ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

“Κατηγοριοποίηση Naive Bayes σε αριθμητικά δεδομένα ”

Του φοιτητή
Πετρούση Ανδρέα
Αρ. Μητρώου: 123893

Επιβλέπων
Ουγιάρογλου Στέφανος
Επ. Καθηγητής

26 Ιανουαρίου 2025

Τίτλος Π.Ε.: Κατηγοριοποίηση Naive Bayes σε αριθμητικά δεδομένα
Κωδικός Π.Ε. 24114
Ονοματεπώνυμο φοιτητή: Ανδρέας Πετρούσης
Ονοματεπώνυμο εισηγητή: Ουγιάρογλου Στέφανος
Ημερομηνία ανάληψης Δ.Ε.: 26-01-2024
Ημερομηνία περάτωσης Δ.Ε.:26-01-2025

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως πτυχιακή εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

Η έγκριση της διπλωματικής εργασίας από το Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου της Ελλάδος, δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του συγγραφέα, εκ μέρους του Τμήματος.

Πρόλογος

Η επιλογή του θέματος αυτής της εργασίας πηγάζει από το έντονο ενδιαφέρον μου για την κατηγοριοποίηση δεδομένων, έναν από τους πιο σημαντικούς και εφαρμοσμένους τομείς της μηχανικής μάθησης. Η αυξανόμενη σημασία της ανάλυσης δεδομένων στη σύγχρονη εποχή καθιστά την κατανόηση και την εφαρμογή αλγορίθμων κατηγοριοποίησης ένα απαραίτητο εργαλείο για την επίλυση πραγματικών προβλημάτων. Ο Naive Bayes, παρά την απλότητά του, έχει διαχρονικά αποδείξει τη χρησιμότητά του σε πληθώρα περιπτώσεων, γεγονός που με ώθησε να εστιάσω σε αυτόν.

Η απόφαση να συγκρίνω τον Gaussian Naive Bayes με τη βασική εκδοχή του Naive Bayes σχετίζεται με το ενδιαφέρον μου να εξετάσω πώς η μορφή των δεδομένων, συνεχής ή κατηγορική, επηρεάζει την απόδοσή τους. Παράλληλα, η διαδικασία της διακριτοποίησης ως μέθοδος προεπεξεργασίας αποτελεί μια ενδιαφέρουσα πτυχή που συνδυάζει τη θεωρητική γνώση με την πρακτική εφαρμογή. Μέσα από αυτή τη μελέτη, στοχεύω να αποκτήσω βαθύτερη κατανόηση των χαρακτηριστικών των αλγορίθμων αυτών και να ενισχύσω τις γνώσεις μου στη μηχανική μάθηση, ανοίγοντας νέους δρόμους για μελλοντικές ερευνητικές ή επαγγελματικές εφαρμογές.

Περίληψη

Η συγκεκριμένη εργασία επικεντρώνεται στην αξιολόγηση της απόδοσης του αλγορίθμου Naive Bayes σε περιπτώσεις κατηγοριοποίησης, τόσο όταν τα δεδομένα είναι συνεχούς μορφής όσο και όταν αυτά μετατρέπονται σε κατηγορικά. Βασικός στόχος είναι η μελέτη της εφαρμογής του Gaussian Naive Bayes σε δεδομένα με συνεχή χαρακτηριστικά και η σύγκριση των αποτελεσμάτων του με εκείνα της βασικής εκδοχής του Naive Bayes, αφού πρώτα τα δεδομένα έχουν μετατραπεί σε διακριτή μορφή. Αρχικά, ο Gaussian Naive Bayes θα εφαρμοστεί σε διάφορα datasets που περιέχουν συνεχείς μεταβλητές, και η απόδοσή του θα εκτιμηθεί με γνώμονα την ακρίβεια. Η επιλογή των δεδομένων θα είναι τέτοια ώστε να καλύπτονται διαφορετικοί τύποι προβλημάτων, επιτρέποντας έτσι την αξιολόγηση της γενικής ικανότητας του μοντέλου να διαχειρίζεται δεδομένα με συνεχή χαρακτηριστικά. Στη συνέχεια, τα ίδια datasets θα υποστούν μετατροπή από συνεχείς σε κατηγορικές τιμές με τη χρήση μεθόδων διακριτοποίησης, όπου οι αρχικές τιμές θα αντικατασταθούν από ομάδες ή διαστήματα. Ακολούθως, η βασική εκδοχή του Naive Bayes, που προϋποθέτει διακριτά δεδομένα, θα εφαρμοστεί στα τροποποιημένα αυτά δεδομένα και τα αποτελέσματά της θα αξιολογηθούν με τα ίδια μέτρα ώστε να διασφαλιστεί η άμεση συγκρισιμότητα με τη μέθοδο του Gaussian Naive Bayes. Η ανάλυση θα εστιάσει στη σύγκριση των δύο προσεγγίσεων, αναδεικνύοντας ποια από αυτές εμφανίζεται πιο αποτελεσματική ανάλογα με το είδος των δεδομένων και τις απαιτήσεις του εκάστοτε προβλήματος. Ένας από τους βασικούς άξονες της σύγκρισης θα είναι να διαπιστωθεί αν η μετατροπή συνεχών δεδομένων σε κατηγορικά παρέχει ουσιαστικά πλεονεκτήματα, ή αν η χρήση του Gaussian Naive Bayes είναι επαρκής για την επίτευξη ικανοποιητικής απόδοσης. Τα αποτελέσματα θα παρουσιαστούν σε συνάρτηση με τα χαρακτηριστικά των δεδομένων, ενώ θα γίνει και αναφορά σε περιορισμούς ή πιθανά μειονεκτήματα της κάθε μεθόδου. Στόχος είναι να εξαχθούν συμπεράσματα που θα βοηθήσουν στην επιλογή της καταλληλότερης μεθόδου για κάθε πρόβλημα, αναδεικνύοντας τα οφέλη και τις προκλήσεις της διακριτοποίησης, καθώς και την αποτελεσματικότητα του Gaussian Naive Bayes.

«Naive Bayes classification in numeral data»

Andreas Petrousis

Abstract

This study focuses on evaluating the performance of the Naive Bayes algorithm in classification tasks, both when the data is in continuous form and when it is transformed into categorical form. The main objective is to examine the application of Gaussian Naive Bayes to datasets with continuous features and to compare its results with those of the standard Naive Bayes, following the transformation of the data into discrete form. Initially, Gaussian Naive Bayes will be applied to various datasets containing continuous variables, and its performance will be assessed using metrics such as accuracy and efficiency. The selection of datasets will aim to cover different types of problems, thus enabling an evaluation of the model's general ability to handle continuous data effectively. Subsequently, the same datasets will be transformed from continuous to categorical values using discretization techniques, where the original values will be replaced with ranges or intervals. The standard version of Naive Bayes, which assumes discrete data, will then be applied to these transformed datasets, and its results will be evaluated using the same metrics to ensure direct comparability with Gaussian Naive Bayes. The analysis will focus on comparing the two approaches, highlighting which one proves more effective depending on the nature of the data and the requirements of the specific problem. One of the main aspects of this comparison will be to determine whether converting continuous data to categorical offers significant advantages or whether Gaussian Naive Bayes is sufficient to achieve satisfactory performance. The results will be presented in relation to the characteristics of the data, with a discussion of the limitations or potential drawbacks of each method. The aim is to draw conclusions that will assist in selecting the most suitable method for each problem, emphasizing the benefits and challenges of discretization as well as the effectiveness of Gaussian Naive Bayes.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω πρωτίστως τους γονείς μου, που έχουν αποτελέσει τις δύο θεμέλιες κολώνες, πάνω στις οποίες έχω χτίσει την ζωή μου· τα αδέλφια μου, Ελευθερία, Κωνσταντίνο, Ευθύμη, Ιωάννη και τον Νίκο, τον επίκτητο αδελφό· τους φίλους μου Γιώργο, Ζωή, Χρυσόστομο, Γιάννη, Κώστα, Βασίλη, Τάκη, Τζουλιάνο και Νίκο (τα ονόματα μετά από συγχώνευση διπλότυπων)· την εξέχουσα συνάδελφο Αφροδίτη και τον πολυγραφότατο, αγαπητό συνάδελφο Δημήτρη τον οποίο σαν μούσα επικαλέστηκα για την συγγραφή αυτής της εργασίας. Τέλος θέλω να ευχαριστήσω τον κύριο Στέφανο Ουγιάρογλου, ο οποίος μου έδωσε την ευκαιρία, αλλά και την έμπνευση για το δημιουργικότερο ίσως μέχρι τώρα έργο μου.

Σας ευχαριστώ

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κεφάλαιο 1ο: Εισαγωγή

1.1 Κατηγοριοποίηση Δεδομένων

1.2 Κατηγοριοποίηση Δεδομένων

Η κατηγοριοποίηση δεδομένων είναι μια από τις πιο βασικές και σημαντικές τεχνικές στη μηχανική μάθηση και την επιστήμη δεδομένων. Ο στόχος της είναι να ταξινομήσει ένα δείγμα ή μια παρατήρηση σε μία από τις προκαθορισμένες κατηγορίες με βάση τα χαρακτηριστικά του. Αυτή η διαδικασία είναι πολύ χρήσιμη σε πολλούς τομείς, όπως η ιατρική, η χρηματοοικονομία και η τεχνητή νοημοσύνη. Η κατηγοριοποίηση βοηθάει στην αναγνώριση μοτίβων στα δεδομένα και επιτρέπει τη λήψη καλύτερων αποφάσεων, τον εντοπισμό ανωμαλιών και την εξαγωγή χρήσιμων πληροφοριών από μεγάλα σύνολα δεδομένων.

Στην ιατρική, η κατηγοριοποίηση δεδομένων μπορεί να χρησιμοποιηθεί για να βοηθήσει στη διάγνωση ασθενειών. Για παράδειγμα, τα δεδομένα από ιατρικές εξετάσεις και το ιστορικό των ασθενών μπορούν να αναλυθούν ώστε να εντοπιστούν μοτίβα που συνδέονται με συγκεκριμένες ασθένειες (1). Με αυτόν τον τρόπο, οι γιατροί μπορούν να εντοπίζουν πιθανές ασθένειες νωρίτερα, επιτρέποντας την πρόληψη ή τη γρηγορότερη αντιμετώπισή τους. Παράδειγμα εφαρμογής είναι η χρήση αλγορίθμων για την ανίχνευση όγκων σε ακτινογραφίες, όπου η ακρίβεια είναι κρίσιμη για την υγεία των ασθενών.

Στον χρηματοοικονομικό τομέα, η κατηγοριοποίηση δεδομένων χρησιμοποιείται για την ανίχνευση απάτης στις συναλλαγές. Οι αλγόριθμοι αναλύουν πληροφορίες όπως το ποσό, την τοποθεσία και τη συχνότητα των συναλλαγών για να αναγνωρίσουν ύποπτες δραστηριότητες. Αυτό βοηθά τις τράπεζες να εντοπίζουν και να αποτρέπουν απάτες, βελτιώνοντας την ασφάλεια των χρηματοοικονομικών συναλλαγών.

Στην τεχνητή νοημοσύνη, η κατηγοριοποίηση δεδομένων χρησιμοποιείται σε εφαρμογές όπως η ανάλυση συναισθημάτων και η σύσταση προϊόντων. Για παράδειγμα, οι αλγόριθμοι ανάλυσης συναισθημάτων μπορούν να αναγνωρίσουν τις αντιδράσεις των χρηστών στα κοινωνικά δίκτυα ή σε κριτικές προϊόντων (2). Επίσης, η κατηγοριοποίηση βοηθά στη σύσταση προϊόντων σε ηλεκτρονικά καταστήματα, προτείνοντας αντικείμενα που είναι πιθανό να ενδιαφέρουν τον χρήστη, βάσει προηγούμενων επιλογών του.

Η διαδικασία κατηγοριοποίησης περιλαμβάνει αρκετά στάδια. Αρχικά, είναι απαραίτητος ο καθαρισμός των δεδομένων, ώστε να αφαιρεθεί ο θόρυβος και να αντιμετωπιστούν ελλιπείς ή λανθασμένες τιμές. Στη συνέχεια, τα δεδομένα κανονικοποιούνται για να εξασφαλιστεί ότι όλα τα χαρακτηριστικά έχουν την ίδια κλίμακα, κάτι που μπορεί να βελτιώσει την ακρίβεια του αλγορίθμου (3).

Η επιλογή του κατάλληλου αλγορίθμου κατηγοριοποίησης εξαρτάται από τα δεδομένα και τις απαιτήσεις του προβλήματος. Για παράδειγμα, η λογιστική παλινδρόμηση (4) είναι κατάλληλη για απλά προβλήματα, ενώ πιο πολύπλοκοι αλγόριθμοι, όπως τα Bayesian Networks (5), μπορούν να χειριστούν περισσότερες εξαρτήσεις στα δεδομένα. Αφού εκπαιδευτεί το μοντέλο, η απόδοσή του αξιολογείται με μετρικές όπως το F1-score, που συνδυάζει την ακρίβεια και την ανάκληση (6).

Η κατηγοριοποίηση δεδομένων αποτελεί ένα σημαντικό εργαλείο για την ανάλυση δεδομένων, με εφαρ-

μογές που επηρεάζουν πολλούς τομείς. Με τη συνεχή εξέλιξη των αλγορίθμων και της ποιότητας των δεδομένων, οι δυνατότητες της κατηγοριοποίησης συνεχίζουν να διευρύνονται (7).

1.3 Αλγόριθμοι κατηγοριοποίησης που βασίζονται σε πιθανότητες

Οι αλγόριθμοι κατηγοριοποίησης που βασίζονται σε πιθανότητες χρησιμοποιούν στατιστικές μεθόδους για να κατατάξουν δεδομένα σε κατηγορίες. Αυτοί οι αλγόριθμοι υπολογίζουν την πιθανότητα ότι ένα δείγμα ανήκει σε μια συγκεκριμένη κατηγορία, βασιζόμενοι σε δεδομένα εκπαίδευσης. Στην κατηγορία αυτή περιλαμβάνονται διάφορες παραλλαγές του Naive Bayes, καθεμία από τις οποίες έχει σχεδιαστεί για διαφορετικούς τύπους δεδομένων.

Η **Λογιστική Παλινδρόμηση** είναι ένας από τους πιο γνωστούς αλγόριθμους ταξινόμησης. Χρησιμοποιεί μια μαθηματική συνάρτηση για να εκτιμήσει την πιθανότητα ότι ένα δείγμα ανήκει σε μία από τις δύο κατηγορίες. Παρόλο που δεν είναι «καθαρά» πιθανοτικός αλγόριθμος, βασίζεται στις πιθανότητες για τη λήψη αποφάσεων. Χρησιμοποιείται ευρέως λόγω της απλότητας και της δυνατότητας να επεκτείνεται σε προβλήματα με περισσότερες από δύο κατηγορίες (8).

Ο **Gaussian Naive Bayes** είναι μια παραλλαγή που θεωρεί ότι τα δεδομένα ακολουθούν μια συγκεκριμένη κατανομή. Είναι ιδιαίτερα κατάλληλος για δεδομένα συνεχούς μορφής, όπως μετρήσεις ή αριθμητικά χαρακτηριστικά. Ο αλγόριθμος αυτός είναι γρήγορος, εύκολος στην υλοποίηση και λειτουργεί καλά ακόμη και με μικρά σύνολα δεδομένων (9).

Ο **Multinomial Naive Bayes** είναι κατάλληλος για δεδομένα που περιλαμβάνουν μετρήσεις ή συχνότητες, όπως η εμφάνιση λέξεων σε ένα κείμενο. Αυτός ο αλγόριθμος χρησιμοποιείται συχνά στην κατηγοριοποίηση κειμένων, όπου μπορεί να αναλύσει αποτελεσματικά τα δεδομένα και να ταξινομήσει τα έγγραφα σε κατηγορίες (10).

Ο **Bernoulli Naive Bayes** επικεντρώνεται σε δεδομένα που μπορούν να αναπαρασταθούν ως δυαδικά, δηλαδή ως «ναι» ή «όχι». Αυτό τον καθιστά ιδανικό για προβλήματα όπως το φιλτράρισμα ανεπιθύμητης αλληλογραφίας, όπου κάθε χαρακτηριστικό αντιπροσωπεύει την παρουσία ή απουσία μιας λέξης σε ένα μήνυμα (11).

Τέλος, ο **Categorical Naive Bayes** έχει σχεδιαστεί για δεδομένα που αποτελούνται από κατηγορίες αντί για αριθμούς. Χρησιμοποιείται σε περιπτώσεις όπου τα χαρακτηριστικά έχουν συγκεκριμένες κατηγορίες, όπως χρώματα ή κατηγορίες προϊόντων (12).

Αυτοί οι αλγόριθμοι, με τις διαφορετικές προσεγγίσεις τους, καλύπτουν ένα ευρύ φάσμα προβλημάτων ταξινόμησης. Η επιλογή του κατάλληλου αλγορίθμου εξαρτάται από τη φύση των δεδομένων και το είδος του προβλήματος που προσπαθούμε να λύσουμε.

1.4 Κίνητρο και Συνεισφορά

Αυτή η εργασία έχει ως στόχο να εξετάσει πιο οργανωμένα τους αλγόριθμους κατηγοριοποίησης, που αποτελούν βασικό εργαλείο στην ανάλυση δεδομένων. Σήμερα, η επεξεργασία μεγάλων και σύνθετων συνόλων δεδομένων είναι μια καθημερινή πρόκληση, και η κατανόηση αυτών των αλγορίθμων, καθώς

και των παραγόντων που επηρεάζουν την απόδοσή τους, είναι εξαιρετικά σημαντική. Ο Naive Bayes, παρότι απλός, συνεχίζει να χρησιμοποιείται ευρέως λόγω της αξιοπιστίας και της ευελιξίας του. Παρ' όλα αυτά, η εφαρμογή του σε διαφορετικούς τύπους δεδομένων, όπως τα συνεχή και τα κατηγορικά, εγείρει ερωτήματα για το ποια μέθοδος είναι πιο κατάλληλη.

Η επιλογή αυτού του θέματος στοχεύει στη μελέτη της σχέσης ανάμεσα στη μορφή των δεδομένων και την απόδοση του αλγορίθμου. Εξετάζεται επίσης αν η μετατροπή των συνεχών δεδομένων σε κατηγορικά μέσω της διαδικασίας της διακριτοποίησης μπορεί να βελτιώσει τα αποτελέσματα, ή αν η απευθείας χρήση του Gaussian Naive Bayes αρκεί. Μέσα από αυτή την ανάλυση, η εργασία επιδιώκει να προτείνει πρακτικές και χρήσιμες κατευθύνσεις για την εφαρμογή αλγορίθμων κατηγοριοποίησης σε διαφορετικούς τύπους προβλημάτων.

Ο σκοπός δεν είναι μόνο να αναλυθεί η απόδοση των διαφορετικών μεθόδων, αλλά και να δοθούν σαφείς οδηγίες για την επιλογή της κατάλληλης μεθόδου ανάλογα με το πρόβλημα. Αυτή η εργασία φιλοδοξεί να συμβάλει στη βελτίωση της κατανόησης των αλγορίθμων που χρησιμοποιούνται στην επιστήμη δεδομένων και να ενισχύσει τη σωστή χρήση τους σε πραγματικές εφαρμογές.

Επιπροσθέτως (και ίσως κατά κύριο λόγο), η εργασία αυτή θα προσφέρει στον συντάκτη της το τόσο απόμακρο κατά τα τελευταία 12 χρόνια πτυχίο, το οποίο έχει περάσει πλέον στον κόσμο του μύθου, καθώς επίσης και στους ανθρώπους που τον έφεραν στον μάταιο αυτό κόσμο, την ανακούφιση πως πλέον ο απολωλός υιός μπορεί να βαδίζει πλάι στους Άλαν Τούρινγκ, Τιμ Μπέρνερς Λι και John von Neumann, στων οποίων την παρουσία δεν θα φαίνεται ανεπαρκής.

1.5 Οργάνωση της εργασίας

Η εργασία, αποτελείται από 7 κεφάλαια. Έως τώρα έχει γίνει μια σύντομη αναφορά στο τι θα ακολουθήσει, καθώς επίσης έχει γίνει μια εισαγωγή σχετικά με την κατηγοριοποίηση που βασίζεται σε πιθανότητες και τους αλγορίθμους της.

Στο δεύτερο κεφάλαιο γίνεται μια εις βάθος ανασκόπηση της κατηγοριοποίησης με βάση τον Naive Bayes. Παρουσιάζονται οι διάφορες παραλλαγές του αλγορίθμου, όπως ο Categorical Naive Bayes, ο Gaussian Naive Bayes, ο Bernoulli Naive Bayes και ο Multinomial Naive Bayes. Ιδιαίτερη έμφαση δίνεται στη χρήση του Categorical Naive Bayes με αριθμητικά δεδομένα, με ανάλυση των τεχνικών διακριτοποίησης που χρησιμοποιούνται, όπως η Uniform, η Quantile και η k-means διακριτοποίηση.

Το τρίτο κεφάλαιο επεκτείνει τη συζήτηση σε άλλους αλγορίθμους κατηγοριοποίησης, παρέχοντας ένα πλαίσιο σύγκρισης με τον Naive Bayes. Αναλύονται μέθοδοι όπως τα Δέντρα Απόφασης, το Random Forest, οι Κ Εγγύτεροι Γείτονες, η Λογιστική Παλινδρόμηση και οι Μηχανές Διανυσμάτων Υποστήριξης (SVMs). Επιπλέον, περιλαμβάνεται και πάλι ο Gaussian Naive Bayes για να υπογραμμιστεί η θέση του μεταξύ αυτών των εναλλακτικών.

Στο τέταρτο κεφάλαιο παρουσιάζεται η πρακτική εφαρμογή της κατηγοριοποίησης Naive Bayes στην Python. Γίνεται αναφορά στη βιβλιοθήκη Sklearn και σε υλοποιήσεις στην Python που επιτρέπουν την εύκολη χρήση των διαφόρων παραλλαγών του αλγορίθμου, με έμφαση στην πρακτική εφαρμογή και τη λειτουργικότητά τους.

Το πέμπτο κεφάλαιο επικεντρώνεται στην πειραματική διαδικασία, όπου περιγράφονται τα σύνολα δεδομένων που χρησιμοποιήθηκαν και η μεθοδολογία που ακολουθήθηκε για τη διεξαγωγή των πειραμάτων. Αναλύονται τα αποτελέσματα της εφαρμογής των μεθόδων, ενώ ακολουθεί συζήτηση για την απόδοση και τη σημασία των ευρημάτων σε σχέση με τους στόχους της εργασίας.

Το έκτο και τελευταίο κεφάλαιο συνοψίζει τα ευρήματα, παρουσιάζει τα συμπεράσματα της μελέτης και διατυπώνει προτάσεις για μελλοντική έρευνα.

Κεφάλαιο 2ο: Κατηγοριοποίηση Naive Bayes

2.1 Categorical Naive Bayes

Ο **Categorical Naive Bayes** αποτελεί μια παραλλαγή του κλασικού αλγορίθμου Naive Bayes, σχεδιασμένη ειδικά για την επεξεργασία δεδομένων κατηγορικής φύσης. Αν και η ακριβής περίοδος δημιουργίας του δεν είναι ξεκάθαρη, στηρίζεται στις ίδιες θεμελιώδεις αρχές που χαρακτηρίζουν τον αρχικό αλγόριθμο Naive Bayes, οι οποίες αντλούν έμπνευση από τη στατιστική και τη θεωρία πιθανοτήτων. Οι βασικές αρχές του Naive Bayes βασίζονται στο θεώρημα του Bayes, το οποίο εισήγαγε ο Thomas Bayes κατά τον 18ο αιώνα, και στην υπόθεση της ανεξαρτησίας μεταξύ των μεταβλητών (13).

Η ανάπτυξη μιας εξειδικευμένης έκδοσης του Naive Bayes για δεδομένα αυτού του τύπου έγινε απαραίτητη λόγω της συχνής εμφάνισης τέτοιων δεδομένων σε πρακτικές εφαρμογές όπως η ανάλυση ερωτηματολογίων, η πρόβλεψη γεγονότων και η κατηγοριοποίηση εγγράφων. Τα κατηγορικά δεδομένα περιλαμβάνουν χαρακτηριστικά που παίρνουν διακριτές τιμές χωρίς καμία εγγενή σειρά, όπως για παράδειγμα τα χρώματα (π.χ., κόκκινο, πράσινο, μπλε) ή οι τύποι προϊόντων (π.χ., ηλεκτρονικά, τρόφιμα, ρούχα) (14).

Ο **Categorical Naive Bayes** κάνει την υπόθεση ότι τα χαρακτηριστικά είναι στατιστικά ανεξάρτητα μεταξύ τους και ότι κάθε κατηγορία μπορεί να περιγραφεί με μια συγκεκριμένη πιθανότητα για κάθε δυνατή τιμή του κάθε χαρακτηριστικού. Η πρόβλεψη της κατηγορίας ενός νέου δείγματος γίνεται με βάση τον υπολογισμό των πιθανοτήτων των χαρακτηριστικών του δείγματος για κάθε κατηγορία, επιλέγοντας τελικά εκείνη με τη μεγαλύτερη πιθανότητα.

Παρακάτω φαίνεται ο ψευδοκώδικας για την εφαρμογή του Categorical Naive Bayes

Algorithm 1 Categorical Naive Bayes

Require: Dataset D με δείγματα και τις κατηγορίες τους
Ensure: Προβλεπόμενη κατηγορία για ένα νέο δείγμα

Υπολογισμός Πιθανοτήτων:
for κάθε χαρακτηριστικό x_i **do**
    Υπολογίστε τις πιθανότητες εμφάνισης όλων των δυνατών τιμών του x_i στις κατηγορίες
end for
Εκτίμηση Πιθανοτήτων Κατηγορίας:
for κάθε κατηγορία y **do**
    Υπολογίστε την πιθανότητα εμφάνισης της κατηγορίας $P(y)$
end for
Συνδυασμός Πιθανοτήτων:
for κάθε κατηγορία y **do**
    Υπολογίστε $P(y \mid \mathbf{x}) = P(y) \prod_i P(x_i \mid y)$
end for
Κατάταξη:
Επιλέξτε την κατηγορία με τη μέγιστη πιθανότητα $P(y \mid \mathbf{x})$

Εφαρμογές και Χρήσεις

Ο **Categorical Naive Bayes** έχει χρησιμοποιηθεί ευρέως σε προβλήματα όπως η ανάλυση κοινωνικών μέσων, η κατηγοριοποίηση εγγράφων, η ανάλυση συναισθημάτων και η ανάλυση δημοσκοπήσεων. Η χαμηλή υπολογιστική του πολυπλοκότητα και η ευκολία υλοποίησης τον καθιστούν ιδανικό εργαλείο για εφαρμογές όπου απαιτείται ταχύτητα και απλότητα. Παρά τις περιοριστικές υποθέσεις του, ο αλγόριθμος παρέχει εντυπωσιακή απόδοση σε πλήθος εφαρμογών (13).

2.2 Categorical Naive Bayes και Αριθμητικά Δεδομένα (Διακριτοποίηση)

Ο **Categorical Naive Bayes** αποτελεί έναν αλγόριθμο ταξινόμησης που απαιτεί τα δεδομένα να είναι κατηγορικής μορφής, δηλαδή κάθε χαρακτηριστικό να αντιπροσωπεύεται από διακριτές τιμές. Ωστόσο, στην πράξη, πολλά datasets περιλαμβάνουν αριθμητικά χαρακτηριστικά, τα οποία δεν μπορούν να δεχθούν επεξεργασία απευθείας από τον αλγόριθμο χωρίς προηγούμενη μετατροπή. Για να γεφυρωθεί αυτό το χάσμα, χρησιμοποιείται η διαδικασία της **διακριτοποίησης**, που μετατρέπει αριθμητικά δεδομένα σε κατηγορικές τιμές μέσω ομαδοποίησης των αριθμητικών τιμών σε διακριτές κατηγορίες ή διαστήματα (15; 16).

Η **διακριτοποίηση** είναι μια τεχνική προεπεξεργασίας δεδομένων που διευκολύνει τη χρήση κατηγορικών αλγορίθμων, όπως ο Categorical Naive Bayes, σε δεδομένα που περιλαμβάνουν συνεχείς ή αριθμητικές τιμές. Ουσιαστικά, η διαδικασία χωρίζει το εύρος των τιμών ενός αριθμητικού χαρακτηριστικού σε διαστήματα, τα οποία αντιπροσωπεύονται από κατηγορικές ετικέτες. Για παράδειγμα, ένα χαρακτηριστικό όπως η ηλικία μπορεί να χωριστεί σε κατηγορίες όπως ”Παιδί” (0-18), ”Νεαρός” (19-30), ”Ώριμος” (31-45), ”Μεσήλικας” (46-65) και ”Ηλικιωμένος” (66+). Η κάθε τιμή ηλικίας αντικαθίσταται από την αντίστοιχη κατηγορική ετικέτα, καθιστώντας το χαρακτηριστικό κατάλληλο για χρήση από τον αλγόριθμο.

Η εφαρμογή της διακριτοποίησης ξεκινά με την ανάλυση του dataset για τον προσδιορισμό του εύρους τιμών κάθε αριθμητικού χαρακτηριστικού, καθώς και της κατανομής τους. Αυτή η αρχική ανάλυση είναι κρίσιμη για την επιλογή της κατάλληλης μεθόδου διακριτοποίησης, αφού διαφορετικές μέθοδοι αποδίδουν καλύτερα σε διαφορετικές κατανομές δεδομένων. Για παράδειγμα, η **uniform διακριτοποίηση** χωρίζει το εύρος τιμών σε ισομεγέθη διαστήματα. Αυτή η προσέγγιση είναι εύκολη στην εφαρμογή, αλλά δεν αποδίδει πάντα καλά όταν τα δεδομένα έχουν άνιση κατανομή, όπως στην περίπτωση χαρακτηριστικών που εμφανίζουν συμπύκνωση των τιμών σε ένα συγκεκριμένο μέρος του εύρους.

Μια εναλλακτική προσέγγιση είναι η **quantile διακριτοποίηση**, όπου οι αριθμητικές τιμές χωρίζονται σε διαστήματα έτσι ώστε κάθε κατηγορία να περιέχει τον ίδιο αριθμό δειγμάτων. Αυτή η μέθοδος είναι πιο αποδοτική για δεδομένα με μη ομοιόμορφη κατανομή, καθώς εξισορροπεί το πλήθος των δειγμάτων ανά κατηγορία. Μια πιο προηγμένη τεχνική είναι η **k-means διακριτοποίηση**, η οποία βασίζεται στον αλγόριθμο k-means clustering. Με αυτήν την προσέγγιση, οι τιμές ομαδοποιούνται με βάση την εγγύτητά τους σε συγκεκριμένα κέντρα συστάδων, δημιουργώντας διαστήματα που αντανακλούν τη φυσική κατανομή των δεδομένων. Παρόλο που η μέθοδος αυτή είναι ευέλικτη, είναι πιο απαιτητική υπολογιστικά και μπορεί να απαιτεί βελτιστοποίηση παραμέτρων.

