ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

UNIVERSITY OF PIRAEUS

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΕΠΙΚΟΙΝΩΝΙΩΝ

ΤΜΗΜΑ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ηλίας Κατσαφάδος

"ΜΙΑ ΕΜΠΕΙΡΙΚΗ ΣΥΓΚΡΙΣΗ, ΜΕΛΕΤΗ ΚΑΙ ΕΝΙΣΧΥΣΗ ΑΠΟΔΟΣΕΩΝ ΣΕ ΑΛΓΟΡΙΘΜΟΥΣ ΕΠΙΒΛΕΠΟΜΕΝΗΣ ΜΗΧΑΝΙΚΗΣ ΜΑΘΗΣΗΣ ΜΕ ΤΗ ΒΟΗΘΕΙΑ ΤΗΣ R"

Επιβλέπων: Δρ. Μιχαήλ Φιλιππάκης, Αναπληρωτής Καθηγητής

Πειραιάς,

01 Φεβρουαρίου 2021

Δήλωση πρωτότυπης εργασίας

Η παρούσα διπλωματική εργασία είναι αποτέλεσμα προσωπικής μελέτης μου. Αναφορές σε βιβλιογραφικές πηγές μέσα στο κείμενο διευκρινίζουν ποιες πληροφορίες, στοιχεία και γνώσεις αντλήθηκαν από άλλες εργασίες ή βιβλία.

Ονοματεπώνυμο : **Ηλίας Κατσαφάδος**

Ευχαριστίες

Η παρούσα διπλωματική εργασία δεν θα μπορούσε να υλοποιηθεί δίχως τη στήριξη του καθηγητή μου Μιχαήλ Φιλιππάκη, ο οποίος με καθοδήγησε σε αυτό το έργο. Επιπροσθέτως, θα ήθελα να ευχαριστήσω την κοπέλα μου Δέσποινα για τις συμβουλές και την βοήθεια που μου παρείχε.

Περίληψη - Λέξεις κλειδιά

Η παρακάτω διπλωματική εργασία αποσκοπεί στο να περιγράψει τις έννοιες και τις μεθόδους της μηχανικής μάθησης με έμφαση στη μάθηση με επίβλεψη. Αρχικά, γίνεται μια βιβλιογραφική ανασκόπηση των δύο κατηγοριών της μάθησης με επίβλεψη (παλινδρόμηση , κατηγοριοποίηση) και των αλγορίθμων που εντάσσονται σε αυτές τις κατηγορίες και στη συνέχεια με τη βοήθεια της γλώσσας προγραμματισμού R, εφαρμόζονται οι αλγόριθμοι σε διαφορετικά σύνολα δεδομένων. Επιπροσθέτως, γίνεται σύγκριση της αποδοτικότητας των αλγορίθμων και μελετώνται τα πλεονεκτήματα και τα μειονεκτήματα τους . Τέλος, αναλύονται τα συμπεράσματα τα οποία παρήχθησαν μέσω της πειραματικής μελέτης,

Λέξεις κλειδιά: R, σύνολο δεδομένων, σύνολο εκπαίδευσης, σύνολο ελέγχου, μηχανική μάθηση, μάθηση με επίβλεψη, αλγόριθμος, μέθοδος, απόδοση, εντολή, παλινδρόμηση, κατηγοριοποίηση, χρόνος εκπαίδευσης, χρόνος ταξινόμησης, μετρική, ακρίβεια, πρόβλεψη

Abstract - Keywords

The following thesis aims to describe the concepts and methods of machine learning with emphasis given on supervised learning. Initially, a literature review of the two categories of supervised learning is conducted (regression, classification) including the algorithms of those categories and then with the aid of the programming language R, those algorithms are applied to different datasets. In addition, the efficiency of the algorithms is being compared and their advantages and disadvantages are being studied. Finally, the conclusions that have been generated through the research are analyzed.

Keywords: R, dataset, training set, test set, machine learning, supervised learning, algorithm, method, performance, command, regression, classification, training time, testing time, metric, accuracy, prediction

Πίνακας Περιεχομένων

Πίνακας Εικόνων

Κεφάλαιο 1^{ο} : Εισαγωγή - Μεθοδολογία έρευνας

Κίνητρο για αυτή την εργασία αποτέλεσαν το ενδιαφέρον μου για την τεχνολογία και την πληροφορική και συγκεκριμένα για τον τομέα της μηχανικής μάθησης και των μεγάλων δεδομένων.

Στόχος της εργασίας είναι η περιγραφή μεθόδων και αλγορίθμων της μηχανικής μάθησης και πιο συγκεκριμένα της μάθησης με επίβλεψη.

Για την παρούσα διπλωματική εργασία, η μεθοδολογία της έρευνας που ακολουθήθηκε βασίζεται κυρίως σε ξενόγλωσσες βιβλιογραφίες από πληροφοριακά βιβλία και άρθρα με σημαντικότερα να ξεχωρίζουν των: Bouveyron, C., and Jacques, J. - Γεωργούλη, Κ . Επιπροσθέτως, χρησιμοποιήθηκε ένας μεγάλος όγκος πληροφοριών που αποκτήθηκε από αξιόλογες ιστοσελίδες στο διαδίκτυο, οι οποίες σε πολλά σημεία πλαισίωσαν και έδωσαν ολοκληρωμένη επεξήγηση σε δύσκολες έννοιες και πληροφορίες που έπρεπε να αναφερθούν.

Παράλληλα στο δεύτερο μέρος, εφαρμόζονται και συγκρίνονται στην πράξη οι αλγόριθμοι που περιγράφτηκαν με τη βοήθεια του περιβάλλοντος R εξετάζοντας τα μειονεκτήματα και τα πλεονεκτήματα τους.

Συνοψίζοντας, η διπλωματική εργασία χωρίζεται σε δύο μέρη: το θεωρητικό και το πρακτικό. Στο πρώτο αναλύονται έννοιες της μηχανικής μάθησης με έμφαση στη μάθηση με επίβλεψη και στο δεύτερο κομμάτι με τη βοήθεια κατάλληλων συνόλων δεδομένων γίνεται η σύγκριση αλγορίθμων παλινδρόμησης και κατηγοριοποίησης σε περιβάλλον R.

Κεφάλαιο 2ο : Θεωρητικό Υπόβαθρο

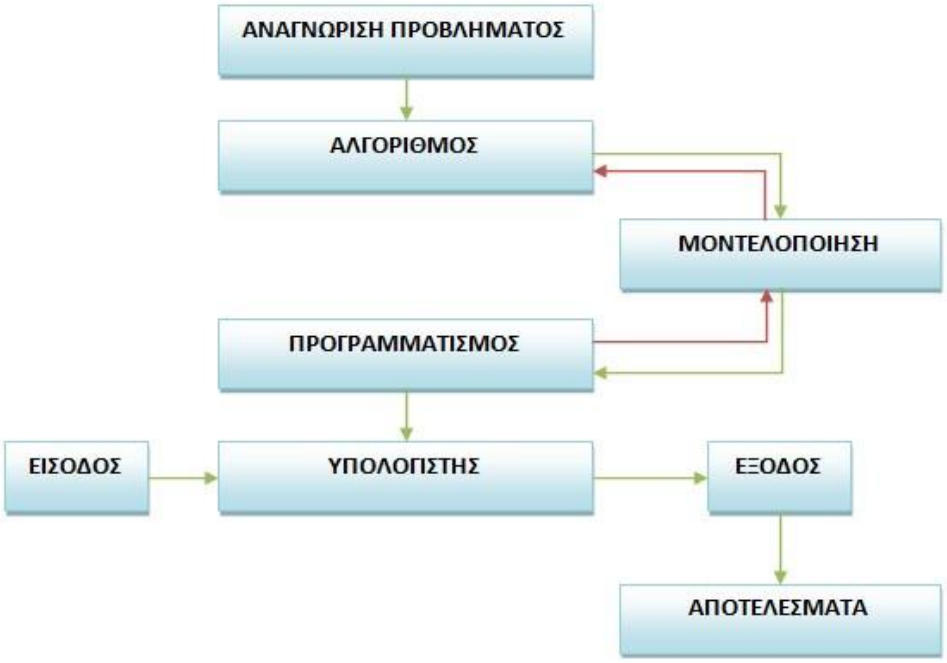
2.1 Μαθηματική μοντελοποίηση και μηχανική μάθηση

2.1.1 Μαθηματική μοντελοποίηση

Η μαθηματική μοντελοποίηση έχει ως αντικείμενο την έκφραση των βασικών νόμων και ιδιοτήτων, οι οποίοι σχετίζονται με τις διεργασίες της φύσης μέσα από σχετικές εξισώσεις και τη μελέτη και επίλυση τους υπό τις αρχικές συνθήκες και με την εφαρμογή των κατάλληλων μαθηματικών μεθόδων. Η αριθμητική μοντελοποίηση χρησιμοποιεί αριθμητικές και υπολογιστικές τεχνικές εισάγοντας τις κατάλληλες οριακές συνθήκες ούτως ώστε να επιλύσει αριθμητικά τα προβλήματα που της ανατίθενται με τη χρήση ηλεκτρονικού υπολογιστή. Εφαρμόζεται σε μία σειρά επιστημών, όπως είναι η μηχανολογία, η γενετική, η φυσική, η ιατρική κλπ. Μετά τη μοντελοποίηση λαμβάνουν χώρα οι εξομοιώσεις, ούτως ώστε το μοντέλο να αξιολογηθεί και εάν κριθεί αναγκαίο να επαναπροσδιοριστεί ώστε να εξασφαλιστεί η ποιότητα του.

Η μαθηματική μοντελοποίηση παρέχει τα παρακάτω οφέλη:

- Δυνατότητα αναπαράστασης της απόδοσης ενός συστήματος ή μίας διεργασίας.
- Δημιουργία χρήσιμων ψηφιακών δεδομένων.
- Αναπαράσταση πραγματικών φαινομένων τα οποία έχουν μοναδική συμπεριφορά, όπως είναι τα φυσικά, οικονομικά και κοινωνικά συστήματα.
- Δυνατότητα πρόβλεψης αποτελεσμάτων βάσει των πιθανοτήτων.
- Η οικονομική μοντελοποίηση αποδίδει περισσότερο σε σχέση με τη διεξαγωγή πειραμάτων στην πραγματική ζωή.
- Οι εφαρμογές μοντελοποίησης στις επιστήμες μηχανικών καθιστούν εφικτή την αξιολόγηση μετασχηματισμών οι οποίοι σχεδιάζεται να υλοποιηθούν.
- Η μοντελοποίηση διαδραματίζει σημαίνοντα ρόλο στην ανάλυση και πραγματοποίηση των αποφάσεων.



Εικόνα 1: Διάγραμμα ροής μοντελοποίησης και ανάπτυξης αλγορίθμων.

Η Εικόνα 1 παραθέτει ένα ενδεικτικό διάγραμμα ροής το οποίο αφορά την ανάπτυξη ενός αλγορίθμου για τη μοντελοποίηση ενός προβλήματος και την υλοποίηση μίας εφαρμογής σε περιβάλλον ηλεκτρονικού υπολογιστή η οποία θα παρέχει τα ζητούμενα αποτελέσματα [1].

2.1.2 Μηχανική μάθηση

Η μάθηση αποτελεί μία εκ των θεμελιωδών ιδιοτήτων της νοήμονος συμπεριφοράς του ανθρώπινου είδους. Ως έννοια δεν έχει γίνει πλήρως κατανοητή παρά τις χρόνιες μελέτες των κλάδων της Γνωστικής Ψυχολογίας και της Φιλοσοφίας [11].

Ο άνθρωπος διαχρονικά παρατηρεί το περιβάλλον του προσπαθώντας να το κατανοήσει και δημιουργώντας μία αφαιρετική του εκδοχή, γνωστή και ως μοντέλο. Η δημιουργία ενός μοντέλου με αυτόν τον τρόπο ονομάζεται επαγωγική μάθηση, ενώ η γενικότερη σχετική διαδικασία ονομάζεται επαγωγή. Επιπροσθέτως, ο άνθρωπος είναι ικανός να οργανώνει και να συσχετίζει τις εμπειρίες και τις παραστάσεις του μέσα από τη δημιουργία νέων δομών οι οποίες ονομάζονται πρότυπα. Η δημιουργία

μοντέλων και προτύπων μέσα από ένα σύνολο δεδομένων με τη χρήση ενός υπολογιστικού συστήματος ονομάζεται μηχανική μάθηση.

Η μηχανική μάθηση θα μπορούσε επίσης να οριστεί ως το φαινόμενο στα πλαίσια του οποίου ένα σύστημα βρίσκεται σε θέση να βελτιώσει την απόδοση του κατά την εκτέλεση κάποιας συγκεκριμένης εργασίας χωρίς να είναι αναγκαίο να προγραμματιστεί εκ νέου. Υπό την προσέγγιση αυτή η μηχανική μάθηση αποσκοπεί στη δημιουργία μηχανών οι οποίες βρίσκονται σε θέση να μαθαίνουν, δηλαδή να βελτιώνουν την απόδοση τους σε διάφορους τομείς αξιοποιώντας τις πρότερες γνώσεις και εμπειρίες [11].

Ως επιστημονικός κλάδος της Τεχνητής Νοημοσύνης η μηχανική μάθηση επικεντρώνεται στη μελέτη αλγορίθμων οι οποίοι βελτιώνουν τη συμπεριφορά τους σε κάποια εργασία η οποία τους έχει ανατεθεί αξιοποιώντας την εμπειρία τους [11].

Αναφορικά με τη σχεδίαση συστημάτων μηχανικής μάθησης για συστήματα τα οποία ανήκουν στη συμβολική τεχνητή νοημοσύνη, ως δυνατότητα μάθησης θεωρείται η δυνατότητα απόκτησης επιπρόσθετης γνώσης, η οποία τροποποιεί την υπάρχουσα γνώση, είτε μέσα από την αλλαγή των χαρακτηριστικών της, είτε αυξομειώνοντας την. Στα συστήματα μη συμβολικής τεχνητής νοημοσύνης, μάθηση είναι η δυνατότητα των συστημάτων να μετασχηματίζουν την εσωτερική τους δομή αντί της κατάλληλης μεταβολής της καταχωρημένης σε αυτά γνώσης κατά τη διαδικασία του σχεδιασμού τους [11].

Αν και ακόμα υπάρχει σημαντική απόσταση από την υλοποίηση μηχανών, οι οποίες έχουν την ικανότητα μάθησης παρομοίου επιπέδου με εκείνης του ανθρώπου, σε συγκεκριμένες περιοχές μάθησης έχουν αναπτυχθεί αλγόριθμοι, οι οποίοι καθιστούν δυνατή την υλοποίηση σύγχρονων εμπορικών εφαρμογών, οι οποίες μάλιστα παρουσιάζουν ιδιαίτερη επιτυχία. Συνάμα, οι εφαρμογές τεχνητής νοημοσύνης παρουσιάζουν ορατά αποτελέσματα, τα οποία δίνουν απαντήσεις σε προβλήματα άλλων κλάδων, οι οποίοι ασχολούνται με τη διερεύνηση της ικανότητας του ανθρώπου να μαθαίνει [11].

Στα πλαίσια της μηχανικής μάθησης αναπτύσσεται επίσης η Εξελικτική Μάθηση, αντικείμενο της οποίας είναι η μίμηση διαδικασιών φυσικής αναπαραγωγής έμβιων

όντων. Εφαρμόζεται κατά κύριο λόγο σε προβλήματα βελτιστοποίησης, ενώ άρρηκτα συνδεδεμένοι μαζί της είναι οι γενετικοί αλγόριθμοι [11].

Πέραν της Τεχνητής Νοημοσύνης, επιστημονικοί κλάδοι οι οποίοι αξιοποιούν τις δυνατότητες της Μηχανικής Μάθησης είναι εκείνοι της Εξόρυξης Δεδομένων, των Πιθανοτήτων και της Στατιστικής, της Θεωρίας Πληροφοριών, της Αριθμητικής Βελτιστοποίησης, της Θεωρίας της Πολυπλοκότητας, της προσαρμοστικής Θεωρίας Ελέγχου, της εξελικτικής και γνωστικής Ψυχολογίας, της Νεύρο-βιολογίας και της Γλωσσολογίας [11].

Η μηχανική μάθηση χωρίζεται σε δύο κύριες κατηγορίες. Η πρώτη από αυτές ονομάζεται μάθηση με επίβλεψη και περιλαμβάνει τη χρήση παραδειγμάτων ενώ η δεύτερη μάθηση χωρίς επίβλεψη ή από παρατήρηση. Στη μάθηση με επίβλεψη το σύστημα μαθαίνει μία έννοια ή συνάρτηση διαμέσου ενός συνόλου δεδομένων το οποίο περιγράφει ένα μοντέλο. Η μάθηση χωρίς επίβλεψη συνίσταται στην ανακάλυψη συσχετίσεων ή ομάδων δεδομένων από το σύστημα μόνο του, το οποίο δημιουργεί το πρότυπο χωρίς να γνωρίζει κάτι σχετικά με την ύπαρξη συσχετίσεων, το είδος και το πλήθος τους.

Για την περίπτωση της μάθησης με επίβλεψη, το σύστημα αρχικά εκπαιδεύεται σε ένα σύνολο παραδειγμάτων στο οποίο το κάθε παράδειγμα χαρακτηρίζεται από μία κατηγορία. Τα προβλήματα ταξινόμησης αποτελούν μία από τις πλέον χαρακτηριστικές περιπτώσεις μάθησης με επίβλεψη. Σε αυτά, το κάθε παράδειγμα εκπαίδευσης αντιστοιχεί σε ένα διάνυσμα. Το διάνυσμα αυτό είναι ένα σύνολο από χαρακτηριστικά ή γνωρίσματα το οποίο περιλαμβάνει μία τιμή κλάσης ή κατηγορίας η οποία περιγράφει το επιθυμητό αποτέλεσμα ή έννοια – στόχο. Μία σειρά αλγορίθμων μηχανικής μάθησης έχουν σχεδιαστεί για προβλήματα τέτοιου είδους, με ορισμένα παραδείγματα τους ID3 [18], C4.5 [19], οι οποίοι αφορούν την εκμάθηση δέντρων αποφάσεων, τη μάθηση στηριζόμενη στον μετασχηματισμό καθοδηγούμενη από σφάλματα (transformation-based error-driven learning – TBED) [20], [21], [22], [23] η οποία εφαρμόζεται σε προβλήματα εκμάθησης λιστών αποφάσεων, τον αλγόριθμο Naive Bayes [24], τον αλγόριθμο των k-κοντινότερων γειτόνων (k-nearest neighbors) [25] και τα κρυφά μοντέλα Markov [26]. Το εκπαιδευμένο μοντέλο που προκύπτει μέσα από έναν αλγόριθμο ταξινόμησης ο οποίος εφαρμόζεται σε ένα

σύνολο διανυσματικών χαρακτηριστικών συχνά αναφέρεται ως ταξινομητής (classifier).

2.2 Μάθηση με επίβλεψη

Η μάθηση με επίβλεψη (supervised learning), γνωστή και ως μάθηση με παραδείγματα (learning from examples) αφορά μία κατηγορία μάθησης στην οποία το σύστημα δέχεται ως είσοδο μία σειρά πληροφοριών (η οποία συνήθως είναι μία περιγραφή του μοντέλου) και πρέπει να είναι ικανό να «μάθει» μία έννοια ή συνάρτηση [6].

2.2.1 Γραμμική παλινδρόμηση

Η παλινδρόμηση είναι μία τεχνική μάθησης με επίβλεψη αντικείμενο της οποίας είναι ο προσδιορισμός της σχέσης μίας εξαρτημένης μεταβλητής y με μία ή περισσότερες άλλες ανεξάρτητες μεταβλητές x_1, x_2, …, x_n. Στις εφαρμογές μηχανικής μάθησης εφαρμόζεται προκειμένου να εκχωρηθούν δεδομένα στις μεταβλητές πρόβλεψης στην περίπτωση κατά την οποία οι μεταβλητές είναι συνεχείς. Η ανάλυση της παλινδρόμησης είναι ιδιαιτέρως σημαντική καθώς παρέχει μία στατιστική εκτίμηση των συσχετίσεων, δηλαδή του κατά πόσο ο βαθμός εμπιστοσύνης βρίσκεται κοντά στην εκτίμηση που πραγματοποιήθηκε [7].

Κατά την ανάλυση παλινδρόμησης, τα δεδομένα $S = \{(\mathbf{x_1}, y_{1)}, \ldots, (\mathbf{x_n}, y_n)\}$ τα οποία προέρχονται από έναν πληθυσμό P, υποτίθεται ότι είναι ανεξάρτητα και ιδανικά κατανεμημένα δείγματα ενός ζεύγους μεταβλητών (**X,** Y) μίας άγνωστης κατανομής. Οι παρατηρήσεις $\mathbf{x_j}$, $j = 1,\ldots,n$ είναι οι τιμές της ντετερμινιστικής επεξηγηματικής μεταβλητής $\boldsymbol{X} = \left(X^{(1)}, \ldots, X^{(p)}\right)^t \in R^p$ ενώ τα αντίστοιχα y_j είναι οι πραγματοποιήσεις (realizations) της στοχαστικής μεταβλητής $Y \in \mathscr{R}$. Ένα γενικό πρόβλημα μοντελοποίησης συνίσταται στον προσδιορισμό της σχέσης μεταξύ της επεξηγηματικής μεταβλητής **X** (γνωστής και ως συμμεταβλητή) και της μεταβλητής

απόκρισης Y (ή εξαρτημένης μεταβλητής). Τόσο οι τυπικές παραμετρικές όσο και οι μη παραμετρικές προσεγγίσεις θεωρούν το παρακάτω μοντέλο παλινδρόμησης:

$$Y = f(\boldsymbol{X}, \beta) + \varepsilon, \quad (2.1.)$$

όπου τα υπόλοιπα $\varepsilon \sim N(0, \sigma^2)$ είναι ανεξάρτητα και όπου **β** είναι το διάνυσμα των παραμέτρων της παλινδρόμησης. Το μοντέλο αυτό είναι ισοδύναμο με την παρακάτω υπόθεση σχετικά με την κατανομή:

$$Y|\boldsymbol{X} \sim N(f(\boldsymbol{X}, \boldsymbol{\beta})\sigma^2)$$

όπου η συνάρτηση παλινδρόμησης f (x,β) ορίζεται ως η υπό όρους προσδοκία E[Y|**X**=**x**]. Επομένως ο μόνος τρόπος για να καθοριστεί η σχέση μεταξύ της μεταβλητής απόκρισης Y και της συμμεταβλητής **X** είναι οι υποθέσεις της f(**x**,**β**). Πιο συγκεκριμένα, η παραμετρική παλινδρόμηση επιτυγχάνει αυτή τη σύνδεση μέσα από την υπόθεση μίας ειδικής μορφής για την f (**x**, **β**). Το πλέον κοινό μοντέλο είναι η γραμμική μορφή:

$$f(x, \beta) = \sum_{i=0}^{d} \beta_i \psi_i(\boldsymbol{x}), \quad (2.2.)$$

όπου $\beta = (\beta_0, \beta_1, \ldots, \beta_d)^t \in R^{d+1}$ είναι οι παράμετροι παλινδρόμησης, $\psi_0(\mathbf{x}) = 1$ και $(\psi_i)_{1 \le i \le d}$ μία βάση των συναρτήσεων παλινδρόμησης:

$$\psi_i = R^p \to R,$$

οι οποίες για παράδειγμα μπορούν να είναι ταυτοτικές, πολυωνυμικές συναρτήσεις, κατά τμήματα πολυωνυμικές καμπύλες ή κυματίδια. Πρέπει να σημειωθεί ότι η συνήθης γραμμική παλινδρόμηση προκύπτει όταν d = p και $\psi_i(\mathbf{x}) = x^{(i)}$ για i = 1,…,d. Η συνάρτηση παλινδρόμησης μπορεί επίσης να γραφεί σε μορφή πίνακα ως:

$$f(\boldsymbol{x}, \boldsymbol{\beta}) = \beta^t \Psi(\boldsymbol{x}),$$

όπου $\Psi(x) = (1, \psi_1(x), \ldots, \psi_d(x))^t$ [8].

2.2.2 Λογιστική παλινδρόμηση

Η πολλαπλή λογιστική παλινδρόμηση (multiple logistic regression) είναι μία γνωστή και ευρέως χρησιμοποιούμενη μέθοδος στην οποία η πιθανότητα της παρουσίας της μεταβλητής Y λαμβάνει τη μορφή Y = 1/{1 + exp – {b(0) + b(1).x(1) + … + b(n).x(n)). Εδώ οι x(1) … x(n) αναπαριστούν τις τιμές των ανεξάρτητων μεταβλητών και οι b(1)…b(n) αναπαριστούν τους συντελεστές. Τα αποτελέσματα της πολλαπλής λογιστικής παλινδρόμησης λαμβάνουν τις Y τιμές στο εύρος 0 – 1. Επομένως η μέθοδος ενδέχεται να δημιουργήσει ένα μοντέλο πιθανοτήτων με πιθανότητες ως εξόδους [9].

2.2.3 Κατηγοριοποίηση

Η κατηγοριοποίηση είναι μία από τις πλέον σημαντικότερες διαδικασίες της εξόρυξης δεδομένων. Στα πλαίσια της εξετάζεται ένα αντικείμενο το οποίο εν συνεχεία και βάσει των χαρακτηριστικών του εντάσσεται σε ένα προκαθορισμένο σύνολο κλάσεων. Η κατηγοριοποίηση χαρακτηρίζεται από [10]:

1. Έναν καλά καθορισμένο ορισμό των κατηγοριών (ή κλάσεων).
2. Ένα σύνολο το οποίο χρησιμοποιείται για την εκπαίδευση του μοντέλου και περιέχει προ-κατηγοριοποιημένα παραδείγματα.

Η βασική εργασία της κατηγοριοποίησης είναι η δημιουργία ενός μοντέλου το οποίο θα μπορούσε να χρησιμοποιηθεί για την κατηγοριοποίηση δεδομένων τα οποία προς το παρόν δεν έχουν κατηγοριοποιηθεί.

2.2.3.1 Μηχανές Διανυσμάτων Υποστήριξης (Support Vector Machines – SVMs)

Οι Μηχανές Διανυσμάτων Υποστήριξης ανήκουν στην οικογένεια των μεθόδων ταξινόμησης που βασίζονται σε πυρήνες. Στηρίζονται στη Θεωρία Στατιστικής Μάθησης και τα νευρωνικά δίκτυα Perceptron. Η σημερινή τους μορφή εισήχθη από τον Vladimir Vapnik το 1992. Η γενικότερη ιδέα η οποία επεκτάθηκε ώστε να δημιουργηθεί η σημερινή μορφή προτάθηκε κατά τη δεκαετία του 1960 [6].

Σήμερα αποτελούν μία από τις δημοφιλέστερες μεθόδους γραμμικής και μη παρεμβολής και κατηγοριοποίησης έχοντας πληθώρα εφαρμογών. Ορισμένα χαρακτηριστικά παραδείγματα είναι τα εξής [6]:

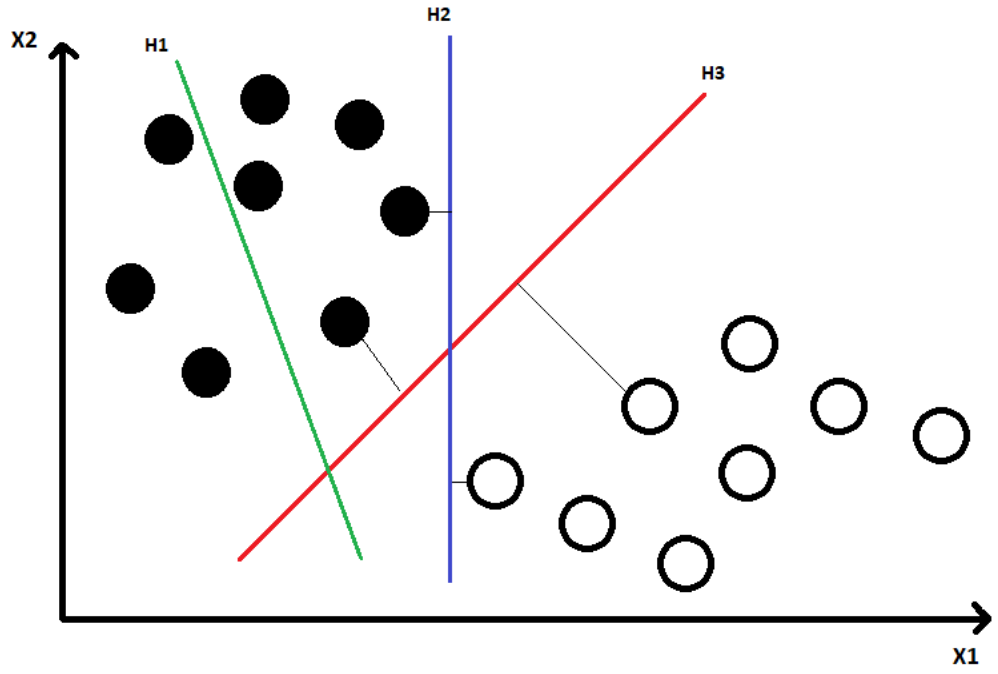
- Αναγνώριση γραφής
- Κατηγοριοποίηση κειμένων
- Κατηγοριοποίηση δεδομένων έκφρασης γονιδίων (gene expression data)

Στις εφαρμογές κατηγοριοποίησης στόχος των SVM είναι η εύρεση μίας υπερ-επιφάνειας (hyper surface) η οποία διαχωρίζει τα αρνητικά από τα θετικά παραδείγματα. Η συγκεκριμένη υπερεπιφάνεια επιλέγεται με γνώμονα τη μέγιστη δυνατή απόσταση από τα πλησιέστερα θετικά και αρνητικά παραδείγματα [6].

