2η Εργασία

Μπακίρης Εμμανουήλ mpked21033

Αναλυτική Δεδομένων και Μηχανική Μάθηση 2021-22
ΠΜΣ Κυβερνοασφάλεια και Επιστήμη Δεδομένων
Πανεπιστήμιο Πειραιώς, Τμήμα Πληροφορικής
{ebakiris}@hotmail.com

Περίληψη Στο παρόν έγγραφο παρουσιάζεται η δεύτερη και προγραμματιστική εργασία του μαθήματος. Συγκεκριμένα, γίνεται ανάλυση του Cardiotocography Dataset που βρίσκεται στο UCI ML Repository, με τεχνικές ομαδοποίησης και ταξινόμησης στα δεδομένα μετά από την προεπεξεργασία τους. Η υλοποίηση έγινε σε Python βασιζόμενος στην ευκολία που παρέχει το Jupyter Lab και notebook για την ανάλυση, εκτέλεση και απεικόνιση των πινάκων και γραφικών παραστάσεων. Οι ενότητες έχουν χωριστεί με βάση τα ερωτήματα που έχουν τεθεί προς υλοποίηση, επομένως στην ενότητα 1 γίνεται περιγραφή της διαδικασίας της προεπεξεργασίας, στην ενότητα 2 παρουσιάζονται οι τεχνικές ομαδοποίησης που χρησιμοποιήθηκαν με την αξιολόγηση τους και στο κεφάλαιο 3 αναλύεται ο τρόπος με τον οποίο έχει γίνει η ταξινόμηση, υλοποιώντας ένα νευρωνικό δίκτυο με την βιβλιοθήκη Keras.

1 Προπαρασκευή δεδομένων

1.1 Περιγραφή προσέγγισης

Το dataset περιέχεται σε ένα αρχείο CTG.xlsx. Αυτό περιέχει 3 καρτέλες. Η πρώτη έχει μια περιγραφή των χαρακτηριστικών του συνόλου δεδομένων, η δεύτερη είναι ένα σύνολο κανονικοποιημένο ώς προς κάποιες στήλες (όχι όλες) και με κάποια πρόσθετα χαρακτηριστικά πέραν των 21 βασικών χαρακτηριστικών του συνόλου και η 3η καρτέλα περιέχει ένα ακατέργαστο dataset που δεν έχει περάσει από κάποια προεπεξεργασία και είναι αυτό και το οποίο θα χρησιμοποιήσω στην εργασία μου.

1.2 Επιλογή κατάλληλων χαρακτηριστικών

Από το σύνολο ”Raw Data” επιλέχθηκαν μόνο τα 21 κύρια χαρακτηριστικά, επομένως οι επιπλέον στήλες που υπάρχουν στην αρχή (Filename,...,LBE) έχουν διαγραφεί από το σύνολο καθώς επίσης και οι στήλες προς το τέλος (A,...,Susp). Μαζί με τα 21 χαρακτηριστικά έχω κρατήσει και τις στήλες CLASS και NSP καθώς είναι οι κλάσεις στις οποίες καλούμαστε να ταξινομήσουμε τα δεδομένα μας. Επίσης προστέθηκε μια επιπλέον στήλη στα δεδομένα, η οποία περιέχει μια μοναδική τιμή id για κάθε μια από τις εγγραφές. Αυτό θα μας είναι χρήσιμο στη συνέχεια, για την αναπαράσταση κάποιων γραφημάτων.

1.3 Καθαρισμός δεδομένων

Το επόμενο βήμα που ακολούθησα για την προπεξεργασία του συνόλου είναι να διαπιστώσω αν το σύνολο χρειάζεται ”καθάρισμα”. Συγκεκριμένα, εξέτασα αν περιέχονται τιμές Nan ή Null η ακόμα και αν τα χαρακτηριστικά έχουν σε κάποια κελιά ελλιπείς τιμές. Επίσης, ελέχθηκε και η ύπαρξη διπλότυπων εγγραφών και κατάργηση τους, αφού δεν προσφέρουν σε κάτι στην ανάλυση μας. Αφού έγινε και ανάλυση των χαρακτηριστικών για να διαπιστώσω αν κάποιο από αυτά έχει πολλές ίδιες τιμές στις εγγραφές του συνόλου, θεώρησα ότι αφού τα χαρακτηριστικά DS, DR περιέχουν ίδια τιμή σε ποσοστό άνω του 99 τοις εκατό των εγγραφών τότε δεν χρειάζεται να τα συμπεριλάβω στην ανάλυση. Μια τέτοια κατανομή των τιμών δεν θα μπορούσε να μας προσφέρει και κάτι ιδιαίτερα σημαντικό, παρά να επιβαρύνει το σύνολο δεδομένων μας με ήδη περισσότερα χαρακτηριστικά.