Αφού καθοριστούν τα διαστήματα, κάθε αριθμητική τιμή αντικαθίσταται από την ετικέτα του αντίστοι-

χου διαστήματος. Το αποτέλεσμα είναι ένα dataset που περιλαμβάνει μόνο κατηγορικά χαρακτηριστικά, τα οποία είναι συμβατά με τον Categorical Naive Bayes. Ο αλγόριθμος χρησιμοποιεί αυτές τις κατηγορίες για να υπολογίσει τις πιθανότητες κάθε κλάσης, στηριζόμενος στην υπόθεση ανεξαρτησίας μεταξύ των χαρακτηριστικών.

Η διακριτοποίηση δεν είναι απαλλαγμένη περιορισμών. Μπορεί να οδηγήσει σε απώλεια πληροφορίας, καθώς οι αριθμητικές τιμές ομαδοποιούνται σε λιγότερες κατηγορίες, μειώνοντας τη λεπτομέρεια των δεδομένων. Η επιλογή του αριθμού και του μεγέθους των διαστημάτων είναι κρίσιμη, καθώς επηρεάζει άμεσα την απόδοση του μοντέλου. Παρά τους περιορισμούς, η διακριτοποίηση παραμένει απαραίτητη για την προσαρμογή αριθμητικών δεδομένων σε κατηγορικούς αλγορίθμους, προσφέροντας μια ευέλικτη λύση για την εφαρμογή του Categorical Naive Bayes σε ένα ευρύ φάσμα προβλημάτων ταξινόμησης.

2.2.1 Uniform Διακριτοποίηση

Η **uniform διακριτοποίηση** είναι μια μέθοδος προεπεξεργασίας που χρησιμοποιείται για τη μετατροπή αριθμητικών δεδομένων σε κατηγορικές τιμές, βασιζόμενη στη διαίρεση του εύρους τιμών ενός χαρακτηριστικού σε ίσα διαστήματα. Αυτή η τεχνική είναι ιδιαίτερα δημοφιλής λόγω της απλότητάς της και της εύκολης εφαρμογής της. Σε περιπτώσεις όπου οι αριθμητικές τιμές ενός χαρακτηριστικού είναι ομοιόμορφα κατανεμημένες, η uniform διακριτοποίηση παρέχει μια ισορροπημένη κατανομή τιμών ανά κατηγορία, διευκολύνοντας τη χρήση τους από κατηγορικούς αλγορίθμους, όπως ο Categorical Naive Bayes (17).

Η διαδικασία ξεκινά με τον προσδιορισμό του εύρους των τιμών του χαρακτηριστικού. Υπολογίζεται η ελάχιστη και η μέγιστη τιμή, ενώ καθορίζεται ο αριθμός των διαστημάτων, γνωστών και ως bins, που θα δημιουργηθούν. Ο αριθμός των διαστημάτων μπορεί να επιλεγεί χειροκίνητα, με βάση την εμπειρία του αναλυτή, ή να καθοριστεί μέσω στατιστικών κανόνων, όπως η τετραγωνική ρίζα του αριθμού των δειγμάτων. Στη συνέχεια, το συνολικό εύρος διαιρείται σε ίσα διαστήματα και κάθε τιμή του χαρακτηριστικού αντιστοιχίζεται στο διάστημα στο οποίο ανήκει. Το αποτέλεσμα είναι η αντικατάσταση των αρχικών αριθμητικών τιμών με κατηγορικές ετικέτες, οι οποίες αντιπροσωπεύουν τα διαστήματα (18).

Η uniform διακριτοποίηση διακρίνεται για την απλότητά της. Ωστόσο, η αποδοτικότητά της εξαρτάται από την κατανομή των δεδομένων. Αν οι αριθμητικές τιμές συγκεντρώνονται σε ένα συγκεκριμένο τμήμα του εύρους ή παρουσιάζουν ασυμμετρία, η μέθοδος αυτή μπορεί να οδηγήσει σε ανισομερή κατανομή δειγμάτων στα διαστήματα. Σε τέτοιες περιπτώσεις, ενδέχεται να χρειαστεί η εφαρμογή πιο σύνθετων μεθόδων διακριτοποίησης, όπως η quantile ή η k-means διακριτοποίηση.

Η uniform διακριτοποίηση περιγράφεται μαθηματικά ως εξής: Αν ένα χαρακτηριστικό X έχει ελάχιστη τιμή $\min(X)$ και μέγιστη τιμή $\max(X)$, και αν χωρίζεται σε k διαστήματα, τότε το εύρος κάθε διαστήματος δίνεται από τη σχέση:

$$\text{Διάστημα} = \frac{\max(X) - \min(X)}{k}$$

Για μια τιμή $x \in X$, η κατηγορία στην οποία ανήκει καθορίζεται από τη σχέση:

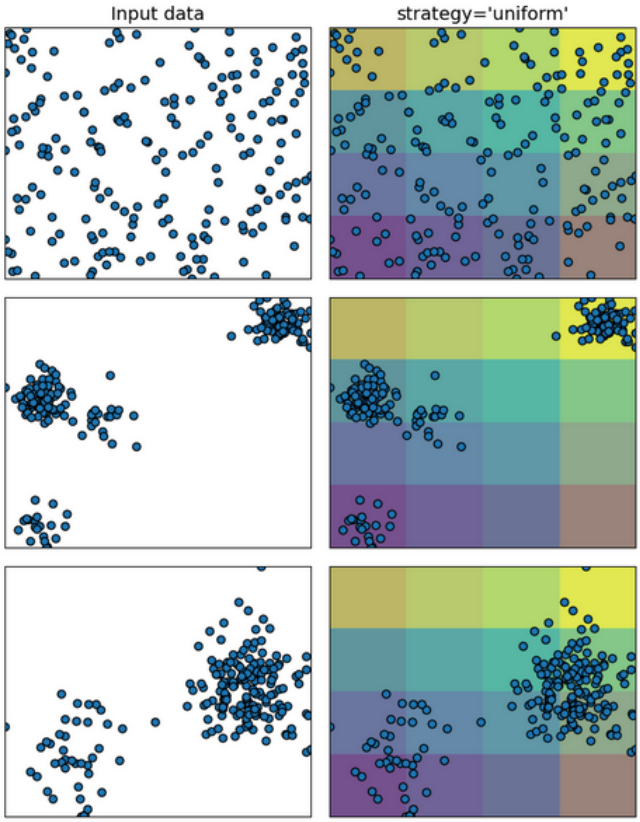
$$\text{Κατηγορία} = \left\lfloor \frac{x - \min(X)}{\text{Διάστημα}} \right\rfloor$$

Παρακάτω φαίνεται ο ψευδοκώδικας για την διακριτοποίηση με την εφαρμογή της μεθόδου Uniform:

Algorithm 2 Uniform Διακριτοποίηση

Require: Dataset X, Number of bins k
Ensure: Discretized dataset $X_{\text{discretized}}$

Calculate min_value $= \min(X)$
Calculate max_value $= \max(X)$
Calculate bin_width $= (\text{max\_value} - \text{min\_value})/k$
Initialize $X_{\text{discretized}} = []$
for each value x in X **do**
    bin_index $= \lfloor (x - \text{min\_value})/\text{bin\_width} \rfloor$
    if bin_index $== k$ **then** ▷ Handle boundary case
        bin_index $= k - 1$
    end if
    Append bin_index to $X_{\text{discretized}}$
end for
return $X_{\text{discretized}}$



Σχήμα 2.1: Χρησιμοποιώντας την μέθοδο uniform για διακριτοποίηση

Η παραπάνω εικόνα δείχνει πώς λειτουργεί η uniform διακριτοποίηση σε δεδομένα με δύο χαρακτηριστικά. Στην αριστερή πλευρά κάθε γραμμής βλέπουμε τα αρχικά δεδομένα, τα οποία αποτελούνται από μπλε σημεία κατανεμημένα σε έναν δισδιάστατο χώρο. Αυτά τα δεδομένα είναι συνεχούς μορφής και δεν έχουν υποστεί επεξεργασία. Στη δεξιά πλευρά, παρουσιάζεται το αποτέλεσμα της uniform διακριτοποίησης, όπου ο χώρος έχει χωριστεί σε ίσα διαστήματα (bins) κατά μήκος κάθε άξονα, και κάθε σημείο έχει αντιστοιχιστεί στο διάστημα στο οποίο ανήκει. Τα διαφορετικά χρώματα στις περιοχές υποδεικνύουν τις κατηγορίες που προκύπτουν από τη διακριτοποίηση.

Στην πρώτη γραμμή της εικόνας, τα δεδομένα είναι ομοιόμορφα κατανεμημένα σε όλο τον χώρο. Η uniform διακριτοποίηση αποδίδει καλά εδώ, χωρίζοντας τον χώρο σε ίσα διαστήματα που καλύπτουν ολόκληρη την περιοχή των δεδομένων. Δεδομένου ότι τα δεδομένα είναι ομοιόμορφα κατανεμημένα, κάθε διάστημα περιέχει περίπου τον ίδιο αριθμό σημείων, με αποτέλεσμα μια σχετικά ισορροπημένη αντιστοίχιση σε κατηγορίες.

Στη δεύτερη γραμμή, τα δεδομένα σχηματίζουν εμφανείς ομάδες (clusters), αλλά η uniform διακριτοποίηση αγνοεί αυτή τη δομή και εξακολουθεί να δημιουργεί ίσα διαστήματα. Το αποτέλεσμα είναι ότι κάποια διαστήματα περιέχουν πολλά δεδομένα λόγω της συγκέντρωσης, ενώ άλλα μένουν κενά. Αυτό μπορεί να περιορίσει την αποδοτικότητα της μεθόδου, καθώς δεν αντικατοπτρίζει την πραγματική ομαδοποίηση των δεδομένων.

Στην τρίτη γραμμή, τα δεδομένα είναι πιο συγκεντρωμένα σε λίγες περιοχές, δημιουργώντας πυκνές ομάδες. Παρόλο που η uniform διακριτοποίηση εφαρμόζει ίσα διαστήματα στον χώρο, τα περισσότερα δεδομένα καταλήγουν να βρίσκονται συγκεντρωμένα σε λίγες κατηγορίες, ενώ άλλες κατηγορίες παραμένουν σχεδόν άδειες. Αυτό δείχνει ότι η μέθοδος δεν προσαρμόζεται στη δομή των δεδομένων, κάτι που μπορεί να οδηγήσει σε απώλεια πληροφορίας.

Συνοπτικά, η uniform διακριτοποίηση είναι μια απλή και εύκολα εφαρμόσιμη μέθοδος που χωρίζει τα δεδομένα σε ίσα διαστήματα, χωρίς να λαμβάνει υπόψη την κατανομή τους. Αυτό την καθιστά αποτελεσματική για δεδομένα με ομοιόμορφη κατανομή, όπως φαίνεται στην πρώτη γραμμή της εικόνας. Ωστόσο, όταν τα δεδομένα σχηματίζουν clusters ή είναι συγκεντρωμένα σε περιορισμένες περιοχές, όπως στη δεύτερη και τρίτη γραμμή, η μέθοδος μπορεί να οδηγήσει σε μη αντιπροσωπευτικές κατηγορίες. Παρά τις προκλήσεις αυτές, η uniform διακριτοποίηση παραμένει χρήσιμη σε εφαρμογές που δίνουν προτεραιότητα στην απλότητα και τη γρήγορη προετοιμασία δεδομένων.(19).

2.2.2 Quantile Διακριτοποίηση

Η **quantile διακριτοποίηση** είναι μια μέθοδος μετατροπής αριθμητικών δεδομένων σε κατηγορικές τιμές, βασισμένη στον διαχωρισμό του συνόλου των τιμών σε διαστήματα που περιέχουν περίπου ίσο αριθμό δειγμάτων. Σε αντίθεση με την uniform διακριτοποίηση, η οποία δημιουργεί ίσα εύρη τιμών ανεξάρτητα από την κατανομή, η quantile διακριτοποίηση λαμβάνει υπόψη τη διασπορά των δεδομένων, κάνοντάς την ιδιαίτερα κατάλληλη για περιπτώσεις με μη κανονική κατανομή. Ο στόχος είναι να διατηρηθεί η ισορροπία μεταξύ των κατηγοριών, εξασφαλίζοντας ότι κάθε κατηγορία περιέχει παρόμοιο αριθμό δειγμάτων, ώστε να διευκολύνεται η εκπαίδευση κατηγορηματικών αλγορίθμων όπως ο Categorical Naive Bayes (17).

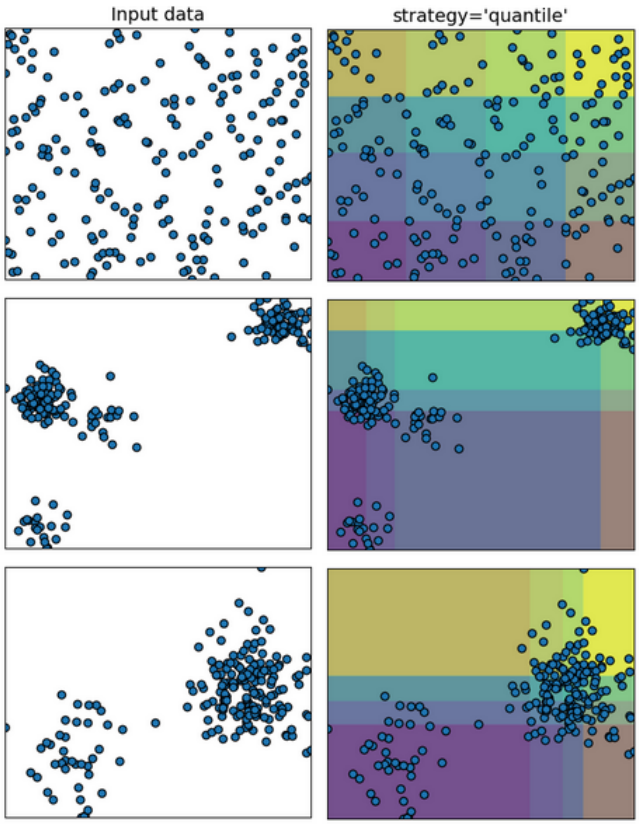
Η διαδικασία ξεκινά με την ταξινόμηση των αριθμητικών τιμών της στήλης (χαρακτηριστικού) του dataset σε αύξουσα σειρά. Στη συνέχεια, τα όρια των διαστημάτων καθορίζονται με βάση τα σημεία διαχωρισμού, τα οποία υποδεικνύουν τις θέσεις των τιμών που χωρίζουν το χαρακτηριστικό σε ίσα μέρη. Για k κατηγορίες, τα όρια υπολογίζονται στις θέσεις των σημείων διαχωρισμού $\frac{n}{k}, \frac{2n}{k}, \ldots, \frac{(k-1)n}{k}$, όπου n είναι το πλήθος των δειγμάτων. Μια αριθμητική τιμή αντιστοιχίζεται στο διάστημα στο οποίο ανήκει και αντικαθίσταται από μια κατηγορική ετικέτα που αντιπροσωπεύει το συγκεκριμένο διάστημα (20).

Μαθηματικά, έστω $X = \{x_1, x_2, \ldots, x_n\}$ ένα χαρακτηριστικό με n δείγματα που πρόκειται να διακριτοποιηθεί σε k κατηγορίες. Τα όρια των διαστημάτων $b_1, b_2, \ldots, b_{k-1}$ υπολογίζονται ως $b_i = Q_i = \text{Quantile}\left[\frac{i}{k}\right[$, όπου $i = 1, 2, \ldots, k-1$. Για μια τιμή $x \in X$, η κατηγορία στην οποία ανήκει υπολογίζεται με τη σχέση $\text{Κατηγορία} = \min\{j \mid b_{j-1} \leq x < b_j\}$, με $b_0 = -\infty$ και $b_k = +\infty$. Η προσέγγιση αυτή εξασφαλίζει ισορροπημένη κατανομή δειγμάτων μεταξύ των κατηγοριών.

Παρακάτω φαίνεται ο ψευδοκώδικας για την εφαρμογή της διακριτοποίησης με την μέθοδο Quantile:

Algorithm 3 Quantile Διακριτοποίηση

Require: Δεδομένα X, Αριθμός διαστημάτων k
Ensure: Διακριτοποιημένα δεδομένα $X_{\text{discretized}}$
Ταξινόμηση του X σε αύξουσα σειρά
Υπολογισμός των ορίων των διαστημάτων:
    $n = \text{μήκος}(X)$
    quantile_indices $= [\lceil n \times i/k \rceil$ για i στο εύρος$(1, k)]$
    bin_boundaries $= [X[\text{quantile\_indices}[i]]$ για i στο εύρος(μήκος(quantile_indices))]
Αρχικοποίηση του $X_{\text{discretized}} = []$
for κάθε τιμή x στο X **do**
    for κάθε όριο στο bin_boundaries **do**
        if x ανήκει σε ένα διάστημα **then**
            Ανάθεση του αντίστοιχου δείκτη διαστήματος στο x
        end if
    end for
    Προσθήκη του δείκτη διαστήματος στο $X_{\text{discretized}}$
end for
return $X_{\text{discretized}}$



Σχήμα 2.2: Χρησιμοποιώντας την μέθοδο Quantile για διακριτοποίηση

Στην εικόνα του παραδείγματος της quantile διακριτοποίησης θα δούμε πως στην δεξιά στήλη, η οποία απεικονίζει το αποτέλεσμα, ο χώρος έχει χωριστεί σε κατηγορίες (bins) με τρόπο που εξασφαλίζει ότι κάθε κατηγορία περιέχει περίπου ίσο αριθμό σημείων. Αυτό διαφοροποιεί την quantile διακριτοποίηση από την uniform, καθώς τα όρια των κατηγοριών εδώ δεν είναι ίσα σε μήκος αλλά εξαρτώνται από την πυκνότητα των δεδομένων.

Στην πρώτη γραμμή, τα δεδομένα έχουν ομοιόμορφη κατανομή στον χώρο, και η quantile διακριτοποίηση λειτουργεί καλά, χωρίζοντας τα δεδομένα σε κατηγορίες με ίσο αριθμό σημείων. Σε αυτή την περίπτωση, τα αποτελέσματα είναι παρόμοια με εκείνα της uniform διακριτοποίησης, αν και τα όρια μεταξύ των κατηγοριών μπορεί να διαφέρουν ελαφρώς λόγω της προσαρμογής στην κατανομή των δεδομένων.

Στη δεύτερη γραμμή, όπου τα δεδομένα σχηματίζουν clusters, η quantile διακριτοποίηση αποδίδει καλύτερα σε σύγκριση με την uniform. Εξισορροπεί τον αριθμό των σημείων σε κάθε κατηγορία προσαρμόζοντας τα όρια των κατηγοριών έτσι ώστε να συμπεριλάβουν περισσότερα δεδομένα στις περιοχές υψηλής πυκνότητας. Αντίθετα, η uniform διακριτοποίηση εφαρμόζει ίσα διαστήματα, με αποτέλεσμα κάποιες κατηγορίες να παραμένουν σχεδόν κενές, αγνοώντας τη φυσική δομή των δεδομένων.

Στην τρίτη γραμμή, όπου τα δεδομένα είναι ιδιαίτερα συγκεντρωμένα σε συγκεκριμένα σημεία, η quantile διακριτοποίηση εξακολουθεί να εξισορροπεί τις κατηγορίες δημιουργώντας μικρότερα διαστήματα στις περιοχές με υψηλή πυκνότητα και μεγαλύτερα στις περιοχές με λιγότερα δεδομένα. Αυτό είναι σημαντικό γιατί επιτρέπει την καλύτερη αντιπροσώπευση των δεδομένων σε κάθε κατηγορία. Στην περίπτωση της uniform διακριτοποίησης, τα ίσα διαστήματα οδηγούν σε υπερβολική συγκέντρωση σημείων σε λίγες κατηγορίες, αφήνοντας άλλες κατηγορίες κενές ή σχεδόν κενές.

Η σύγκριση με την uniform διακριτοποίηση καταδεικνύει τη μεγαλύτερη ευελιξία της quantile διακριτοποίησης, καθώς προσαρμόζεται καλύτερα στην κατανομή των δεδομένων. Ενώ η uniform είναι κατάλληλη για δεδομένα με ομοιόμορφη κατανομή, η quantile διακριτοποίηση αποδίδει καλύτερα σε περιπτώσεις όπου τα δεδομένα σχηματίζουν clusters ή παρουσιάζουν μεταβλητές πυκνότητες. Αυτή η προσαρμοστικότητα την καθιστά ιδανική επιλογή για δεδομένα με πολύπλοκη δομή, ενώ παράλληλα εξασφαλίζει ισορροπία στον αριθμό σημείων ανά κατηγορία, κάτι που είναι κρίσιμο για αλγόριθμους που βασίζονται σε κατηγορικές τιμές.(18).

2.2.3 K-means Διακριτοποίηση

Η **k-means διακριτοποίηση** είναι μια πιο εξελιγμένη μέθοδος που χρησιμοποιεί τον αλγόριθμο k-means clustering για να μετατρέψει αριθμητικά δεδομένα σε κατηγορικές τιμές. Η βασική ιδέα της είναι να ομαδοποιήσει τις τιμές σε k συστάδες (clusters) με βάση την εγγύτητά τους, δημιουργώντας κατηγορίες που αντικατοπτρίζουν τη φυσική δομή των δεδομένων. Αυτή η προσέγγιση είναι ιδιαίτερα χρήσιμη για δεδομένα που έχουν σύνθετες ή μη ομοιόμορφες κατανομές, καθώς δεν βασίζεται σε προκαθορισμένα όρια αλλά σε προσαρμοσμένες ομαδοποιήσεις που παράγονται από τον ίδιο τον αλγόριθμο (17).

Η διαδικασία ξεκινά με την τυχαία αρχικοποίηση k κέντρων συστάδων (centroids) μέσα στο εύρος των τιμών του χαρακτηριστικού. Στη συνέχεια, κάθε τιμή αντιστοιχίζεται στο κέντρο που βρίσκεται πλησιέστερα, σχηματίζοντας έτσι k συστάδες. Ακολούθως, τα κέντρα των συστάδων αναπροσαρμόζονται υπολογίζοντας τη μέση τιμή (mean) των τιμών που ανήκουν σε κάθε συστάδα. Η διαδικασία αυτή επαναλαμβάνεται μέχρι τα κέντρα να σταθεροποιηθούν, δηλαδή να μην αλλάξουν σημαντικά από μία επανάληψη στην επόμενη. Όταν ολοκληρωθεί η διαδικασία, κάθε τιμή του χαρακτηριστικού αντικαθίσταται από την ετικέτα της συστάδας στην οποία ανήκει (20).

Μαθηματικά, ο αλγόριθμος k-means προσπαθεί να ελαχιστοποιήσει τη συνολική ενδοσυσταδική διακύμανση (intra-cluster variance), που εκφράζεται ως:

$$J = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|^2$$

όπου k είναι ο αριθμός των συστάδων, C_i η i-οστή συστάδα, μ_i το κέντρο της i-οστής συστάδας και x τα δεδομένα που ανήκουν στη συστάδα. Ο στόχος είναι να ελαχιστοποιηθεί το J, που αντιπροσωπεύει το άθροισμα των τετραγωνικών αποστάσεων μεταξύ κάθε τιμής και του κέντρου της αντίστοιχης συστάδας (21).

Παρακάτω φαίνεται ο ψευδοκώδικας για την εφαρμογή της διακριτοποίησης με την μέθοδο K-means:

Algorithm 4 K-means Διακριτοποίηση

Require: Δεδομένα X, Αριθμός συστάδων k, Μέγιστος αριθμός επαναλήψεων *max_iter*
Ensure: Διακριτοποιημένο σύνολο δεδομένων $X_{\text{discretized}}$

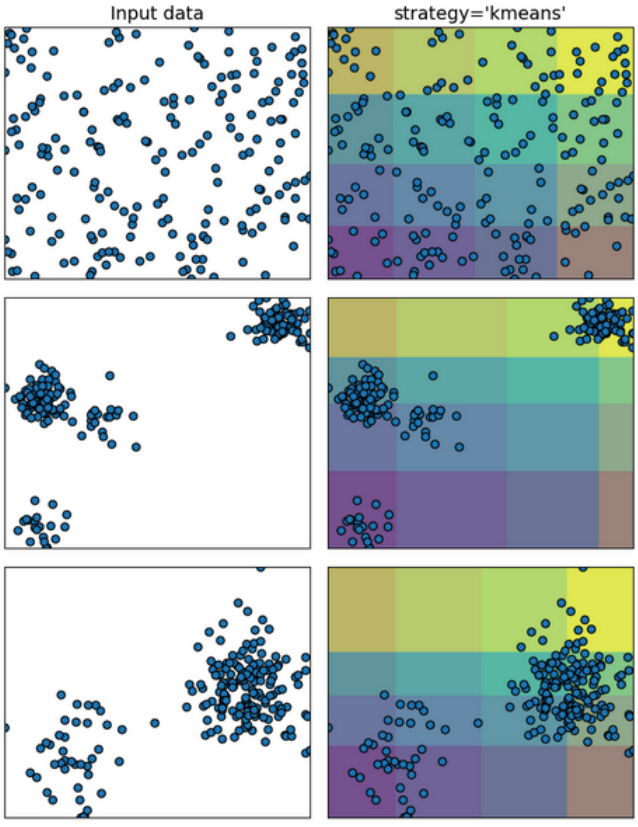
1: Τυχαία αρχικοποίηση k κέντρων συστάδων: *centroids* = $[c_1, c_2, \ldots, c_k]$
2: **for** κάθε επανάληψη έως *max_iter* **do**
3: **Ανάθεση συστάδων:** Ανάθεση κάθε τιμής x στο πλησιέστερο κέντρο συστάδας:

$$cluster\_assignment[x] = \arg\min_i(\|x - c_i\|)$$

4: **Ενημέρωση κέντρων:** Υπολογισμός νέου κέντρου για κάθε συστάδα:

$$c_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$$

5: **end for**
6: Ανάθεση διακριτών ετικετών στις συστάδες:
7: **for** κάθε τιμή x στο X **do**
8: Ανάθεση στο x της ετικέτας της συστάδας του
9: **end for**
10: **return** $X_{\text{discretized}}$



Σχήμα 2.3: Χρησιμοποιώντας την μέθοδο K-means για διακριτοποίηση

Όπως προείπαμε η k-means διακριτοποίηση επικεντρώνεται στη φυσική ομαδοποίηση των σημείων και τη σχέση τους μέσα στον χώρο. Στην πρώτη γραμμή της εικόνας, όπου τα δεδομένα είναι ομοιόμορφα κατανεμημένα στον χώρο, τα όρια των κατηγοριών που δημιουργεί η k-means δεν διαφέρουν ιδιαίτερα από τα όρια των μεθόδων uniform και quantile.

Στη δεύτερη γραμμή, τα δεδομένα σχηματίζουν clusters και η k-means διακριτοποίηση αποδίδει εξαιρετικά, προσαρμόζοντας τα όρια των κατηγοριών στη φυσική δομή των δεδομένων. Η μέθοδος δημιουργεί κατηγορίες που αντιστοιχούν σε περιοχές με συγκέντρωση σημείων, κάτι που την καθιστά πιο ευέλικτη από την uniform, που αγνοεί τις συγκεντρώσεις, και την quantile, που εξισορροπεί μόνο τον αριθμό των σημείων ανά κατηγορία χωρίς να λαμβάνει υπόψη τη γεωμετρική δομή.

Στην τρίτη γραμμή, όπου τα δεδομένα είναι έντονα συγκεντρωμένα σε συγκεκριμένες περιοχές, η k-means διακριτοποίηση προσαρμόζει τα όρια των κατηγοριών ανάλογα με την πυκνότητα των δεδομένων. Οι κατηγορίες στις περιοχές υψηλής πυκνότητας γίνονται μικρότερες, ενώ στις αραιότερες περιοχές γίνονται μεγαλύτερες, εξασφαλίζοντας ότι κάθε κατηγορία αντανακλά καλύτερα τη γεωμετρική δομή του dataset. Σε σύγκριση, η uniform αγνοεί την πυκνότητα και δημιουργεί ίσα διαστήματα, ενώ η quantile εξισορροπεί τα δεδομένα σε κατηγορίες χωρίς να προσαρμόζεται στις τοπικές συγκεντρώσεις.

Συνολικά, η k-means διακριτοποίηση παρέχει μια πιο φυσική ομαδοποίηση, καθώς προσαρμόζεται στη δομή και την κατανομή των δεδομένων. Σε σχέση με τις uniform και quantile διακριτοποιήσεις, είναι πιο αποτελεσματική για δεδομένα που σχηματίζουν clusters ή έχουν περιοχές υψηλής πυκνότητας. Ωστόσο, η αποδοτικότητα της μεθόδου εξαρτάται από την επιλογή του αριθμού των κατηγοριών (clusters) και τη θέση των αρχικών κέντρων, τα οποία μπορούν να επηρεάσουν σημαντικά το τελικό αποτέλεσμα. Παρά τις προκλήσεις αυτές, η k-means διακριτοποίηση αποτελεί μια εξαιρετική επιλογή για την κατηγοριοποίηση δεδομένων με πολύπλοκη ή μη κανονική δομή.(21)

2.3 Gaussian Naive Bayes

Ο **Gaussian Naive Bayes** είναι ένας αλγόριθμος ταξινόμησης που ανήκει στην οικογένεια του Naive Bayes και έχει σχεδιαστεί για να επεξεργάζεται δεδομένα με συνεχείς τιμές. Η βασική του υπόθεση είναι ότι κάθε χαρακτηριστικό για μια συγκεκριμένη κατηγορία ακολουθεί κανονική κατανομή (Gaussian distribution). Ο αλγόριθμος βασίζεται στο θεώρημα του Bayes, το οποίο υπολογίζει την πιθανότητα μιας κατηγορίας δεδομένων τα χαρακτηριστικά του δείγματος. Για να απλοποιηθεί ο υπολογισμός, θεωρείται ότι τα χαρακτηριστικά είναι ανεξάρτητα μεταξύ τους (υπόθεση ανεξαρτησίας), γεγονός που μειώνει τη πολυπλοκότητα της διαδικασίας (22).