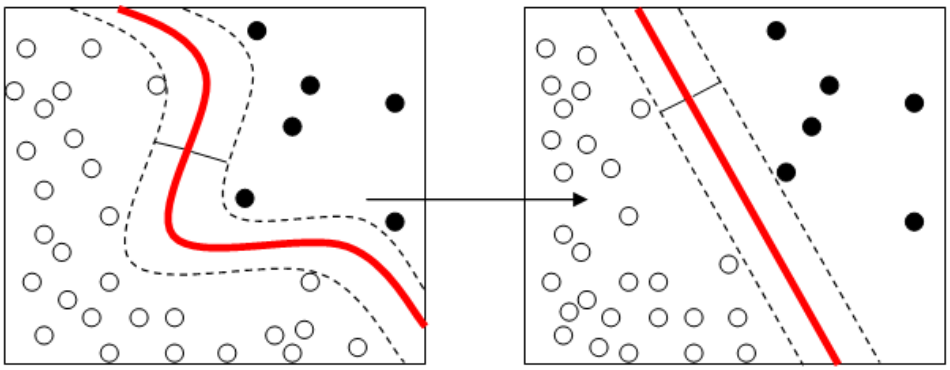
Σύμφωνα με τον ορισμό, ένα SVM κατασκευάζει ένα υπερεπίπεδο ή ένα σύνολο υπερεπιπέδων σε ένα χώρο πολλαπλών ή άπειρων διαστάσεων, το οποίο μπορεί να χρησιμοποιηθεί για ταξινόμηση, παλινδρόμηση ή άλλες εργασίες όπως η ανίχνευση ακραίων τιμών. Ένας καλός διαχωρισμός επιτυγχάνεται από το υπερεπίπεδο που έχει τη μεγαλύτερη απόσταση από το πλησιέστερο σημείο εκπαίδευσης δεδομένων οποιασδήποτε κλάσης (το λεγόμενο λειτουργικό περιθώριο), καθώς γενικά όσο μεγαλύτερο είναι το περιθώριο, τόσο χαμηλότερο είναι το σφάλμα γενίκευσης του κατηγοριοποιητή.

Για να διατηρηθεί το υπολογιστικό φορτίο σε λογικά πλαίσια, οι αντιστοιχίσεις που χρησιμοποιούνται από τα σχήματα SVM έχουν σχεδιαστεί για να διασφαλίζουν ότι τα σημεία των ζευγών των διανυσμάτων δεδομένων εισόδου μπορούν να υπολογιστούν εύκολα από την άποψη των μεταβλητών στον αρχικό χώρο, ορίζοντάς τα ως όρους

μίας συνάρτησης πυρήνα *k(x,y)* η οποία έχει επιλεγεί έτσι ώστε να ταιριάζει στο πρόβλημα. Τα υπερεπίπεδα στον χώρο πολλαπλών διαστάσεων ορίζονται ως το σύνολο σημείων των οποίων το γινόμενο με ένα διάνυσμα σε αυτόν τον χώρο είναι σταθερό. Τα διανύσματα που ορίζουν τα υπερεπίπεδα μπορούν να επιλεγούν να είναι γραμμικοί συνδυασμοί με παραμέτρους α_i των απεικονίσεων των χαρακτηριστικών διανυσμάτων x_i οι οποίες εμφανίζονται στη βάση δεδομένων. Με αυτή την επιλογή υπερεπιπέδου, τα σημεία *x*στον χώρο χαρακτηριστικών που απεικονίζονται στο υπερεπίπεδο ορίζονται από τη σχέση $\sum_i \alpha_i k(x_i, x) = σταθερό$. Πρέπει να σημειωθεί ότι εάν το *k(x,y)* λάβει υπερβολικά χαμηλές τιμές καθώς το y μεγαλώνει αποκτώντας σημαντική διαφορά με το *x*, κάθε όρος στο άθροισμα μετρά τον βαθμό εγγύτητας του σημείου ελέγχου *x*με το αντίστοιχο σημείο της βάσης δεδομένων x_i. Με αυτόν τον τρόπο, το άθροισμα των παραπάνω πυρήνων μπορεί να χρησιμοποιηθεί για τη μέτρηση της σχετικής εγγύτητας κάθε σημείου ελέγχου προς τα σημεία δεδομένων που προέρχονται από κάποιο από τα σύνολα που πρέπει να διακριθούν. Πρέπει να σημειωθεί το γεγονός του ότι το σύνολο των σημείων *x* που έχουν αντιστοιχηθεί σε οποιοδήποτε υπερεπίπεδο μπορεί να είναι αρκετά περίπλοκο, καθιστώντας δυνατή μία πολύ πιο περίπλοκη διάκριση μεταξύ συνόλων τα οποία δεν είναι καθόλου κυρτά στον αρχικό χώρο.



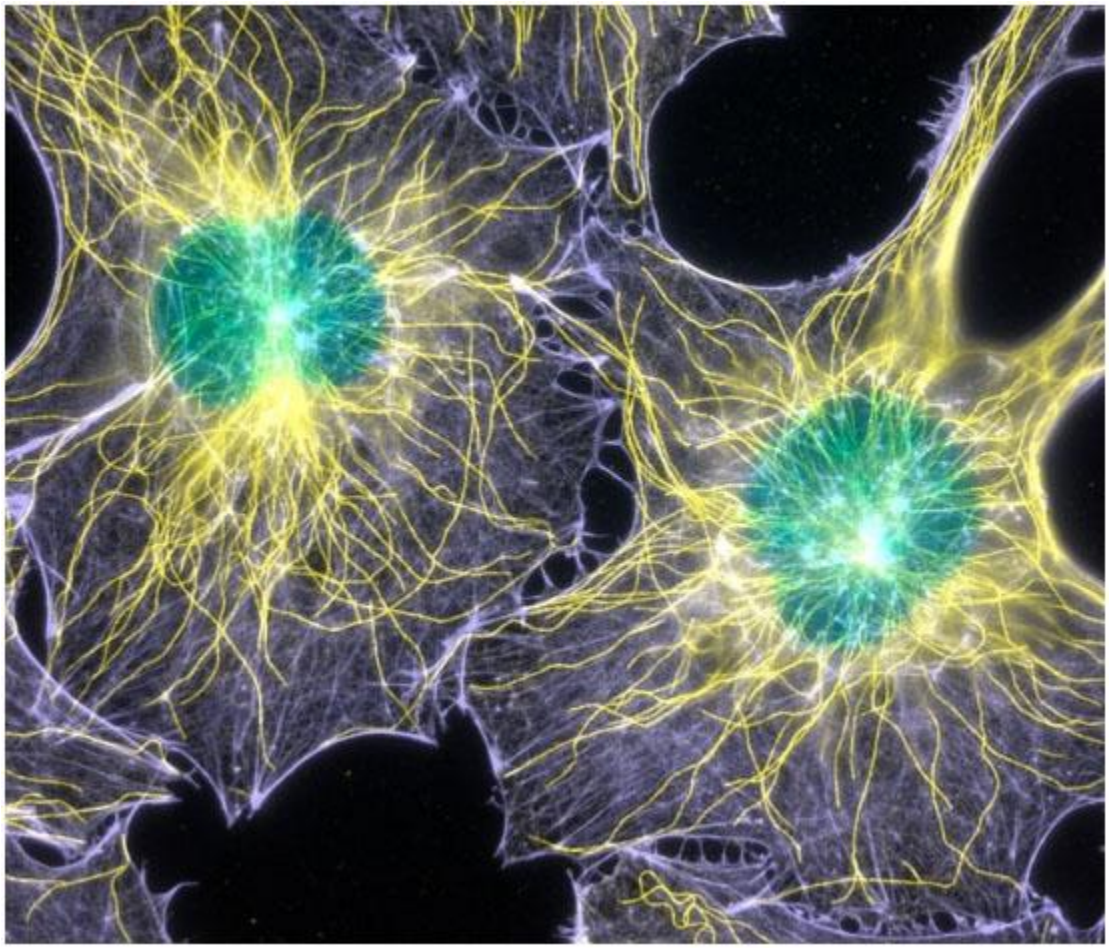
Εικόνα 2 : Η Η1 γραμμή δεν διαχωρίζει τις τάξεις. Η Η2 το κάνει, αλλά μόνο με ένα μικρό περιθώριο. Η Η3 τις χωρίζει με το μέγιστο περιθώριο.



Εικόνα 3: Μηχανή πυρήνα.

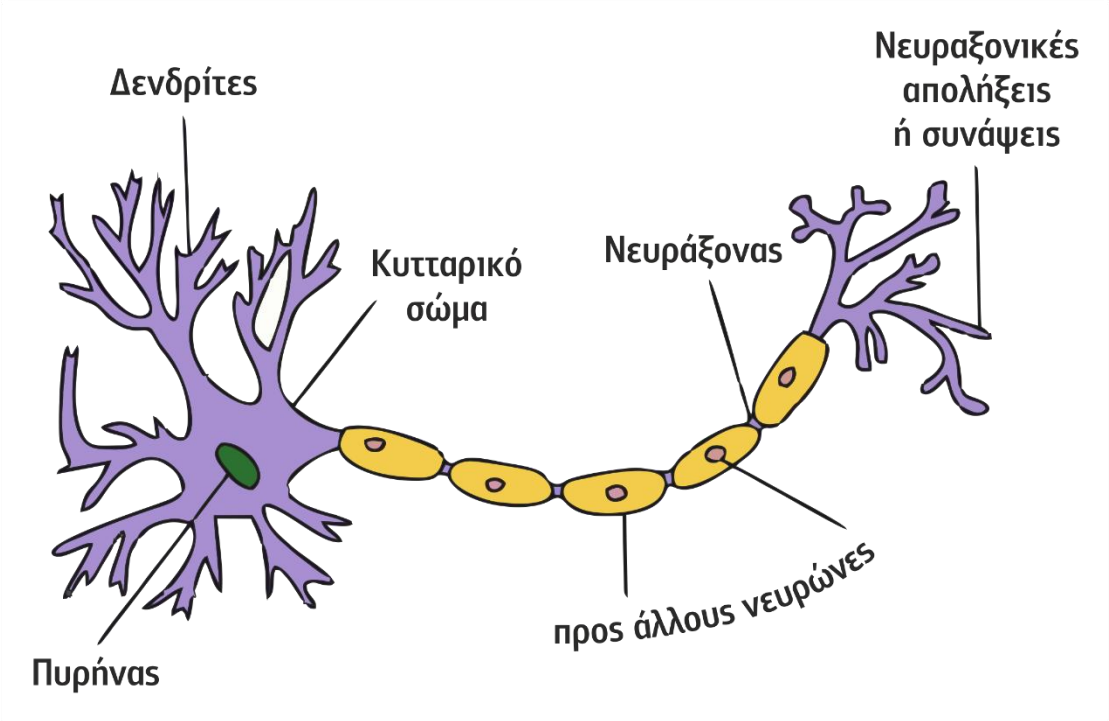
2.2.3.2 Νευρωνικά δίκτυα

Ο όρος νευρωνικά δίκτυα (neural networks) αφορά την περιγραφή διαφόρων μαθηματικών μοντέλων τα οποία έχουν εμπνευστεί από βιολογικά μοντέλα, τα οποία έχουν ως στόχο τους τη μίμηση της λειτουργίας των νευρώνων του ανθρώπινου εγκέφαλου. Κατά τον $19^{ο}$ αιώνα οι επιστήμονες ανακάλυψαν ότι ο ανθρώπινος εγκέφαλος αποτελείται από διακριτά στοιχεία, τα οποία ονομάστηκαν νευρώνες και επικοινωνούν μεταξύ τους. Τα στοιχεία αυτά αποτελούν το θεμελιώδες δομικό στοιχείο του ανθρώπινου εγκέφαλου. Ο αριθμός των νευρώνων του ανθρώπινου εγκεφάλου έχει υπολογιστεί σε περίπου 10 δισεκατομμύρια, οι οποίοι οργανώνονται σε ομάδες, κάθε μία από τις οποίες αποτελεί ένα φυσικό νευρωνικό δίκτυο. Ως εκ τούτου, ο ανθρώπινος εγκέφαλος εμπεριέχει εκατοντάδες φυσικών νευρωνικών δικτύων, το καθένα εκ των οποίων με τη σειρά του εμπεριέχει χιλιάδες διασυνδεδεμένους νευρώνες με τον μέσο αριθμό των διασυνδέσεων ανά νευρώνα να κυμαίνεται μεταξύ 1.000 και 10.000 [11].



Εικόνα 4: Μικροσκοπική φωτογραφία φυσικών νευρώνων [11].

Οι νευρώνες διαχωρίζονται από τα υπόλοιπα κύτταρα διαμέσου μίας μεμβράνης ενώ έχουν τη δυνατότητα μεταφοράς ηλεκτρικών σημάτων προς άλλους νευρώνες με τους οποίους επικοινωνούν.



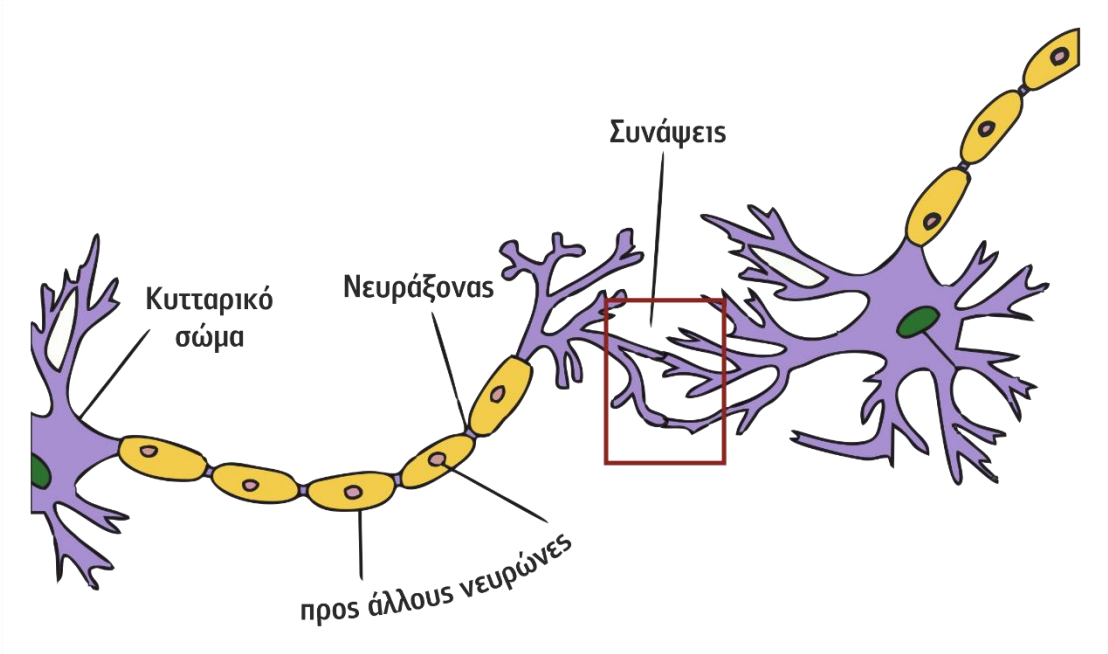
Εικόνα 5: Σχηματική αναπαράσταση φυσικού νευρώνα [11].

Ο κάθε νευρώνας διαχωρίζεται σε τρία κύρια τμήματα, όπως φαίνεται και στην Εικόνα 5 [11]:

- Τους δενδρίτες, οι οποίοι αποτελούν τα κανάλια εισόδου του νευρώνα.
- Το κύριο κυτταρικό σώμα.
- Τον νευράξονα (neuroaxon) ο οποίος συνδέει τον νευρώνα με άλλους νευρώνες.

Τα σήματα μεταφέρονται από τον άξονα ενός νευρώνα στους δενδρίτες γειτονικών νευρώνων διαμέσου ενός σημείου ένωσης, το οποίο καλείται σύναψη. Ένας νευρώνας έχει τη δυνατότητα λήψης σημάτων από ένα σύνολο γειτονικών νευρώνων διαμέσου των δενδριτών, να τα επεξεργαστεί και στη συνέχεια να τροφοδοτήσει μία έξοδο προς ένα άλλο σύνολο γειτονικών νευρώνων από τον άξονα. Μία σχετική αναπαράσταση παρατίθεται στην Εικόνα 6. Τα σήματα που λαμβάνονται από τους δενδρίτες «ζυγίζονται» και τα αποτελέσματα τους προστίθενται. Όταν το άθροισμα υπερβεί μία οριακή τιμή τότε δημιουργείται μία έξοδος από τον νευρώνα στον άξονα του, η οποία

έχει τη μορφή νευρικής ώσης ή ηλεκτρικού σήματος. Η έξοδος αυτή μεταφέρεται μέσα από τις συνάψεις στους γειτονικούς νευρώνες [11].



Εικόνα 6: Αναπαράσταση διασύνδεσης φυσικών νευρώνων [11].

Προκειμένου να παραχθεί ένα σήμα ο νευρώνας δέχεται σήματα εισόδου τα οποία επιδρούν στο δυναμικό του δημιουργώντας αυξομειώσεις. Όταν το συνολικό άθροισμα του δυναμικού υπερβεί κάποια οριακή τιμή (η οποία σχετίζεται με την κατηγορία του κάθε κυττάρου και κυμαίνεται μεταξύ 40 – 75 mV) τότε προκύπτει διέγερση του νευρώνα και παραγωγή του ηλεκτρικού σήματος. Το ηλεκτρικό σήμα μεταφέρεται πάντοτε από τον νευρώνα προς μία σταθερή και προβλέψιμη κατεύθυνση. Οι καταστάσεις των σημάτων διαχωρίζονται σε δύο κατηγορίες [11]:

- Δυναμικό ηρεμίας
- Δυναμικό ενέργειας

Τα σήματα που λαμβάνονται από έναν νευρώνα μεταβάλλονται από τα ηλεκτρικά χαρακτηριστικά των επαφών των συνάψεων ώστε να εμποδίζονται ορισμένα από αυτά και να καθίσταται δυνατή η διάδοση κάποιων άλλων. Τα ηλεκτρικά χαρακτηριστικά των νευρώνων αποτελούν μία μοναδική σε κάθε νευρώνα πληροφορία. Μέσα από αυτόν τον τρόπο οι πληροφορίες που βρίσκονται σε ένα δίκτυο διανέμονται στους νευρώνες του.

Η μεταβίβαση της πληροφορίας λαμβάνει χώρα μέσω του δυναμικού ενέργειας, το οποίο δεν καθορίζεται από τον τύπο του σήματος αλλά από την οδό του εγκεφάλου διαμέσου διακριτά επικοινωνούντων νευρώνων από τους οποίους διέρχεται το σήμα.

Στο σημείο αυτό πρέπει να αναφερθεί ότι ο εγκέφαλος διαφέρει σημαντικά από έναν ψηφιακό ηλεκτρονικό υπολογιστή, χωρίς να είναι δυνατή η αντικατάσταση του ενός από τον άλλο. Μία από τις σημαντικότερες διαφορές είναι το γεγονός του ότι τα πολλαπλά φυσικά νευρωνικά δίκτυα του εγκεφάλου οργανώνονται σε τμήματα τα οποία λειτουργούν παράλληλα. Το κάθε ένα από αυτά έχει τη δυνατότητα πρόκλησης ανεξάρτητων συμπεριφορών με τη διαδικασία ανάληψης λειτουργιών να διέπεται από πλαστικότητα (δηλαδή προσαρμογή σε μεταβολές του εσωτερικού ή εξωτερικού περιβάλλοντος ούτως ώστε να διασφαλιστεί η κατά το δυνατόν επιτυχής τους λειτουργία) επομένως δεν μπορεί να λάβει χώρα εξομοίωση με ηλεκτρονικά κυκλώματα τα οποία δεν διαθέτουν τέτοιου είδους χαρακτηριστικά [11].

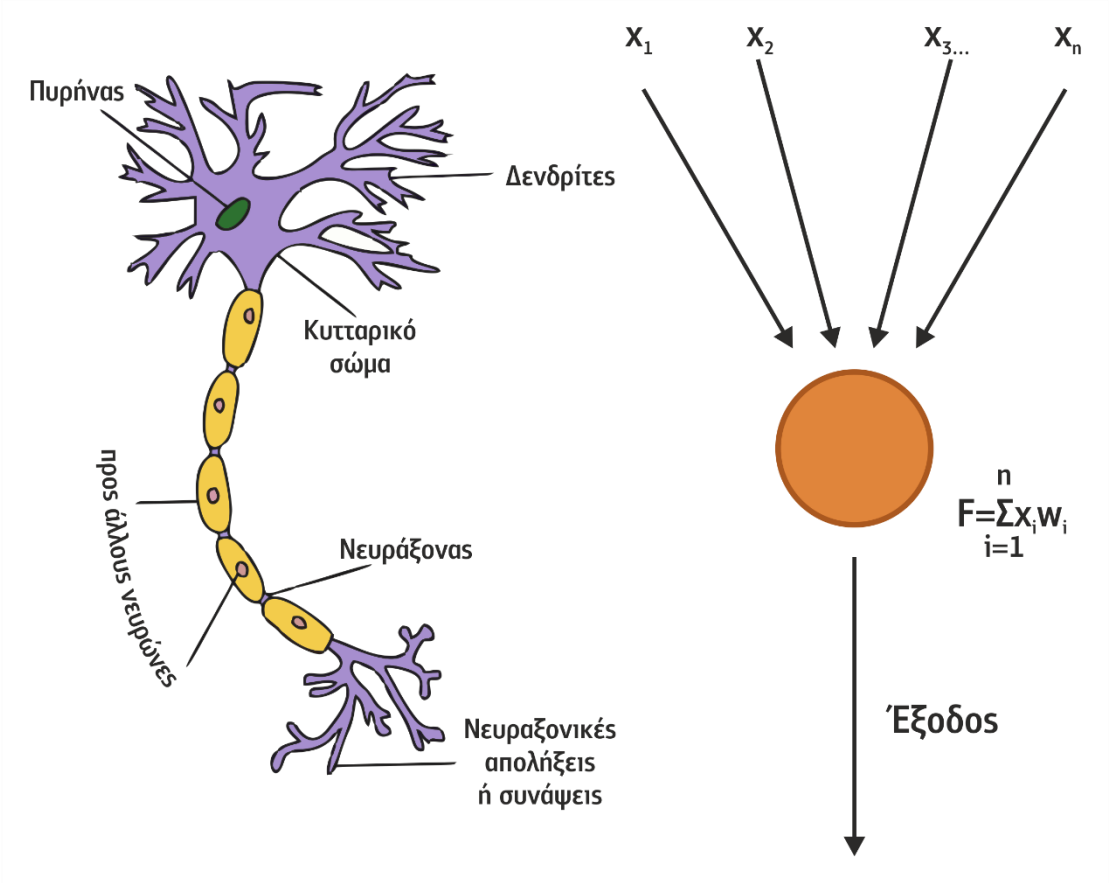
Τα μαθηματικά μοντέλα που εφαρμόζονται στα Τεχνητά Νευρωνικά Δίκτυα (ΤΝΔ – Artificial Neural Networks) βρίσκονται σε πλήρη αντιστοιχία με τα βιολογικά και αποτελούνται από έναν αριθμό απλών μονάδων επεξεργασίας, οι οποίες διέπονται από υψηλό βαθμό εσωτερικής διασύνδεσης, ενώ οργανώνονται σε στρώματα. Τα δίκτυα αυτά επεξεργάζονται πληροφορίες παρουσιάζοντας δυναμική ανταπόκριση σε εξωτερικά ερεθίσματα, τα οποία αποτελούν τις εισόδους των ΤΝΔ. Οι τεχνητοί νευρώνες αποτελούνται από πολλές εισόδους xi και μία έξοδο y. Η κάθε είσοδος σταθμίζεται με ένα βάρος wi, ενώ τα αποτελέσματα προστίθενται μέσα από μία συνάρτηση άθροισης (summation function) F, η οποία έχει την παρακάτω μορφή [11]:

$$F = \sum_{i}^{n} x_i w_i$$

Ο τεχνητός νευρώνας παρέχει μία έξοδο διαμέσου της συνάρτησης μεταφοράς (transfer function) μόνο στην περίπτωση κατά την οποία το σταθμισμένο άθροισμα των εισόδων υπερβαίνει μία συγκεκριμένη τιμή – κατώφλι (threshold value) θ, δηλαδή στην περίπτωση κατά την οποία ισχύει η παρακάτω συνθήκη [11]:

$$\sum_{i}^{n} x_i w_i - \theta > 0$$

Ο τεχνητός νευρώνας είναι μία απλοποιημένη εκδοχή του φυσικού νευρώνα, καθώς τα βάρη των συνδέσεων σχηματίζουν τα ηλεκτρικά χαρακτηριστικά των επαφών των συνάψεων ενώ το κατώφλι αποτελεί μία προσομοίωση του κορεσμού των φυσικών νευρώνων. Στην Εικόνα 7 παρατίθεται μία σχετική σύγκριση.



Εικόνα 7: Φυσικός νευρώνας και τεχνητός νευρώνας Perceptron.

Το στοιχειώδες δίκτυο Perceptron (basic Perceptron) είναι ένα από τα πλέον απλά ΤΝΔ το οποίο προσομοιώνει τη λειτουργία ενός βιολογικού νευρώνα. Πρόκειται για ένα δίκτυο το οποίο αποτελείται μόνο από έναν νευρώνα. Η συνάρτηση μεταφοράς g παρέχει την έξοδο α για ένα διάνυσμα εισόδου x = (x1, x_2, …, x_n) ως εξής [11]:

$$\alpha = g\left(\sum_{i=1}^{n} x_i w_i\right)$$

Μία σύντομη περιγραφή ενός ΤΝΔ έχει ως εξής [11]:

- Είναι οργανωμένο σε στρώματα (layers). Τα ενδιάμεσα στρώματα ονομάζονται κρυφά (hidden layers), ενώ δεν είναι υποχρεωτική η ύπαρξη τους.
- Τα στρώματα αποτελούνται από μονάδες (units) και κόμβους (nodes) οι οποίοι συνδέονται με τέτοιο τρόπο ώστε η κάθε μονάδα να είναι συνδεδεμένη με πολλές άλλες είτε του ίδιου είτε διαφορετικού στρώματος.
- Οι μονάδες παρουσιάζουν επίδραση σε άλλες μονάδες με τη μορφή είτε της διέγερσης τους είτε της αναστολής της ενεργοποίησης τους. Προκειμένου να επιτευχθεί η συγκεκριμένη λειτουργία η μονάδα λαμβάνει το σταθμισμένο άθροισμα όλων των εισόδων μέσα από τους συνδέσμους που καταλήγουν σε αυτήν, ενώ παράγει μία μοναδική έξοδο μέσα από τη συνάρτηση μεταφοράς όταν το άθροισμα υπερβεί μία τιμή – κατώφλι.
- Οι είσοδοι εισέρχονται στο δίκτυο διαμέσου της στρώσης εισόδου (input layer) η οποία επικοινωνεί με ένα ή περισσότερα κρυφά επίπεδα. Τα κρυφά επίπεδα συνδέονται με τη στρώση εξόδου (output layer) από την οποία εξάγεται η απάντηση.

Τα βασικά στοιχεία της αρχιτεκτονικής των ΤΝΔ τα οποία και πρέπει να καθοριστούν κατά την υλοποίηση είναι τα εξής [11]:

- Ο αριθμός των ενδιάμεσων κρυφών στρώσεων.
- Ο αριθμός των μονάδων και κόμβων του κάθε επιπέδου.
- Ο τρόπος διασύνδεσης των μονάδων.
- Η τιμή – κατώφλι.
- Η συνάρτηση μεταφοράς.
- Οι τιμές των αρχικών βαρών μεταξύ των μονάδων.
- Οι κανόνες εκπαίδευσης που θα χρησιμοποιηθούν ώστε να ενισχυθούν οι σύνδεσμοι μεταξύ των μονάδων κατά την εκπαίδευση του δικτύου.

Η εξέλιξη των ΤΝΔ έλαβε χώρα ως εξής [11]:

- Το 1943 οι McCulloch και Pitts δημιουργούν το πρώτο μοντέλο ΤΝΔ.
- Το 1949 ο Hebb δημιουργεί τον ομώνυμο αλγόριθμο εκπαίδευσης σύμφωνα με τον οποίο κάθε φορά που ενεργοποιείται μία σύναψη ενισχύεται και έτσι το δίκτυο μαθαίνει περισσότερο το πρότυπο που του παρουσιάζεται.

- Το 1957 προτάθηκε από τον Rosenblatt το στοιχειώδες ΤΝΔ Perceptron, το οποίο αποτελούσε έναν απλό αισθητήρα.
- Το 1969 οι Minsky και Papert απέδειξαν μαθηματικά ότι τα ΤΝΔ μίας στρώσης δεν έχουν την ικανότητα επίλυσης μη γραμμικών προβλημάτων.
- Το 1982 αποδείχθηκε μαθηματικά ότι οποιαδήποτε πληροφορία μπορεί να αποθηκευθεί σε ένα ΤΝΔ πολλαπλών στρώσεων.
- Το 1986 προτείνεται από τους Werbos και Rumelhart ο αλγόριθμος back propagation για την εκπαίδευση των ΤΝΔ.

