ΑΝΑΦΟΡΑ PART I :
ΕΡΓΑΣΤΗΡΙΑΚΗΣ ΑΣΚΗΣΗΣ 2022-2023

ΘΕΩΡΙΑ ΑΠΟΦΑΣΕΩΝ

(CEID1039)

Στοιχεία

Εργαστηριακή Άσκηση Part I (Ερωτήματα 1-3)

Υπεύθυνη Καθηγητές : Κ. Λυκοθανάσης , Κ. Κοσμόπουλος

Ακαδημαϊκό Έτος : Εαρινό Εξάμηνο 2022-2023

Ονοματεπώνυμο: Ορέστης Αντώνιος Μακρής **Α-Μ** 1084516

ΑΝΤΙΚΕΙΜΕΝΟ:

"**Σχεδιασμός και Αξιολόγηση Συστήματος, από Δημογραφικά και Βιοϊατρικά Δεδομένα, για Πρόβλεψη Ασθένειας** "

Σε αυτή την εργαστηριακή άσκηση καλείστε να χρησιμοποιήσετε ένα σύνολο δεδομένων για να εκπαιδεύσετε και αξιολογήσετε ένα σύνολο ταξινομητών για την πρόβλεψη ασθένειας στο συκώτι.

Το σύνολο δεδομένων που σας δίνεται στο Indian Liver Patient Dataset (ILPD).csv παρέχει ένα σύνολο βιοϊατρικών μετρήσεων για κάθε ασθενή καθώς και την πληροφορία για το αν αυτός πάσχει ή όχι από ασθένεια στο συκώτι. Κάθε γραμμή στο αρχείο αυτό περιέχει πληροφορία για διαφορετικό ασθενή. Η πρώτη στήλη αφορά τον την ηλικία και η δεύτερη το φύλλο, ενώ η τελευταία δείχνει το αν η καταγραφή αφορά μέτρηση ασθενούς ατόμου στο συκώτι (τιμή 2) ή υγιούς ατόμου (τιμή 1). Όλες οι άλλες στήλες αντιστοιχούν σε βιοϊατρικές μετρήσεις που θα πρέπει επίσης να χρησιμοποιήσετε σαν εισόδους στους ταξινομητές που θα δημιουργήσετε. Για περισσότερες πληροφορίες για τα δεδομένα αυτά μπορείτε να δείτε την αναλυτική περιγραφή τους στην βάση δεδομένων μηχανικής μάθησης UCI (http://archive.ics.uci.edu/ml/datasets/ILPD+%28Indian+Liver+Patient+Dataset%29#) από την οποία προέρχονται.

Εργαλεία και βιβλιοθήκες που χρησιμοποιήθηκαν για την υλοποίηση της εργαστηριακής άσκησης:

Το project υλοποιήθηκε στην γλώσσα προγραμματισμού Python, η οποία επιλεκτικέ για το μεγάλο οικοσύστημα από βιβλιοθήκες (Machine Learning , Good visualization options .etc) , ύπαρξη καλής υποστηρίξεις από την κοινότητα , πήγες (documentation), Flexibility , Platform independence.

Πακέτα που αξιοποιήθηκαν για την υλοποίηση του πρώτου part του project και αιτιολόγηση της επιλογής τους :

NumPy : βιβλιοθήκη της Python, παρέχει υποστήριξη για μεγάλους, πολυδιάστατους πίνακες ,ενώ σύγχρονος προσφέρει μια μεγάλη συλλογή μαθηματικών συναρτήσεων υψηλού επιπέδου για μαθηματικές πράξεις σε αυτούς τους πίνακες.

Pandas : βιβλιοθήκη της Python, ανοιχτού κώδικα που χρησιμοποιείται ευρέως για εργασίες επιστήμης δεδομένων/ανάλυσης δεδομένων και μηχανικής μάθησης. Συγκεκριμένα, προσφέρει δομές δεδομένων και λειτουργίες για το χειρισμό αριθμητικών πινάκων και χρονοσειρών είναι χτισμένη πάνω από ένα άλλο πακέτο που ονομάζεται Numpy. Μερικές χρήσεις περιλαμβάνουν την κανονικοποίηση , συγχωνεύεται , ενοποιήσει , οπτικοποίηση, επιθεώρηση , φόρτωση και αποθήκευση των δεδομένων

Matplotlib : είναι μια ολοκληρωμένη βιβλιοθήκη για τη δημιουργία στατικών, κινούμενων και διαδραστικών απεικονίσεων στην Python. Παρέχει ένα αντικειμενοστραφή API για την ενσωμάτωση σχεδίων σε εφαρμογές χρησιμοποιώντας κιτ εργαλείων GUI γενικής χρήσης.

Seaborn: βοηθά στην απλοποίηση σύνθετων απεικονίσεων με την απλότητά του και συμβάλλει στην προσθήκη μιας πρόσθετης αισθητικής έλξης, είναι δομημένο γύρο από τη βιβλιοθήκη matplotlib.

Sklearn : είναι μια βιβλιοθήκη μηχανικής μαθήσεις για τη γλώσσα Python. Περιέχει πολλά αποτελεσματικά εργαλεία για μηχανική μάθηση και στατιστική μοντελοποίηση, όπως ταξινόμηση, παλινδρόμηση, SVM , Naïve Bayes , k - Means. Μας επιτρέπει να ορίζουμε ένα μοντέλο πρόβλεψης δεδομένων σε λίγες μόνο γραμμές κώδικα και, στη συνέχεια, να εκπαιδεύσουμε αυτό το μοντέλο με τα κατάλληλα δεδομένα. Ενσωματώνεται καλά με άλλες βιβλιοθήκες Python, όπως το matplotlib για σχεδίαση, το numpy για τη διανυσματική διάταξη και τα panda για τα πλαίσια δεδομένων.

Πιο συγκεκριμένα πακέτα sklearn που χρησιμοποιήθηκαν:

sklearn.model_selection - > train_test_split χρησιμοποιείται για να χωρίσει τα δεδομένα μας σε σύνολα εκπαιδεύσεις και δοκιμών. Το πλαίσιο δεδομένων χωρίζεται σε X_train, X_test, y_train και y_test. Τα σετ X_train και y_train χρησιμοποιούνται για την εκπαίδευση του μοντέλου. Τα σύνολα X_test και y_test χρησιμοποιούνται για το evaluation του μοντέλου δηλαδή αν προβλέπει τις σωστές εξόδους/ετικέτες. Μπορούμε να ορίσουμε ρητά το μέγεθος του training set και του testing set και τον τρόπο που ανακατεύονται τα δεδομένα πριν το 'σπάσιμο'.

sklearn.naive_bayes - > GaussianNB χρησιμοποιείται για την εφαρμογή των μεθόδων Naive Bayes που βασίζονται στην εφαρμογή του θεωρήματος του Bayes με την «αφελή» υπόθεση της ανεξαρτησίας υπό όρους μεταξύ κάθε ζεύγους χαρακτηριστικών, δεδομένης της τιμής της μεταβλητής κλάσης. Πιο συγκεκριμένα στο παρόν project επικαλείται και χρησιμοποιεί το πακέτο GaussianNB το οποίο εφαρμόζει τον αλγόριθμο Gaussian Naive Bayes για ταξινόμηση. Η πιθανότητα των χαρακτηριστικών θεωρείται ότι είναι Gaussian και υπολογίζεται με βάση τον παρακάτω τύπο:

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

Οι παράμετροι σy και μy υπολογίζονται χρησιμοποιώντας τη μέγιστη πιθανοφάνια.

sklearn. metrics - > confusion_matrix, accuracy_score , ConfusionMatrixDisplay το πακέτο υλοποιεί συναρτήσεις που αξιολογούν το σφάλμα πρόβλεψης για συγκεκριμένους σκοπούς , πιο συγκεκριμένα καλούμε την συνάρτηση accuracy_score , την confusion_matrix και την ConfusionMatrixDisplay.

accuracy_score : η συνάρτηση accuracy_score του sklearn υπολογίζει τη βαθμολογία ακρίβειας για ένα σύνολο προβλεπόμενων ετικετών έναντι των πραγματικών ετικετών.

confusion_matrix: υπολογίζει το confusion matrix του classifier the count of true negatives is $C_{0,0}$, false negatives is $C_{1,0}$, true positives is $C_{1,1}$ and false positives is $C_{0,1}$.