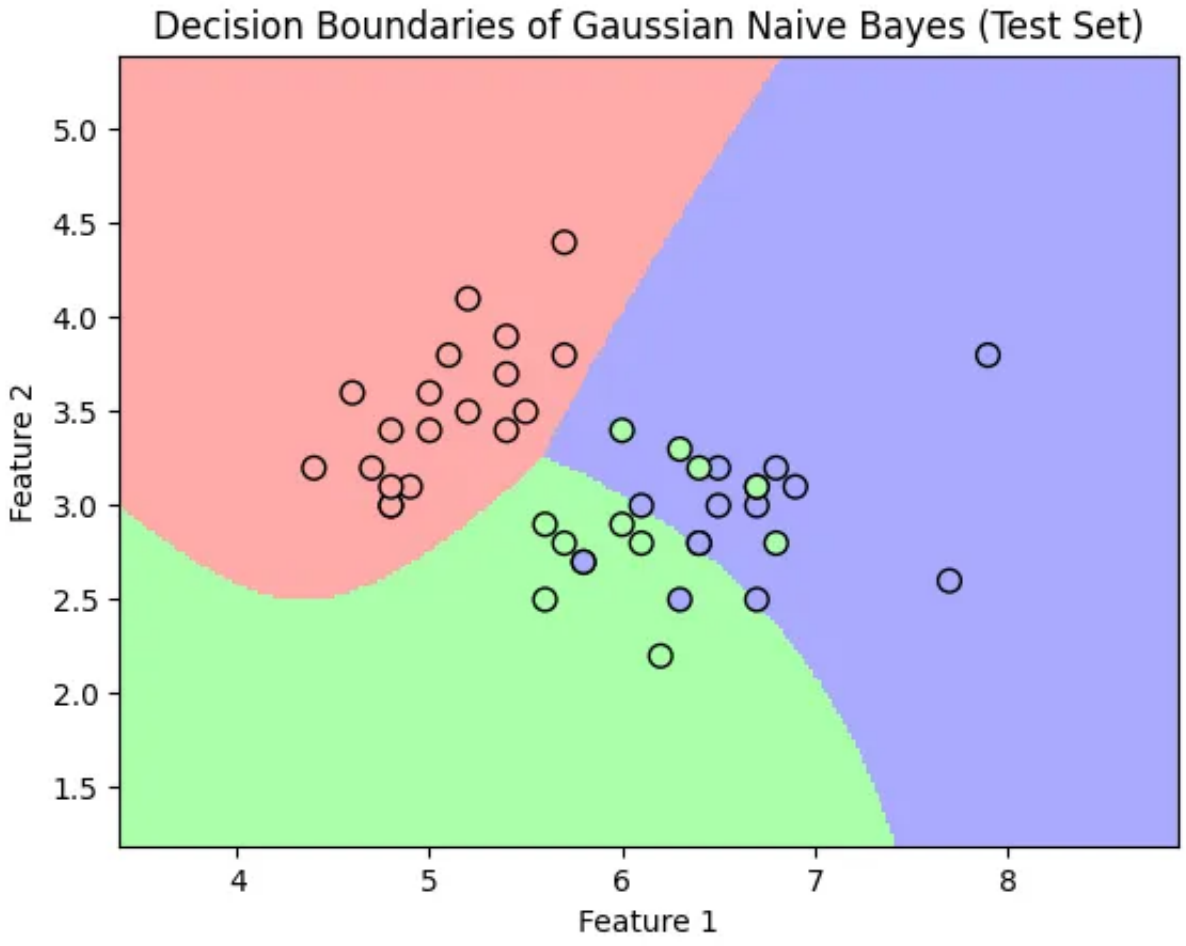
Κατά τη λειτουργία του, ο Gaussian Naive Bayes προσπαθεί να εκτιμήσει την πιθανότητα $P(c|X)$ για κάθε κατηγορία c, όπου $X = \{x_1, x_2, \ldots, x_n\}$ είναι ένα δείγμα με n χαρακτηριστικά. Η εκτίμηση αυτή πραγματοποιείται με βάση το θεώρημα του Bayes:

$$P(c|X) \propto P(c) \cdot \prod_{i=1}^{n} P(x_i|c)$$

Η πιθανότητα $P(x_i|c)$ εκφράζεται μέσω της συνάρτησης πυκνότητας πιθανότητας της κανονικής κατανομής:

$$P(x_i|c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} \exp\left[-\frac{(x_i - \mu_c)^2}{2\sigma_c^2}\right[$$

όπου: - μ_c είναι η μέση τιμή του χαρακτηριστικού x_i στην κατηγορία c, - σ_c είναι η τυπική απόκλιση του χαρακτηριστικού x_i στην κατηγορία c.



Σχήμα 2.4: Χρησιμοποιώντας τον αλγόριθμο Gaussian Naive Bayes για κατηγοριοποίηση

Στην εικόνα που βλέπουμε, τα όρια απόφασης που διαχωρίζουν τις κατηγορίες έχουν υπολογιστεί από τον Gaussian Naive Bayes, λαμβάνοντας υπόψη δύο χαρακτηριστικά του dataset, το Feature 1 και το Feature 2. Ο χώρος χαρακτηριστικών χωρίζεται σε περιοχές, καθεμία από τις οποίες αντιστοιχεί σε μία κατηγορία. Τα χρώματα της εικόνας (κόκκινο, πράσινο, μπλε) αντιπροσωπεύουν αυτές τις διαφορετικές περιοχές, ενώ οι γραμμές μεταξύ τους είναι τα όρια απόφασης που υποδεικνύουν τη μετάβαση από τη μία κατηγορία στην άλλη. Τα όρια αυτά καθορίζονται από την κανονική κατανομή που υποθέτει ο αλγόριθμος για τα δεδομένα κάθε κατηγορίας.

Κάθε σημείο στον χώρο αποδίδεται στην κατηγορία με τη μεγαλύτερη πιθανότητα $P(c|X)$, η οποία υπολογίζεται με βάση τα χαρακτηριστικά του σημείου (Feature 1 και Feature 2) και τις παραμέτρους της κανονικής κατανομής (μέση τιμή μ και τυπική απόκλιση σ) που προκύπτουν από τη φάση εκπαίδευσης. Οι κύκλοι στην εικόνα αντιπροσωπεύουν τα δεδομένα του συνόλου δοκιμών (test set), τα οποία δεν συμμετείχαν στην εκπαίδευση, και χρησιμοποιούνται για να αξιολογηθεί η ακρίβεια και η απόδοση του αλγορίθμου. Η θέση κάθε κύκλου σε σχέση με τα όρια απόφασης καθορίζει σε ποια κατηγορία ανήκει η πρόβλεψη του αλγορίθμου.

Κάθε κατηγορία ”καλύπτει” μία περιοχή του χώρου χαρακτηριστικών, και τα όρια μεταξύ των κατηγοριών προσαρμόζονται ανάλογα με τη στατιστική συμπεριφορά των δεδομένων (κανονική κατανομή) για κάθε κατηγορία. Τα όρια δεν είναι γραμμικά, γεγονός που υποδεικνύει ότι ο αλγόριθμος μπορεί να προσαρμόζεται σε πιο σύνθετες κατανομές δεδομένων. Συνολικά, η εικόνα καταδεικνύει πώς ο Gaussian Naive Bayes εκμεταλλεύεται την υπόθεση της κανονικής κατανομής για να δημιουργήσει προβλέψεις και να ταξινομήσει τα δεδομένα σε διάφορες κατηγορίες.

Ο αλγόριθμος λειτουργεί σε δύο φάσεις: την **εκπαίδευση** και την **πρόβλεψη**. Στη φάση εκπαίδευσης, υπολογίζονται οι εκ των προτέρων πιθανότητες $P(c)$ για κάθε κατηγορία, καθώς και οι παράμετροι της κανονικής κατανομής (μ_c και σ_c) για κάθε χαρακτηριστικό και κατηγορία.

Algorithm 5 Gaussian Naive Bayes Training

Input: Σύνολο εκπαίδευσης D
Output: $P(c), \mu_c, \sigma_c$
for κάθε κατηγορία c **do**
    Υπολόγισε $P(c)$
    for κάθε χαρακτηριστικό x_i **do**
        Υπολόγισε $\mu_{c,i}$, $\sigma_{c,i}$
    end for
end for
Return: $P(c), \mu_c, \sigma_c$

Στη φάση πρόβλεψης, για ένα νέο δείγμα υπολογίζεται η πιθανότητα $P(c|X)$ για κάθε κατηγορία c, και η κατηγορία με τη μέγιστη πιθανότητα επιλέγεται ως η πρόβλεψη (23).

Algorithm 6 Prediction Phase of Gaussian Naive Bayes

Require: Εκ των προτέρων πιθανότητες $P(c)$, Μέση τιμή μ_c, Τυπική απόκλιση σ_c, Νέο δείγμα X
Ensure: Προβλεπόμενη κατηγορία c

Αρχικοποίησε τη μέγιστη πιθανότητα: max_likelihood $\leftarrow -\infty$
for κάθε κατηγορία c **do**
    Αρχικοποίησε την πιθανότητα: $P(c|X) \leftarrow P(c)$
    for κάθε χαρακτηριστικό x_i στο X **do**
        Υπολόγισε την πιθανότητα $P(x_i|c)$ χρησιμοποιώντας τον Gaussian τύπο:

$$P(x_i|c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} \exp\left[-\frac{(x_i - \mu_c)^2}{2\sigma_c^2}\right[$$

        Ενημέρωσε την πιθανότητα: $P(c|X) \leftarrow P(c|X) \cdot P(x_i|c)$
    end for
    if $P(c|X) >$ max_likelihood **then**
        Ενημέρωσε τη μέγιστη πιθανότητα: max_likelihood $\leftarrow P(c|X)$
        Ενημέρωσε την καλύτερη κατηγορία: best_class $\leftarrow c$
    end if
end for
return best_class

Ο Gaussian Naive Bayes είναι δημοφιλής λόγω της απλότητάς του και της ταχύτητάς του, ιδίως σε περιπτώσεις μεγάλων datasets. Ωστόσο, η απόδοσή του εξαρτάται από την υπόθεση ότι τα δεδομένα ακολουθούν κανονική κατανομή. Αν αυτή η υπόθεση παραβιαστεί, η ακρίβεια μπορεί να μειωθεί. Παρά τις προκλήσεις, αποτελεί μια από τις πιο αποδοτικές μεθόδους για την κατηγοριοποίηση δεδομένων συνεχούς μορφής.

2.4 Bernoulli Naive Bayes

Ο **Bernoulli Naive Bayes** είναι μια παραλλαγή του Naive Bayes που είναι ιδιαίτερα κατάλληλη για δεδομένα δυαδικής μορφής (binary data). Χρησιμοποιείται συχνά σε προβλήματα κατηγοριοποίησης όπου τα χαρακτηριστικά ενός δείγματος μπορούν να πάρουν μόνο δύο τιμές, όπως η παρουσία ή η απουσία μιας λέξης σε ανάλυση κειμένου (24). Ο αλγόριθμος βασίζεται στο θεώρημα του Bayes, ενώ υποθέτει την ανεξαρτησία των χαρακτηριστικών, γεγονός που απλοποιεί τους υπολογισμούς (25). Στο πλαίσιο του Bernoulli Naive Bayes, κάθε χαρακτηριστικό θεωρείται ότι ακολουθεί μια δυαδική κατανομή. Αυτό σημαίνει ότι για κάθε κατηγορία, υπολογίζονται οι πιθανότητες εμφάνισης ($P(x_i = 1|c)$) και απουσίας ($P(x_i = 0|c)$) κάθε χαρακτηριστικού.

Η πιθανότητα ενός δείγματος $X = \{x_1, x_2, \ldots, x_n\}$ να ανήκει σε μια κατηγορία c υπολογίζεται ως το γινόμενο των πιθανοτήτων κάθε χαρακτηριστικού να είναι παρόν ή απόν για την κατηγορία c, συνδυασμένο με την εκ των προτέρων πιθανότητα της κατηγορίας ($P(c)$) (25).

Μαθηματικά, η πιθανότητα $P(c|X)$ εκφράζεται ως:

$$P(c|X) \propto P(c) \cdot \prod_{i=1}^{n} P(x_i|c),$$

όπου:

$$P(x_i|c) = \begin{cases} P(x_i = 1|c), & \text{αν } x_i = 1, \\ P(x_i = 0|c) = 1 - P(x_i = 1|c), & \text{αν } x_i = 0. \end{cases}$$

Ο αλγόριθμος εκτελείται σε δύο φάσεις: την εκπαίδευση και την πρόβλεψη. Στη φάση της εκπαίδευσης, υπολογίζονται οι εκ των προτέρων πιθανότητες ($P(c)$) για κάθε κατηγορία, καθώς και οι πιθανότητες εμφάνισης κάθε χαρακτηριστικού ($P(x_i = 1|c)$) εντός της κατηγορίας. Αυτές οι πιθανότητες υπολογίζονται διαιρώντας τον αριθμό των δειγμάτων που περιέχουν το χαρακτηριστικό x_i για την κατηγορία c με το συνολικό αριθμό των δειγμάτων στην ίδια κατηγορία.

Algorithm 7 Training Phase of Bernoulli Naive Bayes

Require: Σύνολο εκπαίδευσης D
Ensure: Εκ των προτέρων πιθανότητες $P(c)$ και πιθανότητες χαρακτηριστικών $P(x_i = 1|c)$

1: Αρχικοποίησε τις εκ των προτέρων πιθανότητες $P(c)$ και τις πιθανότητες χαρακτηριστικών $P(x_i = 1|c)$
2: **for** κάθε κατηγορία c στο D **do**
3: Υπολόγισε $P(c)$:

$$P(c) = \frac{\text{Αριθμός δειγμάτων στην κατηγορία } c}{\text{Συνολικός αριθμός δειγμάτων}}$$

4: **for** κάθε χαρακτηριστικό x_i **do**
5: Υπολόγισε $P(x_i = 1|c)$:

$$P(x_i = 1|c) = \frac{\text{Αριθμός δειγμάτων με } x_i = 1 \text{ στην κατηγορία } c}{\text{Αριθμός δειγμάτων στην κατηγορία } c}$$

6: **end for**
7: **end for**
8: **return** $P(c)$, $P(x_i = 1|c)$

Στη φάση της πρόβλεψης, για ένα νέο δείγμα, ο αλγόριθμος υπολογίζει την πιθανότητα $P(c|X)$ για κάθε κατηγορία c ως το γινόμενο της εκ των προτέρων πιθανότητας της κατηγορίας $P(c)$ και των πιθανοτήτων εμφάνισης ή απουσίας κάθε χαρακτηριστικού του δείγματος (24).

Algorithm 8 Prediction Phase of Bernoulli Naive Bayes

Require: Εκ των προτέρων πιθανότητες $P(c)$, πιθανότητες χαρακτηριστικών $P(x_i = 1|c)$, Νέο δείγμα X
Ensure: Προβλεπόμενη κατηγορία c

```
for κάθε κατηγορία c do
    Αρχικοποίησε την πιθανότητα: P(c|X) = P(c)
    for κάθε χαρακτηριστικό x_i στο X do
        if x_i = 1 then
            P(c|X) ← P(c|X) · P(x_i = 1|c)
        else
            P(c|X) ← P(c|X) · (1 – P(x_i = 1|c))
        end if
    end for
end for
Επίλεξε την κατηγορία c με τη μέγιστη πιθανότητα P(c|X) return c
```

Ο Bernoulli Naive Bayes είναι ιδιαίτερα χρήσιμος για προβλήματα όπως η κατηγοριοποίηση κειμένου, όπου κάθε χαρακτηριστικό αντιπροσωπεύει την παρουσία ή την απουσία μιας λέξης. Σε τέτοιες περιπτώσεις, μπορεί να υπερτερεί σε απόδοση σε σύγκριση με άλλες παραλλαγές του Naive Bayes, καθώς λαμβάνει υπόψη όχι μόνο την παρουσία αλλά και την απουσία χαρακτηριστικών, γεγονός που μπορεί να είναι κρίσιμο για την ακρίβεια της πρόβλεψης. Ωστόσο, η απόδοση του αλγορίθμου εξαρτάται από τη σωστή προετοιμασία των δεδομένων και τη φύση του προβλήματος, καθώς η ανεξαρτησία των χαρακτηριστικών δεν ισχύει πάντα στην πράξη.

2.5 Multinomial Naive Bayes

Ο **Multinomial Naive Bayes** χρησιμοποιείται κυρίως για δεδομένα που αντιπροσωπεύουν συχνότητες ή πλήθη, όπως η ανάλυση κειμένου και η κατηγοριοποίηση εγγράφων. Σε αντίθεση με τον Bernoulli Naive Bayes, ο οποίος εξετάζει μόνο την παρουσία ή την απουσία χαρακτηριστικών, ο Multinomial Naive Bayes λαμβάνει υπόψη τη συχνότητα εμφάνισης κάθε χαρακτηριστικού στο δείγμα (26). Είναι ιδιαίτερα κατάλληλος για προβλήματα όπως η κατηγοριοποίηση εγγράφων, όπου κάθε χαρακτηριστικό αντιπροσωπεύει τον αριθμό των φορών που εμφανίζεται μια λέξη σε ένα κείμενο (27).

Ο αλγόριθμος βασίζεται στο θεώρημα του Bayes, το οποίο χρησιμοποιείται για την εκτίμηση της πιθανότητας $P(c|X)$, δηλαδή της πιθανότητας ένα δείγμα $X = \{x_1, x_2, \ldots, x_n\}$ να ανήκει σε μια κατηγορία c. Η πιθανότητα αυτή εκφράζεται ως:

$$P(c|X) \propto P(c) \cdot \prod_{i=1}^{n} P(x_i|c)$$

Η πιθανότητα $P(x_i|c)$, που αντιπροσωπεύει τη συχνότητα εμφάνισης του χαρακτηριστικού x_i στην κατηγορία c, υπολογίζεται με την εξής μαθηματική παράσταση (28):

$$P(x_i|c) = \frac{N_{i,c} + \alpha}{N_c + \alpha \cdot V}$$

όπου το $N_{i,c}$ είναι ο αριθμός των φορών που εμφανίζεται το χαρακτηριστικό x_i στην κατηγορία c. Το N_c αναφέρεται στο συνολικό πλήθος όλων των χαρακτηριστικών (οι συνολικές συχνότητες όλων των χαρακτηριστικών) στην κατηγορία c. Το V αντιπροσωπεύει το πλήθος των μοναδικών χαρακτηριστικών στο σύνολο δεδομένων, γνωστό και ως μέγεθος του λεξιλογίου. Το α είναι ο όρος εξομάλυνσης, συνήθως ίσος με 1 (Laplace smoothing), που χρησιμοποιείται για να αποτραπεί η μηδενική πιθανότητα σε χαρακτηριστικά που δεν εμφανίζονται σε κάποια κατηγορία.

Κατά τη φάση της εκπαίδευσης, ο αλγόριθμος υπολογίζει τις εκ των προτέρων πιθανότητες $P(c)$ για κάθε κατηγορία. Αυτές οι πιθανότητες προκύπτουν από τη συχνότητα εμφάνισης των δειγμάτων της κατηγορίας στο σύνολο εκπαίδευσης, ως εξής:

$$P(c) = \frac{\text{Αριθμός δειγμάτων στην κατηγορία } c}{\text{Συνολικός αριθμός δειγμάτων}}$$

Παράλληλα, υπολογίζονται οι πιθανότητες $P(x_i|c)$ για κάθε χαρακτηριστικό x_i και κατηγορία c με τη χρήση της μαθηματικής σχέσης που περιλαμβάνει την εξομάλυνση. Αυτές οι πιθανότητες λαμβάνουν υπόψη τις συχνότητες εμφάνισης των χαρακτηριστικών σε κάθε κατηγορία και προσαρμόζονται ώστε να αποφεύγονται τα προβλήματα που προκύπτουν από την απουσία χαρακτηριστικών σε συγκεκριμένες κατηγορίες.

Algorithm 9 Training Phase of Multinomial Naive Bayes

Require: Σύνολο εκπαίδευσης D
Ensure: Εκ των προτέρων πιθανότητες $P(c)$ και πιθανότητες χαρακτηριστικών $P(x_i|c)$
1: **for** κάθε κατηγορία c στο D **do**
2: Υπολόγισε την εκ των προτέρων πιθανότητα $P(c)$:

$$P(c) = \frac{\text{Αριθμός δειγμάτων στην κατηγορία } c}{\text{Συνολικός αριθμός δειγμάτων}}$$

3: **for** κάθε χαρακτηριστικό x_i **do**
4: Υπολόγισε την πιθανότητα $P(x_i|c)$:

$$P(x_i|c) = \frac{N_{i,c} + \alpha}{N_c + \alpha \cdot V}$$

όπου:

- $N_{i,c}$: Αριθμός φορών που εμφανίζεται το χαρακτηριστικό x_i στην κατηγορία c,
- N_c: Συνολικό πλήθος χαρακτηριστικών στην κατηγορία c,
- V: Μέγεθος λεξιλογίου (πλήθος μοναδικών χαρακτηριστικών),
- α: Παράμετρος εξομάλυνσης (Laplace smoothing).

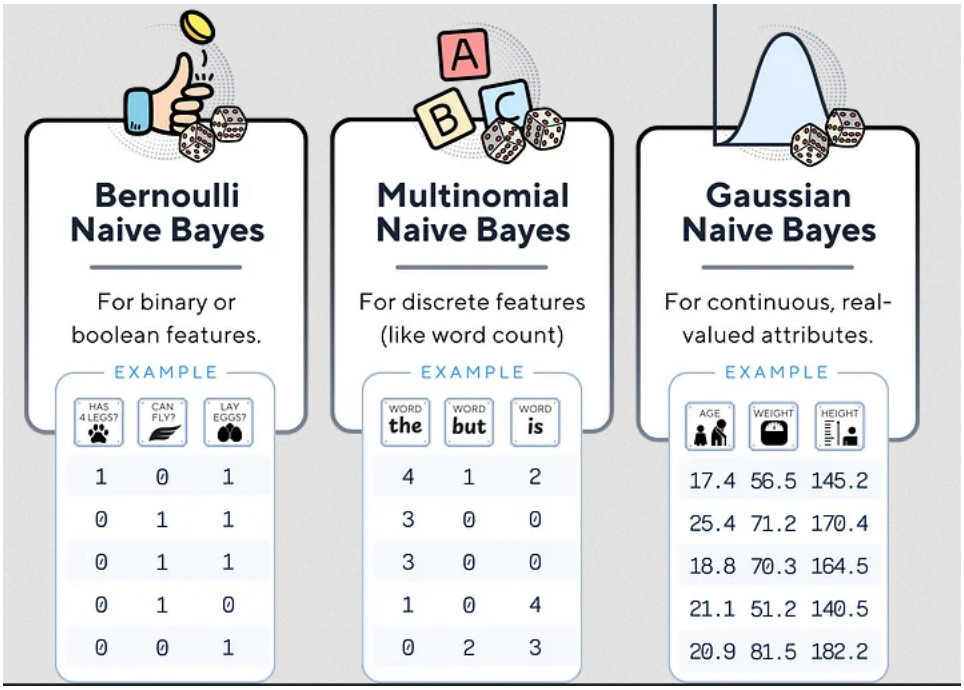
5: **end for**
6: **end for**
7: **return** $P(c)$, $P(x_i|c)$

Στη φάση της πρόβλεψης, για ένα νέο δείγμα $X = \{x_1, x_2, \ldots, x_n\}$, ο αλγόριθμος υπολογίζει την πιθανότητα $P(c|X)$ για κάθε κατηγορία c. Αυτό επιτυγχάνεται πολλαπλασιάζοντας την εκ των προτέρων πιθανότητα $P(c)$ της κατηγορίας με το γινόμενο των πιθανοτήτων $P(x_i|c)$ για όλα τα χαρακτηριστικά του δείγματος. Η κατηγορία με τη μέγιστη πιθανότητα $P(c|X)$ επιλέγεται ως η τελική πρόβλεψη.

Algorithm 10 Prediction Phase of Multinomial Naive Bayes

Require: Εκ των προτέρων πιθανότητες $P(c)$, πιθανότητες χαρακτηριστικών $P(x_i|c)$, Νέο δείγμα X
Ensure: Προβλεπόμενη κατηγορία c

```
for κάθε κατηγορία c do
    Αρχικοποίησε την πιθανότητα: P(c|X) = P(c)
    for κάθε χαρακτηριστικό x_i στο X do
        Ενημέρωσε την πιθανότητα: P(c|X) ← P(c|X) · P(x_i|c)
    end for
end for
Επίλεξε την κατηγορία c με τη μέγιστη πιθανότητα P(c|X) return c
```



Σχήμα 2.5: Οι διαφορετικές περιπτώσεις χρήσης των αλγορίθμων

Ανακεφαλαιώνοντας, οι τρεις παραλλαγές του Naive Bayes: Bernoulli, Multinomial και Gaussian, προορίζονται για διαφορετικούς τύπους δεδομένων, ανάλογα με τις ιδιαιτερότητες του προβλήματος. Ο **Bernoulli Naive Bayes** σχεδιάστηκε για δυαδικά ή Boolean δεδομένα, όπου τα χαρακτηριστικά έχουν μόνο δύο τιμές, όπως 0 και 1, οι οποίες δηλώνουν την παρουσία ή την απουσία ενός χαρακτηριστικού. Αυτή η προσέγγιση είναι ιδιαίτερα χρήσιμη σε περιπτώσεις όπως η ανάλυση κειμένων, όπου το ζητούμενο είναι αν μια συγκεκριμένη λέξη εμφανίζεται σε ένα έγγραφο. Από την άλλη πλευρά, ο **Multinomial Naive Bayes** εξειδικεύεται σε δεδομένα διακριτών τιμών, όπως συχνότητες εμφάνισης. Είναι ιδιαίτερα διαδεδομένος στην κατηγοριοποίηση κειμένων, όπου τα χαρακτηριστικά αναπαριστούν πόσες φορές εμφανίζεται μια λέξη σε ένα έγγραφο. Η λειτουργία του βασίζεται στη συχνότητα ή την κατανομή των

χαρακτηριστικών, κάτι που τον καθιστά ιδανικό για τέτοιου είδους δεδομένα (28). Ο **Gaussian Naive Bayes**, με τη σειρά του, προορίζεται για δεδομένα συνεχούς μορφής και βασίζεται στην υπόθεση ότι τα χαρακτηριστικά ακολουθούν κανονική κατανομή. Είναι ιδιαίτερα κατάλληλος για ταξινομήσεις όπου τα χαρακτηριστικά είναι αριθμητικά, όπως ηλικία, βάρος ή ύψος.

Οι βασικές διαφορές μεταξύ των τριών αλγορίθμων έγκεινται στο είδος των δεδομένων που μπορούν να διαχειριστούν και στη μέθοδο μοντελοποίησής τους. Ο Bernoulli επικεντρώνεται αποκλειστικά στην παρουσία ή απουσία χαρακτηριστικών, ο Multinomial εστιάζει στη συχνότητα εμφάνισης των χαρακτηριστικών, ενώ ο Gaussian ασχολείται με τη μοντελοποίηση δεδομένων συνεχούς μορφής. Η επιλογή της κατάλληλης μεθόδου εξαρτάται πάντα από τη φύση του dataset και τις απαιτήσεις του προβλήματος που εξετάζεται. Με αυτόν τον τρόπο, κάθε παραλλαγή προσφέρει μια εξειδικευμένη προσέγγιση, η οποία μπορεί να αποδώσει καλύτερα σε συγκεκριμένα είδη δεδομένων και εφαρμογές.

Κεφάλαιο 3ο: Άλλοι αλγόριθμοι κατηγοριοποίησης

3.1 Δέντρα απόφασης

Τα **δέντρα απόφασης (decision trees)** είναι ένας από τους πιο διαισθητικούς αλγόριθμους μηχανικής μάθησης και παρέχουν μια διαφορετική προσέγγιση από τους Naive Bayes. Ενώ οι Naive Bayes βασίζονται σε στατιστικές πιθανότητες και υποθέσεις ανεξαρτησίας, τα δέντρα απόφασης λειτουργούν με τη δημιουργία ιεραρχικών κανόνων που κατευθύνουν τα δεδομένα μέσα από κόμβους (nodes) έως ότου φτάσουν σε μια τελική πρόβλεψη. Αυτή η δομή επιτρέπει την επίλυση προβλημάτων κατηγοριοποίησης και παλινδρόμησης (29).

Η βασική αρχή ενός δέντρου απόφασης είναι η διάσπαση των δεδομένων με τρόπο που να μειώνει την αβεβαιότητα ή την ετερογένεια σε κάθε επίπεδο του δέντρου. Για την κατηγοριοποίηση, ένα δημοφιλές κριτήριο είναι το *κέρδος πληροφορίας* (*information gain*), το οποίο υπολογίζεται με βάση την εντροπία (*entropy*). Η εντροπία ορίζεται ως:

$$H(S) = -\sum_{i=1}^{c} p_i \cdot \log_2(p_i)$$

όπου p_i είναι η πιθανότητα εμφάνισης της κατηγορίας i στο σύνολο S (30). Το *κέρδος πληροφορίας* για μια διάσπαση A υπολογίζεται ως:

$$IG(S, A) = H(S) - \sum_{v \in V} \frac{|S_v|}{|S|} \cdot H(S_v)$$

όπου S_v είναι το υποσύνολο των δεδομένων που ανήκουν στην τιμή v του χαρακτηριστικού A. Ο αλγόριθμος επιλέγει τη διάσπαση που μεγιστοποιεί το $IG(S, A)$.

Για την παλινδρόμηση, η διάσπαση γίνεται με στόχο τη μείωση του *μέσου τετραγωνικού σφάλματος* (*mean squared error, MSE*), το οποίο υπολογίζεται ως:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y})^2$$

όπου y_i είναι η πραγματική τιμή, και $\hat{y}$ η προβλεπόμενη τιμή. Με αυτόν τον τρόπο, τα δέντρα απόφασης προσαρμόζονται είτε για κατηγοριοποίηση είτε για παλινδρόμηση, ανάλογα με το πρόβλημα που εξετάζεται.

Algorithm 11 Building a Decision Tree

Require: Σύνολο εκπαίδευσης D, Σύνολο χαρακτηριστικών F
Ensure: Δέντρο απόφασης T

if όλα τα δείγματα στο D ανήκουν στην ίδια κατηγορία **ή** F είναι κενό **then**
    Επιστρέφει έναν κόμβο φύλλου με την πλειοψηφική κατηγορία.
end if
for κάθε χαρακτηριστικό A στο F **do**
    Υπολόγισε το κέρδος πληροφορίας $IG(D, A)$ (ή το MSE για παλινδρόμηση).
end for
Επίλεξε το χαρακτηριστικό A_{best} με το μέγιστο IG (ή το ελάχιστο MSE).
Διάσπασε το D σε υποσύνολα D_v για κάθε τιμή v του A_{best}.
for κάθε υποσύνολο D_v **do**
    Κατασκεύασε αναδρομικά το υποδέντρο T_v χρησιμοποιώντας D_v και $F \setminus \{A_{\text{best}}\}$.
end for
Επιστρέφει το δέντρο T με ρίζα το A_{best} και παιδιά τα T_v.