1.4 Μετασχηματισμός

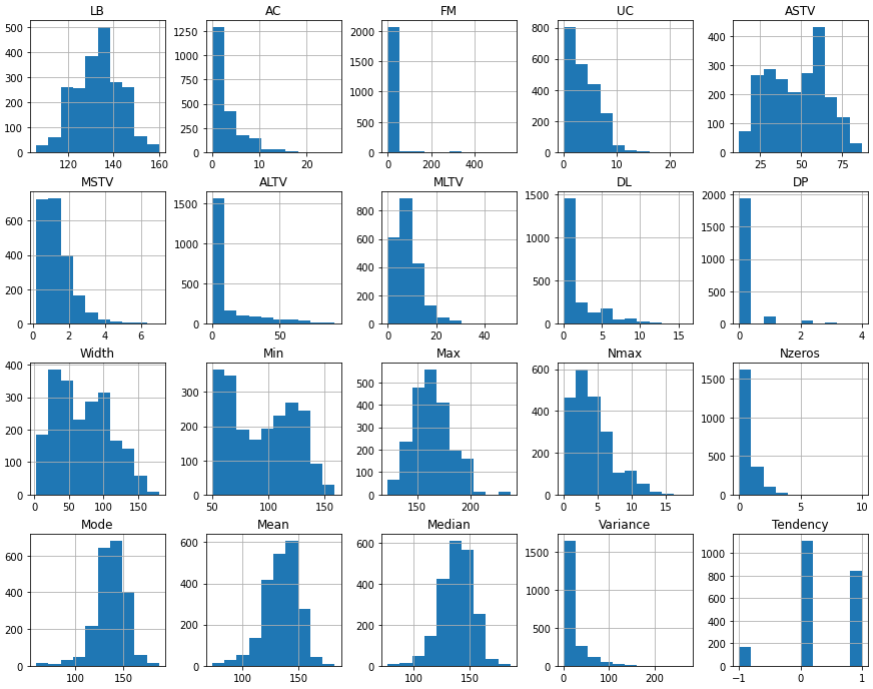
Το επόμενο βήμα αφορά τον μετασχηματισμό της κλάσης NSP. Σύμφωνα με την περιγραφή του dataset, η κατάσταση του εμβρύου (NSP) αποτελείται από 3 κατηγορίες. Κανονική, ύποπτη και παθολογική. Για την απλούστευση και καλύτερη ανάλυση του προβλήματος μετέτρεψα το πρόβλημα σε δυαδικό. Πλέον θα είχα δύο κατηγορίες, την κανονική και την ύποπτη/παθολογική με τιμές 1 και 0 αντίστοιχα. Υποθέτω ότι ο γιατρός θα χρησιμοποιούσε την ανάλυση για την πρόβλεψη της κατάστασης του εμβρύου και όχι μεταγενέστερα, οπότε ο μετασχηματισμός θεωρώ είναι μια λογική προσέγγιση.

1.5 Αναπαράσταση των δεδομένων

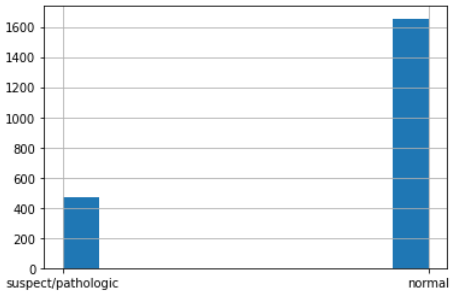
Προτού προχωρήσω στην κανονικοποίηση των δεδομένων, έκανα κάποιες αναπαραστάσεις. Πιο συγκεκριμένα, αναπαραστάθηκαν τα ιστογράμματα των κλάσεων και των χαρακτηριστικών. Αυτό το έκανα για να διαπιστώσω αν κάποιο από τα χαρακτηριστικά ακολουθεί την κανονική (Gaussian) κατανομή. Αν τα δεδομένα ακολουθούν την κανονική κατανομή, τότε θα μπορούσα να αποκλείσω την τεχνική της κανονικοποίησης (Normalization) στο διάστημα [0,1] και θα προχωρούσα αμέσως σε κάποιου είδους τυποποίησης των δεδομένων (Standardization). Δεν συνίσταται η κανονικοποίηση όταν τα δεδομένα ακολουθούν την κανονική κατανομή. Όμως, από τα ιστογράμματα των χαρακτηριστικών φάνηκε ότι υπήρχαν κάποια που πιθανόν να ακολουθούν τη κανονική κατανομή όπως το LB ή το Median, όπως φαίνεται και στην εικόνα 1. Για να σιγουρευτώ για την κατανομή, εφάρμοσα και άλλες δύο μετρικές. Τα Kolmogorov-Smirnov και Shapiro-Wilk Test, τα οποία έδειξαν ότι κανένα χαρακτηριστικό δεν ακολουθεί την κανονική κατανομή, τελικά.