ConfusionMatrixDisplay : αναπαριστά γραφικά το confusion_matrix

sklearn.model_selection -> cross_val_score , KFold

Η συνάρτηση KFold διαιρεί ένα περιορισμένο σύνολο δεδομένων σε k μη επικαλυπτόμενες πτυχές. Σε καθεμία από τις διπλώσεις k δίνεται η ευκαιρία να χρησιμοποιηθεί ως σετ ελέγχου συγκράτησης, ενώ όλες οι άλλες πτυχές χρησιμοποιούνται συλλογικά ως σύνολο δεδομένων εκπαίδευσης.

Το cross_val_score αξιολογεί τη βαθμολογία χρησιμοποιώντας using cross validation χωρίζοντας σε τυχαία τα σετ εκπαίδευσης σε ξεχωριστά υποσύνολα που ονομάζονται folds, στη συνέχεια εκπαιδεύει και αξιολογεί το μοντέλο στις πτυχές, επιλέγοντας διαφορετικό πάσο για αξιολόγηση κάθε φορά.

Ερώτημα 1. Προεπεξεργασία δεδομένων

Να αναφέρετε, πόσα είναι τα χαρακτηριστικά κάθε δείγματος και πόσα δείγματα εκπαίδευσης περιέχει το αρχείο.

Η δεύτερη στήλη περιέχει το φύλο του ανθρώπου που συμμετείχε στο δείγμα. Στο αρχείο όμως είναι σημειωμένη με Male για αρσενικό και Female για θηλυκό. Προκειμένου να την χρησιμοποιήσουμε σαν είσοδο θα πρέπει να αντιστοιχίσετε το Male με την τιμή 0 και το Female με την τιμή 1.

Το εύρος τιμών των δεδομένων που σας έχουν δοθεί διαφέρει σημαντικά ανά χαρακτηριστικό. Για αυτό τον λόγο, για να μην υπερεκτιμηθεί η συνεισφορά κάποιου χαρακτηριστικού έναντι άλλων, θα πρέπει πριν την επεξεργασία των χαρακτηριστικών εισόδου να κανονικοποιηθούν στο εύρος [-1,1]. Χρησιμοποιήστε το matlab (ή όποια άλλη εφαρμογή θέλετε) τόσο για το διάβασμα του αρχείου που σας δίνεται όσο και για την κανονικοποίηση των δεδομένων εισόδου στο εύρος τιμών [-1,1].

Στην αναφορά για το ερώτημα 1 προ επεξεργασία δεδομένων θα αναφερθούν όλες οι τεχνικές που χρησιμοποιήθηκαν πάνω στα δεδομένα πριν την εκπαιδεύσει και την αξιολόγηση του μοντέλου η αιτιολόγηση για την επιλογή τους σε σχέση με την απόδοση του μοντέλου θα γίνει στην αναφορά των αντιστοίχων υπό ερωτημάτων .

Αρχικά φορτώνομε τα δεδομένα από το csv σε ένα pandas data frame στο οποίο με την χρήσει της εντολής replace της βιβλιοθήκης pandas αντικαταστούμε της string gender ως εξής male = 0 και female = 1 . Ομοίως αντικαταστούμε τα δεδομένα των κλάσεων κατηγοριοποιήσεις με της ακέραιές τιμές από 2 ->1 και από 1 ->0 για καλύτερη διαχείριση των πόρων και της μνήμης. Επίσης αφαιρούμε της γραμμές του dataset που περιέχουν τιμές NaN (σύνολο 4 2 liver patient και 2 non liver patient) με την χρήση της συναρτήσεις dropna της pandas. Σε αυτό το στάδιο είμαστε σε θέση να εξετάσουμε το δεδομένα:

Το data set περιέχει 10 χαρακτηριστικά κάθε δείγματος και συνολικά 579 δείγματα πιο αναλυτικά από αυτά:

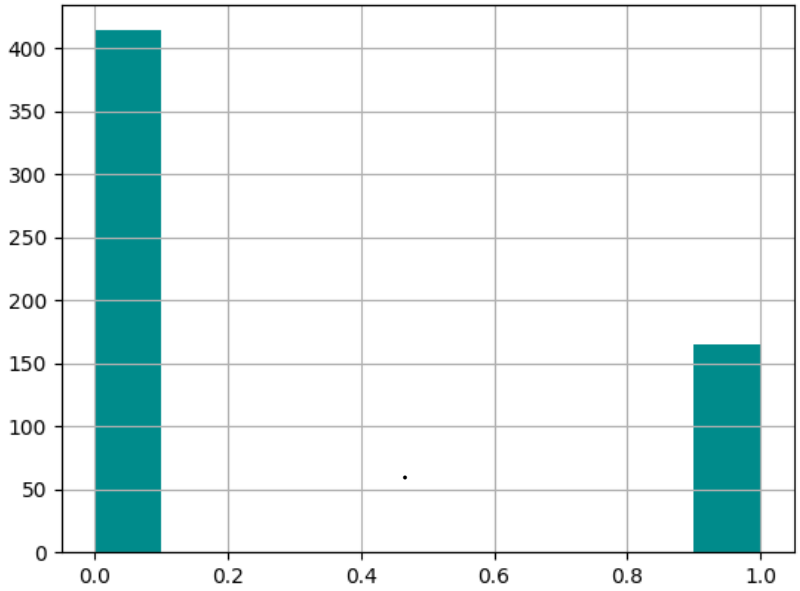
Classes	Sample
0 liver patient	414
1 non liver patient	165

Από το site ILPD (Indian Liver Patient Dataset) Data Set

Data Set Information:

