted classes
Random Forest	no	no	no	0.922	2

6.4.2.1.3 K Nearest Neighbors

	filter	fourier	pca	Accuracy	Predicted classes
KNeighbors	no	no	no	0.941	3

6.4.2.1.4 SVM

	filter	fourier	pca	Accuracy	Predicted classes
SVM	no	no	no	0.920	3

6.4.2.2 Με Προεργασία, χωρίς παραμετροποίηση στις μεθόδους

6.4.2.2.1 Weight Avg Accuracy

Χρήση Φίλτρων, χωρίς Fourier, χωρίς PCA

Classifier	Score	Classifier	Score	Classifier	Score	Weighted Avg Accuracy
SVM	0.485	DecisionTree	0.403	KNN	0.445	46,099
LogisticRegression	0,340	DecisionTree	0,380	GaussianNB	0,469	44,840
SVM	0.485	Random Forest	0.532	KNN	0.445	52,321
AdaBoost	0,453	QDA	0,331	KNN	0,445	39,086

Με χρήση φίλτρων, με Fourier, χωρίς PCA.

Classifier	Score	Classifier	Score	Classifier	Score	Weighted Avg Accuracy
SVM	0,455	DecisionTree	0,413	KNN	0,437	50,272
LogisticRegression	0,335	DecisionTree	0,413	GaussianNB	0,490	45,012
SVM	0,455	Random Forest	0,564	KNN	0,437	57,160

AdaBoost	0,533	QDA	0,337	KNN	0,437	40,420

Με χρήση φίλτρων, χωρίς Fourier, με PCA (n_components=19).

Classifier	Score	Classifier	Score	Classifier	Score	Weighted Avg Accuracy
SVM	0,459	DecisionTree	0,425	KNN	0,446	44,321
LogisticRegression	0,330	DecisionTree	0,418	GaussianNB	0,441	42,222
SVM	0,459	Random Forest	0,498	KNN	0,446	49,728
AdaBoost	0,451	QDA	0,501	KNN	0,446	49,333

Με χρήση φίλτρων, με Fourier, με PCA (n_components=19).

Classifier	Score	Classifier	Score	Classifier	Score	Weighted Avg Accuracy
SVM	0,454	DecisionTree	0,418	KNN	0,456	46,593
LogisticRegression	0,338	DecisionTree	0,419	GaussianNB	0,451	43,728
SVM	0,454	Random Forest	0,518	KNN	0,456	52,247
AdaBoost	0,445	QDA	0,509	KNN	0,456	52,247

Χωρίς τη χρήση φίλτρων, χωρίς Fourier, με PCA (n_components=19).

Classifier	Score	Classifier	Score	Classifier	Score	Weighted Avg Accuracy
SVM	0.908	DecisionTree	0.739	KNN	0.916	94,099
LogisticRegression	0.502	DecisionTree	0.746	GaussianNB	0.516	76,247
SVM	0.908	Random Forest	0.928	KNN	0.916	94,938
AdaBoost	0.655	QDA	0.754	KNN	0.916	92,840

Χωρίς τη χρήση φίλτρων, με Fourier, χωρίς PCA (n_components=190).

Classifier	Score	Classifier	Score	Classifier	Score	Weighted Avg Accuracy
SVM	0.89	DecisionTree	0.721	KNN	0.921	94,494
LogisticRegression	0.438	DecisionTree	0.732	GaussianNB	0.433	75,086
SVM	0.890	Random Forest	0.679	KNN	0.921	95,111
AdaBoost	0.658	QDA	0.415	KNN	0.921	93,630

Χωρίς τη χρήση φίλτρων, με Fourier, με PCA (n_components=19).

Classifier	Score	Classifier	Score	Classifier	Score	Weighted Avg Accuracy
SVM	0,908	DecisionTree	0,743	KNN	0,916	93,975
LogisticRegression	0,496	DecisionTree	0,738	GaussianNB	0,519	76,716
SVM	0,908	Random Forest	0,936	KNN	0,916	95,136
AdaBoost	0,657	QDA	0,762	KNN	0,916	93,111

Χωρίς τη χρήση φίλτρων, με Fourier, με PCA (n_components=19) και StandarSceler.

Classifier	Score	Classifier	Score	Classifier	Score	Weighted Avg Accuracy
SVM	0,941	DecisionTree	0,742	KNN	0,942	96,198
LogisticRegression	0,499	DecisionTree	0,729	GaussianNB	0,519	76,593
SVM	0,941	Random Forest	0,928	KNN	0,942	96,765
AdaBoost	0,657	QDA	0,762	KNN	0,942	92,963

6.4.2.2.2 Random Forest

Δοκιμή συνδυασμού φίλτρου, Fourier και PCA, Δοκιμές με 3 κλάσεις χωρισμός ανά Segment. Στην περίπτωση που δεν χρησιμοποιήθηκαν φίλτρα, αλλά χρησιμοποιήθηκε Fourier και PCA εφαρμόστηκε επίσης και StandardScaler.

Filter	Fourier	PCA	Accuracy	StandardScaler
no	no	n_components=19	0.946	
		n_components=190	0.894	
		n_components=1900	0.733	
no	yes	no	0.764	
		n_components=19	0.952	0.951
		n_components=190	0.901	0.914
		n_components=1900	0.722	0.743
yes	no	no	0.545	
		n_components=19	0.512	
		n_components=190	0.516	
		n_components=1900	0.464	
yes	yes	no	0.594	
		n_components=19	0.539	
		n_components=190	0.551	
		n_components=1900	0.471	

6.4.2.2.3 K Nearest Neighbors

Δοκιμή συνδυασμού φίλτρου, Fourier και PCA, Δοκιμές με 3 κλάσεις χωρισμός ανά Segment. Εφαρμόστηκε και StandardScaler σε αυτά που δεν είχαν φίλτρα, αλλά είχαν Fourier και PCA με τα αποτελέσματα από τη χρήση της στα συγκεκριμένα πειράματα να εμφανίζονται ως επί το πλείστον χειρότερα σε σχέση με αυτά που δεν χρησιμοποιήθηκε.

Filter	Fourier	PCA	Accuracy	StandardScaler
no	no	n_components=19	0,954	
		n_components=190	0,945	
		n_components=1900	0,94	

no	yes	no	0,957	0.343
		n_components=19	0,957	0.961
		n_components=190	0,954	0.576
		n_components=1900	0,956	0.332
yes	yes	no	0,471	
		n_components=19	0,474	
		n_components=190	0,485	
		n_components=1900	0,471	
yes	no	no	0,458	
		n_components=1900	0,464	
		n_components=190	0,471	
		n_components=19	0,46	

6.4.2.2.4 SVM

Δοκιμή συνδυασμού φίλτρου, Fourier και PCA, Δοκιμές με 3 κλάσεις χωρισμός ανά Segment. Εφαρμόστηκε και StandardScaler σε αυτά που δεν είχαν φίλτρα, αλλά είχαν Fourier και PCA με τα αποτελέσματα από τη χρήση της στα συγκεκριμένα πειράματα να εμφανίζονται χειρότερα σε σχέση με αυτά που δεν χρησιμοποιήθηκε.

Filter	Fourier	PCA	Accuracy	StandardScaler
no	no	no	0.920	
		n_components=19	0.927	
		n_components=190	0.932	
		n_components=1900	0.930	
no	yes	no	0.923	
		n_components=19	0.943	0.952
		n_components=190	0.951	0.784
		n_components=1900	0,925	0.496
yes	yes	no	0.552	
		n_components=19	0.530	
		n_components=190	0.548	
		n_components=1900	0.552	
yes	no	no	0.532	
		n_components=19	0.533	
		n_components=190	0.539	
		n_components=1900	0.516	

6.4.2.3 Με παραμετροποίηση

Επιλέχθηκαν οι τρεις μέθοδοι, Random Forest, KNN και SVM, που έδωσαν τα καλύτερα ποσοστά και έγιναν παραμετροποιήσεις και ως προς το διαχωρισμό των δεδομένων πριν την είσοδο σε κάθε μέθοδο, όσο και ως προς τις παραμέτρους της κάθε μεθόδου. Δεν έγινε η χρήση

φίλτρων αφού φάνηκε στα προηγούμενα πειράματα η χρήση τους να δίνει μικρότερο ποσοστό επιτυχίας. Έγινε χρήση Fourier.

6.4.2.3.1 Παράμετροι στο διαχωρισμό των δεδομένων

Οι δυο παράμετροι που χρησιμοποιήθηκαν στη διαδικασία του διαχωρισμού των δεδομένων ήταν το random_state και stratify=y. Δοκιμάστηκαν με και χωρίς StandarScaler και με PCA.

6.4.2.3.1.1 Random Forest

PCA	**random_state=1**		**stratify=y**	
n_components	StandardScaler		StandardScaler	
	no	yes	no	yes
19	0.955	0.954	0.955	0.946
190	0.904	0.913	0.903	0.897
1900	0.747	0.736	0.723	0.734

6.4.2.3.1.2 K Nearest Neighbors

PCA	**random_state=1**		**stratify=y**	
n_components	StandardScaler		StandardScaler	
	no	yes	no	yes
19	0.962	0.369	0.959	0.358
190	0.959	0.369	0.956	0.358
1900	0.958	0.369	0.957	0.358

6.4.2.3.1.3 SVM

PCA	**random_state=1**		**stratify=y**	
n_components	StandardScaler		StandardScaler	
	no	yes	no	yes
19	0.946	0.961	0.938	0.955
190	0.949	0.788	0.940	0.777
1900	0.949	0.465	0.941	0.483

6.4.2.3.2 Παράμετροι στην εκτέλεση μεθόδων

6.4.2.3.2.1 Random Forest

Οι δοκιμές σε όλες τις αλλαγές παραμέτρων της μεθόδου έγιναν με PCA n_components = 19.
Παράμετρος n_estimators

			random_state=1		stratify=y	
	StandardScaler		StandardScaler		StandardScaler	
n_estimators	no	yes	no	yes	no	yes
50	0.704	0.948	0.949	0.959	0.954	0.952
100	0.954	0.957	0.959	0.964	0.967	0.962
150	0.954	0.962	0.965	0.969	0.965	0.962
170	0.963	0.965	0.964	0.969	0.964	0.964
200	0.964	0.966	0.968	0.965	0.96	0.965

Οι δοκιμές που ακολουθούν έγιναν με n_estimators=200 και χωρίς StandardScaler.

Παράμετροι Bootstrap και oob_score

bootstrap		**oob_score**	
TRUE	FALSE	TRUE	FALSE
0,967	0,975	0,975	0,975

Δοκιμή συγχρόνως και στα random_state.

	Accuracy
random_state=1	0.975
random_state=2	0.962
random_state=20	0.962

Παράμετρος n_jobs

	Accuracy
2	0.972
3	0.976
5	0.973
10	0.971

Παράμετρος warm_start

TRUE	FALSE
0.971	0.971

Παράμετρος class_weight

	Accuracy
balanced	0.972
balanced_subsample	0.970

Εκτίμηση αποτελεσμάτων με Cross Validation

cross_val_score	Accuracy		
	0.970	n_splits=15	n_repeats=3
	0.970	n_splits=15	n_repeats=5
	0.970	n_splits=19	n_repeats=3
	0.972	n_splits=19	n_repeats=5

Δίνει σταθερότητα

10-Fold Cross validation	0.495

6.4.2.3.2.2 K Nearest Neighbors

Όλες οι δοκιμές έγιναν χωρίς τη χρήση φίλτρων με χρήση Fourier

Παράμετρος n

n=3	n=5	n=7
PCA(1900)	PCA(1900)	PCA(1900)
StadarScaler	StadarScaler	StadarScaler

yes	no	yes	no	yes	no
0.364	0.962	0.370	0.954	0.379	0.951
random_state = 1		random_state = 1		random_state = 1	
0.361	0.967	0.369	0.957	0.361	0.958
stratify=y		stratify=y		stratify=y	
0.378	0.964	0.358	0.954	0.369	0.948

PCA(190)		PCA(190)		PCA(190)	
StadarScaler		StadarScaler		StadarScaler	
yes	no	yes	no	yes	no
0.573	0.958	0.546	0.962	0.539	0.956
random_state = 1		random_state = 1		random_state = 1	
0.595	0.968	0.570	0.958	0.569	0.958
stratify=y		stratify=y		stratify=y	
0.552	0.968	0.530	0.956	0.524	0.951

PCA(19)		PCA(19)		PCA(19)	
StadarScaler		StadarScaler		StadarScaler	
yes	no	yes	no	yes	no
0.965	0.957	0.957	0.963	0.953	0.953
random_state = 1		random_state = 1		random_state = 1	
0.978	0.968	0.968	0.962	0.963	0.958
stratify=y		stratify=y		stratify=y	
0.975	0.967	0.960	0.960	0.955	0.949

Παράμετρος leaf_size. Με n=3 και random_state=1.

	StadarScaler	
leaf_size=	yes	no
10	0.978	0.968
20	0.978	0.968
30	0.978	0.968
50	0.978	0.968

Παράμετρος p. Με n=3 και random_state=1 και StadarScaler.

	Accuracy
p=1	0.980
p=2	0.978

Παράμετρος Metric. Με n=3 και random_state=1 και StadarScaler.

	Accuracy
chebyshev	0.941
minkowski	0.980
manhattan	0.980

Παράμετρος Weights . Με n=3 και random_state=1.

	StadarScaler	
	Yes	no
uniform	0.980	0.975
distance	0.980	0.977

Παράμετρος Algorithm. Με n=3 και random_state=1 και StadarScaler.

	Accuracy
auto	0.980
ball_tree	0.980
kd_tree	0.980
brute	0.980

Εκτίμηση αποτελεσμάτων με Cross Validation

cross_val_score	Accuracy		
	0.976	n_splits=19	n_repeats=5
	0.977		n_repeats=3
	0.976	n_splits=15	n_repeats=5
	0.977		n_repeats=3

10-Fold Cross validation	0.511

6.4.2.3.2.3 SVM

Παράμετρος gamma.

gamma	Accuracy	StandardScaler	PCA
gamma=1.4e-1	0.322	yes	n_components=1900
gamma=1.4e-1	0.322	yes	n_components=190
gamma=1.4e-1	0.974	yes	n_components=19
gamma=1.4e-1	0.322	no	n_components=1900
gamma=1.4e-1	0.322	no	n_components=190
gamma=1.4e-1	0.322	no	n_components=19
gamma=1e-1	0.971	yes	n_components=19
gamma=1.7e-1	0.972	yes	n_components=19
gamma=1.5e-1	0.973	yes	n_components=19
gamma=1.6e-1	0.973	yes	n_components=19
gamma=1.2e-1	0.972	yes	n_components=19
gamma=1.5e-1	0.322	no	n_components=190
gamma=1.5e-1	0.322	no	n_components=190

Παράμετρος C. Με gamma=1.4e-1, StandardScaler και PCA n_components=19.