Επίσης απεικόνισα και τα ιστογράμματα των δύο χαρακτηριστικών-στόχων NSP και FHR για να έχω μια εικόνα πως κατανέμονται οι εγγραφές του συνόλου δεδομένων (εικόνες 2,3).

Στη συνέχεια, πραγματοποίησα ενα box plot όλων των χαρακτηριστικών (εικόνα 4). Αυτό, διότι βοηθάει στην ανίχνευση εκείνων των στηλών που έχουμε πολύ θόρυβο, απομακρυσμένα σημεία (outliers or irregular data). Από αυτό, μπορεί κανείς να δει ότι

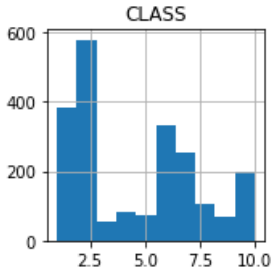


Εικ. 1. Ιστογράμματα των κύριων χαρακτηριστικών

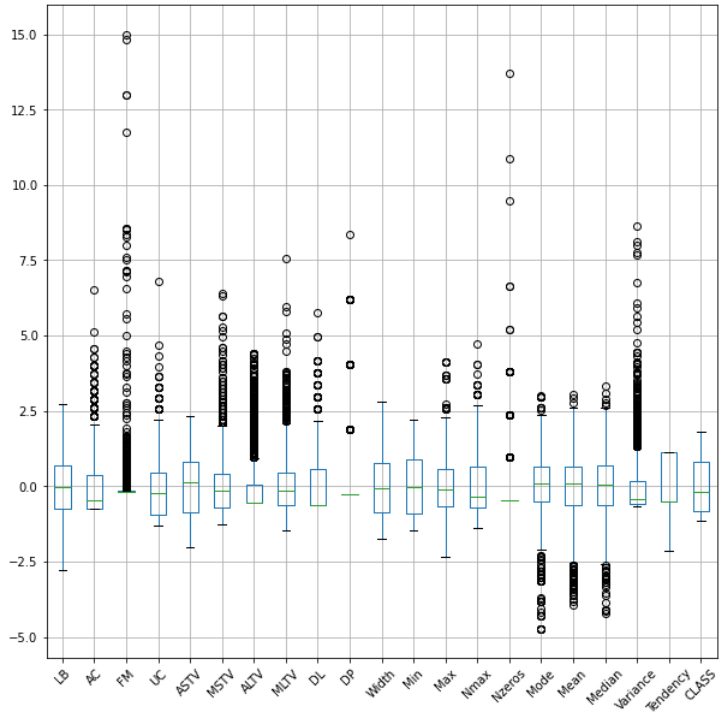


Εικ. 2. Ιστόγραμμα της κλάσης NSP

έχουμε χαρακτηριστικά με πολλά outliers. Επομένως, θεώρησα ότι μια υποψήφια μέθοδος για την κανονικόποιηση των τιμών των χαρακτηριστικών η οποία είναι και κατάλληλη για δεδομένα με πολλά outliers, είναι η μέθοδος robust scaling.



Εικ. 3. Ιστόγραμμα της κλάσης FHR



Εικ. 4. Box plot όλων των χαρακτηριστικών

1.6 Κανονικοποίηση

Σε αυτό το στάδιο της προεπεξεργασίας των δεδομένων, εφάρμοσα τρεις μεθόδους με σκοπό να επιλέξω αυτή που έχει καλύτερα αποτελέσματα στα επόμενα δύο ερωτήματα της εργασίας. Οι τρεις αυτές μέθοδοι είναι οι min-max scaling, max scaling και robust scaling.

2 Clustering - υλοποίηση με scikit-learn

Οι τεχνικές-αλγόριθμοι clustering που υλοποιήθηκαν, βασίστηκαν στην περιγραφή που παρουσιάζεται στην παρακάτω εικόνα ενός πίνακα που παρέχεται από την βιβλιοθήκη scikit-learn. Η στήλη Scalability είναι αυτή που με καθοδήγησε στις κατάλληλες επιλογές.