2.2.3.3 Δέντρα απόφασης

Τα δέντρα απόφασης (decision trees) αποτελούν μία από τις πλέον διαδεδομένες μεθόδους κατηγοριοποίησης. Επί της ουσίας πρόκειται για γράφους με την κλασική δενδρική δομή στους οποίους διακρίνονται ένας αρχικός κόμβος (αναφερόμενος ως ρίζα), οι εσωτερικοί κόμβοι και οι εξωτερικοί κόμβοι (αναφερόμενοι και ως φύλλα). Σε όλους τους κόμβους εκτός από τη ρίζα εισέρχεται μία ακμή κατευθυνόμενη από κάποιον άλλο κόμβο. Στον κάθε εσωτερικό κόμβο αντιστοιχεί ένα χαρακτηριστικό το οποίο χρησιμοποιείται για τον περαιτέρω διαχωρισμό του δέντρου. Σε κάθε ακμή η οποία εξέρχεται από τη ρίζα ή κάποιον εσωτερικό κόμβο αντιστοιχεί μία συνθήκη ελέγχου βάσει του χαρακτηριστικού διαχωρισμού. Τα δέντρα απόφασης κατασκευάζονται βάσει μίας επαναληπτικής διαδικασίας η οποία έχει ως εξής: Αρχικά επιλέγεται ένα χαρακτηριστικό το οποίο αφορά τη ρίζα του δέντρου και στη συνέχεια κατασκευάζεται μία ακμή και ένας κόμβος για κάθε μία από τις διακριτές τιμές του χαρακτηριστικού. Τα βήματα αυτά επαναλαμβάνονται έως ότου κάθε χαρακτηριστικό να έχει εισαχθεί στους κόμβους του δέντρου [14].

Ο αλγόριθμος ID3 χρησιμοποιείται για τη δημιουργία ενός δέντρου απόφασης από ένα σύνολο δεδομένων. Ξεκινά με το αρχικό σύνολο S ως τον ριζικό κόμβο. Σε κάθε επανάληψη του αλγορίθμου, επαναλαμβάνεται για κάθε αχρησιμοποίητο χαρακτηριστικό του συνόλου S και υπολογίζεται η εντροπία H (S) ή το κέρδος πληροφοριών IG (S) αυτού του χαρακτηριστικού. Στη συνέχεια επιλέγεται το χαρακτηριστικό που έχει τη μικρότερη τιμή εντροπίας (ή μεγαλύτερο κέρδος πληροφοριών). Στη συνέχεια, το σύνολο S διαιρείται ή χωρίζεται από το επιλεγμένο χαρακτηριστικό ούτως ώστε να παραχθούν υποσύνολα των δεδομένων. Για

παράδειγμα, ένας κόμβος μπορεί να χωριστεί σε θυγατρικούς κόμβους με βάση τα υποσύνολα του πληθυσμού των οποίων οι ηλικίες είναι μικρότερες από 50, μεταξύ 50 και 100 και μεγαλύτερες από 100. Ο αλγόριθμος συνεχίζει να επαναλαμβάνεται σε κάθε υποσύνολο, λαμβάνοντας υπόψη μόνο τα χαρακτηριστικά που δεν έχουν ποτέ επιλεχθεί πριν.

Η επανάληψη σε ένα υποσύνολο μπορεί να σταματήσει σε μία από τις παρακάτω περιπτώσεις:

- Κάθε στοιχείο του υποσυνόλου ανήκει στην ίδια κλάση. Σε αυτήν την περίπτωση ο κόμβος μετατρέπεται σε έναν κόμβο φύλλων και επισημαίνεται με την κατηγορία των παραδειγμάτων.
- Δεν υπάρχουν άλλα χαρακτηριστικά που πρέπει να επιλεγούν, αλλά τα παραδείγματα εξακολουθούν να μην ανήκουν στην ίδια κλάση. Σε αυτήν την περίπτωση, ο κόμβος γίνεται ένας κόμβος φύλλων και επισημαίνεται με την πιο κοινή κατηγορία παραδειγμάτων στο υποσύνολο.
- Δεν υπάρχουν παραδείγματα στο υποσύνολο, κάτι το οποίο συμβαίνει όταν κανένα παράδειγμα στο γονικό σύνολο δεν έχει βρεθεί να ταιριάζει με μία συγκεκριμένη τιμή του επιλεγμένου χαρακτηριστικού. Ένα παράδειγμα θα μπορούσε να είναι η απουσία ενός ατόμου μεταξύ του πληθυσμού με ηλικία άνω των 100 ετών. Στη συνέχεια, δημιουργείται και επισημαίνεται ένας κόμβος φύλλων με την πιο κοινή κλάση των παραδειγμάτων στο σύνολο του γονικού κόμβου.

Σε όλο τον αλγόριθμο, το δέντρο αποφάσεων κατασκευάζεται με κάθε μη τερματικό κόμβο (εσωτερικό κόμβο) που αντιπροσωπεύει το επιλεγμένο χαρακτηριστικό στο οποίο χωρίστηκαν τα δεδομένα και τερματικούς κόμβους (κόμβους φύλλων) που αντιπροσωπεύουν την ετικέτα κλάσης του τελικού υποσυνόλου αυτού του κλάδου.

Ο αλγόριθμος μπορεί να συνοψιστεί ως εξής:

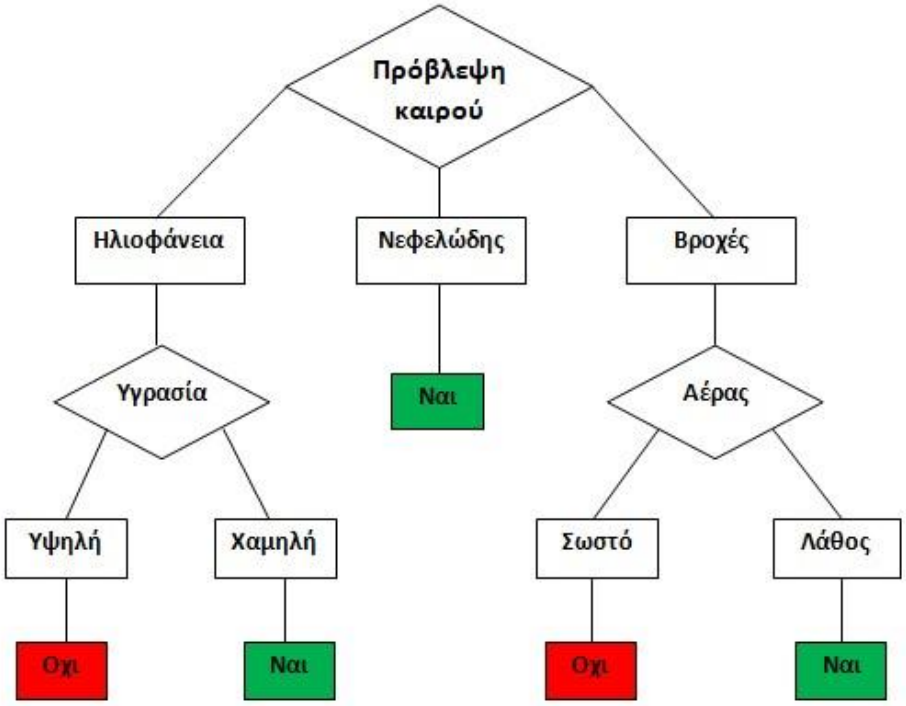
1. Υπολογίζεται η εντροπία κάθε χαρακτηριστικού *a* του συνόλου δεδομένων *S*.
2. Διαχωρίζεται το σύνολο *S* σε υποσύνολα χρησιμοποιώντας το χαρακτηριστικό για το οποίο ελαχιστοποιείται η προκύπτουσα εντροπία μετά το διαχωρισμό ή ισοδύναμα μεγιστοποιείται το κέρδος πληροφοριών.

3. Δημιουργείται ένας κόμβος δέντρου αποφάσεων που περιέχει αυτό το χαρακτηριστικό.
4. Η διαδικασία επαναλαμβάνεται στα υποσύνολα χρησιμοποιώντας τα υπόλοιπα χαρακτηριστικά.

Ο αλγόριθμος ID3:

- Δεν εγγυάται την εξεύρεση της βέλτιστης λύσης. Μπορεί να συγκλίνει με τοπικά βέλτιστα. Χρησιμοποιεί μία άπληστη στρατηγική επιλέγοντας το τοπικά καλύτερο χαρακτηριστικό για να χωρίσει το σύνολο δεδομένων σε κάθε επανάληψη. Η απόδοση του αλγορίθμου μπορεί να βελτιωθεί χρησιμοποιώντας οπισθοδρόμηση κατά την αναζήτηση του βέλτιστου δέντρου αποφάσεων με κόστος πιθανώς περισσότερο χρόνο.
- Ενδέχεται να υπερ-αντιστοιχίσει τα δεδομένα εκπαίδευσης. Για να αποφευχθεί αυτό, τα μικρότερα δέντρα αποφάσεων θα πρέπει να προτιμώνται από τα μεγαλύτερα. Ο αλγόριθμος παράγει συνήθως μικρά δέντρα, αλλά δεν παράγει πάντα το μικρότερο δυνατό δέντρο αποφάσεων.
- Είναι πιο δύσκολο να χρησιμοποιηθεί σε συνεχή δεδομένα από ό,τι σε δεδομένα με συντελεστή (τα παραγοντικά δεδομένα έχουν έναν διακριτό αριθμό πιθανών τιμών, μειώνοντας έτσι τα πιθανά σημεία διακλάδωσης). Εάν οι τιμές οποιουδήποτε δεδομένου χαρακτηριστικού είναι συνεχείς, τότε υπάρχουν πολλά περισσότερα μέρη για διαχωρισμό των δεδομένων σε αυτό το χαρακτηριστικό και η αναζήτηση της βέλτιστης τιμής για διαχωρισμό ενδέχεται να είναι χρονοβόρα.

Ο αλγόριθμος ID3 χρησιμοποιείται εφαρμόζοντας εκπαίδευση σε ένα σύνολο δεδομένων S για την παραγωγή ενός δέντρου αποφάσεων που είναι αποθηκευμένο στη μνήμη. Κατά το χρόνο εκτέλεσης, αυτό το δέντρο αποφάσεων χρησιμοποιείται για την ταξινόμηση νέων δοκιμαστικών περιπτώσεων (διανύσματα χαρακτηριστικών) διασχίζοντας το δέντρο αποφάσεων χρησιμοποιώντας τα χαρακτηριστικά του σημείου δεδομένων ώστε να φτάσει σε έναν κόμβο φύλλων. Η κλάση αυτού του τερματικού κόμβου είναι η κλάση στην οποία η δοκιμαστική περίπτωση ταξινομείται.



Εικόνα 8: Δέντρο απόφασης το οποίο αναπαριστά τις καιρικές συνθήκες για να αποφασίσει κάποιος να παίξει ποδόσφαιρο.

2.2.3.4 Απλοϊκός αλγόριθμος Bayes

Στην κατηγοριοποίηση προβλημάτων μάθησης, δίδεται ένα σύνολο παραδειγμάτων εκπαίδευσης και οι αντίστοιχες ετικέτες κλάσης και στην έξοδο δίδεται ένας κατηγοριοποιητής (classifier). Ο κατηγοριοποιητής παίρνει ένα παράδειγμα χωρίς ετικέτα και το εκχωρεί σε μία κλάση. Πολλοί κατηγοριοποιητές μπορούν να θεωρηθούν ως ο υπολογισμός ενός συνόλου διακριτικών συναρτήσεων του παραδείγματος, μία για κάθε κλάση και η εκχώρηση του παραδείγματος στην κλάση της οποίας η συνάρτηση μεγιστοποιείται. Αν το E είναι ένα παράδειγμα, και $f_i(E)$ είναι η διακριτική συνάρτηση που αντιστοιχεί στην i-οστή κλάση, η επιλεγμένη κλάση C_k είναι εκείνη για την οποία ισχύει [27]:

$$f_k(E) > f_i(E) \forall\, i \neq k \quad (1)$$

Ας υποθέσουμε ότι ένα παράδειγμα είναι ένα διάνυσμα α χαρακτηριστικών, όπως συμβαίνει συνήθως στις εφαρμογές κατηγοριοποίησης. Έστω v_{jk} η τιμή του χαρακτηριστικού A_j στο παράδειγμα, το $P(X)$ υποδηλώνει την πιθανότητα των X, και

το *P(Y|X)* υποδηλώνει την υπό συνθήκη πιθανότητα του Y δεδομένου του X. Στη συνέχεια, ένα πιθανό σύνολο διακριτικών συναρτήσεων είναι το παρακάτω [27]:

$$f_i(E) = P(C_i) \prod_{j=1}^{a} P\left(A_j = v_{jk} \,|C_i\right) (2)$$

Ο κατηγοριοποιητής που αποκτήθηκε χρησιμοποιώντας αυτό το σύνολο διακριτικών συναρτήσεων και εκτιμώντας τις σχετικές πιθανότητες από το σύνολο εκπαίδευσης, συχνά ονομάζεται αφελής Bayesian κατηγοριοποιητής (naive Bayesian classifier). Αυτό συμβαίνει επειδή, εάν πραγματοποιηθεί η «αφελής» υπόθεση ότι τα χαρακτηριστικά είναι ανεξάρτητα της κλάσης, αυτός ο κατηγοριοποιητής μπορεί εύκολα να αποδειχθεί βέλτιστος, με την έννοια της ελαχιστοποίησης του ποσοστού εσφαλμένης ταξινόμησης ή μηδενικής απώλειας, με άμεση εφαρμογή του θεωρήματος του Bayes, ως εξής: Εάν *P(Ci|E)* είναι η πιθανότητα το παράδειγμα *E* της κλάσης C_i, η μηδενική απώλεια ελαχιστοποιείται εάν, και μόνο εάν, το *E* έχει αντιστοιχηθεί στην κλάση C_k για την οποία το *P(Ck|E)* μεγιστοποιείται. Με άλλα λόγια, η χρήση των *P(Ci|E)* ως διακριτικών συναρτήσεων $f_i(E)$ είναι η βέλτιστη διαδικασία κατηγοριοποίησης. Από το θεώρημα του Bayes,

$$P(C_i|E) = \frac{P(C_i)P(E|C_i)}{P(E)}$$

το *P(E)* μπορεί να αγνοηθεί, καθώς είναι το ίδιο για όλες τις κλάσεις και δεν επηρεάζει τις σχετικές τιμές των πιθανοτήτων τους. Εάν τα χαρακτηριστικά είναι ανεξάρτητα δεδομένης της κλάσης, $P(E|C_i)$ μπορεί να αναλυθεί στο γινόμενο $P(A_1 = v_{1k}|C_i)...P(A_a = v_{ak}|C_i)$, οδηγώντας στο $P(C_i|E) = f_i(E)$ [27].

Στην πράξη, τα χαρακτηριστικά σπάνια είναι ανεξάρτητα δεδομένης της κλάσης, ως εκ τούτου και αυτή η υπόθεση είναι «αφελής». Ωστόσο, τίθεται το ερώτημα εάν ο Bayesian κατηγοριοποιητής μπορεί να είναι βέλτιστος ακόμη και όταν η υπόθεση της ανεξαρτησίας των χαρακτηριστικών δεν ισχύει, και ως εκ τούτου $P(Ci|E) \neq f_i(E)$. Σε αυτές τις συνθήκες, ο Bayesian κατηγοριοποιητής δεν μπορεί πλέον να υποστηριχθεί ότι υπολογίζει τις πιθανότητες κλάσεων που δίδονται στο παράδειγμα [27].

2.2.2.5 Ο αλγόριθμος k-nearest neighbors

Ο αλγόριθμος k-nearest neighbors αποτελεί έναν εκ των απλούστερων αλγορίθμων μηχανικής μάθησης. Παρά την απλότητα του είναι σε θέση να παρέχει ικανοποιητικές λύσεις σε μία σειρά προβλημάτων. Η είσοδος του αλγορίθμου αποτελείται από δεδομένα τα οποία αντιπροσωπεύουν το σύνολο εκπαίδευσης (training set) καθώς και από εκείνα τα οποία αντιπροσωπεύουν το σύνολο ελέγχου (testing set) στον ν-διάστατο χώρο του προβλήματος [28].

Κατά την εφαρμογή του αλγορίθμου σε προβλήματα κατηγοριοποίησης η έξοδος του αλγορίθμου παρέχει την κλάση στην οποία ανήκει το σημείο του συνόλου εκπαίδευσης. Η κατηγοριοποίηση του σημείου του συνόλου εκπαίδευσης στην κατάλληλη κλάση λαμβάνει χώρα μέσα από τον έλεγχο ανάμεσα στα k πλησιέστερα σε αυτό σημεία του συνόλου εκπαίδευσης της κλάσης πλειοψηφίας, δηλαδή εκείνης στην οποία ανήκουν τα περισσότερα σημεία [28].

Το σύνολο ελέγχου του αλγορίθμου αποτελείται από διανύσματα του ν-διάστατου χώρου συνοδευόμενα από την τιμή της κλάσης. Εάν το σύνολο των τιμών των κλάσεων ισούται με C και το σύνολο εκπαίδευσης ισούται με m έχουμε τις δυάδες [28]:

$$(X_1, Y_1), (X_2, Y_2), \dots, (X_m, Y_m) \in R^n xC$$

Έστω x ένα σημείο του συνόλου ελέγχου και εάν υπάρχει κατάλληλη μετρική της απόστασης ||.|| του ν-διάστατου χώρου του προβλήματος ο αλγόριθμος αναζητά τα k πλησιέστερα διανύσματα του συνόλου εκπαίδευσης

$$\left\|X_{(1)} - X\right\| \leq \cdots \left\|X_{(k)} - X\right\|$$

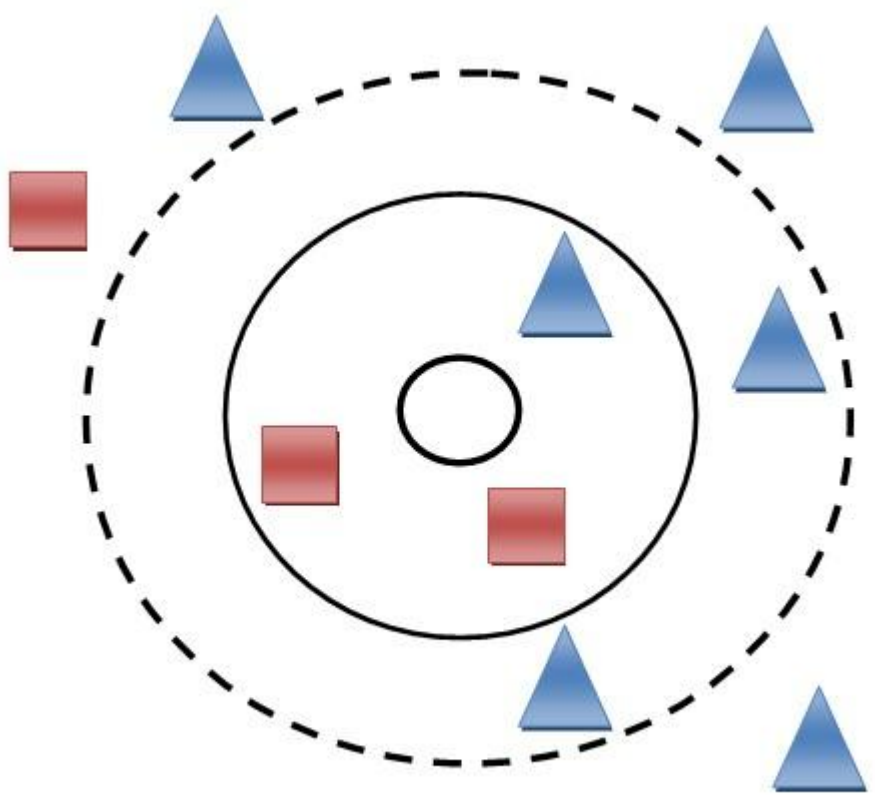
και έπειτα αναθέτει στο σημείο x την πλέον πολυάριθμη κλάση από τις $X_{(1)},\dots,X_{(k)}$.

Η πλέον διαδεδομένη μετρική της απόστασης στους χώρους συνεχών μεταβλητών είναι η Ευκλείδεια. Στην περίπτωση κατά την οποία το σύνολο αποτελείται από διακριτές τιμές χρησιμοποιούνται μετρικές όπως η Hamming [28].

Ο παραπάνω αλγόριθμος μπορεί να τροποποιηθεί με την ανάθεση βαρών στους k κοντινότερους γείτονες ώστε αυτοί να συνεισφέρουν βάσει του πόσο κοντά βρίσκονται στο σημείο που πρόκειται να ταξινομηθεί [28].

Η εκλογή της παραμέτρου k λαμβάνει χώρα βάσει των δεδομένων που πρόκειται να επεξεργαστούν. Γενικά, υψηλές τιμές του k έχουν ως αποτέλεσμα τη μείωση του θορύβου αλλά και της ακρίβειας καθώς τα όρια ανάμεσα στις κλάσεις γίνονται δυσδιάκριτα. Υπάρχουν διάφορες τεχνικές για την εύρεση καλών τιμών για το k. Στην περίπτωση κατά την οποία η επιλογή πρόκειται να λάβει χώρα ανάμεσα σε δύο μόνο κλάσεις η εκλογή περιττού k μπορεί να συμβάλλει στην εξάλειψη των ισοπαλιών [28].

Στην Εικόνα 9 παρουσιάζονται δύο παραδείγματα του αλγορίθμου. Και στις δύο εφαρμογές πρέπει να ταξινομηθεί το σημείο στο κέντρο που παριστάνεται από έναν κύκλο. Για k=3 τα πλησιέστερα σημεία περιέχονται στον κύκλο με τη συνεχή διαγράμμιση ενώ τα περισσότερα από αυτά έχουν κόκκινο χρώμα. Ως εκ τούτου θα κατηγοριοποιηθεί στην κόκκινη κλάση. Στη δεύτερη εφαρμογή έχουμε k=5 ενώ τα πέντε πλησιέστερα σημεία περιέχονται στον κύκλο με τη διακεκομμένη διαγράμμιση. Τα περισσότερα σημεία όμως έχουν μπλε χρώμα, ως εκ τούτου το σημείο θα κατηγοριοποιηθεί στη μπλε κλάση [28].



Εικόνα 9: Παράδειγμα κατηγοριοποίησης με τον αλγόριθμο k-nearest neighbors.

Στην απλούστερη εκδοχή του αλγορίθμου πρέπει να υπολογιστεί ο πίνακας των αποστάσεων ανάμεσα στα σύνολα εκπαίδευσης και ελέγχου. Έστω X ο πίνακας των σημείων εκπαίδευσης και Y ο πίνακας των σημείων ελέγχου. Αν το πρώτο σύνολο έχει πλήθος στοιχείων m και το δεύτερο r και ο χώρος του προβλήματος διάσταση n, οι διαστάσεις των πινάκων θα είναι X[m][n], Y[r][n]. Η απόσταση ανάμεσα στα σημεία X_i και Y_j ορίζεται ως:

$$D_{ij} = ||X_i - Y_j||.$$

Ο αλγόριθμος απαιτεί μόνο τη σχετική ταξινόμηση των αποστάσεων, επομένως από τη στιγμή που χρησιμοποιείται η ευκλείδεια απόσταση ως μετρική το τετράγωνο της απόστασης υπολογίζεται ως εξής [28]:

$$D_{ij} = \|X_i - Y_j\|^2 = X_i X_i^T - 2X_i Y_j^T + Y_j Y_j^T \Rightarrow$$

$$D = \underbrace{\begin{bmatrix} X_0 X_0^T & X_0 X_0^T \dots & X_0 X_0^T \\ X_1 X_1^T & X_1 X_1^T \dots & X_1 Y_1^T \\ \vdots & & \\ X_{m-1} X_{m-1}^T & X_{m-1} X_{m-1}^T \dots & X_{m-1} X_{m-1}^T \end{bmatrix}}_{Array dimension m \times r} - 2XY^T + \underbrace{\begin{bmatrix} Y_0 Y_0^T & Y_1 Y_1^T \dots & Y_{r-1} Y_{r-1}^T \\ Y_0 Y_0^T & Y_1 Y_1^T \dots & Y_{r-1} Y_{r-1}^T \\ \vdots & & \\ Y_0 Y_0^T & Y_1 Y_1^T \dots & Y_{r-1} Y_{r-1}^T \end{bmatrix}}_{Array dimension m \times r}$$

Εικόνα 10: Πίνακας πρόσθεσης τριών μερών.

Ο παραπάνω πίνακας είναι το αποτέλεσμα της πρόσθεσης τριών μερών. Το πρώτο και το τρίτο μέρος απαιτούν mκαι rεσωτερικά γινόμενα διανυσμάτων αντίστοιχα ή m*n και r*n βαθμωτά γινόμενα. Το πλέον χρονοβόρο μέρος των υπολογισμών είναι ο πολλαπλασιασμός των πινάκων X, Y^T, καθώς απαιτεί τον υπολογισμό m*r εσωτερικών γινομένων ή m*n*r βαθμωτών γινομένων [28].

Μία εναλλακτική είναι ο πολλαπλασιασμός με υποπίνακες. Οι πίνακες A και B διαιρούνται σε υποπίνακες ίδιου μεγέθους και έπειτα πολλαπλασιάζονται κατάληλα ώστε να προκύψει το τελικό γινόμενο. Η μαθηματική διατύπωση της συγκεκριμένης πράξης παρατίθεται στον επόμενο πίνακα, όπου $\mathbf{A_i}$ οι υποπίνακες [28]:

$$AB = \begin{bmatrix} \boldsymbol{A_1} \\ \boldsymbol{A_2} \\ \vdots \\ \boldsymbol{A_N} \end{bmatrix} \begin{bmatrix} \boldsymbol{B_1} & \boldsymbol{B_2} \dots \boldsymbol{B_M} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A_1 B_1} & \boldsymbol{A_1 B_2} \dots \boldsymbol{A_1 B_M} \\ \boldsymbol{A_2 B_1} & \boldsymbol{A_2 B_2} \dots \boldsymbol{A_2 B_M} \\ & \ddots \\ \boldsymbol{A_N B_1} & \boldsymbol{A_N B_2} \dots \boldsymbol{A_N B_M} \end{bmatrix}$$

Εικόνα 11: Πολλαπλασιασμός με υποπίνακες.

2.3 Μάθηση χωρίς επίβλεψη

Στη μάθηση χωρίς επίβλεψη το εκπαιδευόμενο σύστημα ανακαλύπτει από μόνο του τις συσχετίσεις ή ομάδες ενός συνόλου δεδομένων δημιουργώντας πρότυπο χωρίς να γνωρίζει εκ των προτέρων εάν υπάρχουν συσχετίσεις ή ομάδες, ποια είναι τα δεδομένα και πιο το πλήθος τους [15].

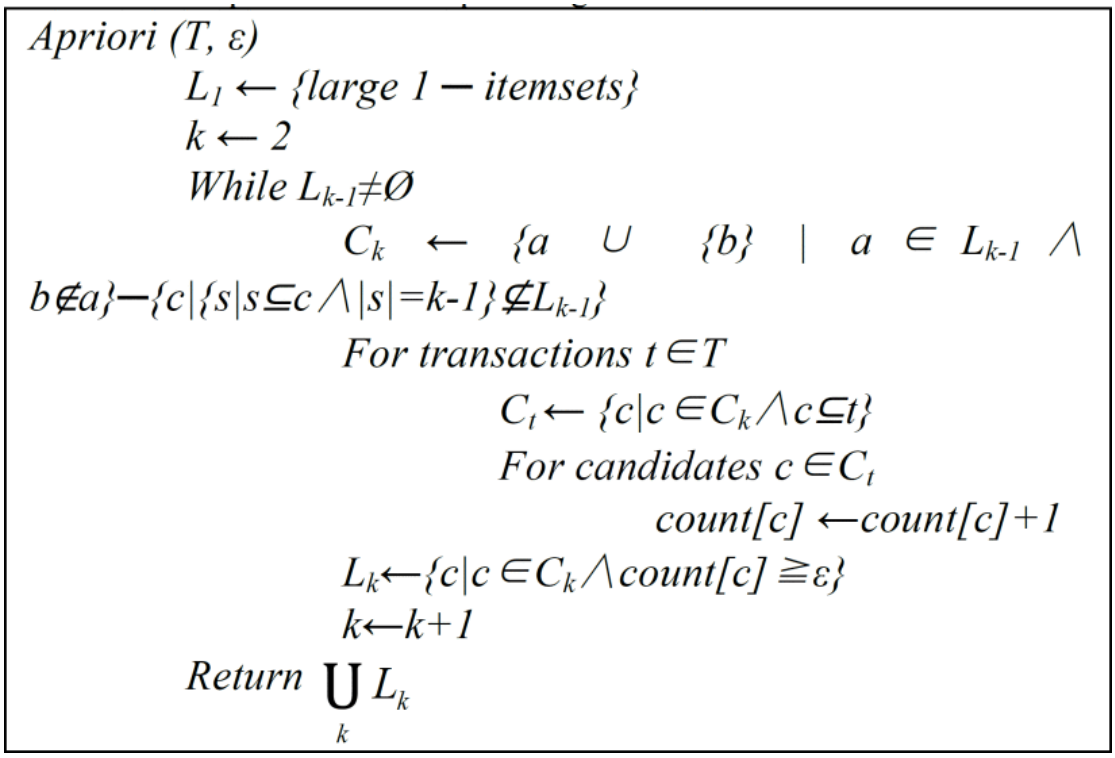
2.3.1 Συσχέτιση

Η εκμάθηση με κανόνες συσχέτισης είναι μια μέθοδος της μηχανικής μάθησης βασισμένη σε κανόνες, που ως σκοπό έχει, την ανακάλυψη ενδιαφερουσών σχέσεων μεταξύ των μεταβλητών σε μεγάλες βάσεις δεδομένων.

2.3.1.1 Ο αλγόριθμος Apriori

Ο αλγόριθμος Apriori αναπτύχθηκε για την εύρεση συχνοτήτων αντικειμένων και κανόνων σύνδεσης από σύνολα δεδομένων συναλλαγών (transactions datasets). Ο αλγόριθμος λειτουργεί δημιουργώντας αρχικά υποψήφια (candidate) σύνολα αντικειμένων μήκους k από σύνολα αντικειμένων μήκους k-1 χρησιμοποιώντας μία αναζήτηση breadth-first και μία δομή δέντρου κατακερματισμού για τη μέτρηση των υποψηφίων συνόλων αντικειμένων και στη συνέχεια προσαρμόζει τις υποψηφιότητες που έχουν σπάνια υποτμήματα έως ότου το υποψήφιο σύνολο περιέχει όλα τα συχνά σύνολα στοιχείων k-μήκους. Έπειτα η βάση δεδομένων συναλλαγών σαρώνεται για τον προσδιορισμό συχνών συνόλων αντικειμένων μεταξύ των υποψηφίων [16].