	Accuracy
C=1.0	0.974
C =1.2	0.977
C =1.3	0.977
C=1.5	0.978

C =1.6	0.978
C =1.7	0.977
C =1.8	0.977

Παράμετρος kernel. Με gamma=1.4e-1, StandardScaler και PCA n_components=19 και C=1.5.

	Accuracy
sigmoid'	0.247
rbf'	0.978
poly'	0.918

Παράμετροι Shrinking και Probability. Με gamma=1.4e-1, StandardScaler και PCA n_components=19 και C=1.5 και kernel= 'rgf'

Shrinking		probability	
TRUE	FALSE	TRUE	FALSE
0.978	0.978	0.978	0.978

Παράμετρος tol. Με gamma=1.4e-1, StandardScaler και PCA n_components=19 και C=1.5 και kernel= 'rgf'

	Accuracy
tol=1e-3	0.978
tol=1.3e-3	0.978
tol=1.5e-3	0.978
tol=1.8e-3	0.978

Εκτίμηση αποτελεσμάτων με Cross Validation

cross_val_score	Accuracy		
	0.978	n_splits=19	n_repeats=5
	0. 978		n_repeats=3
	0. 977	n_splits=15	n_repeats=5
	0. 978		n_repeats=3

10-Fold Cross validation	0.548

6.4.2.4 Εκτέλεση πειραμάτων ανά ζεύγη δυο κλάσεων

Τα πειράματα εκτελέστηκαν με τις παραμέτρους να έχουν τις τιμές που είχαν στα προηγούμενα πειράματα που έδωσαν προηγουμένως τα μεγαλύτερα ποσοστά ανά μέθοδο. Class 0 είναι η Healthy, class 1 η MCI και class 2 η AD.

6.4.2.4.1 Random Forest

	Accuracy	cross_val_score
class 0 - 1	0.976	0.977
class 0 - 2	0.973	0.977
class 1 - 2	0.974	0.976

6.4.2.4.2 K Nearest Neighbors

	Accuracy	cross_val_score
class 0 - 1	0.987	0.989
class 0 - 2	0.984	0.985
class 1 - 2	0.984	0.985

6.4.2.4.3 SVM

	Accuracy	cross_val_score
class 0 - 1	0.987	0.990
class 0 - 2	0.810	0.984
class 1 - 2	0.980	0.985

6.4.2.5 Εκτέλεση πειραμάτων ανά ασθενή

Τα πειράματα εκτελέστηκαν με τις παραμέτρους να έχουν τις τιμές που είχαν στα προηγούμενα πειράματα που έδωσαν προηγουμένως τα μεγαλύτερα ποσοστά ανά μέθοδο. Έγιναν δοκιμές πάραυτα στην προεπεξεργασία δεδομένων ως προς τη χρήση ή μη της StandardScaler και ως PCA με n_ components 19, 190 και 1900. Η κίτρινη επισήμανση υπάρχει στα πειράματα που το ποσοστό ήταν 50-50 και τα αποτελέσματα αυτά δεν λήφθηκαν ως ορθές ταξινομήσεις.

6.4.2.5.1 Random Forest

Χρήση StandardScaler και PCA n_ components = 19

Healthy	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	53	28	69	0,353
2	**121**	26	3	0,806
3	42	91	17	0,28
4	**141**	4	5	0,94
5	21	25	104	0,14
6	38	96	16	0,253
7	**143**	1	6	0,953
8	**121**	6	23	0,806
9	52	98	0	0,346
10	51	25	74	0,34
11	34	92	24	0,226
12	**92**	55	3	0,613
13	60	25	65	0,4
14	17	18	115	0,113
15	3	56	91	0,02
16	2	8	140	0,013
17	5	22	123	0,033

MCI	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	145	5	0	0,033
2	136	2	12	0,013
3	57	4	89	0,026
4	48	**94**	8	0,626
5	24	3	123	0,02
6	1	1	148	0,006
7	19	7	124	0,046
8	28	2	120	0,013
9	129	2	19	0,013
10	36	**65**	49	0,433
11	109	5	36	0,033
12	51	3	96	0,02
13	22	15	113	0,1
14	31	22	97	0,146
15	25	4	121	0,026
16	47	**99**	4	0,66
17	72	58	20	0,386

18	33	37	80	0,22
			MO	**0,381**

18	141	7	2	0,046
			MO	**0,147**

AD	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	27	57	**66**	0,440
2	5	54	**91**	0,606
3	26	83	41	0,273
4	43	1	**106**	0,706
5	16	1	**133**	0,886
6	1	0	**149**	0,993
7	83	3	64	0,426
8	47	63	40	0,266
9	102	20	28	0,186
10	47	19	**84**	0,560
11	14	129	7	0,046
12	89	26	35	0,233
13	81	17	52	0,346
14	5	56	**89**	0,593
15	15	17	**118**	0,786
16	74	17	59	0,393
17	26	15	**109**	0,726
18	44	16	**90**	0,600
			MO	**0,504**

Χωρίς StandardScaler και PCA

Healthy	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	20	15	114	0,133
2	**129**	0	21	0,86
3	72	4	74	0,48
4	35	19	96	0,233
5	26	31	93	0,173
6	41	73	36	0,273
7	0	114	6	0
8	0	141	9	0
9	5	110	35	0,033
10	**71**	21	58	0,473
11	5	131	14	0,033
12	29	92	29	0,193
13	**117**	4	29	0,78
14	**61**	45	44	0,407
15	6	67	77	0,04
16	36	86	28	0,24

MCI	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	18	**122**	10	0,813
2	7	**134**	9	0,893
3	89	33	28	0,22
4	8	**93**	49	0,62
5	114	30	6	0,2
6	120	13	7	0,087
7	106	14	30	0,093
8	19	38	93	0,253
9	105	23	22	0,153
10	64	53	33	0,353
11	108	0	42	0
12	17	**100**	33	0,667
13	61	**67**	22	0,447
14	7	16	127	0,107
15	49	3	98	0,02
16	17	**132**	1	0,88

17	10	25	115	0,067
18	9	115	26	0,06
			MO	**0,249**

17	1	72	77	0,48
18	50	77	23	0,513
			MO	**0,378**

AD	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	89	56	5	0,033
2	69	28	53	0,353
3	33	3	**114**	0,760
4	73	14	63	0,420
5	28	42	**80**	0,533
6	7	68	**75**	0,500
7	14	103	33	0,220
8	73	10	67	0,447
9	3	141	6	0,040
10	71	71	8	0,053
11	63	0	**87**	0,580
12	54	30	**66**	0,440
13	30	64	56	0,373
14	71	22	57	0,380
15	131	11	8	0,053
16	80	18	52	0,347
17	2	129	19	0,127
18	48	99	3	0,020
			MO	**0,316**

Χωρίς StandardScaler, με PCA n_ components = 19

Healthy	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	54	23	73	0,36
2	**120**	27	3	0,8
3	44	95	11	0,293
4	**138**	3	9	0,92
5	20	22	108	0,133
6	26	112	12	0,173
7	**143**	1	6	0,953
8	**121**	6	23	0,807
9	51	99	0	0,34
10	63	20	67	0,42
11	34	94	22	0,227
12	**97**	52	1	0,647
13	54	24	72	0,36
14	14	18	118	0,093

MCI	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	145	5	0	0,033
2	136	2	12	0,013
3	66	3	81	0,02
4	56	**82**	12	0,547
5	28	2	120	0,013
6	1	1	148	0,007
7	14	8	128	0,053
8	28	2	120	0,013
9	129	0	21	0
10	26	**85**	39	0,567
11	113	9	28	0,06
12	55	3	92	0,02
13	31	17	102	0,113
14	37	19	94	0,127

15	3	66	81	0,02
16	0	11	139	0
17	5	22	123	0,033
18	33	49	68	0,22
			MO	**0,378**

15	30	4	116	0,027
16	43	**104**	3	0,693
17	74	57	19	0,38
18	142	6	2	0,04
			MO	**0,151**

AD	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	30	51	**69**	0,460
2	4	61	**85**	0,567
3	24	76	50	0,333
4	43	1	**106**	0,707
5	15	0	**135**	0,900
6	1	0	**149**	0,993
7	84	0	66	0,440
8	41	63	46	0,307
9	94	22	34	0,227
10	52	18	**80**	0,533
11	13	131	6	0,040
12	94	26	30	0,200
13	85	14	51	0,340
14	5	49	**96**	0,640
15	14	13	**123**	0,820
16	78	13	59	0,393
17	24	15	**111**	0,740
18	39	25	**86**	0,573
			MO	**0,512**

Χωρίς StandardScaler, με PCA n_ components = 190

Healthy	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	48	13	89	0,32
2	**116**	32	2	0,773
3	45	90	15	0,3
4	**103**	18	29	0,687
5	17	18	115	0,113
6	21	127	2	0,14
7	**123**	12	15	0,82
8	**98**	41	11	0,653
9	56	93	1	0,373
10	36	32	82	0,24
11	38	90	22	0,253
12	**73**	67	10	0,487
13	**93**	19	38	0,62

MCI	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	150	0	0	0
2	137	2	11	0,013
3	94	2	54	0,013
4	48	**89**	13	0,593
5	76	7	67	0,047
6	21	2	127	0,013
7	11	6	133	0,04
8	17	12	121	0,08
9	86	20	44	0,133
10	29	**79**	42	0,527
11	85	4	61	0,027
12	58	7	85	0,047
13	21	7	122	0,047

14	19	64	67	0,127
15	0	28	122	0
16	2	12	136	0,013
17	3	13	134	0,02
18	21	53	76	0,14
			MO	**0,338**

14	16	36	98	0,24
15	25	5	124	0,033
16	56	**94**	0	0,627
17	89	52	9	0,347
18	139	7	4	0,047
			MO	**0,16**

AD	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	15	87	48	0,320
2	2	51	**97**	0,647
3	36	47	**67**	0,447
4	39	4	**107**	0,713
5	37	4	**109**	0,727
6	3	2	**145**	0,967
7	61	2	**87**	0,580
8	42	40	**68**	0,453
9	89	31	30	0,200
10	91	26	33	0,220
11	22	105	23	0,153
12	95	13	42	0,280
13	37	21	**92**	0,613
14	7	71	**72**	0,480
15	14	65	**71**	0,473
16	61	19	**70**	0,467
17	4	28	**118**	0,787
18	25	36	**89**	0,593
			MO	**0,507**

6.4.2.5.2 K Nearest Neighbors

Χρήση StandardScaler και PCA n_ components = 19

Healthy	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	**68**	43	39	0,453
2	**127**	14	9	0,846
3	**86**	62	2	0,573
4	**132**	14	4	0,88
5	72	73	5	0,48
6	**94**	56	0	0,626
7	**146**	3	1	0,973
8	**137**	3	10	0,913
9	**76**	72	2	0,506
10	**71**	70	9	0,473

MCI	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	83	59	8	0,393
2	99	45	6	0,3
3	111	5	34	0,033
4	67	**71**	12	0,473
5	80	33	37	0,22
6	18	49	83	0,326
7	47	**56**	47	0,373
8	42	20	88	0,133
9	87	35	28	0,233
10	37	**67**	46	0,446

11	33	105	12	0,22
12	23	125	2	0,153
13	38	90	22	0,253
14	49	54	47	0,326
15	19	96	35	0,126
16	3	30	117	0,02
17	31	57	62	0,206
18	41	51	58	0,273
			MO	**0,461**

11	88	50	12	0,333
12	88	43	19	0,286
13	50	**64**	36	0,426
14	88	59	3	0,393
15	33	40	77	0,266
16	93	56	1	0,373
17	27	**120**	3	0,8
18	145	3	2	0,02
			MO	**0,324**

AD	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	15	38	**97**	0,646
2	33	71	46	0,306
3	31	96	23	0,153
4	69	12	**69**	0,460
5	58	36	56	0,373
6	21	2	**127**	0,846
7	60	18	**72**	0,480
8	47	97	6	0,040
9	71	50	29	0,193
10	37	39	**74**	0,493
11	57	76	17	0,113
12	77	46	27	0,180
13	76	37	37	0,246
14	21	75	54	0,360
15	33	40	**77**	0,513
16	60	41	49	0,326
17	38	52	**60**	0,400
18	84	43	23	0,153
			MO	**0,349**

Χωρίς StandardScaler και PCA

Healthy	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	**101**	11	38	0,673
2	**140**	6	4	0,933
3	**76**	59	15	0,507
4	**95**	12	43	0,633
5	59	10	81	0,393
6	42	60	48	0,28
7	**110**	5	35	0,733
8	**141**	6	3	0,94
9	**120**	13	17	0,8

MCI	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	135	2	13	0,013
2	76	48	26	0,32
3	118	12	20	0,08
4	70	62	18	0,413
5	50	30	70	0,2
6	8	27	115	0,18
7	49	4	97	0,027
8	22	33	95	0,22
9	17	42	91	0,28

10	**124**	0	26	0,827
11	**122**	15	13	0,813
12	**133**	3	14	0,887
13	**121**	4	25	0,807
14	16	15	119	0,107
15	8	45	97	0,053
16	11	14	125	0,073
17	8	19	123	0,053
18	36	82	32	0,24
			MO	**0,542**

10	62	13	75	0,087
11	10	0	140	0
12	46	**86**	18	0,573
13	105	14	31	0,093
14	134	4	12	0,027
15	79	21	50	0,14
16	66	57	27	0,38
17	141	1	8	0,007
18	131	6	13	0,04
			MO	**0,171**

AD	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	122	5	23	0,153
2	122	5	23	0,153
3	94	8	48	0,320
4	87	7	56	0,373
5	129	2	19	0,127
6	58	19	**73**	0,487
7	75	1	**74**	0,493
8	17	12	**121**	0,807
9	65	36	49	0,327
10	52	33	65	0,433
11	20	38	**92**	0,613
12	30	28	**92**	0,613
13	22	24	**104**	0,693
14	58	41	51	0,340
15	65	72	13	0,087
16	109	10	31	0,207
17	90	27	33	0,220
18	55	7	**88**	0,587
			MO	**0,391**

Χωρίς StandardScaler, με PCA n_ components = 19

Healthy	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	**72**	52	26	0,48
2	**136**	8	6	0,907
3	**99**	48	3	0,66
4	**130**	17	3	0,867
5	**79**	61	10	0,527
6	61	89	0	0,407
7	**146**	2	2	0,973
8	**126**	5	19	0,84