Method name	Parameters	Scalability	Usecase	Geometry (metric used)
K-Means	number of clusters	Very large `n_samples`, medium `n_clusters` with MiniBatch code	General-purpose, even cluster size, flat geometry, not too many clusters, inductive	Distances between points
Affinity propagation	damping, sample preference	Not scalable with n_samples	Many clusters, uneven cluster size, non-flat geometry, inductive	Graph distance (e.g. nearest-neighbor graph)
Mean-shift	bandwidth	Not scalable with `n_samples`	Many clusters, uneven cluster size, non-flat geometry, inductive	Distances between points
Spectral clustering	number of clusters	Medium `n_samples`, small `n_clusters`	Few clusters, even cluster size, non-flat geometry, transductive	Graph distance (e.g. nearest-neighbor graph)
Ward hierarchical clustering	number of clusters or distance threshold	Large `n_samples` and `n_clusters`	Many clusters, possibly connectivity constraints, transductive	Distances between points
Agglomerative clustering	number of clusters or distance threshold, linkage type, distance	Large `n_samples` and `n_clusters`	Many clusters, possibly connectivity constraints, non Euclidean distances, transductive	Any pairwise distance
DBSCAN	neighborhood size	Very large `n_samples`, medium `n_clusters`	Non-flat geometry, uneven cluster sizes, outlier removal, transductive	Distances between nearest points
OPTICS	minimum cluster membership	Very large `n_samples`, large `n_clusters`	Non-flat geometry, uneven cluster sizes, variable cluster density, outlier removal, transductive	Distances between points
Gaussian mixtures	many	Not scalable	Flat geometry, good for density estimation, inductive	Mahalanobis distances to centers
BIRCH	branching factor, threshold, optional global clusterer.	Large `n_clusters` and `n_samples`	Large dataset, outlier removal, data reduction, inductive	Euclidean distance between points

Εικ. 5. Αλγόριθμοι clustering scikit-learn

Επομένως, υλοποίησα 5 αλγόριθμους. Τους K-means, Spectral clustering και Birch οι οποίοι προορίζονται για την ομαδοποίηση σε 2 clusters και σύγκριση με την κλάση-στόχο NSP και τους Agglomerative (ward) hierarchical clustering και DBSCAN για την ομαδοποίηση σε 10 clusters και σύγκριση με την κλάση-στόχο FHR.

2.1 Επιλογή χαρακτηριστικών για clustering

Αρχικά, επέλεξα να κάνω clustering και δοκιμές, πειράματα παίρνοντας ανά δύο χαρακτηριστικά του συνόλου δεδομένων. Αυτό διότι έτσι μπορούσα να έχω και μια γραφική αναπαράσταση των clusters από κάθε αλγόριθμο στο επίπεδο. Παρόλα αυτά όμως, αυτό ήταν ανέφικτο, διότι έχουμε πολλά χαρακτηριστικά και τα πιθανά ζευγάρια συνδυασμών για την εφαρμογή των αλγορίθμων clustering ήταν 324 συνδυασμοί (18 επί 18). Επικεντρώθηκα σε 2 διαφορετικές προσεγγίσεις, οι οποίες αναλύονται παρακάτω.

Προσέγγιση 1 Για να καταφέρω να επικεντρωθώ σε κάποια μόνο ζευγάρια χαρακτηριστικών, έκανα αναπαράσταση όλων των ζευγαριών (356) χωρίς να εφαρμόσω κάποια

μέθοδο clustering. Έτσι, παρατήρησα τα δεδομένα και αποφάσισα ποιες γραφικές παραστάσεις μπορούν εύκολα να υποβληθούν σε διαδικασία ομαδοποίησης. Κατέληξα με αυτόν τον τρόπο σε 33 ζευγάρια που φαινόταν ότι μια μέθοδος clustering μπορεί να διαχωρίσει αποτελεσματικά σε 2 ή 10 κλάσεις τις εγγραφές. Η μεθοδολογία που ακολούθησα στη συνέχεια ήταν η εξής: Κατέταξα τα 5 πρώτα χαρακτηριστικά (column) βάση των περισσοτέρων εμφανίσεων στα 33 ζευγαριά που ανακαλύφθηκαν προηγουμένως. Έτρεξα πειράματα ξεκινώντας από αυτό με τις περισσότερες εμφανίσεις και κατέγραψα τις τιμές του confusion matrix στις διαγωνίους, και για τους 3 αλγόριθμους Spectral clustering, Birch και K-means clustering συγκρίνοντας με τις πραγματικές τιμές της κλάσης NSP. Ομοίως και για την κλάση FHR με τους αλγόριθμους DBSCAN και Agglomerative hierarchical clustering.

Προσέγγιση 2 Η δεύτερη προσέγγιση αφορά την χρήση της τεχνικής Principal component analysis (PCA). Έτρεξα διάφορα πειράματα κάνοντας PCA σε 2 components από 18 χαρακτηριστικά καθώς και επιλεγμένα χαρακτηριστικά.

2.2 Αξιολόγηση αλγορίθμων

Η αξιολόγηση γίνεται με βάση το ποσοστό σωστών προβλέψεων και τα αποτελέσματα του confusion matrix. Για την κλάση NSP οι εγγραφές που ανήκουν στην κλάση 0 είναι 471 ενώ αυτές που ανήκουν στην κλάση 1 είναι 1655.