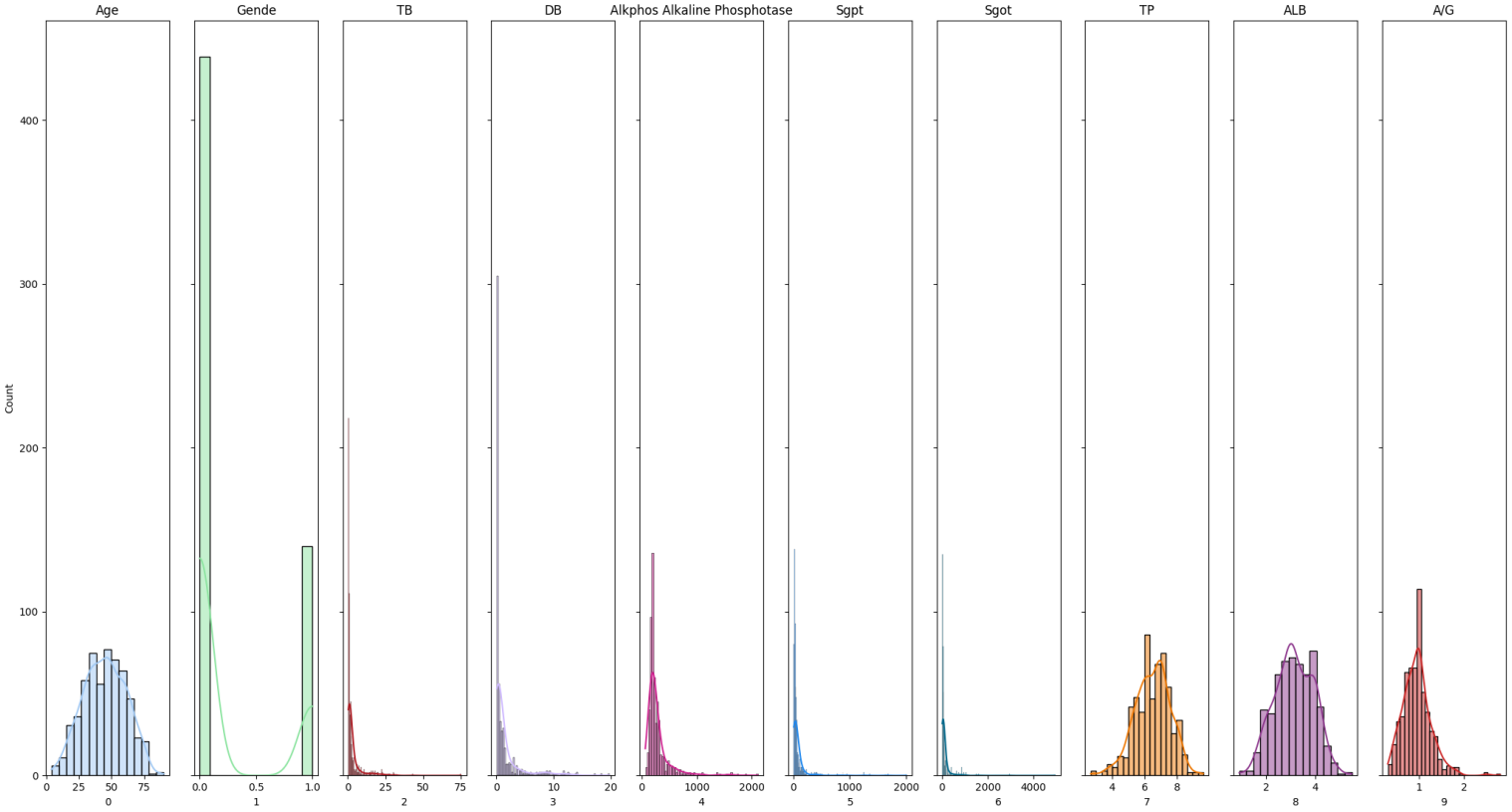
This data set contains 416 liver patient records and 167 non liver patient records.

Number of each class plotting (Histogram)

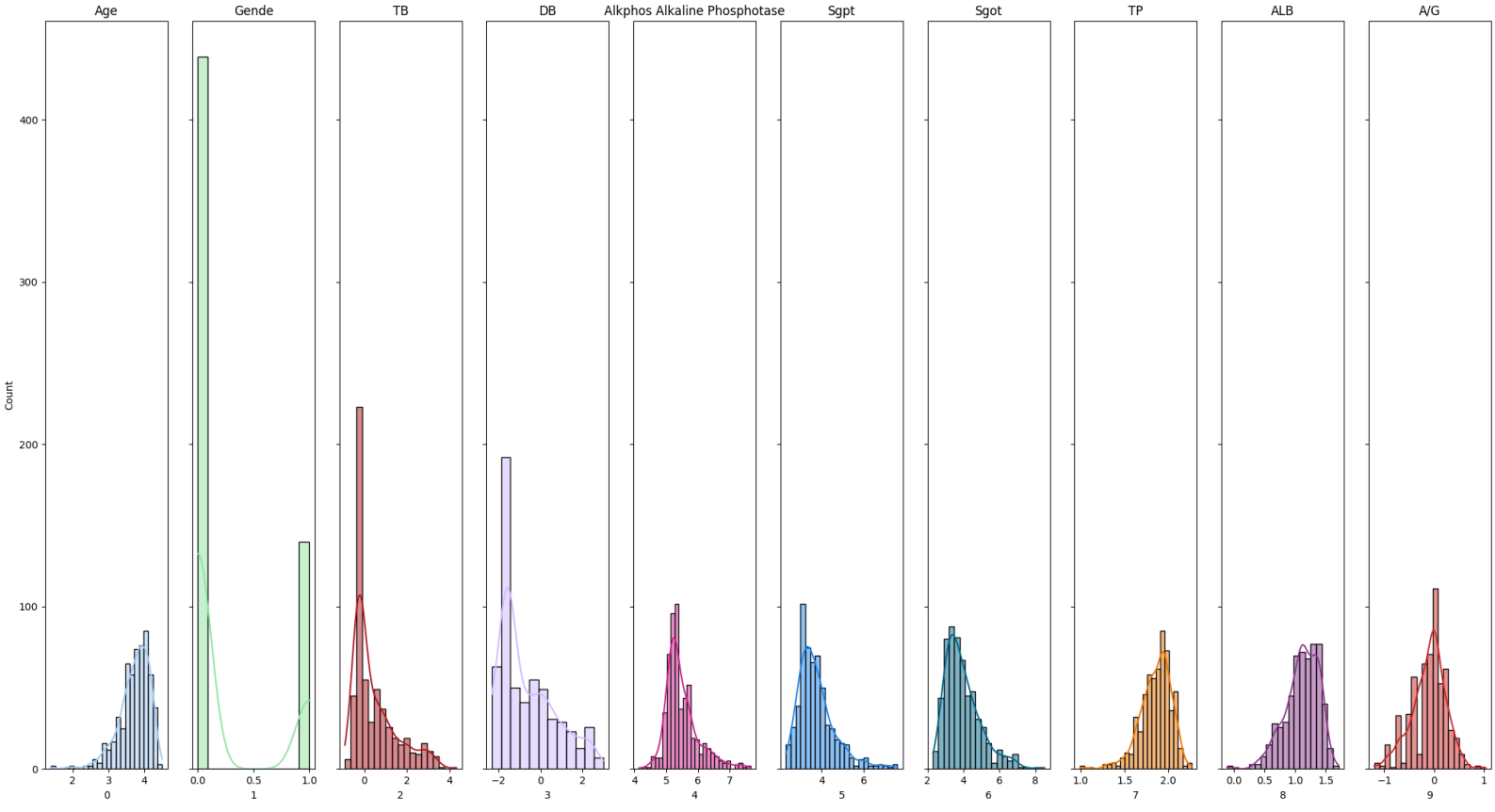


Στην συνέχεια φτιάχνουμε το Histogram τον χαρακτηριστικών των δεδομένων (Age , Gender , TB , DB, Alkphos Alkaline Phosphotase , Sgpt , Sgot , TP . ALB , A/G) με απώτερο να παρατηρήσουμε την κατανομή τους όπως βλέπουμε από το παρακάτω γράφημα για τα δεδομένα υπάρχει περιθώριο βελτιώσεις ώστε να τείνουν προς την κανονική κατανομή ιδικά για τα χαρακτηριστικά TB , DB, Alkphos Alkaline Phosphotase , Sgpt , Sgot και A/G. Για την βελτιώσει των δεδομένων τα περνάμε από λογάριθμο column by column με την χρήση της συνάρτησης log() της βιβλιοθήκης numpy (Η διαφοροποίησε ως προς την απόδοση του μοντέλου θα εξεταστεί στο κατάλληλο υποερώτημα)