MCI	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	78	71	1	0,473
2	81	69	0	0,46
3	126	7	17	0,047
4	61	**83**	6	0,553
5	66	36	48	0,24
6	27	**66**	57	0,44
7	44	38	68	0,253
8	57	19	74	0,127

9	13	137		0,087
10	**86**	61	3	0,573
11	47	89	14	0,313
12	16	134	0	0,107
13	28	84	38	0,187
14	18	80	52	0,12
15	26	97	27	0,173
16	5	39	106	0,033
17	29	56	65	0,193
18	48	60	42	0,32
			MO	**0,431**

9	97	35	18	0,233
10	21	60	69	0,4
11	91	40	19	0,267
12	68	50	32	0,333
13	59	**73**	18	0,487
14	85	61	4	0,407
15	58	20	72	0,133
16	75	75	0	0,5
17	21	**129**	0	0,86
18	144	5	1	0,033
			MO	**0,347**

AD	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	45	36	**69**	0,460
2	38	74	38	0,253
3	45	86	19	0,127
4	77	7	66	0,440
5	51	9	**90**	0,600
6	14	0	**136**	0,907
7	26	13	**111**	0,740
8	38	104	8	0,053
9	80	44	26	0,173
10	40	42	**68**	0,453
11	45	85	20	0,133
12	77	41	32	0,213
13	71	25	54	0,360
14	23	76	51	0,340
15	21	52	**77**	0,513
16	50	45	**55**	0,367
17	45	41	**64**	0,427
18	95	23	32	0,213
			MO	**0,376**

Χωρίς StandardScaler, με PCA n_ components = 190

Healthy	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	**66**	52	32	0,44
2	**123**	10	17	0,82
3	69	79	2	0,46
4	**120**	28	2	0,8
5	66	71	13	0,44
6	35	115	0	0,233
7	**147**	2	1	0,98

MCI	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	133	16	1	0,106
2	133	16	1	0,106
3	75	47	28	0,313
4	36	**114**	0	0,76
5	51	46	53	0,306
6	13	48	89	0,32
7	43	42	65	0,28

8	**125**	1	24	0,833
9	**100**	49	1	0,666
10	**90**	58	2	0,6
11	46	87	17	0,306
12	21	129	0	0,14
13	49	27	74	0,326
14	30	57	63	0,2
15	7	113	30	0,046
16	3	38	109	0,02
17	39	63	48	0,26
18	60	70	20	0,4
			MO	**0,443**

8	41	28	81	0,186
9	21	**106**	23	0,706
10	19	**100**	31	0,666
11	101	21	28	0,14
12	80	31	39	0,206
13	75	49	26	0,327
14	54	**88**	8	0,587
15	26	44	80	0,293
16	97	53	0	0,353
17	44	**106**	0	0,707
18	142	7	1	0,047
			MO	**0,356**

AD	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	40	28	**82**	0,547
2	11	68	**71**	0,473
3	66	70	14	0,093
4	70	14	66	0,440
5	43	13	**94**	0,627
6	7	0	**143**	0,953
7	81	7	62	0,413
8	22	107	21	0,140
9	89	42	19	0,127
10	29	50	**71**	0,473
11	49	85	16	0,107
12	66	55	29	0,193
13	32	31	**87**	0,580
14	8	87	55	0,367
15	9	98	43	0,287
16	56	39	55	0,367
17	18	67	65	0,433
18	85	29	36	0,240
			MO	**0,381**

6.4.2.5.3 SVM

Χρήση StandardScaler και PCA n_ components = 19

Healthy	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	16	33	101	0,106
2	**107**	40	3	0,713
3	**89**	49	12	0,596

MCI	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	139	11	0	0,073
2	107	32	11	0.213
3	26	10	104	0,066

4	**114**	3	33	0,76		4	33	**114**	3	0,76
5	38	51	61	0,253		5	44	3	103	0,02
6	**98**	52	0	0,653		6	0	4	146	0,026
7	**143**	0	7	0,953		7	22	33	95	0,22
8	**120**	1	29	0,8		8	25	0	125	0
9	53	95	2	0,353		9	125	4	21	0,026
10	**106**	15	29	0,706		10	30	**89**	31	0,593
11	17	110	23	0,113		11	110	5	35	0,033
12	**144**	2	4	0,96		12	20	12	118	0,08
13	**58**	56	36	0,386		13	11	32	107	0,213
14	16	27	107	0,106		14	129	6	15	0,04
15	3	59	88	0,02		15	3	1	146	0,006
16	1	3	146	0,006		16	33	**115**	2	0,766
17	9	17	124	0,06		17	29	**118**	3	0,786
18	11	18	121	0,073		18	145	4	1	0,026
			MO	**0,423**					**MO**	**0,22**

AD	Κατάταξη Segment σε κλάση			Accuracy
Ασθενής	Healthy	MCI	AD	test
1	1	19	**130**	0,866
2	13	44	**93**	0,620
3	51	65	34	0,226
4	57	1	**92**	0,613
5	32	3	**115**	0,766
6	2	0	**148**	0,986
7	45	0	**105**	0,700
8	33	46	**71**	0,473
9	74	34	41	0,273
10	29	22	**99**	0,660
11	22	65	63	0,420
12	80	25	45	0,300
13	20	21	**109**	0,726
14	3	10	**137**	0,913
15	7	14	**129**	0,860
16	34	18	**98**	0,653
17	12	42	**96**	0,640
18	67	18	65	0,433
		MO	**0,618**	

Στις περιπτώσεις χωρίς StandardScaler και χωρίς PCA, χωρίς StandardScaler, με PCA n_ components = 19 χωρίς StandardScaler, με PCA n_ components = 190 το αποτέλεσμα ήταν 0% σε όλα τα πειράματα.

6.4.2.6 Εκτέλεση πειραμάτων ανά ασθενή ανά ζεύγη δυο κλάσεων

Τα πειράματα εκτελέστηκαν με τις παραμέτρους να έχουν τις τιμές που είχαν στα προηγούμενα πειράματα που έδωσαν προηγουμένως τα μεγαλύτερα ποσοστά ανά μέθοδο. Class 0 είναι η Healthy, class 1 η MCI και class 2 η AD.

6.4.2.6.1 Random Forest

Χρήση StandardScaler και PCA n_ components = 19

Ζεύγος Healthy – MCI

Healthy	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	MCI	test
1	**97**	53	0,646
2	**119**	31	0,793
3	60	90	0,400
4	**143**	7	0,953
5	**90**	60	0,600
6	41	109	0,273
7	**146**	4	0,973
8	**134**	16	0,893
9	53	97	0,353
10	**124**	26	0,826
11	38	112	0,253
12	**121**	29	0,806
13	**95**	55	0,633
14	**76**	74	0,506
15	9	141	0,060
16	37	113	0,246
17	56	94	0,373
18	68	82	0,453
		MO	**0,558**

MCI	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	MCI	test
1	146	4	0,026
2	136	14	0,093
3	131	19	0,126
4	59	**91**	0,606
5	128	22	0,146
6	85	65	0,433
7	109	41	0,273
8	117	33	0,220
9	150	0	0,000
10	30	**120**	0,800
11	145	5	0,033
12	83	67	0,446
13	61	**89**	0,593
14	115	35	0,233
15	142	8	0,053
16	44	**106**	0,706
17	97	53	0,353
18	141	9	0,060
		MO	**0,289**

Ζεύγος Healthy – AD

Healthy	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	AD	test
1	53	97	0,353
2	**142**	8	0,946
3	**86**	64	0,573
4	**139**	11	0,926
5	49	101	0,326
6	42	108	0,280
7	**143**	7	0,953

AD	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	AD	test
1	31	**119**	0,793
2	16	**134**	0,893
3	84	66	0,440
4	47	**103**	0,686
5	12	**138**	0,920
6	0	**150**	1,000
7	84	66	0,440

8	**84**	66	0,853
9	**141**	9	0,940
10	68	82	0,453
11	74	76	0,493
12	**148**	2	0,986
13	**102**	48	0,680
14	23	127	0,153
15	3	147	0,020
16	1	149	0,006
17	8	142	0,053
18	40	110	0,266
		MO	**0,514**

8	73	**77**	0,513
9	103	47	0,313
10	47	**103**	0,686
11	110	40	0,266
12	111	39	0,260
13	75	75	0,500
14	23	**127**	0,846
15	21	**129**	0,860
16	68	82	0,546
17	26	**124**	0,826
18	42	**108**	0,720
		MO	**0,639**

Ζεύγος MCI – AD

MCI	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	MCI	AD	test
1	**127**	23	0,846
2	**127**	23	0,846
3	48	102	0,320
4	**139**	11	0,926
5	7	142	0,046
6	3	147	0,020
7	13	137	0,086
8	1	141	0,060
9	**89**	61	0,593
10	**76**	74	0,506
11	25	125	0,166
12	62	88	0,413
13	15	135	0,100
14	33	117	0,220
15	18	132	0,120
16	**143**	7	0,953
17	**122**	28	0,813
18	**130**	20	0,866
		MO	**0,439**

AD	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	MCI	AD	test
1	78	72	0,480
2	43	**107**	0,713
3	101	49	0,326
4	8	**142**	0,946
5	9	**141**	0,940
6	0	**150**	1,000
7	26	**124**	0,826
8	56	**94**	0,626
9	73	**77**	0,513
10	50	**100**	0,666
11	142	8	0,053
12	54	**96**	0,640
13	33	**117**	0,780
14	46	**104**	0,693
15	27	**123**	0,820
16	34	**116**	0,773
17	29	**121**	0,806
18	35	**114**	0,766
		MO	**0,687**

Χωρίς StandardScaler και PCA

Ζεύγος Healthy – MCI

Healthy	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	MCI	test
1	**106**	44	0,707
2	**150**	0	1,000
3	**133**	17	0,887

MCI	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	MCI	test
1	31	**119**	0,207
2	16	**134**	0,107
3	106	44	0,707

4	**101**	49	0,673
5	**92**	58	0,613
6	**83**	67	0,553
7	0	150	0,000
8	0	150	0,000
9	10	140	0,067
10	**117**	33	0,780
11	8	142	0,053
12	28	122	0,187
13	**144**	6	0,960
14	**94**	56	0,627
15	26	124	0,173
16	63	87	0,420
17	**78**	72	0,520
18	4	146	0,027
		MO	**0,458**

4	5	**145**	0,033
5	120	30	0,800
6	142	8	0,947
7	140	10	0,933
8	50	**100**	0,333
9	118	32	0,787
10	70	**80**	0,467
11	149	1	0,993
12	26	**124**	0,173
13	77	73	0,513
14	43	**107**	0,287
15	124	16	0,827
16	31	**119**	0,207
17	0	**150**	0,000
18	53	97	0,353
		MO	**0,482**

Ζεύγος Healthy – AD

Healthy	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	AD	test
1	35	115	0,233
2	**132**	18	0,880
3	**80**	70	0,533
4	40	110	0,267
5	49	101	0,327
6	**95**	55	0,633
7	23	127	0,153
8	38	112	0,253
9	35	115	0,233
10	74	76	0,493
11	75	75	0,500
12	**110**	40	0,733
13	**111**	39	0,740
14	67	83	0,447
15	17	133	0,113
16	**98**	52	0,653
17	25	125	0,167
18	44	106	0,293
		MO	**0,000**

AD	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	AD	test
1	142	8	0,053
2	103	47	0,313
3	42	**108**	0,720
4	54	**96**	0,640
5	60	**90**	0,600
6	16	**134**	0,893
7	68	**82**	0,547
8	86	64	0,427
9	44	**106**	0,707
10	148	2	0,013
11	46	**104**	0,693
12	45	**105**	0,700
13	47	**103**	0,687
14	79	71	0,473
15	138	12	0,080
16	101	49	0,327
17	13	137	0,913
18	149	1	0,007
		MO	**0,489**

Ζεύγος MCI – AD

MCI	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	MCI	AD	test
1	**148**	2	0,987

AD	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	MCI	AD	test
1	145	5	0,033

2	**146**	4	0,973
3	**104**	46	0,693
4	**84**	66	0,560
5	**132**	18	0,880
6	**117**	33	0,780
7	48	102	0,320
8	51	99	0,340
9	43	107	0,287
10	**103**	47	0,687
11	0	150	0,000
12	**120**	30	0,800
13	**85**	65	0,567
14	31	119	0,207
15	3	147	0,020
16	**150**	0	1,000
17	**81**	69	0,540
18	**129**	21	0,860
		MO	**0,583**

2	41	**109**	0,727
3	5	**145**	0,967
4	31	**119**	0,793
5	72	**78**	0,520
6	50	**100**	0,667
7	116	34	0,227
8	41	**109**	0,727
9	146	4	0,027
10	142	8	0,053
11	2	**148**	0,987
12	16	**134**	0,893
13	71	**79**	0,527
14	31	**119**	0,793
15	98	52	0,347
16	53	**97**	0,647
17	131	19	0,127
18	148	2	0,013
		MO	**0,504**

Χωρίς StandardScaler, με PCA n_ components = 19

Ζεύγος Healthy – MCI

Healthy	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	MCI	test
1	**97**	53	0,647
2	**119**	31	0,793
3	65	85	0,433
4	**142**	8	0,947
5	**84**	66	0,560
6	41	109	0,273
7	**148**	2	0,987
8	**137**	13	0,913
9	51	99	0,340
10	**114**	36	0,760
11	40	110	0,267
12	**133**	17	0,887
13	**99**	51	0,660
14	**80**	70	0,533
15	9	141	0,060
16	26	124	0,173
17	53	97	0,353
18	64	86	0,427
		MO	**0,556**

MCI	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	MCI	test
1	147	3	0,980
2	136	14	0,907
3	135	15	0,900
4	66	**84**	0,440
5	128	22	0,853
6	78	72	0,520
7	112	38	0,747
8	120	3	0,800
9	150	0	1,000
10	48	**102**	0,320
11	146	4	0,973
12	82	68	0,547
13	65	**85**	0,433
14	124	26	0,827
15	139	11	0,927
16	51	**99**	0,340
17	93	57	0,620
18	142	8	0,947
		MO	**0,727**

Ζεύγος Healthy – AD

Healthy	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	AD	test
1	56	94	0,373
2	**141**	9	0,940
3	72	78	0,480
4	**143**	7	0,953
5	43	107	0,287
6	48	102	0,320
7	**142**	8	0,947
8	**119**	31	0,793
9	**136**	14	0,907
10	60	90	0,400
11	**79**	71	0,527
12	**148**	2	0,987
13	**103**	47	0,687
14	18	132	0,120
15	3	147	0,020
16	0	150	0,000
17	7	143	0,047
18	37	113	0,247
		MO	**0,502**