Προσέγγιση 1 - Kmeans clustering Οι καλύτερες επιδόσεις του αλγορίθμου K-means είναι στους συνδυασμούς που περιέχουν το χαρακτηριστικό FM, με πολύ καλή πρόγνωση για την κλάση 0 όπου βρήκε σωστά 459 εγγραφές, αλλά πολύ κακή για την κλάση 1 όπου βρήκε σωστά μόνο 14. Στα πειράματα με το χαρακτηριστικό Variance είχε γενικά καλές προβλέψεις για την κλάση 1 που κυμαίνονται από 1500-1510 σωστές εγγραφές αλλά για την κλάση 0 είχε σωστές προβλέψεις από 80-110. Στους συνδυασμούς που περιείχαν το χαρακτηριστικό ALTV και τα χαρακτηριστικά Width, Median, Mean ο kmeans είχε μια σχετικά καλύτερη απόδοση με σωστές προβλέψεις για την κλάση 0 που κυμαίνονται από 207-270 ενώ για την κλάση 1 σωστές προβλέψεις 1530-1540.

Προσέγγιση 1 - Spectral clustering Η καλύτερη επίδοση αυτού του αλγορίθμου με πολύ καλή πρόβλεψη με 468 σωστές για την κλάση 0 είναι με τα χαρακτηριστικά LB, MLTV. Όσον αφορά τις προβλέψεις για την κλάση 1, αυτές είναι αρκετά καλές με τα χαρακτηριστικά Variance, Median με 1527 σωστές προβλέψεις. Και οι 2 αυτές περιπτώσεις καλών προβλέψεων για την μία κλάση, ακολοθούν μια πολύ κακή πρόβλεψη για την άλλη κλάση.

Προσέγγιση 1 - Birch clustering Ο συγκεκριμένος αλγόριθμος, παρουσιάζει πολύ καλή πρόβλεψη για την κλάση 1 με 1637 σωστές προβλέψεις, όταν το ένα χαρακτηριστικό είναι το FM, με κακή όμως πρόβλεψη για την κλάση 0 (13 σωστές). Επίσης, στον συνδυασμό MSTV, AC έχει προβλέψει σωστά όλες τις εγγραφές που ανήκουν στην κλάση 0 (471) αλλά κακή πρόβλεψη στην κλάση 1 με 67 σωστές. Τέλος, όταν το

ένα χαρακτηριστικό είναι ALTV και το άλλο ένα εκ των Width, Min, Median, Mean οι σωστές προβλέψεις για την κλάση 0 κυμαίνονται από 150-216 ενώ για την κλάση 1 από 1524 - 1586, σχετικά καλές προβλέψεις και τις δύο κλάσεις συγκριτικά με τους άλλους αλγορίθμους.

Προσέγγιση 2 - Kmeans clustering Οι επιδόσεις του Kmeans εφαρμόζοντας PCA με όλα τα χαρακτηριστικά σε 2 ή και παραπάνω components, κυμαίνονται σε 456 σωστές προβλέψεις για την κλάση 0 και 14 σωστές προβλέψεις για την κλάση 1. Η εφαρμογή PCA σε όλα τα χαρακτηριστικά, δεν επέφερε κάποια σημαντική βελτιώση στην εκτέλεση του αλγορίθμου Kmeans, ο οποίος προβλέπει αρκετά καλά, την μία μόνο κλάση.

Προσέγγιση 2 - Spectral clustering Η εφαρμογή του αλγορίθμου με PCA σε όλα τα χαρακτηριστικά έχει σχετικά καλά αποτελέσματα. Οι σωστές προβλέψεις για την κλάση 0 είναι 413 ενώ για την κλάση 1 812, καλύτερα ποσοστά από προηγούμενες εκτελέσεις. Η αλλαγή των παραμέτρων του αλγορίθμου, όπως πχ ο αριθμός των κοντινότερων γειτόνων επιφέρει μικρές αλλαγές, το βέλτιστο όμως αποτελέσμα που παρουσιάστηκε παραπάνω προκύπτει με 10 κοντινότερους γείτονες. Αξίζει να τονιστεί, ότι ο αριθμός των components στον οποίο κάνουμε PCA δεν επηρεάζει σημαντικά το αποτέλεσμα.

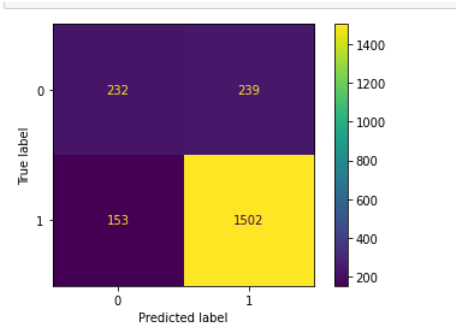
Προσέγγιση 2 - Birch clustering Ανεξαρτήτως τον αριθμό των components στον οποίο θα κάνουμε PCA και ανεξαρτήτως την όποια μεταβολή στις παραμέτρους του, ο συγκεκριμένος αλγόριθμος με PCA με όλα τα χαρακτηριστικά του συνόλου δεδομένων παρουσιάζει την ίδια συμπεριφορά, 1637 σωστές προβλέψεις για την κλάση 1 και 13 σωστές προβλέψεις για την κλάση 0.