Ο παραπάνω αλγόριθμος περιγράφει τη διαδικασία δημιουργίας ενός δέντρου απόφασης. Αρχικά, ελέγχεται αν όλα τα δείγματα ανήκουν στην ίδια κατηγορία ή αν δεν υπάρχουν άλλα χαρακτηριστικά για διάσπαση. Σε αυτές τις περιπτώσεις, δημιουργείται ένας τελικός κόμβος (φύλλο) που αντιπροσωπεύει την πλειοψηφική κατηγορία των δεδομένων.

Στη συνέχεια, για κάθε διαθέσιμο χαρακτηριστικό, υπολογίζεται το κέρδος πληροφορίας (IG) για προβλήματα κατηγοριοποίησης ή το μέσο τετραγωνικό σφάλμα (MSE) για προβλήματα παλινδρόμησης. Το χαρακτηριστικό που μεγιστοποιεί το IG ή ελαχιστοποιεί το MSE επιλέγεται ως το καλύτερο για τη διάσπαση των δεδομένων.

Με βάση το καλύτερο χαρακτηριστικό, τα δεδομένα διαχωρίζονται σε μικρότερα σύνολα, ανάλογα με τις τιμές του χαρακτηριστικού. Στη συνέχεια, για κάθε υποσύνολο δεδομένων, ο αλγόριθμος εφαρμόζεται ξανά αναδρομικά, με τα χαρακτηριστικά που έχουν απομείνει. Έτσι, το δέντρο απόφασης κατασκευάζεται σταδιακά, με το επιλεγμένο χαρακτηριστικό να γίνεται η ρίζα, ενώ τα υποσύνολα δεδομένων δημιουργούν τα υποδέντρα. Η διαδικασία συνεχίζεται έως ότου ικανοποιηθεί κάποιο κριτήριο τερματισμού, όπως η ύπαρξη μόνο μιας κατηγορίας σε ένα υποσύνολο ή η απουσία διαθέσιμων χαρακτηριστικών.(31).

3.2 Random Forest

Το **Random Forest** είναι ένας ισχυρός και ευέλικτος αλγόριθμος μηχανικής μάθησης που χρησιμοποιεί πολλά δέντρα απόφασης για να λύσει προβλήματα ταξινόμησης και παλινδρόμησης. Ανήκει στις μεθόδους *ensemble*, που σημαίνει ότι συνδυάζει πολλά απλά μοντέλα (τα δέντρα απόφασης) για να δημιουργήσει ένα πιο ισχυρό και αξιόπιστο μοντέλο. Το μεγάλο πλεονέκτημά του είναι ότι μειώνει την αστάθεια και την τάση των δέντρων απόφασης να προσαρμόζονται υπερβολικά στα δεδομένα εκπαίδευσης (*overfitting*) (32).

Ο τρόπος λειτουργίας του είναι απλός. Δημιουργεί ένα σύνολο από δέντρα απόφασης, όπου κάθε δέντρο εκπαιδεύεται σε ένα τυχαίο δείγμα των δεδομένων. Αυτή η διαδικασία ονομάζεται *bagging* (*bootstrap aggregating*) και περιλαμβάνει τη δημιουργία δειγμάτων με αντικατάσταση από το αρχικό dataset. Επιπλέον, σε κάθε κόμβο του δέντρου, αντί να εξετάζονται όλα τα διαθέσιμα χαρακτηριστικά για διάσπαση,

επιλέγεται ένα τυχαίο υποσύνολο χαρακτηριστικών. Αυτό προσθέτει περισσότερη τυχαιότητα, κάνοντας τα δέντρα λιγότερο εξαρτημένα μεταξύ τους (33).

Στη φάση της πρόβλεψης, τα αποτελέσματα από όλα τα δέντρα συνδυάζονται. Για ταξινόμηση, η τελική πρόβλεψη είναι η κατηγορία που λαμβάνει τις περισσότερες ψήφους από τα δέντρα. Για παλινδρόμηση, η πρόβλεψη είναι ο μέσος όρος των αποτελεσμάτων των δέντρων. Αυτή η διαδικασία επιτρέπει στο *Random Forest* να είναι πιο σταθερό και ακριβές από ένα μεμονωμένο δέντρο απόφασης.

Για ένα σύνολο N δέντρων απόφασης, η πρόβλεψη του *Random Forest* για ένα νέο δείγμα X δίνεται ως εξής:

- **Στην ταξινόμηση:**
$$\hat{y} = \arg\max_c \left[\sum_{i=1}^{N} I(T_i(X) = c) \right[$$
όπου $T_i(X)$ είναι η πρόβλεψη του i-ου δέντρου και c οι κατηγορίες.

- **Στην παλινδρόμηση:**
$$\hat{y} = \frac{1}{N} \sum_{i=1}^{N} T_i(X)$$
όπου $T_i(X)$ είναι η πρόβλεψη του i-οστού δέντρου.

Algorithm 12 Random Forest Training

Require: Σύνολο εκπαίδευσης D, Αριθμός δέντρων N, Αριθμός χαρακτηριστικών F
Ensure: Μοντέλο Random Forest $\mathcal{F}$
1: Αρχικοποίησε ένα κενό δάσος $\mathcal{F}$.
2: **for** κάθε δέντρο T_i από $i = 1$ έως N **do**
3: Δημιούργησε ένα δείγμα bootstrap D_i από το D.
4: Κατασκεύασε το δέντρο απόφασης T_i χρησιμοποιώντας το D_i:
- Σε κάθε κόμβο, επέλεξε τυχαία F χαρακτηριστικά.
- Επέλεξε το καλύτερο χαρακτηριστικό για διάσπαση μεταξύ των F.
- Διαχώρισε τα δεδομένα και επανάλαβε αναδρομικά.

5: Πρόσθεσε το T_i στο $\mathcal{F}$.
6: **end for**
7: **return** το δάσος $\mathcal{F}$.

Στη φάση της εκπαίδευσης, το Random Forest ξεκινά με την αρχικοποίηση ενός κενού δάσους $\mathcal{F}$. Στη συνέχεια, για κάθε ένα από τα N δέντρα, δημιουργείται ένα τυχαίο δείγμα (*bootstrap sample*) D_i από το αρχικό σύνολο δεδομένων D. Το δείγμα αυτό δημιουργείται με αντικατάσταση, που σημαίνει ότι κάποια δείγματα μπορούν να εμφανιστούν περισσότερες από μία φορές. Στη συνέχεια, κάθε δέντρο απόφασης T_i εκπαιδεύεται στο δείγμα D_i. Κατά την κατασκευή του δέντρου, σε κάθε κόμβο εξετάζεται μόνο ένα τυχαίο υποσύνολο F των συνολικών χαρακτηριστικών. Το καλύτερο χαρακτηριστικό για τη διάσπαση του κόμβου επιλέγεται με βάση κριτήρια όπως το κέρδος πληροφορίας (*information gain*) για ταξινόμηση ή τη μείωση του μέσου τετραγωνικού σφάλματος (*MSE*) για παλινδρόμηση. Μετά την εκπαίδευση, το δέντρο T_i προστίθεται στο δάσος $\mathcal{F}$. Αυτή η διαδικασία επαναλαμβάνεται για όλα τα δέντρα.

Algorithm 13 Random Forest Prediction

Require: Μοντέλο Random Forest $\mathcal{F}$, Δείγμα προς πρόβλεψη X
Ensure: Προβλεπόμενη κατηγορία ή τιμή $\hat{y}$

```
for κάθε δείγμα X do
    Συλλέξτε τις προβλέψεις από όλα τα δέντρα του F.
    if πρόβλημα ταξινόμησης then
        Επιστρέψτε την κατηγορία με τις περισσότερες ψήφους.
    else if πρόβλημα παλινδρόμησης then
        Επιστρέψτε τον μέσο όρο των προβλέψεων.
    end if
end for
```

Στη φάση της πρόβλεψης, η απόφαση για ένα νέο δείγμα X βασίζεται στη συνεργασία όλων των δέντρων του δάσους. Αρχικά, οι προβλέψεις συλλέγονται από όλα τα δέντρα T_i στο $\mathcal{F}$. Για προβλήματα ταξινόμησης, η τελική πρόβλεψη είναι η κατηγορία που λαμβάνει τις περισσότερες ψήφους (*majority voting*). Αντίθετα, για προβλήματα παλινδρόμησης, η τελική πρόβλεψη είναι ο μέσος όρος των προβλέψεων των δέντρων. Αυτή η διαδικασία επιτρέπει στον Random Forest να είναι πιο σταθερός και ακριβής, καθώς συνδυάζει την ισχύ πολλών ανεξάρτητων δέντρων, μειώνοντας την αστάθεια και την υπερπροσαρμογή (*overfitting*) που μπορεί να εμφανιστεί σε μεμονωμένα δέντρα (32).

3.3 Κ Εγγύτεροι Γείτονες (KNN)

Ο αλγόριθμος **Κ Εγγύτεροι Γείτονες (K-Nearest Neighbors, KNN)** είναι μια από τις πιο απλές μεθόδους μηχανικής μάθησης, που χρησιμοποιείται τόσο για ταξινόμηση όσο και για παλινδρόμηση. Σε αντίθεση με άλλους αλγόριθμους, ο KNN δεν κατασκευάζει κάποιο μαθηματικό μοντέλο κατά την εκπαίδευση. Αντίθετα, αποθηκεύει όλα τα δεδομένα εκπαίδευσης και τα χρησιμοποιεί κατευθείαν για να κάνει προβλέψεις. Για αυτό τον λόγο, ο KNN συχνά ονομάζεται «αλγόριθμος βάσει παραδειγμάτων» (34).

Για να βρει τους πιο κοντινούς γείτονες, ο αλγόριθμος υπολογίζει την απόσταση μεταξύ του νέου δείγματος και κάθε δείγματος εκπαίδευσης. Η πιο συχνά χρησιμοποιούμενη μετρική απόστασης είναι η **Ευκλείδεια απόσταση**, που υπολογίζεται με τον εξής τύπο:

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

όπου x και y είναι δύο δείγματα με n χαρακτηριστικά. Μόλις υπολογιστούν οι αποστάσεις, επιλέγονται οι k μικρότερες (35).

Στην ταξινόμηση, η κατηγορία του νέου δείγματος προκύπτει από την πλειοψηφία των κατηγοριών στους k γείτονες. Για παράδειγμα, αν οι περισσότεροι από τους k γείτονες ανήκουν σε μια συγκεκριμένη κατηγορία, αυτή η κατηγορία επιλέγεται ως πρόβλεψη. Στην παλινδρόμηση, η πρόβλεψη είναι ο μέσος όρος των τιμών των k γειτόνων.

Algorithm 14 Κ Εγγύτεροι Γείτονες (KNN)

Require: Σύνολο εκπαίδευσης D, Νέο δείγμα X, Αριθμός γειτόνων k
Ensure: Προβλεπόμενη κατηγορία ή τιμή
for κάθε δείγμα x στο D **do**
    Υπολόγισε την απόσταση $d(X, x)$ μεταξύ X και x
end for
Ταξινόμησε όλα τα δείγματα κατά αύξουσα απόσταση
Επίλεξε τα k πλησιέστερα δείγματα
if πρόβλημα ταξινόμησης **then**
    Πρόβλεψε την πλειοψηφική κατηγορία μεταξύ των k γειτόνων
else if πρόβλημα παλινδρόμησης **then**
    Πρόβλεψε τον μέσο όρο των τιμών των k γειτόνων
end if
return την πρόβλεψη

Ο KNN είναι πολύ εύκολος να κατανοηθεί, αλλά έχει μερικές αδυναμίες. Είναι υπολογιστικά ακριβός, ειδικά σε μεγάλα σύνολα δεδομένων, καθώς πρέπει να υπολογίζει τις αποστάσεις για κάθε δείγμα. Επίσης, η απόδοσή του εξαρτάται από την επιλογή του k: πολύ μικρό k μπορεί να οδηγήσει σε θόρυβο, ενώ πολύ μεγάλο k μπορεί να αγνοήσει σημαντικές τοπικές πληροφορίες. Παρόλα αυτά, είναι ένας ευέλικτος αλγόριθμος που χρησιμοποιείται συχνά σε εφαρμογές όπως η αναγνώριση μοτίβων, η ανάλυση εικόνων και η ανίχνευση ανωμαλιών (36).

3.4 Λογιστική Παλινδρόμηση (Logistic Regression)

Η **Λογιστική Παλινδρόμηση (Logistic Regression)** χρησιμοποιείται για προβλήματα όπου η μεταβλητή στόχος έχει δύο κατηγορίες (δυαδική ταξινόμηση). Παρότι το όνομά της περιέχει τη λέξη "παλινδρόμηση", η λογιστική παλινδρόμηση δεν χρησιμοποιείται για πρόβλεψη συνεχών τιμών, αλλά για την εκτίμηση της πιθανότητας ένα δείγμα να ανήκει σε μία από τις δύο κατηγορίες (37).

Η βασική ιδέα της λογιστικής παλινδρόμησης είναι να μοντελοποιήσει την πιθανότητα μιας κατηγορίας χρησιμοποιώντας μια λογιστική συνάρτηση (logistic function), επίσης γνωστή ως συνάρτηση sigmoid. Η συνάρτηση sigmoid μετατρέπει οποιαδήποτε πραγματική τιμή σε μια πιθανότητα μεταξύ 0 και 1. Η μαθηματική της έκφραση είναι:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

όπου $z = w^T x + b$, με w να είναι τα βάρη, x τα χαρακτηριστικά του δείγματος και b η σταθερά μετατόπισης (bias).

Ο στόχος της λογιστικής παλινδρόμησης είναι να εκπαιδεύσει ένα μοντέλο που θα εκτιμά τις πιθανότητες αυτές, έτσι ώστε να μπορεί να ταξινομήσει ένα νέο δείγμα με βάση το κατώφλι πιθανότητας (συνήθως 0.5). Αν η πιθανότητα είναι μεγαλύτερη από 0.5, το δείγμα ανήκει στη μία κατηγορία, αλλιώς στην άλλη.

Η εκπαίδευση του μοντέλου πραγματοποιείται με τη βελτιστοποίηση των βαρών w και της σταθεράς b, ώστε να ελαχιστοποιείται η **συνάρτηση κόστους**. Η συνάρτηση κόστους που χρησιμοποιείται είναι η

διασταυρούμενη εντροπία (cross-entropy loss):

$$J(w,b) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)\right]$$

όπου y_i είναι η πραγματική κατηγορία του δείγματος, $\hat{y}_i$ η προβλεπόμενη πιθανότητα, και N ο αριθμός των δειγμάτων (38).

Ο αλγόριθμος της λογιστικής παλινδρόμησης παρουσιάζεται παρακάτω:

Algorithm 15 Λογιστική Παλινδρόμηση (Logistic Regression)

Require: Σύνολο εκπαίδευσης D, Ρυθμός μάθησης α, Αριθμός επαναλήψεων T
Ensure: Εκπαιδευμένα βάρη w, σταθερά μετατόπισης b

1: Αρχικοποίησε τα βάρη w και τη σταθερά b σε μηδέν
2: **for** $t = 1$ to T **do**
3: Υπολόγισε τις προβλέψεις για όλα τα δείγματα:

$$\hat{y}_i = \sigma(w^T x_i + b)$$

4: Υπολόγισε τις παραγώγους της συνάρτησης κόστους:

$$\frac{\partial J}{\partial w} = \frac{1}{N}\sum (\hat{y}_i - y_i)\, x_i$$

$$\frac{\partial J}{\partial b} = \frac{1}{N}\sum (\hat{y}_i - y_i)$$

5: Ενημέρωσε τα βάρη και τη σταθερά:

$$w = w - \alpha \cdot \frac{\partial J}{\partial w}$$

$$b = b - \alpha \cdot \frac{\partial J}{\partial b}$$

6: **end for**
7: **return** w, b

Η λογιστική παλινδρόμηση είναι εύκολη να κατανοηθεί και να υλοποιηθεί. Είναι ιδιαίτερα αποτελεσματική όταν τα δεδομένα μπορούν να διαχωριστούν γραμμικά, δηλαδή όταν υπάρχει μια ευθεία γραμμή που διαχωρίζει τις δύο κατηγορίες. Ωστόσο, η απόδοσή της μπορεί να μειωθεί σε περιπτώσεις όπου τα δεδομένα είναι πολύπλοκα και μη γραμμικά. Παρόλα αυτά, παραμένει ένας εξαιρετικός αλγόριθμος για πολλές πρακτικές εφαρμογές, όπως η ανάλυση δεδομένων, η ιατρική διάγνωση και η αναγνώριση εικόνων.

3.5 Μηχανές Διανυσμάτων Υποστήριξης (SVMs)

Ο κύριος στόχος των **Μηχανών Διανυσμάτων Υποστήριξης (Support Vector Machines, SVMs)** είναι να βρουν την καλύτερη δυνατή γραμμή ή υπερεπίπεδο (*hyperplane*) που διαχωρίζει τα δεδομένα σε διαφορετικές κατηγορίες. Το «καλύτερο» υπερεπίπεδο είναι αυτό που μεγιστοποιεί την απόσταση (περιθώριο) μεταξύ του υπερεπιπέδου και των πιο κοντινών δειγμάτων από κάθε κατηγορία. Αυτά τα δείγματα

ονομάζονται *διανύσματα υποστήριξης (support vectors)*, καθώς καθορίζουν τη θέση του υπερεπιπέδου (**?**).

Στην πιο απλή μορφή του, το SVM χρησιμοποιείται για γραμμικά διαχωρίσιμα δεδομένα. Το υπερεπίπεδο ορίζεται ως:

$$w^T x + b = 0$$

όπου w είναι το διάνυσμα των βαρών, x το διάνυσμα χαρακτηριστικών και b η σταθερά μετατόπισης (*bias*). Το πρόβλημα βελτιστοποίησης που λύνει το SVM είναι να βρει w και b που μεγιστοποιούν το περιθώριο, ελαχιστοποιώντας ταυτόχρονα το σφάλμα ταξινόμησης:

$$\min_{w,b} \frac{1}{2}\|w\|^2$$

υπό τους περιορισμούς:

$$y_i(w^T x_i + b) \geq 1, \quad \forall i$$

όπου y_i είναι η ετικέτα του δείγματος ($+1$ ή -1) και x_i το διάνυσμα χαρακτηριστικών του δείγματος.

Για δεδομένα που δεν είναι γραμμικά διαχωρίσιμα, το SVM χρησιμοποιεί πυρηνικές συναρτήσεις (*kernel functions*) για να μετασχηματίσει τα δεδομένα σε έναν υψηλότερων διαστάσεων χώρο όπου μπορούν να διαχωριστούν γραμμικά (39). Οι πιο κοινές πυρηνικές συναρτήσεις είναι:

- **Γραμμική (Linear):** $K(x, x') = x^T x'$
- **Πολυωνυμική (Polynomial):** $K(x, x') = (x^T x' + c)^d$
- **Radial Basis Function (RBF):** $K(x, x') = \exp(-\gamma\|x - x'\|^2)$

Algorithm 16 Support Vector Machines (SVMs)

Require: Σύνολο εκπαίδευσης D, Παράμετρος κανονικοποίησης C, Πυρηνική συνάρτηση K
Ensure: Βάρη w, σταθερά μετατόπισης b

1: Αρχικοποίησε w και b σε μηδέν
2: Ορίστε το πρόβλημα βελτιστοποίησης:

$$\min \frac{1}{2}\|w\|^2 + C\sum \xi_i$$

υπό τους περιορισμούς:

$$y_i(w^T \phi(x_i) + b) \geq 1 - \xi_i,\ \xi_i \geq 0$$

3: Επίλυσε το πρόβλημα βελτιστοποίησης με χρήση ενός solver τετραγωνικού προγραμματισμού
4: Προσδιόρισε τα διανύσματα υποστήριξης: Δείγματα για τα οποία $y_i(w^T \phi(x_i) + b) = 1$
5: **return** w, b

Οι SVMs είναι ιδιαίτερα αποτελεσματικές σε προβλήματα με υψηλές διαστάσεις και δεδομένα που δεν είναι γραμμικά διαχωρίσιμα, χάρη στη χρήση πυρηνικών συναρτήσεων. Παράλληλα, είναι ανθεκτικές στο *overfitting* όταν το σύνολο δεδομένων είναι μικρό και καλά οργανωμένο. Ωστόσο, η εκπαίδευσή τους μπορεί να είναι αργή για πολύ μεγάλα σύνολα δεδομένων, ενώ η επιλογή της σωστής πυρηνικής συνάρτησης και των παραμέτρων C και γ απαιτεί πειραματισμό. Παρόλα αυτά, οι SVMs έχουν εφαρμο-

γές σε τομείς όπως η αναγνώριση προσώπων, η ανάλυση κειμένου και η κατηγοριοποίηση βιολογικών δεδομένων.

3.6 Gaussian Naive Bayes Vs All

Ο καθένας από τους αλγορίθμους που είδαμε προσφέρει διαφορετικά πλεονεκτήματα και μειονεκτήματα ανάλογα με τη φύση του προβλήματος και των δεδομένων. Ο **Gaussian Naive Bayes** υποθέτει ότι τα χαρακτηριστικά ακολουθούν κανονική κατανομή και ότι είναι ανεξάρτητα μεταξύ τους. Αυτές οι υποθέσεις τον καθιστούν ιδιαίτερα γρήγορο και εύκολο στην εφαρμογή, ειδικά σε μεγάλα σύνολα δεδομένων. Ωστόσο, η ακρίβειά του μπορεί να μειωθεί αν τα δεδομένα δεν είναι ανεξάρτητα ή δεν ακολουθούν κανονική κατανομή (9).

Αντίθετα, τα **Δέντρα Απόφασης** δεν κάνουν τέτοιες υποθέσεις. Είναι ευέλικτα, εύκολα στην ερμηνεία και λειτουργούν καλά με μη γραμμικά δεδομένα. Παρόλα αυτά, μπορεί να υπερπροσαρμοστούν (*overfitting*) στα δεδομένα εκπαίδευσης, κάτι που μειώνει την απόδοσή τους σε δεδομένα δοκιμής (30).

Το **Random Forest**, μια μέθοδος *ensemble*, ξεπερνά αυτό το πρόβλημα συνδυάζοντας πολλά δέντρα απόφασης για να δημιουργήσει πιο σταθερά και ακριβή αποτελέσματα. Είναι ανθεκτικό στον θόρυβο και αποδίδει καλά σε σύνθετα προβλήματα, αλλά η εκπαίδευσή του είναι πιο αργή και απαιτεί περισσότερη υπολογιστική ισχύ σε σχέση με τον Gaussian Naive Bayes (32).

Οι K **Εγγύτεροι Γείτονες (KNN)** είναι ένας απλός αλγόριθμος που δεν κάνει υποθέσεις για την κατανομή των δεδομένων. Ωστόσο, είναι πιο αργός κατά την πρόβλεψη, καθώς πρέπει να υπολογίζει αποστάσεις για κάθε δείγμα, και η απόδοσή του εξαρτάται από την επιλογή του k και της μετρικής απόστασης (40).

Η **Λογιστική Παλινδρόμηση** είναι παρόμοια με τον Gaussian Naive Bayes, καθώς χρησιμοποιεί πιθανότητες για ταξινόμηση, αλλά δεν υποθέτει ανεξαρτησία μεταξύ των χαρακτηριστικών. Αυτό την καθιστά πιο ακριβή σε περιπτώσεις με εξαρτημένα δεδομένα, αν και η εκπαίδευσή της είναι πιο απαιτητική (8).

Οι **Μηχανές Διανυσμάτων Υποστήριξης (SVMs)** είναι ιδανικές για δεδομένα που δεν είναι γραμμικά διαχωρίσιμα, χρησιμοποιώντας πυρηνικές συναρτήσεις (*kernels*) για τη δημιουργία πιο πολύπλοκων ορίων απόφασης. Ωστόσο, απαιτούν περισσότερη υπολογιστική ισχύ και προσεκτική ρύθμιση των παραμέτρων (4).

Κεφάλαιο 4ο: Κατηγοριοποίηση Naive Bayes στην Python

4.1 Χρήση της Scikit-learn για Αλγόριθμους Naive Bayes

Η **scikit-learn** είναι μια από τις πιο γνωστές βιβλιοθήκες για μηχανική μάθηση στην Python. Παρέχει πολλά εργαλεία για την εφαρμογή αλγορίθμων κατηγοριοποίησης, όπως ο Gaussian Naive Bayes, ο Multinomial Naive Bayes και ο Bernoulli Naive Bayes. Στα προηγούμενα κεφάλαια συζητήσαμε πώς οι αλγόριθμοι αυτοί λειτουργούν και πώς η διακριτοποίηση μπορεί να βοηθήσει στην εφαρμογή τους. Η scikit-learn κάνει τη διαδικασία αυτή πολύ πιο εύκολη, προσφέροντας έτοιμες υλοποιήσεις που επιτρέπουν στους χρήστες να εστιάσουν στην ανάλυση των δεδομένων και στην ερμηνεία των αποτελεσμάτων (41).

Ένα από τα βασικά πλεονεκτήματα της scikit-learn είναι ότι όλοι οι αλγόριθμοι ακολουθούν μια απλή και συνεπή διαδικασία χρήσης. Για παράδειγμα, η υλοποίηση του *Gaussian Naive Bayes* είναι εξαιρετικά απλή (42):

```python
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

# Example data
X = [[1.0, 2.0], [1.1, 1.9], [2.0, 2.1], [2.1, 1.8]]
y = [0, 0, 1, 1]

# Splitting into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25,
    random_state=42)

# Creating and training the model
model = GaussianNB()
model.fit(X_train, y_train)

# Making predictions
y_pred = model.predict(X_test)

# Calculating accuracy
print(f"Accuracy: {accuracy_score(y_test, y_pred):.2f}")
```

Η scikit-learn προσφέρει επίσης εργαλεία για την προεπεξεργασία δεδομένων. Για παράδειγμα, για να μετατρέψουμε συνεχή δεδομένα σε κατηγορικά με διακριτοποίηση, η scikit-learn προσφέρει τον `KBinsDiscretizer` (42):

```
from sklearn.preprocessing import KBinsDiscretizer

# Continuous data
continuous_data = [[1.0], [2.5], [4.0], [6.0]]

# Discretizing into 3 bins
discretizer = KBinsDiscretizer(n_bins=3, encode='ordinal', strategy='uniform')
discretized_data = discretizer.fit_transform(continuous_data)

print(discretized_data)
```

Επιπλέον, η scikit-learn παρέχει εργαλεία για την αξιολόγηση μοντέλων, όπως το *cross-validation*:

```
from sklearn.model_selection import cross_val_score

# Cross-validation    Gaussian Naive Bayes
scores = cross_val_score(model, X, y, cv=5)
print(f"Cross-validation scores: {scores}")
print(f"Mean accuracy: {scores.mean():.2f}")
```

Η **scikit-learn** είναι εξαιρετικά ευέλικτη, διευκολύνοντας την προετοιμασία των δεδομένων, την εκπαίδευση μοντέλων και την αξιολόγηση της απόδοσής τους. Με εργαλεία σαν αυτά, οι αλγόριθμοι Naive Bayes και άλλες τεχνικές μηχανικής μάθησης γίνονται πιο προσιτές στους αναλυτές δεδομένων και τους επιστήμονες.

4.2 Υλοποιήσεις στην Python

Σε αυτό το υποκεφάλαιο, θα παρουσιάσουμε πώς εφαρμόζουμε διαφορετικές προσεγγίσεις κατηγοριοποίησης στην Python, για να απαντήσουμε στο βασικό ερώτημα της εργασίας: είναι καλύτερο να κατηγοριοποιούμε τα συνεχή δεδομένα όπως είναι ή να τα μετατρέπουμε πρώτα σε κατηγορικά μέσω διακριτοποίησης; Για την υλοποίηση όλων των μεθόδων, θα χρησιμοποιήσουμε τη βιβλιοθήκη scikit-learn, η οποία παρέχει εργαλεία για την εφαρμογή τόσο αλγορίθμων κατηγοριοποίησης όσο και τεχνικών διακριτοποίησης.

Στο πρώτο στάδιο, εφαρμόζουμε κατηγοριοποίηση στα αρχικά, συνεχούς μορφής δεδομένα χωρίς καμία μετατροπή. Χρησιμοποιούμε αλγορίθμους όπως οι Logistic Regression, Decision Trees, Random Forest, Support Vector Machines (SVM), K-Nearest Neighbors (KNN) και Gaussian Naive Bayes. Αυτοί οι αλγόριθμοι μπορούν να δουλέψουν απευθείας με συνεχή χαρακτηριστικά. Τα αποτελέσματα αξιολογούνται με βάση την ακρίβεια, ώστε να έχουμε μια απλή και ξεκάθαρη μέτρηση της απόδοσής τους.

Στο δεύτερο στάδιο, διακριτοποιούμε τα συνεχή δεδομένα χρησιμοποιώντας τρεις διαφορετικές μεθόδους: Uniform, Quantile και k-means διακριτοποίηση. Για αυτή τη διαδικασία θα χρησιμοποιήσουμε το εργαλείο KBinsDiscretizer της scikit-learn, το οποίο μας επιτρέπει να μετατρέψουμε συνεχείς τιμές σε κατηγορικές με ευκολία. Μετά τη διακριτοποίηση, εφαρμόζουμε αλγορίθμους που είναι σχεδιασμένοι για κατηγορικά δεδομένα, όπως οι Multinomial Naive Bayes, Bernoulli Naive Bayes και Categorical Naive Bayes.

Στη συνέχεια, συγκρίνουμε την ακρίβεια των δύο προσεγγίσεων – της απευθείας κατηγοριοποίησης και της κατηγοριοποίησης μετά από διακριτοποίηση. Ο στόχος είναι να δούμε ποια μέθοδος αποδίδει καλύτερα και υπό ποιες συνθήκες. Ξεκινάμε με την απευθείας κατηγοριοποίηση των συνεχών δεδομένων χρησιμοποιώντας τους αλγορίθμους της scikit-learn.