AD	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	AD	test
1	26	**124**	0,827
2	16	**134**	0,893
3	88	62	0,413
4	42	**108**	0,720
5	13	**137**	0,913
6	0	**150**	1,000
7	90	60	0,400
8	68	**82**	0,547
9	102	48	0,320
10	48	**102**	0,680
11	108	42	0,280
12	111	39	0,260
13	81	69	0,460
14	28	**122**	0,813
15	20	**130**	0,867
16	55	**95**	0,633
17	21	**129**	0,860
18	44	**106**	0,707
		MO	**0,644**

Ζεύγος MCI – AD

MCI	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	MCI	AD	test
1	**125**	25	0,833
2	**123**	27	0,820
3	43	107	0,287
4	**140**	10	0,933
5	7	143	0,047
6	2	148	0,013
7	12	138	0,080
8	9	141	0,060
9	**91**	59	0,607
10	75	75	0,500
11	28	122	0,187
12	65	85	0,433
13	7	143	0,047
14	30	120	0,200
15	18	132	0,120
16	**143**	7	0,953
17	**124**	26	0,827

AD	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	MCI	AD	test
1	67	**83**	0,553
2	52	**98**	0,653
3	100	50	0,333
4	8	**142**	0,947
5	4	**146**	0,973
6	5	**145**	0,967
7	13	**137**	0,913
8	70	**80**	0,533
9	77	73	0,487
10	44	**106**	0,707
11	143	7	0,047
12	61	**89**	0,593
13	34	**116**	0,773
14	40	**110**	0,733
15	22	**128**	0,853
16	35	**115**	0,767
17	34	**116**	0,773

18	**128**	22	0,853	18	38	**112**	0,747
		MO	**0,433**			**MO**	**0,686**

Χωρίς StandardScaler, με PCA n_ components = 190

Ζεύγος Healthy – MCI

Healthy	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	MCI	test
1	**103**	47	0,687
2	**117**	33	0,780
3	67	83	0,447
4	**112**	38	0,747
5	**101**	49	0,673
6	22	128	0,147
7	**124**	26	0,827
8	**116**	34	0,773
9	56	94	0,373
10	**86**	64	0,573
11	41	109	0,273
12	**119**	31	0,793
13	**120**	30	0,800
14	60	90	0,400
15	15	135	0,100
16	57	93	0,380
17	67	83	0,447
18	59	91	0,393
		MO	**0,534**

MCI	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	MCI	test
1	149	1	0,007
2	140	10	0,067
3	141	9	0,060
4	69	**81**	0,540
5	130	20	0,133
6	79	71	0,473
7	102	48	0,320
8	89	61	0,407
9	144	6	0,040
10	58	**92**	0,613
11	144	6	0,040
12	71	**79**	0,527
13	50	**100**	0,667
14	108	42	0,280
15	138	12	0,080
16	51	**99**	0,660
17	102	48	0,320
18	141	9	0,060
		MO	**0,294**

Ζεύγος Healthy – AD

Healthy	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	AD	test
1	39	111	0,260
2	**146**	4	0,973
3	**109**	41	0,727
4	**120**	30	0,800
5	31	119	0,207
6	48	102	0,320
7	**140**	10	0,933
8	**116**	34	0,773
9	**114**	36	0,760
10	47	103	0,313
11	**80**	70	0,533
12	**146**	4	0,973
13	**119**	31	0,793

AD	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	AD	test
1	20	**130**	0,867
2	12	**138**	0,920
3	77	73	0,487
4	41	**109**	0,727
5	29	**121**	0,807
6	1	**149**	0,993
7	65	**85**	0,567
8	72	**78**	0,520
9	109	41	0,273
10	131	19	0,127
11	72	**78**	0,520
12	112	38	0,253
13	44	**106**	0,707

14	15	135	0,100
15	0	150	0,000
16	3	147	0,020
17	2	148	0,013
18	28	122	0,187
		MO	**0,483**

14	22	**128**	0,853
15	33	**117**	0,780
16	77	73	0,487
17	13	**137**	0,913
18	31	**119**	0,793
		MO	**0,644**

Ζεύγος MCI – AD

MCI	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	MCI	AD	test
1	**117**	33	0,780
2	**86**	64	0,573
3	61	89	0,407
4	**139**	11	0,927
5	15	135	0,100
6	6	144	0,040
7	8	142	0,053
8	16	134	0,107
9	40	110	0,267
10	**90**	60	0,600
11	24	126	0,160
12	72	78	0,480
13	9	141	0,060
14	46	104	0,307
15	26	124	0,173
16	**147**	3	0,980
17	**131**	19	0,873
18	**125**	25	0,833
		MO	**0,429**

AD	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	MCI	AD	test
1	81	69	0,460
2	53	**97**	0,647
3	73	**77**	0,513
4	22	**128**	0,853
5	3	**147**	0,980
6	4	**146**	0,973
7	25	**125**	0,833
8	15	**135**	0,900
9	88	62	0,413
10	89	61	0,407
11	96	54	0,360
12	22	**128**	0,853
13	27	**123**	0,820
14	62	**88**	0,587
15	83	67	0,447
16	36	**114**	0,760
17	17	**133**	0,887
18	31	**119**	0,793
		MO	**0,694**

6.4.2.6.2 K Nearest Neighbors

Χρήση StandardScaler και PCA n_ components = 19

Ζεύγος Healthy – MCI

Healthy	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	MCI	test
1	**94**	56	0,626
2	**135**	15	0,900
3	**88**	62	0,586
4	**134**	16	0,893
5	74	76	0,493
6	**93**	57	0,620
7	**147**	3	0,980

MCI	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	MCI	test
1	89	61	0,406
2	101	49	0,326
3	135	15	0,100
4	79	71	0,473
5	101	49	0,326
6	68	**82**	0,546
7	65	**85**	0,566

8	**145**	5	0,966
9	**78**	72	0,520
10	**78**	72	0,520
11	38	112	0,253
12	25	125	0,166
13	43	107	0,286
14	54	96	0,360
15	35	115	0,233
16	45	105	0,300
17	53	94	0,373
18	66	84	0,440
		MO	**0,529**

8	93	57	0,380
9	115	35	0,233
10	65	**85**	0,566
11	99	51	0,340
12	99	51	0,340
13	60	**90**	0,600
14	91	59	0,393
15	75	75	0,500
16	93	57	0,380
17	28	**122**	0,813
18	145	5	0,033
		MO	**0,407**

Ζεύγος Healthy – AD

Healthy	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	AD	test
1	**104**	46	0,693
2	**142**	8	0,946
3	**148**	2	0,986
4	**143**	7	0,953
5	**136**	14	0,906
6	**146**	4	0,973
7	**149**	1	0,993
8	**140**	10	0,933
9	**149**	1	0,993
10	**139**	11	0,926
11	**120**	30	0,800
12	**148**	2	0,986
13	**114**	36	0,760
14	66	84	0,440
15	69	81	0,460
16	4	146	0,026
17	55	95	0,366
18	64	86	0,426
		MO	**0,754**

AD	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	AD	test
1	34	**116**	0,733
2	72	**78**	0,520
3	114	36	0,240
4	81	69	0,460
5	79	71	0,473
6	23	**127**	0,846
7	79	71	0,473
8	115	35	0,233
9	116	34	0,226
10	53	**97**	0,646
11	115	35	0,233
12	119	31	0,206
13	90	60	0,400
14	63	**87**	0,580
15	52	**98**	0,653
16	95	55	0,366
17	67	**83**	0,553
18	119	31	0,206
		MO	**0,447**

Ζεύγος MCI – AD

MCI	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	MCI	AD	test
1	**134**	16	0,893
2	**145**	5	0,966
3	**83**	67	0,553
4	**134**	16	0,893
5	69	81	0,460

AD	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	MCI	AD	test
1	42	**108**	0,720
2	93	57	0,380
3	124	26	0,173
4	39	**111**	0,740
5	57	**93**	0,620

6	51	99	0,340
7	**86**	64	0,573
8	41	109	0,273
9	**83**	67	0,553
10	**95**	55	0,633
11	**116**	34	0,773
12	**113**	37	0,753
13	**113**	37	0,753
14	**125**	25	0,833
15	61	89	0,406
16	**148**	2	0,986
17	**146**	4	0,973
18	**132**	18	0,880
		MO	**0,694**

6	2	**148**	0,986
7	54	**96**	0,640
8	129	21	0,140
9	112	38	0,253
10	62	**88**	0,586
11	129	21	0,140
12	101	49	0,326
13	81	69	0,460
14	89	61	0,406
15	55	**95**	0,633
16	72	**78**	0,520
17	79	71	0,473
18	102	48	0,320
		MO	**0,473**

Χωρίς StandardScaler και PCA

Ζεύγος Healthy – MCI

Healthy	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	MCI	test
1	**135**	15	0,900
2	**144**	6	0,960
3	**81**	69	0,540
4	**132**	18	0,880
5	**119**	31	0,793
6	41	109	0,273
7	**139**	11	0,926
8	**143**	7	0,953
9	**139**	11	0,926
10	**148**	2	0,986
11	**122**	28	0,813
12	**141**	9	0,940
13	**142**	8	0,946
14	**93**	57	0,620
15	37	113	0,246
16	44	106	0,293
17	29	121	0,193
18	48	102	0,320
		MO	**0,695**

MCI	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	MCI	test
1	149	1	0,006
2	86	64	0,426
3	133	17	0,113
4	82	68	0,453
5	81	69	0,540
6	35	**115**	0,766
7	121	29	0,193
8	72	**78**	0,520
9	40	**110**	0,733
10	124	26	0,173
11	111	39	0,260
12	49	**101**	0,673
13	119	31	0,206
14	139	11	0,073
15	100	50	0,333
16	68	**82**	0,546
17	149	1	0,006
18	141	9	0,060
		MO	**0,338**

Ζεύγος Healthy – AD

Healthy	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	AD	test
1	**137**	13	0,913

AD	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	AD	test
1	96	54	0,360

2	**138**	12	0,920
3	**85**	65	0,566
4	**77**	73	0,513
5	**101**	49	0,673
6	**109**	41	0,726
7	**133**	17	0,886
8	**128**	22	0,853
9	**115**	35	0,766
10	**147**	3	0,980
11	**127**	23	0,846
12	**150**	0	1,000
13	**82**	68	0,546
14	**93**	57	0,620
15	29	121	0,193
16	26	124	0,173
17	**87**	63	0,580
18	**146**	4	0,973
		MO	**0,707**

2	94	56	0,373
3	56	**94**	0,626
4	140	10	0,066
5	17	**133**	0,886
6	3	**147**	0,980
7	75	75	0,500
8	65	**85**	0,566
9	123	27	0,180
10	63	**87**	0,580
11	75	75	0,500
12	98	52	0,346
13	24	**126**	0,840
14	84	66	0,440
15	93	57	0,380
16	46	**104**	0,693
17	72	**78**	0,520
18	127	23	0,153
		MO	**0,499**

Ζεύγος MCI - AD

MCI	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	MCI	AD	test
1	12	138	0,080
2	75	75	0,500
3	14	136	0,093
4	**101**	49	0,673
5	42	108	0,280
6	28	122	0,186
7	5	145	0,033
8	37	113	0,246
9	47	103	0,313
10	23	127	0,153
11	0	150	0,000
12	**116**	34	0,773
13	46	104	0,306
14	29	121	0,193
15	41	109	0,273
16	**91**	59	0,606
17	7	143	0,046
18	23	127	0,153
		MO	**0,273**

AD	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	MCI	AD	test
1	9	**141**	0,940
2	39	**111**	0,740
3	17	**133**	0,886
4	45	**105**	0,700
5	8	**142**	0,946
6	27	**123**	0,820
7	1	**149**	0,993
8	10	**140**	0,933
9	55	**95**	0,633
10	46	**104**	0,693
11	44	**106**	0,706
12	34	**116**	0,773
13	32	**118**	0,786
14	80	70	0,466
15	131	19	0,126
16	28	**122**	0,813
17	42	**108**	0,720
18	21	**129**	0,860
		MO	**0,752**

Χωρίς StandardScaler, με PCA n_ components = 19

Ζεύγος Healthy – MCI

Healthy	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	MCI	test
1	**86**	64	0,573
2	**140**	10	0,933
3	**102**	48	0,680
4	**132**	18	0,880
5	**80**	70	0,533
6	61	89	0,406
7	**149**	1	0,993
8	**146**	4	0,973
9	13	137	0,086
10	**88**	62	0,586
11	50	100	0,333
12	16	134	0,106
13	39	111	0,260
14	10	110	0,266
15	33	117	0,220
16	35	115	0,233
17	60	90	0,400
18	52	98	0,346
		MO	**0,489**

MCI	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	MCI	test
1	79	71	0,473
2	82	68	0,453
3	137	13	0,086
4	64	**86**	0,573
5	101	49	0,326
6	57	**93**	0,620
7	96	54	0,360
8	99	51	0,340
9	112	38	0,253
10	39	**111**	0,740
11	105	45	0,300
12	77	73	0,486
13	69	**81**	0,540
14	89	61	0,406
15	104	46	0,306
16	75	75	0,500
17	21	**129**	0,860
18	144	6	0,040
		MO	**0,426**

Ζεύγος Healthy - AD

Healthy	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	AD	test
1	**109**	41	0,726
2	**145**	5	0,966
3	**147**	3	0,980
4	**146**	4	0,973
5	**127**	23	0,846
6	**145**	5	0,966
7	**147**	3	0,980
8	**131**	19	0,873
9	**150**	0	1,000
10	**145**	5	0,966
11	**123**	27	0,820
12	**149**	1	0,993
13	**81**	69	0,540
14	46	104	0,306
15	73	77	0,486

AD	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	AD	test
1	74	**76**	0,506
2	76	74	0,493
3	115	35	0,233
4	82	68	0,453
5	57	**93**	0,620
6	14	**136**	0,906
7	33	**117**	0,780
8	129	21	0,140
9	114	36	0,240
10	61	**89**	0,593
11	113	37	0,246
12	104	46	0,306
13	84	66	0,440
14	73	**77**	0,513
15	39	**111**	0,740