Συνδυασμός των 2 προσεγγίσεων - όλοι οι αλγόριθμοι Επιπλέον των παραπάνω πειραμάτων, επιχείρησα να τρέξω και κάποια με συνδυασμό των 2 προσεγγίσεων. Δηλαδή, χρησιμοποιήθηκαν επιλεγμένα χαρακτηριστικά του dataset προς εφαρμογή PCA. Τα χαρακτηριστικά επιλέχθηκαν με βάση προηγούμενα πειράματα κατά τη διαδικασία της πρώτης προσέγγισης. Παρατήρησα πως ο αλγόριθμος Birch με το χαρακτηριστικό ALTV και τα Width, Min, Median και Mean είχε σαν αποτέλεσμα σχετικά καλύτερα αποτελέσματα και για τις δύο κλάσεις. Οπότε, επιλέχτηκαν οι στήλες με τα παραπάνω χαρακτηριστικά για εφαρμογή PCA. Το αποτελέσμα ήταν ενθαρρυντικό, αφού προέβλεψε σωστά 1447 εγγραφές για την κλάση 1 και 277 εγγραφές για την κλάση 0 ο αλγόριθμος Birch. Στις εικόνες 6 και 7 φαίνεται ο confusion matrix και τα αποτελέσματα των μετρικών για αυτά τα πειράματα, τα οποία είναι και αυτά που είχαν την καλύτερη επίδοση.

2.3 Αξιολόγηση για την κλάση FHR

Σχετικά με την σύγκριση με τις τιμές της κλάσης στόχο FHR, κατέληξα στα παρακάτω συμπεράσματα , μετά από σειρά διαδοχικών πειραμάτων:

1. Ο αλγόριθμος Agglomerative hierarchical clustering λειτουργεί αποδοτικότερα με την μορφή ward.



Εικ. 6. Confusion matrix αλγορίθμου Birch με PCA σε επιλεγμένα χαρακτηριστικά (components=3)

```
In [138]: # Rand index
          print(metrics.rand_score(labels_true, labels_pred))
          print(metrics.adjusted_rand_score(labels_true, labels_pred))

          0.6990856067732831
          0.3114438400350549

In [139]: # Mutual Information based scores
          print(metrics.adjusted_mutual_info_score(labels_true, labels_pred))
          print(metrics.normalized_mutual_info_score(labels_true, labels_pred))
          print(metrics.mutual_info_score(labels_true, labels_pred))

          0.1585322974154884
          0.15892820573499855
          0.07961491676435944

In [140]: # Homogeneity, completeness and V-measure
          print (metrics.homogeneity_score(labels_true, labels_pred))
          print (metrics.completeness_score(labels_true, labels_pred))
          print (metrics.v_measure_score(labels_true, labels_pred, beta= 1))

          0.150542418375125
          0.1683033444764122
          0.15892820573499852

In [141]: # Fowlkes-Mallows scores
          metrics.fowlkes_mallows_score(labels_true, labels_pred)

Out[141]: 0.7789367632542058
```

Εικ. 7. Metrics αλγορίθμου Birch με PCA σε επιλεγμένα χαρακτηριστικά (components=3)

2. Ο αλγόριθμος Agglomerative hierarchical clustering (Ward) υπερτερεί κατά πολύ του DBSCAN στην ταξινόμηση χωρίς επίβλεψη για την κλάση στόχο FHR.
3. Η επιλογή του αλγορίθμου DBSCAN δεν αποδίδει ενθαρρυντικά αποτελέσματα, παρά την εναλλαγή των παραμέτρων του ή και της μεθόδου κανονικοποίησης των δεδομένων.
4. Ο αλγόριθμος Ward hierarchical clustering λειτουργεί αρκετά αποδοτικότερα όταν τα δεδομένα έχουν κανονικοποιηθεί με τη μέθοδο min-max scaling, και όχι τη robust scaling.

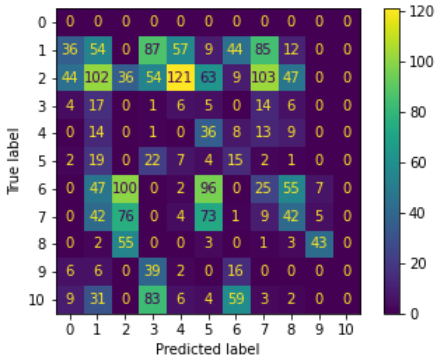
5. Επίσης, ο ward έχει καλύτερες μετρικές και καλύτερο confusion matrix , όταν συμπεριλαμβάνουμε όλα τα χαρακτηριστικά στην ομαδοποίηση και όχι μόνο μερικά από αυτά.
6. Συμπεριφέρεται εξίσου καλά, όταν κάνουμε PCA με όλα τα χαρακτηριστικά, και όχι με υποσύνολο αυτών.