Ο ψευδοκώδικας για τον αλγόριθμο δίνεται παρακάτω για μία βάση δεδομένων συναλλαγών K και ένα κατώφλι υποστήριξης ε [16]:



Εικόνα 12 :Η αρχή του αλγορίθμου Apriori.

Τα συχνά αντικείμενα θα μπορούσαν να φιλτραριστούν εκτελώντας τον παραπάνω αλγόριθμο, αλλά προκειμένου να προσδιοριστούν κανόνες από ένα σύνολο όλων των πιθανών κανόνων χρησιμοποιούνται αρκετοί περιορισμοί σε διαφορετικά μέτρα σημασίας, και ειδικότερα η ελάχιστη υποστήριξη (min_sup) και εμπιστοσύνη (min_con) [16].

Η υποστήριξη είναι ένας δείκτης σχετικά με το πόσο συχνά εμφανίζεται το σύνολο αντικειμένων στο σύνολο δεδομένων. Η υποστήριξη του X σε σχέση με το T ορίζεται ως η ποσότητα των συναλλαγών t στο σύνολο δεδομένων, το οποίο περιέχει το σύνολο αντικειμένων X, σύμφωνα με την επόμενη σχέση [16]:

$$sup(X) = \frac{|\{t \in T; X \in t\}|}{|T|}$$

Η εμπιστοσύνη είναι μία ένδειξη του πόσο συχνά ο κανόνας έχει βρεθεί αληθής. Η τιμή εμπιστοσύνης ενός κανόνα, X⇒Y, σε σχέση με ένα σύνολο συναλλαγών T, είναι η ποσότητα των συναλλαγών που περιέχουν το X που περιέχουν επίσης το Y, όπου η εμπιστοσύνη ορίζεται ως [16]:

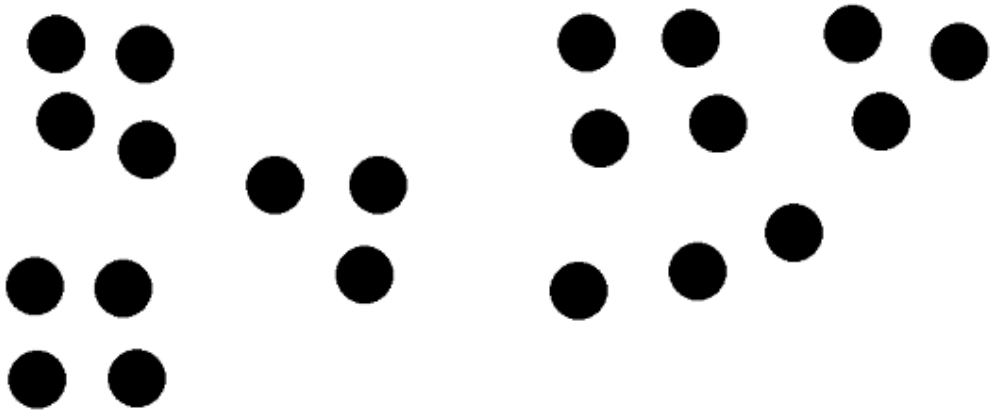
$$Conf(X \Rightarrow Y) = sup(XUY)/sup(X)$$

2.3.2 Ομαδοποίηση

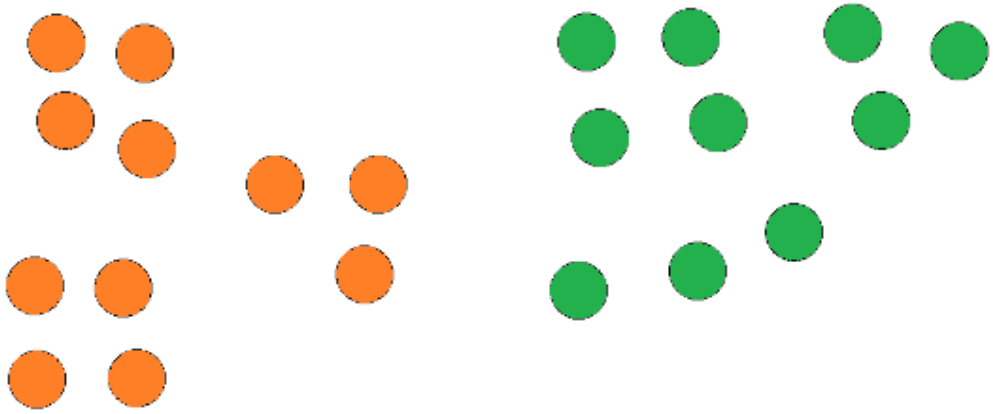
Η ομαδοποίηση (clustering) αποτελεί τη διαδικασία οργάνωσης των στοιχείων ενός συνόλου δεδομένων σε ομάδες βάσει ενός μέτρου ομοιότητας. Οι κύριοι στόχοι των αλγορίθμων ομαδοποίησης καθορίζονται από αυτήν την προσέγγιση και έχουν ως εξής:

- Η κάθε ομάδα αντικειμένων που δημιουργείται από το σύνολο των δεδομένων να αποτελείται από όμοια αντικείμενα, δηλαδή να είναι ομοιογενής.
- Τα αντικείμενα της κάθε ομάδας να είναι ανόμοια με εκείνα οποιασδήποτε άλλης ομάδας.

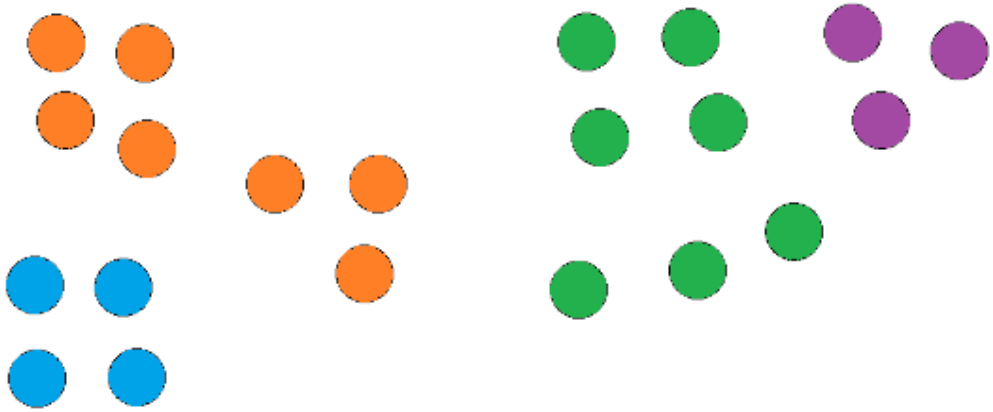
Η ομαδοποίηση είναι μία δύσκολη διαδικασία. Η δυσκολία απορρέει από το γεγονός του ότι δεν υπάρχει κάποιος συγκεκριμένος ορισμός σχετικά με το τι αποτελεί μία ομάδα αλλά και το πώς θα μπορούσε να προκύψει η βέλτιστη λύση. Το πρόβλημα αυτό μπορεί να αντιμετωπιστεί στην περίπτωση κατά την οποία ο χρήστης έχει ήδη κάποια γνώση σχετικά με τη φύση των αντικειμένων που αποτελούν το σύνολο δεδομένων που εξετάζεται και ως εκ τούτου μπορούν να παραχθούν αποδοτικές λύσεις και να καταστεί ευκολότερη η ανάλυση τους. Το πρόβλημα αυτό μπορεί να γίνει αντιληπτό με την εξέταση των Εικόνων 11-14. Στην Εικόνα 11 έχουμε ένα σύνολο δεδομένων το οποίο απαρτίζεται από 20 σημεία τα οποία αντιπροσωπεύουν τα αντικείμενα της εισόδου πριν την ομαδοποίηση τους. Στις Εικόνες 12-14 παρουσιάζονται διάφορες λύσεις. Παρατηρούμε ότι ο αριθμός των ομάδων είναι κάθε φορά διαφορετικός βάσει της προσέγγισης που ακολουθήθηκε κατά τον ορισμό των ομάδων με την κάθε ομάδα να αναπαρίσταται με διαφορετικό χρωματισμό. Όλες οι λύσεις θεωρούνται σωστές, ως εκ τούτου η ερμηνεία τους είναι εξίσου σημαντική με την ομαδοποίηση.



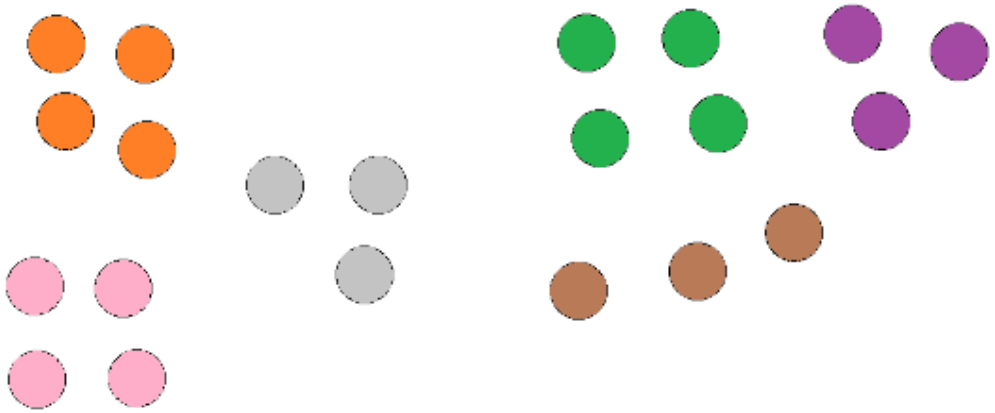
Εικόνα 13: Αρχικό σύνολο δεδομένων εισόδου.



Εικόνα 14: Ομαδοποίηση δύο ομάδων.



Εικόνα 15: Ομαδοποίηση τεσσάρων ομάδων.



Εικόνα 16: Ομαδοποίηση έξι ομάδων.

Οι παρατηρήσεις συνήθως παρουσιάζονται ως διανύσματα συνεχών ή τιμών. Η δυσδιάστατη ή τρισδιάστατη αναπαράσταση τους στον χώρο ως σημεία αποτελεί μία προσέγγιση η οποία διευκολύνει την ανθρώπινη εποπτεία και ερμηνεία των αποτελεσμάτων. Βέβαια σε διάφορες εφαρμογές δεν είναι δυνατή η χρήση των δεδομένων στην αρχική τους μορφή λόγω της πολυπλοκότητας τους ή της μεγάλης τους διάστασης, γεγονός το οποίο καθιστά απαραίτητη την εφαρμογή διαφόρων μετασχηματισμών, όπως είναι η απομάκρυνση του θορύβου στο στάδιο της προεπεξεργασίας. Σχετικά παραδείγματα αποτελούν τα εικονοστοιχεία μίας εικόνας ή οι συμβολοσειρές ενός κειμένου.

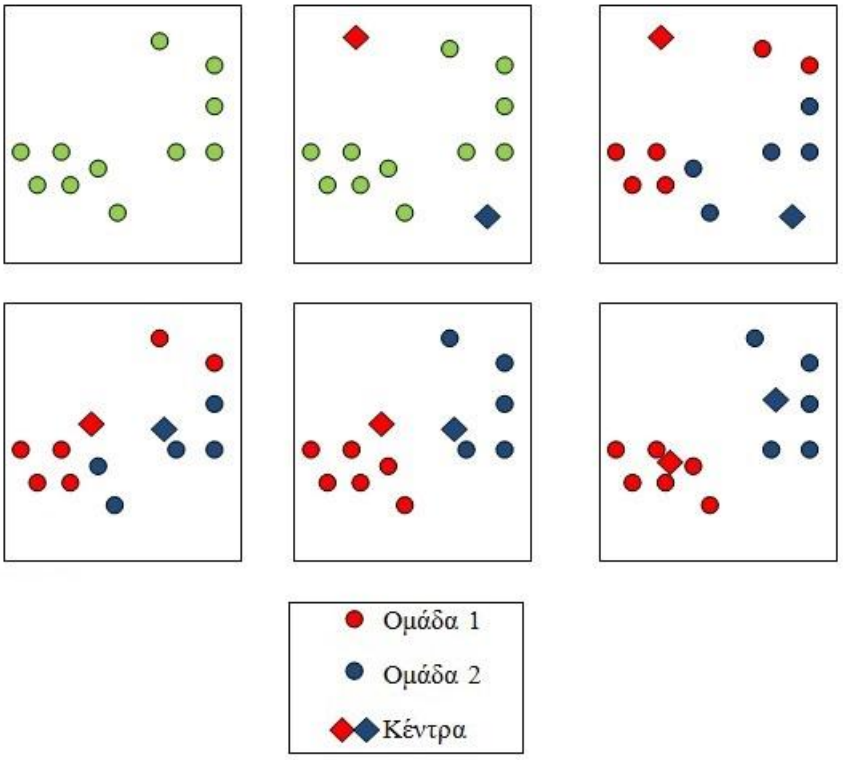
Η κατηγοριοποίηση και η ομαδοποίηση παρουσιάζουν μία σειρά ομοιοτήτων. Η ομαδοποίηση θα μπορούσε να θεωρηθεί ως ένα είδος κατηγοριοποίησης το οποίο έχει τη δυνατότητα αυτόματης δημιουργίας των κλάσεων. Η διαφορά με την κατηγοριοποίηση εδώ πέραν συνίσταται στο ότι η τελευταία απαιτεί την προγενέστερη κατηγοριοποίηση των αντικειμένων της εισόδου ώστε να ταξινομηθούν έπειτα τα νέα άγνωστα αντικείμενα στις σωστές κλάσεις. Ως εκ τούτου η ομαδοποίηση μπορεί να θεωρηθεί μία διαδικασία μη εποπτευόμενης κατηγοριοποίησης [12].

2.3.2.1 Ο αλγόριθμος K-means

Η ομαδοποίηση k-means είναι μια μέθοδος που χρησιμοποιείται συχνά για τον αυτόματο διαχωρισμό ενός συνόλου δεδομένων σε k-ομάδες. Λαμβάνει χώρα επιλέγοντας k αρχικά κέντρα συστάδων και στη συνέχεια επαναπροσδιορίζοντάς τα επαναληπτικά ως εξής [17]:

1. Κάθε στιγμιότυπο d_i αντιστοιχίζεται στο κοντινότερο του κέντρο συστάδων.
2. Κάθε κέντρο συστάδων C_j ενημερώνεται ούτως ώστε να είναι ο μέσος των συστατικών του στιγμιότυπων.

Ο αλγόριθμος συγκλίνει όταν δεν υπάρχει περαιτέρω αλλαγή στην εκχώρηση στιγμιότυπων σε συστάδες [17].



Εικόνα 17: Ο αλγόριθμος k-means.

2.3.2.2 Ιεραρχική ομαδοποίηση

Οι αλγόριθμοι ιεραρχικής ομαδοποίησης παράγουν μία ιεραρχική αποσύνθεση των δεδομένων εισόδου. Διαχωρίζονται σε σωρευτικούς (agglomerative – bottom up) και διαιρετικούς (divisive – top down).

Οι σωρευτικοί αλγόριθμοι αρχικά αντιμετωπίζουν το κάθε στοιχείο ως μία ομάδα και στη συνέχεια συνενώνουν διαδοχικά τις ομάδες σύμφωνα με ένα μέτρο εγγύτητας. Η διαδικασία σταματά όταν όλα τα στοιχεία έχουν ενταχθεί σε μία ομάδα ή διαφορετικά σύμφωνα με τις επιλογές του χρήστη. Η προσέγγιση που ακολουθείται σε τέτοιου

είδους εφαρμογές συνήθως είναι η άπληστη συγχώνευση (greedy-like bottom up merging).

Στους διαιρετικούς αλγορίθμους εφαρμόζεται η αντίθετη προσέγγιση. Αρχικά υπάρχει μία ομάδα η οποία περιλαμβάνει όλα τα αντικείμενα της εισόδου. Η ομάδα αυτή στη συνέχεια διαχωρίζεται σε μικρότερες μέχρι τη στιγμή που το κάθε αντικείμενο θα αποτελεί μία ομάδα ή διαφορετικά σύμφωνα με κριτήρια που έχουν εισαχθεί από τον χρήστη. Τα δεδομένα των αντικειμένων σε κάθε βήμα διαχωρίζονται σε ομάδες χωρίς κοινά στοιχεία μέχρις ότου όλα τα στοιχεία να βρίσκονται σε διαφορετική ομάδα, με την όλη προσέγγιση να είναι παρόμοια με εκείνη των αλγορίθμων τύπου «διαίρει και βασίλευε» (divide-and-conquer).

Το σημαντικότερο μειονέκτημα των αλγορίθμων ιεραρχικής ομαδοποίησης είναι ότι στην περίπτωση εκτέλεσης μίας συγχώνευσης ή διαίρεσης δεν υπάρχει η δυνατότητα αναίρεσης ή βελτίωσης [13].

2.4 Το περιβάλλον R

Η R είναι μία γλώσσα προγραμματισμού ανοικτού κώδικα και αποτελεί ένα πολύ σημαντικό εργαλείο για την επίλυση προβλημάτων που σχετίζονται με την αριθμητική ανάλυση και με τη μηχανική μάθηση. Επιπροσθέτως. αποτελεί ένα περιβάλλον το οποίο δίνει τη δυνατότητα στους χρήστες παρέχει στους χρήστες τη να υλοποιήσουν εφαρμογές υπολογιστικής στατιστικής και σχεδίασης γραφημάτων. Η R δημιουργήθηκε από τους Ross Ihaka και Robert Gentleman στο πανεπιστήμιο University of Auckland της Νέας Ζηλανδίας αλλά στη συνέχεια συμμετείχαν και άλλοι στην εξέλιξη και ανάπτυξη της [2]. Παρουσιάζει αρκετές ομοιότητες με το περιβάλλον της γλώσσας προγραμματισμού S, η οποία είχε υλοποιηθεί στα εργαστήρια Bell Laboratories από τον John Chambers [3]. Παρέχει τα αναγκαία στατιστικά εργαλεία για τη διεξαγωγή στατιστικών αναλύσεων. Ορισμένα από αυτά είναι:

- Δημιουργία τυχαίων δειγμάτων

- Κατανομές διακριτών και συνεχών μεταβλητών (για παράδειγμα, Poisson, Gamma, Exponential).
- Έλεγχοι υποθέσεων
- Στατιστικά τεστ (Kolmogorov – Smirnov)
- Σχεδίαση γραφημάτων (όπως ιστογράμματα, qq-plots, pie-charts, bar-charts)

Εικόνα 18: Το λογότυπο της R.

2.4.1 Πλεονεκτήματα και μειονεκτήματα της R

Η R ως εξέλιξη της S μπορεί να επιτελέσει τις ίδιες λειτουργίες με την S με πολύ λιγότερο κώδικα. Εφόσον είναι λογισμικό ανοικτού κώδικα κάθε χρήστης έχει τη δυνατότητα επεξεργασίας του κώδικα της και βελτιστοποίησης τους. Επιπλέον, καθιστά δυνατή την αλληλεπίδραση με άλλες γλώσσες προγραμματισμού (C, C++, Java, Python), με φύλλα επεξεργασίας και βάσεις δεδομένων (Excel, Access), καθώς και με διάφορα στατιστικά πακέτα (SAS, Stata, SPSS, Minitab). Το περιβάλλον της R είναι δωρεάν διαθέσιμο [4].

Όσο αναφορά τα μειονεκτήματα της R, το συγκεκριμένο περιβάλλον δεν ενδείκνυται για ανάλυση μεγάλων δεδομένων εξαιτίας των υψηλών απαιτήσεων μνήμης. Επιπρόσθετα, η R δεν είναι ιδιαιτέρως αποδοτική γλώσσα σε ότι αφορά τον χρόνο εκτέλεσης των εντολών [5].

2.4.2 Εφαρμογές της R

Το περιβάλλον της R παρέχει περισσότερα από 15.000 πακέτα, γεγονός το οποίο έχει ως αποτέλεσμα την αξιοποίηση του από ένα ευρύ φάσμα επιστημονικών κλάδων. Εταιρείες όπως η Google, LinkedIn και Facebook χρησιμοποιούν τη συγκεκριμένη γλώσσα για εργασίες που σχετίζονται με την ανάλυση δεδομένων. Επιπρόσθετα εφαρμόζεται σε μία σειρά άλλων κλάδων, όπως είναι οι οικονομικές επιστήμες, η αστρονομία, η χημεία, η φαρμακευτική, η ιατρική, η εμπορευματική (μάρκετινγκ) κλπ. [4].

Κεφάλαιο 3^{ο} : Πειραματική Μελέτη

4.1 Περιγραφή μεθόδων και ορών

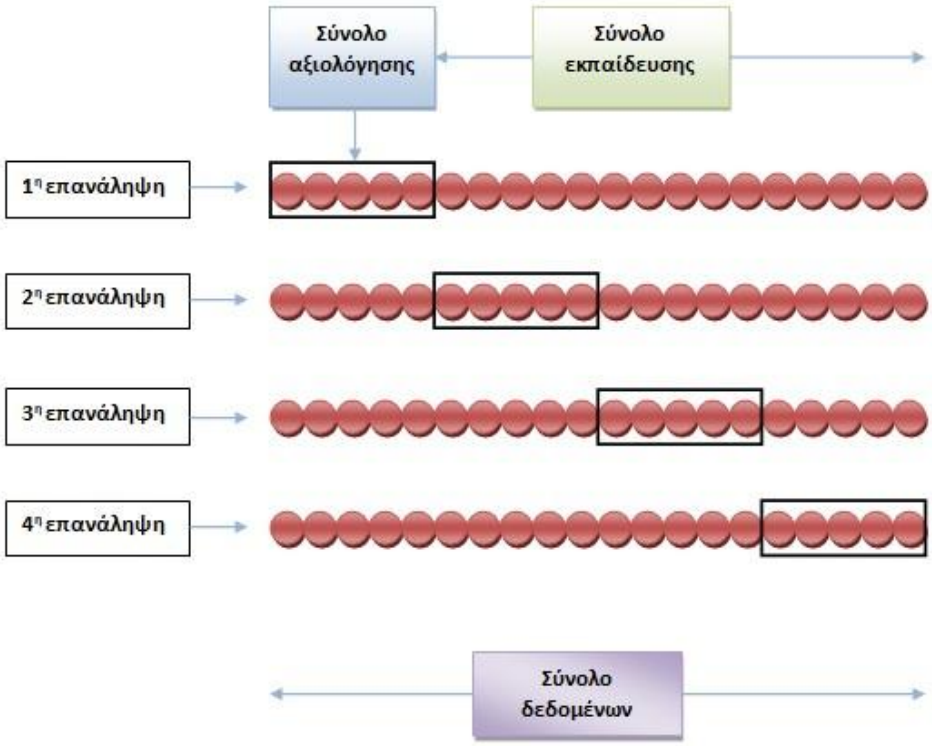
Για τη διεξαγωγή των παρακάτω πειραμάτων χρησιμοποιούνται συγκεκριμένες μέθοδοι και μετρικές, οι οποίες εξυπηρετούν στην επεξεργασία των συνόλων δεδομένων αλλά και στη μέτρηση της αποδοτικότητας των μοντέλων που θα δημιουργήσουμε.

Percentage split

Μέθοδος αξιολόγησης ταξινομητή. Διαχωρίζει το σύνολο δεδομένων σε υποσύνολο εκπαίδευσης και ταξινόμησης σύμφωνα με ένα ποσοστό.

K Cross Validation

Μέθοδος διαχωρισμού του συνόλου δεδομένων και αξιολόγησης του ταξινομητή . Το μέγεθος του συνόλου δεδομένων διαιρείται με έναν ακέραιο αριθμό k και το πηλίκο της διαίρεσης ορίζει το μέγεθος του υποσυνόλου αξιολόγησης, ενώ οι υπόλοιπες εγγραφές ορίζουν το υποσύνολο εκπαίδευσης. Η διαδικασία αυτή επαναλαμβάνεται k φορές και κάθε φορά επιλέγονται οι επόμενες N/k εγγραφές ως υποσύνολο αξιολόγησης και οι υπόλοιπες ως υποσύνολο εκπαίδευσης. Το αποτέλεσμα της παραπάνω διαδικασίας είναι ο αλγόριθμος να εκπαιδεύεται και να εξετάζεται με k διαφορετικά υποσύνολα και ως συνέπεια να παρέχει πιο ακριβείς μετρήσεις.



Εικόνα 19: Μέθοδος k cross validation για N=20 και k=4.

TF-IDF

Μέθοδος μετατροπής ενός συνόλου κειμένων σε διάνυσμα. Η ποσότητα tf υποδηλώνει το πόσες φορές εμφανίζεται ένας όρος σε ένα κείμενο, ενώ η ποσότητα idf υποδηλώνει το πόσο συχνά εμφανίζεται ένας όρος σε ολόκληρη τη συλλογή κειμένων.

$$TFIDF\ score\ for\ term\ i\ in\ document\ j = TF(i,j) * IDF(i)$$

where

$$IDF = Inverse\ Document\ Frequency$$

$$TF = Term\ Frequency$$

$$TF(i,j) = \frac{\text{Term i frequency in document j}}{\text{Total words in document j}}$$

$$IDF(i) = \log_2\left(\frac{\text{Total documents}}{\text{documents with term i}}\right)$$

and

$$t = Term$$

$$j = Document$$

Εικόνα 20: Ψευδοκώδικας αλγορίθμου TF-IDF.

Normalization

Η μέθοδος αυτή εφαρμόζει κανονικοποίηση στις τιμές των χαρακτηριστικών, δηλαδή τις μετατρέπει στο διάστημα [-1,1] με την εξίσωση:

$$Xji = \frac{Xji - \text{mean}(j)}{\max(j) - \min(j)}$$

Ακρίβεια και Ανάκληση

Η ακρίβεια και η ανάκληση είναι μετρικές που χρησιμοποιούνται για την σύγκριση των αλγορίθμων.

$$Ακρίβεια = \frac{(True\ Negative + True\ Positive)}{True\ Negative + True\ Positive + False\ Positive + False\ Negative}$$

$$Ανάκληση = \frac{True\ Positive}{(True\ Positive + False\ Negative)}$$

Ως true positive ορίζεται το πλήθος των προβλέψεων που ήταν 0 δεδομένου ότι η πραγματική τιμή ήταν 0.

Ως true negative ορίζεται το πλήθος των προβλέψεων που ήταν 1 δεδομένου ότι η πραγματική τιμή ήταν 1.

Ως false negative ορίζεται το πλήθος των προβλέψεων που ήταν 1 δεδομένου ότι η πραγματική τιμή ήταν 0.

Ως false positive ορίζεται το πλήθος των προβλέψεων που ήταν 0 δεδομένου ότι η πραγματική τιμή ήταν 1.

Ειδικότητα

Αντίστοιχα με την ακρίβεια και την ανάκληση, η ειδικότητα είναι μετρική που χρησιμοποιείται για την σύγκριση αλγορίθμων.

$$Ειδικότητα = \frac{True\ Negative}{(True\ Negative + False\ Positive)}$$

RMSE

Το RMSE είναι ένα από τα πιο συνηθισμένα μέτρα αξιολόγησης μοντέλων παλινδρόμησης. Το μέτρο αυτό μετράει τη διαφορά ανάμεσα στις πραγματικές yi και στις προβλεπόμενες τιμές $\hat{y}i$. Η διαφορά αυτή ονομάζεται κατάλοιπο ei όπου ei = $yi - \hat{y}i$.

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(yi - \hat{y}i)^2}{n}}$$

όπου n=σύνολο των παρατηρήσεων.

One -vs-All

Η μέθοδος one-vs-all χρησιμοποιείται για την εκπαίδευση ταξινομητών, όταν το σύνολο δεδομένων αποτελείται από περισσότερες από δύο κλάσεις. Η βασική ιδέα της μεθόδου είναι η δημιουργία τόσων ταξινομητών, όσο και ο αριθμός των κλάσεων. Σε κάθε ταξινομητή η κλάση που του αντιστοιχεί θα έχει την τιμή 1 και οι υπόλοιπες -1. Σημαντικό μειονέκτημα της μεθόδου αυτής είναι πως δημιουργούνται σύνολα δεδομένων που ο αριθμός των κλάσεων με τιμή -1 υπερισχύει αυτών με τιμή 1.

4.2 Διεξαγωγή πειραμάτων

Στο κεφάλαιο αυτό θα διεξαχθεί μια πειραματική μελέτη, η οποία έχει ως σκοπό να εξηγηθούν σαφέστερα οι λειτουργίες και οι χρησιμότητες των αλγορίθμων που περιγράφτηκαν στο προηγούμενο κεφάλαιο. Θα χρησιμοποιηθούν μερικά σύνολα δεδομένων τα οποία περιγράφονται παρακάτω και με τη βοήθεια του περιβάλλοντος R θα γίνει συγκριτική μελέτη μεταξύ των αλγορίθμων.

Πείραμα 1ο

Σύνολο δεδομένων: Body Fat [30]

Περιγραφή: Το σύνολο δεδομένων αφορά μετρήσεις της περιφέρειας σώματος για 252 άνδρες και εκτιμήσεις του ποσοστού του σωματικού λίπους τους, οι οποίες καθορίστηκαν από υποβρύχια ζύγιση .