16	12	138	0,080
17	57	93	0,380
18	**80**	70	0,533
		MO	**0,745**

16	90	60	0,400
17	62	**88**	0,586
18	112	38	0,253
		MO	**0,469**

Ζεύγος MCI - AD

MCI	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	MCI	AD	test
1	**149**	1	0,993
2	**147**	3	0,980
3	**111**	39	0,740
4	**135**	15	0,900
5	52	98	0,346
6	**76**	74	0,506
7	61	89	0,406
8	37	113	0,246
9	**104**	46	0,693
10	**79**	71	0,526
11	**95**	55	0,633
12	**109**	41	0,726
13	**125**	25	0,833
14	**122**	28	0,813
15	52	98	0,346
16	**150**	0	1,000
17	**149**	1	0,993
18	**138**	12	0,920
		MO	**0,700**

AD	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	MCI	AD	test
1	69	**81**	0,540
2	104	46	0,306
3	120	30	0,200
4	36	**114**	0,760
5	33	**117**	0,780
6	1	**149**	0,993
7	18	**132**	0,880
8	136	14	0,093
9	100	50	0,333
10	66	**84**	0,560
11	123	27	0,180
12	84	66	0,440
13	67	**83**	0,553
14	93	57	0,380
15	63	**87**	0,580
16	76	74	0,493
17	61	**89**	0,593
18	76	74	0,493
		MO	**0,509**

Χωρίς StandardScaler, με PCA n_ components = 190

Ζεύγος Healthy – MCI

Healthy	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	MCI	test
1	**78**	72	0,520
2	**139**	11	0,926
3	71	79	0,473
4	**119**	31	0,793
5	71	79	0,473
6	35	115	0,233
7	**148**	2	0,986
8	**149**	1	0,933
9	**100**	50	0,666
10	**92**	58	0,613
11	51	99	0,340
12	20	130	0,133

MCI	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	MCI	test
1	136	14	0,093
2	135	15	0,100
3	83	67	0,446
4	37	**113**	0,753
5	90	60	0,400
6	65	**85**	0,566
7	69	**81**	0,540
8	83	67	0,446
9	34	**116**	0,773
10	25	**125**	0,833
11	123	27	0,180
12	105	45	0,300

13	**78**	72	0,520
14	55	95	0,366
15	12	138	0,080
16	28	122	0,186
17	54	96	0,360
18	68	82	0,453
		MO	**0,503**

13	95	55	0,366
14	59	**91**	0,606
15	71	**79**	0,526
16	94	56	0,373
17	43	**107**	0,713
18	143	7	0,046
		MO	**0,448**

Ζεύγος Healthy – AD

Healthy	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	AD	τεστ
1	**111**	39	0,740
2	**132**	18	0,880
3	**147**	3	0,980
4	**143**	7	0,953
5	**109**	41	0,726
6	**136**	14	0,906
7	**149**	1	0,993
8	**124**	26	0,826
9	**149**	1	0,993
10	**148**	2	0,986
11	**111**	39	0,740
12	**149**	1	0,993
13	64	86	0,426
14	57	93	0,380
15	30	120	0,200
16	10	140	0,066
17	70	80	0,466
18	**105**	45	0,700
		MO	**0,720**

AD	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	AD	test
1	60	**90**	0,600
2	48	**102**	0,680
3	121	29	0,193
4	87	63	0,420
5	49	**101**	0,673
6	8	**142**	0,946
7	83	67	0,446
8	95	55	0,366
9	111	39	0,260
10	55	**95**	0,633
11	120	30	0,200
12	98	52	0,346
13	52	**98**	0,653
14	59	**91**	0,606
15	58	**92**	0,613
16	89	61	0,406
17	48	**102**	0,680
18	111	39	0,260
		MO	**0,499**

Ζεύγος MCI – AD

MCI	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	MCI	AD	test
1	**146**	4	0,973
2	**132**	18	0,880
3	**110**	40	0,733
4	**150**	0	1,000
5	62	88	0,413
6	51	99	0,340
7	70	80	0,466
8	47	103	0,313
9	**118**	32	0,786
10	**119**	31	0,793

AD	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	MCI	AD	test
1	57	**93**	0,620
2	75	75	0,500
3	120	30	0,200
4	62	**88**	0,586
5	27	**123**	0,820
6	1	**149**	0,993
7	26	**124**	0,826
8	122	28	0,186
9	108	42	0,280
10	64	**86**	0,573

11	59	91	0,393
12	**89**	61	0,593
13	**102**	48	0,680
14	**125**	25	0,833
15	57	93	0,380
16	**142**	8	0,946
17	**150**	0	1,000
18	**138**	12	0,920
		MO	**0,691**

11	120	30	0,200
12	96	54	0,360
13	44	**106**	0,706
14	93	57	0,380
15	106	44	0,293
16	72	**78**	0,520
17	77	73	0,486
18	73	**77**	0,513
		MO	**0,502**

6.4.2.6.3 SVM

Χρήση StandardScaler και PCA n_ components = 19

Ζεύγος Healthy – MCI

Healthy	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	MCI	test
1	39	111	0,260
2	**110**	4	0,730
3	**97**	53	0,646
4	**138**	12	0,920
5	**85**	65	0,560
6	**98**	52	0,653
7	**150**	0	1,000
8	**146**	4	0,973
9	54	96	0,360
10	**120**	30	0,800
11	22	128	0,146
12	**148**	2	0,986
13	65	85	0,433
14	62	88	0,413
15	6	144	0,040
16	15	135	0,100
17	53	97	0,353
18	33	117	0,220
		MO	**0,533**

MCI	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	MCI	test
1	138	12	0,08
2	109	41	0,273
3	123	27	0,180
4	34	**116**	0,733
5	136	14	0,093
6	54	**96**	0,640
7	72	**78**	0,520
8	142	8	0,053
9	141	9	0,060
10	39	**111**	0,740
11	142	8	0,053
12	102	48	0,320
13	45	**105**	0,700
14	144	6	0,040
15	131	19	0,126
16	33	**117**	0,780
17	28	**122**	0,813
18	145	5	0,033
		MO	**0,347**

Ζεύγος Healthy – AD

Healthy	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	AD	test
1	29	121	0,193
2	**143**	7	0,953
3	**136**	14	0,906
4	**117**	33	0,780

AD	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	Healthy	AD	test
1	1	**149**	0,993
2	33	**117**	0,780
3	105	45	0,300
4	58	**92**	0,613

5	70	80	0,466
6	**143**	7	0,953
7	**143**	7	0,953
8	**121**	29	0,806
9	**147**	3	0,980
10	**120**	30	0,800
11	**104**	46	0,693
12	**146**	4	0,973
13	**104**	46	0,693
14	20	130	0,133
15	10	140	0,066
16	1	149	0,006
17	10	140	0,066
18	9	141	0,060
		MO	**0,582**

5	29	**121**	0,806
6	2	**148**	0,986
7	45	**105**	0,700
8	62	**88**	0,586
9	94	56	0,373
10	33	**117**	0,780
11	64	**86**	0,573
12	90	60	0,400
13	22	**128**	0,853
14	7	**143**	0,953
15	13	**137**	0,913
16	40	**110**	0,733
17	28	**122**	0,813
18	76	74	0,493
		MO	**0,703**

Ζεύγος MCI - AD

MCI	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	MCI	AD	test
1	**130**	20	0,866
2	**119**	31	0,793
3	40	110	0,266
4	**146**	4	0,973
5	7	143	0,046
6	4	146	0,026
7	38	112	0,253
8	4	146	0,026
9	14	136	0,093
10	**113**	37	0,753
11	21	129	0,140
12	17	133	0,113
13	33	117	0,220
14	52	98	0,346
15	1	149	0,006
16	**147**	3	0,980
17	**136**	14	0,906
18	**105**	45	0,700
		MO	**0,417**

AD	Κατάταξη Segment σε κλάση		Accuracy
Ασθενής	MCI	AD	test
1	20	**130**	0,866
2	54	**96**	0,640
3	108	42	0,280
4	16	**134**	0,893
5	15	**135**	0,900
6	0	**150**	1,000
7	6	**144**	0,960
8	59	**91**	0,606
9	94	56	0,,373
10	35	**115**	0,766
11	81	69	0,460
12	69	**81**	0,540
13	26	**124**	0,826
14	10	**140**	0,933
15	17	**133**	0,886
16	31	**119**	0,793
17	50	**100**	0,666
18	58	**92**	0,613
		MO	**0,743**

Στις περιπτώσεις χωρίς StandardScaler και χωρίς PCA, χωρίς StandardScaler, με PCA n_ components = 19 και χωρίς StandardScaler, με PCA n_ components = 190 το αποτέλεσμα ήταν 0% σε όλα τα πειράματα.

6.5 Σύνοψη Αποτελεσμάτων

6.5.1 Συνοπτικά αποτελέσματα ανά Segment

Συνοπτικός πίνακας με τα καλύτερα αποτελέσματα ανά μέθοδο με σύγκριση και των τριών κλάσεων.

Μέθοδος	Accuracy	Cross - Validation
Random Forest	0.976	0.972
K Nearest Neighbors	0.98	0.977
SVM	0.978	0.978

Συνοπτικός πίνακας με τα καλύτερα αποτελέσματα ανά μέθοδο με σύγκριση ανά δυο κλάσεις.

Μέθοδος	Ζεύγη κλάσεων	Accuracy	Cross - Validation
Random Forest			
	Healthy – MCI	0.976	0.977
	Healthy – AD	0.973	0.977
	MCI - AD	0.974	0.976
K Nearest Neighbors			
	Healthy – MCI	0,987	0,989
	Healthy – AD	0,984	0,985
	MCI - AD	0,984	0,985
SVM			
	Healthy – MCI	0.987	0.990
	Healthy – AD	0.810	0.984
	MCI - AD	0.980	0.985

6.5.2 Συνοπτικά αποτελέσματα ανά Ασθενή

Ακολουθούν συνοπτικοί πίνακες των αποτελεσμάτων των πειραμάτων ανά ασθενή των μεθόδων Random Forest, KNN και SVM.

Μέθοδος			PCA(19) και StadarScaler		χωρίς PCA, χωρίς StadarScaler		PCA(190) χωρίς StadarScaler		PCA(19) χωρίς StadarScaler	
Ramdom Forest										
Ανά ασθενή				MO Accurancy		MO Accurancy		MO Accurancy		MO Accurancy
	Healthy		5/18	0,381	4/18	0,249	**6/18**	**0,338**	5/18	0,378
	MCI		3/18	0,147	6/18	0,378	**3/18**	**0,160**	3/18	0,151
	AD		10/18	0,504	5/18	0,316	**13/18**	**0,507**	10/18	0,512
Ανά 2 κλάσεις ανά Ασθενή										
	Healthy - MCI									
		Healthy	10/18	0,558	**10/18**	**0,458**	9/18	0,534	10/18	0,556
		MCI	4/18	0,289	**9/18**	**0,514**	5/18	0,294	4/18	0,263
	Healthy-AD									
		Healthy	8/18	0,514	6/18	0,425	**9/18**	**0,483**	8/18	0,502
		AD	11/18	0,639	9/18	0,489	**13/18**	**0,644**	12/18	0,644
	MCI-AD									
		MCI	**8/18**	**0,439**	12/18	0,583	7/18	0,429	7/18	0,433
		AD	**15/18**	**0,687**	11/18	0,504	13/18	0,694	15/18	0,686

Figure 12. Αποτελέσματα Ανά Ασθενή με τη μέθοδο Random Forest.

Μέθοδος			PCA(19) και StadarScaler		χωρίς PCA, χωρίς StadarScaler		PCA(190) χωρίς StadarScaler		PCA(19) χωρίς StadarScaler	
K Nearest Neighboors										
Ανά ασθενή				MO Accurancy		MO Accurancy		MO Accurancy		MO Accurancy
	Healthy		**9/18**	**0,461**	11/18	0,542	7/18	0,443	**8/18**	**0,431**
	MCI		**5/18**	**0,324**	1/18	0,171	5/18	0,356	**4/18**	**347**
	AD		**7/18**	**0,349**	7/18	0,391	6/18	0,381	**8/18**	**0,376**
Ανά 2 κλάσεις ανά Ασθενή										
	Healthy - MCI									
		Healthy	9/18	0,529	**13/18**	**0,695**	**8/18**	**0,503**	8/18	0,489
		MCI	5/18	0,407	**5/18**	**0,338**	**8/18**	**0,448**	6/18	0,426
	Healthy-AD									
		Healthy	13/18	0,754	**16/18**	**0,707**	**13/18**	**0,72**	14/18	0,745
		AD	7/18	0,447	**8/18**	**0,449**	**9/18**	**0,449**	8/18	0,469
	MCI-AD								50%	
		MCI	14/18	0,694	3/18	0,273	12/18	0,691	**14/18**	**0,7**
		AD	8/18	0,473	16/18	0,752	9/18	0,502	1 **9/18**	**0,509**

Figure 13. Αποτελέσματα Ανά Ασθενή με τη μέθοδο K Nearest Neighbors.

Μέθοδος			PCA(19) και StadarScaler		χωρίς PCA, χωρίς StadarScaler		PCA(190) χωρίς StadarScaler		PCA(19) χωρίς StadarScaler	
SVM										
Ανά ασθενή				MO Accurancy		MO Accurancy		MO Accurancy		MO Accurancy
	Healthy		**9/18**	**0,423**	0/18	0	0/18	0	0/18	0
	MCI		**4/18**	**0,220**	0/18	0	0/18	0	0/18	0
	AD		**13/18**	**0,618**	0/18	0	0/18	0	0/18	0
Ανά 2 κλάσεις ανά Ασθενή										
	Healthy - MCI									
		Healthy	**9/18**	**0,533**	0/18	0	0/18	0	0/18	0
		MCI	**7/18**	**0,347**	0/18	0	0/18	0	0/18	0
	Healthy-AD									
		Healthy	**11/18**	**0,582**	0/18	0	0/18	0	0/18	0
		AD	**14/18**	**0,703**	0/18	0	0/18	0	0/18	0
	MCI-AD									
		MCI	**7/18**	**0,417**	0/18	0	0/18	0	0/18	0
		AD	**15/18**	**0,743**	0/18	0	0/18	0	0/18	0

Figure 14. Αποτελέσματα Ανά Ασθενή με τη μέθοδο SVM.

Συγκεντρωτικοί πίνακες για κάθε ασθενή. Τα αποτελέσματα αφορούν τις περιπτώσεις που χρησιμοποιήθηκε η StandardScaler, καθώς στις περιπτώσεις που δεν χρησιμοποιήθηκε η SVM έδινε σε όλες τις περιπτώσεις 0, και η PCA έχει n_components=19.