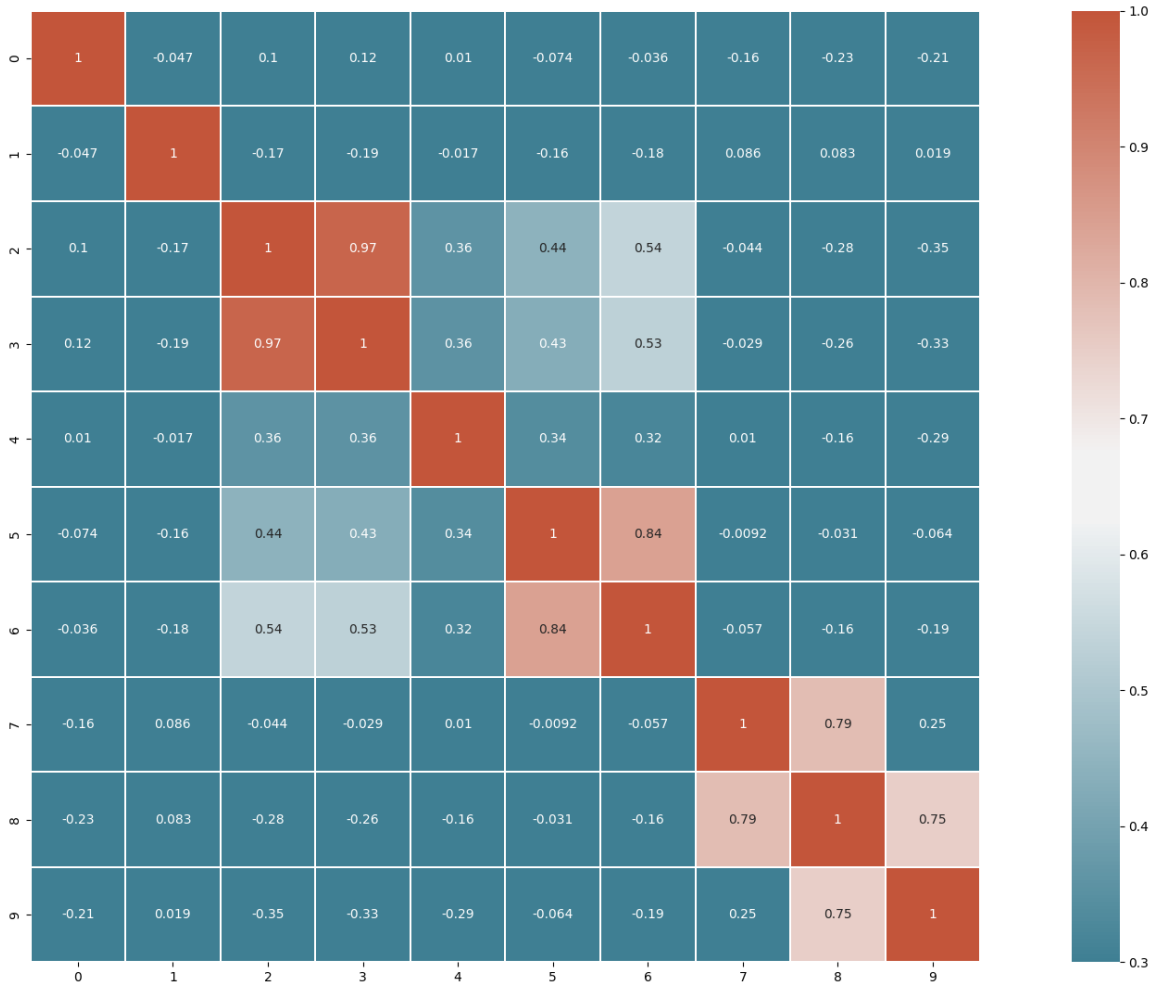
Features Histogram before applying the logarithmic function



Features Histogram after applying the logarithmic function



Εν συνεχεια εξεταζουμαι την σχεση εξαρτης μεταξει των χαρακτηριστικων του συνολου δεδομενων με την χρησει ενως heatmap παρατηερουμε τα χαρακτηρισιτκα με την λιγοετρη εξατειση είναι τα Age , Gender και Alkphos Alkaline Phosphotase και την μεγαλύτερη TB , DB , Sgpt , Sgot , ALB ,& A/G



Για τα δεδομένα πρέπει να υπάρξει κανονικοποιηση στο διάστημα [-1,1] πριν την εφαρμογή και την εκπαίδευση του μοντέλου διότι διαφέρουν σημαντικά ανά χαρακτηριστικό για αυτό τον σκοπό χρησιμοποιούμε την συνάρτηση:

$$X = 2 * \frac{X - \min(X)}{\max(X) - \min(X)} - 1$$

Χωρίζουμε τα δεδομένα σε δύο νέα pandas arrays ένα που περιέχει μόνο τα χαρακτηριστικά και ένα που περιέχει της κλάσης κατηγοριοποίησης Patient_data_Y contains the class data and Patient_data_X contains the features data.

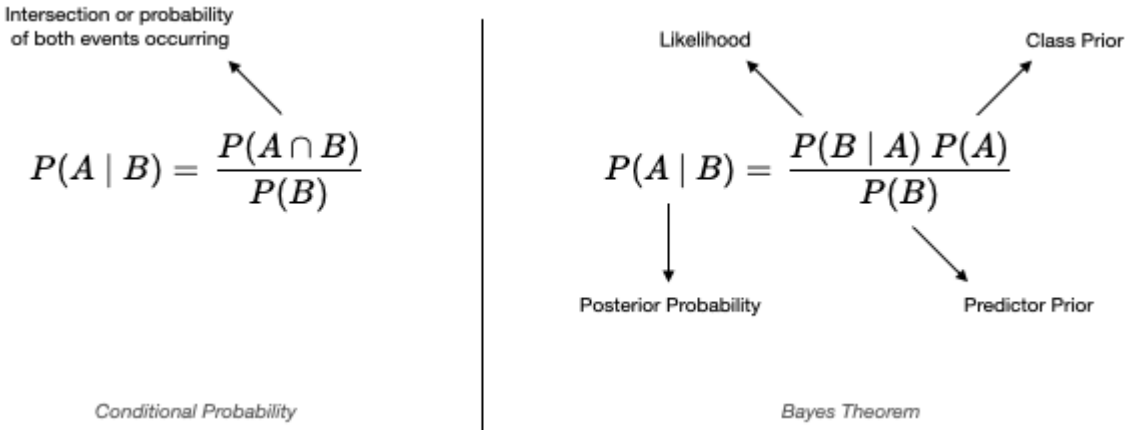
Με την χρήσει της μέθοδου train_test_split() χωρίζουμε τα δεδομένα σε σύνολα training and test data. Το πλαίσιο δεδομένων χωρίζεται σε Patient_data_X_train , Patient_data_X_test , Patient_data_Y_train και Patient_data_Y_test. Τα σετ Patient_data_X_train και Patient_data_Y_train χρησιμοποιούνται για την εκπαίδευση του μοντέλου. Τα σύνολα Patient_data_X_test και Patient_data_Y_test χρησιμοποιούνται για τη δοκιμή του μοντέλου εάν προβλέπει τις σωστές εξόδους/ετικέτες. Χωρίζουμε τα δεδομένα μας σε 80 % training και 20% split με randomness 210.

Ερώτημα 2.

Στο μάθημα συζητήθηκε εκτεταμένα ο ταξινομητής Bayes. Στη βιβλιογραφία, υπάρχει μια παραλλαγή του που λέγεται Αφελής Ταξινομητής Bayes (Naïve Bayes), με την υπόθεση ότι τα χαρακτηριστικά είναι στατιστικά ανεξάρτητα. Αναζητήστε τη σχετική βιβλιογραφία στο Internet, και να κάνετε μια σύντομη παρουσίαση του αλγορίθμου. Στη συνέχεια να κάνετε μια σύγκριση με τον Ταξινομητή Bayes.