Παρακάτω, βλέπουμε τις μετρικές και τον confusion matrix σε μία από τις καλύτερες εκτελέσεις του αλγορίθμου Agglomerative (ward) hierarchical clustering.

```
In [325]: # Confusion Matrix
          cm = confusion_matrix(labels_true, labels_pred)

          disp = ConfusionMatrixDisplay(confusion_matrix=cm)
          disp.plot()

          plt.show()
```



Εικ. 8. Ward Hierarchical clustering with PCA to all features - Confusion Matrix

2.4 Συμπέρασμα

Το συμπέρασμα της αξιολόγησης των αλγορίθμων clustering συνοψίζεται παρακάτω. Δεν υπάρχει κάποιος αλγόριθμος από αυτούς που χρησιμοποίησα που να παρουσιάζει πολύ καλά αποτελέσματα. Μπορούμε να παρατηρήσουμε όμως, ότι είχαν καλές προβλέψεις μόνο για μία από την δύο κλάσεις και όχι και για τις δύο. Με συγκεκριμένα χαρακτηριστικά όμως, είδαμε μια βελτίωση και μια ισορροπία στις προβλέψεις των 2 κλάσεων. Με εφαρμογή PCA δε, σε επιλεγμένα χαρακτηριστικά είχαμε τα καλύτερα αποτελέσματα. Αξίζει τέλος να σημειωθεί, ότι τα αποτελέσματα είναι ελαφρώς καλύτερα, όταν εφαρμόζουμε κανονικοποίηση min-max scaling, παρόλα αυτά η απόδοση σε αρκετές περιπτώσεις είναι πολύ παρόμοια με κανονικοποίηση robust scaling. Η τελευταία μέθοδος, είναι και αυτή που χρησιμοποιήθηκε για τα δεδομένα, για το ερώτημα της ταξινόμησης (classification) που παρουσιάζεται στην επόμενη ενότητα.

```
In [321]: # Rand index
          print(metrics.rand_score(labels_true, labels_pred))
          print(metrics.adjusted_rand_score(labels_true, labels_pred))

          0.7823044657185546
          0.08911454230663869

In [322]: # Mutual Information based scores
          print(metrics.adjusted_mutual_info_score(labels_true, labels_pred))
          print(metrics.normalized_mutual_info_score(labels_true, labels_pred))
          print(metrics.mutual_info_score(labels_true, labels_pred))

          0.22481207089475044
          0.2320093891512496
          0.4905788487373758

In [323]: # Homogeneity, completeness and V-measure
          print (metrics.homogeneity_score(labels_true, labels_pred))
          print (metrics.completeness_score(labels_true, labels_pred))
          print (metrics.v_measure_score(labels_true, labels_pred, beta= 1))

          0.24293480711208346
          0.22202436866768593
          0.23200938915124958

In [324]: # Fowlkes-Mallows scores
          metrics.fowlkes_mallows_score(labels_true, labels_pred)

Out[324]: 0.21441917786418316
```

Εικ. 9. Ward Hierarchical clustering with PCA to all features - Metrics

3 Classification - υλοποίηση με Keras

Στην εικόνα 10 φαίνεται η δομή του νευρωνικού δικτύου, αριθμός των νευρώνων και των στρωμάτων για τα οποία προέκυψε η καλύτερη επίδοση. Αρχικά, το classification

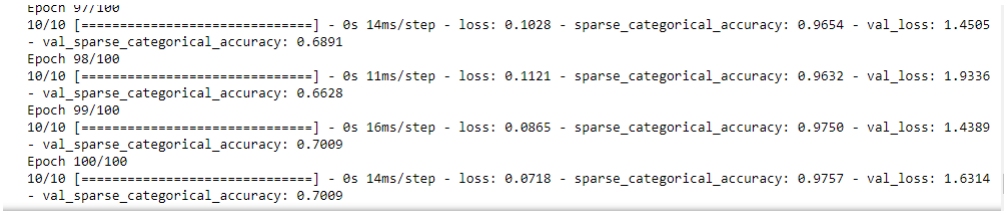
```
# First define baseline model. Then use it in Keras Classifier for the training
# Create model here
model = Sequential()
model.add(Dense(150, input_dim = 18, activation = 'relu')) # Rectified Linear Unit Activation Function
model.add(Dense(150, activation = 'relu'))
model.add(Dense(150, activation = 'relu'))
model.add(Dense(150, activation = 'relu'))
model.add(Dense(11, activation = 'softmax')) # Softmax for multi-class classification
# Compile model here
model.compile(loss = keras.losses.SparseCategoricalCrossentropy(), # default from_logits=False
              optimizer = 'adam', metrics=[keras.metrics.SparseCategoricalAccuracy()])
```

Εικ. 10. Το νευρωνικό δίκτυο με τα καλύτερα αποτελέσματα για το δεδομένο πρόβλημα