Healthy	SVM			Random Forest			K Nearest Neighbors			σωστές προβλεψεις ασθενή/ σύνολο πειραμάτων
	Healthy - MCI - AD	Healthy - MCI	Healthy - AD	Healthy - MCI - AD	Healthy - MCI	Healthy - AD	Healthy - MCI - AD	Healthy - MCI	Healthy - AD	
1	16 /150	39 /150	29 /150	53 /150	**97 /150**	53 /150	**68 /150**	**94 /150**	**104 /150**	4/9
2	**107 /150**	**110 /150**	**143 /150**	**121 /150**	**119 /150**	**142 /150**	**127 /150**	**135 /150**	**142 /150**	9/9
3	**89 /150**	**97 /150**	**136 /150**	42 /150	60 /150	**86 /150**	**86 /150**	**88 /150**	**148 /150**	7/9
4	**114 /150**	**138 /150**	**117 /150**	**141 /150**	**143 /150**	**139 /150**	**132 /150**	**134 /150**	**143 /150**	9/9
5	38 /150	**85 /150**	70 /150	21 /150	**90 /150**	49 /150	72 /150	74 /150	**136 /150**	3/9
6	**98 /150**	**98 /150**	**143 /150**	38 /150	41 /150	42 /150	**94 /150**	**93 /150**	**146 /150**	6/9
7	**143 /150**	/150 /150	**143 /150**	**143 /150**	**146 /150**	**143 /150**	**146 /150**	**147 /150**	**149 /150**	9/9
8	**120 /150**	**146 /150**	**121 /150**	**121 /150**	**134 /150**	**84 /150**	**137 /150**	**145 /150**	**140 /150**	9/9
9	53 /150	54 /150	**147 /150**	52 /150	53 /150	**141 /150**	**76 /150**	**78 /150**	**149 /150**	5/9
10	**106 /150**	**120 /150**	**120 /150**	51 /150	**124 /150**	68 /150	**71 /150**	**78 /150**	**139 /150**	7/9
11	17 /150	22 /150	**104 /150**	34 /150	38 /150	**74 /150**	33 /150	38 /150	**120 /150**	3/9
12	**144 /150**	**148 /150**	**146 /150**	**92 /150**	**121 /150**	**148 /150**	23 /150	25 /150	**148 /150**	7/9
13	**58 /150**	65 /150	**104 /150**	60 /150	**95 /150**	**102 /150**	38 /150	43 /150	**114 /150**	5/9
14	16 /150	62 /150	20 /150	17 /150	**76 /150**	23 /150	49 /150	54 /150	66 /150	1/9
15	3 /150	6 /150	10 /150	3 /150	9 /150	3 /150	19 /150	35 /150	69 /150	0/9
16	1 /150	15 /150	1 /150	2 /150	37 /150	1 /150	3 /150	45 /150	4 /150	0/9
17	9 /150	53 /150	10 /150	5 /150	56 /150	8 /150	31 /150	53 /150	55 /150	0/9
18	11 /150	33 /150	9 /150	33 /150	68 /150	40 /150	41 /150	66 /150	64 /150	0/9

Figure 15. Αποτελέσματα των υγιών ασθενών.

MCI	SVM			Random Forest			K Nearest Neighbors			σωστές προβλεψεις ασθενή/ σύνολο πειραμάτων
	Healthy - MCI - AD	Healthy - MCI	MCI - AD	Healthy - MCI - AD	Healthy - MCI	MCI - AD	Healthy - MCI - AD	Healthy - MCI	MCI - AD	
1	11 /150	12 /150	**130 /150**	5 /150	4 /150	**127 /150**	59 /150	61 /150	**134 /150**	3/9
2	32 /150	41 /150	**119 /150**	2 /150	14 /150	**127 /150**	45 /150	49 /150	**145 /150**	3/9
3	10 /150	27 /150	40 /150	4 /150	19 /150	48 /150	5 /150	15 /150	**83 /150**	1/9
4	**114 /150**	**116 /150**	**146 /150**	**94 /150**	**91 /150**	**139 /150**	**71 /150**	71 /150	**134 /150**	8/9
5	3 /150	14 /150	7 /150	3 /150	22 /150	7 /150	33 /150	49 /150	69 /150	0/9
6	4 /150	**96 /150**	4 /150	1 /150	65 /150	3 /150	49 /150	**82 /150**	51 /150	2/9
7	33 /150	**78 /150**	38 /150	7 /150	41 /150	13 /150	**56 /150**	**85 /150**	**86 /150**	4/9
8	0 /150	8 /150	4 /150	2 /150	33 /150	1 /150	20 /150	57 /150	41 /150	0/9
9	4 /150	9 /150	14 /150	2 /150	0 /150	**89 /150**	35 /150	35 /150	**83 /150**	2/9
10	**89 /150**	**111 /150**	**113 /150**	65 /150	**120 /150**	**76 /150**	**67 /150**	**85 /150**	**95 /150**	8/9
11	5 /150	8 /150	21 /150	5 /150	5 /150	25 /150	50 /150	51 /150	**116 /150**	1/9
12	12 /150	48 /150	17 /150	3 /150	67 /150	62 /150	43 /150	51 /150	**113 /150**	1/9
13	32 /150	**105 /150**	33 /150	15 /150	**89 /150**	15 /150	**64 /150**	**90 /150**	**113 /150**	5/9
14	6 /150	6 /150	52 /150	22 /150	35 /150	33 /150	59 /150	59 /150	**125 /150**	1/9
15	1 /150	19 /150	1 /150	4 /150	8 /150	18 /150	40 /150	75 /150	61 /150	0/9
16	**115 /150**	**117 /150**	**147 /150**	**99 /150**	**106 /150**	**143 /150**	56 /150	57 /150	**148 /150**	7/9
17	**118 /150**	**122 /150**	**136 /150**	58 /150	53 /150	**122 /150**	**120 /150**	**122 /150**	**146 /150**	7/9
18	4 /150	3 /150	**105 /150**	7 /150	9 /150	**130 /150**	3 /150	5 /150	**132 /150**	3/9

Figure 16. Αποτελέσματα ασθενών με MCI.

AD	SVM			Random Forest			K Nearest Neighbors			σωστές προβλεψεις ασθενή/ σύνολο πειραμάτων
	Healthy - MCI - AD	Healthy - AD	MCI - AD	Healthy - MCI - AD	Healthy - AD	MCI - AD	Healthy - MCI - AD	Healthy - AD	MCI - AD	
1	**130 /150**	**149 /150**	**130 /150**	66 /150	**119 /150**	72 /150	**97 /150**	**116 /150**	**108 /150**	7/9
2	**93 /150**	**117 /150**	**96 /150**	91 /150	**134 /150**	**107 /150**	46 /150	**78 /150**	57 /150	6/9
3	34 /150	45 /150	42 /150	41 /150	66 /150	49 /150	23 /150	36 /150	26 /150	0/9
4	**92 /150**	**92 /150**	**134 /150**	106 /150	**103 /150**	**142 /150**	**69 /150**	69 /150	**111 /150**	7/9
5	**115 /150**	**121 /150**	**135 /150**	133 /150	**138 /150**	**141 /150**	56 /150	71 /150	**93 /150**	6/9
6	**148 /150**	**148 /150**	/150 /150	149 /150	**150 /150**	**150 /150**	**127 /150**	**127 /150**	**148 /150**	8/9
7	**105 /150**	**105 /150**	**144 /150**	64 /150	66 /150	**124 /150**	**72 /150**	71 /150	**96 /150**	6/9
8	**71 /150**	**88 /150**	**91 /150**	40 /150	**77 /150**	**94 /150**	6 /150	35 /150	21 /150	5/9
9	41 /150	56 /150	56 /150	28 /150	47 /150	**77 /150**	29 /150	34 /150	38 /150	1/9
10	**99 /150**	**117 /150**	**115 /150**	**84 /150**	**103 /150**	**100 /150**	**74 /150**	**97 /150**	**88 /150**	9/9
11	63 /150	**86 /150**	69 /150	7 /150	40 /150	8 /150	17 /150	35 /150	21 /150	1/9
12	45 /150	60 /150	**81 /150**	35 /150	39 /150	**96 /150**	27 /150	31 /150	49 /150	2/9
13	**109 /150**	**128 /150**	**124 /150**	52 /150	75 /150	**117 /150**	37 /150	60 /150	69 /150	4/9
14	**137 /150**	**143 /150**	**140 /150**	**89 /150**	**127 /150**	**104 /150**	54 /150	**87 /150**	61 /150	7/9
15	**129 /150**	**137 /150**	**133 /150**	**118 /150**	**129 /150**	**123 /150**	**77 /150**	**98 /150**	**95 /150**	9/9
16	**98 /150**	**110 /150**	**119 /150**	59 /150	82 /150	**116 /150**	49 /150	55 /150	**78 /150**	5/9
17	**96 /150**	**122 /150**	**100 /150**	**109 /150**	**124 /150**	**121 /150**	**60 /150**	**83 /150**	71 /150	8/9
18	65 /150	74 /150	**92 /150**	**90 /150**	**108 /150**	**114 /150**	23 /150	31 /150	48 /150	4/9

Figure 17. Αποτελέσματα ασθενών με AD.

7 Τεχνικές λεπτομέρειες

Ακολουθήσουν τεχνικές λεπτομέρειες της διπλωματικής.

7.1 Λεπτομέρειες υλοποίησης

Στις παρακάτω ενότητες δίνονται λεπτομερώς θέματα της διπλωματικής που έχουν τεχνικό ενδιαφέρον. Μέσω screen shot παρατίθενται τμήματα κώδικα που δείχνουν τον τρόπο υλοποίησης των μεθόδων.

7.1.1 2D Convolutional Neural Networks

7.1.1.1 Διαχωρισμός Δεδομένων

```
[ ]  # In the first step we will split the data in training and remaining dataset
     X_train, X_rem, y_train, y_rem = train_test_split(x,y, train_size=0.8)

     # Now since we want the valid and test size to be equal (10% each of overall data).
     # we have to define valid_size=0.5 (that is 50% of remaining data) test_size = 0.5
     X_valid, X_test, y_valid, y_test = train_test_split(X_rem,y_rem, test_size=0.5)
```

Με αυτόν τον τρόπο ορίζουμε το ποσοστό των δεδομένων που θα χρησιμοποιηθούν για εκπαίδευση (train), επικύρωση (validation) και τεστ (test). Στο συγκεκριμένο παράδειγμα το 80% κρατήθηκε για εκπαίδευση και μετά από το 20% το μισό (10% του αρχικού) χρησιμοποιήθηκε για validation και το υπόλοιπο για τεστ.

7.1.1.2 Χτίσιμο Μοντέλου

```
model = models.Sequential()
model.add(layers.Conv2D(64,(7,7), strides=(5, 5), input_shape=(432,288,19)))
model.add(Dropout(0.5)) # το 0.5 δίνει το καλύτερο στη δική μας περίπτωση
model.add(BatchNormalization())
model.add(Dense(128, activation='relu'))

#model.add(layers.MaxPooling2D(pool_size=(3, 3), strides=1, padding='same'))
model.add(layers.AveragePooling2D(pool_size=(3, 3), strides=1, padding='same'))
model.add(layers.Flatten())
model.add(layers.Dense(3, activation='softmax'))
```

Μπορεί κανείς να επιλέξει να προσθέσει επιπλέον στρώματα. Εξαρτάται από πολλές παραμέτρους το αν θα προσθέσει ή όχι και αν αυτό τελικά θα αποβεί ωφέλιμο ως προς τα

αποτελέσματα. Οι σημασία των υπολοίπων παραμέτρων εξηγούνται στο κεφάλαιο 5.10. Δεν υπάρχει ένας τρόπος να ξέρει από την αρχή κάποιος ποιες είναι αυτές που θα αποδώσουν το βέλτιστο αποτέλεσμα οπότε πρέπει να γίνονται δοκιμές.

7.1.1.3 Εκτέλεση – Εκπαίδευση Μοντέλου

```
[ ]  model.compile( optimizer=tf.keras.optimizers.Adam(learning_rate=0.0001), loss="categorical_crossentropy", metrics= ["accuracy"])
```

Με τη συγκεκριμένη εντολή κάνουμε το compile και μπορούμε να επιλέξουμε βελτιστοποιητή (optimizer), το ρυθμό εκμάθησης (learning rate), τύπο για την απώλεια (loss) καθώς επίσης και με τον τρόπο που επιθυμούμε να μετρήσουμε το αποτέλεσμα. Στην παρούσα εργασία επιλέχθηκε η ακρίβεια.

```
#Δοκιμάζουμε το μοντέλο μας, τρέχουμε για να δούμε πόσο καλά πήγε
Modelhistory = model.fit(X_train,
                         y_train,
                         validation_data=(X_valid, y_valid),
                         batch_size=20,
                         steps_per_epoch = 127,
                         epochs = 20,
                         verbose = 1)
```

Εδώ ρυθμίζονται οι παράμετροι για την εκπαίδευση του μοντέλου. Πρέπει να προσέξουμε το batch_size * steps_per_epoch δεν πρέπει να ξεπερνάει σε αριθμό τον αριθμό των δεδομένων που έχουμε ορίσει για train.

7.1.1.4 Εκτίμηση Μοντέλου

```
scores = model.evaluate(X_test, to_categorical(y_test,3))
print('Loss: %.3f' % scores[0])
print('Accuracy: %.3f' % scores[1])
```

Με τις συγκεκριμένες εντολές γίνεται η εκτίμηση του μοντέλου που κατασκεύασε κάποιος.

7.1.2 Κλασικοί Ταξινομητές

7.1.2.1 Διαχωρισμός Δεδομένων

```
[ ]  # Splitting the dataset into the Training set and Test set

     X_train, X_test, y_train, y_test = train_test_split(x2, y, test_size = 0.20, random_state=1,stratify=y)
```

Διαχωρισμός των δεδομένων ανά segment, 80% για εκπαίδευση και 20% για τεστ.

```
#Healthy

# Ασθενής 1ος
y_test=[]
X_test=[]
y_train=[]
X_train=[]

for i in range(150):
  y_test.append(y[i])
  k=[]
  for j in range(19):
    k.append(x2[i][j])
  X_test.append(k)

for i in range(150,8100):
  y_train.append(y[i])
  k=[]
  for j in range(19):
    k.append(x2[i][j])
  X_train.append(k)

print(len(y_test), len(X_test),len(y_train),len(X_train))
```

Παράδειγμα διαχωρισμού ανά ασθενή. Πρόκειται για τον πρώτο ασθενή από την κατηγορία των υγιών.

```
x=[]
y1=[]
#class 0-1

for i in range(5400):
  y1.append(y[i])
  k=[]
  for j in range(19):
    k.append(x2[i][j])
  x.append(k)
print(len(x), len(y1))
```

Παράδειγμα τρόπου διαχωρισμού ανά δυο κλάσεις. Το συγκεκριμένο αφορά τις κλάσεις υγιών και MCI.