θεώρημα Bayes

Για να εξηγήσουμε το Naive Bayes πρέπει πρώτα να εξηγήσουμε το θεώρημα Bayes. Το θεμέλιο του θεωρήματος Bayes είναι η υπό όρους πιθανότητα παρακάτω συνάρτηση . Στην πραγματικότητα, το θεώρημα Bayes είναι απλώς ένας εναλλακτικός ή αντίστροφος τρόπος για τον υπολογισμό της υπό όρους πιθανότητας. Όταν η κοινή πιθανότητα, P(A∩B), είναι δύσκολο να υπολογιστεί ή εάν η αντίστροφη ή η πιθανότητα Bayes, P(B|A), είναι ευκολότερο να υπολογιστεί, τότε μπορεί να εφαρμοστεί το θεώρημα Bayes.



Περεταίρω επεξήγηση εννοιών του θεώρημα Bayes:

Class prior or prior probability: πιθανότητα να συμβεί το συμβάν Α πριν μάθουμε οτιδήποτε για το γεγονός Β.

Predictor prior or evidence: ίδια με την προηγούμενη τάξη αλλά για το συμβάν Β.

Posterior probability: πιθανότητα του συμβάντος Α μετά την εκμάθηση του συμβάντος Β.

Likelihood: αντίστροφη της posterior πιθανότητας.

Η διάκριση μεταξύ του θεωρήματος Bayes και του Θεωρήματος του Naive Bayes είναι ότι ο Naive Bayes αναλαμβάνει την υπό όρους ανεξαρτησία εκεί όπου το θεώρημα Bayes δεν το κάνει. Αυτό σημαίνει ότι η σχέση μεταξύ όλων των χαρακτηριστικών εισόδου είναι ανεξάρτητη. Δεν είναι μια μεγάλη υπόθεση, αλλά αυτός είναι ο λόγος που ο αλγόριθμος ονομάζεται "αφελής" και είναι λόγος που ο αλγόριθμος είναι γρήγορος. Παρόλο που ο αλγόριθμος είναι "αφελής", μπορεί να έχει καλύτερη απόδοση από πιο πολύπλοκα μοντέλα όπως και θα δούμε παρακάτω (Part ii of project).

Ερώτημα 3.

Με χρήση της μεθόδου 5-fold cross validation, να εκπαιδεύσετε τον Naïve Bayes ταξινομητή, να παρουσιάστε και να σχολιάσετε την απόδοσή του. Μπορείτε να χρησιμοποιήσετε κατάλληλες συναρτήσεις του matlab ή οποιαδήποτε εφαρμογή επιθυμείτε (ή να υλοποιήσετε δικό σας κώδικα). Για την αξιολόγηση της απόδοσης του ταξινομητή να χρησιμοποιήσετε τις μετρικές του ερωτήματος 4, παρακάτω.

Μετρικές από εκφώνηση

Ποιο συγκεκριμένα ζητείται να παρουσιάσετε για κάθε ταξινομητή την μέση απόδοση του με χρήση 5 fold cross validation σε σχέση με την μετρική του γεωμετρικού μέσου (Geometric Mean) της ευαισθησίας (Sensitivity) και της ειδίκευσής (Specificity) του:

Geometric Mean = sqrt (Sensitivity * Specificity)

Η μετρική αυτή χρησιμοποιείται για προβλήματα ταξινόμησης όπου παραδείγματα εκπαίδευσης της μίας κλάσης είναι περισσότερα από τα παραδείγματα εκπαίδευσης της άλλης κλάσης.

Στη συνέχεια, παρουσιάστε τα ενδιάμεσα αποτελέσματα που πήρατε από τα πειράματα για την ρύθμιση των παραμέτρων των αλγορίθμων. Γιατί η κάθε μέθοδος ταξινόμησης δίνει διαφορετικά αποτελέσματα;

Αρχικα οριζουμε το GaussianNB() classifier από την Βιβλιοθηκη skit learn , τον όπιο τον εκπαιδεύουμε με τα δεδομένα των χαρακτηριστικών και των κλάσεων Patient_data_X_train Patient_data_Y_train αντίστοιχα. Εν συνεχεία με την χρήσει της συναρτήσεις predict από την skit_learn προβλέπουμε τιμές για ένα νέο data set δίνοντας τα Patient_data_X_test δεδομένα. Το παραπάνω όμως είναι για ένα ,split του data set όπως έγινε με την συνάρτηση train_test_split για να έχουμε καλύτερη εικόνα για την αξιοπιστία του μοντέλου χρησιμοποιούμε την μέθοδο 5 fold cross validation training. Η οποία υλοποιείτε με την χρήσει των συναρτήσεων KFold για την διάσπαση του συνόλου δεόμενων σε 5folds και cross_val_score η οποία εκπαιδεύει και αξιολογεί το μοντέλο για ποικίλα διαφορετικά folds 5 στην προκείμενη περίπτωση.

- Πρώτη εκπαίδευση γίνεται χωρίς τα λογαριθμικά δεδόμενα και περνούμε τις εξής μετρήσεις αξιολόγησης του μοντέλου NB:

- Trainig with one split (train_test_split 80% - 20%)

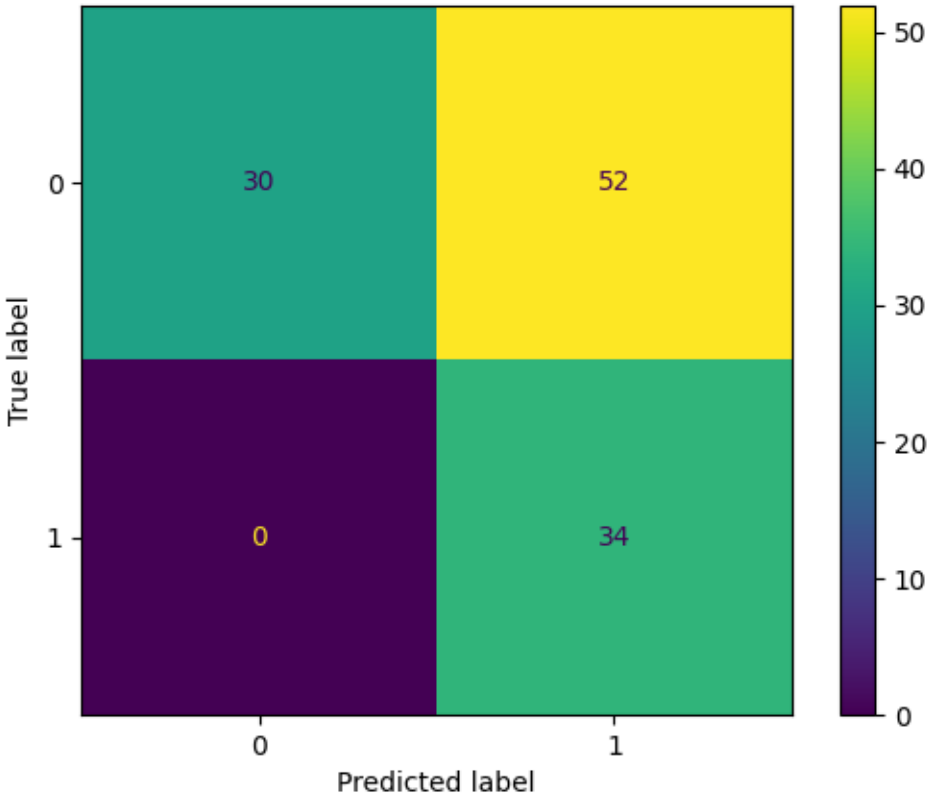
  Accuracy = 0.55

- 5 fold Cross Validation Training Method :