BODY FAT	
ΓΡΑΜΜΕΣ	252
ΣΤΗΛΕΣ	15
ΤΙΜΗ ΣΤΟΧΟΣ	Body Fat (αριθμητική)

1) **Density** - Πυκνότητα σώματος (υπολογισμένη από υποβρύχια ζύγιση)
2) **Body Fat** - Ποσοστό σωματικού λίπους από την εξίσωση Siri (Body Fat = (495 / Body Density) - 450)
3) **Age** - Ηλικία
4) **Weight** - Βάρος (lbs)
5) **Height** - Ύψος (inches)
6) **Neck** - Περιφέρεια λαιμού (cm)
7) **Chest** - Περιφέρεια στήθους (cm)
8) **Abdomen** - Περιφέρεια κοιλίας (cm)
9) **Hip** - Περιφέρεια γοφού (cm)
10) **Thigh** - Περιφέρεια μηρού (cm)
11) **Knee** - Περιφέρεια γονάτου(cm)
12) **Ankle** - Περιφέρεια αστραγάλου(cm)
13) **Biceps** -Περιφέρεια δικέφαλου (cm)
14) **Forearm** - Περιφέρεια πήχη (cm)
15) **Wrist** - Περιφέρεια καρπού (cm)

Αλγόριθμος : Γραμμική παλινδρόμηση (Split percentage)

Αρχικά εισάγουμε το σύνολο δεδομένων στην R και στη συνέχεια μετατρέπουμε τα βάρη της στήλης Weight σε κιλά και τα ύψη της στήλης Height σε cm. Οι εντολές στην R είναι οι εξής:

```
>Body_Fat$Weight=Body_Fat$Weight*0.45359237
>Body_Fat$Height=Body_Fat$Height*2.54
```

Θα εξετάσουμε τα περιγραφικά στατιστικά των χαρακτηριστικών μας με την εντολή **summary().**

```
>summary(Body_Fat)
```

Density	Body Fat	Age	Weight	Height
Min. :0.995	Min. : 0.00	Min. :22.00	Min. : 53.75	Min. :162.6
1st Qu.:1.041	1st Qu.:12.47	1st Qu.:35.75	1st Qu.: 72.12	1st Qu.:173.4
Median :1.055	Median :19.20	Median :43.00	Median : 80.06	Median :177.8
Mean :1.055	Mean :19.16	Mean :44.88	Mean : 81.16	Mean :178.6
3rd Qu.:1.070	3rd Qu.:25.30	3rd Qu.:54.00	3rd Qu.: 89.36	3rd Qu.:183.5
Max. :1.109	Max. :47.50	Max. :81.00	Max. :164.72	Max. :197.5

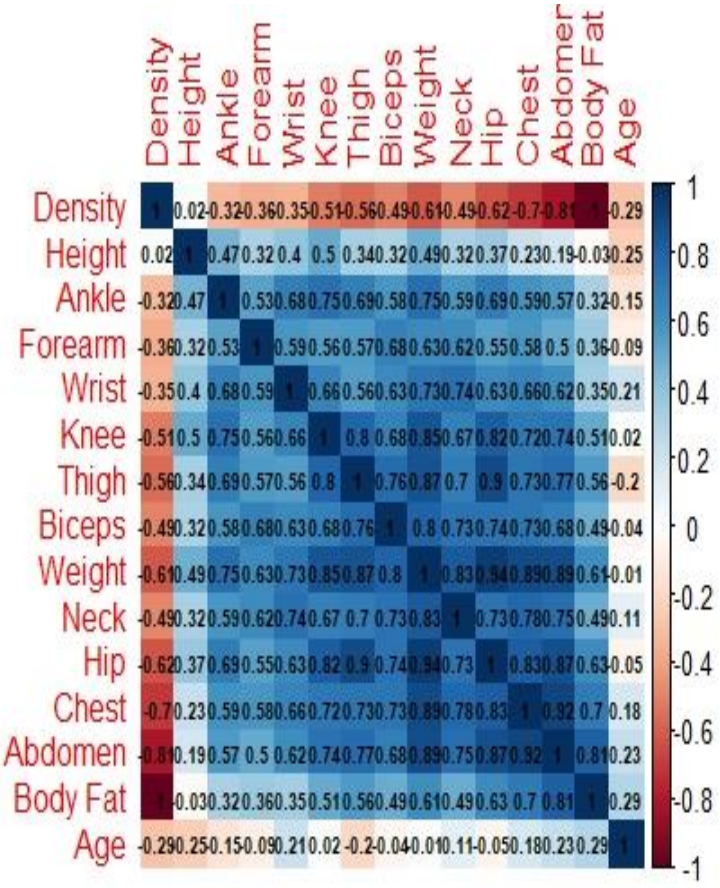
Neck	Chest	Abdomen	Hip	Thigh
Min. :31.10	Min. : 79.30	Min. : 69.40	Min. : 85.0	Min. :47.20
1st Qu.:36.40	1st Qu.: 94.35	1st Qu.: 84.58	1st Qu.: 95.5	1st Qu.:56.00
Median :38.00	Median : 99.65	Median : 90.95	Median : 99.3	Median :59.00
Mean :37.99	Mean :100.82	Mean : 92.56	Mean : 99.9	Mean :59.41
3rd Qu.:39.42	3rd Qu.:105.38	3rd Qu.: 99.33	3rd Qu.:103.5	3rd Qu.:62.35
Max. :51.20	Max. :136.20	Max. :148.10	Max. :147.7	Max. :87.30

Knee	Ankle	Biceps	Forearm	Wrist
Min. :33.00	Min. :19.10	Min. :24.80	Min. :21.00	Min. :15.80
1st Qu.:36.98	1st Qu.:22.00	1st Qu.:30.20	1st Qu.:27.30	1st Qu.:17.60
Median :38.50	Median :22.80	Median :32.05	Median :28.70	Median :18.30
Mean :38.59	Mean :23.02	Mean :32.27	Mean :28.66	Mean :18.23
3rd Qu.:39.92	3rd Qu.:24.00	3rd Qu.:34.33	3rd Qu.:30.00	3rd Qu.:18.80
Max. :49.10	Max. :29.60	Max. :45.00	Max. :34.90	Max. :21.40

Εικόνα 21: Εμφάνιση των περιγραφικών στατιστικών των μεταβλητών του συνόλου δεδομένων Body Fat.

Στη συνέχεια με την εντολή **cor()** και με τη βοήθεια της βιβλιοθήκης **corrplot**. θα μελετήσουμε τις συσχετίσεις μεταξύ των μεταβλητών.

```
>library(corrplot)
>corrplot(cor(Body_Fat),method="color",addCoef.col="black",
order="AOE",number.cex=0.55)
```



Εικόνα 22: Διάγραμμα συσχετίσεων του συνόλου δεδομένων Body Fat.

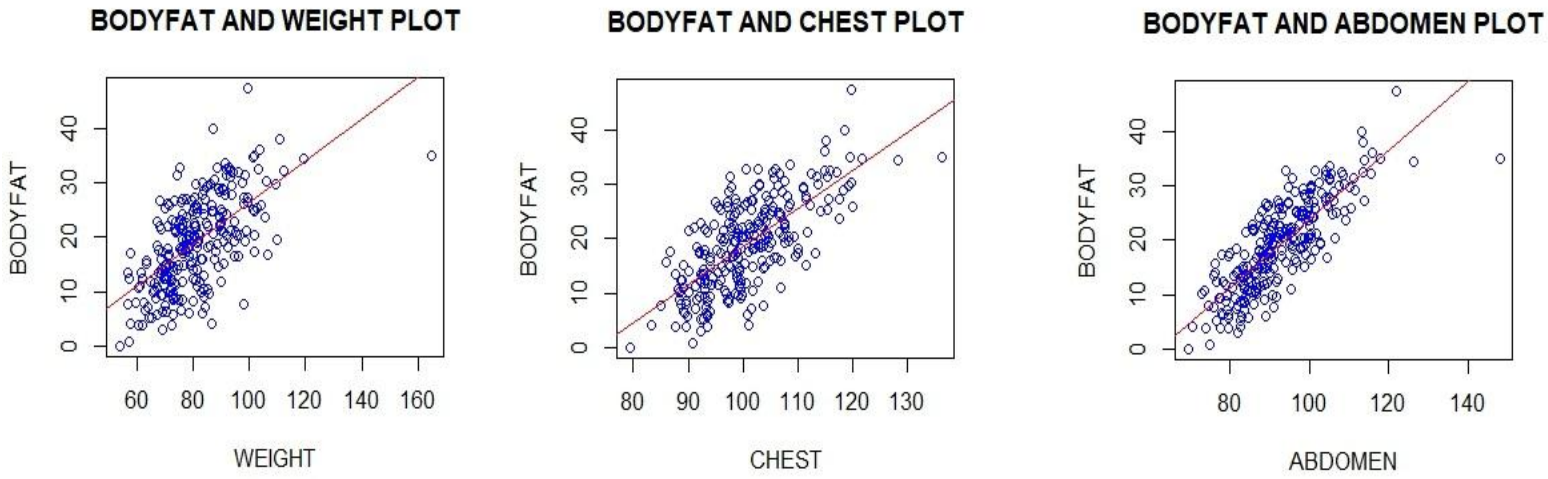
Όπως ήταν αναμενόμενο φαίνεται να υπάρχει συσχέτιση του σωματικού λίπους με το βάρος, αλλά ακόμα πιο έντονη φαίνεται να είναι η συσχέτιση του σωματικού λίπους με τη περιφέρεια στήθους και κοιλιάς. Το ύψος και η ηλικία μοιάζουν να έχουν την χαμηλότερη συσχέτιση με το σωματικό λίπος. Η πυκνότητα του σωματικού λίπους έχει δείκτη -1 πράγμα λογικό καθώς σύμφωνα με την εξίσωση siri οι τιμές είναι αντιστρόφως ανάλογες. Τα παρακάτω διαγράμματα αποδεικνύουν τις συσχετίσεις που αναφέρθηκαν.

```
>model=lm(Body_Fat$`Body Fat`~Body_Fat$Weight)
>plot(Body_Fat$Weight,Body_Fat$`Body
Fat`,col="blue",xlab="WEIGHT",ylab="BODYFAT",main="BODYFAT AND
WEIGHT PLOT")
>abline(model,col="red")
```

```
>model=lm(Body_Fat$`Body Fat`~Body_Fat$Chest)
>plot(Body_Fat$Chest,Body_Fat$`Body
Fat`,col="blue",xlab="CHEST",ylab="BODYFAT",main="BODYFAT AND CHEST
PLOT")
```

```
>abline(model,col="red")
```

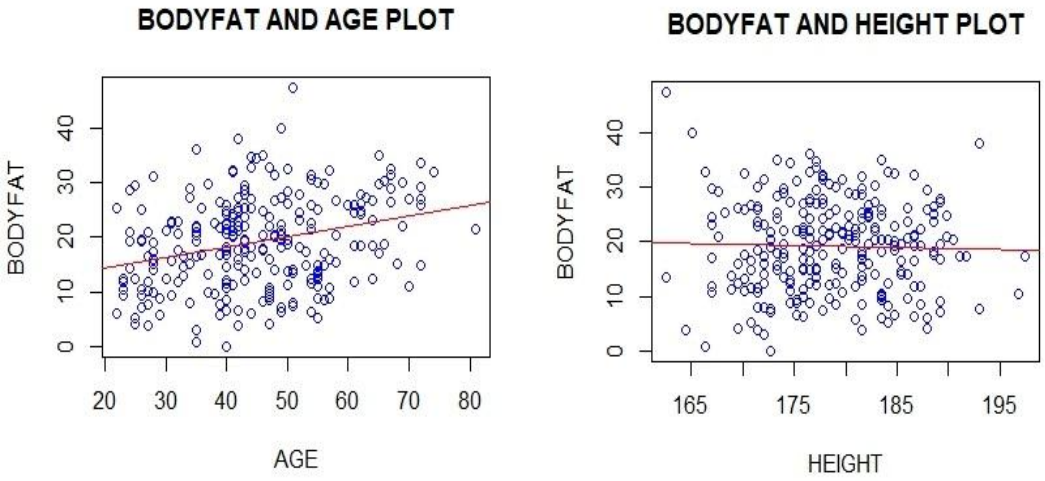
```
>model=lm(Body_Fat$`Body Fat`~Body_Fat$Abdomen)
>plot(Body_Fat$Chest,Body_Fat$`Body
Fat`,col="blue",xlab="ABDOMEN",ylab="BODYFAT",main="BODYFAT AND
ABDOMEN PLOT")
>abline(model,col="red")
```



Εικόνα 23: Διαγράμματα Weight-BodyFat , Chest-BodyFat και Abdomen-BodyFat.

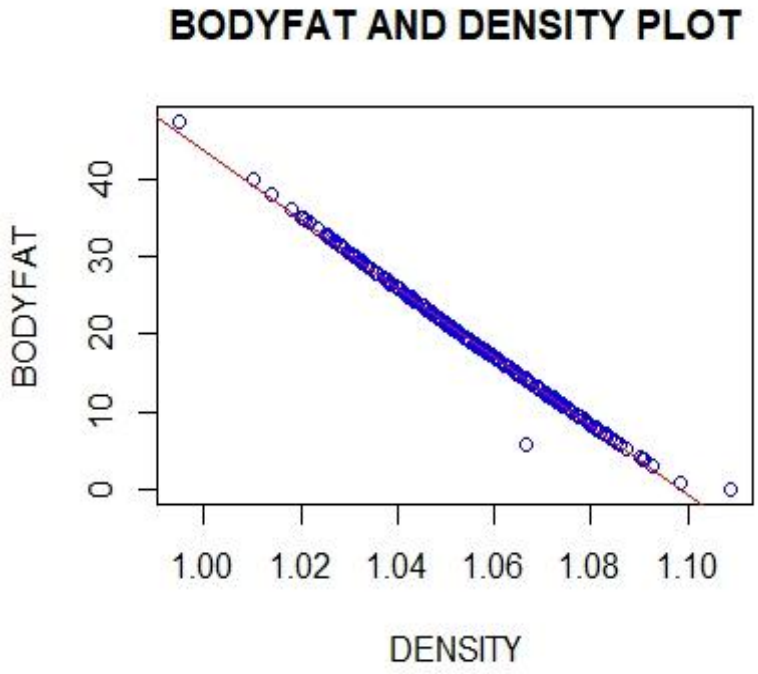
```
>model=lm(Body_Fat$`Body Fat`~Body_Fat$Age)
>plot(Body_Fat$Age,Body_Fat$`Body
Fat`,col="blue",xlab="AGE",ylab="BODYFAT",main="BODYFAT AND AGE
PLOT")
>abline(model,col="red")
```

```
>model=lm(Body_Fat$`Body Fat`~Body_Fat$Height)
>plot(Body_Fat$Height,Body_Fat$`Body
Fat`,col="blue",xlab="HEIGHT",ylab="BODYFAT",main="BODYFAT AND
HEIGHT PLOT")
>abline(model,col="red")
```



Εικόνα 24: Διαγράμματα Age-BodyFat και Height-BodyFat .

```
>model=lm(Body_Fat$`Body Fat`~Body_Fat$Density)
>plot(Body_Fat$Density,Body_Fat$`Body
Fat`,col="blue",xlab="DENSITY",ylab="BODYFAT",main="BODYFAT AND
DENSITY PLOT")
>abline(model,col="red")
```



Εικόνα 25: Διάγραμμα Density-BodyFat .

Έπειτα θα χρησιμοποιήσουμε τη μέθοδο split percentage για να διαχωρίσουμε το σύνολο δεδομένων σε σύνολο εκπαίδευσης και ελέγχου με τη χρήση της βιβλιοθήκης **caTools** .Το μοντέλο θα εκπαιδευτεί με το 80% των παρατηρήσεων και θα εξεταστεί με το υπόλοιπο 20%. Θα δημιουργήσουμε ένα μοντέλο το οποίο θα προβλέπει το σωματικό λίπος με τις μετρήσεις των μελών του σώματος, την ηλικία, το ύψος και το βάρος του ανθρώπου και θα το χρησιμοποιήσουμε για τον υπολογισμό του δικού μου σωματικού λίπους. Η μεταβλητή Density δεν θα συμπεριληφθεί.

```
>library(caTools)
>set.seed(123)
>split=sample.split(Body_Fat$`Body Fat`, SplitRatio = 0.8)
>Train=subset(Body_Fat[!(names(Body_Fat)  %in%  c("Density"))],split
== TRUE)
>Test=subset(Body_Fat[!(names(Body_Fat) %in% c("Density"))],split ==
FALSE)
>model=lm(Train$`Body Fat`~.,data=Train)
```

Εφόσον έχει δημιουργηθεί το μοντέλο, τώρα θα μελετήσουμε την απόδοση του χρησιμοποιώντας το υποσύνολο ελέγχου και μετρώντας το RMSE των προβλέψεων.

```
>predicted=predict(model,Test)
>s=list()

>for(i in 1:nrow(Test)){
     s[[i]]=(Test$`Body Fat`[i]-predicted[i])^2
 }
>rmse=sqrt(mean(unlist(s)))
>rmse
```

```
[1] 3.876338
```

Εικόνα 26: RMSE των πραγματικών τιμών και των προβλέψεων .

Ο υπολογισμός του σωματικού μου λίπος υπολογίζεται με την παρακάτω εντολή.

```
>body=predict.lm(model,data.frame(Age=26,Weight=70,Height=178
,Neck=37 ,Chest=92 ,Abdomen=80 ,Hip=94 , Thigh=45 ,Knee=36 ,Ankle=21
,Biceps=32 ,Forearm=24 ,Wrist=16),type="response")
>body
```

```
8.044473
```

Εικόνα 27: Σωματικό λίπος σύμφωνα με τις μετρήσεις μου.

Πείραμα 2^{ο}

Σύνολο δεδομένων :Skin / No skin dataset [31]

Περιγραφή : Το σύνολο δεδομένων αφορά τυχαίες τιμές B,G,R οι οποίες λήφθηκαν από pixels φωτογραφιών που απεικόνιζαν πρόσωπα ανθρώπων διαφορετικών ηλικιών και εθνικοτήτων.

Skin / No skin dataset	
ΓΡΑΜΜΕΣ	245.057
ΣΤΗΛΕΣ	4
ΤΙΜΗ ΣΤΟΧΟΣ	Isskin (κατηγορηματική)

1) **B** - Τιμή μπλε χρώματος
2) **G** - Τιμή πράσινου χρώματος
3) **R** - Τιμή κόκκινου χρώματος
4) **Isskin** - Αν το δείγμα είναι επιδερμίδα (1 = είναι ,2 = δεν είναι)

Με το συγκεκριμένο πείραμα θα μελετήσουμε τη μέθοδο λογιστικής παλινδρόμησης και το δέντρο απόφασης και θα συγκρίνουμε τις αποδόσεις τους. Τα δυο μοντέλα θα συγκριθούν ως προς το χρόνο εκπαίδευσης και ελέγχου, αλλά και ως προς την ακρίβεια. Οι αλγόριθμοι θα εξεταστούν με δύο διαφορετικούς τρόπους. Τη πρώτη φορά το σύνολο δεδομένων θα διαχωριστεί με τη μέθοδο split percentage και τη δεύτερη φορά θα διαχωριστεί με τη μέθοδο k-cross validation.

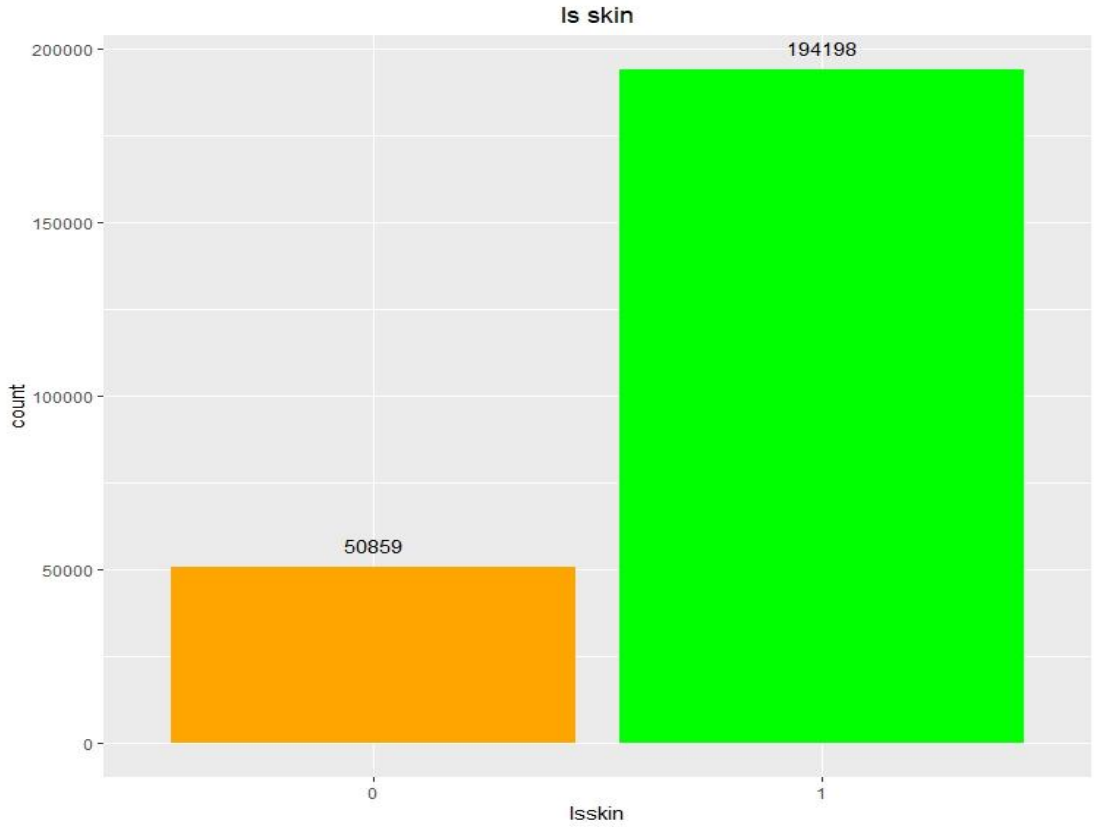
Αλγόριθμος: Λογιστική Παλινδρόμηση (Split percentage)

Αρχικά θα μετατρέψουμε τις δύο κατηγορίες της τέταρτης στήλης σε 0 και 1.

```
>Skin_NonSkin$Isskin[Skin_NonSkin$Isskin=="1"]=0
>Skin_NonSkin$Isskin[Skin_NonSkin$Isskin=="2"]=1
```

Παρατηρούμε ότι οι περισσότερες παρατηρήσεις ανήκουν στην δεύτερη κατηγορία (δ εν είναι επιδερμίδα).

```
>library(ggplot2)
>ggplot(data=Skin_NonSkin, aes(x=Isskin)) +
geom_bar(fill = c('orange','green')) +
geom_text(stat='count', aes(label=..count..), vjust=-1)+
ggtitle("Is skin")+theme(plot.title = element_text(hjust = 0.5))
```



Εικόνα 28: Ραβδόγραμμα της μεταβλητής Isskin.

Όπως και στο πρώτο πείραμα θα διαχωρίσουμε το σύνολο δεδομένων με τη χρήση της βιβλιοθήκης **caTools**, αλλά αυτή τη φορά θα χρησιμοποιήσουμε το μοντέλο λογιστικής παλινδρόμησης. Το αρχικό δείγμα διαχωρίζεται κατά 75% σε υποσύνολο εκπαίδευσης και το υπόλοιπο 25% σε υποσύνολο ελέγχου.

```
>library(caTools)
>set.seed(123)
>split=sample.split(Skin_NonSkin$Isskin, SplitRatio = 0.75)
>Train=subset(Skin_NonSkin,split == TRUE)
>Test=subset(Skin_NonSkin,split == FALSE)
>start.time <- Sys.time()
>model=glm(Isskin~B+G+R,data=Train,family = binomial)
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 1.513939 secs
```

Εικόνα 29: Χρόνος εκπαίδευσης του αλγορίθμου λογιστικής παλινδρόμησης.

Για να τον υπολογισμό της ακρίβειας του μοντέλου θα χρησιμοποιήσουμε τις παρακάτω εντολές.

```
>start.time <- Sys.time()
>pred <- predict(model, newdata=Test,type="response")
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 0.04683304 secs
```

Εικόνα 30: Χρόνος ταξινόμησης του αλγορίθμου λογιστικής παλινδρόμησης.

```
>library(InformationValue)
>optCutOff<-optimalCutoff(Test$Isskin, pred)[1]
>optCutOff
```

```
[1] 0.6999929
```

Εικόνα 31: Βέλτιστο κατώφλι.

```
>table(pred>=optCutOff,Test$Isskin)
```

```
          0     1
FALSE 11709  3618
TRUE   1006 44932
```

Εικόνα 32: Αποτελέσματα προβλέψεων.

Η ακρίβεια του μοντέλου είναι η εξής :

$$Ακρίβεια = \frac{11709 + 44932}{11709 + 44932 + 1006 + 3618} = 0.924$$

Αλγόριθμος: Δέντρο απόφασης (Split percentage)

Για τη δημιουργία του δέντρου απόφασης θα χρησιμοποιήσουμε τη βιβλιοθήκη **rpart** και **rpart.plot**.

Θα εκπαιδεύσουμε το μοντέλο μας με τη μέθοδο του δέντρου απόφασης με το ίδιο δεί γμα εκπαίδευσης που χρησιμοποιήσαμε για τη μέθοδο λογιστικής παλινδρόμησης.

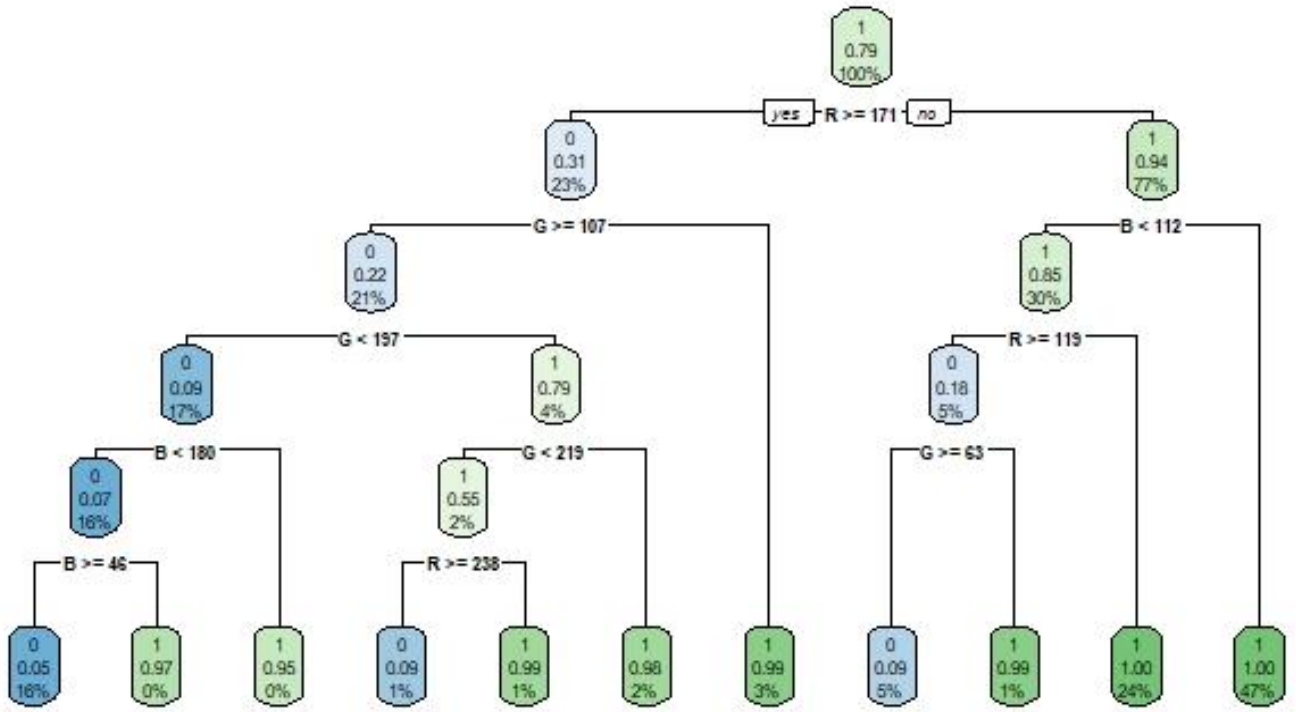
```
>library(rpart)
>library(rpart.plot)
>start.time <- Sys.time()
>model=rpart(Isskin~B+G+R, data = Train, method = 'class')
```

```
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 1.780343 secs
```

Εικόνα 33: Χρόνος εκπαίδευσης του αλγορίθμου δένδρου απόφασης.

```
>rpart.plot(model)
```



Εικόνα 34: Γραφική αναπαράσταση του μοντέλου του δέντρου απόφασης.

Αντίστοιχα, για να βρούμε την ακρίβεια του μοντέλου θα χρησιμοποιήσουμε τις παρακάτω εντολές.

```
>start.time <- Sys.time()
>pred <- predict(model, newdata=Test,type="class")
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 0.05482912 secs
```

Εικόνα 35: Χρόνος ταξινόμησης του αλγορίθμου δένδρου απόφασης.

```
>table(as.matrix(pred),Test$Isskin)
```

```
       0     1
0  12587   847
1    128 47703
```

Εικόνα 36: Αποτελέσματα προβλέψεων.

Η ακρίβεια του μοντέλου είναι η εξής :

$$Ακρίβεια = \frac{12587 + 47703}{12587 + 47703 + 128 + 857} = 0.984$$

Παρατηρούμε ότι η μέθοδος του δέντρου απόφασης πετυχαίνει καλύτερα αποτελέσματα από τη μέθοδο λογιστικής παλινδρόμησης, ενώ οι χρόνοι εκπαίδευσης και ταξινόμησης είναι σχεδόν παρόμοιοι.

	Ακρίβεια	**Χρόνος εκπαίδευσης (sec)**	**Χρόνος ταξινόμησης (sec)**
Λογιστική Παλινδρόμηση	92.4%	1.51	0.046
Δέντρο απόφασης	98.4%	1.78	0.054

Στη συνέχεια θα επαναλάβουμε το πείραμα, αλλά αυτή τη φορά θα χρησιμοποιήσουμε τη μέθοδο k-cross validation.