Η **κλιμάκωση των δεδομένων (scaling)** είναι μια σημαντική διαδικασία προεπεξεργασίας στη μηχανική μάθηση, η οποία διασφαλίζει ότι όλα τα χαρακτηριστικά ενός dataset έχουν την ίδια κλίμακα. Αυτό είναι κρίσιμο για αλγορίθμους που βασίζονται σε αποστάσεις, όπως οι K-Nearest Neighbors (KNN) και οι Support Vector Machines (SVMs), ή για αλγορίθμους που επηρεάζονται από μεγέθη χαρακτηριστικών, όπως η Logistic Regression. Όταν τα δεδομένα δεν είναι κλιμακωμένα, τα χαρακτηριστικά με μεγαλύτερες τιμές μπορεί να κυριαρχήσουν, ενώ τα μικρότερα να αγνοηθούν. Με την κλιμάκωση, όλα τα χαρακτηριστικά συνεισφέρουν ισότιμα στη διαδικασία εκπαίδευσης του μοντέλου.

```
def scale_column(column_data):
    data_reshaped = np.array(column_data).reshape(-1, 1)
    # Fit and transform the data
    return scaler.fit_transform(data_reshaped)
```

Σχήμα 4.1: Σύνταξη της μεθόδου scale_column()

Σε αυτό το κομμάτι κώδικα, βλέπουμε τη συνάρτηση **scale_column()**, η οποία χρησιμοποιείται για την κλιμάκωση μιας μεμονωμένης στήλης δεδομένων. Αρχικά, η στήλη μετατρέπεται σε πίνακα NumPy με μία μόνο στήλη, ώστε να είναι συμβατή με τον scaler. Στη συνέχεια, εφαρμόζεται η κλιμάκωση μέσω του `fit_transform`, που προσαρμόζει τον scaler στα δεδομένα και τα μετασχηματίζει κατάλληλα. Το αποτέλεσμα είναι μια κλιμακωμένη εκδοχή της αρχικής στήλης, όπου οι τιμές έχουν προσαρμοστεί για να βρίσκονται σε συγκεκριμένο εύρος ή να έχουν μηδενική μέση τιμή και τυπική απόκλιση ίση με ένα.

```
def scale_table(data):
    for column in list(data):
        if column == "Class":
            continue
        data[column] = scale_column(data[column])
    return data
```

Σχήμα 4.2: Σύνταξη της μεθόδου scale_table()

Στον δεύτερο κομμάτι κώδικα, έχουμε τη συνάρτηση **scale_table()**, η οποία εφαρμόζει την κλιμάκωση σε έναν ολόκληρο πίνακα δεδομένων. Με τη βοήθεια ενός βρόχου `for`, η συνάρτηση περνάει από κάθε στήλη του πίνακα. Αν η στήλη είναι η `Class`, που συνήθως περιέχει την κατηγορία-στόχο, παραλείπεται, γιατί δεν χρειάζεται κλιμάκωση. Για τις υπόλοιπες στήλες, καλείται η `scale_column` για να γίνει η

κλιμάκωση. Στο τέλος, επιστρέφεται ο πίνακας, όπου όλες οι αριθμητικές στήλες έχουν κλιμακωθεί κατάλληλα.

```python
def read_and_save_folds(relative_file_path: str):
    fold_counter = 0
    root_path = "/".join(relative_file_path.split("/")[:-1]) + "/"
    dataset_name = relative_file_path.split("/")[-1].split(".")[0]
    initial_data = read_data(relative_file_path)
    print(f"Data shape: {initial_data.shape}")
    scaled_data = scale_table(initial_data)
    x_data = scaled_data.loc[:, scaled_data.columns != "Class"]

    for train_indices, test_indices in skf.split(x_data, scaled_data["Class"]):
        x_train = x_data.iloc[train_indices]
        y_train = scaled_data["Class"].iloc[train_indices]
        x_y_data_train_data = x_train.join(y_train)
        new_train_file_name = f"{root_path + dataset_name}_train_{fold_counter}.csv"
        x_y_data_train_data.to_csv(new_train_file_name, index=False)

        x_test = x_data.iloc[test_indices]
        y_test = scaled_data["Class"].iloc[test_indices]
        new_test_file_name = f"{root_path + dataset_name}_test_{fold_counter}.csv"
        x_y_data_test_data = x_test.join(y_test)
        x_y_data_test_data.to_csv(new_test_file_name, index=False)
        fold_counter += 1
```

Σχήμα 4.3: Σύνταξη της μεθόδου read_and_save_folds()

Εδώ βλέπουμε την υλοποίηση της **read_and_save_folds()**, η οποία χρησιμοποιείται για τη διαχείριση ενός dataset μέσω της μεθόδου **cross-validation**. Πιο συγκεκριμένα, η συνάρτηση διαβάζει δεδομένα από ένα αρχείο, κλιμακώνει τις αριθμητικές στήλες, δημιουργεί τα train και test splits για κάθε fold της cross-validation, και αποθηκεύει τα αντίστοιχα σύνολα δεδομένων σε αρχεία CSV.

Η συνάρτηση ξεκινά λαμβάνοντας το μονοπάτι του αρχείου ως είσοδο. Αρχικά, καθορίζει τη ρίζα του μονοπατιού (`root_path`) και το όνομα του dataset, ενώ στη συνέχεια διαβάζει τα δεδομένα από το αρχείο μέσω της `read_data`. Τα αρχικά δεδομένα αποθηκεύονται και η μορφή τους εκτυπώνεται για έλεγχο.

Μετά την ανάγνωση, τα δεδομένα κλιμακώνονται μέσω της `scale_table`, εξαιρώντας τη στήλη `Class`, που περιέχει την κατηγορία στόχο. Το κλιμακωμένο dataset διαχωρίζεται σε χαρακτηριστικά (`x_data`) και κατηγορίες (`Class`).

Έπειτα, η συνάρτηση χρησιμοποιεί ένα αντικείμενο `skf` (`StratifiedKFold` από τη *scikit-learn*) για να δημιουργήσει τα train και test splits για κάθε fold. Για κάθε fold, τα indices των train και test συνόλων χρησιμοποιούνται για να διαχωριστούν τα δεδομένα. Οι αντίστοιχες στήλες με τις κατηγορίες (`Class`) προστίθενται στα train και test δεδομένα. Τα δεδομένα κάθε fold αποθηκεύονται σε ξεχωριστά αρχεία CSV, με το όνομα του αρχείου να περιλαμβάνει το dataset και τον αριθμό του fold, τόσο για τα train όσο και για τα test δεδομένα.

Ο μετρητής `fold_counter` διασφαλίζει ότι κάθε fold έχει ένα μοναδικό όνομα αρχείου. Με αυτόν τον τρόπο, η συνάρτηση επιτρέπει τη διαχείριση και αποθήκευση των splits ενός dataset για χρήση σε

διαδικασίες εκπαίδευσης και αξιολόγησης μοντέλων. Για να αποδείξουμε ότι τα πειράματα που τρέχουμε έχουν **στατιστική σημαντικότητα (statistical significance)**, πρέπει να τα τρέξουμε με k-cross fold validation

```
def discretize_column(column_data, n_bins, strategy, encode='ordinal'):
    kbd = KBinsDiscretizer(n_bins=n_bins, encode=encode, strategy=strategy)
    data_reshaped = np.array(column_data).reshape(-1, 1)
    return kbd.fit_transform(data_reshaped).reshape(-1)
```

Σχήμα 4.4: Σύνταξη της μεθόδου discretize_column()

Σε αυτό το σημείο ξεκινάει η διαδικασία διακριτοποίησης δεδομένων, με πρώτη συνάρτηση, την **discretize_column()**, η οποία εφαρμόζει τη διακριτοποίηση σε μια μόνο στήλη δεδομένων. Χρησιμοποιεί το εργαλείο **KBinsDiscretizer** από τη *scikit-learn*, το οποίο δημιουργεί κατηγορίες για τα δεδομένα βασισμένο στον αριθμό των bins (`n_bins`), τη στρατηγική (`strategy`: *uniform*, *quantile* ή *k-means*) και τον τρόπο κωδικοποίησης (`encode`). Η στήλη μετατρέπεται σε μορφή πίνακα και επιστρέφονται οι διακριτοποιημένες τιμές.

```
def calculate_bins(column_data: pd.Series) -> int:
    # Apply Freedman-Diaconis rule for bin calculation on individual columns
    edges = np.histogram_bin_edges(column_data, bins='fd')
    n_bins = len(edges) - 1
    return n_bins
```

Σχήμα 4.5: Σύνταξη της μεθόδου calculate_bins()

Η συνάρτηση **calculate_bins()** υπολογίζει αυτόματα τον βέλτιστο αριθμό bins για μια συγκεκριμένη στήλη. Βασίζεται στα άκρα (`edges`) που προκύπτουν από τη συνάρτηση `np.histogram_bin_edges` και υπολογίζει τον αριθμό των διαστημάτων αφαιρώντας ένα από το συνολικό πλήθος άκρων.

```
def discretize_table(data: pd.DataFrame, n_bins=None, strategy='uniform', encode='ordinal')
-> tuple[pd.DataFrame, int]:
    for column in data.columns:
        if column != "Class":
            if n_bins is None:
                n_bins = calculate_bins(data[column])
            data.loc[:, column] = discretize_column(data[column], n_bins=n_bins,
                                                    strategy=strategy, encode=encode)
    return data, n_bins
```

Σχήμα 4.6: Σύνταξη της μεθόδου discretize_table()

Η συνάρτηση **discretize_table()** εφαρμόζει τη διακριτοποίηση σε έναν ολόκληρο πίνακα δεδομένων (`DataFrame`). Περνάει μέσα από κάθε στήλη του πίνακα, εκτός από τη στήλη `Class`, που θεωρείται στήλη κατηγορίας στόχου. Για κάθε στήλη, αν δεν έχει οριστεί συγκεκριμένος αριθμός bins, χρησιμοποιείται η `calculate_bins` για να υπολογιστούν αυτόματα. Στη συνέχεια, καλείται η `discretize_column` για να διακριτοποιήσει τη στήλη, και τα αποτελέσματα επιστρέφονται στον πίνακα. Η συνάρτηση επιστρέφει τον διακριτοποιημένο πίνακα μαζί με τον αριθμό των bins που χρησιμοποιήθηκαν.

```
classification_algorithms = [
    ("Logistic regression", LogisticRegression(solver='saga', max_iter=500)),
    ("Decision trees", DecisionTreeClassifier()),
    ("Random forest", RandomForestClassifier()),
    ("SVMs", SVC()),
    ("KNN", KNeighborsClassifier()),
    ("Gaussian Naive Bayes", GaussianNB()),
    # ========== Categorical Classifiers ======================
    ("Multinomial Naive Bayes", MultinomialNB()),
    ("Bernoulli Naive Bayes", BernoulliNB()),
    ("Categorical Naive Bayes", CategoricalNB())
]
```

Σχήμα 4.7: Πίνακας αλγορίθμων κατηγοριοποίησης που θα χρησιμοποιηθούν

Εδώ δημιουργείται ένας πίνακας που ονομάζεται `classification_algorithms`, ο οποίος περιλαμβάνει τους αλγορίθμους ταξινόμησης που θα χρησιμοποιηθούν. Κάθε στοιχείο του πίνακα αποτελείται από το όνομα του αλγορίθμου (ως κείμενο) και τον αντίστοιχο ταξινομητή από τη βιβλιοθήκη scikit-learn. Στον πίνακα περιλαμβάνονται δύο κατηγορίες αλγορίθμων. Η πρώτη περιέχει αλγόριθμους που λειτουργούν με συνεχόμενα δεδομένα, (Logistic Regression, τα Decision Trees, το Random Forest, τα SVMs, το KNN και ο Gaussian Naive Bayes). Αυτοί οι αλγόριθμοι μπορούν να εφαρμοστούν απευθείας σε δεδομένα αριθμητικής μορφής χωρίς περαιτέρω προεπεξεργασία. Η δεύτερη κατηγορία περιλαμβάνει αλγόριθμους που είναι σχεδιασμένοι για κατηγορικά δεδομένα (Multinomial Naive Bayes, Bernoulli Naive Bayes και Categorical Naive Bayes). Αυτοί οι αλγόριθμοι χρησιμοποιούνται όταν τα δεδομένα έχουν διακριτοποιηθεί.

```
def get_file_tuples(directory):
    dataset_name = directory.split("/")[-2]
    data_tuples = []
    for i in range(4):
        pattern_train = f"{dataset_name}_train_{i}.csv"
        pattern_test = f"{dataset_name}_test_{i}.csv"
        train_file_path = directory + pattern_train
        test_file_path = directory + pattern_test
        data_tuples.append((train_file_path, test_file_path))
    return data_tuples
```

Σχήμα 4.8: Σύνταξη της μεθόδου get_file_tuples()

Η συνάρτηση **get_file_tuples()** αναλαμβάνει να δημιουργήσει ζεύγη αρχείων για κάθε fold των δεδομένων cross-validation. Χρησιμοποιώντας τον φάκελο που παρέχεται ως είσοδος, παράγει τα paths για τα αρχεία train και test για κάθε fold και επιστρέφει μια λίστα από ζεύγη (train, test).

```
def train_and_eval(directory: str):
    reports_dir = f"{directory}/reports"
    if not os.path.exists(reports_dir):
        os.makedirs(reports_dir)
    report_path = f"{reports_dir}/report.txt"
    if os.path.exists(report_path):
        os.remove(report_path)

    for classification_algorithm_name, classification_impl in classification_algorithms:
        classification_algorithm_names = ["Decision trees - Categorical",
                                          "KNN - Categorical",
                                          "Multinomial Naive Bayes",
                                          "Bernoulli Naive Bayes",
                                          "Categorical Naive Bayes"
                                          ]
        discretize = classification_algorithm_name in classification_algorithm_names

        truths = []
        predictions = []

        file_tuples = get_file_tuples(directory)
        for file_tuple in file_tuples:
            train_csv = pd.read_csv(file_tuple[0])
            test_csv = pd.read_csv(file_tuple[1])

            x_train = train_csv.loc[:, train_csv.columns != 'Class']
            y_train = train_csv.loc[:, train_csv.columns == 'Class']

            x_test = test_csv.loc[:, test_csv.columns != 'Class']
            y_test = test_csv.loc[:, test_csv.columns == 'Class']

            if discretize:
                x_train, n_bins = discretize_table(x_train, strategy='uniform')
                x_test, _ = discretize_table(x_test, n_bins=n_bins, strategy='uniform')
                if x_train.shape[1] != x_test.shape[1]:
                    raise ValueError("Mismatched columns....")

            y_train = y_train.squeeze()
            classification_impl.fit(x_train, y_train)
            # Predict the labels for the test set
            y_pred = classification_impl.predict(x_test)

            # Evaluate the classifier
            for v in y_test.values.flatten().tolist():
                truths.append(v)
            for v in y_pred:
                predictions.append(v)
```

Σχήμα 4.9: Σύνταξη της μεθόδου train_and_eval() [1 από 2]

```
        if len(truths) != len(predictions):
            raise ValueError("Mismatched shapes between predictions and true labels.")
        accuracy = accuracy_score(truths, predictions)
        report = classification_report(truths, predictions, zero_division=1)
        # tn, fp, fn, tp = confusion_matrix(y_tests, y_preds).ravel()

        classification_algorithm_print_text = f"Classification  algorithm:
        {classification_algorithm_name}\n"
        # confusion_matrix_print_text = f"TP: {tp}\nTN: {tn}\nFP: {fp}\nFN: {fn}\n"
        accuracy_print_text = f"Accuracy: {accuracy}\n"
        classification_report_print_text = f"Classification  report: \n{report}\n"

        with open(report_path, "a") as f:
            print(f"{classification_algorithm_print_text}")
            f.write(f"{classification_algorithm_print_text}")
            print(f"Has data been descritised: {discretize}\n")
            f.write(f"Has data been descritised: {discretize}\n")
            print(accuracy_print_text)
            f.write(accuracy_print_text)
            print(f"{classification_report_print_text}\n\n")
            f.write(f"{classification_report_print_text}\n\n")
```

Σχήμα 4.10: Σύνταξη της μεθόδου train_and_eval() [2 από 2]

Σε αυτό το σημείο υλοποιείται η διαδικασία εκπαίδευσης και αξιολόγησης των ταξινομητών (classifiers) σε datasets μέσω της συνάρτησης **train_and_eval()**. Ο κώδικας είναι δομημένος για να χειρίζεται αρχεία δεδομένων, να εκπαιδεύει διαφορετικούς αλγορίθμους ταξινόμησης, να αξιολογεί την απόδοσή τους και να αποθηκεύει τα αποτελέσματα σε αναφορές.

Η συνάρτηση ξεκινά δημιουργώντας έναν φάκελο αναφορών και διαγράφοντας τυχόν υπάρχουσα αναφορά, για να διασφαλίσει ότι τα αποτελέσματα είναι καθαρά. Στη συνέχεια, χρησιμοποιεί μια λίστα ταξινομητών (classification algorithms) και για κάθε ταξινομητή ελέγχει αν τα δεδομένα πρέπει να διακριτοποιηθούν. Η διακριτοποίηση εφαρμόζεται αν ο ταξινομητής είναι σχεδιασμένος για κατηγορικά δεδομένα, όπως οι Multinomial, Bernoulli ή Categorical Naive Bayes.

Για κάθε fold (train-test ζεύγος) από τα δεδομένα, διαβάζονται τα αντίστοιχα αρχεία train και test. Τα δεδομένα εκπαιδεύονται με τον τρέχοντα ταξινομητή, ενώ αν χρειάζεται διακριτοποίηση, εφαρμόζεται πριν την εκπαίδευση. Μετά την εκπαίδευση, ο ταξινομητής κάνει προβλέψεις για το test set. Τα πραγματικά labels (truths) και οι προβλέψεις (predictions) καταγράφονται για αξιολόγηση.

Στο τέλος, συγκρίνονται τα πραγματικά labels με τις προβλέψεις για να υπολογιστεί η ακρίβεια και να δημιουργηθεί αναφορά ταξινόμησης. Τα αποτελέσματα κάθε ταξινομητή αποθηκεύονται σε ένα αρχείο αναφοράς (`report.txt`), το οποίο περιλαμβάνει την ακρίβεια, την αναφορά ταξινόμησης και αν χρησιμοποιήθηκε διακριτοποίηση. Έτσι έχουμε έναν ολοκληρωμένο μηχανισμό εκπαίδευσης και αξιολόγησης, ενσωματώνοντας τη διακριτοποίηση και υποστηρίζοντας τη σύγκριση ταξινομητών σε διαφορετικά datasets.

Κεφάλαιο 5ο: Πειραματική μελέτη

5.1 Σύνολα Δεδομένων

Στην παρούσα εργασία χρησιμοποιήθηκαν δέκα διαφορετικά σύνολα δεδομένων, τα οποία λήφθηκαν από το αποθετήριο δεδομένων του KEEL. Τα datasets αυτά επιλέχθηκαν για την ποικιλομορφία τους, καλύπτοντας διαφορετικούς αριθμούς χαρακτηριστικών, μεγεθών και τύπων κατηγοριών. Αυτό επιτρέπει την αξιολόγηση της απόδοσης των αλγορίθμων ταξινόμησης σε διάφορες περιπτώσεις. Όλα τα datasets είχαν αρχική μορφή ”.dat” και συνοδεύονταν από περιγραφή των χαρακτηριστικών τους.

Το dataset Bupa περιέχει δεδομένα που σχετίζονται με ηπατικές διαταραχές. Έχει 345 δείγματα και 7 αριθμητικά χαρακτηριστικά. Το πρόβλημα αφορά τη διάκριση μεταξύ ασθενών και μη ασθενών, κάνοντάς το χρήσιμο για δοκιμές σε ιατρικά δεδομένα.

Το dataset Iris είναι ένα από τα πιο γνωστά σύνολα δεδομένων και περιέχει 150 δείγματα με 4 αριθμητικά χαρακτηριστικά. Αυτά περιγράφουν το μήκος και το πλάτος του κάλυκα και των πετάλων τριών ειδών λουλουδιών: setosa, versicolor και virginica.

Το dataset Letter αποτελείται από 20.000 δείγματα και περιλαμβάνει 16 αριθμητικά χαρακτηριστικά. Χρησιμοποιείται για την αναγνώριση γραμμάτων του λατινικού αλφαβήτου (A-Z). Λόγω του μεγάλου μεγέθους του, είναι ιδανικό για τη μελέτη της απόδοσης αλγορίθμων σε μεγάλης κλίμακας προβλήματα.

Το dataset Magic περιλαμβάνει 19.000 δείγματα και έχει 10 αριθμητικά χαρακτηριστικά. Αφορά τη διάκριση μεταξύ πραγματικών συμβάντων ακτίνων γ και ψευδών συναγερμών, κάνοντάς το χρήσιμο για προβλήματα ανάλυσης δεδομένων προσομοίωσης.

Το dataset Ring περιέχει 7400 δείγματα και 20 αριθμητικά χαρακτηριστικά. Χρησιμοποιείται για τη διάκριση δύο κατηγοριών που σχηματίζουν ομόκεντρους δακτυλίους, αποτελώντας ενδιαφέρον παράδειγμα για γεωμετρικά δεδομένα.

Το dataset Segment έχει 2310 δείγματα και 19 αριθμητικά χαρακτηριστικά. Αφορά την ταξινόμηση τμημάτων εικόνας σε προκαθορισμένες κατηγορίες, κάνοντάς το ιδανικό για εφαρμογές στην επεξεργασία εικόνας.

Το dataset Texture περιλαμβάνει 5500 δείγματα και 40 αριθμητικά χαρακτηριστικά. Εστιάζει στην ανάλυση της υφής εικόνων και στην κατηγοριοποίηση σε διαφορετικές τάξεις, προσφέροντας δεδομένα για προβλήματα ανάλυσης εικόνας.

Το dataset Wine περιλαμβάνει δεδομένα χημικής ανάλυσης κρασιών. Έχει 178 δείγματα και 13 αριθμητικά χαρακτηριστικά, που σχετίζονται με τη χημική σύσταση κρασιών από τρεις διαφορετικές ποικιλίες. Χρησιμοποιείται συχνά για προβλήματα πολυκατηγορικής ταξινόμησης.

Το dataset Wisconsin περιέχει 699 δείγματα και 10 χαρακτηριστικά. Χρησιμοποιείται ευρέως για τη διάγνωση καρκίνου του μαστού, με στόχο τη διάκριση μεταξύ καλοήθων και κακοήθων όγκων.

Τέλος, το dataset Yeast περιλαμβάνει 1484 δείγματα και 8 αριθμητικά χαρακτηριστικά. Αφορά την ταξι-

νόμηση πρωτεϊνών σε διαφορετικές τοποθεσίες εντός του κυττάρου, κάνοντάς το χρήσιμο για προβλήματα βιολογικών δεδομένων.

Όλα τα παραπάνω σύνολα δεδομένων προσφέρουν ποικιλία στις διαστάσεις, τον αριθμό κατηγοριών και τα χαρακτηριστικά, καθιστώντας τα ιδανικά για την αξιολόγηση αλγορίθμων κατηγοριοποίησης. Επίσης, η χρήση γνωστών datasets εξασφαλίζει τη δυνατότητα σύγκρισης των αποτελεσμάτων με πιθανές μελλοντικές μελέτες, εφόσον είναι εύκολα προσβάσιμα σε κάθε ενδιαφερόμενο. Τα δεδομένα δέχθηκαν προεπεξεργασία, όπου ήταν απαραίτητο, με κλιμάκωση και διακριτοποίηση για να προσαρμοστούν στις απαιτήσεις των αλγορίθμων ταξινόμησης.

5.2 Εγκαθίδρυση Πειραμάτων (Experimental Setup)

Για τη διεξαγωγή της πειραματικής μελέτης, ακολουθήθηκε μια οργανωμένη διαδικασία που περιλαμβάνει τη χρήση του k-fold cross-validation, την εφαρμογή διαφορετικών αλγορίθμων κατηγοριοποίησης και τη μέτρηση της απόδοσής τους με βάση την ακρίβεια. Αυτή η μέθοδος βοηθά να εξασφαλίσουμε ότι τα αποτελέσματά μας είναι αξιόπιστα και αντιπροσωπευτικά. Όπως έχουμε ήδη αναφέρει στο προηγούμενο κεφάλαιο, η ανάλυση μας βασίζεται σε διαφορετικούς αλγορίθμους και στη χρήση της scikit-learn για την υλοποίηση.

Ξεκινήσαμε με το διαχωρισμό κάθε dataset σε σύνολα εκπαίδευσης και δοκιμής, χρησιμοποιώντας τη μέθοδο του k-fold cross-validation. Αυτή η τεχνική διασφαλίζει ότι όλα τα δεδομένα χρησιμοποιούνται τόσο για εκπαίδευση όσο και για δοκιμή, μειώνοντας την πιθανότητα μεροληψίας στα αποτελέσματα. Στη συγκεκριμένη μελέτη χρησιμοποιήσαμε 4-fold cross-validation, χωρίζοντας τα δεδομένα σε τέσσερα ίσα μέρη. Σε κάθε επανάληψη, τρία μέρη χρησιμοποιούνταν για εκπαίδευση και το τέταρτο για δοκιμή.

Όπως εξηγήσαμε στο προηγούμενο κεφάλαιο, η πειραματική διαδικασία περιλαμβάνει δύο βασικά στάδια. Στο πρώτο στάδιο, εφαρμόσαμε αλγορίθμους κατηγοριοποίησης απευθείας στα συνεχή δεδομένα. Οι αλγόριθμοι που χρησιμοποιήθηκαν σε αυτό το στάδιο είναι οι Logistic Regression, Decision Trees, Random Forest, SVM, KNN και Gaussian Naive Bayes. Αυτοί οι αλγόριθμοι είναι σχεδιασμένοι να δουλεύουν με δεδομένα συνεχούς μορφής και η απόδοσή τους αξιολογήθηκε με βάση την ακρίβεια των προβλέψεών τους.

Στο δεύτερο στάδιο, τα δεδομένα μετατράπηκαν σε κατηγορικά μέσω της διαδικασίας διακριτοποίησης, όπως έχουμε ήδη παρουσιάσει στο προηγούμενο κεφάλαιο. Χρησιμοποιήθηκαν τρεις στρατηγικές διακριτοποίησης: uniform, quantile και k-means. Η διακριτοποίηση πραγματοποιήθηκε μέσω της συνάρτησης discretize_table(), που υλοποιήθηκε με τη βοήθεια της scikit-learn. Στη συνέχεια, εφαρμόστηκαν οι αλγόριθμοι Multinomial Naive Bayes, Bernoulli Naive Bayes και Categorical Naive Bayes, οι οποίοι απαιτούν κατηγορικά δεδομένα.

Η απόδοση όλων των αλγορίθμων αξιολογήθηκε αποκλειστικά με βάση την ακρίβεια, δηλαδή το ποσοστό των σωστών προβλέψεων. Τα αποτελέσματα καταγράφηκαν σε αρχεία αναφοράς για κάθε συνδυασμό dataset, μεθόδου διακριτοποίησης και αλγορίθμου κατηγοριοποίησης. Η αυτόματη δημιουργία αυτών των αρχείων εξασφάλισε τη συνέπεια και την οργανωμένη καταγραφή των δεδομένων.

Όπως αναφέραμε στο προηγούμενο κεφάλαιο, η scikit-learn αποτέλεσε το κύριο εργαλείο μας για την

υλοποίηση των αλγορίθμων, τη διακριτοποίηση δεδομένων και την αξιολόγηση των αποτελεσμάτων. Η χρήση της μας επέτρεψε να υλοποιήσουμε τη διαδικασία εύκολα και αποδοτικά, εξασφαλίζοντας ακρίβεια και δυνατότητα επαναληψιμότητας στα πειράματα.

5.3 Πειραματικά αποτελέσματα

Στην ενότητα αυτή θα παρουσιαστούν τα αποτελέσματα των πειραμάτων που διεξήχθησαν για την αξιολόγηση των αλγορίθμων κατηγοριοποίησης. Εστιάζουμε στη σύγκριση των αλγορίθμων που εφαρμόζονται σε συνεχή δεδομένα με εκείνους που λειτουργούν σε διακριτοποιημένα δεδομένα. Για κάθε dataset, θα παρουσιαστούν δύο πίνακες. Ο πρώτος πίνακας περιλαμβάνει τις ακρίβειες των αλγορίθμων που δουλεύουν με συνεχή δεδομένα. Ο δεύτερος πίνακας θα περιλαμβάνει τις ακρίβειες των αλγορίθμων που εφαρμόζονται σε διακριτοποιημένα δεδομένα, καθώς και την ακρίβεια του αλγορίθμου με την καλύτερη απόδοση από τον πρώτο πίνακα, ώστε να διευκολύνεται η σύγκριση. Η παρουσίαση αυτή έχει στόχο να αναδείξει ποιες προσεγγίσεις είναι πιο αποτελεσματικές σε διαφορετικούς τύπους δεδομένων. Παράλληλα, θα χρησιμοποιηθούν διαγράμματα για την οπτικοποίηση των αποτελεσμάτων και την καλύτερη κατανόηση των διαφορών. Η περιγραφή που ακολουθεί θα δώσει έμφαση στις διαφορές που προκύπτουν από τη χρήση διακριτοποίησης και στη συνολική απόδοση των αλγορίθμων.