  All folds : [0.56, 0.525, 0.58, 0.525, 0.565]

  Average : 0.552

  Confusion Matrix : [[30 52] , [0 34]]



  Geometric Mean of sqrt (Sensitivity * Specificity):

  Sensitivity : 0.36

  Specificity : 1.0

  Geometric Mean: 0.604

Compiler Screen Screenshots

```
Accuracy :  0.5517241379310345
Sensitivity :  0.36585365853658536
Specificity :  1.0
The Geometric Mean is: 0.6048583789091339
Confusion Matrix : [[30 52]
 [ 0 34]]
```

```
All folds Scores  [0.56034483 0.52586207 0.5862069  0.52586207 0.56521739]
Average of all Folds Scores 0.5526986506746627
```

- Δευτερη εκπαίδευση γίνεται με τα λογαριθμικά δεδόμενα και πέρνουμε τις εξής μετρήσεις αξιολόγησης του μοντέλου Gaussian NB:

    - Trainig with one split (train_test_split 80% - 20%)

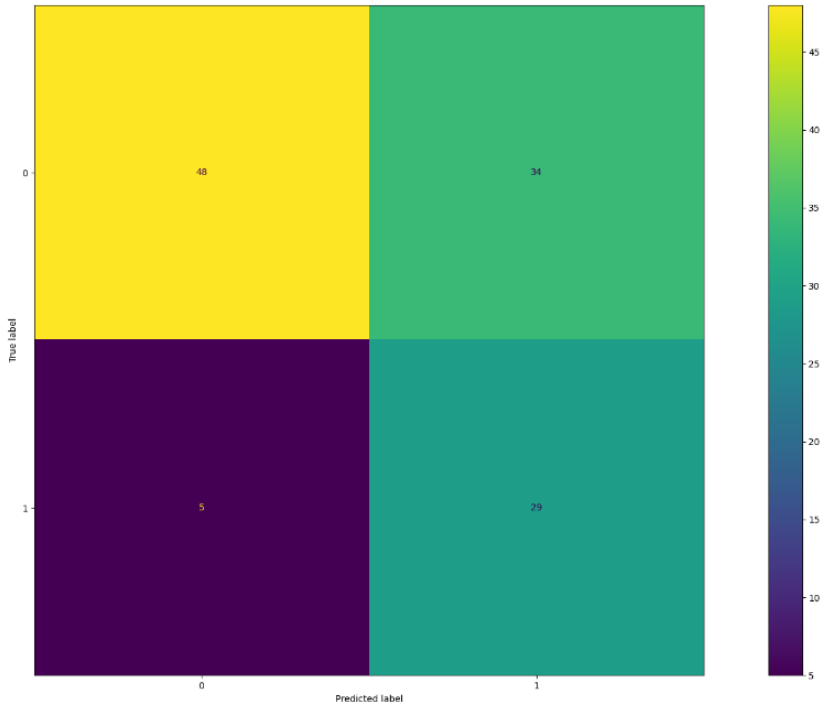
      Accuracy = 0.6637

    - 5 fold Cross Validation Training Method :

      All folds : [0.65517241 0.5862069 0.64655172 0.65517241 0.65217391]

      Average : 0.639

      Confusion Matrix : [[48 34], [5 29]]



Geometric Mean of sqrt (Sensitivity * Specificity):

Sensitivity : 0.585

Specificity : 0.852

Geometric Mean: 0.7065993489136699

Compiler Screen Screenshots

```
All folds Scores  [0.65517241 0.5862069  0.64655172 0.65517241 0.65217391]
Average of all Folds Scores 0.6390554722638682
```

```
Accuracy :  0.6637931034482759
Sensitivity :  0.5853658536585366
Specificity :  0.8529411764705882
The Geometric Mean is: 0.7065993489136699
Confusion Matrix : [[48 34]
 [ 5 29]]
```

- Τρίτη εκπαίδευση γίνεται με τα λογαριθμικά δεδομένα και με τα δεδομένα με χαρακτηριστικά με μικρότερη εξάρτηση μεταξύ τους.

  Τα χαρακτηριστικά υψηλής συσχέτισης υπολογίζονται δύο φορές στο μοντέλο. Η διπλή καταμέτρηση οδηγεί σε υπερεκτίμηση της σημασίας αυτών των χαρακτηριστικών. Έτσι, η απόδοση του αταξινόμητή Naive Bayes υποβαθμίζεται. Η εξάλειψη των σχετιζόμενων χαρακτηριστικών βοηθάει στην τήρηση της υπόθεσης των ανεξάρτητων χαρακτηριστικών. Για να τα εντοπίσουμε και να τα εξαλείψουμε, χρησιμοποιήσουμε έναν πίνακα συσχέτισης heatmap για να συγκρίνουμε ζεύγη χαρακτηριστικών. Τα χαρακτηριστικά των δεδομένων με την μικρότερη εξάρτησει , από το heatmap είναι τα εξείς Age (0), Gender (1) και Alkphos Alkaline Phosphotase (5).
  Παίρνουμε τις εξής μετρήσεις αξιολόγησης του μοντέλου Gaussian NB:

  - Trainig with one split (train_test_split 80% - 20%)

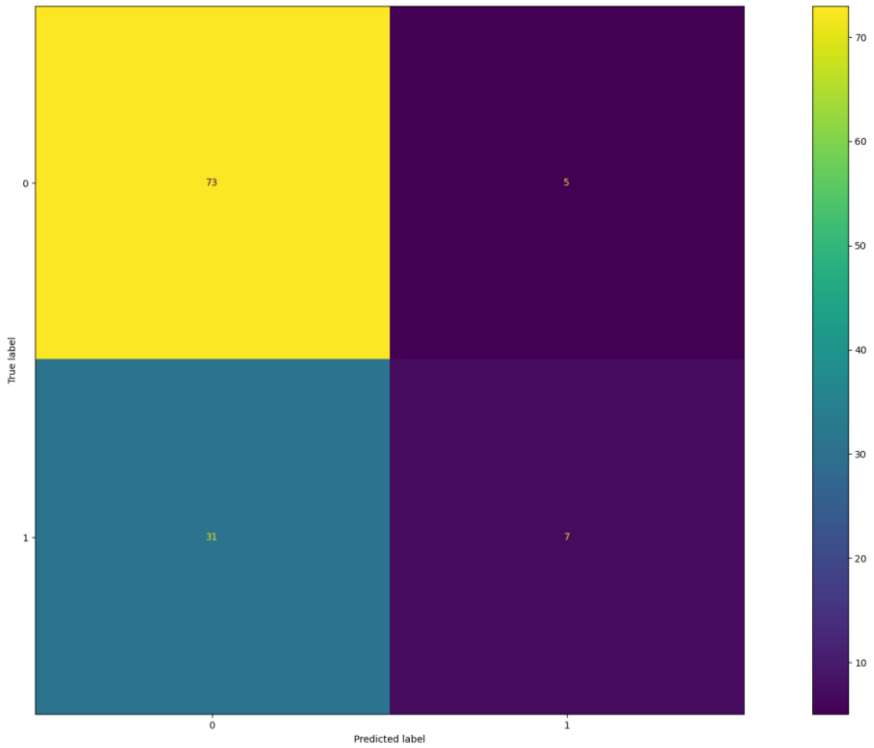
    Reduced Dependency Accuracy : 0.68965