Αλγόριθμος: Λογιστική Παλινδρόμηση (K-cross validation)

Με τις παρακάτω εντολές ο αλγόριθμος λογιστικής παλινδρόμησης εκπαιδεύεται και ταξινομεί με διαφορετικά δείγματα, τα οποία προκύπτουν με τη μέθοδο k-cross validation. Αυτό έχει ως αποτέλεσμα να προκύψουν δέκα διαφορετικές τιμές για την ακρίβεια του αλγορίθμου. Θα χρησιμοποιηθεί η μέση τιμή των διαφορετικών μετρήσεων για τη σύγκριση της ακρίβειας του μοντέλου λογιστικής παλινδρόμησης με το μοντέλο του δέντρου απόφασης.

```
>glmkcross<-Skin_NonSkin[sample(nrow(Skin_NonSkin)),]
>folds <- cut(seq(1,nrow(glmkcross)),breaks=10,labels=FALSE)
>accuracy=list()
>start.time <- Sys.time()
>for(k in 1:10){

     indexes <- which(folds==k,arr.ind=TRUE)
     test <- glmkcross[indexes, ]
     train <- glmkcross[-indexes, ]
     model=glm(Isskin~B+G+R,data=train,family = binomial)
     pred <- predict(model, newdata=test,type="response")
     optCutOff<-optimalCutoff(test$Isskin, pred)[1]
     p=table(pred>=optCutOff,test$Isskin)
     accuracy[[k]] <-(p[1]+p[4])/(p[1]+p[2]+p[3]+p[4])
 }
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 56.6352 secs
```

Εικόνα 37: Χρόνος εκπαίδευσης και ταξινόμησης του αλγορίθμου λογιστικής παλινδρόμησης.

```
>print(accuracy)
```

```
> print(accuracy)
[[1]]
[1] 0.9236922

[[2]]
[1] 0.9252836

[[3]]
[1] 0.9253214

[[4]]
[1] 0.9222639

[[5]]
[1] 0.9225496

[[6]]
[1] 0.9247092

[[7]]
[1] 0.9246715

[[8]]
[1] 0.925403

[[9]]
[1] 0.9255284

[[10]]
[1] 0.9257733
```

Εικόνα 38:Αποτελέσματα του αλγορίθμου για κάθε επανάληψη της μεθόδου k cross validation .

```
>mean(unlist(accuracy))
```

```
[1] 0.9245196
```

Εικόνα 39: Μέση τιμή των μετρήσεων της ακρίβειας .

Αλγόριθμος: Δέντρο απόφασης (K-cross validation)

Αντίστοιχα, θα εκπαιδεύσουμε τον αλγόριθμο του δέντρου απόφασης.

```
>rpartkcross<-Skin_NonSkin[sample(nrow(Skin_NonSkin)),]
>folds <- cut(seq(1,nrow(rpartkcross)),breaks=10,labels=FALSE)
>accuracy=list()
>start.time <- Sys.time()
>for(k in 1:10){
     indexes <- which(folds==k,arr.ind=TRUE)
     test <- rpartkcross[indexes, ]
     train <- rpartkcross[-indexes, ]
     model=rpart(Isskin~B+G+R, data = train, method = 'class')
     pred <- predict(model, newdata=test,type="class")
     p=table(as.matrix(pred),test$Isskin)
     accuracy[[k]] <-(p[1]+p[4])/(p[1]+p[2]+p[3]+p[4])
 }
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 26.79002 secs
```

Εικόνα 40: Χρόνος εκπαίδευσης και ταξινόμησης του αλγορίθμου δέντρου απόφασης.

```
>print(accuracy)
```

```
> print(accuracy)
[[1]]
[1] 0.9839631

[[2]]
[1] 0.9834326

[[3]]
[1] 0.9835544

[[4]]
[1] 0.9848609

[[5]]
[1] 0.9853913

[[6]]
[1] 0.9837584

[[7]]
[1] 0.9834734

[[8]]
[1] 0.9831463

[[9]]
[1] 0.9837183

[[10]]
[1] 0.9833102
```

Εικόνα 41: Αποτελέσματα του αλγορίθμου για κάθε επανάληψη της μεθόδου k cross validation .

```
>mean(unlist(accuracy))
```

```
[1] 0.9838609
```

Εικόνα 42: Μέση τιμή των μετρήσεων της ακρίβειας .

Από τον παρακάτω πίνακα παρατηρούμε ότι οι αλγόριθμοι κατάφεραν να πετύχουν περίπου τις ίδιες ακρίβειες με τη μέθοδο split percentage. Παρατηρούμε όμως ότι το δέντρο απόφασης εκπαιδεύεται και ταξινομεί σε καλύτερα χρονικά πλαίσια από τη λογιστική παλινδρόμηση.

	Ακρίβεια	**Χρόνος εκπαίδευσης & Ταξινόμησης (sec)**
Λογιστική Παλινδρόμηση	92.4%	56.53
Δέντρο απόφασης	98.3%	26.79

Πείραμα 3ο

Σύνολο δεδομένων :Wine dataset [32]

Περιγραφή : Το σύνολο δεδομένων αποτελεί το αποτέλεσμα μια χημικής ανάλυσης κρασιών που καλλιεργήθηκαν στην ίδια περιοχή της Ιταλίας, αλλά προέρχονται από τρεις διαφορετικές ποικιλίες. Η ανάλυση καθορίζει τις ποσότητες των δεκατριών συστατικών που βρέθηκαν σε καθέναν από τους τρεις τύπους κρασιών.

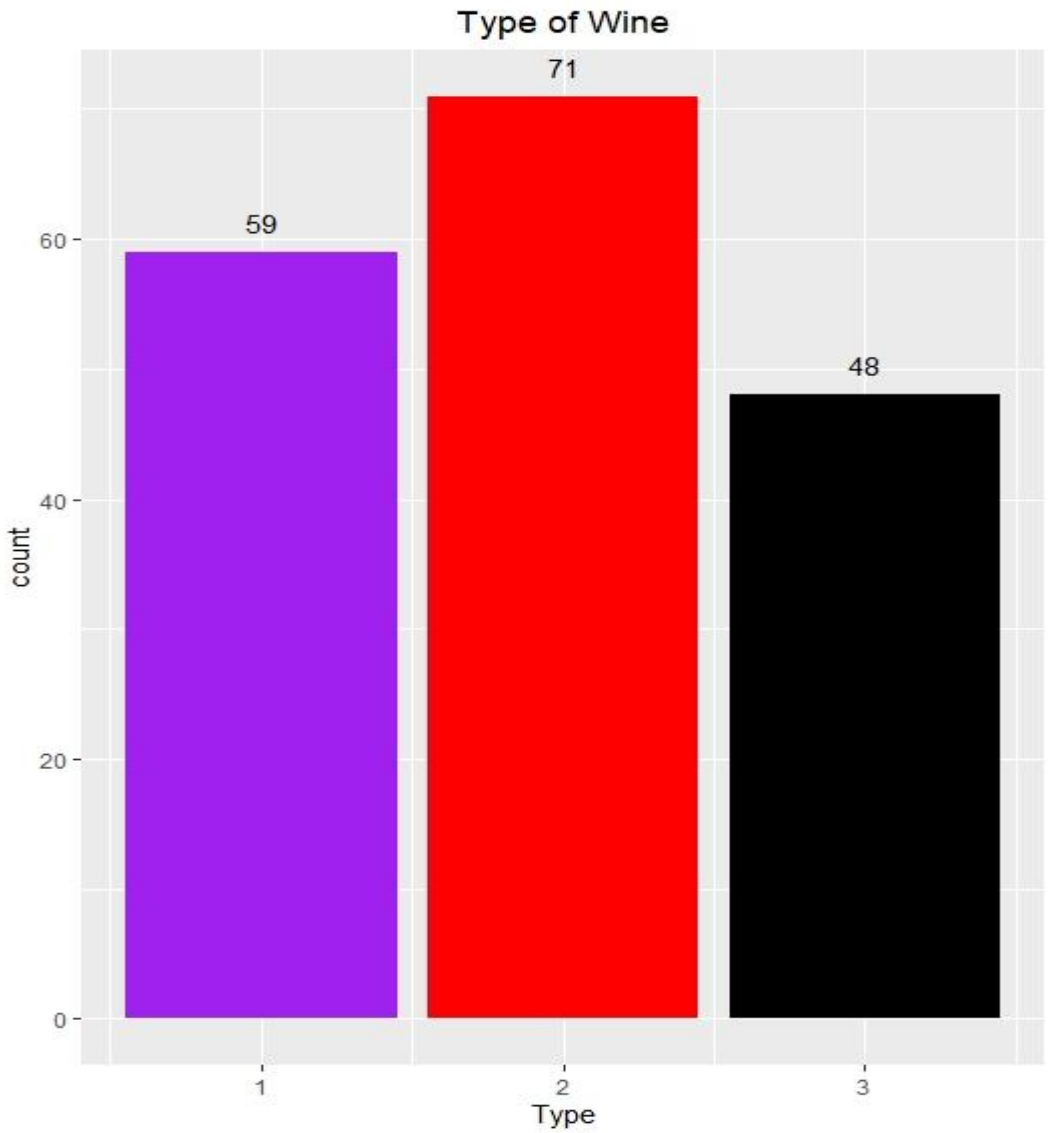
Wine dataset	
ΓΡΑΜΜΕΣ	178
ΣΤΗΛΕΣ	13
ΤΙΜΗ ΣΤΟΧΟΣ	Type (κατηγορηματική)

1) **Type** - Κατηγορία κρασιού (A-B-C)
2) **Malic** - Μηλικό οξύ
3) **Ash** - Τέφρα
4) **Alcalinity** - Αλκαλικότητα της τέφρας
5) **Magnesium** - Μαγνήσιο
6) **Phenols** - Σύνολο φαινολών
7) **Flavanoids** - Σύνολο φλαβονοειδών
8) **Nonflavanoid** - Μη φλαβονοειδείς φαινόλες
9) **Proanthocyanins** - Προανθοκυανιδίνες
10) **Color** - Ένταση χρώματος
11) **Hue** - Απόχρωση
12) **Dilution** - Διαλυτότητα
13) **Proline** - Προλίνη

Με το συγκεκριμένο πείραμα θα μελετήσουμε τη μέθοδο του νευρωνικού δικτύου και της gradient descent και θα συγκρίνουμε τις αποδόσεις τους. Τα δυο μοντέλα θα συγκριθούν ως προς το χρόνο εκπαίδευσης και ελέγχου, αλλά και ως προς την ακρίβεια. Θα χρησιμοποιήσουμε μόνο τη μέθοδο split percentage, καθώς το σύνολο δεδομένων έχει λίγες παρατηρήσεις και δεν απαιτείται η μέθοδος k-cross validation.

Αρχικά, θα μετατρέψουμε τις τιμές της πρώτης κατηγορίας από αλφαριθμητικές σε αριθμητικές. Γνωρίζουμε ότι οι τιμές στη πρώτη κατηγορία είναι A, B ή C οπότε θα αναθέσουμε την τιμή 1 στο A, την τιμή 2 στο B και την τιμή 3 στο C.

```
>Wine$Type[Wine$Type=="A"]=1
>Wine$Type[Wine$Type=="B"]=2
>Wine$Type[Wine$Type=="C"]=3
>Wine$Type=as.factor(Wine$Type)
>ggplot(data=Wine, aes(x=Type)) +
geom_bar(fill = c('purple','red',"black")) +
geom_text(stat='count', aes(label=..count..), vjust=-1)+
ggtitle("Type of Wine")+theme(plot.title=element_text(hjust=0.5))
```



Εικόνα 43: Ραβδόγραμμα της μεταβλητής Type.

Έπειτα θα εφαρμόσουμε κανονικοποίηση στις τιμές με τις παρακάτω εντολές.

```
>normalize <- function(x) {
     for(i in 1 : ncol(x)){
         me=sum(x[,i])/nrow(x)
         mi=min(x[,i])
         ma=max(x[,i])
         for(j in 1 : nrow(x)){
             x[j,i]= (x[j,i]-me)/(ma-mi)
         }
     }
     return(x)
 }
>Wine[2:14]=normalize(Wine[2:14])
```

Αλγόριθμος: Νευρωνικό Δίκτυο (Split percentage)

Θα δημιουργήσουμε τρεις επιπλέον στήλες σύμφωνα με τις τιμές των κλάσεων με τη βιβλιοθήκη **neuralent** και **nnet.**. Για τη στήλη Wine1 αν η τιμή του χαρακτηριστικού Type είναι 1 τότε θα εκχωρηθεί η τιμή 1. Διαφορετικά αν η τιμή του χαρακτηριστικού Type είναι 2 ή 3 θα εκχωρηθεί η τιμή 0. Αντίστοιχα, για το Wine 2 θα εκχωρηθεί 1 αν η τιμή του χαρακτηριστικού Type είναι 2 και 0 σε διαφορετική περίπτωση. Για το Wine 3 θα εκχωρηθεί 1 αν η τιμή του χαρακτηριστικού Type είναι 3 ειδάλλως 0. Η αναγκαιότητα της δημιουργίας αυτών των στηλών είναι για την εκπαίδευση του νευρωνικού δικτύου.

```
>library(neuralnet)
>library(nnet)
>Type=Wine$Type
>Wine<- cbind(Wine[, 2:14], class.ind(as.factor(Wine$Type)))
>names(Wine) <- c(names(Wine)[1:13],"Wine1","Wine2","Wine3")
>Wine=cbind(Wine,Type)
```

Χωρίζουμε το δείγμα μας σε υποσύνολο εκπαίδευσης και ελέγχου όπως στα προηγούμενα πειράματα και θα συγκρίνουμε τους αλγορίθμους ως προς την ακρίβεια τους και ως προς το χρόνο εκπαίδευσης και ταξινόμησης.

```
>library(caTools)
>set.seed(123)
```

```
>split=sample.split(Wine$Type, SplitRatio = 0.75)
>Train=subset(Wine[,1:16],split == TRUE)
>Test=subset(Wine,split == FALSE)
```

Για την κατάλληλη επιλογή των κρυφών νευρώνων θα χρησιμοποιηθεί ο κανόνας του αντίχειρα:

$$Κρυφοί\ Νευρώνες = \frac{Αριθμός\ εγγραφών}{(Αριθμός\ χαρακτηριστικών + Αριθμός\ κλάσεων) * α}$$

Θα ορίσουμε το α=2 εφόσον ο αριθμός των εγγραφών είναι πολύ μικρός άρα:

$$Κρυφοί\ Νευρώνες = \frac{178}{(13+3)*2} = 5.56$$

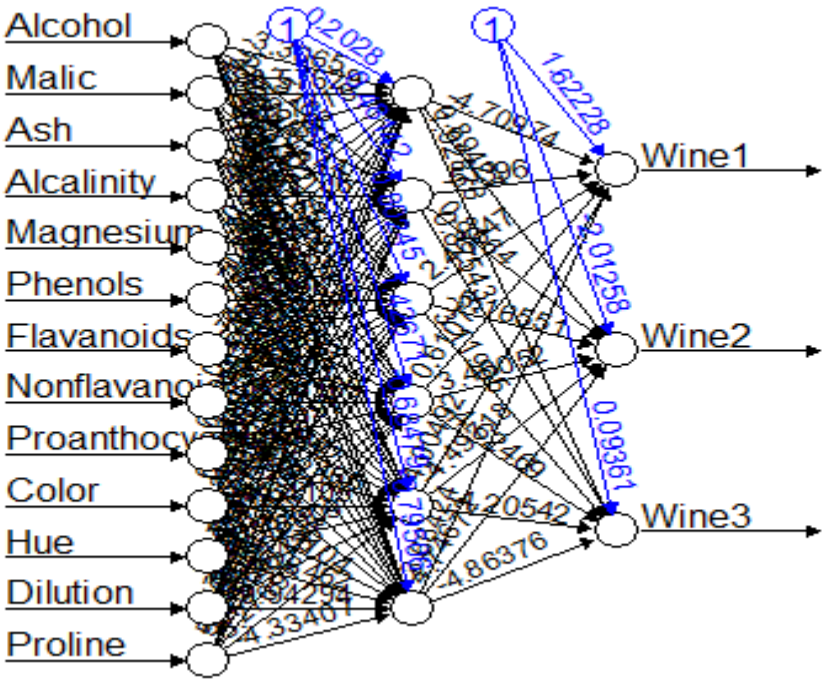
Θα χρησιμοποιήσουμε έξι κρυφούς νευρώνες και θα δημιουργήσουμε το νευρωνικό δίκτυο.

```
>start.time <- Sys.time()
>n=colnames(Train)
>n=n[!n %in% "Wine1"]
>n=n[!n %in% "Wine2"]
>n=n[!n %in% "Wine3"]
>f=as.formula(paste("Wine1+Wine2+Wine3~",paste(n,collapse =" + ")))
>model=neuralnet(f,data=Train,hidden=6,act.fct="logistic",linear.out
put = FALSE)
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 0.387645 secs
```

Εικόνα 44: Χρόνος εκπαίδευσης του αλγορίθμου νευρωνικού δικτύου.

```
>plot(model)
```



Εικόνα 45: Γραφική αναπαράσταση του νευρωνικού δικτύου.

```
>start.time <- Sys.time()
>pred <- predict(model, Test)
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 0.01609516 secs
```

Εικόνα 46: Χρόνος ταξινόμησης του αλγορίθμου νευρωνικού δικτύου.

```
>table(Test$Type, as.matrix(apply(pred, 1, which.max)))
```

```
   1  2  3
1 15  0  0
2  0 16  2
3  0  0 12
```

Εικόνα 47: Αποτελέσματα προβλέψεων.

Η ακρίβεια του μοντέλου είναι η εξής :

$$Ακρίβεια = \frac{15 + 16 + 12}{15 + 16 + 2 + 12} = 0.955$$

Αλγόριθμος: Gradient Descent (Split percentage)

Στη συνέχεια του πειράματος, θα συγκρίνουμε τις αποδόσεις του νευρωνικού δικτύου με τη μέθοδο Gradient Descent.

Ξεκινώντας, θα χρησιμοποιήσουμε το αρχικό σύνολο δεδομένων πριν τη δημιουργία των 3 επιπλέον στηλών και θα εφαρμόσουμε κανονικοποίηση.

```
>Wine2$Type[Wine2$Type=="A"]=1
>Wine2$Type[Wine2$Type=="B"]=2
>Wine2$Type[Wine2$Type=="C"]=3
>Wine2$Type=as.numeric(Wine2$Type)
>normalize <- function(x) {
     for(i in 1 : ncol(x)){
         me=sum(x[,i])/nrow(x)
         mi=min(x[,i])
         ma=max(x[,i])
         for(j in 1 : nrow(x)){
             x[j,i]= (x[j,i]-me)/(ma-mi)
         }
     }
     return(x)
 }
>Wine2[2:14]=normalize(Wine2[2:14])
```

Στη συνέχεια, θα χωρίσουμε το δείγμα μας σε υποσύνολο εκπαίδευσης και υποσύνολο ελέγχου και εφόσον το δείγμα αποτελείται από τρεις κλάσεις, με τη μέθοδο one-vs-all θα δημιουργήσουμε τρία νέα δείγματα (ένα για κάθε κλάση). Επιπλέον, στα καινούργια δείγματα προσθέτεται μια καινούργια στήλη με την τιμή 1 σε όλες τις γραμμές της και αυτή η καινούργια στήλη ορίζεται ως αμερόληπτη μεταβλητή.

```
>set.seed(123)
>index=sort(sample(nrow(Wine2), nrow(Wine2)*0.75))
>Xtrain=Wine2[index,]
>Xtest=Wine2[-index,]
>Xtrain$bias=1
>Xtrain=Xtrain[,c(1,15,2:14)]
>Xtest$bias=1
>Xtest=Xtest[,c(1,15,2:14)]
>Xtrain1=Xtrain
>Xtrain2=Xtrain
```

```
>Xtrain3=Xtrain
>for(v in 1: nrow(Xtrain)){
     if(Xtrain[v,1]==1){
         Xtrain1[v,1]=1
     }else{
         Xtrain1[v,1]=-1
     }
 }
>for(v in 1: nrow(Xtrain)){
     if(Xtrain[v,1]==2){
         Xtrain2[v,1]=1
     }
     else{
         Xtrain2[v,1]=-1
     }
 }
>for(v in 1: nrow(Xtrain)){
     if(Xtrain[v,1]==3){
         Xtrain3[v,1]=1
     }
     else{
         Xtrain3[v,1]=-1
     }
 }
```

Έπειτα, γίνεται αρχικοποίηση των διανυσμάτων με τα βάρη Theta με την τιμή 0 και ορίζεται ο αριθμός των επαναλήψεων και η τιμή a. Οι παράμετροι αυτές μαζί με τα σύνολα εκπαίδευσης εισάγονται στον αλγόριθμο της Gradient Descent, παράγονται τα καινούργια βάρη Theta και ως συνέπεια εκπαιδεύεται το μοντέλο μας.

```
>iterations <- 15000
>alpha <- 0.5
>theta1 <- matrix(c(0,0,0,0,0,0,0,0,0,0,0,0,0,0))
>theta2 <- matrix(c(0,0,0,0,0,0,0,0,0,0,0,0,0,0))
>theta3 <- matrix(c(0,0,0,0,0,0,0,0,0,0,0,0,0,0))
>GradientDescent <- function(X, Y, theta, iterations,alpha) {
     m <- length(Y)
     for(iter in 1:iterations) {
         deriv <- X %*% theta - Y
         theta <- theta - ((alpha/m) * t(t(deriv) %*% X))
     }
     return(theta)
 }
>start.time <- Sys.time()
```

```
>theta1<-
GradientDescent(as.matrix(Xtrain1[2:15]),as.matrix(Xtrain1[1]),
theta1, iterations,alpha)
>theta2<-
GradientDescent(as.matrix(Xtrain2[2:15]),as.matrix(Xtrain2[1]),
theta2, iterations,alpha)
>theta3<-
GradientDescent(as.matrix(Xtrain3[2:15]),as.matrix(Xtrain3[1]),
theta3, iterations,alpha)
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 1.394708 secs
```

Εικόνα 48: Χρόνος εκπαίδευσης του αλγορίθμου Gradient Descent.

Στη συνέχεια, θα προβλέψουμε τις τιμές στο σύνολο δεδομένων Xtest.

```
>start.time <- Sys.time()
>pr=list()
>max=0
>for(i in 1:nrow(Xtest)){


if((as.matrix(Xtest[i,2:15])%*%theta1>as.matrix(Xtest[i,2:15])%*%the
ta2)                                                                &&
(as.matrix(Xtest[i,2:15])%*%theta1>as.matrix(Xtest[i,2:15])%*%theta3
)){
        max=1
    }
else
if((as.matrix(Xtest[i,2:15])%*%theta2>as.matrix(Xtest[i,2:15])%*%the
ta1)                                                                &&
(as.matrix(Xtest[i,2:15])%*%theta2>as.matrix(Xtest[i,2:15])%*%theta3
)){
        max=2
    }
    else {
        max=3
    }

    pr[[i]]=max

    max=0
 }
>end.time <- Sys.time()
```

```
>time.taken <- end.time - start.time
>time.taken
Time difference of 0.2058079 secs
```

Εικόνα 49: Χρόνος ταξινόμησης του αλγορίθμου Gradient Descent..

```
>table(unlist(pr),Xtest$Type)
   1  2  3
1 13  0  0
2  0 18  0
3  0  1 13
```

Εικόνα 50: Αποτελέσματα προβλέψεων.

$$Ακρίβεια = \frac{13 + 18 + 13}{13 + 18 + 1 + 13} = 0.978$$

Παρατηρούμε πως και τα δύο μοντέλα πέτυχαν πολύ υψηλές τιμές ακρίβειας παρότι το σύνολο δεδομένων αποτελούταν από τρεις κλάσεις αντί για δύο. Οι χρόνοι των δυο μοντέλων είναι σχεδόν ίδιοι, αλλά αυτό οφείλεται κατά κύριο λόγο στο μικρό μέγεθος του συνόλου δεδομένων.

	Ακρίβεια	Χρόνος εκπαίδευσης (sec)	Χρόνος ταξινόμησης (sec)
Νευρωνικό Δίκτυο	95.5%	0.38	0.01
Gradient Descent.	97.8%	1.39	0.2

Πείραμα 4^{ο}

Σύνολο δεδομένων : TrainingSet - TrainingSubset- Testset – TestSubset [33]

Περιγραφή : Το σύνολο δεδομένων αποτελείται από τέσσερα αρχεία τα οποία αφορούν ηλεκτρονικές δημοπρασίες στο Ebay. Τα δύο αρχεία (TrainingSet - TrainingSubset) χρησιμοποιούνται για την εκπαίδευση των μοντέλων, ενώ τα άλλα δυο (Testset – TestSubset) χρησιμοποιούνται για να συγκρίνουμε τις επιδόσεις των μοντέλων. Τα αρχεία TrainingSubset/TestSubset είναι υποσύνολα των TrainingSet/Testset και αφορούν μόνο τις δημοπρασίες που οδήγησαν σε πώληση.

TrainingSet	
ΓΡΑΜΜΕΣ	258.588
ΣΤΗΛΕΣ	28
ΤΙΜΗ ΣΤΟΧΟΣ	QuantitySold (κατηγορηματική)

TestSet	
ΓΡΑΜΜΕΣ	37.460
ΣΤΗΛΕΣ	28
ΤΙΜΗ ΣΤΟΧΟΣ	QuantitySold (κατηγορηματική)

Από τις 28 στήλες επιλέχθηκαν οι παρακάτω 11.

TrainingSet / TestSet:

1) **QuantitySold** - Αν το αντικείμενο πουλήθηκε ή δεν πουλήθηκε (1 ή 0)
2) **Price** - Τελική τιμή που πουλήθηκε το αντικείμενο στη δημοπρασία
3) **Category** - Η κατηγορία του αντικειμένου π.χ. (μπάλα, καπέλο, κράνος)
4) **StartingBid** - Ελάχιστη προσφορά για το αντικείμενο
5) **AvgPrice** - Μέση τιμή πώλησης του αντικειμένου

6) **HitCount** - Αριθμός ατόμων που συμμετείχαν στη δημοπρασία

7) **SellerAvg** - Μέσος όρος χρημάτων που ξοδεύει ο αγοραστής

8) **BestOffer** - Καλύτερη προσφορά για το αντικείμενο

9) **IsHOF** - Αν ο παίχτης κατείχε το αντικείμενο άνηκε το Hall of fame του αθλήματος

10) **AuctionCount** - Αριθμός δημοπρασιών που συμμετείχε το αντικείμενο

11) **AuctionSaleCount** - Αριθμός πωλήσεων που έγιναν στη δημοπρασία

TrainingSubSet	
ΓΡΑΜΜΕΣ	79.732
ΣΤΗΛΕΣ	33
ΤΙΜΗ ΣΤΟΧΟΣ	Price (αριθμητική)

TestSubSet	
ΓΡΑΜΜΕΣ	9.392
ΣΤΗΛΕΣ	33
ΤΙΜΗ ΣΤΟΧΟΣ	Price (αριθμητική)

Από τις 33 στήλες επιλέχθηκαν οι παρακάτω 10.

TrainingSubSet / TestSubset:

1) **Price** - Τελική τιμή που πουλήθηκε το αντικείμενο στη δημοπρασία

2) **Category** - Η κατηγορία του αντικειμένου π.χ. (μπάλα, καπέλο, κράνος)

3) **StartingBid** - Ελάχιστη προσφορά για το αντικείμενο

4) **AvgPrice** - Μέση τιμή πώλησης του αντικειμένου

5) **HitCount** - Αριθμός ατόμων που συμμετείχαν στη δημοπρασία

6) **Authenticated** - Αν το αντικείμενο είναι αυθεντικό

7) **SellerAvg** - Μέσος όρος χρημάτων που ξοδεύει ο αγοραστής

8) **ReturnsAccepted** - Αν επιτρέπεται η επιστροφή χρημάτων

9) **IsHOF** - Αν ο παίχτης που κατείχε το αντικείμενο άνηκε το Hall of fame του αθλήματος

10) **BidCount** - Προσφορές που έγιναν για το αντικείμενο

Ο λόγος που δεν συμπεριλήφθηκαν τα υπόλοιπα χαρακτηριστικά είναι επειδή δεν θεωρήθηκαν αξιόλογα ως προς τη συσχέτιση τους με την τιμή στόχο

TrainingSubSet & TestSubSet:

Για το συγκριμένο σύνολο δεδομένων θα συγκρίνουμε τους χρόνους εκπαίδευσης, ταξινόμησης και το μέσο τετραγωνικό σφάλμα που προκύπτει από τις προβλέψεις των αλγορίθμων της γραμμικής παλινδρόμησης και του δέντρου απόφασης. Καθώς το σύνολο δεδομένων είναι ήδη διαχωρισμένο σε υποσύνολο εκπαίδευσης και ελέγχου δεν θα χρησιμοποιήσουμε τη μέθοδο split percentage, αλλά θα εφαρμοστεί ενοποίηση των 2 συνόλων πριν την εφαρμογή της μεθόδου k-cross validation.