7.1.2.2 Weight Average

```
# get a list of base models
def get_models():
  models = list()
  models.append(('lr', LogisticRegression()))
  models.append(('cart', DecisionTreeClassifier()))
  models.append(('bayes', GaussianNB()))
  return models
```

Παράδειγμα καθορισμού 3 κλάσεων που θα χρησιμοποιηθούν αργότερα.

```
# create the base models
models = get_models()
```

```
# evaluate each base model
def evaluate_models(models, X_train, X_val, y_train, y_val):
  # fit and evaluate the models
  scores = list()
  for name, model in models:
    # fit the model
    model.fit(X_train, y_train)
    # evaluate the model
    yhat = model.predict(X_val)
    acc = accuracy_score(y_val, yhat)
    # store the performance
    scores.append(acc)
    # report model performance
  return scores
```

```
# fit and evaluate each model
scores = evaluate_models(models, X_train, X_val, y_train, y_val)
print(scores)
```

```
# create the ensemble
ensemble = VotingClassifier(estimators=models, voting='soft', weights=scores)
```

```
# fit the ensemble on the training dataset
ensemble.fit(X_train_full, y_train_full)
```

```
# make predictions on test set
yhat = ensemble.predict(X_test)
```

```
# evaluate predictions
score = accuracy_score(y_test, yhat)
print('Weighted Avg Accuracy: %.3f' % (score*100))
```

Κώδικας για την εκτέλεση και σύγκριση των τριών μεθόδων. Στο τέλος εκτυπώνεται η απόδοση του κάθε μοντέλου αλλά και η απόδοση του συνδυασμού και των τριών.

7.1.2.3 Random Forest

```
classifier = RandomForestClassifier(n_estimators=200, bootstrap=False, n_jobs=3)
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)

# report performance
ac = accuracy_score(y_test, y_pred)
print('Accuracy: %.3f' % ac)
```

Δημιουργία ταξινομητή Random Forest, εκτέλεση του μοντέλου καθώς και πρόβλεψη και εκτύπωση ακρίβειας.

7.1.2.4 K Nearest Neighbors

```
# Training the K-NN model on the Training set

#classifier = KNeighborsClassifier(n_neighbors=3, weights='distance' , p=1, metric='manhattan')
classifier = KNeighborsClassifier(n_neighbors=3, p=1, metric='manhattan', weights='uniform')
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)

# Making the Confusion Matrix

#cm = confusion_matrix(y_test, y_pred)
ac = accuracy_score(y_test, y_pred)
print(ac)
```

Δημιουργία ταξινομητή Kneighbors, εκτέλεση του μοντέλου καθώς και πρόβλεψη και εκτύπωση ακρίβειας.

7.1.2.5 SVM

```
classifier = SVC(C=1.5, gamma=1.4e-1)
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)

#cm = confusion_matrix(y_test, y_pred)
ac = accuracy_score(y_test, y_pred)
print('Accuracy: %.3f' % ac)
```

Δημιουργία ταξινομητή SVM, εκτέλεση του μοντέλου καθώς και πρόβλεψη και εκτύπωση ακρίβειας.

7.1.2.6 Εκτίμηση αποτελεσμάτων με Cross-Validation

```
# define the model
model = RandomForestClassifier(n_estimators=200, bootstrap=False, verbose=1)
```

```
# evaluate the model
cv = RepeatedStratifiedKFold(n_splits=19, n_repeats=5, random_state=1)
n_scores = cross_val_score(model, x2, y, scoring='accuracy', cv=cv, n_jobs=-1, error_score='raise')
# report performance
print('Accuracy: %.3f (%.3f)' % (mean(n_scores), std(n_scores)))
```

Παράδειγμα Cross-Validationμε ταξινομητή Random Forest.

```
# 10-Fold Cross validation

n_scores = cross_val_score(model, x2, y, scoring='accuracy', cv=10, n_jobs=-1, error_score='raise')
# report performance
print('Accuracy: %.3f (%.3f)' % (mean(n_scores), std(n_scores)))
```

Παράδειγμα 10-Fold cross validation.

7.2 Πλατφόρμες και προγραμματιστικά εργαλεία

Δεν εγκαταστάθηκε κάποιο εργαλείο στον υπολογιστή για τη διεξαγωγή των πειραμάτων. Έγιναν όλα online με τη χρήση του ColabPro. Λόγω του ότι απαιτείται μεγάλη μνήμη για να εκτελεστούν και μάλιστα γρήγορα, δυνατότητα που δεν υπήρχε στο laptop, επιλέχθηκε το colab. Επειδή όμως η κλασσική έκδοση δεν ήταν επαρκής για την διεκπεραίωση των εργασιών προτιμήθηκε η συνδρομή στο Colab Pro μέσω της οποίας τα πειράματα εκτελούνταν σε πολύ λιγότερο χρόνο και επιπροσθέτως είχαμε την ευκαιρία να ανεβάζουμε μεγάλα αρχεία από το Google Drive. Επιπλέον στα αρχικά δεδομένα χρειάστηκε να γίνουν πολλές δοκιμές μέχρι να καταλήξουμε σε αυτά που θέλουμε να χρησιμοποιήσουμε. Οι πίνακες που δημιουργήθηκαν στο πρώτο πειραματικό στάδιο ήταν αρκετά μεγάλοι με αποτέλεσμα να πρέπει να αγοραστεί επιπλέον χώρος στο Google Drive.

8 *Επίλογος*

Ακολουθεί σύνοψη της διπλωματικής.

8.1 Σύνοψη και συμπεράσματα

Στην παρούσα ενότητα συνοψίζονται τα αποτελέσματα της διπλωματικής και περιγράφονται τα συμπεράσματα που προέκυψαν, αρνητικά και θετικά. Η διαφοροποίηση σχετικά με τις υπόλοιπες εργασίες, πλην αυτής των Ieracitano κα [24], είναι η εξέταση ανά ασθενή, τουλάχιστον κατά την έρευνα που διεξήχθη δεν βρέθηκε κάποια άλλη.

Τα αποτελέσματα από τα 2D - Convolutional Neural Networks δεν ήταν ικανοποιητικά. Με το μεγαλύτερο ποσοστό να είναι 71% ανά segment δεν θεωρείται αξιόπιστο για να μπορέσει να δώσει ασφαλή αποτελέσματα, ειδικά στην περίπτωση των δοθέντων δεδομένων που αφορούν ιατρικά δεδομένα. Επιπλέον είχαν γίνει πειράματα ανά ασθενή αλλά με πειράματα τα οποία έγιναν απομονώνοντας έναν ένα τα κανάλια ή σε συνδυασμό κάποια κανάλια αλλά το ποσοστό επιτυχίας δεν ξεπερνούσε το 50%. Αυτός ήταν και ο λόγος που δεν παρατέθηκαν τα αποτελέσματα αυτών των πειραμάτων καθώς θεωρήθηκαν αναξιόπιστα.

Αναφορικά με τη σύγκριση των υπόλοιπων 8 μεθόδων την καλύτερη απόδοση φάνηκε να έχουν τα Random Forest, KNN και SVM. Η εκτενέστερη ενασχόληση με αυτά έγινε μέσω πολλών παραμετροποιήσεων και ως προς τα δεδομένα εισαγωγής αλλά και ως προς τις παραμέτρους των μεθόδων. Επιπροσθέτως εξετάστηκαν και ανά segment αλλά και ως προς ασθενή και τέλος ανά ζεύγη κλάσεων. Την καλύτερη απόδοση ανά segment, με τη χρήση μόνο προεπεξεργασίας δεδομένων και καμία άλλη παραμετροποίησης είχε η Keneighbors με ποσοστό 96.10%. Το ποσοστό ανεβαίνει ανεπαίσθητα στο 96.20% στην ίδια μέθοδο όταν γίνονται τροποποιήσεις στον τρόπο διαχωρισμού των δεδομένων κρατώντας την στην πρώτη θέση. Ακολούθως τα ίδια αποτελέσματα υπάρχουν και όταν γίνονται παραμετροποιήσεις κατά την εκτέλεση των μοντέλων με την KNN να φτάνει το 98%. Με δεύτερη την SVM με ποσοστό 97.80% και τρίτη τη Random Forest με ποσοστό 97.60%, χωρίς όμως η τελευταία να παρουσιάζει σταθερότητα ως προς τα αποτελέσματα.

Ως προς το διαχωρισμό ανά δυο κλάσεις η KNN είχε την καλύτερη απόδοση στους 5 από τους 6 συνδυασμούς. Ακολούθησαν τα πειράματα ανά ασθενή. Εδώ η SVM φάνηκε να έχει αποτελέσματα μόνο όταν γινόταν χρήση της StandardScaler, σε οποιαδήποτε άλλη περίπτωση

το αποτέλεσμα ήταν 0.0%. Στις υπόλοιπες μεθόδους λειτούργησε κανονικά η μέθοδος χωρίς τη χρήση της. Τα αποτελέσματα ήταν πολύ πιο χαμηλά απ' ότι τα πειράματα ανά segment. Οι ασθενείς με MCI έχουν το μικρότερο ποσοστό επιτυχίας, ταξινομήθηκαν σωστά από 3-εως 8/18. Στις δοκιμές ανά ασθενή ανά ζεύγη κλάσεων η SVM και πάλι χωρίς τη χρήση StandardScaler είχε αποτελέσματα 0.0%. Οι δυο άλλοι μέθοδοι όμως έδωσαν καλύτερα αποτελέσματα και μάλιστα στην περίπτωση που δε χρησιμοποιήθηκε ούτε StandardScaler ούτε PCA.

Τέλος όσο αφορά τα αποτέλεσμα που δείχνει ο κάθε ασθενής σε όλες τις μεθόδους (εικόνες 15-17), μπορεί να διακρίνει κανείς ότι υπάρχουν ασθενείς οι οποίοι όποια μέθοδο και αν χρησιμοποιήθηκε το αποτέλεσμα ήταν 0%, καθώς όλα τα τμήματα κατατασσόταν σε άλλες μεθόδους. Επιπλέον υπάρχουν και κάποιοι ασθενείς οι οποίοι όποια μέθοδο και αν χρησιμοποιήθηκε τα περισσότερα μέρη του EEG τοποθετήθηκαν στη σωστή κλάση. Η SVM φάνηκε να δίνει καλύτερα αποτελέσματα και στις τρεις κλάσεις σε σχέση με τις άλλες δυο ομάδες. Επίσης η Random Forest δεν είχε σταθερά αποτελέσματα όσο αφορά τα δικά μας δεδομένα, δηλαδή αν κάποιος ξανά τρέξει τα πειράματα θα έχει άλλες τιμές. Οι άλλες δυο μέθοδοι έχουν σταθερή απόδοση.

8.2 Μελλοντικές επεκτάσεις

Αρκετές από τις δυσκολίες που παρουσιάστηκαν είχαν να κάνουν με την προ-επεξεργασία των δεδομένων. Και κυρίως με τη διαφορετικότητα που παρουσιάζουν τα δεδομένα και έχει να κάνει με το εργαστήριο από το οποίο προέρχονται. Ενδιαφέρον θα είχε να αναπτυχθεί μια εφαρμογή η οποία θα μπορούσε να κάνει την κατάλληλη προ-επεξεργασία χωρίς να επηρεάζεται από το εργαστήριο το οποίο προήρθαν. Επιπροσθέτως να γίνουν περισσότερες έρευνες για την αποτελεσματικότητα των μεθόδων ανά ασθενή γιατί αυτό είναι το ζητούμενο, να μπορεί μια μέθοδος να δίνει αποτελέσματα όταν δίνονται νέα δεδομένα ενός ασθενή με σιγουριά.

9 *Βιβλιογραφία*

[1] S. Li, C. Zhao, και Y. Wang, 'Optimal Lead Research on Bispectral Features of Driving Fatigue EEG Signal', στο *Computer Technology and Transportation ISCTT 2021; 6th International Conference on Information Science*, Αυγούστου 2021, σσ. 1–4.

[2] A. S. Abdulhussien, A. T. AbdulSadda, και A. A. Farawn, 'New Automatic EEG Epileptic Seizure Detection Approach Using Sliding Discrete Fourier Transform and Machine Learning Techniques', στο *2021 2nd Asia Conference on Computers and Communications (ACCC)*, Singapore, Σεπτεμβρίου 2021, σσ. 26–31. doi: 10.1109/ACCC54619.2021.00011.

[3] L. Hu και Z. Zhang, Επιμ., *EEG Signal Processing and Feature Extraction*. Singapore: Springer Singapore, 2019. doi: 10.1007/978-981-13-9113-2.

[4] G. S. Mihov και D. H. Badarov, 'Application of a Reduced Band-pass Filter in the Extraction of Power-line Interference from ECG Signals', στο *2020 XXIX International Scientific Conference Electronics (ET)*, Σεπτεμβρίου 2020, σσ. 1–4. doi: 10.1109/ET50336.2020.9238202.

[5] S. S. Daud και R. Sudirman, 'Butterworth Bandpass and Stationary Wavelet Transform Filter Comparison for Electroencephalography Signal', στο *Modelling and Simulation 2015 6th International Conference on Intelligent Systems*, Οκτωβρίου 2015, σσ. 123–126. doi: 10.1109/ISMS.2015.29.

[6] Kumar Sarangi Shubhendu, Rutuparna Panda, Pradeep Kumar Das, και Ajith Abraham, 'Elsevier Enhanced Reader', 2018. https://reader.elsevier.com/reader/sd/pii/S0952197618300058?token=8BD59E79F0F03046E2666CF37D81CB78FAC48F041F347F7DD67E220C06FFA6E70A5EA30A32D89B1A5DB5B8C567573EBD&originRegion=eu-west-1&originCreation=20220917081839 (ημερομηνία πρόσβασης 17 Σεπτέμβριος 2022).

[7] J. V. Stone, *Independent Component Analysis: A Tutorial Introduction*. MIT Press, 2004.

[8] V. Duc Nguyen, E. Zwanenburg, S. Limmer, W. Luijben, T. Back, και M. Olhofer, 'A Combination of Fourier Transform and Machine Learning for Fault Detection and Diagnosis of Induction Motors', στο *2021 8th International Conference on Dependable Systems and Their Applications (DSA)*, Yinchuan, China, Αυγούστου 2021, σσ. 344–351. doi: 10.1109/DSA52907.2021.00053.