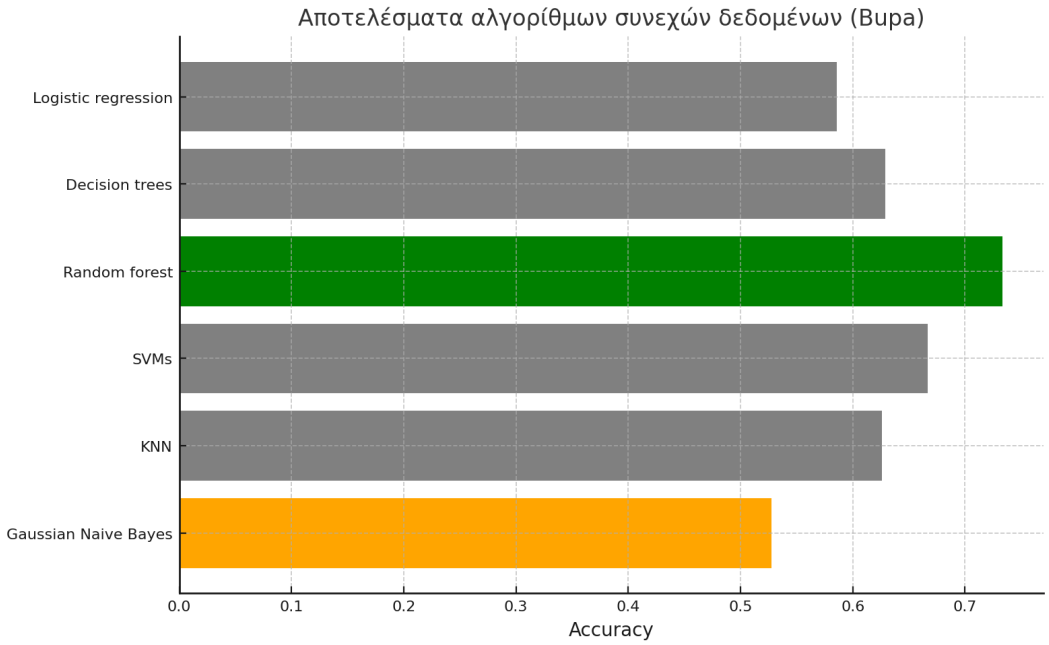
5.3.1 Αποτελέσματα Bupa dataset

Algorithm	**Accuracy**
Logistic regression	0.5855
Decision trees	0.629
Random forest	**0.7333**
SVMs	0.6667
KNN	0.6261
Gaussian Naive Bayes	**0.5275**

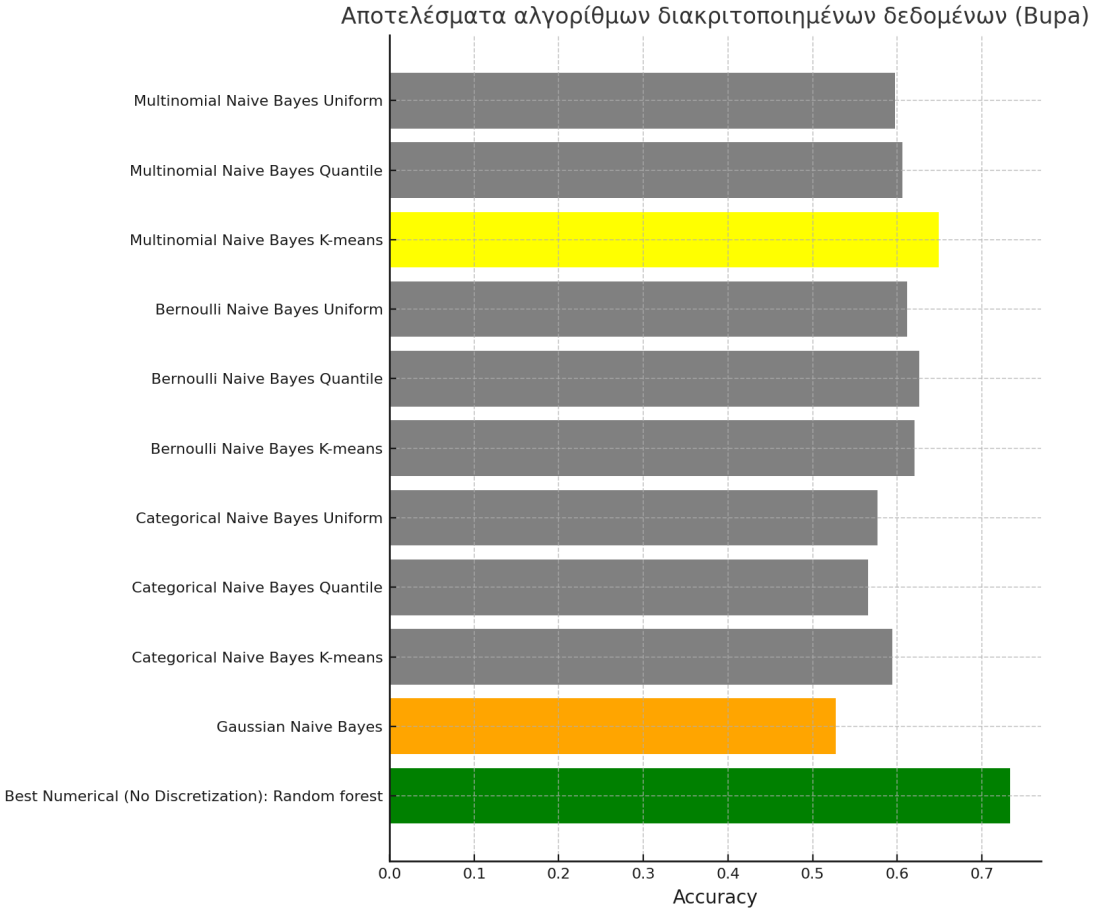
Πίνακας 5.1: Πίνακας ακριβείας αλγορίθμων συνεχών δεδομένων στο dataset Bupa

Algorithm	**Accuracy**
Multinomial Naive Bayes Uniform	0.5971
Multinomial Naive Bayes Quantile	0.6058
Multinomial Naive Bayes K-means	**0.6493**
Bernoulli Naive Bayes Uniform	0.6116
Bernoulli Naive Bayes Quantile	0.6261
Bernoulli Naive Bayes K-means	0.6203
Categorical Naive Bayes Uniform	0.5768
Categorical Naive Bayes Quantile	0.5652
Categorical Naive Bayes K-means	0.5942
Gaussian Naive Bayes	**0.5275**
Best Numerical (No Discretization): Random forest	**0.7333**

Πίνακας 5.2: Πίνακας ακριβείας αλγορίθμων διακριτοποιημένων δεδομένων στο dataset Bupa



Σχήμα 5.1: Διάγραμμα αρκιβείας αλγορίθμων συνεχών δεδομένων στο dataset Bupa



Σχήμα 5.2: Διάγραμμα αρκιβείας αλγορίθμων διακριτοποιημένων δεδομένων στο dataset Bupa

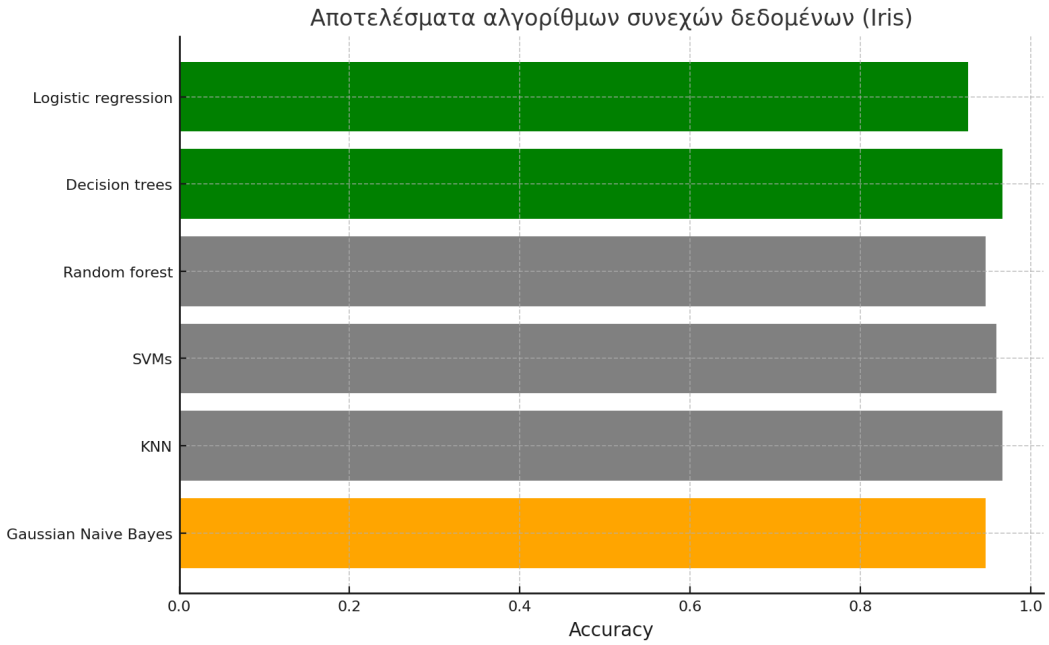
5.3.2 Αποτελέσματα Iris dataset

Algorithm	Accuracy
Logistic regression	0.9267
Decision trees	**0.9667**
Random forest	0.9467
SVMs	0.96
KNN	0.9667
Gaussian Naive Bayes	**0.9467**

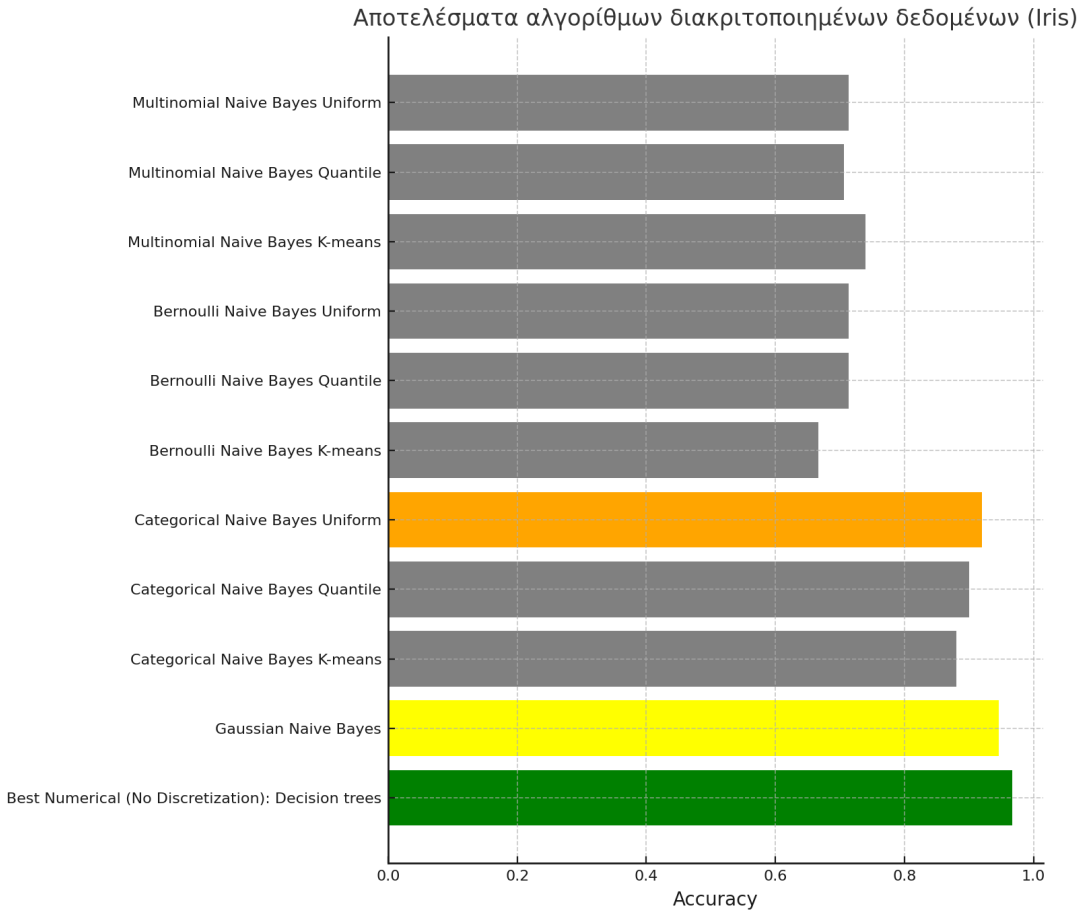
Πίνακας 5.3: Πίνακας ακριβείας αλγορίθμων συνεχών δεδομένων στο dataset Iris

Algorithm	Accuracy
Multinomial Naive Bayes Uniform	0.7133
Multinomial Naive Bayes Quantile	0.7067
Multinomial Naive Bayes K-means	0.74
Bernoulli Naive Bayes Uniform	0.7133
Bernoulli Naive Bayes Quantile	0.7133
Bernoulli Naive Bayes K-means	0.6667
Categorical Naive Bayes Uniform	**0.92**
Categorical Naive Bayes Quantile	0.9
Categorical Naive Bayes K-means	0.88
Gaussian Naive Bayes	**0.9467**
Best Numerical (No Discretization): Decision trees	**0.9667**

Πίνακας 5.4: Πίνακας ακριβείας αλγορίθμων διακριτοποιημένων δεδομένων στο dataset Iris



Σχήμα 5.3: Διάγραμμα αρκιβείας αλγορίθμων συνεχών δεδομένων στο dataset Iris



Σχήμα 5.4: Διάγραμμα αρκιβείας αλγορίθμων διακριτοποιημένων δεδομένων στο dataset Iris

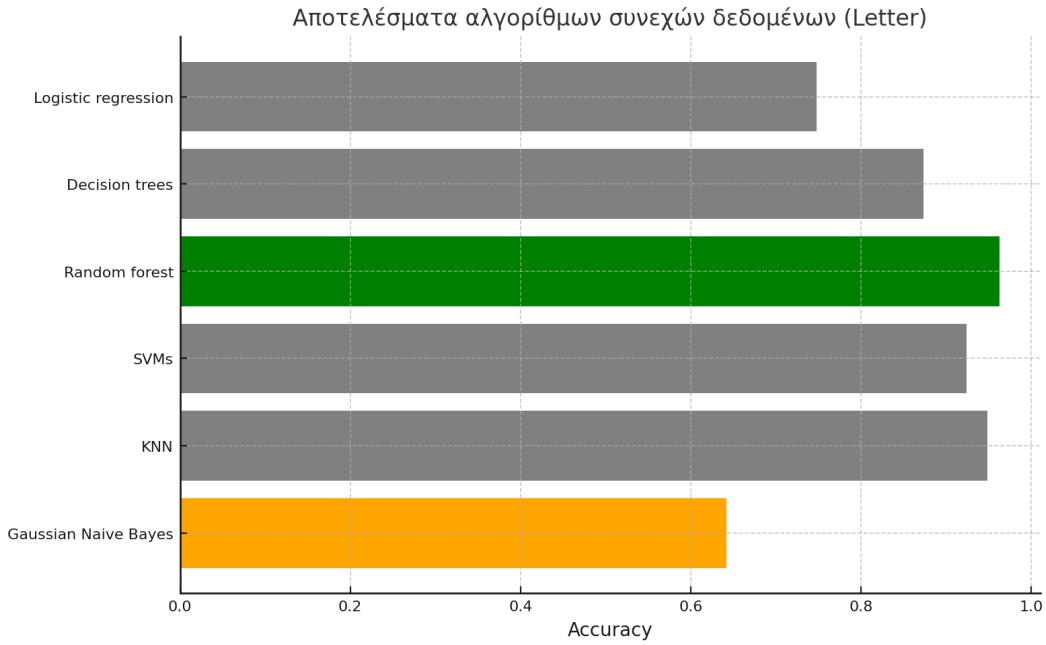
5.3.3 Αποτελέσματα Letter dataset

Algorithm	Accuracy
Logistic regression	0.74785
Decision trees	0.874
Random forest	**0.96265**
SVMs	0.9246
KNN	0.9488
Gaussian Naive Bayes	**0.6417**

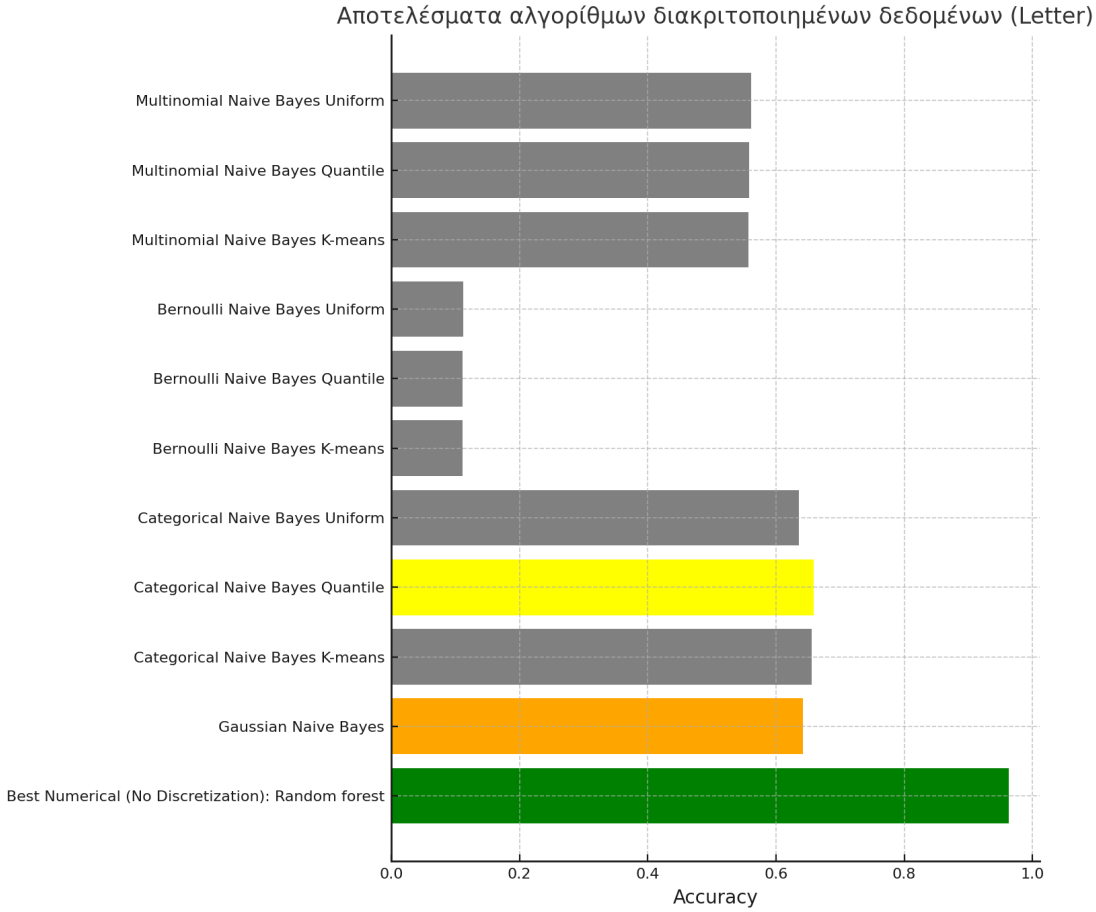
Πίνακας 5.5: Πίνακας ακριβείας αλγορίθμων συνεχών δεδομένων στο dataset Letter

Algorithm	Accuracy
Multinomial Naive Bayes Uniform	0.5615
Multinomial Naive Bayes Quantile	0.5583
Multinomial Naive Bayes K-means	0.5575
Bernoulli Naive Bayes Uniform	0.11255
Bernoulli Naive Bayes Quantile	0.11165
Bernoulli Naive Bayes K-means	0.11125
Categorical Naive Bayes Uniform	0.63555
Categorical Naive Bayes Quantile	**0.6589**
Categorical Naive Bayes K-means	0.65605
Gaussian Naive Bayes	**0.6417**
Best Numerical (No Discretization): Random forest	**0.96265**

Πίνακας 5.6: Πίνακας ακριβείας αλγορίθμων διακριτοποιημένων δεδομένων στο dataset Letter



Σχήμα 5.5: Διάγραμμα αρκιβείας αλγορίθμων συνεχών δεδομένων στο dataset Letter



Σχήμα 5.6: Διάγραμμα αρκιβείας αλγορίθμων διακριτοποιημένων δεδομένων στο dataset Letter

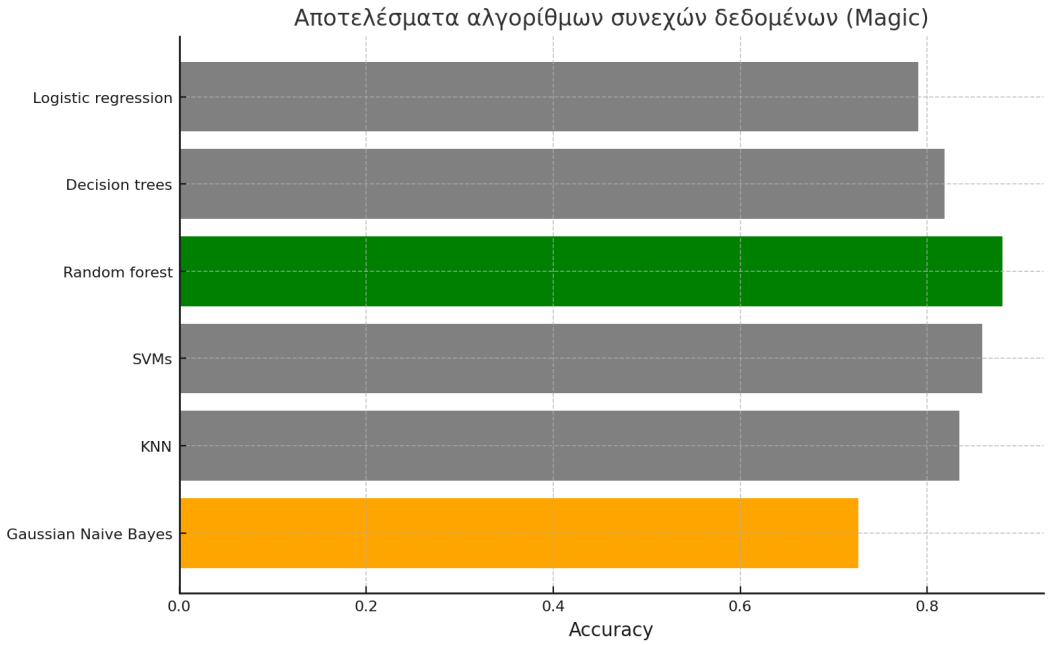
5.3.4 Αποτελέσματα Magic dataset

Algorithm	Accuracy
Logistic regression	0.7905
Decision trees	0.8183
Random forest	**0.8804**
SVMs	0.8587
KNN	0.8343
Gaussian Naive Bayes	**0.7267**

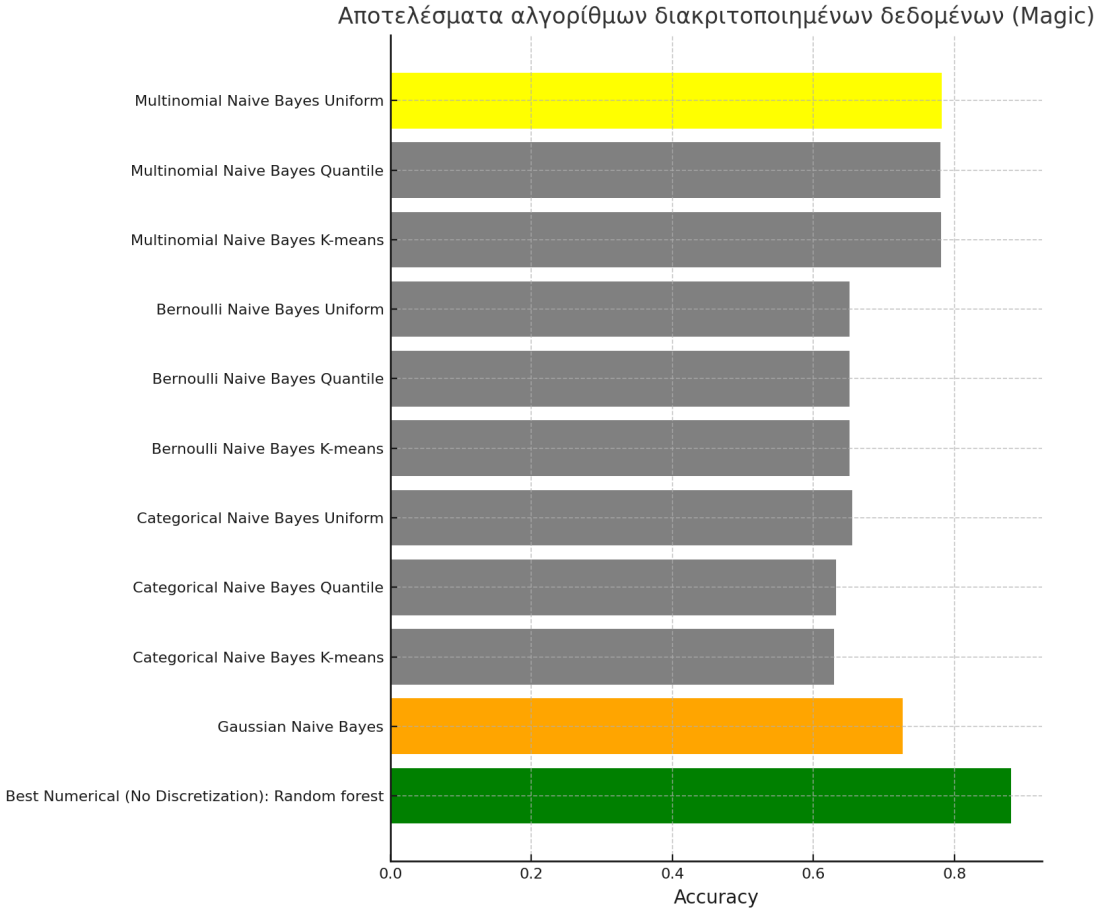
Πίνακας 5.7: Πίνακας ακριβείας αλγορίθμων συνεχών δεδομένων στο dataset Magic

Algorithm	Accuracy
Multinomial Naive Bayes Uniform	**0.7818**
Multinomial Naive Bayes Quantile	0.7802
Multinomial Naive Bayes K-means	0.781
Bernoulli Naive Bayes Uniform	0.6512
Bernoulli Naive Bayes Quantile	0.6513
Bernoulli Naive Bayes K-means	0.6515
Categorical Naive Bayes Uniform	0.655
Categorical Naive Bayes Quantile	0.6323
Categorical Naive Bayes K-means	0.6298
Gaussian Naive Bayes	**0.7267**
Best Numerical (No Discretization): Random forest	**0.8804**

Πίνακας 5.8: Πίνακας ακριβείας αλγορίθμων διακριτοποιημένων δεδομένων στο dataset Magic



Σχήμα 5.7: Διάγραμμα αρκιβείας αλγορίθμων συνεχών δεδομένων στο dataset Magic



Σχήμα 5.8: Διάγραμμα αρκιβείας αλγορίθμων διακριτοποιημένων δεδομένων στο dataset Magic

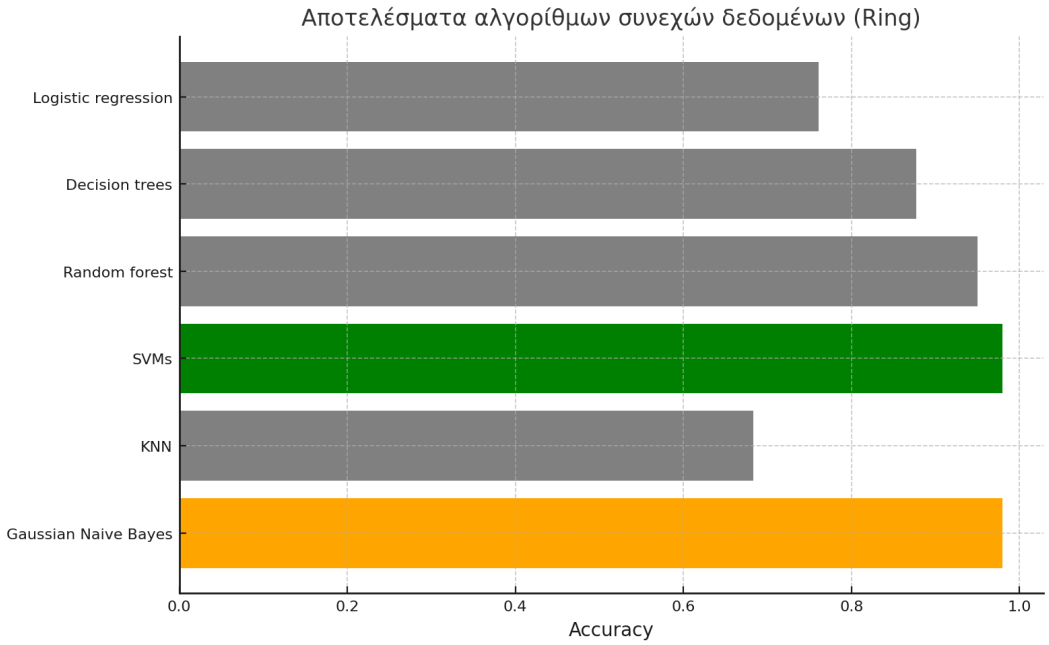
5.3.5 Αποτελέσματα Ring dataset

Algorithm	Accuracy
Logistic regression	0.7608
Decision trees	0.8773
Random forest	0.95
SVMs	**0.9797**
KNN	0.6831
Gaussian Naive Bayes	**0.9795**

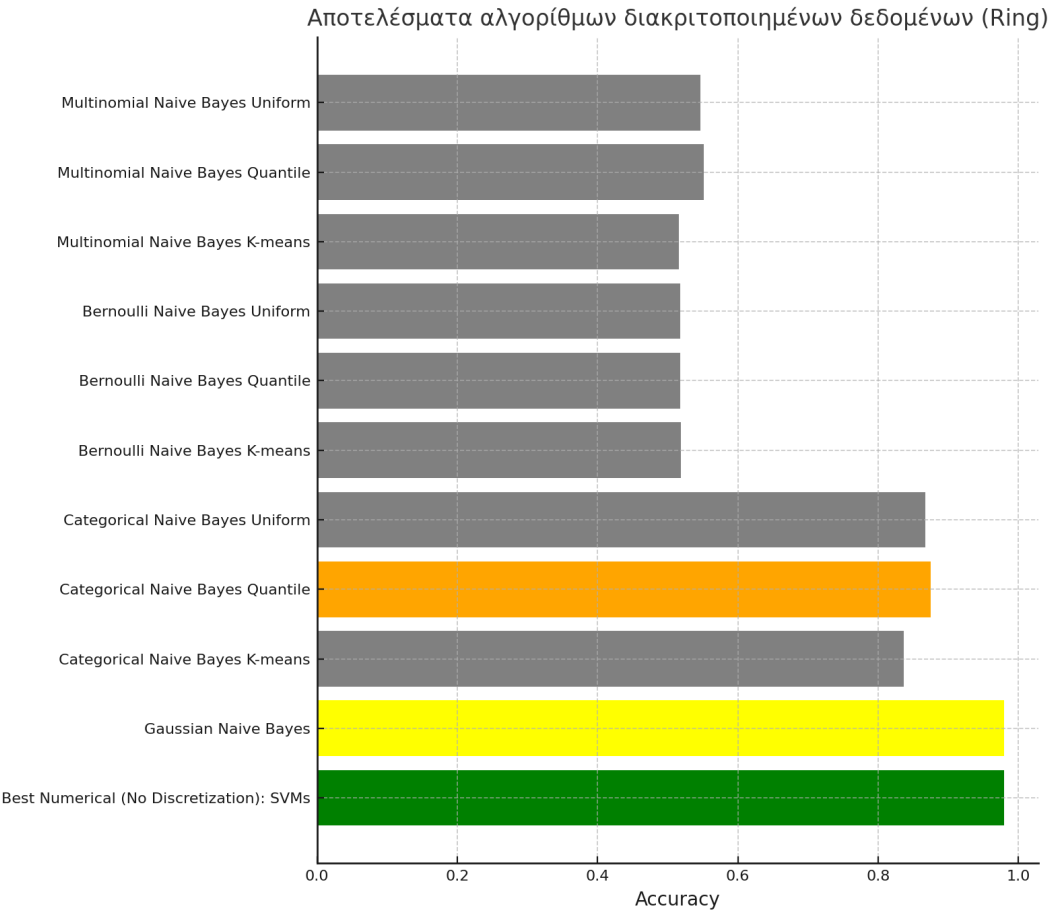
Πίνακας 5.9: Πίνακας ακριβείας αλγορίθμων συνεχών δεδομένων στο dataset Ring