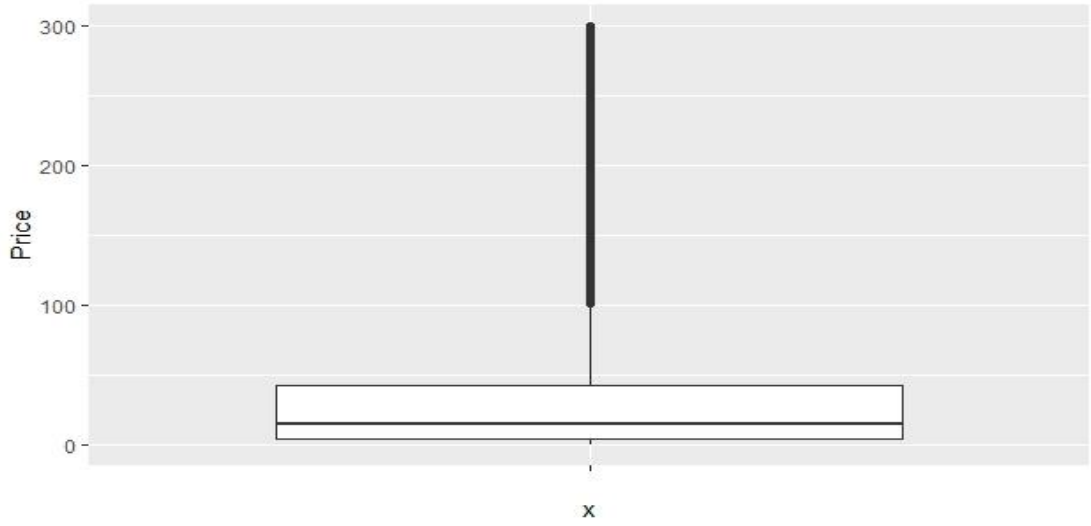
Αρχικά με την εντολή **summary()** θα εξετάσουμε τα περιγραφικά στατιστικά των χαρακτηριστικών μας.

```
>summary(TrainingSubset)
```

```
     Price            Category        StartingBid        AvgPrice
Min.   :  0.01   Min.   :    53   Min.   :  0.01   Min.   :    0.010
1st Qu.:  3.82   1st Qu.: 27265   1st Qu.:  0.99   1st Qu.:    7.281
Median : 14.99   Median : 27277   Median :  2.00   Median :   22.933
Mean   : 33.04   Mean   : 43611   Mean   : 12.27   Mean   :   41.859
3rd Qu.: 42.70   3rd Qu.: 73396   3rd Qu.:  9.99   3rd Qu.:   55.070
Max.   :299.99   Max.   :101810   Max.   :299.99   Max.   :12672.082
    HitCount       Authenticated      SellerAvg       ReturnsAccepted
Min.   :   1.00   Min.   :0.0000   Min.   :   0.00   Min.   :0.000
1st Qu.:   8.00   1st Qu.:0.0000   1st Qu.:   7.00   1st Qu.:0.000
Median :  18.00   Median :0.0000   Median :  25.00   Median :1.000
Mean   :  31.76   Mean   :0.1754   Mean   :  40.18   Mean   :0.692
3rd Qu.:  42.00   3rd Qu.:0.0000   3rd Qu.:  59.00   3rd Qu.:1.000
Max.   :1161.00   Max.   :1.0000   Max.   :1711.00   Max.   :1.000
     IsHOF           BidCount
Min.   :0.0000   Min.   : 1.000
1st Qu.:0.0000   1st Qu.: 1.000
Median :0.0000   Median : 3.000
Mean   :0.2595   Mean   : 6.042
3rd Qu.:1.0000   3rd Qu.: 9.000
Max.   :1.0000   Max.   :93.000
```

Εικόνα 51: Εμφάνιση των περιγραφικών στατιστικών των μεταβλητών.

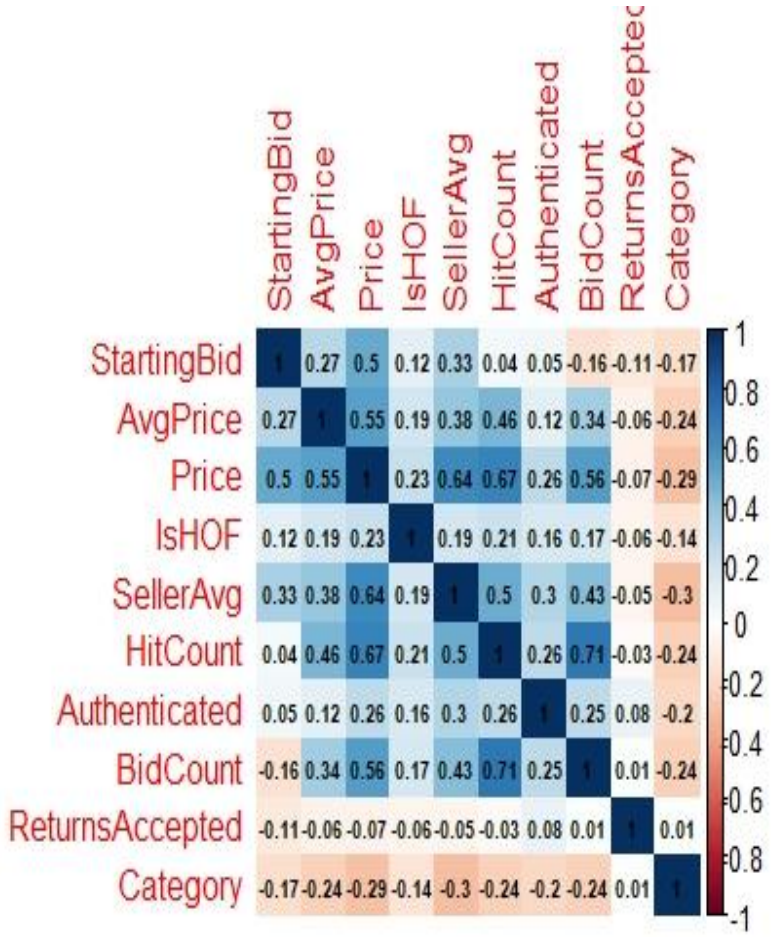
```
>bp=ggplot(TrainingSubset, aes(x= "", y=Price)) + geom_boxplot()
>plot(bp)
```



Εικόνα 52: Boxplot της μεταβλητής Price.

Στη συνέχεια με την εντολή **cor()** θα μελετήσουμε τις συσχετίσεις μεταξύ των μεταβλητών.

```
>corrplot(cor(TrainingSubset),method="color",addCoef.col="blac
k", order = "AOE",number.cex=0.55)
```



Εικόνα 53: Διάγραμμα συσχετίσεων του συνόλου δεδομένων TrainingSubSet .

Παρατηρούμε ότι οι μεταβλητές με τη μεγαλύτερη συσχέτιση με την τιμή στόχου μας είναι η HitCount και η SellerAvg. Οι τιμές αυτές ήταν αναμενόμενες, καθώς είναι λογικό ο αριθμός των ατόμων και ο μέσος όρος χρημάτων που ξοδεύει ο αγοραστής στις δημοπρασίες του να επηρεάζουν την τελική τιμή πώλησης του προϊόντος.

Αλγόριθμος: Γραμμική Παλινδρόμηση

Θα δημιουργήσουμε το μοντέλο της γραμμικής παλινδρόμησης.

```
>start.time <- Sys.time()
>model=lm(Price~.,data=TrainingSubset)
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 0.09990788 secs
```

Εικόνα 54: Χρόνος εκπαίδευσης του αλγορίθμου γραμμικής παλινδρόμησης.

```
>start.time <- Sys.time()
>predicted=predict(model,TestSubset)
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 0.01589203 secs
```

Εικόνα 55: Χρόνος εκπαίδευσης του αλγορίθμου γραμμικής παλινδρόμησης.

Το μέσο τετραγωνικό σφάλμα θα υπολογιστεί με τον παρακάτω κώδικα.

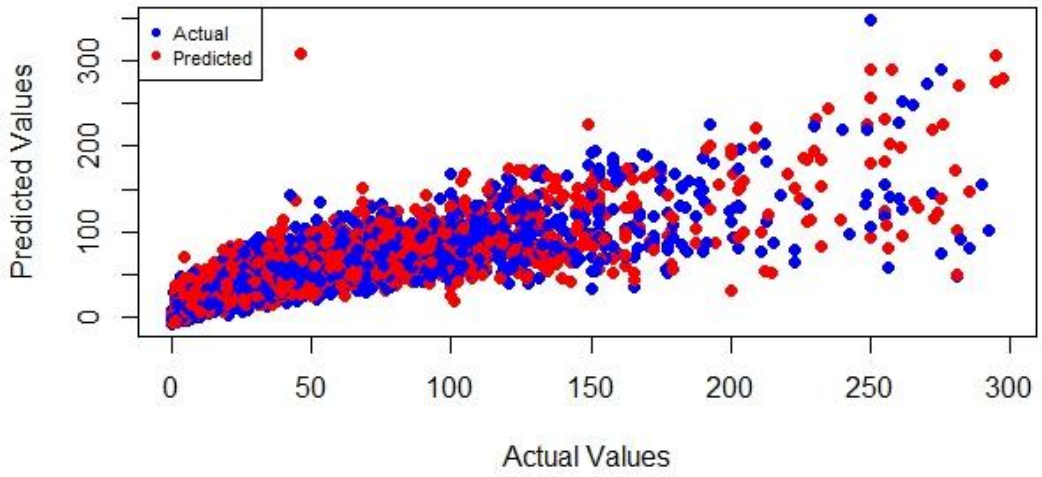
```
>s=list()
>for(i in 1:nrow(TestSubset)){

     s[[i]]=(TestSubset$Price[i]-predicted[i])^2
 }
>rmse=sqrt(mean(unlist(s)))
>rmse
```

```
[1] 20.32991
```

Εικόνα 56: RMSE των πραγματικών τιμών και των προβλέψεων.

```
>plot(TestSubset$Price,predicted,col=c("blue","red"),pch=19,ylab="Pr
edicted Values",xlab="Actual Values")
>legend("topleft",legend=c("Actual","Predicted"),col=c("blue","red")
,pch=19,bty ='y,cex=0.7)
```



Εικόνα 57: Γραφική αναπαράσταση των προβλέψεων και των πραγματικών τιμών (Γραμμική παλινδρόμηση).

Αλγόριθμος: Δέντρο απόφασης

Θα δημιουργήσουμε το μοντέλο του δέντρου απόφασης.

```
>library(rpart)
>start.time <- Sys.time()
>model=rpart(Price~.,data=TrainingSubset, method = "anova")
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 1.451747 secs
```

Εικόνα 58: Χρόνος εκπαίδευσης του μοντέλου δένδρου απόφασης.

```
>start.time <- Sys.time()
>predicted=predict(model,TestSubset)
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 0.02808595 secs
```

Εικόνα 59: Χρόνος ταξινόμησης του μοντέλου δένδρου απόφασης.

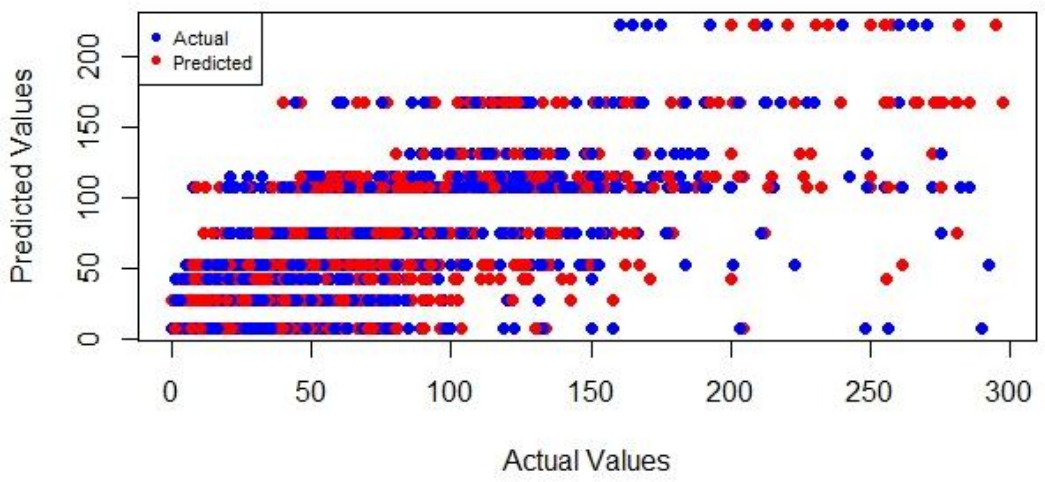
```
>s=list()
>for(i in 1:nrow(TestSubset)){

    s[[i]]=(TestSubset$Price[i]-predicted[i])^2
 }
>sqrt(mean(unlist(s)))
```

```
[1] 24.03326
```

Εικόνα 60: RMSE των πραγματικών τιμών και των προβλέψεων.

```
>plot(TestSubset$Price,predicted,col=c("blue","red"),pch=19,ylab="Pr
edicted Values",xlab="Actual Values")
>legend("topleft",legend=c("Actual","Predicted"),col=c("blue","red")
,pch=19,bty='y,cex=0.7)
```



Εικόνα 61: Γραφική αναπαράσταση των προβλέψεων και των πραγματικών τιμών (Δέντρο απόφασης).

	RMSE	Χρόνος εκπαίδευσης (sec)	Χρόνος ταξινόμησης (sec)
Γραμμική Παλινδρόμηση	20.32	0.09	0.01
Δέντρο απόφασης	24.03	1.45	0.02

Από τον παραπάνω πίνακα παρατηρούμε ότι η γραμμική παλινδρόμηση πετυχαίνει καλύτερα αποτελέσματα από το δέντρο απόφασης. Θα εξετάσουμε αν ισχύει το ίδιο αφού εφαρμόσουμε την μέθοδο k-cross validation. Για τη ενοποίηση των δύο συνόλων χρησιμοποιείται η παρακάτω εντολή.

```
>WholeDataset=rbind(TrainingSubset,TestSubset)
```

Αλγόριθμος: Γραμμική Παλινδρόμηση(K-cross validation)

```
>lmkcross<-WholeDataset[sample(nrow(WholeDataset)),]
>folds <- cut(seq(1,nrow(lmkcross)),breaks=10,labels=FALSE)
>rmse=list()
>start.time <- Sys.time()
>for(k in 1:10){
    indexes <- which(folds==k,arr.ind=TRUE)
```

```
    test <- lmkcross[indexes, ]
    train <- lmkcross[-indexes, ]
    model=lm(Price~.,data=train)
    predicted <- predict(model, newdata=test)
    s=list()
    for(i in 1:nrow(test)){
        s[[i]]=(test$Price[i]-predicted[i])^2
    }
    rmse[[k]] <-sqrt(mean(unlist(s)))
 }
 end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 1.35872 secs
```

Εικόνα 62: Χρόνος εκπαίδευσης και ταξινόμησης του αλγορίθμου γραμμικής παλινδρόμησης.

```
>print(rmse)
```

```
[[1]]
[1] 21.49723

[[2]]
[1] 21.73879

[[3]]
[1] 21.98306

[[4]]
[1] 21.83582

[[5]]
[1] 21.20959

[[6]]
[1] 21.39268

[[7]]
[1] 22.19347

[[8]]
[1] 24.2668

[[9]]
[1] 21.99553

[[10]]
[1] 21.41993
```

Εικόνα 63: Αποτελέσματα του αλγορίθμου για κάθε επανάληψη της μεθόδου k cross validation .

```
>mean(unlist(rmse))
```

```
[1] 21.95329
```

Εικόνα 64: Μέση τιμή των μετρήσεων RMSE .

Αλγόριθμος: Δέντρο απόφασης (K-cross validation)

```
>rpartkcross<-WholeDataset[sample(nrow(WholeDataset)),]
>folds <- cut(seq(1,nrow(rpartkcross)),breaks=10,labels=FALSE)
>rmse=list()
>start.time <- Sys.time()
>for(k in 1:10){

     indexes <- which(folds==k,arr.ind=TRUE)
     test <- rpartkcross[indexes, ]
     train <- rpartkcross[-indexes, ]
     model=rpart(Price~.,data=train, method = 'anova')
     predicted <- predict(model, newdata=test)
     s=list()
     for(i in 1:nrow(test)){
         s[[i]]=(test$Price[i]-predicted[i])^2
     }
     rmse[[k]] <-sqrt(mean(unlist(s)))
 }
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 16.00008 secs
```

Εικόνα 65: Χρόνος εκπαίδευσης και ταξινόμησης του αλγορίθμου δένδρου απόφασης.

```
>print(rmse)
```

```
> print(rmse)
[[1]]
[1] 25.04963

[[2]]
[1] 24.8586

[[3]]
[1] 25.05314

[[4]]
[1] 25.0248

[[5]]
[1] 25.55999

[[6]]
[1] 24.87772

[[7]]
[1] 25.32039

[[8]]
[1] 25.89155

[[9]]
[1] 25.77559

[[10]]
[1] 24.12827
```

Εικόνα 66: Αποτελέσματα του αλγορίθμου για κάθε επανάληψη της μεθόδου k cross validation .

```
>mean(unlist(rmse))
```

```
[1] 25.15397
```

Εικόνα 67: Μέση τιμή των μετρήσεων RMSE .

Παρατηρούμε πως η γραμμική παλινδρόμηση καταφέρνει πάλι να πετύχει καλύτερα αποτελέσματα από το δέντρο απόφασης.

	Mean RMSE	Χρόνος εκπαίδευσης & Ταξινόμησης (sec)
Γραμμική Παλινδρόμηση	21.95	1.35
Δέντρο απόφασης	25.15	16

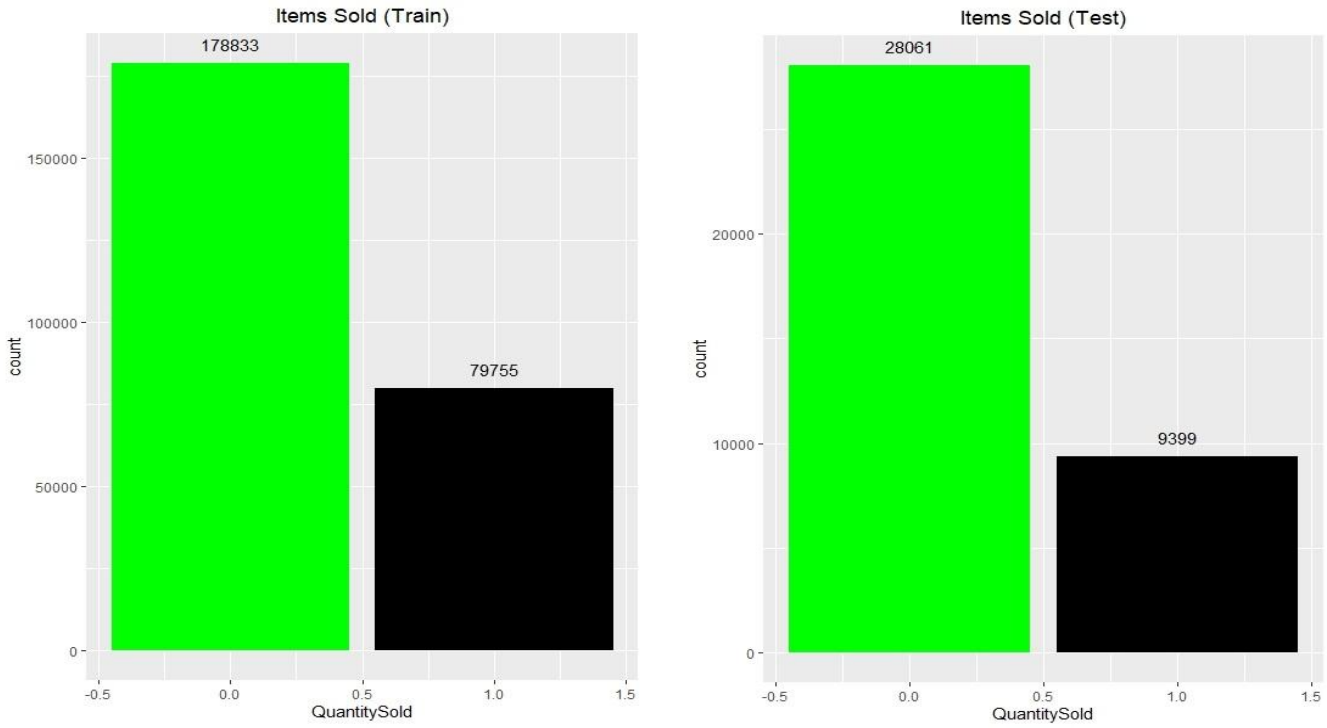
TrainingSet & TestSet:

Για το συγκριμένο σύνολο δεδομένων θα συγκρίνουμε τις αποδόσεις του απλοϊκού Bayes της βιβλιοθήκης **e1071** και του K-nearest neighbors της βιβλιοθήκης **caret**. Θα συγκρίνουμε τους χρόνους εκπαίδευσης και ταξινόμησης και την ακρίβεια των προβλέψεων. Όπως και στο πρώτο σκέλος του πειράματος δεν θα εφαρμόσουμε split percentage, καθώς το σύνολο δεδομένων είναι ήδη διαχωρισμένο σε υποσύνολα εκπαίδευσης και ταξινόμησης.

Αρχικά, θα μετατρέψουμε τις τιμές της μεταβλητής QuantitySold από αριθμητικές σε παραγοντικές.

```
>TrainingSet$QuantitySold=as.factor(TrainingSet$QuantitySold)

>ggplot(data=TrainingSet, aes(x=QuantitySold)) +
geom_bar(fill = c('green',"black")) +
geom_text(stat='count', aes(label=..count..), vjust=-1)+
ggtitle("Items Sold (Train)")+theme(plot.title = element_text(hjust
= 0.5))
>ggplot(data=TestSet, aes(x=QuantitySold)) +
geom_bar(fill = c('green',"black")) +
geom_text(stat='count', aes(label=..count..), vjust=-1)+
ggtitle("Items Sold (Test)")+theme(plot.title = element_text(hjust =
0.5))
```



Εικόνα 68: Ραβδογράμματα της μεταβλητής QuantitySold στα σύνολα δεδομένων TrainingSet και TestSet.

Αλγόριθμος: Απλοϊκός Bayes

Δημιουργία του μοντέλου απλοϊκού Bayes.

```
>library(e1071)
>start.time <- Sys.time()
>model=naiveBayes(QuantitySold~., data=TrainingSet)
>end.time<- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 7.46208 secs
```

Εικόνα 69: Χρόνος εκπαίδευσης του μοντέλου απλοϊκού Bayes.

```
>start.time <- Sys.time()
>pred <- predict(model, newdata=TestSet)
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 15.65549 secs
```

Εικόνα 70: Χρόνος ταξινόμησης του μοντέλου απλοϊκού Bayes.

```
>table(pred,TestSet$QuantitySold)
```

```
pred     0     1
   0 26884  4779
   1  1177  4620
```

Εικόνα 71: Αποτελέσματα προβλέψεων.

$$Ακρίβεια = \frac{26884 + 4620}{26884 + 1177 + 4779 + 4620} = 0.84$$

Παρατηρούμε ότι για τις τιμές της κλάσης 0 η ακρίβεια του μοντέλου είναι υψηλή, σε αντίθεση με τις τιμές της κλάσης 1 οι οποίες δεν πετυχαίνουν καλά αποτελέσματα . Αυτό συμβαίνει γιατί το μεγαλύτερο ποσοστό των τιμών ανήκουν στην κατηγορία 0 και αυτό έχει ως αποτέλεσμα ο αλγόριθμος να μπορεί να ξεχωρίζει τις τιμές της μιας κλάσης, αλλά όχι της άλλης. Άρα η ειδικότητα του μοντέλου είναι αρκετά χαμηλή.

$$Ανάκληση = \frac{26884}{26884 + 1177} = 0.95$$

$$Ειδικότητα = \frac{4620}{4779 + 4620} = 0.49$$

Αλγόριθμος: K-Nearest Neighbors

```
>library(caret)
>data=as.data.frame(TrainingSet[,2:11])
>knn_control <- trainControl(method = "none")
>start.time <- Sys.time()
>model=train(data,TrainingSet$QuantitySold,method="knn",trControl
=knn_control)
>end.time<- Sys.time()
>time.taken <- end.time - start.time
```

```
>time.taken
Time difference of 2.733455 secs
```

Εικόνα 72: Χρόνος εκπαίδευσης του μοντέλου k-nearest neighbors.

```
>start.time <- Sys.time()
>pred <- predict(model, newdata=TestSet)
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
Time difference of 2.675703 mins
```

Εικόνα 73: Χρόνος ταξινόμησης του μοντέλου k-nearest neighbors.

```
>table(pred,TestSet$QuantitySold)
```

pred	0	1
0	26678	3221
1	1383	6178

Εικόνα 74: Αποτελέσματα προβλέψεων.

$$Ακρίβεια = \frac{26678 + 6178}{26678 + 1383 + 3221 + 6178} = 0.87$$

$$Ανάκληση = \frac{26678}{26678 + 1383} = 0.95$$

$$Ειδικότητα = \frac{6178}{6178 + 3221} = 0.65$$

Παρατηρούμε ότι, ο αλγόριθμος του απλοϊκού Bayes και ο αλγόριθμος K-nearest neighbors πετυχαίνουν συγκρίσιμο ποσοστό ακρίβειας και χρόνο εκπάιδευσης. Επιπροσθέτως, ο αλγόριθμος ταξινομεί σε πολύ καλύτερα χρονικά πλαίσια. Ωστόσο, η ειδικότητα του είναι αρκετά πιο χαμηλή με αποτέλεσμα ο K-nearest neighbors να κατέχει σημαντικό πλεονέκτημα στο συγκεκριμένο πείραμα.

	Ακρίβεια	Ανάκληση	Ειδικότητα	Χρόνος εκπαίδευσης (sec)	Χρόνος ταξινόμησης (sec)
Απλοϊκός Bayes	84%	95%	49%	7.46	15.65
K-Nearest Neighbors	87%	95%	65%	2.73	160

Θα εξετάσουμε ξανά τις μετρικές των αλγορίθμων με την μέθοδο cross-validation.

Αρχικά θα ενώσουμε τα δυο σύνολα δεδομένων σε ένα με τις παρακάτω εντολές.

```
>WholeDataset=rbind(TrainingSet,TestSet)
>WholeDataset$QuantitySold=as.factor(WholeDataset$QuantitySold)
>set.seed(123)
>rows <- sample(nrow(WholeDataset))
>WholeDataset=WholeDataset[rows,]
```

Αλγόριθμος: Απλοϊκός Bayes (cross validation)

```
>naivekcross<-WholeDataset[sample(nrow(WholeDataset)),]
>folds <- cut(seq(1,nrow(naivekcross)),breaks=10,labels=FALSE)
>accuracy=list()
>recall=list()
>sensitivity=list()
>start.time <- Sys.time()
>for(k in 1:10){
     indexes <- which(folds==k,arr.ind=TRUE)
     test <- naivekcross[indexes, ]
     train <- naivekcross[-indexes, ]
     model=naiveBayes(QuantitySold~., data=train)
     pred <- predict(model, newdata=test)
     p=table(pred,test$QuantitySold)
     accuracy[[k]] <-(p[1]+p[4])/(p[1]+p[2]+p[3]+p[4])
     recall[[k]] <-(p[1])/(p[1]+p[2])
     sensitivity[[k]] <-(p[4])/(p[3]+p[4])
 }
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 3.467715 mins
```

Εικόνα 75: Χρόνος εκπαίδευσης και ταξινόμησης του αλγορίθμου απλοϊκού Bayes .

```
>print(cbind("fold",1:10,accuracy,recall,sensitivity))
```

```
           accuracy  recall    sensitivity
"fold" 1   0.8143219 0.9393895 0.5233172
"fold" 2   0.8160108 0.9411194 0.5289863
"fold" 3   0.8123966 0.9368574 0.5267668
"fold" 4   0.8148223 0.9380107 0.5293063
"fold" 5   0.8157068 0.9398907 0.5280081
"fold" 6   0.8172944 0.9397218 0.5325244
"fold" 7   0.8151263 0.9384925 0.52163
"fold" 8   0.8172268 0.9399662 0.5316854
"fold" 9   0.8158757 0.9374606 0.5358179
"fold" 10  0.813849  0.9379077 0.5257015
```

Εικόνα 76: Αποτελέσματα του αλγορίθμου για κάθε επανάληψη της μεθόδου k cross validation .

```
>print(cbind(rbind("Accuracy",c(mean(unlist(accuracy)))),rbind
("Recall",c(mean(unlist(recall)))),rbind("Sensitivity",c(mean(
unlist(sensitivity))))))
```

```
     [,1]                [,2]                [,3]
[1,] "Accuracy"          "Recall"            "Sensitivity"
[2,] "0.815263063498246" "0.938881669568032" "0.528374386373598"
```

Εικόνα 77: Μέση τιμή των μετρήσεων της ακρίβειας, της ανάκλησης και της ειδικότητα .

Αλγόριθμος: K-Nearest Neighbors (cross validation)

```
>knnkcross<-WholeDataset[sample(nrow(WholeDataset)),]
>folds <- cut(seq(1,nrow(knnkcross)),breaks=10,labels=FALSE)
>accuracy=list()
>recall=list()
>sensitivity=list()
>start.time <- Sys.time()
>for(k in 1:10){
     indexes <- which(folds==k,arr.ind=TRUE)
     test <- knnkcross[indexes, ]
     train <- knnkcross[-indexes, ]
     data=as.data.frame(train[,2:11])
     knn_control <- trainControl(method = "none")
     model=train(data,train$QuantitySold,method="knn",trControl
=knn_control)
     pred <- predict(model, newdata=test)
     p=table(pred,test$QuantitySold)
     accuracy[[k]] <-(p[1]+p[4])/(p[1]+p[2]+p[3]+p[4])
     recall[[k]] <-(p[1])/(p[1]+p[2])
     sensitivity[[k]] <-(p[4])/(p[3]+p[4])
 }
```

```
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 24.22124 mins
```

Εικόνα 78: Χρόνος εκπαίδευσης και ταξινόμησης του αλγορίθμου k-nearest neighbors .

```
>print(cbind("fold",1:10,accuracy,recall,sensitivity))
```

```
          accuracy  recall    sensitivity
"fold" 1  0.8735349 0.9452623 0.7063456
"fold" 2  0.866813  0.9403589 0.7010661
"fold" 3  0.8724202 0.9445599 0.7050247
"fold" 4  0.8704567 0.9417092 0.7032557
"fold" 5  0.8725891 0.9441639 0.7036995
"fold" 6  0.8720149 0.9445091 0.7052725
"fold" 7  0.8741724 0.9460817 0.7070514
"fold" 8  0.8741429 0.9449879 0.7107761
"fold" 9  0.875494  0.9440586 0.7151877
"fold" 10 0.8734335 0.9469415 0.7022042
```

Εικόνα 79: Αποτελέσματα του αλγορίθμου για κάθε επανάληψη της μεθόδου k cross validation .

```
>print(cbind(rbind("Accuracy",c(mean(unlist(accuracy)))),rbind
("Recall",c(mean(unlist(recall)))),rbind("Sensitivity",c(mean(
unlist(sensitivity))))))
```

```
[,1]                [,2]                [,3]
"Accuracy"          "Recall"            "Sensitivity"
"0.872507159699727" "0.94426331766737" "0.705988360467497"
```

Εικόνα 80: Μέση τιμή των μετρήσεων της ακρίβειας, της ανάκλησης και της ειδικότητα .