[9] Ά. Μιχάλας, 'Τηλεπικοινωνίες', παρουσιάστηκε στο Αναλυση Fourier.

[10] H. J. Nussbaumer, στο *Fast Fourier transform and convolution algorithms*, 2., corr.Updated ed., 2. Print., Berlin Heidelberg: Springer, 1990.

[11] V. M. Anu, G. S. A. Mala, και K. Mathi, 'Comparison of RFID data processing using dimensionality reduction techniques', στο *2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*, Kanyakumari District, India, Ιουλίου 2014, σσ. 265–268. doi: 10.1109/ICCICCT.2014.6992967.

[12] F. Miraglia, F. Vecchio, P. Bramanti, και P. M. Rossini, 'EEG characteristics in "eyes-open" versus "eyes-closed" conditions: Small-world network architecture in healthy aging and age-related brain degeneration', *Clinical Neurophysiology*, τ. 127, τχ. 2, σσ. 1261–1268, Φεβρουαρίου 2016, doi: 10.1016/j.clinph.2015.07.040.

[13] M. S. Fathillah, R. Jaafar, K. Chellappan, και R. Remli, 'A study on EEG signals during eye-closed and eye-open using discrete wavelet transform', στο *2016 IEEE EMBS*

Conference on Biomedical Engineering and Sciences (IECBES), Malaysia, Δεκεμβρίου 2016, σσ. 674–678. doi: 10.1109/IECBES.2016.7843535.
[14] L. R. Trambaiolli, A. C. Lorena, F. J. Fraga, P. A. M. Kanda, R. Anghinah, και R. Nitrini, 'Improving Alzheimer's Disease Diagnosis with Machine Learning Techniques', *Clin EEG Neurosci*, τ. 42, τχ. 3, σσ. 160–165, Ιουλίου 2011, doi: 10.1177/155005941104200304.
[15] T. Staudinger και R. Polikar, 'Analysis of complexity based EEG features for the diagnosis of Alzheimer's disease', στο *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, Δεκεμβρίου 2011, σσ. 2033–2036. doi: 10.1109/IEMBS.2011.6090374.
[16] V. Podgorelec, 'Analyzing EEG Signals with Machine Learning for Diagnosing Alzheimer's Disease', *Elektronika ir Elektrotechnika*, τ. 18, τχ. 8, Art. τχ. 8, Οκτωβρίου 2012, doi: 10.5755/j01.eee.18.8.2627.
[17] N. N. Kulkarni και V. K. Bairagi, 'Extracting Salient Features for EEG-based Diagnosis of Alzheimer's Disease Using Support Vector Machine Classifier', *IETE Journal of Research*, τ. 63, τχ. 1, σσ. 11–22, Ιανουαρίου 2017, doi: 10.1080/03772063.2016.1241164.
[18] A. H. Al-nuaimi, E. Jammeh, L. Sun, και E. Ifeachor, 'Higuchi fractal dimension of the electroencephalogram as a biomarker for early detection of Alzheimer's disease', στο *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Ιουλίου 2017, σσ. 2320–2324. doi: 10.1109/EMBC.2017.8037320.
[19] P. Durongbhan *κ.ά.*, 'A Dementia Classification Framework Using Frequency and Time-Frequency Features Based on EEG Signals', *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, τ. 27, τχ. 5, σσ. 826–835, Φεβρουαρίου 2019, doi: 10.1109/TNSRE.2019.2909100.
[20] G. Tavares, R. San-Martin, J. N. Ianof, R. Anghinah, και F. J. Fraga, 'Improvement in the automatic classification of Alzheimer's disease using EEG after feature selection', στο *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, Ιουλίου 2019, σσ. 1264–1269. doi: 10.1109/SMC.2019.8914006.
[21] M. S. Safi και S. M. M. Safi, 'Early detection of Alzheimer's disease from EEG signals using Hjorth parameters', *Biomedical Signal Processing and Control*, τ. 65, σ. 102338, Μαρτίου 2021, doi: 10.1016/j.bspc.2020.102338.
[22] D. Puri, S. Nalbalwar, A. Nandgaonkar, και A. Wagh, 'Alzheimer's disease detection with Optimal EEG channel selection using Wavelet Transform', στο *2022 International Conference on Decision Aid Sciences and Applications (DASA)*, Μαρτίου 2022, σσ. 443–448. doi: 10.1109/DASA54658.2022.9765166.
[23] F. C. Morabito *κ.ά.*, 'Deep convolutional neural networks for classification of mild cognitive impaired and Alzheimer's disease patients from scalp EEG recordings', στο *2016 IEEE 2nd International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI)*, Σεπτεμβρίου 2016, σσ. 1–6. doi: 10.1109/RTSI.2016.7740576.
[24] C. Ieracitano, N. Mammone, A. Bramanti, A. Hussain, και F. C. Morabito, 'A Convolutional Neural Network approach for classification of dementia stages based on 2D-spectral representation of EEG recordings', *Neurocomputing*, τ. 323, σσ. 96–107, Ιανουαρίου 2019, doi: 10.1016/j.neucom.2018.09.071.
[25] X. Bi και H. Wang, 'Early Alzheimer's disease diagnosis based on EEG spectral images using deep learning', *Neural Networks*, τ. 114, σσ. 119–135, Ιουνίου 2019, doi: 10.1016/j.neunet.2019.02.005.
[26] Z. You *κ.ά.*, 'Alzheimer's Disease Classification With a Cascade Neural Network', *Frontiers in Public Health*, τ. 8, 2020, Ημερομηνία πρόσβασης: 9 Αύγουστος 2022. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: https://www.frontiersin.org/articles/10.3389/fpubh.2020.584387
[27] M. Amini, M. M. Pedram, A. Moradi, και M. Ouchani, 'Diagnosis of Alzheimer's Disease by Time-Dependent Power Spectrum Descriptors and Convolutional Neural Network Using EEG Signal', *Computational and Mathematical Methods in Medicine*, τ. 2021, σ. e5511922, Απριλίου 2021, doi: 10.1155/2021/5511922.

[28] J. Brownlee, ‘How to Develop a Weighted Average Ensemble With Python’, *Machine Learning Mastery*, 4 Μάιος 2021. https://machinelearningmastery.com/weighted-average-ensemble-with-python/ (ημερομηνία πρόσβασης 10 Αύγουστος 2022).
[29] M. Wang, K. Gao, L. Wang, και X. Miu, ‘A Novel Hyperspectral Classification Method Based on C5.0 Decision Tree of Multiple Combined Classifiers’, στο *2012 Fourth International Conference on Computational and Information Sciences*, Chongqing, China, Αυγούστου 2012, σσ. 373–376. doi: 10.1109/ICCIS.2012.33.
[30] D. V. Kumar και V. V. J. R. Krishniah, ‘An automated framework for stroke and hemorrhage detection using decision tree classifier’, στο *2016 International Conference on Communication and Electronics Systems (ICCES)*, Coimbatore, India, Οκτωβρίου 2016, σσ. 1–6. doi: 10.1109/CESYS.2016.7889861.
[31] A. L. Latifah, A. Shabrina, I. N. Wahyuni, και R. Sadikin, ‘Evaluation of Random Forest model for forest fire prediction based on climatology over Borneo’, στο *2019 International Conference on Computer, Control, Informatics and its Applications (IC3INA)*, Tangerang, Indonesia, Οκτωβρίου 2019, σσ. 4–8. doi: 10.1109/IC3INA48034.2019.8949588.
[32] M. Bicego και F. Escolano, ‘On learning Random Forests for Random Forest-clustering’, στο *2020 25th International Conference on Pattern Recognition (ICPR)*, Milan, Italy, Ιανουαρίου 2021, σσ. 3451–3458. doi: 10.1109/ICPR48806.2021.9412014.
[33] M. Bicego, ‘K-Random Forests: a K-means style algorithm for Random Forest clustering’, στο *2019 International Joint Conference on Neural Networks (IJCNN)*, Budapest, Hungary, Ιουλίου 2019, σσ. 1–8. doi: 10.1109/IJCNN.2019.8851820.
[34] S. Hrouda-Rasmussen, ‘(Gaussian) Naive Bayes’, *Medium*, 7 Μάιος 2021. https://towardsdatascience.com/gaussian-naive-bayes-4d2895d139a (ημερομηνία πρόσβασης 12 Αύγουστος 2022).
[35] P. Pandey και A. Jain, ‘A Comparative study of Classification techniques: Support vector Machine, Fuzzy Support vector Machine & Decision Trees’, σ. 5, 2016.
[36] M. Auleria, A. I. Arrahmah, και D. E. Saputra, ‘A Review on K-N earest Neighbour Based Classification for Object Recognition’, στο *2021 International Conference on Data Science and Its Applications (ICoDSA)*, Bandung, Indonesia, Οκτωβρίου 2021, σσ. 274–280. doi: 10.1109/ICoDSA53588.2021.9617466.
[37] G. Wang, T. Xu, H. Wang, και Y. Zou, ‘AdaBoost and Least Square Based Failure Prediction of Railway Turnouts’, στο *2016 9th International Symposium on Computational Intelligence and Design (ISCID)*, Hangzhou, Δεκεμβρίου 2016, σσ. 434–437. doi: 10.1109/ISCID.2016.1107.
[38] ‘9.2.8 - Quadratic Discriminant Analysis (QDA) | STAT 508’, *PennState: Statistics Online Courses*. https://online.stat.psu.edu/stat508/lesson/9/9.2/9.2.8 (ημερομηνία πρόσβασης 21 Ιούνιος 2022).
[39] ‘Linear, Quadratic, and Regularized Discriminant Analysis’, 30 Νοέμβριος 2018. https://www.datascienceblog.net/post/machine-learning/linear-discriminant-analysis/ (ημερομηνία πρόσβασης 21 Ιούνιος 2022).
[40] S. Hrouda-Rasmussen, ‘Quadratic Discriminant Analysis’, *Medium*, 7 Μάιος 2021. https://towardsdatascience.com/quadratic-discriminant-analysis-ae55d8a8148a (ημερομηνία πρόσβασης 21 Ιούνιος 2022).
[41] M. Baldeon Calisto και S. K. Lai-Yuen, ‘AdaEn-Net: An ensemble of adaptive 2D–3D Fully Convolutional Networks for medical image segmentation’, *Neural Networks*, τ. 126, σσ. 76–94, Ιουνίου 2020, doi: 10.1016/j.neunet.2020.03.007.
[42] Aphex34, *English: typical CNN architecture*. 2015. Ημερομηνία πρόσβασης: 11 Αύγουστος 2022. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: https://commons.wikimedia.org/wiki/File:Typical_cnn.png
[43] S. Dobilas, ‘Convolutional Neural Networks Explained — How To Successfully Classify Images in Python’, *Medium*, 6 Ιούνιος 2022. https://towardsdatascience.com/convolutional-neural-networks-explained-how-to-successfully-classify-images-in-python-df829d4ba761 (ημερομηνία πρόσβασης 11 Αύγουστος 2022).

[44] J. Ranjani, A. Sheela, και K. P. Meena, 'Combination of NumPy, SciPy and Matplotlib/Pylab -a good alternative methodology to MATLAB - A Comparative analysis', στο *2019 1st International Conference on Innovations in Information and Communication Technology (ICIICT)*, Απριλίου 2019, σσ. 1–5. doi: 10.1109/ICIICT1.2019.8741475.

[45] S. van der Walt, S. C. Colbert, και G. Varoquaux, 'The NumPy Array: A Structure for Efficient Numerical Computation', *Computing in Science & Engineering*, τ. 13, τχ. 2, σσ. 22–30, Μαρτίου 2011, doi: 10.1109/MCSE.2011.37.

[46] A. Gramfort *κ.ά.*, 'MEG and EEG data analysis with MNE-Python', *Frontiers in Neuroscience*, τ. 7, 2013, Ημερομηνία πρόσβασης: 7 Αύγουστος 2022. [Έκδοση σε ψηφιακή μορφή]. Διαθέσιμο στο: https://www.frontiersin.org/articles/10.3389/fnins.2013.00267

[47] P. Ablin, J.-F. Cardoso, και A. Gramfort, 'Faster Independent Component Analysis by Preconditioning With Hessian Approximations', *IEEE Trans. Signal Process.*, τ. 66, τχ. 15, σσ. 4040–4049, Αυγούστου 2018, doi: 10.1109/TSP.2018.2844203.

[48] I. Winkler, S. Debener, K.-R. Muller, και M. Tangermann, 'On the influence of high-pass filtering on ICA-based artifact reduction in EEG-ERP', στο *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Milan, Αυγούστου 2015, σσ. 4101–4105. doi: 10.1109/EMBC.2015.7319296.

[49] J. Dammers *κ.ά.*, 'Integration of Amplitude and Phase Statistics for Complete Artifact Removal in Independent Components of Neuromagnetic Recordings', *IEEE Trans. Biomed. Eng.*, τ. 55, τχ. 10, σσ. 2353–2362, Οκτωβρίου 2008, doi: 10.1109/TBME.2008.926677.

[50] F. Campos Viola, J. Thorne, B. Edmonds, T. Schneider, T. Eichele, και S. Debener, 'Semi-automatic identification of independent components representing EEG artifact', *Clinical Neurophysiology*, τ. 120, τχ. 5, σσ. 868–877, Μαΐου 2009, doi: 10.1016/j.clinph.2009.01.015.

[51] G. Schalk, D. J. McFarland, T. Hinterberger, N. Birbaumer, και J. R. Wolpaw, 'BCI2000: A General-Purpose Brain-Computer Interface (BCI) System', *IEEE Trans. Biomed. Eng.*, τ. 51, τχ. 6, σσ. 1034–1043, Ιουνίου 2004, doi: 10.1109/TBME.2004.827072.

[52] A. L. Goldberger *κ.ά.*, 'PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals', *Circulation*, τ. 101, τχ. 23, Ιουνίου 2000, doi: 10.1161/01.CIR.101.23.e215.

[53] 'Module: tf.keras | TensorFlow v2.10.0', *TensorFlow*. https://www.tensorflow.org/api_docs/python/tf/keras (ημερομηνία πρόσβασης 17 Σεπτέμβριος 2022).

[54] 'scikit-learn: machine learning in Python — scikit-learn 1.1.1 documentation'. https://scikit-learn.org/stable/ (ημερομηνία πρόσβασης 23 Ιούνιος 2022).

[55] T. P. D. Team, 'pandas: Powerful data structures for data analysis, time series, and statistics'. Ημερομηνία πρόσβασης: 18 Σεπτέμβριος 2022. [OS Independent]. Διαθέσιμο στο: https://pandas.pydata.org