Algorithm	Accuracy
Multinomial Naive Bayes Uniform	0.5468
Multinomial Naive Bayes Quantile	0.5518
Multinomial Naive Bayes K-means	0.5162
Bernoulli Naive Bayes Uniform	0.518
Bernoulli Naive Bayes Quantile	0.5181
Bernoulli Naive Bayes K-means	0.5184
Categorical Naive Bayes Uniform	0.8669
Categorical Naive Bayes Quantile	**0.8746**
Categorical Naive Bayes K-means	0.837
Gaussian Naive Bayes	**0.9795**
Best Numerical (No Discretization): SVMs	**0.9797**

Πίνακας 5.10: Πίνακας ακριβείας αλγορίθμων διακριτοποιημένων δεδομένων στο dataset Ring



Σχήμα 5.9: Διάγραμμα αρκιβείας αλγορίθμων συνεχών δεδομένων στο dataset Ring



Σχήμα 5.10: Διάγραμμα αρκιβείας αλγορίθμων διακριτοποιημένων δεδομένων στο dataset Ring

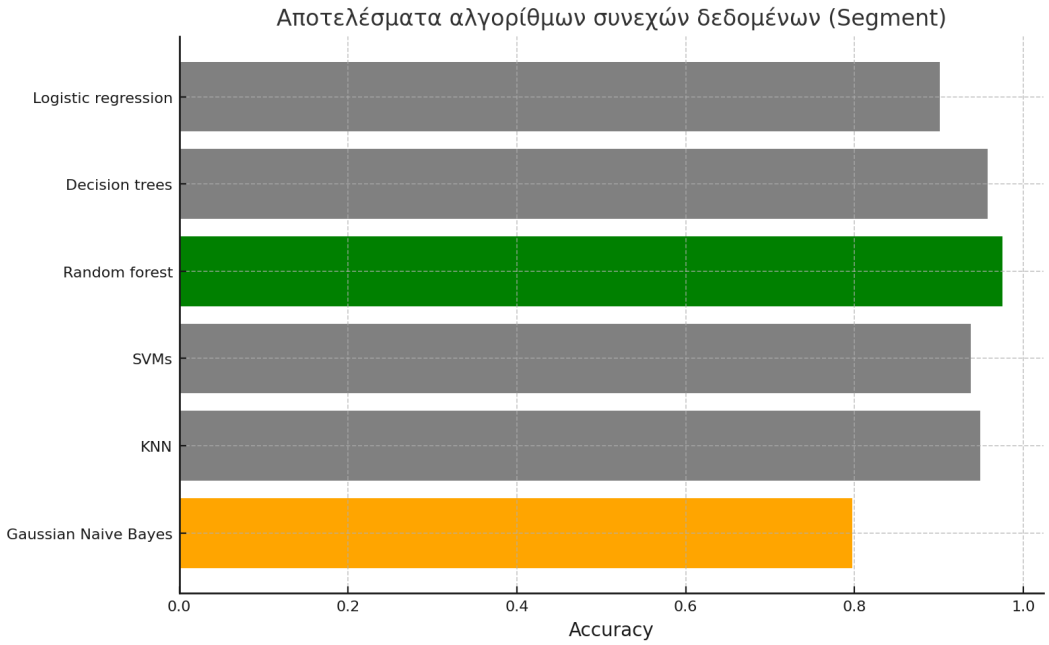
5.3.6 Αποτελέσματα Segment dataset

Algorithm	Accuracy
Logistic regression	0.9009
Decision trees	0.958
Random forest	**0.9753**
SVMs	0.9377
KNN	0.9494
Gaussian Naive Bayes	**0.7974**

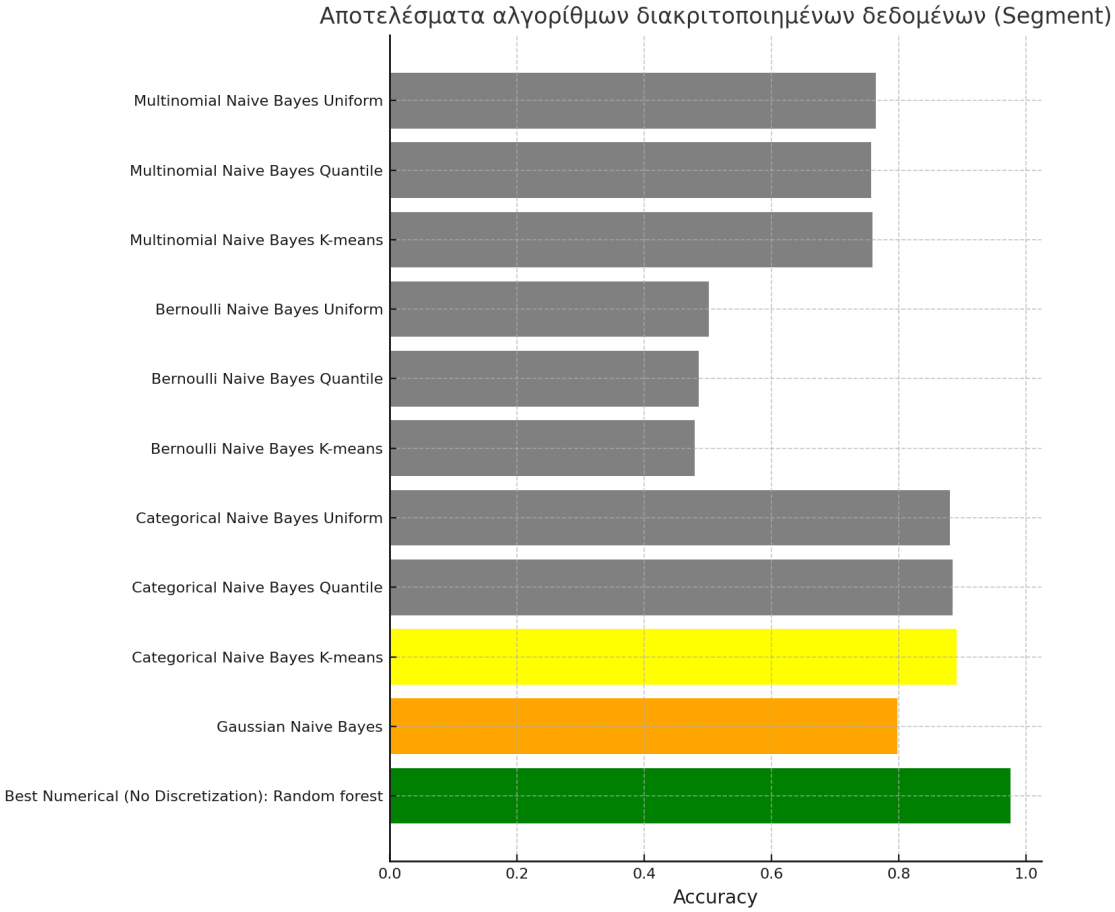
Πίνακας 5.11: Πίνακας ακριβείας αλγορίθμων συνεχών δεδομένων στο dataset Segment

Algorithm	Accuracy
Multinomial Naive Bayes Uniform	0.7636
Multinomial Naive Bayes Quantile	0.7567
Multinomial Naive Bayes K-means	0.7584
Bernoulli Naive Bayes Uniform	0.5017
Bernoulli Naive Bayes Quantile	0.4861
Bernoulli Naive Bayes K-means	0.4788
Categorical Naive Bayes Uniform	0.8797
Categorical Naive Bayes Quantile	0.884
Categorical Naive Bayes K-means	**0.8905**
Gaussian Naive Bayes	**0.7974**
Best Numerical (No Discretization): Random forest	**0.9753**

Πίνακας 5.12: Πίνακας ακριβείας αλγορίθμων διακριτοποιημένων δεδομένων στο dataset Segment



Σχήμα 5.11: Διάγραμμα αρκιβείας αλγορίθμων συνεχών δεδομένων στο dataset Segment



Σχήμα 5.12: Διάγραμμα αρκιβείας αλγορίθμων διακριτοποιημένων δεδομένων στο dataset Segment

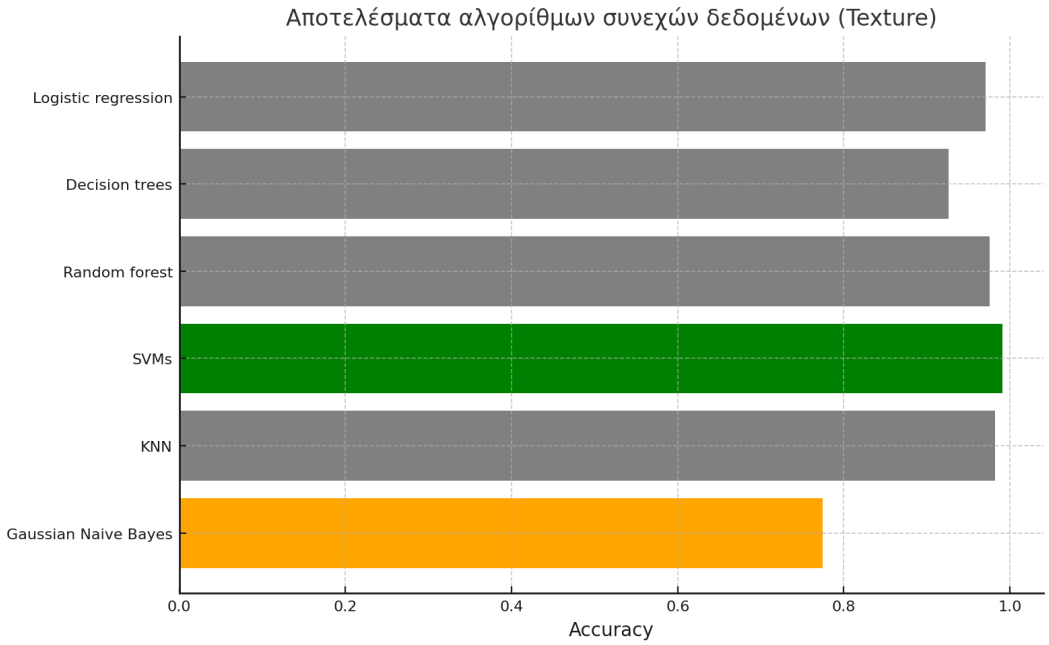
5.3.7 Αποτελέσματα Texture dataset

Algorithm	Accuracy
Logistic regression	0.9704
Decision trees	0.9264
Random forest	0.9755
SVMs	**0.9909**
KNN	0.9822
Gaussian Naive Bayes	**0.7744**

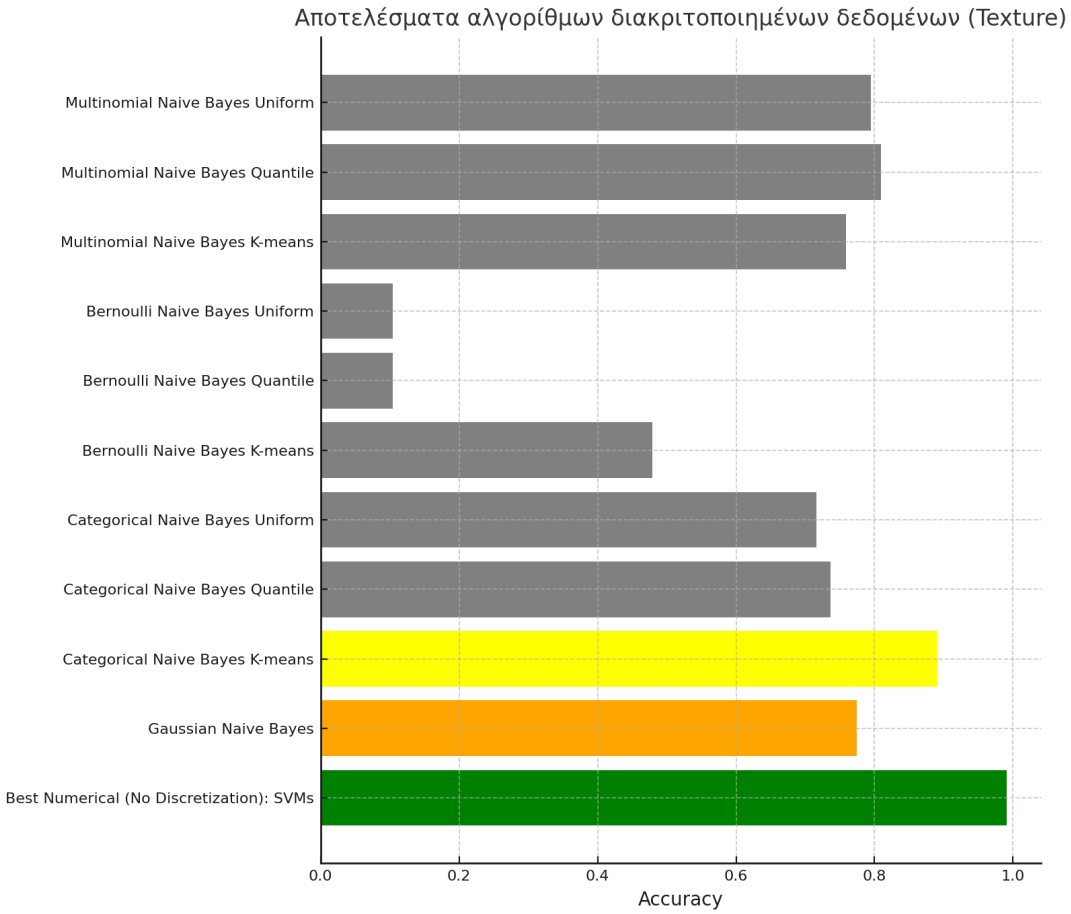
Πίνακας 5.13: Πίνακας ακριβείας αλγορίθμων συνεχών δεδομένων στο dataset Texture

Algorithm	Accuracy
Multinomial Naive Bayes Uniform	0.7947
Multinomial Naive Bayes Quantile	0.8089
Multinomial Naive Bayes K-means	0.7584
Bernoulli Naive Bayes Uniform	0.1045
Bernoulli Naive Bayes Quantile	0.1044
Bernoulli Naive Bayes K-means	0.4788
Categorical Naive Bayes Uniform	0.7156
Categorical Naive Bayes Quantile	0.7362
Categorical Naive Bayes K-means	**0.8905**
Gaussian Naive Bayes	**0.7744**
Best Numerical (No Discretization): SVMs	**0.9909**

Πίνακας 5.14: Πίνακας ακριβείας αλγορίθμων διακριτοποιημένων δεδομένων στο dataset Texture



Σχήμα 5.13: Διάγραμμα αρκιβείας αλγορίθμων συνεχών δεδομένων στο dataset Texture



Σχήμα 5.14: Διάγραμμα αρκιβείας αλγορίθμων διακριτοποιημένων δεδομένων στο dataset Texture

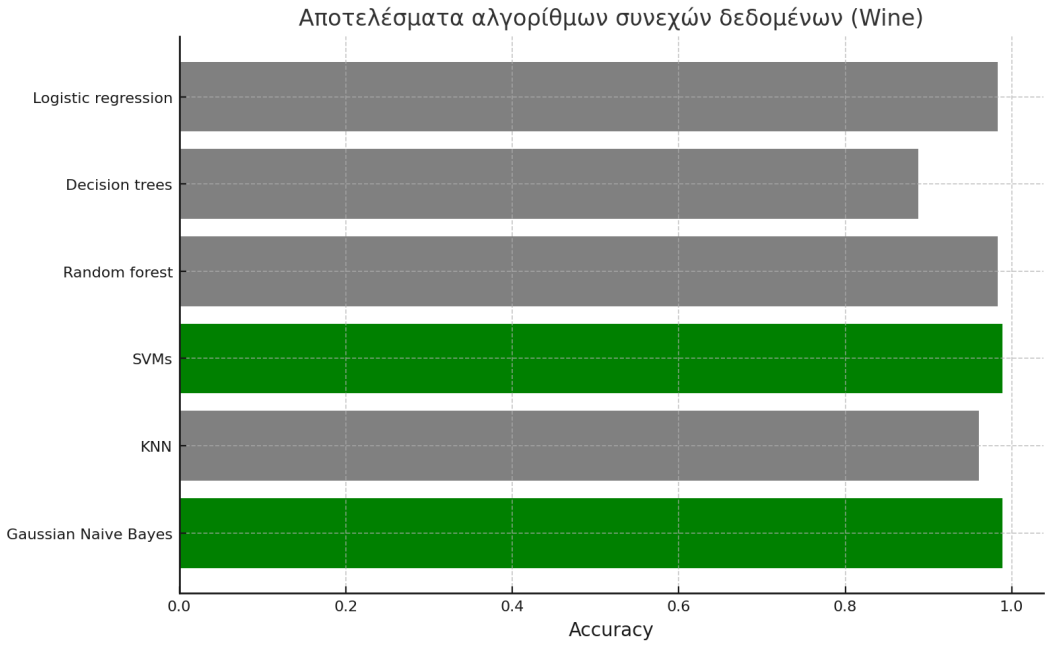
5.3.8 Αποτελέσματα Wine dataset

Algorithm	Accuracy
Logistic regression	0.9831
Decision trees	0.8876
Random forest	0.9831
SVMs	**0.9888**
KNN	0.9607
Gaussian Naive Bayes	**0.9888**

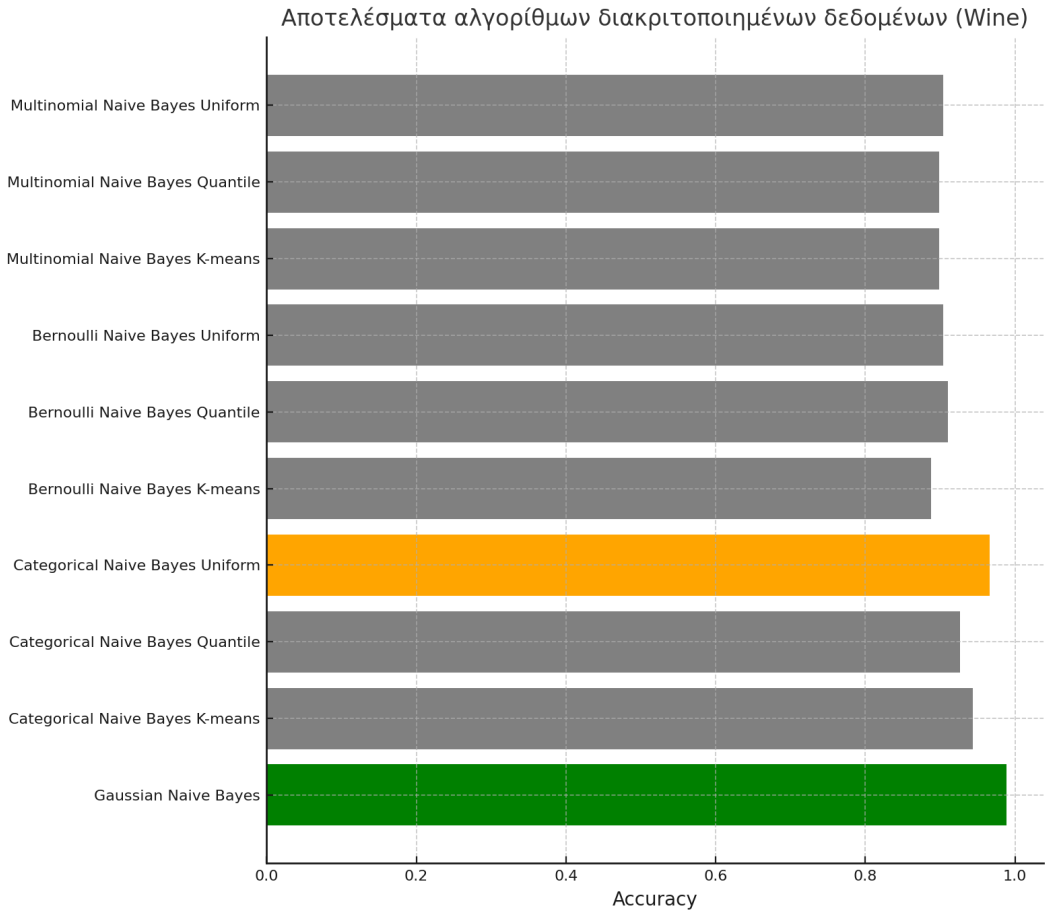
Πίνακας 5.15: Πίνακας ακριβείας αλγορίθμων συνεχών δεδομένων στο dataset Wine

Algorithm	Accuracy
Multinomial Naive Bayes Uniform	0.9045
Multinomial Naive Bayes Quantile	0.8989
Multinomial Naive Bayes K-means	0.8989
Bernoulli Naive Bayes Uniform	0.9045
Bernoulli Naive Bayes Quantile	0.9101
Bernoulli Naive Bayes K-means	0.8876
Categorical Naive Bayes Uniform	**0.9663**
Categorical Naive Bayes Quantile	0.927
Categorical Naive Bayes K-means	0.9438
Gaussian Naive Bayes	**0.9888**

Πίνακας 5.16: Πίνακας ακριβείας αλγορίθμων διακριτοποιημένων δεδομένων στο dataset Wine



Σχήμα 5.15: Διάγραμμα αρκιβείας αλγορίθμων συνεχών δεδομένων στο dataset Wine



Σχήμα 5.16: Διάγραμμα αρκιβείας αλγορίθμων διακριτοποιημένων δεδομένων στο dataset Wine

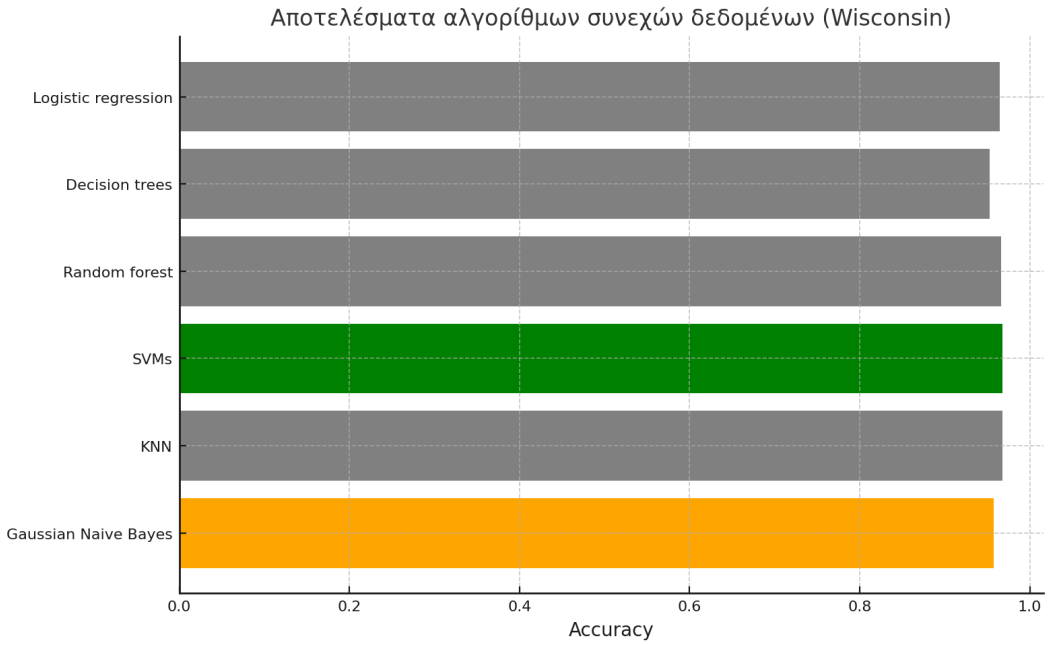
5.3.9 Αποτελέσματα Wisconsin dataset

Algorithm	Accuracy
Logistic regression	0.9649
Decision trees	0.9531
Random forest	0.9663
SVMs	**0.9678**
KNN	0.9678
Gaussian Naive Bayes	**0.9575**

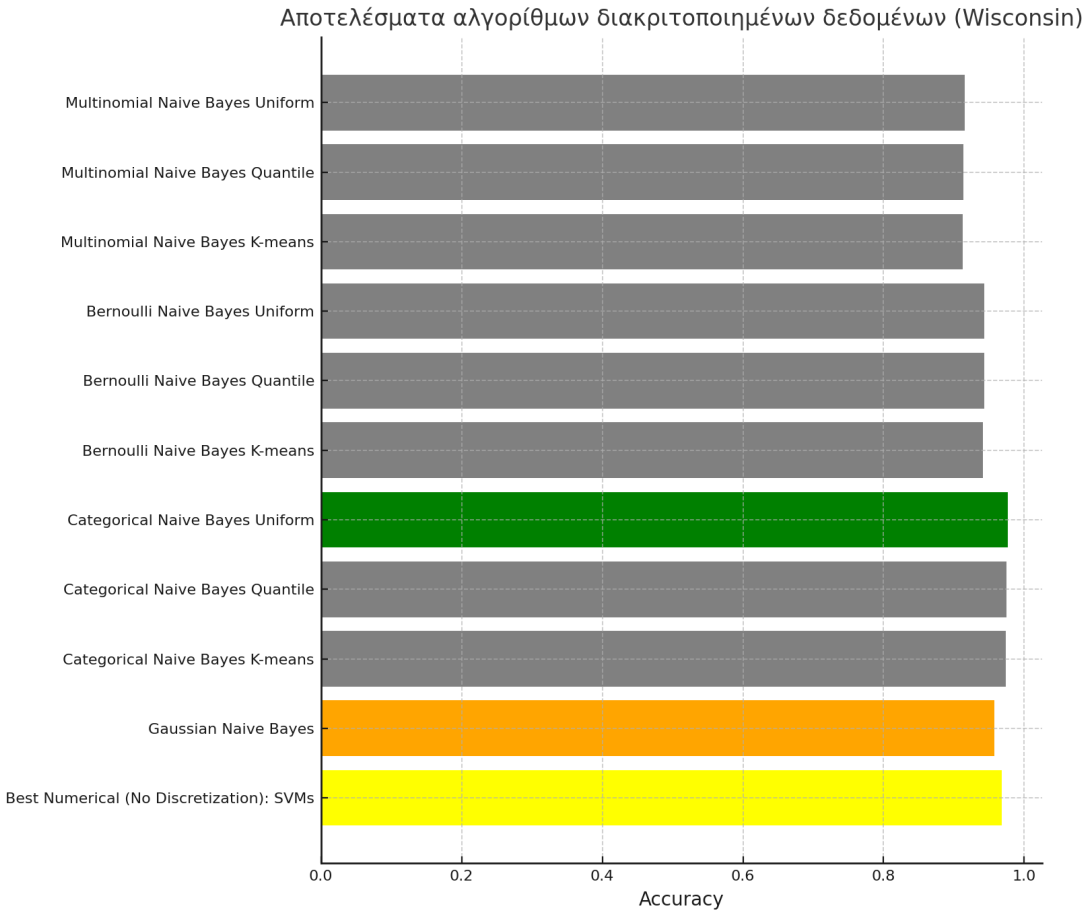
Πίνακας 5.17: Πίνακας ακριβείας αλγορίθμων συνεχών δεδομένων στο dataset Wisconsin

Algorithm	Accuracy
Multinomial Naive Bayes Uniform	0.9151
Multinomial Naive Bayes Quantile	0.9136
Multinomial Naive Bayes K-means	0.9122
Bernoulli Naive Bayes Uniform	0.9429
Bernoulli Naive Bayes Quantile	0.9429
Bernoulli Naive Bayes K-means	0.9414
Categorical Naive Bayes Uniform	**0.9766**
Categorical Naive Bayes Quantile	0.9751
Categorical Naive Bayes K-means	0.9736
Gaussian Naive Bayes	**0.9575**
Best Numerical (No Discretization): SVMs	**0.9678**

Πίνακας 5.18: Πίνακας ακριβείας αλγορίθμων διακριτοποιημένων δεδομένων στο dataset Wisconsin



Σχήμα 5.17: Διάγραμμα αρκιβείας αλγορίθμων συνεχών δεδομένων στο dataset Wisconsin



Σχήμα 5.18: Διάγραμμα αρκιβείας αλγορίθμων διακριτοποιημένων δεδομένων στο dataset Wisconsin

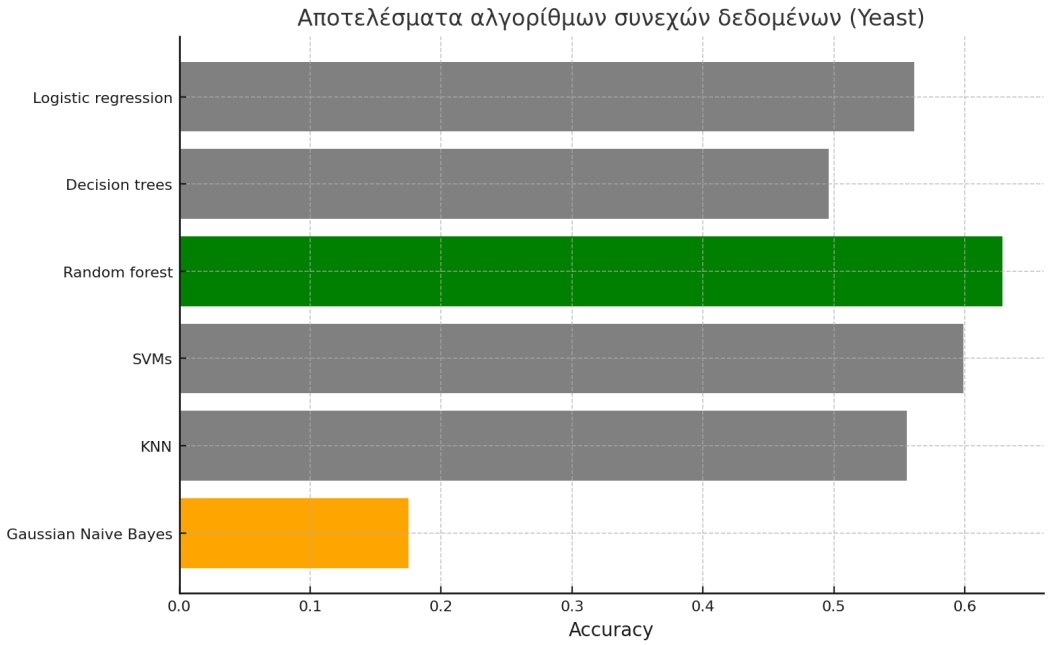
5.3.10 Αποτελέσματα Yeast dataset

Algorithm	Accuracy
Logistic regression	0.5613
Decision trees	0.496
Random forest	**0.6287**
SVMs	0.5991
KNN	0.5559
Gaussian Naive Bayes	**0.1752**

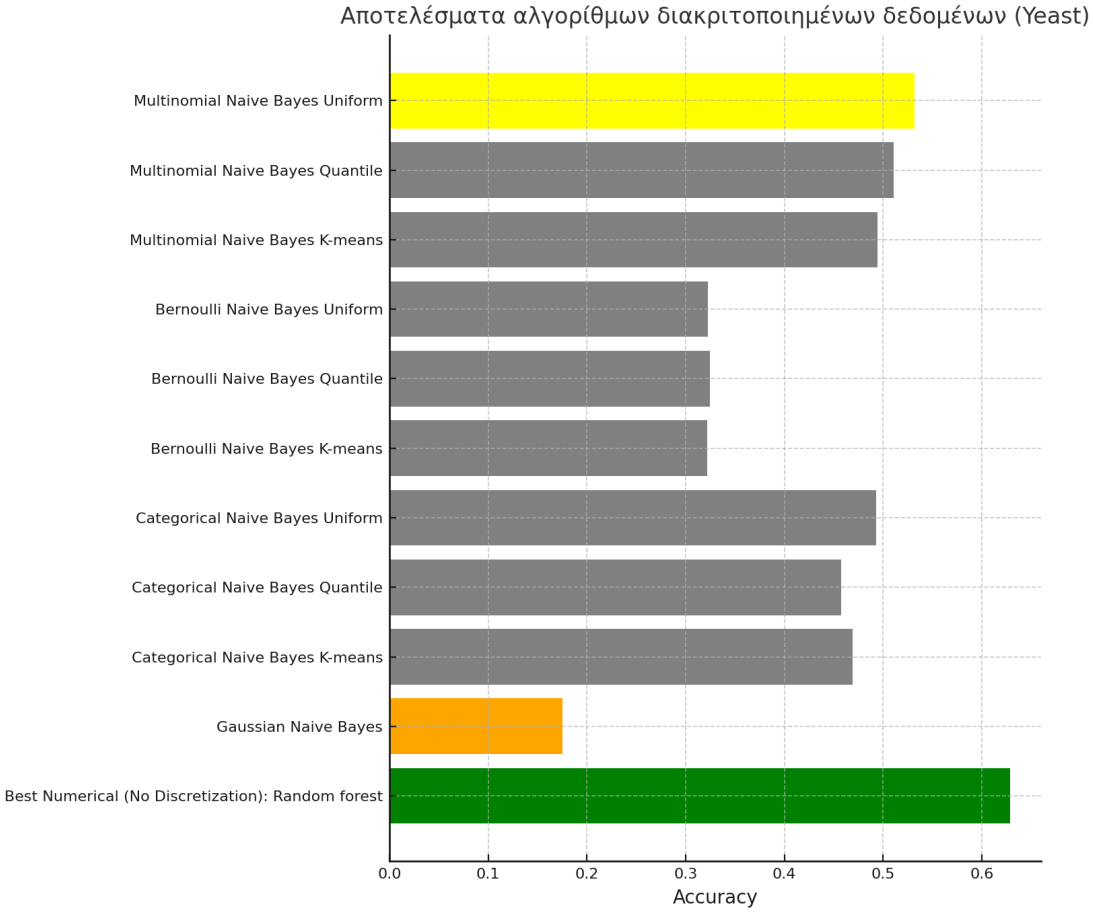
Πίνακας 5.19: Πίνακας ακριβείας αλγορίθμων συνεχών δεδομένων στο dataset Yeast

Algorithm	Accuracy
Multinomial Naive Bayes Uniform	**0.5317**
Multinomial Naive Bayes Quantile	0.5108
Multinomial Naive Bayes K-means	0.4946
Bernoulli Naive Bayes Uniform	0.3228
Bernoulli Naive Bayes Quantile	0.3248
Bernoulli Naive Bayes K-means	0.3221
Categorical Naive Bayes Uniform	0.4933
Categorical Naive Bayes Quantile	0.4575
Categorical Naive Bayes K-means	0.469
Gaussian Naive Bayes	**0.1752**
Best Numerical (No Discretization): Random forest	**0.6287**

Πίνακας 5.20: Πίνακας ακριβείας αλγορίθμων διακριτοποιημένων δεδομένων στο dataset Yeast



Σχήμα 5.19: Διάγραμμα αρκιβείας αλγορίθμων συνεχών δεδομένων στο dataset Yeast



Σχήμα 5.20: Διάγραμμα αρκιβείας αλγορίθμων διακριτοποιημένων δεδομένων στο dataset Yeast

5.3.11 Όλα τα αποτελέσματα

Παρακάτω περιλαμβάνονται συγκεντρωτικά τα αποτελέσματα accuracy, precision, recall, F1-score και support που προέκυψαν από την εκτέλεση των αλγορίθμων.