έγινε με τα δεδομένα κανονικοποιήμενα με τη μέθοδο min-max scaling. Όμως, παρατηρήθηκαν κακές προβλέψεις για τις κλάσεις 2,3,4. Με αλλαγή στη μέθοδο κανονικοποίησης με αυτής της robust scaling προέκυψε αρκετά καλύτερη πρόβλεψη για αυτές τις

κλάσεις και καλύτερο accuracy καθώς και χαμηλότερο loss (training loss και validation loss). Καθώς εκτελέστηκαν πολλά πειράματα, κατέληξα σε κάποιες παρατηρήσεις σχετικά με τα αποτελέσματα και τις επιδόσεις του νευρωνικού δικτύου. Αυτές συνοψίζονται παρακάτω :

1. Η προσθήκη περισσότερων νευρώνων στην έξοδο ή και γενικά, έχει σαν αποτέλεσμα το validation loss να αυξάνεται.
2. Περισσότεροι νευρώνες επιφέρουν υψηλότερο accuracy, μέχρι ένα σημείο, από το οποίο και έπειτα δεν παρατηρείται περαιτέρω βελτίωση.
3. Η εφαρμογή της robust scaling μεθόδου βελτίωσε το validation loss αλλά και το accuracy.
4. Όσο αυξάνουμε τον αριθμό των εποχών (epochs), το accuracy αυξάνεται και το loss μειώνεται. Πάλι μέχρι ένα σημείο, πέρα από το οποίο δεν παρατηρείται περαιτέρω σημαντική βελτίωση.
5. Μια ομοιόμορφη κατανομή των νευρώνων στα στρώματα αυξάνει το accuracy και μειώνει τα σφάλματα.
6. Από ένα αριθμό στρωμάτων νευρώνων και μετά, η αύξηση των στρωμάτων δεν επιφέρει κάποια βελτίωση. Αρκεί να είναι ομοιόμορφα, να έχουν δηλαδή ίσο αριθμό νευρώνων για καλύτερη απόδοση.
7. Αν αυξηθεί ο αριθμός των νευρώνων στα στρώματα, το validation loss αυξάνεται. Επομένως, λιγότεροι νευρώνες, χαμηλότερο val loss. Μέχρι ενός σημείου και τηρούμενης της ομοιομορφίας.
8. Αν το batch size είναι μικρό, το validation loss αυξάνεται.
9. Αν έχουμε σχετικά χαμηλό validation loss και σχετικά υψηλό accuracy τότε έχουμε και ένα σχετικά υψηλό recall.
10. Κατά τη διαδικασία του training, όσο περνάνε οι εποχές, το accuracy αυξάνεται, το loss μειώνεται και το validation loss αυξάνεται.
11. Υπάρχει μια αντίστροφη σχέση μεταξύ του validation loss και του accuracy. Προσπαθώντας να πετύχουμε καλύτερο, δηλαδή χαμηλότερο, validation loss θυσιάζουμε λίγο από το accuracy (πιο χαμηλό accuracy), και αντίστροφα αν θέλουμε υψηλό accuracy το validation loss θα είναι πιο ψηλό (ενώ μπορεί να φτάσει σε χαμηλότερα επίπεδα).

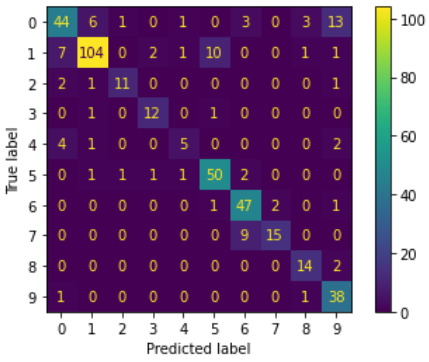


Εικ. 11. Το αποτέλεσμα του training του νευρωνικού δικτύου που υλοποιήθηκε

```
In [31]: # Confusion Matrix for the classification task
cm = confusion_matrix(y_true, y_pred)

disp = ConfusionMatrixDisplay(confusion_matrix=cm)
disp.plot()

plt.show()
```



Εικ. 12. Confusion matrix για την κλάση FHR

```
In [32]: # classification report
from sklearn.metrics import classification_report

print(classification_report(y_true, y_pred))
```

	precision	recall	f1-score	support
1.0	0.76	0.62	0.68	71
2.0	0.91	0.83	0.87	126
3.0	0.85	0.73	0.79	15
4.0	0.80	0.86	0.83	14
5.0	0.62	0.42	0.50	12
6.0	0.81	0.89	0.85	56
7.0	0.77	0.92	0.84	51
8.0	0.88	0.62	0.73	24
9.0	0.74	0.88	0.80	16
10.0	0.66	0.95	0.78	40
accuracy			0.80	425
macro avg	0.78	0.77	0.77	425
weighted avg	0.81	0.80	0.80	425

Εικ. 13. Το αποτέλεσμα του classification για το νευρωνικό δίκτυο