Η μέθοδος k-cross validation επίφερε τα ίδια αποτελέσματα με το προηγούμενο παράδειγμα καθώς ο αλγόριθμος του απλοϊκού Bayes δεν ξεχωρίζει αποδοτικά τις τιμές της μιας κλάσης.

	Ακρίβεια	Ανάκληση	Ειδικότητα	Χρόνος εκπαίδευσης & Ταξινόμησης (min)
Απλοϊκός Bayes	81.5%	93.8%	52.8%	3.46
K-Nearest Neighbors	87.2%	94.4%	70.5%	24.22

Πείραμα 5^{ο}

Σύνολο δεδομένων : SentimentAll [34]

Περιγραφή: Το σύνολο δεδομένων αφορά κριτικές ταινιών που έγραψαν οι χρήστες της ιστοσελίδας imbd.com. Οι κριτικές αυτές χωρίζονται σε θετικές, αρνητικές και απροσδιόριστες.

SentimentAll	
ΓΡΑΜΜΕΣ	100.000
ΣΤΗΛΕΣ	5
ΤΙΜΗ ΣΤΟΧΟΣ	Sentiment(κατηγορηματική)

Το αρχικό σύνολο δεδομένων θα μετασχηματιστεί στο υποσύνολο SentimentTrain, το οποίο θα περιέχει τις κριτικές που ανήκουν στη θετική ή στην αρνητική κατηγορία. Στη συνέχεια θα δημιουργηθεί ένα καινούργιο σύνολο δεδομένων με τη μέθοδο tf-idf. Στο καινούργιο σύνολο δεδομένων θα εφαρμόσουμε και θα συγκρίνουμε τις αποδόσεις των μηχανών διανυσμάτων υποστήριξης, του νευρωνικού δικτύου και του δέντρου απόφασης. Εξαιτίας του μεγάλου όγκου των δεδομένων θα χρησιμοποιήσουμε μόνο τη μέθοδο split percentage καθώς η μέθοδος k-cross validation θα απαιτούσε αρκετό χρόνο για να επιφέρει αποτελέσματα.

SentimentAll:

1) **Row -** Αριθμός εγγραφής
2) **Category** - Κατηγορία στην οποία θα χρησιμοποιηθεί η εγγραφή (train ή test)
3) **Text -** Κριτική Ταινίας
4) **Sentiment** - Το συναίσθημα της κριτικής. (Θετική κριτική , Αρνητική κριτική **,** Απροσδιόριστη)
5) **File -** Όνομα αρχείο που προήλθε η κριτική

Με τις παρακάτω εντολές παραβλέπονται όλες οι απροσδιόριστες εγγραφές. Έπειτα, ορίζουμε τις τιμές neg ως 0 και τις τιμές pos ως 1 και θέτουμε την μεταβλητή Sentime nt ως κατηγορηματική.

```
>SentimentAll=SentimentAll[SentimentAll$Sentiment!='unsup',]
>SentimentAll$Sentiment[SentimentAll$Sentiment=="neg"]=0
>SentimentAll$Sentiment[SentimentAll$Sentiment=="pos"]=1
>SentimentAll$Sentiment=as.factor(SentimentAll$Sentiment)
```

Στη συνέχεια, θα μετασχηματίσουμε όλα τα κεφαλαία γράμματα σε πεζά και θα φιλτράρουμε όλους τους χαρακτήρες που δεν είναι γράμματα. Αυτό γίνεται για να δημιουργηθεί αποδοτικότερο διάνυσμα. Η αφαίρεση των χαρακτήρων θα γίνει με τη βοήθεια της βιβλιοθήκης **stringgr**.

```
>SentimentAll$Text=tolower(SentimentAll$Text)
>library(stringr)
>SentimentAll$Text=str_replace_all(SentimentAll$Text, "[^a-zA-Z]", " 
")
```

Έπειτα θα «ανακατεύσουμε τις τιμές του συνόλου δεδομένων και θα δημιουργήσουμε το καινούργιο σύνολο δεδομένων.

```
>set.seed(123)
>rows=sample(nrow(SentimentAll))
>SentimentAll=SentimentAll[rows,]
>SentimentTrain=as.data.frame(SentimentAll[,3:4])
```

SentimentTrain	
ΓΡΑΜΜΕΣ	25.000
ΣΤΗΛΕΣ	2
ΤΙΜΗ ΣΤΟΧΟΣ	Sentiment(κατηγορηματική)

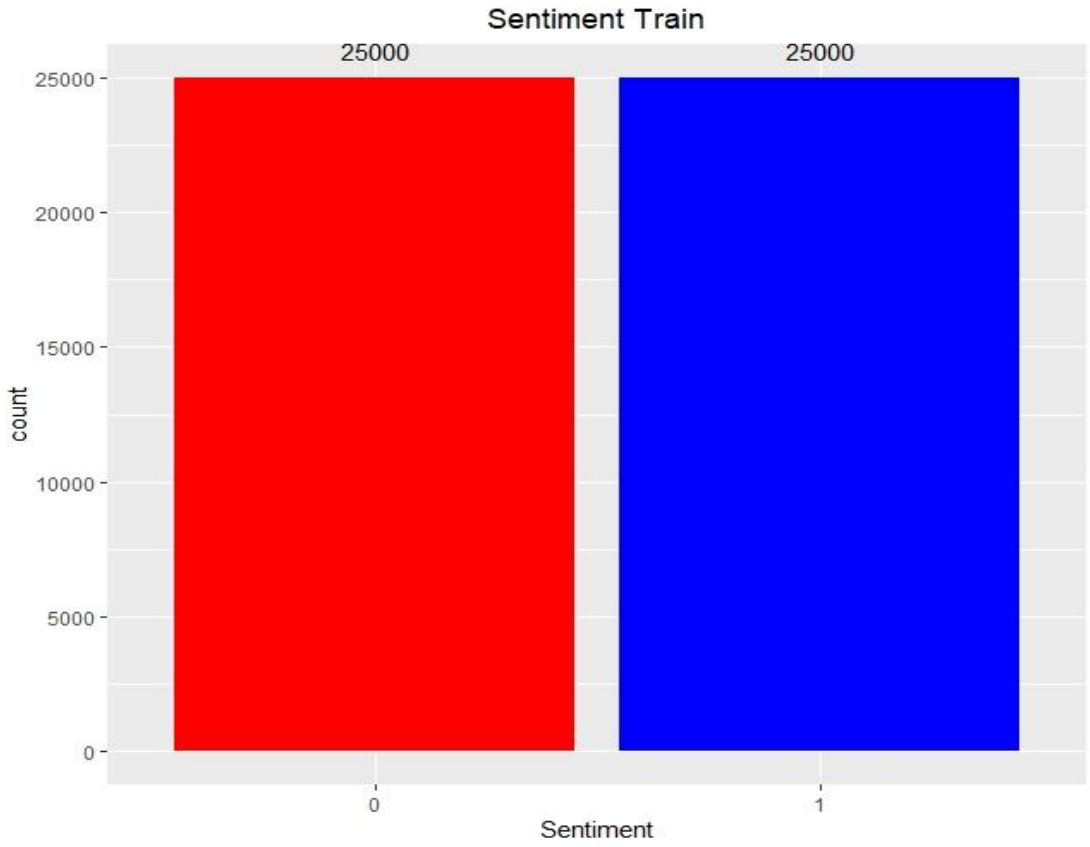
SentimentTrain :

1) **Text** - Κριτική Ταινίας
2) **Sentiment** - Το συναίσθημα της κριτικής. (Θετική κριτική , Αρνητική κριτική

Παρατηρούμε πως και στο σύνολο δεδομένων οι κατηγορίες είναι ισάριθμες.

```
>ggplot(data=SentimentTrain, aes(x=Sentiment)) +
geom_bar(fill = c('red','blue'))+
geom_text(stat='count',aes(label=..count..), vjust=-1)+
```

```
ggtitle("Sentiment    Train")+theme(plot.title=element_text(hjust    =
0.5))
```



Εικόνα 81: Ραβδόγραμμα της μεταβλητής Sentiment στο σύνολο δεδομένων SentimentTrain.

Πλέον μπορούμε να δημιουργήσουμε το διάνυσμα tf-idf. Λόγω του μεγάλου όγκου δεδομένων θα χρησιμοποιήσουμε το ¼ των εγγραφών για την εκπαίδευση και τον έλεγχο των αλγορίθμων. Θα κρατήσουμε τις 2.000 λέξεις με την μεγαλύτερη συχνότητα εμφάνισης στις πρώτες 12.500 εγγραφές. Έτσι θα δημιουργήσουμε έναν πίνακα 12.500 X 2.000 εγγραφών.

```
>sent=as.data.frame(SentimentTrain[1:12500,])
>library(tm)
>library(qdap)
>library(stringi)
>words=list()
>tf=list()
>tfs=list()
>for(i in 1:12500){
     w=rm_stopwords(sent$Text[i], tm::stopwords("english"))
```

```
     for(j in 1:length(unlist(w))){
         k=unlist(w)[j]
         if(!(k %in% names(tf))){
             tf[[k]]=1
         }
         else{
             tf[[k]]=tf[[k]]+1
         }
         if(!(k %in% names(words))){
             words[[k]]=1
         }
         else{
             if(tf[[k]]==1){
                 words[[k]]=words[[k]]+1
             }
         }
     }
     tfs[[i]]=tf
     tf=list()
 }
>while(length(words)>2000){
     temp=words
     words=words[words>min(unlist(words))]
 }
>while(length(words)<2000){

    g=temp[which.min(temp)]
     words[[names(g)]]=min(unlist(temp))
     temp[names(g)] <- NULL
 }
>tfidf <- data.frame(matrix(ncol = length(words), nrow = 0))
>colnames(tfidf) <-names(words)
>tf=list()
>for(i in 1:12500){
     for(v in 1:length(words)){
         if(names(words[v]) %in% names(unlist(tfs[[i]]))){
             tf=unlist(tfs[[i]])
             t=tf[[names(words[v])]]/length(tf)
             idf=log2(12500/words[[v]])
             tfidf[i,names(words[v])]=t*idf
         }
         else{
             tfidf[i,names(words[v])]=0
         }
         tf=list()
     }
 }
```

```
>Dataset=as.data.frame(tfidf)
>Dataset$Sentiment=SentimentTrain$Sentiment[1:12500]
```

Από την πειραματική μελέτη παρατηρήθηκε πως το σύνολο δεδομένων περιέχει τη μεταβλητή “else” η οποία είναι δεσμευμένη λέξη στην R οπότε θα αφαιρεθεί.

```
>colnames(Dataset[grep('else', colnames(Dataset))])
```

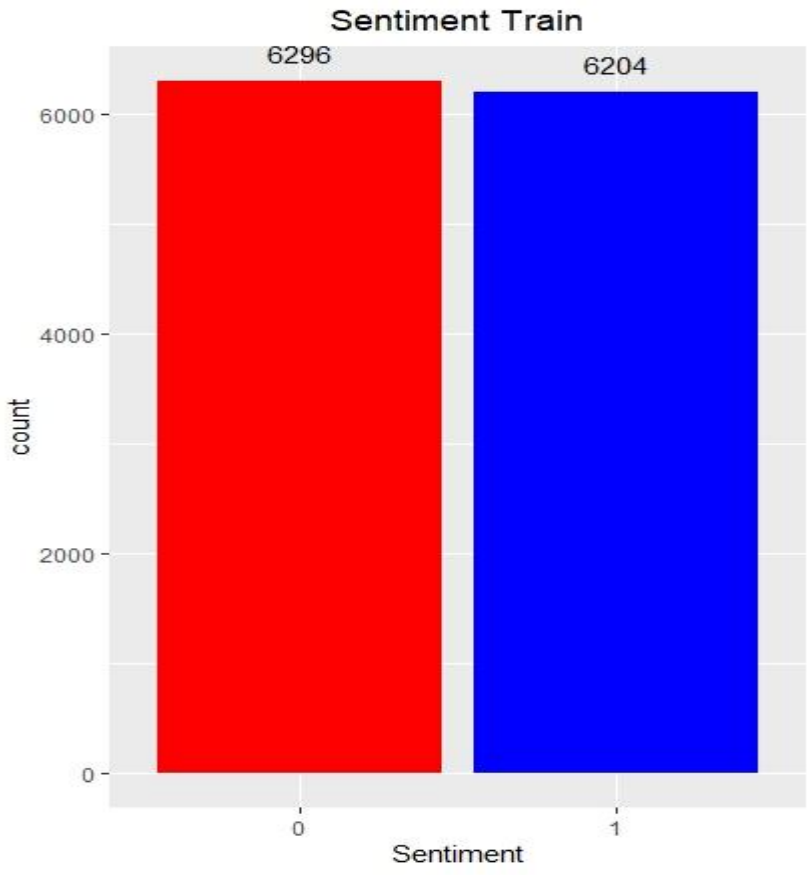
```
[1] "else"
```

Εικόνα 82: Η στήλη της δεσμευμένης λέξης else.

```
>Dataset[,grep('else', colnames(Dataset))]=NULL
```

Στις πρώτες 12.500 εγγραφές οι κατηγορίες χωρίζονται ως εξής:

```
>ggplot(data=SentimentTrain[1:12500,], aes(x=Sentiment)) +
geom_bar(fill = c('red','blue')) +
geom_text(stat='count', aes(label=..count..), vjust=-1)+
ggtitle("Sentiment      Train")+theme(plot.title=element_text(hjust=
0.5))
```



Εικόνα 83: Ραβδόγραμμα της μεταβλητής Sentiment στο σύνολο δεδομένων Dataset.

Αλγόριθμος: Μηχανή Διανυσμάτων υποστήριξης (Split percentage)

Θα διαχωρίσουμε το σύνολο δεδομένων κατά 88% σε σύνολο εκπαίδευσης και κατά 12% σε σύνολο ελέγχου, Έπειτα θα εκπαιδεύσουμε το μοντέλο μας.

```
>library(caTools)
>set.seed(123)
>split=sample.split(Dataset$Sentiment, SplitRatio = 0.88)
>Train=subset(Dataset,split == TRUE)
>Test=subset(Dataset,split == FALSE)
>library(e1071)
>start.time <- Sys.time()
>svm<-svm(Sentiment~.,data=Train,method="C-
classification",kernal="radial",gamma=0.1,cost=10,scale=FALSE)
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 3.097016 mins
```

Εικόνα 84: Χρόνος εκπαίδευσης του μοντέλου svm.

```
>start.time <- Sys.time()
>pred=predict(svm,Test)
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 5.17367 secs
```

Εικόνα 85: Χρόνος ταξινόμησης του μοντέλου svm.

```
>table(pred,Test$Sentiment)
```

```
pred   0   1
   0 642  89
   1 117 652
```

Εικόνα 86: Αποτελέσματα προβλέψεων.

$$Ακρίβεια = \frac{642 + 652}{642 + 89 + 117 + 652} = 0.862$$

Αλγόριθμος: Δέντρο απόφασης(Split percentage)

Με το ίδιο σύνολο εκπαίδευσης που χρησιμοποιήθηκε στο προηγούμενο παράδειγμα θα εκπαιδεύσουμε το μοντέλο του δέντρου απόφασης.

```
>library(rpart)
>start.time <- Sys.time()
>tree=rpart(Sentiment~., data = Train, method = 'class')
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 1.34334 mins
```

Εικόνα 87: Χρόνος εκπαίδευσης του μοντέλου δένδρου απόφασης.

```
>start.time <- Sys.time()
>pred=predict(tree,Test,type="class")
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 0.3684368 secs
```

Εικόνα 88: Χρόνος ταξινόμησης του μοντέλου δένδρου απόφασης.

```
>table(as.matrix(pred),Test$Sentiment)
```

```
   0   1
0 488 132
1 271 609
```

Εικόνα 89: Αποτελέσματα προβλέψεων.

$$Ακρίβεια = \frac{488 + 609}{488 + 132 + 271 + 609} = 0.731$$

Αλγόριθμος: Νευρωνικό δίκτυο (Split percentage)

Αρχικά θα δημιουργήσουμε δύο επιπλέον στήλες σύμφωνα με τις τιμές των κλάσεων με τη βιβλιοθήκη **neuralent** και **nnet.**. Για τη στήλη Sentiment1 αν η τιμή του χαρακτηριστικού Sentiment είναι 0 τότε θα εκχωρηθεί η τιμή 1, διαφορετικά θα εκχωρηθεί η τιμή 0. Αντίστοιχα, για το Sentiment2 θα εκχωρηθεί 1 αν η τιμή του χαρακτηριστικού Sentiment είναι 1 και 0 σε αντίθετη περίπτωση. Έπειτα θα εκπαιδεύσουμε το νευρωνικό μας δίκτυο.

```
>library(neuralnet)
>library(nnet)
>Sentiment=Dataset$Sentiment
>Dataset<-
cbind(Dataset[,1:1999],class.ind(as.factor(Dataset$Sentiment)))
>names(Dataset)<-c(names(Dataset)[1:1999],"Sentiment1","Sentiment2")
>Dataset=cbind(Dataset,Sentiment)
>library(caTools)
>set.seed(123)
>split=sample.split(Dataset$Sentiment, SplitRatio = 0.88)
>Train2=subset(Dataset[,1:2001],split == TRUE)
>Test2=subset(Dataset,split == FALSE)
>n=colnames(Train2)
>n=n[!n %in% "Sentiment1"]
>n=n[!n %in% "Sentiment2"]
>f=as.formula(paste("Sentiment1+Sentiment2 ~", paste(n, collapse = "
+ ")))
>start.time <- Sys.time()
>neural=neuralnet(f,data=Train2,hidden=8,
act.fct="logistic",linear.output = FALSE)
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 1.715061 mins
```

Εικόνα 90: Χρόνος εκπαίδευσης του μοντέλου νευρωνικού δικτύου.

```
>start.time <- Sys.time()
>pred <- predict(neural, Test2)
>end.time <- Sys.time()
>time.taken <- end.time - start.time
>time.taken
```

```
Time difference of 0.38764 secs
```

Εικόνα 91: Χρόνος ταξινόμησης του μοντέλου νευρωνικού δικτύου.

```
>table(as.matrix(apply(pred, 1, which.max)),Test2$Sentiment)
```

```
     0   1
1  622 100
2  137 641
```

Εικόνα 92: Αποτελέσματα προβλέψεων.

$$Ακρίβεια = \frac{622 + 641}{622 + 100 + 137 + 641} = 0.842$$

	Ακρίβεια	Χρόνος εκπαίδευσης (min)	Χρόνος ταξινόμησης (sec)
Μηχανή διανυσμάτων υποστήριξης	86.2%	3.09	5.17
Δέντρο απόφασης	73.1%	1.34	0.36
Νευρωνικό Δίκτυο	84.2%	1.71	0.38

Παρατηρούμε ότι από τους τρεις αλγορίθμους το μοντέλο της μηχανής διανυσμάτων υποστήριξης κατάφερε νε πετύχει την καλύτερη ακρίβεια, αλλά απαιτείται αρκετός χρόνος για να εκπαιδευτεί και να ταξινομήσει καινούργιες εγγραφές. Το νευρωνικό δίκτυο κατάφερε να πετύχει και αυτό υψηλή απόδοση σε πολύ καλύτερα χρονικά πλαίσια.

Κεφάλαιο 4^{ο} : Συμπεράσματα πειραματικής μελέτης και προτάσεις για μελλοντική έρευνα

4.1 Συμπεράσματα

Από την παραπάνω πειραματική μελέτη παρατηρούμε τις δυνατότητες των αλγορίθμων στο να εντοπίζουν συσχετίσεις και να μπορούν να κατηγοριοποιούν σε όλα τα σύνολα δεδομένων που τους δόθηκαν. Ακόμα και στο 5^{ο} πείραμα που το τελικό σύνολο δεδομένων αποτελούταν από 2.000 στήλες και 12.500 γραμμές, με την κατάλληλη προ-επεξεργασία όλοι οι αλγόριθμοι κατάφεραν να πετύχουν αποδεκτές αποδόσεις (>70%) σε καλά χρονικά πλαίσια. Επιπροσθέτως, παρατηρούμε την αναγκαιότητα ύπαρξης διαφορετικών αλγορίθμων κατηγοριοποίησης και παλινδρόμησης καθώς οι αποδόσεις των αλγορίθμων εξαρτώνται άμεσα από τη φύση των χαρακτηριστικών των συνόλων δεδομένων. Επιπλέον, γίνεται αντιληπτό το πως η σωστή επεξεργασία των δεδομένων και η κατάλληλη επιλογή μεθόδων μπορούν να επιφέρουν υψηλές τιμές ακρίβειας σε μικρά χρονικά πλαίσια. Τέλος, παρατηρώντας των κώδικα που γράφτηκε για την διεξαγωγή των πειραμάτων, παρουσιάζεται ένα κομμάτι των δυνατοτήτων της R και γίνεται ξεκάθαρος λόγος που χρησιμοποιείται ευρέως σε πειραματικές μελέτες που αφορούν τον τομέα της μηχανικής μάθησης.

4.2 Προτάσεις για μελλοντική έρευνα

Η πειραματική μελέτη που προήλθε, θα μπορούσε να αναπτυχθεί συγκρίνοντας περισσότερες μεθόδους μηχανικής μάθησης ή βελτιώνοντας τις ήδη υπάρχοντες . Επίσης, με την χρήση ειδικών γραφημάτων που διαθέτουν οι βιβλιοθήκες της R θα μπορούσαν να μελετηθούν καλύτερα οι συσχετίσεις των μεταβλητών με την τιμή στόχο, βελτιώνοντας έτσι την απόδοση των μοντέλων. Τέλος, με τους κατάλληλους πόρους θα ήταν δυνατή η μελέτη μεγαλύτερων συνόλων δεδομένων, π.χ. του αρχικού συνόλου δεδομένων του 5^{ου} πειράματος , και ως συνέπεια θα είχαμε πιο βελτιωμένα μοντέλα και καλύτερες αποδόσεις.

Αναφορές

[1] Mapoka, K., Masebu, H., and Zuva, T. (2013). Mathematical Models and Algorithms Challenges. *International Journal of Control Theory and Computer Modeling*, 3 (6), pp. 21-28.

[2] R. Project. Contributors. Διαθέσιμο στο link: https://www.r-project.org/contributors.html. Προσπελάστηκε στις: 28/05/2020.

[3] R Project. What is R? Διαθέσιμο στο link: https://www.r-project.org/about.html.

[4] Agrawal, V. (2016).Applications Of R Programming In R-eal World. *eLearning Industry*. Διαθέσιμο στο link: https://elearningindustry.com/applications-r-programming-r-eal-world. Προσπελάστηκε στις: 22/05/2020.

[5] Krill, P. (2015).Why R? The pros and cons of the R language. *InfoWorld*. Διαθέσιμο στο link:https://www.infoworld.com/article/2940864/r-programming-language-statistical-data-analysis.html. Προσπελάστηκε στις: 10/04/2020.

[6] Βλαχάβας, Ι., Κεφαλάς, Π., Βασιλειάδης, Ν., Κόκκορας, Φ. και Σακελλαρίου, Η. (2011). *Τεχνητή Νοημοσύνη*, 3η Έκδοση. Θεσσαλονίκη: Εκδόσεις Πανεπιστημίου Μακεδονίας.

[7] Figueiredo, M.A.T. (2003). Adaptive sparseness for supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25 (9), pp. 1150 – 1159.

[8] Bouveyron, C., and Jacques, J. (2010). Adaptive Linear Models for Regression: improving prediction when population has changed. *Pattern Recognition Letters*, 31 (14), pp. 2237-2247.

[9] Felicísimo, A.M., Cuartero, A., Remondo, J., andQuirós, E. (2013).Mapping landslide susceptibility with logistic regression, multiple adaptive regression splines, classification and regression trees, and maximum entropy methods: a comparative study. *Landslides*, 10, pp. 175-189.

[10] Σκούρα, Α. Κατηγοριοποίηση. *Πανεπιστήμιο Πατρών*. Διαθέσιμο στο link: https://www.ceid.upatras.gr/webpages/courses/cplusplus/dm/2_Classification1.pdf. Προσπελάστηκε στις: 10/04/2020.

[11] Γεωργούλη, Κ. (2015). *Τεχνητή Νοημοσύνη – Μία Εισαγωγική Προσέγγιση*. Ζωγράφου: ΣΕΑΒ.

[12] Χαμάλης, Θ. (2015). *Μέθοδοι Ομαδοποίησης Βασισμένες σε Στατιστικό Έλεγχο της Μονοτροπικότητας των Δεδομένων*. Μεταπτυχιακή Εργασία Εξειδίκευσης. Ιωάννινα: Πανεπιστήμιο Ιωαννίνων.

[13] Αντωνίου, Α. Α. (2016). *Ανασκόπηση Τεχνικών Ομαδοποίησης για την Εξόρυξη Δεδομένων από το Διαδίκτυο των Πραγμάτων*. Διπλωματική Εργασία. Αθήνα: Εθνικό Μετσόβιο Πολυτεχνείο.

[14] Συμεωνίδης, Π., και Γούναρης, Α. (2015). *Βάσεις, αποθήκες και εξόρυξη δεδομένων με τον SQL Server*. Εργαστηριακός Οδηγός. Θεσσαλονίκη: Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης.

[15] Σκιπητάρης, Δ., και Σακιώτης, Α. (2019). *Μοντελοποίηση αλγορίθμων συσταδοποίησης και εφαρμογές στη μηχανική μάθηση*. Πτυχιακή Εργασία. Αντίρριο: ΤΕΙ Δυτικής Ελλάδος.

[16] Chen, P., and Kurland, J. (2018). Time, Place, and Modus Operandi: A Simple Apriori Algorithm Experiment for Crime Pattern Detection. *2018 9th International Conference on Information, Intelligence, Systems and Applications (IISA)*, Zakynthos, Greece, 2018, pp. 1-3.

[17] Wagstaff, K., Cardie, C., Rogers, S., and Schroedl, S. (2001). *Proceedings of the Eighteenth International Conference on Machine Learning*, pp. 577-584.

[18] Quinlan, J.R. (1986). Induction of Decision Trees. *Machine Learning*, 1, (1), pp. 81-106.

[19] Quinlan, J.R. (1993). *C4.5: programs for machine learning*, Burlington, MA: Morgan Kaufmann Publishers Inc.

[20] Brill, E.D. (1992). A simple rule-based part-of-speech tagger. *Proceedings of the third Conference on Applied Natural Language Processing (ANLP'92)*, Trento, Italy.

[21] Brill, E.D. (1993). *A Corpus-based Approach to Language Learning*. Philadelphia: IRCS Technical Reports Series.

[22] Brill, E.D. (1995). Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part of Speech Tagging. *Computational Linguistics*, 21 (4), pp. 543—566.

[23] Brill, E.D. (1995). Unsupervised Learning of Disambiguation Rules for Part of Speech Tagging. *Proceedings of the 3rd Workshop on Natural Language Processing Using Very Large Corpora*, Massachusetts, USA.

[24] John, G.H., and Langley, P. (1995). Estimating Continuous Distributions in Bayesian Classifiers. *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence.*

[25] Aha, D.W., Kibler, D., and Albert, M.K. (1991). Instance-based learning algorithms. *Machine Learning*, 6, pp. 37-66.

[26] Rabiner, L.R., and Juang, B.H. (1986). An introduction to hidden Markov models. *IEEE ASSP Magazine*, pp. 4-15.

[27] Domingos, P., and Pazzani, M. (1997). On the Optimality of the Simple Bayesian Classifier under Zero-One Loss. *Machine Learning*, 29, pp. 103–130.

[28] Σταμπουλής, Θ. (2019). *Επιτάχυνση του αλγορίθμου μηχανικής μάθησης k-nearest neighbors με τη χρήση FPGA*. Διπλωματική Εργασία. Θεσσαλονίκη: Πανεπιστήμιο Μακεδονίας.

[29]Taggart, A.J., DeSimone, A.M., Shih, J.S,Filloux, M.E., and Fairbrother, W.G. (2012). Large-scale mapping of branch points in human pre-mRNA transcripts in vivo. *Nature Structural & Molecular Biology*, 19 (7), pp. 719–721.

[30] Johnson W (1996, Vol 4). Fitting percentage of body fat to simple body measurements. Journal of Statistics Education.

Dataset retrieved from: http://lib.stat.cmu.edu/datasets/bodyfat

[31] Rajen Bhatt, Abhinav Dhall, 'Skin Segmentation Dataset', UCI Machine Learning Repository Machine Learning by Andrew Ng at Coursera

Dataset retrieved from: https://archive.ics.uci.edu/ml/datasets/Skin+Segmentation

[32]Forina, M. et al, PARVUS - An Extendible Package for Data Exploration, Classification and Correlation. Institute of Pharmaceutical and Food Analysis and Technologies, Via Brigata Salerno, 16147 Genoa, Italy.

Dataset retrieved from: http://faculty.smu.edu/tfomby/eco5385_eco6380/Cases/Wine.xls

[33] Grossman, J. (2013). Predicting-ebay-auction-sales-with-machine-learning. Working Paper, Published on Web.Research http://jaygrossman.com/post/2013/06/10/Predicting-eBay-Auction-Sales-with-Machine-Learning.aspx

Datasets retrieved from: https://cims.nyu.edu/~munoz/data/

[34] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. (2011). Learning Word Vectors for Sentiment Analysis. The 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011)

Dataset retrieved from: https://www.kaggle.com/uttam94/imdb-mastercsv