Algorithm	Accuracy	Precision	Recall	F1-score	Support
Logistic Regression	0.5855	0.56	0.52	0.44	345
Decision Trees	0.6290	0.62	0.62	0.62	345
Random Forest	0.7333	0.73	0.72	0.72	345
SVMs	0.6667	0.67	0.63	0.62	345
KNN	0.6261	0.61	0.61	0.61	345
Gaussian Naive Bayes	0.5275	0.55	0.55	0.53	345
Multinomial Naive Bayes Uniform	0.5971	0.58	0.58	0.58	345
Multinomial Naive Bayes Quantile	0.6058	0.59	0.59	0.59	345
Multinomial Naive Bayes K-means	0.6493	0.64	0.63	0.63	345
Bernoulli Naive Bayes Uniform	0.6116	0.60	0.57	0.56	345
Bernoulli Naive Bayes Quantile	0.6261	0.62	0.58	0.56	345
Bernoulli Naive Bayes K-means	0.6203	0.61	0.58	0.57	345
Categorical Naive Bayes Uniform	0.5768	0.56	0.56	0.56	345
Categorical Naive Bayes Quantile	0.5652	0.55	0.55	0.55	345
Categorical Naive Bayes K-means	0.5942	0.59	0.59	0.59	345

Πίνακας 5.21: Πίνακας αποτελεσμάτων αλγορίθμων συνεχών δεδομένων στο dataset Bupa

Algorithm	Accuracy	Precision	Recall	F1-score	Support
Logistic Regression	0.9267	0.93	0.93	0.93	150
Decision Trees	0.9667	0.97	0.97	0.97	150
Random Forest	0.9467	0.95	0.95	0.95	150
SVMs	0.9600	0.96	0.96	0.96	150
KNN	0.9667	0.97	0.97	0.97	150
Gaussian Naive Bayes	0.9467	0.95	0.95	0.95	150
Multinomial Naive Bayes Uniform	0.7133	0.71	0.71	0.71	150
Multinomial Naive Bayes Quantile	0.7067	0.71	0.71	0.70	150
Multinomial Naive Bayes K-means	0.7400	0.74	0.74	0.74	150
Bernoulli Naive Bayes Uniform	0.7133	0.75	0.71	0.68	150
Bernoulli Naive Bayes Quantile	0.7133	0.75	0.71	0.68	150
Bernoulli Naive Bayes K-means	0.6667	0.69	0.67	0.62	150
Categorical Naive Bayes Uniform	0.9200	0.92	0.92	0.92	150
Categorical Naive Bayes Quantile	0.9000	0.90	0.90	0.90	150
Categorical Naive Bayes K-means	0.8800	0.88	0.88	0.88	150

Πίνακας 5.22: Πίνακας αποτελεσμάτων αλγορίθμων συνεχών δεδομένων στο dataset Iris

Algorithm	Accuracy	Precision	Recall	F1-score	Support
Logistic Regression	0.7479	0.75	0.75	0.74	20000
Decision Trees	0.8740	0.87	0.87	0.87	20000
Random Forest	0.9627	0.96	0.96	0.96	20000
SVMs	0.9246	0.93	0.92	0.92	20000
KNN	0.9488	0.95	0.95	0.95	20000
Gaussian Naive Bayes	0.6417	0.65	0.64	0.64	20000
Multinomial Naive Bayes Uniform	0.5615	0.57	0.56	0.55	20000
Multinomial Naive Bayes Quantile	0.5583	0.57	0.56	0.55	20000
Multinomial Naive Bayes K-means	0.5575	0.56	0.56	0.55	20000
Bernoulli Naive Bayes Uniform	0.1126	0.44	0.11	0.09	20000
Bernoulli Naive Bayes Quantile	0.1117	0.46	0.11	0.09	20000
Bernoulli Naive Bayes K-means	0.1113	0.45	0.11	0.09	20000
Categorical Naive Bayes Uniform	0.6356	0.65	0.64	0.64	20000
Categorical Naive Bayes Quantile	0.6589	0.68	0.66	0.66	20000
Categorical Naive Bayes K-means	0.6561	0.68	0.66	0.66	20000

Πίνακας 5.23: Πίνακας αποτελεσμάτων αλγορίθμων συνεχών δεδομένων στο dataset Letter

Algorithm	Accuracy	Precision	Recall	F1-score	Support
Logistic Regression	0.7905	0.79	0.79	0.78	19020
Decision Trees	0.8183	0.82	0.82	0.82	19020
Random Forest	0.8804	0.88	0.88	0.88	19020
SVMs	0.8587	0.86	0.86	0.85	19020
KNN	0.8343	0.83	0.83	0.83	19020
Gaussian Naive Bayes	0.7267	0.72	0.73	0.70	19020
Multinomial Naive Bayes Uniform	0.7818	0.78	0.78	0.78	19020
Multinomial Naive Bayes Quantile	0.7802	0.78	0.78	0.78	19020
Multinomial Naive Bayes K-means	0.7810	0.78	0.78	0.78	19020
Bernoulli Naive Bayes Uniform	0.6512	0.68	0.65	0.52	19020
Bernoulli Naive Bayes Quantile	0.6513	0.70	0.65	0.52	19020
Bernoulli Naive Bayes K-means	0.6515	0.69	0.65	0.52	19020
Categorical Naive Bayes Uniform	0.6550	0.70	0.65	0.66	19020
Categorical Naive Bayes Quantile	0.6323	0.69	0.63	0.64	19020
Categorical Naive Bayes K-means	0.6298	0.72	0.63	0.63	19020

Πίνακας 5.24: Πίνακας αποτελεσμάτων αλγορίθμων συνεχών δεδομένων στο dataset Magic

Algorithm	Accuracy	Precision	Recall	F1-score	Support
Logistic Regression	0.7608	0.76	0.76	0.76	7400
Decision Trees	0.8773	0.88	0.88	0.88	7400
Random Forest	0.9500	0.95	0.95	0.95	7400
SVMs	0.9797	0.98	0.98	0.98	7400
KNN	0.6831	0.81	0.68	0.65	7400
Gaussian Naive Bayes	0.9795	0.98	0.98	0.98	7400
Multinomial Naive Bayes Uniform	0.5468	0.55	0.54	0.53	7400
Multinomial Naive Bayes Quantile	0.5518	0.56	0.55	0.53	7400
Multinomial Naive Bayes K-means	0.5162	0.52	0.52	0.51	7400
Bernoulli Naive Bayes Uniform	0.5180	0.75	0.52	0.37	7400
Bernoulli Naive Bayes Quantile	0.5181	0.75	0.52	0.37	7400
Bernoulli Naive Bayes K-means	0.5184	0.75	0.52	0.37	7400
Categorical Naive Bayes Uniform	0.8669	0.89	0.87	0.87	7400
Categorical Naive Bayes Quantile	0.8746	0.89	0.88	0.87	7400
Categorical Naive Bayes K-means	0.8370	0.87	0.84	0.83	7400

Πίνακας 5.25: Πίνακας αποτελεσμάτων αλγορίθμων συνεχών δεδομένων στο dataset Ring

Algorithm	Accuracy	Precision	Recall	F1-score	Support
Logistic Regression	0.9009	0.90	0.90	0.90	2310
Decision Trees	0.9580	0.96	0.96	0.96	2310
Random Forest	0.9753	0.98	0.98	0.98	2310
SVMs	0.9377	0.94	0.94	0.94	2310
KNN	0.9494	0.95	0.95	0.95	2310
Gaussian Naive Bayes	0.7974	0.81	0.80	0.78	2310
Multinomial Naive Bayes Uniform	0.7636	0.76	0.76	0.76	2310
Multinomial Naive Bayes Quantile	0.7567	0.75	0.76	0.75	2310
Multinomial Naive Bayes K-means	0.7584	0.76	0.76	0.75	2310
Bernoulli Naive Bayes Uniform	0.5017	0.51	0.50	0.45	2310
Bernoulli Naive Bayes Quantile	0.4861	0.50	0.49	0.44	2310
Bernoulli Naive Bayes K-means	0.4788	0.49	0.48	0.44	2310
Categorical Naive Bayes Uniform	0.8797	0.88	0.88	0.88	2310
Categorical Naive Bayes Quantile	0.8840	0.88	0.88	0.88	2310
Categorical Naive Bayes K-means	0.8905	0.89	0.89	0.89	2310

Πίνακας 5.26: Πίνακας αποτελεσμάτων αλγορίθμων συνεχών δεδομένων στο dataset Segment

Algorithm	Accuracy	Precision	Recall	F1-score	Support
Logistic Regression	0.9704	0.97	0.97	0.97	5500
Decision Trees	0.9264	0.93	0.93	0.93	5500
Random Forest	0.9755	0.98	0.98	0.98	5500
SVMs	0.9909	0.99	0.99	0.99	5500
KNN	0.9822	0.98	0.98	0.98	5500
Gaussian Naive Bayes	0.7744	0.78	0.77	0.77	5500
Multinomial Naive Bayes Uniform	0.7947	0.81	0.79	0.79	5500
Multinomial Naive Bayes Quantile	0.8089	0.82	0.81	0.81	5500
Multinomial Naive Bayes K-means	0.7584	0.76	0.76	0.75	5500
Bernoulli Naive Bayes Uniform	0.1045	0.84	0.10	0.04	5500
Bernoulli Naive Bayes Quantile	0.1044	0.84	0.10	0.04	5500
Bernoulli Naive Bayes K-means	0.4788	0.49	0.48	0.44	5500
Categorical Naive Bayes Uniform	0.7156	0.72	0.72	0.71	5500
Categorical Naive Bayes Quantile	0.7362	0.74	0.74	0.73	5500
Categorical Naive Bayes K-means	0.8905	0.89	0.89	0.89	5500

Πίνακας 5.27: Πίνακας αποτελεσμάτων αλγορίθμων συνεχών δεδομένων στο dataset Texture

Algorithm	Accuracy	Precision	Recall	F1-score	Support
Logistic Regression	0.9831	0.98	0.99	0.98	178
Decision Trees	0.8876	0.89	0.89	0.89	178
Random Forest	0.9831	0.98	0.99	0.98	178
SVMs	0.9888	0.99	0.99	0.99	178
KNN	0.9607	0.96	0.96	0.96	178
Gaussian Naive Bayes	0.9888	0.99	0.99	0.99	178
Multinomial Naive Bayes Uniform	0.9045	0.91	0.92	0.91	178
Multinomial Naive Bayes Quantile	0.8989	0.90	0.91	0.90	178
Multinomial Naive Bayes K-means	0.8989	0.90	0.91	0.90	178
Bernoulli Naive Bayes Uniform	0.9045	0.92	0.91	0.91	178
Bernoulli Naive Bayes Quantile	0.9101	0.92	0.91	0.91	178
Bernoulli Naive Bayes K-means	0.8876	0.89	0.89	0.89	178
Categorical Naive Bayes Uniform	0.9663	0.97	0.97	0.97	178
Categorical Naive Bayes Quantile	0.9270	0.93	0.93	0.93	178
Categorical Naive Bayes K-means	0.9438	0.95	0.94	0.95	178

Πίνακας 5.28: Πίνακας αποτελεσμάτων αλγορίθμων συνεχών δεδομένων στο dataset Wine

Algorithm	Accuracy	Precision	Recall	F1-score	Support
Logistic Regression	0.9649	0.96	0.96	0.96	683
Decision Trees	0.9531	0.95	0.95	0.95	683
Random Forest	0.9663	0.96	0.96	0.96	683
SVMs	0.9678	0.97	0.97	0.97	683
KNN	0.9678	0.97	0.97	0.97	683
Gaussian Naive Bayes	0.9575	0.96	0.96	0.95	683
Multinomial Naive Bayes Uniform	0.9151	0.90	0.93	0.91	683
Multinomial Naive Bayes Quantile	0.9136	0.90	0.92	0.91	683
Multinomial Naive Bayes K-means	0.9122	0.90	0.92	0.91	683
Bernoulli Naive Bayes Uniform	0.9429	0.93	0.95	0.94	683
Bernoulli Naive Bayes Quantile	0.9429	0.93	0.95	0.94	683
Bernoulli Naive Bayes K-means	0.9414	0.93	0.95	0.94	683
Categorical Naive Bayes Uniform	0.9766	0.97	0.98	0.97	683
Categorical Naive Bayes Quantile	0.9751	0.97	0.98	0.97	683
Categorical Naive Bayes K-means	0.9736	0.97	0.97	0.97	683

Πίνακας 5.29: Πίνακας αποτελεσμάτων αλγορίθμων συνεχών δεδομένων στο dataset Wisconsin

Algorithm	Accuracy	Precision	Recall	F1-score	Support
Logistic Regression	0.5613	0.75	0.36	0.37	1484
Decision Trees	0.4960	0.41	0.39	0.39	1484
Random Forest	0.6287	0.70	0.57	0.58	1484
SVMs	0.5991	0.70	0.56	0.57	1484
KNN	0.5559	0.67	0.56	0.56	1484
Gaussian Naive Bayes	0.1752	0.37	0.40	0.30	1484
Multinomial Naive Bayes Uniform	0.5317	0.46	0.51	0.45	1484
Multinomial Naive Bayes Quantile	0.5108	0.49	0.49	0.45	1484
Multinomial Naive Bayes K-means	0.4946	0.43	0.49	0.43	1484
Bernoulli Naive Bayes Uniform	0.3228	0.64	0.16	0.12	1484
Bernoulli Naive Bayes Quantile	0.3248	0.66	0.16	0.12	1484
Bernoulli Naive Bayes K-means	0.3221	0.61	0.16	0.12	1484
Categorical Naive Bayes Uniform	0.4933	0.52	0.30	0.31	1484
Categorical Naive Bayes Quantile	0.4575	0.40	0.26	0.28	1484
Categorical Naive Bayes K-means	0.4690	0.41	0.28	0.28	1484

Πίνακας 5.30: Πίνακας αποτελεσμάτων αλγορίθμων συνεχών δεδομένων στο dataset Yeast

5.4 Συζήτηση

Τα πειραματικά αποτελέσματα που παρουσιάστηκαν προηγουμένως έδειξαν σημαντικές διαφορές στην απόδοση των αλγορίθμων, ανάλογα με το είδος και την κατανομή των δεδομένων. Σε σύνολα δεδομένων όπως το Iris και το Wine, ο Gaussian Naive Bayes πέτυχε υψηλή ακρίβεια, επιδεικνύοντας την ικανότητά του να διαχειρίζεται δεδομένα με κανονική κατανομή. Αντίθετα, η διακριτοποίηση σε αυτά τα datasets δεν οδήγησε σε βελτίωση της απόδοσης. Ωστόσο, σε άλλα δεδομένα, όπως το Yeast, το Wisconsin και το Bupa, η διακριτοποίηση απέδωσε καλύτερα αποτελέσματα, προσφέροντας τη βέλτιστη ακρίβεια και βελτιώνοντας τη δυνατότητα κατηγοριοποίησης.

Ο Gaussian Naive Bayes λειτούργησε καλά όταν τα δεδομένα είχαν κανονική κατανομή. Αντίθετα, όταν τα δεδομένα είχαν ασυνήθιστες τιμές ή δεν ακολουθούσαν κανονική κατανομή, η διακριτοποίηση βοήθησε στην καλύτερη ταξινόμηση, κάνοντας τα δεδομένα πιο εύκολα στη διαχείριση. Η επιλογή της μεθόδου διακριτοποίησης είναι σημαντική, καθώς παρατηρήθηκαν διαφορές στις επιδόσεις μεταξύ των μεθόδων. Η uniform διακριτοποίηση φάνηκε να αποδίδει καλύτερα στις περισσότερες περιπτώσεις, ενώ η quantile είχε μικρότερη ακρίβεια σε ορισμένα datasets. Επιπλέον, η k-means διακριτοποίηση προσέφερε μια πιο προσαρμοστική προσέγγιση, επιτυγχάνοντας καλά αποτελέσματα σε ορισμένα σύνολα δεδομένων με πολύπλοκη δομή. Για παράδειγμα, στο Segment dataset, η k-means διακριτοποίηση πέτυχε καλύτερη ομαδοποίηση των δεδομένων, καταφέρνοντας να αποτυπώσει τις εσωτερικές σχέσεις και να βελτιώσει την ακρίβεια της ταξινόμησης.

Η ανάλυση της απόδοσης έδειξε ότι δεν υπάρχει μία τέλεια λύση που να ταιριάζει σε όλα τα προβλήματα. Η διακριτοποίηση είναι χρήσιμη σε κάποιες περιπτώσεις, αλλά πρέπει να επιλεγούν σωστά τα όρια των κατηγοριών, ώστε να μην χαθούν σημαντικές πληροφορίες. Επιπλέον, διαπιστώθηκε ότι η ποιότητα των δεδομένων επηρεάζει τα αποτελέσματα. Για παράδειγμα, στο Wisconsin dataset, η ανισορροπία στις κατηγορίες επηρέασε την απόδοση, αλλά η διακριτοποίηση βοήθησε στη βελτίωση της ταξινόμησης των λιγότερο αντιπροσωπευτικών κλάσεων. Αντίθετα, στο Bupa dataset, η παρουσία θορύβου προκάλεσε δυσκολίες στον Gaussian Naive Bayes, αλλά η διακριτοποίηση μείωσε τον αντίκτυπο αυτόν, βελτιώνοντας τα αποτελέσματα.

Συμπερασματικά, η επιλογή της κατάλληλης μεθόδου εξαρτάται από τα δεδομένα και τις ανάγκες της εφαρμογής. Ο Gaussian Naive Bayes είναι χρήσιμος όταν τα δεδομένα είναι ομαλά και η ταχύτητα είναι κρίσιμη. Από την άλλη, η διακριτοποίηση μπορεί να είναι πιο αποτελεσματική όταν τα δεδομένα είναι πολύπλοκα ή ποικίλα. Είναι σημαντικό να γίνει ανάλυση των δεδομένων πριν επιλεγεί κάποιος αλγόριθμος και να δοκιμαστούν διαφορετικές μέθοδοι για την καλύτερη δυνατή απόδοση, καθώς άλλες τεχνικές μπορεί να δώσουν καλύτερα αποτελέσματα σε συγκεκριμένες περιπτώσεις.

Κεφάλαιο 6ο: Συμπεράσματα

Σε αυτό το κεφάλαιο συγκεντρώνουμε τα βασικά συμπεράσματα της μελέτης, αναδεικνύοντας τα σημαντικότερα ευρήματα και προτείνοντας ιδέες για μελλοντική έρευνα. Μέσα από την ανάλυση που πραγματοποιήθηκε, προέκυψαν πολύτιμες πληροφορίες σχετικά με την απόδοση των αλγορίθμων και τον αντίκτυπο της διακριτοποίησης στη βελτίωση της ακρίβειας.

Γενικά, τα αποτελέσματα έδειξαν ότι ο Gaussian Naive Bayes αποδίδει ικανοποιητικά όταν τα δεδομένα έχουν κανονική κατανομή, όπως στα σύνολα δεδομένων Iris και Wine. Αντίθετα, η διακριτοποίηση αποδείχθηκε πιο αποτελεσματική σε πιο πολύπλοκα δεδομένα, όπως τα Wisconsin και Bupa, όπου η ποιότητα των δεδομένων και η παρουσία ανισορροπιών επηρέασαν την απόδοση. Οι διαφορετικές μέθοδοι διακριτοποίησης έπαιξαν επίσης σημαντικό ρόλο στην απόδοση του αλγορίθμου, με την uniform διακριτοποίηση να αποδεικνύεται πιο αξιόπιστη στις περισσότερες περιπτώσεις, ενώ η k-means διακριτοποίηση έδειξε καλή προσαρμογή σε datasets με πολύπλοκη δομή, όπως το Segment.

Η παρούσα μελέτη κατέδειξε ότι οι μέθοδοι διακριτοποίησης μπορούν να χρησιμοποιηθούν αποτελεσματικά για τη βελτίωση των επιδόσεων του Naive Bayes. Παρ' όλα αυτά, η επιλογή της κατάλληλης μεθόδου διακριτοποίησης θα πρέπει να γίνεται με βάση τη φύση των δεδομένων, ώστε να αποφεύγεται η απώλεια σημαντικών πληροφοριών και η υποβάθμιση της ακρίβειας. Επιπλέον, τα αποτελέσματα έδειξαν ότι η επιλογή του κατάλληλου αλγορίθμου εξαρτάται σε μεγάλο βαθμό από τις απαιτήσεις της ταξινόμησης, με τον Naive Bayes να αποτελεί μια καλή επιλογή όταν η ταχύτητα και η απλότητα είναι πρωταρχικής σημασίας.

Για μελλοντική έρευνα, προτείνεται η διερεύνηση προηγμένων τεχνικών διακριτοποίησης που να προσαρμόζονται δυναμικά στα δεδομένα με χρήση machine learning. Επίσης, μπορεί να εξεταστεί η συνδυαστική χρήση του Naive Bayes με άλλους αλγορίθμους ταξινόμησης για την επίτευξη καλύτερης γενίκευσης. Επιπλέον, η προσθήκη περισσότερων αλγορίθμων μηχανικής μάθησης, όπως τα Νευρωνικά Δίκτυα και τα Τυχαία Δάση, θα μπορούσε να δώσει καλύτερη εικόνα για την απόδοση της διακριτοποίησης σε διαφορετικά είδη δεδομένων. Για παράδειγμα, η σύγκριση με Νευρωνικά Δίκτυα θα μπορούσε να αναδείξει τα πλεονεκτήματα και τα μειονεκτήματα των στατιστικών και των βαθιών μαθησιακών προσεγγίσεων. Τέλος, μία ενδιαφέρουσα κατεύθυνση είναι η αξιολόγηση των μεθόδων αυτών σε πραγματικά σενάρια, προκειμένου να διερευνηθεί η πρακτική εφαρμογή τους σε διαφορετικούς τομείς.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] D. Bhowmik and A. Chattopadhyay, "Machine learning techniques in medical diagnosis: A systematic review," *Journal of Medical Systems*, 2020.

[2] C. C. Aggarwal and C. Zhai, *Mining Text Data: Applications and Techniques*. Springer Science & Business Media, 2015.

[3] D. W. Hosmer, S. Lemeshow, and R. X. Sturdivant, *Applied Logistic Regression*. Wiley Series in Probability and Statistics, 2013.

[4] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, 1995.

[5] D. Heckerman, "A tutorial on learning with bayesian networks," tech. rep., Microsoft Research Technical Report, 1995.

[6] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, 1992.

[7] H. Zhang, "The optimality of naive bayes," in *AAAI Conference on Artificial Intelligence*, 2004.

[8] T. H. R. T. Gareth James, Daniela Witten, *Introduction to Statistical Learning*. Springer, 2013.

[9] R. O. Duda, P. E. Hart, and D. G. Stork, "Pattern classification," *Wiley*, 2000.

[10] J. D. M. Rennie, L. Shih, J. Teevan, and D. R. Karger, "Tackling the poor assumptions of naive bayes text classifiers," *Proceedings of the International Conference on Machine Learning*, 2003.

[11] H. Zhang, "The optimality of naive bayes," *AAAI*, 2004.

[12] C. M. Bishop, "Pattern recognition and machine learning," *Springer*, 2006.

[13] P. Domingos and M. Pazzani, "On the optimality of the simple bayesian classifier under zero-one loss," *Machine Learning*, 1997.

[14] I. Wickramasinghe and H. Kalutarage, "Naive bayes: applications, variations and vulnerabilities: a review of literature with code snippets for implementation," *Soft Computing*, vol. 25, no. 3, pp. 2277–2293, 2021.

[15] S. Zhang, X. Zhu, and X. Zhang, "A semi-supervised adaptive discriminative discretization method improving discrimination power of regularized naive bayes," *arXiv preprint*, 2021.

[16] J. C. Figueira, R. M. de Sousa, and A. T. Freitas, "A max-relevance-min-divergence criterion for data discretization with applications on naive bayes," *arXiv preprint*, 2022.

[17] J. Dougherty, R. Kohavi, and M. Sahami, "Supervised and unsupervised discretization of continuous features," in *Proceedings of the Twelfth International Conference on Machine Learning*, pp. 194–202, 1995.

[18] S. Kotsiantis and D. Kanellopoulos, "Discretization techniques: A recent survey," *GESTS International Transactions on Computer Science and Engineering*, vol. 32, no. 1, pp. 47–58, 2006.

[19] U. M. Fayyad and K. B. Irani, "Multi-interval discretization of continuous-valued attributes for classification learning," in *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, pp. 1022–1027, 1993.

[20] C. Chang and C. Lin, "Quantile-based discretization for machine learning applications," *Journal of Data Science*, vol. 11, pp. 111–123, 2013.

[21] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, "An efficient k-means clustering algorithm: Analysis and implementation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 881–892, 2002.

[22] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[23] H. Zhang, "The optimality of naive bayes," *AA*, vol. 1, no. 2, pp. 3–3, 2004.

[24] K. P. Murphy, *Probabilistic Machine Learning: An Introduction*. MIT Press, 2021. Accessed: 22 December 2024.

[25] Scikit-learn Development Team, "Bernoulli naive bayes documentation," 2024. Accessed: 22 December 2024.

[26] A. McCallum and K. Nigam, "A comparison of event models for naive bayes text classification," in *AAAI-98 workshop on learning for text categorization*, pp. 41–48, 1998.

[27] Scikit-learn Development Team, "Naive bayes documentation - scikit-learn," 2024. Accessed: 22 December 2024.

[28] I. Rish, "An empirical study of the naive bayes classifier," *IJCAI 2001 workshop on empirical methods in artificial intelligence*, vol. 3, no. 22, pp. 41–46, 2001.

[29] J. R. Quinlan, "Induction of decision trees," *Machine learning*, vol. 1, no. 1, pp. 81–106, 1986.

[30] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and regression trees*. Wadsworth International Group, 1984.

[31] GeeksforGeeks, "Decision tree algorithms," 2023. Accessed: 2024-12-21.

[32] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[33] A. Liaw and M. Wiener, "Classification and regression by randomforest," *R news*, vol. 2, no. 3, pp. 18–22, 2002.

[34] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE transactions on information theory*, vol. 13, no. 1, pp. 21–27, 1967.

[35] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[36] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.

[37] D. W. Hosmer, S. Lemeshow, and R. X. Sturdivant, *Applied logistic regression*. John Wiley & Sons, 2013.

[38] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006.

[39] B. Schölkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

[40] T. M. Cover and P. E. Hart, "A study of the k-nearest neighbor algorithm," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[41] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[42] H. Chen, M. Zhang, Z. Yuan, and B. Xu, "A review of data preprocessing in machine learning for big data handling," *Journal of Big Data*, vol. 8, pp. 1–26, 2021.

[43] Scikit-learn Development Team, "Discretization strategies example," 2024. Accessed: 22 December 2024.

[44] N. Verma, "Understanding and implementing gaussian naive bayes classification with python," 2023. Accessed: 22 December 2024.