  - 5 fold Cross Validation Training Method :

    Reduced Dependency all folds Scores [0.68965517 0.74137931 0.65517241 0.6637931 0.75652174]

    Reduced Dependency Average of all Folds Scores 0.70

    Reduced Dependency Confusion Matrix : [[73 5],[31 7]]



Geometric Mean of sqrt (Sensitivity * Specificity):

Sensitivity : 0.935

Specificity : 0.184

Geometric Mean: 0.689

Compiler Screen Screenshots

```
Reduced Dependency all folds Scores  [0.68965517 0.74137931 0.65517241 0.6637931  0.75652174]
Reduced Dependency Average of all Folds Scores  0.701304347826087
```

```
Reduced Dependency Accuracy :  0.6896551724137931
Reduced Dependency Confusion Matrix : [[73  5]
 [31  7]]
Reduced Dependency Sensitivity :  0.9358974358974359
Reduced Dependency Specificity :  0.18421052631578946
Reduced Dependency Geometric Mean is: 0.41521339001080454
```

CONCLUSIONS

Από τις παραπάνω μετρήσεις παρατηρούμε ότι το πιο ακριβές μοντέλο είναι το τελευταίο του οποίου η εκπαίδευση γίνεται με τα λογαριθμικά δεδομένα και με τα δεδομένα με χαρακτηριστικά με μικρότερη εξάρτηση μεταξύ τους. Επίσης παρατηρούμε ότι το sensitivity του που είναι η μετρική που μας λέει το ποσοστό των liver patients που έγιναν classify ορθά είναι αρκετά πιο υψηλή από τα υπόλοιπα εκπαιδευμένα μοντέλα πράγμα που είναι θετικό διότι είναι προτιμότερο να κάνουμε classify έναν ασθενή ότι έχει ασθένεια στο συκώτι από το να γίνει misclassify και να θέσουμε σε κίνδυνο την ζωή του.

Python CODE PART I NB – K FOLD

```
#Importing the necessary libraries
#Numpy for the math Operations
import numpy as np
#Pandas library for data preprocessing
import pandas as pd
#Matplotlib and seaborn for plotting the data
import matplotlib.pyplot as plt
import seaborn as sns
#skit_learn library from model_selection package importing train_test_split in order
to split the arrays into random train and test subsets.
from sklearn.model_selection import train_test_split
#skit_learn library from naive_bayes package we use GaussianNB in order to implement
the naive bayes classifier.
from sklearn.naive_bayes import GaussianNB
#skit_learn library from metrics package importing confusion_matrix, accuracy_score in
order to evaluate the model and ConfusionMatrixDisplay for plotting the matrix.
from sklearn.metrics import confusion_matrix,accuracy_score , ConfusionMatrixDisplay
#skit_learn library from model_selection package importing cross_val_score in order to
use the k-fold cross validation method to evaluate our model with multiple folds.
from sklearn.model_selection import cross_val_score , KFold

#------------------------------------------------------- Data Preprocessing Erotima 1
-----------------------------------------------------------------------

#Importing Project data from csv using the Pandas dataframe
Patient_data = pd.read_csv("Indian Liver Patient Dataset (ILPD).csv" , header=None )
```

```
#Using replace function from Pandas data frame to encode the genders with the numeric
values 0 Male 1 Female
Patient_data[1].replace('Male' ,0,inplace=True)
Patient_data[1].replace('Female' ,1,inplace=True)

#Using replace function from Pandas data frame to replace the Class encoding with 0=1
, 1=2
Patient_data[10].replace(1 ,0,inplace=True)
Patient_data[10].replace(2 ,1,inplace=True)

#Displaying all the data from the Pandas dataframe using to_string() method
print(Patient_data.to_string())

#Using dropna method from Pandas dataframe to remove all the rows that contains NULL
or NAN values.
Patient_data=Patient_data.dropna()

#Using value_counts method from Pandas dataframe to return an object with the counts
of unique values of the array .
print(Patient_data[10].value_counts())


#Using hist method from Pandas dataframe to display a histogram a representation of
the distribution of data
#for the last column of the data array which contains the classes.
Patient_data[10].hist(color = "darkCyan")
plt.show()


#Using the seaborn and matplotlib libraries we plot 10 histograms one per column
(features) of the data array.
fig,axes = plt.subplots(1 ,10 ,figsize=(60,15),sharey=True)
sns.histplot(Patient_data , ax =axes[0] , x=0 , kde=True , color =
'#a1c9f4').set(title='Age')
sns.histplot(Patient_data , ax =axes[1] , x=1, kde=True , color =
'#8de5a1').set(title='Gender')
sns.histplot(Patient_data , ax =axes[2] , x=2 , kde=True , color
=  '#B0171F').set(title='TB')
sns.histplot(Patient_data , ax =axes[3] , x=3 , kde=True, color = '#d0bbff'
).set(title='DB')
sns.histplot(Patient_data , ax =axes[4] , x=4 , kde=True , color =
'#D02090').set(title='Alkphos Alkaline Phosphotase')
sns.histplot(Patient_data , ax =axes[5] , x=5 , kde=True , color =
'#1C86EE').set(title='Sgpt')
sns.histplot(Patient_data , ax =axes[6] , x=6 , kde=True, color =
'#00688B').set(title='Sgot')
sns.histplot(Patient_data , ax =axes[7] , x=7 , kde=True , color =
'#EE7600').set(title='TP')
```

```
sns.histplot(Patient_data , ax =axes[8] , x=8 , kde=True, color =
'#8E388E').set(title='ALB')
sns.histplot(Patient_data , ax =axes[9] , x=9 , kde=True , color = '#CD2626')
.set(title='A/G')
plt.show()


#The log() numpy method a mathematical function that calculates natural logarithm of
echa allmento
#of the array here for the first feature
Patient_data[0] = np.log(Patient_data[0])

#The log() numpy method a mathematical function that calculates natural logarithm of
each element of the array
#for the remaining features expect the gender.
for i in range(2,10):
 Patient_data[i] = np.log(Patient_data[i])


#Using the seaborn and matplotlib libraries we plot 10 histograms one per column
(features)
#of the data array in order to see the result of the logarithmic process of the data.
fig,axes = plt.subplots(1 ,10 ,figsize=(60,15),sharey=True)
sns.histplot(Patient_data , ax =axes[0] , x=0 , kde=True , color =
'#a1c9f4').set(title='Age')
sns.histplot(Patient_data , ax =axes[1] , x=1, kde=True , color =
'#8de5a1').set(title='Gender')
sns.histplot(Patient_data , ax =axes[2] , x=2 , kde=True , color
=  '#B0171F').set(title='TB')
sns.histplot(Patient_data , ax =axes[3] , x=3 , kde=True, color = '#d0bbff'
).set(title='DB')
sns.histplot(Patient_data , ax =axes[4] , x=4 , kde=True , color =
'#D02090').set(title='Alkphos Alkaline Phosphotase')
sns.histplot(Patient_data , ax =axes[5] , x=5 , kde=True , color =
'#1C86EE').set(title='Sgpt')
sns.histplot(Patient_data , ax =axes[6] , x=6 , kde=True, color =
'#00688B').set(title='Sgot')
sns.histplot(Patient_data , ax =axes[7] , x=7 , kde=True , color =
'#EE7600').set(title='TP')
sns.histplot(Patient_data , ax =axes[8] , x=8 , kde=True, color =
'#8E388E').set(title='ALB')
sns.histplot(Patient_data , ax =axes[9] , x=9 , kde=True , color = '#CD2626')
.set(title='A/G')
plt.show()

#using seaborn heatmap method in order to plot the heatmap of the features in order to
see the dependency of each feature with the another.
corr = Patient_data.iloc[:,:-1].corr(method='pearson')
```

```
cmap = sns.diverging_palette(220, 20, sep=20, as_cmap=True)
sns.heatmap(corr , vmax =1 , vmin =.3 , cmap=cmap ,  annot=True,square =True ,
linewidths = .2)
plt.show()


#Normalise the data of each feature between -1 and 1 expect the gender (column with
index 1)
Patient_data[0] = 2*((Patient_data[0] - min(Patient_data[0])) / ( max(Patient_data[0])
- min(Patient_data[0]) ))-1
for i in range(2,10):
    Patient_data[i] = 2*((Patient_data[i] - min(Patient_data[i])) / (
max(Patient_data[i]) - min(Patient_data[i]) ))-1


#separate the data in two pandas data frames Patient_data_Y contains the class data
#and Patient_data_X contains the features data
Patient_data_Y = Patient_data.iloc[:,10:]
Patient_data_X = Patient_data.iloc[:,:-1]

# Split dataset into training set and test set
# 80% training and 20% test
Patient_data_X_train, Patient_data_X_test, Patient_data_Y_train, Patient_data_Y_test =
train_test_split(Patient_data_X, Patient_data_Y, test_size=0.2,random_state=210)

#Begin the process of implementing Naive Bayes Classifier Erotima 2.

# Using GaussianNB() classifier from scikit-learn library
#in order to implement the Gaussian Naïve Bayes algorithm for classification.
classifier = GaussianNB()

# The fit method from scikit-learn library trains the algorithm on the training data,
after the model is initialized.
classifier.fit(Patient_data_X_train, Patient_data_Y_train)

# the Predict Method from scikit-learn library given a trained model, it predicts the
label of a new set of data.
y_pred = classifier.predict(Patient_data_X_test)

# The accuracy_score method from scikit-learn is a function that computes subset
accuracy: the set of labels predicted
# for a sample must exactly match the corresponding set of labels in
Patient_data_Y_test
ac = accuracy_score(Patient_data_Y_test ,y_pred)

# The Predict Method from scikit-learn library given a trained model, it predicts the
label of a new set of data.
cm = confusion_matrix(Patient_data_Y_test , y_pred)
```

```
print('Accuracy : ' , ac)

# Computes the sensitivity of the algorithm from the confusion matrix
sensitivity = cm[0,0]/(cm[0,0]+cm[0,1])
print('Sensitivity : ', sensitivity )

# Computes the specificity of the algorithm from the confusion matrix
specificity = cm[1,1]/(cm[1,0]+cm[1,1])
print('Specificity : ', specificity)

# Computes the geometric mean from the sensitivity and specificity of the algorithm
geometric_mean = (sensitivity*specificity)**(1/2)
print ('The Geometric Mean is: ' + str(geometric_mean))

print('Confusion Matrix :' , cm)

# Ploting Confusion matrix
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=classifier.classes_)
disp.plot()
plt.show()

#implementing the naïve bayes algorithm using the 5-fold cross validation technique
Erotima 3
#K-Folds cross-validator Pprovides train/test indices to split data in train/test
sets. Split dataset into k consecutive folds
cv = KFold(n_splits=5, random_state=1, shuffle=True)

#cross_val_score is a function in the scikit-learn package which trains and tests a
model over multiple folds of your dataset
scores = cross_val_score(classifier, Patient_data_X, Patient_data_Y,
scoring='accuracy', cv=cv, n_jobs=-1)

#Show accuracy for each fold performance
print('All folds Scores ',scores)

print('Average of all Folds Scores' ,np.average(scores))

#||--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------||
#||--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------||
#||--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------||
```

```
#||-----------------------------------------------------------------------------
-----------------------------------------------------------------------------
--------------||


#After using the heatmap to determine the dependency between the feature of the data
set we retrain the classifier
#This new naive bayes classifier is called reduced_dependency (rd) and all new
variables are called rd_previous_name


#Creating a new training data set using the features (ccolumns) with the least
correlation with the other features
Patient_data_X_rd = Patient_data_X[[0,1,4]]

# Split dataset into training set and test set
Patient_data_X_train_rd, Patient_data_X_test_rd, Patient_data_Y_train_rd,
Patient_data_Y_test_rd = train_test_split(Patient_data_X_rd, Patient_data_Y,
test_size=0.2,random_state=218)

# Using GaussianNB() classifier from scikit-learn library
#in order to implement the Gaussian Naïve Bayes algorithm for classification for
reduced dependency data.
classifier_reduced_dependency =  GaussianNB()

#The fit method from scikit-learn library trains the algorithm on the training data,
after the model is initialized.
classifier_reduced_dependency.fit(Patient_data_X_train_rd, Patient_data_Y_train_rd)

# the Predict Method from scikit-learn library given a trained model, it predicts the
label of a new set of data.
y_pred_rd = classifier_reduced_dependency.predict(Patient_data_X_test_rd)

# The accuracy_score method from scikit-learn is a function that computes subset
accuracy: the set of labels predicted
# for a sample must exactly match the corresponding set of labels in
Patient_data_Y_test
ac_rd = accuracy_score(Patient_data_Y_test_rd ,y_pred_rd)

# Computes the geometric mean from the sensitivity and specificity of the algorithm NB
with reduced dpendancie data
cm_rd = confusion_matrix(Patient_data_Y_test_rd, y_pred_rd)


print ('Reduced Dependency Accuracy : ' , ac_rd)
print('Reduced Dependency Confusion Matrix :' , cm_rd)

# Computes the sensitivity of the algorithm from the confusion matrix
```

```
sensitivity_rd = cm_rd[0,0]/(cm_rd[0,0]+cm_rd[0,1])
print('Reduced Dependency Sensitivity : ', sensitivity_rd )

# Computes the Specificity of the algorithm from the confusion matrix
specificity_rd = cm_rd[1,1]/(cm_rd[1,0]+cm_rd[1,1])
print('Reduced Dependency Specificity : ', specificity_rd)

# Computes the geometric mean from the sensitivity and specificity of the algorithm
geometric_mean_rd = (sensitivity_rd*specificity_rd)**(1/2)
print ('Reduced Dependency Geometric Mean is: ' + str(geometric_mean_rd))

# Ploting Confusion matrix
disp_rd = ConfusionMatrixDisplay(confusion_matrix=cm_rd,
display_labels=classifier_reduced_dependency.classes_)
disp_rd.plot()
plt.show()

#implementing the naïve bayes algorithm using the 5-fold cross validation technique
Erotima 3

#K-Folds cross-validator Pprovides train/test indices to split data in train/test
sets. Split dataset into k consecutive folds
cv_rd = KFold(n_splits=5, random_state=1, shuffle=True)

#cross_val_score is a function in the scikit-learn package which trains and tests a
model over multiple folds of your dataset
scores_rd = cross_val_score(classifier_reduced_dependency, Patient_data_X_rd,
Patient_data_Y, scoring='accuracy', cv=cv, n_jobs=-1)

#Show accuracy for each fold performance
print('Reduced Dependency all folds Scores ' , scores_rd)
print('Reduced Dependency Average of all Folds Scores ' , np.average(scores_rd))
```

Bibliography

https://pandas.pydata.org/docs/

https://numpy.org/doc/

https://scikit-learn.org/stable/

https://scikit-learn.org/stable/modules/naive_bayes.html

https://scikit-learn.org/stable/modules/model_evaluation.html

https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.hist2d.html

https://seaborn.pydata.org/

https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html

https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html